**UPC** **epsc** Escola Politècnica Superior de Castelldefels

UNIVERSITAT POLITÈCNICA DE CATALUNYA

# MASTER THESIS

**TITLE: WiiMote and recognize gestures**

**MASTER DEGREE: Master in Science in Telecommunication Engineering & Management**

**AUTHOR: Angela Abad Román**

**DIRECTOR: Roc Messeguer**

**DATE: July 22th 2010**

**Título:** WiiMote and recognize gestures

**Autora:** Angela Abad Román

**Director:** Roc Messeguer

**Fecha:** 22 Julio 2010

## Resumen

Este documento contiene la memoria de la Tesis final de Máster "Wiimote and recognize gestures". Los objetivos de este se basan en la búsqueda de librerías que sean capaces de conectar las posibilidades que ofrece el mando de la consola Wii, el Wiimote, con una interfaz gráfica determinada, e interactuar con ella.

Para todo ello se procede a un análisis exhaustivo del las librerías disponibles, y se comparan las posibilidades que ofrecen. Se elegirán algunas librerías, las más completas o interesantes, para su posterior implementación junto con la interfaz gráfica que se diseñará para la ocasión.

Una vez completada la implementación de todas las partes, las librerías escogidas y la interfaz gráfica, se hace una demostración del sistema, y el uso de Wiimote para controlar la interfaz, ya sea mediante movimientos intuitivos del mando o mediante los gestos que se han definido, en el caso de interactuar mediante gestos.

Para la interacción de los usuarios mediante gestos, ha sido necesario plantear unas pruebas de usabilidad de los gestos, para verificar que esos gestos pueden ser interpretados por cualquier persona.

Finalmente, se plantearán las conclusiones obtenidas, y se propondrá una posible continuidad de esta Tesis, junto con otras consideraciones.

**Title:** WiiMote and recognize gestures

**Author:** Angela Abad Román

**Director:** Roc Messeguer

**Date:** July, 22nd 2010

## Overview

This document contains the memory of the final master thesis "Wiimote and recognize gestures". The objectives of this are based on the searching of libraries that are able to connect the possibilities offered by the Nintendo Wii controller, the Wiimote, with a specific graphical interface and interact with it.

All this is come to a thorough analysis of the available libraries, and compared their potential. Some libraries will be chosen, the most complete and an interesting one, for further implementation in conjunction with the graphical interface designed for the occasion.

Once completed the development of all parts, the selected libraries and the graphical interface, is done a demonstration of the system and the use of Wiimote to control the interface, controlling either through intuitive movements or through gestures that have been defined.

For user interaction through gestures, it was necessary to test the usability of gestures, just to verify that these gestures can be interpreted by anyone.

Finally, the conclusions are presented, and it is proposed a possible continuation of this thesis, along with other considerations.

# TABLE OF CONTENTS

# List of figures

# List of tables

# CHAPTER 1. INTRODUCTION

In this first chapter is done an introduction to the topic of the master thesis, which has been the overall motivation in doing it. It also includes a description of the project and the focus is fixed by the objectives that have to be accomplished. Then, is explained the structure that will follow this thesis.

## 1.1 Motivation

Nowadays, everything related to ubiquitous computing is growing due to the need to improve the Human Computer Interaction (HCI), in which information processing has been thoroughly integrated into everyday objects and activities.

The study of interaction between people and computer is changing; currently the most common is to use peripherals such as keyboard and mouse to interact with the computer, this makes the user to be far away of being able to get closer, in a cognitive way, to what they want to accomplish.

Researchers in HCI are interested in developing new design methodologies, experimenting with new hardware devices, prototyping new software systems, exploring new paradigms for interaction, and developing models and theories of interaction.

As new hardware devices are being experimented to be used as HCI, the motivation of this project is use an existing controller, such as Wiimote. Because, Nintendo Wii has been the first console to change the way of interaction with a new concept of playing with games, designing a hardware device that allows to recognize the user movements. In a near future Sony in the PlayStation will give the users the same possibility as Nintendo did, using a controller for gesture recognition.

## 1.2 Definition

As mentioned above, the project focuses in the use of a hardware device as a HCI to control any kind of Graphical User Interface (GUI), the Wiimote is the controller selected.

Although, the main focus is the evaluation and comparison of the existing libraries that can use the Wiimote as a HCI. Once compared, chose some of them, to demonstrate what it has been proposed above, been able to control a GUI.

To do a demonstration that it can control a GUI, it has to be design thinking in all the aspects that are going to be shown in the project.

