Human-Display Interaction Technology

Emerging remote interfaces for pervasive display environments

Andrea Bellucci, Alessio Malizia[1], Paloma Diaz and Ignacio Aedo
Universidad Carlos III de Madrid, Laboratorio DEI. Madrid, Spain.
abellucc@inf.uc3m.es, amalizia@inf.uc3m.es, pdp@inf.uc3m.es, aedo@ia.uc3m.es

1. Introduction

We are living in a world where information processing is not only confined in desktop computers, but is being integrated into everyday objects and activities. Pervasive computation is human-centered: it permeates our physical world, helping us to achieve our goals and fulfill our needs with a minimum effort, exploiting natural interaction styles.

In such an environment, "*every object is confined in space by its surface. Surfaces are pervasive and play a predominant role in human perception of the environment.*"[2]. Therefore, as "*surfaces dominate the physical world*", the use of pervasive screen displays, LCD or based on front and rear projection, has gained interest in the last years and several prototypes have been developed for exploring their potential applications. In particular, we are assisting in a widespread development of new emerging interaction technologies allowing researchers to experiment with multi-modal remote interaction methods in the effort of providing a more intuitive human-display interaction.

Remote interaction with screen displays requires a sensor-based multi-modal touchless approach. By allowing user to employ hand gestures, this paradigm removes the constrains related to physical contact interactions so to permit natural interaction with digital information, made tangible in our real world. In fact, touchless interaction can be multimodal: in this case the interaction events are generated exploiting different human senses (visual, auditory and olfactory).

2. Emerging human-display remote interfaces

We provide here a list of emerging technologies allowing remote interaction with screen displays (see Table 1).

[1] Corresponding Author
[2] S. Borkowski, J. Letessier, J. Crowley. "*Spatial Control of Interactive Surfaces in an Augmented Environment*", in "European Conference on Human Computer Interaction, EHCI 04", July 2004.

Table 1.

A BASIC REFERENCE LIST FOR TOUCHLESS INTERACTION DEVICES AND LIBRARIES.

DEVICE	LIBRARY	*SCOPE*	*FEATURES*	REFERENCE
WIIMOTE, VR GLOVES	GLOVEPIE	*Supporting basic interactions with Nintendo's Wii Remote Controller.*	*Provides a Scripting Language.*	http://carl.kenner.googlepages.com/glovepie
WIIMOTE	WIIMOTELIB(CODEPLEX)	*Supporting basic and advanced interactions with Wiimote.*	*Can be combined with C# and DirectX.*	http://Wiimotelib.codeplex.com/Release/ProjectReleases.aspx?ReleaseId=21997
HAND GESTURES AND WEBCAM	MICROSOFT TOUCHLESS	*Natural interaction experiences using only a webcam*	*Webcam multi-"touch" object tracking SDK using color markers*	http://www.codeplex.com/touchless
HAND GESTURES AND CAMERA	SIXTHSENSE	*Hand gesture recognition using fiducial markers and a camera*	*Hand gestures act as interaction instructions for projected application interfaces*	http://www.pranavmistry.com/projects/sixthsense/
HUMAN BODY AND WEBCAM	MICROSOFT PROJECT NATAL	*Natural user interface using gestures, spoken commands or presented objects and images*	*SDK not yet available for developers.*	http://www.xbox.com/en-US/live/projectnatal/
HAND GESTURES AND ELECTRIC FIELD	POINTSCREEN	*Providing touchless interaction using Electric Field proximity sensors*	*Proprietary SDK*	http://www.iais.fraunhofer.de/600.html
EYES AND GLASS-MOUNTED CAMERA	EYEWRITER	*Tracking eye movements*	*Provides an eye-tracking and a drawing software designed for drawing with eye movements*	http://www.eyewriter.org/

