Wii R Free - Google Earth In Your Hands

Denzil Ferreira
University of Madeira - UMa
Colégio dos Jesuitas, 9000-081 Funchal
lizned.arierref@gmail.com

Emanuel Fernandes
University of Madeira - UMa
Colégio dos Jesuitas, 9000-081 Funchal
emanuel.m.fernandes@gmail.com

Maria Freitas
University of Madeira - UMa
Colégio dos Jesuitas, 9000-081 Funchal
carmofreitas@gmail.com

Tiago Camacho
University of Madeira - UMa
Colégio dos Jesuitas, 9000-081 Funchal
tcamacho@netmadeira.com

ABSTRACT
Recently the Wiimote has been used for exploring a new set of uses besides its original intent, gaming. With the built-in infrared (IR) and accelerometer capabilities, the Wiimote provides different ways for interacting with computers. Taking advantage of the Wiimote IR camera, we can use our fingers to interact with a map application, such as Google Earth. Presented are the results of a brief work about how to control Google Earth using hands gestures and the Wiimote.

Author Keywords
Wiimote, Gesture Recognition, Google Earth, Finger Tracking.

MOTIVATION
With the proliferation of Wii devices in the market and due to its low cost hardware, there has been an increasing creation of applications that take advantage of its built-in IR and accelerometer capabilities. We chose Google Earth, because it's an application that requires commands that can be more intuitive to be performed with hand gestures (panning, rotating and zooming).

EXISTING WORK
Gestures have been shown to be useful and intuitive in mobile augmented reality systems, because hands can be used for performing other tasks (like holding a device) besides the interaction with the system [1].

Gesture recognition is a complex task, as we need to take into account that one gesture can be interpreted differently from user to user, as we discovered when testing our system.

Chatty has identified three types of interaction for two-handed interaction: independent, parallel and combined [4]. In our work, we explored combined interaction, as one hand performed one main action (panning or rotating) and the other performed one action that didn't require precision (stopping the current command).

As Buxton and Myers [6] and also Moscovich have shown, parallel interaction is best used for solving computer tasks when dividing the task by both hands [7]. Taking this into account, in our work, the dominant hand is used to pan and rotate the Google Earth's map and globe, while the non-dominant hand stops the current gesture action.

A user perceives a related sequence of actions if there's a period of time between them of approximately 50 milliseconds [6], so we needed to make sure that the feedback given by the system was on that interval.

As Bérard's previous work [2] demonstrated, gestures have been shown to be intuitive but when the hands or fingers don't rest on a surface, the physical tiredness of the limbs is a consequence.

At last, the gestures used for the interaction must be natural and at the same time feasible to implement. Hinckey concluded that this can be troublesome [3] as actions tend to overlap, specially when there aren't any restrictions to the gestures, as happens in our work.

INTRODUCTION
The Wiimote presents itself as a powerful input device. It is capable of detecting IR light sources, like candles, incandescent lights and LED's and also provides accelerometer data on three axes. Therefore the Wiimote can be used on a wide range of applications. In our work, we explored only the IR tracking capability of the device, to execute the rotation, panning and zooming commands in Google Earth.

HARDWARE CONFIGURATION
Influenced by Johnny Lee videos[1], we explored his technique for tracking multi IR light sources. We built an electronic board with 40 IR LED's, which would be our source of IR beams. The Wiimote was positioned above the

[1]Lee, J. Tracking Your Fingers with the Wiimote, http://www.cs.cmu.edu/~johnny/projects/wii/

board, and then it would detect the reflection of the IR beams from the user's fingers or hands. We used reflective material to accomplish this reflection. The Wiimote would then communicate with the computer via Bluetooth (wireless transmission protocol).

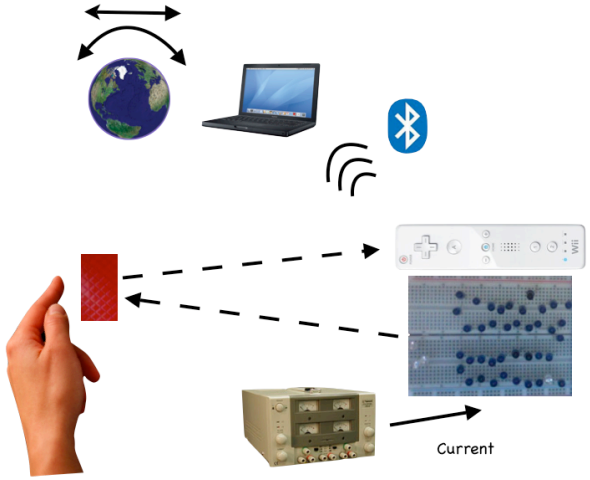

Figure 1 - Hardware configuration of Wii R Free system

As we can see in figure 1, the IR light beams reflect in the users' fingers, the Wiimote detects this point and wirelessly transmits the signal to the computer resulting in some action in Google Earth.