## 1.3 Objectives

The main objective of this thesis, is to control any kind of platform GUI with the all the options the Wiimote controller have.

The specific goals for this project are:

- To get a general knowledge of how the Wiimote controller works, general characteristics, the available communication channels and the use as user interface.

- To search, classify and compare the available libraries that allow the use of the Wiimote the HCI with a computer.

- To select the most significant library, with more controller possibilities, and use it, in a design GUI.

- To use gestures done with the Wiimote to be able to have more possibilities, in case of running out of buttons in the controller.

- To test the usability of the gestures in a few people, to obtain as a result if the trained gestures are universal or only for the person who trained the gestures.

- To design a GUI where it can be shown how the Wiimote is able to control the elements presented.

- To assess the state of maturity of the libraries

## 1.4 Structure Overview

The organization of this master thesis is as follows:

- Chapter 2 presents all the necessary concepts, both conceptual and technological, for the proper understanding of the project

- Chapter 3 provides a comparative of the libraries found, then a selection of the most significant, and a deeper explanation of the selected.

- Chapter 4 introduces the way of implement GUI with the tool processing.

- Chapter 5 focuses on the demonstration of how has been developed with the libraries chosen, to control the implemented GUI. It also includes a usability testing of the gestures proposed for the training gestures library.

- Chapter 6 draws some conclusions and proposes some future works. Furthermore, this chapter presents some issues related to the environmental impact of this project.

# CHAPTER 2. Previous knowledge

This chapter explains some introductory concepts for the reader to have a prior conceptual vision of the project, following a technical explanation of the technologies commonly involved.

## 2.1 Concepts

### 2.1.1 Gesture Recognition

Gesture Recognition is the interpretation of human gestures via mathematical algorithms, in computer science. Gestures can originate from any bodily motion or state but commonly originate from the face or hand.

This technique can be seen as a way for computers to understand human body language, thus building a richer bridge between machines and humans than primitive text user interfaces or even graphical user interfaces, which still limit the majority of input to keyboard and mouse.

Gesture recognition enables HCI and interact naturally without mechanical devices, such as keyboard and mouse. Using the concept of gesture recognition, it is possible to point a finger at the computer screen so that the cursor will move accordingly. This could potentially make conventional input devices such as mouse, keyboards and even touch-screens redundant.

There are several techniques to track person's movements and determine which is the gesture that is being performed. Although there is a large amount of research done in image/video based gesture recognition, there is some variation within the tools and environments used between implementations.

*2.1.1.1 Depth-aware cameras*
   Using specialized cameras such as time-of-flight cameras, can generate a depth map of what is being seen through the camera at a short range, and use this data to approximate a 3d representation of what is being seen. These is effective for detection of hand gestures.

*2.1.1.2 Stereo cameras.*
   Using two cameras whose relations to one another are known, a 3D representation can be approximated by the output of the cameras.

*2.1.1.3 Controller-based gestures.*
   These controllers act as an extension of the body so that when gestures are performed, some of their motion can be conveniently captured by software.

The option 2.1.1.3, is the one applied in this thesis.

## 2.1.1 WiiMote

The Wiimote is the primary remote controller for the Nintendo's Wii console [1]. It has many different aspects from the others remote controllers, it has been the first one, that allows the user to interact in a new different way, with motion sensing capabilities (see Fig 2.1), using it as a pointer or via gesture recognition, to make the movement of the character more realistic of what the user does, making it suitable for all audiences and ages.
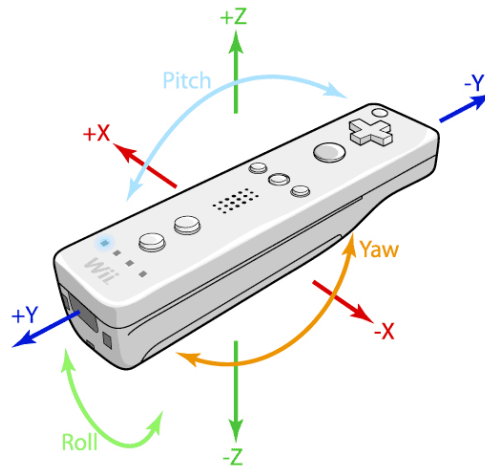


**Fig 2.1** Wiimote 3-axes movement

*2.1.1.2 How it works?*

Everything works using an accelerometer (see Fig 2.2), a device that can capture movement and rotation in three axes. So the Wiimote has motion detection capabilities in three dimensional space. The control also serves as a pointer to point at objects or look on the screen. More technically, a piece of silicon, taken from one end and positioned between the electric field produced by two capacitors, is the one responsible for transmitting motion, when moving the Wiimote, the silicon bar is closer to one of the capacitors, which makes the electric field change, which is detected, and is translated into an action of the character.