Microsoft Touchless (http://www.officelabs.com/projects/touchless/Pages/default.aspx) is an open-source Office Labs Grassroots prototype (currently available for download from the Office Labs website) which goal is to create a multi-touch remote environment by simply using a webcam. Touchless Software Development Kit (SDK) enables developers to create multi-touch based applications (but without touching the screen) using a webcam and color markers (visual tracking fiducial markers). Following the same rationale Pranav Mistry (a PhD student of the Fluid Interface Group at MIT Media Lab) developed SixthSense (http://www.pranavmistry.com/projects/sixthsense/), a wearable gestural interface that exploits natural hand gesture interactions with digital information displayed in the tangible world. At its very core, the prototype is a pocket projector, a mirror and a camera. The projector turns tangible surfaces, walls and physical objects into screen displays by projecting visual information. The camera is employed to recognize and track user's hand gestures. As for Microsoft Touchless, a software processes the data captured by the camera and tracks the locations of the colored markers at the tip of the user's fingers using naive computer-vision techniques. As Pranav states, the current prototype system costs approximate $350 to build.

Hand gestures recognition has also been employed in the PointScreen project (http://www.iais.fraunhofer.de/600.html) from the Fraunhofer IAIS. PointScreen is a novel interface to manipulate digital artefacts touchlessly: the user navigates by pointing toward the screen and the interaction with the system is carried out completely without touch. Conversely from the previous project, the technology used in PointScreen is not based on fiducial markers but on Electric Field (EF) proximity sensors. Employing non-contact sensors that measure the interaction of a person (the dynamics of the body such as gestures and movement) with the electric fields, it provides the possibility to produce touchless input systems that efficiently use gestures for real-time navigation and interaction.

Virtual Reality wireless data gloves can be considered as other valuable devices for providing touchless interaction through hand gestures. The Cybergloves II from Imation, for example, is an instrumented glove that provides up to 22 high-accuracy joint-angle measurements. The gloves use proprietary resistive bend-sensing technology that allows transforming hand and finger motions into real-time digital data. Nevertheless, due to their high price, such devices cannot be considered suitable for being adopted in low-cost prototyping projects.

Another example of using human body as an interface is EyeWriter (www.eyewriter.org). By providing a low-cost eye-tracking hardware, this research project aims at allowing people who are suffering from Amyotrophic Lateral Sclerosis (ALS) to draw using only their eyes. The eye-tracking software detects and tracks the position of a pupil from a glass-mounted camera, and uses a calibration sequence to map the tracked eye/pupil coordinates to positions on a computer screen or projection. The software library of the EyeWriter project has been developed using openframeworks (http://openframeworks.cc), a cross platform c++ library for creative development. The eyetracking component exploits the clear and dark image of the pupil for traking the eye movements. The DIY glasses they designed, in fact, use near-infrared leds to illuminate the eye and create a dark pupil effect. The eye-drawing part was designed to work with the EyeWriter tracking software as well as commercial eye-trackers such as the MyTobii (http://www.tobii.com).

The idea of *body-as-sensor* is supposed to be the main feature of the Microsoft Project Natal (http://www.xbox.com/en-US/live/projectnatal/), which is the code name for a *"controller-free gaming and entertainment experience"* for the Xbox 360 video game platform. It aims at enabling users to control and interact with the Xbox 360 without the need to touch a game controller through a natural user interface using body gestures, spoken commands, or present objects and images. The device features: an RGB camera, depth sensor, multi-array microphone, and custom processor running proprietary software, which should provide full-body 3D motion capture, facial recognition, and voice recognition capabilities.

Muscle sensing is another cutting-edge technology for touchless user interfaces. Electromyography (EMG) sensors can decode muscle signals from the skin's surface as a person performs certain gestures. Researchers attached such sensors to their forearms, and built a gesture recognition library by monitoring muscle signals related to each gesture. The project emerged as a collaborative effort between Microsoft, University of Washington in Seattle, and the University of Toronto in Canada. Scott Saponas et al. demonstrated the feasibility of *"using muscle-computer interfaces for always-available input in real-world applications"* at the last UIST conference.

