

# The Friday Afternoon Project: A Two-Hour VoodooIO Prototyping Exercise

Nicolas Villar, Hans Gellersen

Computing Department, Lancaster University  
Infolab21, South Drive, Lancaster LA1 4WA, U.K.  
{villar,hwg}@comp.lancs.ac.uk

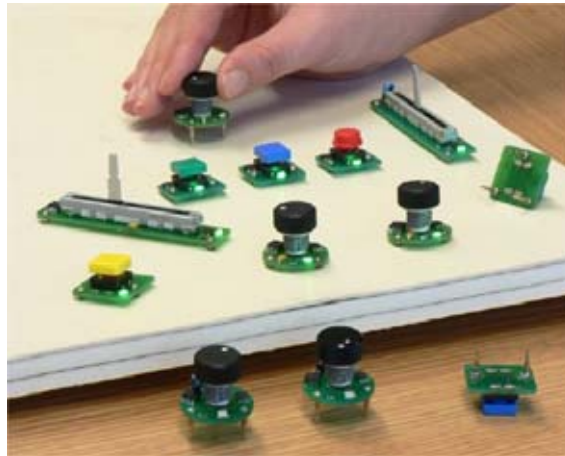
**Abstract:** VoodooIO is a new architecture for physical user interfaces, providing interface devices that can be flexibly arranged on an interactive substrate. The architecture supports rapid assembly of physical interfaces and dynamic re-arrangement of devices. In this paper we provide an account of a case study in which VoodooIO was used to rapidly build and explore a physical interface to an existing application, Google Earth. The case study gives insight into VoodooIO's support for rapid prototyping of physical user interface applications.

## 1 Introduction

Rapid prototyping of interfaces is an important step to test design ideas very early in the process. For GUI applications a plethora of tools are available that let designers rapidly construct and modify interfaces in order to explore design alternatives. However as computing moves from screen-based toward more tangible interaction, it becomes a challenge to provide comparable support for rapid prototyping of physical user interface applications.

VoodooIO is a novel architecture for physical user interfaces that we develop with the aim of making physical interfaces more malleable, in the sense that they can be rapidly assembled and re-configured. The architecture is based on physical controls that can be flexibly arranged on an interface substrate, and has been implemented based on the Pin&Play concept of networking devices through surfaces with embedded conductive layers [VSG02]. Figure 1 shows a selection of devices and the substrate material. The set of interface devices is comparable to the components provided by the Phidgets toolkit [GF01] but it is a distinct property of VoodooIO that the devices can be very easily inserted and removed from the interface using sharp coaxial pin connectors.

In previous work we have shown how VoodooIO facilitates interface adaptation at runtime, for example in interfaces to music software [VLG05] and games [Vi06]. In this paper we focus on the use of VoodooIO for rapid prototyping, and provide an account of a case study in which VoodooIO was used to build a physical interface to the Google Earth application. This prototyping exercise highlights two aspect of VoodooIO: its support for rapid construction of a physical interface to existing software; and the flexibility it provides for exploration of different physical interface configurations.



**Fig. 1.** VoodooIO provides devices that can be flexibly arranged on an interactive substrate.

## **2 A Physical Interface for Google Earth**

Google Earth allows a user to browse a collection of satellite imagery spanning the whole surface of the globe, making it possible to point and zoom to view any place on the planet (Fig.2, left). The resolution of some of the imagery is astounding, and after initial exploration on a desktop computer we sketched ideas for a larger-scale interface using physical controls in conjunction with a wall-mounted projection surface. After playing with Google Earth for a while, we found the most interesting actions to be zooming and panning (others include rotating the image, jumping to a specific place, and adding place-markers) – and decided to provide a VoodooIO interface to control these parameters.

Below is a sketch of the interface idea (Fig 2, right). The Google Earth screen (without graphical controls) is projected onto a wall-mounted VoodooIO substrate. Four VoodooIO buttons allow the user to pan over the image, and are intended to work like the arrow keys on the keyboard. Notice how the button at the top actually makes the image scroll downwards, or the button at the left makes the image scroll to the right. This is the same effect that would result from using the arrow keys - much like pressing the down-arrow in Word causes the text to scroll upwards - and is a convention of graphical user interfaces. The fourth control is a rotary knob that controls the zoom level of the image.