IMPLEMENTATION

Our system was developed on a Linux Ubuntu system, with the library Cwiid[2]. This library provides an API for accessing the Wiimote sensor detected values for the IR light beams.

We started by developing a plugin, the Wii R Free. We used the existing code for IR tracking and mouse movement available in the IR Pointer plugin, and extended the code in order to provide gesture recognition.

After establishing a connection by Bluetooth between our plugin and the Wiimote, we were able to receive the data stream from Wiimote that is then analyzed and translated into gestures, which then start mouse events that would trigger an action on Google Earth.

We implemented rotation, panning and partially the zooming gestures. Rotation and panning in Google Earth are essentially the same. The difference is that while rotation is performed in the globe view the panning is performed in the map view.

Panning and Rotating

Baudel [8] divided the gesture detection and interpretation into three phases: first there is start position with an initial gesture in an active zone; second, there is the dynamics phase and at last an optional gesture to end the command. We followed this structure to implement the rotating and panning commands, and we describe it as follows:

Initial gesture

Initially, using her dominant hand, the user points one IR reflective point to the Wiimote and moves the cursor freely in Google Earth, with no interaction with the application. This way the user can position the cursor where she wants to perform the panning or rotation. Once there is a movement of the cursor bigger than 200 pixels within 100 milliseconds, the panning/rotation gesture is triggered. The threshold of 200 pixels in 100 milliseconds was determined to be best one after some tests with users.

Dynamics phase

In this phase the user is panning the map or rotating the globe. The movement of the cursor is directly mapped to the movement of the user's hand. Once in this phase if the user wanted to perform this command continuously (the globe would rotate or the map would pan without stopping), she would make again a quick cursor movement with the threshold of 200 pixels movement within 100 milliseconds.

End of command

To release the cursor from the command being executed, using her non-dominant hand the user points to the Wiimote the second reflective IR point. The detection of two points triggers the end of the current command being executed.

Figure 2 - User interacting with our setup

As we can see in Figure 1, the user is stopping the current command by pointing the two reflective points to the Wiimote.

Zooming

The Zooming initial gesture would be described as approaching of two IR reflections for zooming out and distancing for zooming as described by Hinckley [3]. Despite the fact that we coded this gesture in the plugin, during our tests, this gesture had a faulty behavior, not

[2] A collection of Linux tools written in C for interfacing to the Nintendo Wiimote, http://abstrakraft.org/cwiid/

being able to control speed for the gesture to work properly, loosing the signal of one of the IR reflections on the process.

GESTURES MAPPING TO GOOGLE EARTH COMMANDS

The rotation, panning and zooming gestures were all translated to Google Earth commands. Google Earth was running in Full Screen mode, so that all the mouse and keyboard events would be used to interact only with Google Earth and not another application.

The panning and rotation commands were mapped to holding the mouse left click and moving the mouse around the screen. The zooming command was mapped to the mouse scroll and the keystrokes plus and minus.

Using the keyboard, the feedback provided by Google Earth is notoriously slower than using the mouse. This is due to the keystroke rate settings on the operating system, but even if we increased it to the maximum values, it was not as fluid as if we were using the mouse as input.

LIMITATIONS

The plugin we developed only detects up to two IR points reflections, in ideal conditions. Our hardware setup presents some physical restrictions, such as the motion detection range (on a 45º radius) and the distance from the Wiimote (limited by 1,5 meters), decreasing the gesture detection area. These restrictions influenced negatively the detection of the zooming command. When zooming using both hands, each hand movement requires a certain amount of space that was not available using our hardware configuration.

Due to the gestures being performed with no surface support for the hands, users complained on their arms tiredness, something that we were expecting [11], [2].

Another issue detected was the involuntary double clicking event generated occasionally. This is due to tracking problems with our software, originating sequential left mouse clicks, which on Google Earth are mapped to perform zooming in on the cursor location. Google Earth interprets double clicking as a zoom command, and sometimes this command would be triggered unintentionally.

We tried to implement the zoom command using the z-axis readings from the Wiimote, but this revealed to be unsuccessful because of the limited value variation of the data stream readings. The variation results of this sensor are very small, being difficult to interpret any continuous movement and this variation would only increase when positioning the IR reflecting points very close the to Wiimote.