Within the emerging remote human-computer interaction devices, the Nintendo Wii controller (Wii Remote or Wiimote) can be considered as one of the most popular, due to the widespread of the console. It also turns out to be one of the most sophisticated, for providing a variety of multimodal I/O functionalities. The Wiimote is mostly advertised for its motion sensing capabilities: users can interact with a computer system via gesture recognition or pointing, by exploiting the built-in accelerometer and the InfraRed camera tracker. The most interesting part of the controller is the IR camera builted-in in the front of it. The camera provides an image processing engine, which can track up to 4 moving objects and can send coordinates to a host relative to objects position, thus giving the user fast and high precision tracking at a very low cost. Specification of the camera's hardware and software are confidential, but information can be found on Wii related websites like *wiibrew.org* or *wiili.org* (check *http://www.wiibrew.org/wiki/Wiimote* for further information as well as an exhaustive list of Wiimote known features and status of the reverse engineering process).

3. Remote interactions for screen displays with the Wiimote

The Wiimote clearly stimulates the development of touchless post-WIMP interactions intended as going beyond the WIMP (Windows, Icons, Mouse and Pointers) paradigm and interacting by exploiting new styles like multi-touch input in augmented reality environments.

Due to its interaction capabilities and its low cost, the Wiimote has gained significant attention within the homebrew software developer and Do-It-Yourself (DIY) communities, boosting the creation of several project involving multimodal interaction techniques by means of a Wiimote. These projects are usually shared over the Internet (via youtube.com or DIY websites such as instructables.com) so that other can reproduce and extend them. As professor Eric Klopfer, at MIT, said: "*The advantage of the Wiimote is that it's a human-centric device*". There are many input devices that: "*map well onto the computer's interface, but not to the person's*". By a contrast: "*The Wiimote fits the user [...] People know intuitively what to do with it when they pick it up because we use it like devices we are familiar with*". In fact, post-WIMP interfaces are about "Minority Report" style interactions (see *http://www.youtube.com/watch?v=NwVBzx0LMNQ* for an excerpt of the movie) and the Wiimote provides a cheap and effective solution to develop these kind of applications at home or in our lab without requiring so much effort than employ the adequate APIs, build small led-based devices and programming.

The integration of the Wiimote with the surrounding environment is thus quite easy and can promote the use of pervasive post-WIMP applications even when both economic and engineering factors are an issue. In fact, developing a solution like a low-cost multi-point interactive whiteboard using the Wiimote can be achieved with a relatively small budget (the cost of a Wiimote, a pen with few leds and a laptop computer, aproximately 600$) compared to off-the-shelf solutions according to the DIY philosophy.

Researchers and developers, like Johnny Chung Lee (http://johnnylee.net/projects/wii/), demonstrated how it is possible to develop applications, exploiting the Wiimote, to perform: 3D head-tracking, touchless interactions and interacting with haptic feedback. Lee used a library called *WiimoteLib* released under the Microsoft Public License (Ms-PL) to develop his prototypes. Nevertheless there are many libraries available for public use providing the same kind of interactions with Wiimote, like: GolvePie, and RMX automation under windows or LinuxCWiid under Linux, and the WiiRemoteJ which is platform independent (Java code). Readers could refer to *http://wiibrew.org/wiki/Wiimote/Library* for a quite complete list of available libraries.

The *WiimoteLib* library is based on the well-known Microsoft .Net Framework providing a set of visual Express Tools for developing code. The Coding4Fun web site (*http://blogs.msdn.com/coding4fun/archive/2007/03/14/1879033.aspx*) supplies articles and information on different prototypes and applications developed for the Wiimote by using

WiimoteLib such as many C# and Visual Basic examples provided by Brian Peek. He also publishes instructive articles on its blog (http://www.brianpeek.com/blog/) about many subjects related to the use of Wiimote for advanced interactions (post-WIMP) such as: WiiEarthVR that shows how to link Wiimote to Microsoft's Live Maps for interfacing with geo-referenced data. Starting from these examples it is quite easy to develop complex prototypes such as the one published by Johnny Lee on *http://johnnylee.net/projects/wii/* web site.