Fig. 2. Google Earth on the desktop (left) and our design sketch for a physical interface (right).

### 3 Prototype Implementation

A trick that we have often used to interface VoodooIO control events to an existing program is to write a small application that generates keystrokes for the target application in response to VoodooIO interaction events. For this exercise we picked four VoodooIO buttons, and for each one, mapped their *Pressed / Not Pressed* states to simulate *Key Up / Key Down* events of a virtual keyboard. Below is an extract of the event handler that is executed whenever a VoodooIO button generates an interaction event.

```

if (buttonNode.GetID().Equals("378F8"))
{
    if (buttonNode.IsPressed())
    {
        keybd_event(KEY_LEFT_ARROW, 0x45, KEYEVENT_KEYDOWN, 0);
    }
    else
    {
        keybd_event(KEY_LEFT_ARROW, 0x45, KEYEVENT_KEYUP, 0);
    }
}

```

By default the arrow keys can be used to pan across the Google Earth visualization, so the application needed no modification. Mapping a VoodooIO knob to the Zoom In/Out function was only slightly more involved. Rotating the knob to the right causes a virtual '+' key to be pressed, which causes Google Earth to zoom in. Rotating it to the left generates a virtual '-' key down event, and rotating the knob to a middle position generates a '+' and '-' key-up events (equivalent of releasing the physical keys) and bringing the zooming to a halt.

As an afterthought we also picked up two Slider devices, and using the same three-state mapping as the zoom knob (back-stop-forward) we mapped one to the Left/Right arrow key events and the other to the Up/Down arrows. Sliding one slider to one side left causes the image to scroll continuously to the left, slide it to the other end and the image begins to pan to the right.

This was quickly implemented and tested on a VoodooIO tablet (Fig 3). The complete implementation and debugging time was about an hour.



**Fig. 3.** Desk prototype using a VoodooIO tablet.

#### **4 Deployment of the Prototype**

Once everything was working we copied the VoodooIO mapping program onto a memory flash drive, grabbed the buttons, sliders and dial and headed down to the XS lab. We loaded the program onto a machine in the lab, which is connected to a large wall-mounted VoodooIO notice-board and to a steerable projector. We downloaded and installed Google Earth, steered its image onto the board, ran the VoodooIO application and things began to work (Fig. 4). The setup of the system, from the time of going down to the lab, took about thirty minutes to get everything in place and working.

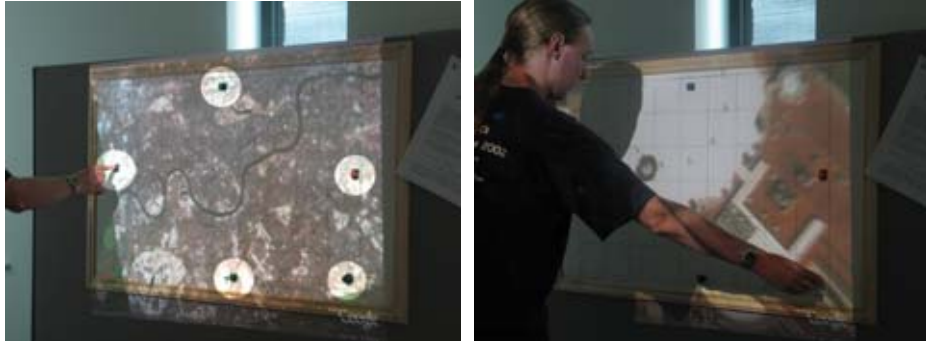


**Fig. 4.** Dave tests the system after deployment in the lab.

## 5 Testing and Exploration

We had originally arranged the four “arrow” buttons on the board as in the original concept sketch. As Dave, a colleague in the lab, tried it, he very quickly mentioned that the mapping felt wrong, and we could understand what he meant: pressing the button on the right caused the image to scroll to the left, as it had done sensibly on the desktop computer screen – but now, in the context of a larger and more immersive display it felt wrong. We physically switched the left and right buttons around, so that pressing on the right-hand one caused the image to scroll to the right. Note that no software remapping of button-to-action took place, we simply rearranged the physical arrangement of the controls to achieve this in a few seconds. To our surprise, the new mapping felt much more natural, so we swapped the up and down buttons around as well. Is this a result of the interaction being situated over the visualization, rather than separate from it as in a keyboard/screen situation? – We had not intended to explore this particular effect but the flexible arrangement of controls as supported by VoodooIO will make it easy to study different user preferences.