The room light intensity also influenced the readings by the Wiimote, making it harder to detect the IR reflecting points. As IR's are directional, the IR light source needs to be directed at the users hands and the user needs to keep the hand facing the Wiimote for accurate tracking.

The IR reflectors on the user hands also didn't work as good as we hoped for. This is because of the reduced number of LED's we used. More accuracy could certainly be achieved if we used more LED's as IR light source.

Since we rely on LED's, these are susceptible to overheat and stop functioning. The problem is that these are IR LED's, meaning that the system can stop working and the user isn't aware of the system status, unless she has a device to detect the IR light, such as a video camera, to check if the IR light source is working or not.

Finally, if the user wears glasses or a bright shirt, the Wiimote detects false readings, as the IR bounces off from these.

CONCLUSIONS

We used an electronic board with LED's emitting IR light that would reflect in users hands (using a reflective material). The Wiimote detects these points and then our plugin maps these gestures to Google Earth commands. This approach ended not to be the best one, due to the sensitivity of IR reflection point's detection. The area of detecting was very small and the robustness of point's detection was poor. Using an IR LED pointed directly towards the Wiimote seemed to be better than reflection because it was easier to keep it directed to the camera.

The zooming action required a bigger area for the hands gestures that was not available with our hardware setup. This revealed that the zoom action when performed as a gesture without any surface support can only be achieved using a larger detection area.

As future work a different approach to gestures detection should be explored to uncover if they can be more robust. Perhaps taking advantage of the accelerometer on the Wiimote for Google Earth rotation and panning and instead of using reflective material and the LED's board, we should use LEDs in the fingers. Also gestures could be triggered by voice instead of relying on mid-air gesture recognition, or using a combination of both.

REFERENCES

1. Veigl, S., Kaltenbach, A., Ledermann, F., Reitmayr, G., Schmalstieg, D. (2002). "Two-Handed Direct Interaction with ARToolkit", Poster paper in Proc. ART'02, Darmstadt, Germany, Sept. 29, 2002.
2. von Hardenberg, C. and Bérard, F. 2001. Bare-hand human-computer interaction. In *Proceedings of the 2001 Workshop on Perceptive User interfaces*, Orlando, Florida, November 15 - 16, 2001.
3. Hinckley, K., Czerwinski, M., and Sinclair, M. 1998. Interaction and modeling techniques for desktop two-handed input. In *Proceedings of the 11th Annual ACM Symposium on User interface Software and Technology*, San Francisco, California, United States, November 01 - 04, 1998.

4. S. Chatty. Extending a graphical toolkit for two-handed interaction. In ACM UIST'94, pages 195-204. ACM Press, 1994.
5. Chatty, S. 1994. Issues and experience in designing two-handed interaction. In *Conference Companion on Human Factors in Computing Systems*, Boston, Massachusetts, United States, April 24 - 28, 1994.
6. Buxton, W. and Myers, B. 1986. A study in two-handed input. *SIGCHI Bull.* 17, 4 (Apr. 1986), 321-326.
7. Moscovich, T. and Hughes, J. F. 2008. Indirect mappings of multi-touch input using one and two hands. In *Proceeding of the Twenty-Sixth Annual SIGCHI Conference on Human Factors in Computing Systems,* Florence, Italy, April 05 - 10, 2008. CHI '08.
8. Baudel, T. and Beaudouin-Lafon, M. 1993. Charade: remote control of objects using free-hand gestures. *Commun. ACM* 36, 7 (Jul. 1993), 28-35.
9. Oviatt, S. 1999. Mutual disambiguation of recognition errors in a multimodel architecture. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: the CHI Is the Limit*, Pittsburgh, Pennsylvania, United States, May 15 - 20, 1999. CHI '99.
10. Bowman D, Wingrave C, Campbell J, Ly V. Using pinch gloves for both natural and abstract interaction techniques in virtual environments. In: HCI international, 2001. p. 629-33.
11. Freeman, W.T. & Weissman, C.D. (1994) Television Control by Hand Gestures. In 1st Intl. Conf. on Automatic Face and Gesture Recognition.
12. Quek, F. K. 1996. Unencumbered Gestural Interaction. *IEEE MultiMedia* 3, 4 (Dec. 1996), 36-47.
13. Hansen, T. (2004) Interaction without Keyboard and Mouse, http://www.pervasivehealthcare.com/ubicomp2004/papers/final_papers/hansen.pdf