Another interesting example is *GlovePIE*[3], a software application developed by Carl Kenner (*http://carl.kenner.googlepages.com/glovepie*), intended to emulate computer input hardwares. The PIE acronym stands for Programmable Input Emulator and, as its name suggests, it was originally developed for Virtual Reality data gloves. Nevertheless, in its last updates the library offer support to a wide range of input hardware, including the Wiimote controller. Exploiting a simple scripting language it is possible to use the Wiimote to interact with different applications taking advantage of a wide library of existing script.

4. Case study: GoogleEarth and the Wiimote.

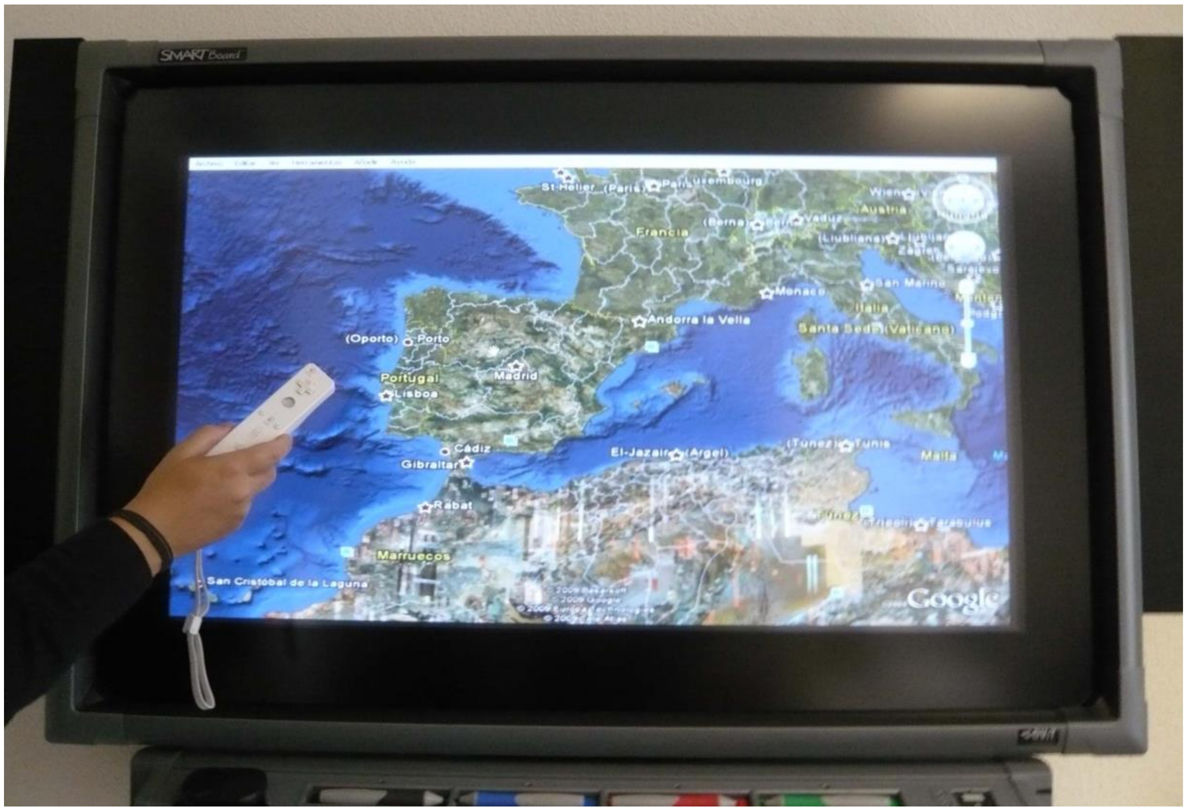

Figure 1. GLOVEPIE: USING THE WIIMOTE TO INTERACT WITH GOOGLE EARTH.