During this time Martyn, another colleague was carrying out some ultrasonic data collection in the XS lab, and while waiting for the process to finish he became an unsuspecting user-tester of the interface. The buttons remained in their place, and the zoom dial was attached to the lower-right hand corner of the board, as in the original sketch (Fig. 5, left).



**Fig. 5.** The layout of controls as originally sketched (left) and interface occlusion (right)

He was able to pan and zoom in and out using the controls, although the interaction seemed slightly uncomfortable at times. For example he was standing to the left of the board most of the time as to not occlude the projected image, except when reaching for the zoom dial (Fig 5, right). Also, he is left-handed so manipulating this control with his right was not ideal. So he unfastened the zoom dial from its original position and attached it instead to the top left corner of the board (Fig 6, left). Problems solved.



**Fig. 6.** The zoom dial quickly repositioned (left) and trying out a slider for panning (right)

We then suggested that instead of using the buttons to pan he try one of the sliders that I had also brought down with me. He attached the vertical-scroll slider alongside the left side of the board, underneath the zoom control (Fig 6, right). He tried scrolling with it, found that the mapping was “reversed” (I didn’t catch whether it was the same one that Dave and I found unnatural, but which is natural in GUI window scroll bars) so he simply unfastened the slider, turned it 180 degrees, and attached it again.

An added advantage of using the sliders over the buttons was the fact that you could set the map to drift in a particular direction without having to keep a button constantly depressed, which allowed one to step back and appreciate the effect of flying over the landscape. Martyn then added the second (horizontal) scroll slider, oriented underneath and perpendicular to the first one (Fig 7, left). Finally the now redundant buttons were removed to clear up the display area – they weren't being used anyway now that the sliders appeared to be a better way of panning around (Fig 7, right).



**Fig. 7.** Adding a second scroll slider for panning (left) and removing the buttons (right)

## 6 Conclusion

This prototyping exercise demonstrates how VoodooIO can contribute to fast development of application ideas for physical interfaces. The result is not bad for the two hours that it took to put together and test the original concept! But more importantly this exercise highlights the flexibility that VoodooIO provides to designers and users: it allows exploration of interface configurations in very fluid manner – devices are just added, removed, or repositioned in order to modify the user experience without need for reprogramming, recompilation, or stopping and restarting of the application.

Notice how much the interface in Fig. 7 differs from the sketch in Fig. 2. We believe that the end result wasn't necessarily better than the original design but it is safe to say that it was better suited to Martyn's preferences for many reasons. One of them being the fact that he is left handed and his arrangement better suits his ergonomic preference. The ability to try, compare and exchange different types of control on-the-fly also yielded interesting results. This is only made possible because the physical aspects of the architecture are just as flexible as the software that it is supported by and integrated with.

With the flexibility provided by VoodooIO, it begins to approach the idea of a "blank piece of paper" that users and interaction designers can be presented with to sketch out their own physical interfaces.

## References

- [GF01] S. Greenberg and C. Fitchett. Phidgets: easy development of physical interfaces through physical widgets. Proc. UIST '01: User Interface Software and Technology, ACM Press (2001), 209–218.
- [VSG02] Van Laerhoven, K.; Schmidt A.; Gellersen, H.: Pin&Play: Networking Objects through Pins". Proc. of 4<sup>th</sup> Intl. Conf. on Ubiquitous Computing (UbiComp 2002), Gothenburg, Sweden, LNCS No. 2498, Springer Verlag, pp.219 - 229.
- [VLG05] Villar, N.; Lindsay, A.; Gellersen, H.: Pin&Play&Perform: A Rearrangeable Tangible Interface for Musical Composition and Performance Proc. NIME '05 (New Interfaces for Musical Expression), Vancouver, 2005.
- [Vi06] Villar, N. et al.: VoodooIO Gaming Kit: A real-time adaptable gaming controller. Proc. of ACM International Conference on Advances in Computer Entertainment Technology (ACE 2006), Hollywood, June 2006, ACM Press.