[3] GlovePIE is only compatible with Windows Operating Systems.

We present in this section a simple example to show how the Wiimote can be succesfully employed as a device to control and remotely interact with physical object with digital properties on large display surfaces. Our example regards employing the *GlovePIE* library to map Wiimote inputs into mouse and keyboard outputs.

Writing scripts for *GlovePIE* does not require any particular programming experience for the application has been designed to be user-friendly. The language syntax is extremely simple and it is easy to master in a short time for Java or C programmers. Moreover, there exists a large community of *GlovePIE* developers willing to share some tips and code with novices. You can try to figure out how things work by picking apart some of the Wiimote scripts that come bundled with *GlovePIE*.

In order to start using the *GlovePIE* application you need to get your PC to recognize the Wiimote, via a Bluetooth wireless device (built-in in for laptops or an USB dongle for desktop computers). The Wiimote, in fact, is essentially a wireless device that employs the standard Bluetooth HID (Human Interface Device) protocol to communicate and that any Bluetooth host can recognize as a standard input device. Once started the Bluetooth connection procedure you need to put the controller in *discovery mode* holding down the buttons 1 and 2, while the Bluetooth controller is searching for new devices to coupling with. No PIN is needed for the Wiimote, so you can skip this step. The pairing procedure is not fully reliable and you may need to repeat it several times, once the Blueetooth controller recognize the Wiimote as a proper HID devices (named Nintendo RVL-CNT-01).

GlovePIE comes with a library of existing scripts, that can be really useful to start developing your own script. By opening the GoogleEarth.PIE file placed in the WiimoteScripts directory (in GlovePIE 0.29), we can analize the steps needed to calibrate and use the Wiimote as an input device to interact with the GoogleEarth user interface.

Calibrating the Wiimote results in a simple: you have to place the Wiimote face up on a flat surface and adjusting the offsets for the three axis, changing these values until reaching a zero value for each axis. After calibrating you are ready to interact with the GoogleEarth application through your Wiimote: tilting up, down, left and right will simulate arrow keys; holding *B* while tilting up and down will tilt the view up and down; holding B while tilting left and right will rotate the view; pressing + and - will zoom when the Wiimote is level; pressing *Home* will center the view and pressing *1* will toggle fullscreen. You can now add your own functions into the code or edit premade ones. As an example you can assign the buttons to execute certain Wiimote functions such as rumble.

GlovePIE provides also variables to control: ***a)*** the force with respect to the three axis (a stationary Wiimote has a force of 1G) *Wiimote.gx*, *Wiimote.gy* and *Wiimote.gz*, ***b)*** the acceleration on the three axis (in m/s/s) *RelAcc* (all 3 axis), *RelAccX*, *RelAccY*, *RelAccZ*, ***c)*** the roll/pitch rotation in the Wiimote (in degrees) and **d)** the extension port. Moreover, *GlovePIE* support more than one Wiimote at time, giving the developer to build collaborative map-based applications for interact with in a pervasive screen display environment.

Nowadays, we live in an environment where digital artifacts and especially large pervasive displays represent a continuous support for our activities: communication and collaboration, entertainment, daily life, working and learning. The emergent use of touchless interfaces and efforts of both industries and academics seem to stimulate a pervasive and human-centric approach in Human-Display Interaction research. Among the possibilities offered by technology, both at hardware and software level, we think that the Wiimote is impacting research in this area especially for its abilities to provide interesting features at low cost and the wide support offered by the developers and researchers communities. Nevertheless, we can foresee that other approaches like Microsoft project Natal will encourage the growing community of researchers in Human-Display Interaction.