**Fig 2.2** Outline the technical operation [2]

The controller communicates with the console via Bluetooth (see 2.2.1), the console uses infrared light to detect a group of light emitting diodes (LEDs) in the sensor bar and the location and orientation acquires control to aim and shoot. So, the Wiimote can detect different movements in 3D with accurate results on screen.

## 2.2 Technologies

## 2.2.1 Bluetooth

Bluetooth is a proprietary open wireless technology standard for exchanging data over short distances (using short length radio waves) from fixed and mobile devices, creating personal area networks (PANs) with high levels of security. The key features of Bluetooth technology are robustness, low power, and low cost. The Bluetooth specification defines a uniform structure for a wide range of devices to connect and communicate with each other.

## 2.2.2 Infra-red light

Infrared radiation is electromagnetic radiation with a wavelength between 0.7 and 300 micrometers, its wavelength is longer than that of visible light, so it is not a visible light.

### 2.2.2.1 Wii Sensor Bar

The Sensor Bar is about 20 cm long and features ten infrared LEDs, five at each end of the bar (see Fig 2.3). The Wiimote senses light from the console's Sensor Bar, which allows consistent usage. The bar may be placed above or below the television, and should be centered.
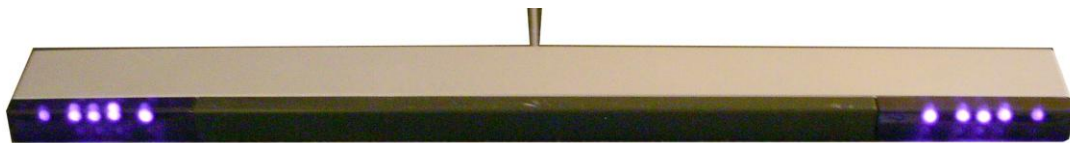
**Fig 2.3** Wii Sensor Bar with infrared lights

Use of the Sensor Bar allows the Wiimote to be used as an accurate pointing device. The Wiimote's image sensor is used to locate the Sensor Bar's points of light in the Wiimote's field of view. The light emitted from each end of the Sensor Bar is focused onto the image sensor which sees the light as two bright dots separated by a distance *"D"* on the on the image sensor. The second distance *"L"* between the two clusters of light emitters in the Sensor Bar is a fixed distance. From these two distances, the Wii central processing unit (CPU) calculates the distance between the Wiimote and the Sensor Bar using triangulation. In addition, rotation of the Wii Remote with respect to the ground can also be calculated from the relative angle of the two dots of light on the image sensor.

# CHAPTER 3. Using the Wiimote as controller

This chapter focuses on using the Wiimote as a controller of any GUI, to start a comparative study is made of existing tools already developed, which provide access to the parameters of the Wiimote, to exploit its potential with a computer. With the comparative results is done and analysis and implementation of some tools, to demonstrate its use.

## 3.1 Comparison

After an intense search of tools, the most significant are the ones presented in Table 3.1, where the main features are compared.

**Table 3.1** Comparative

|  | **Wiimotelib** [3] | **Wiigee** [4] | **Pywii** [5] | **Darwiin** [6] | **GlovePIE** [7] |
|---|---|---|---|---|---|
| **Language** | C# | Java | Python | C | Script |
| **Type** | Lib | Lib | Lib | Driver | Lib |
| **License** | Ms-PL | LGPL | GPL | BSD | CC by-nc-sa |
| **SO** | Windows | independent | Linux | Mac OS X | Windows |
| **Training** | ✖ | 5 to 10 sessions | ✖ | ✖ | ✖ |
| **INPUT** | | | | | |
| **3-axes acc. sensors** | ✔ | ✔ | ✔ | ✔ | ✔ |
| **Infrared camera** | ✔ | ✔ | ? | ✔ | ✔ |
| **Buttons** | ✔ | ✔ | ✔ | ✔ | ✔ |
| **OUTPUT** | | | | | |
| **Vibration** | ? | ✔ | ? | ? | ✔ |
| **LED lights** | ✔ | ✔ | ? | ✔ | ✔ |
| **Battery Feedback** | ? | ✖ | ? | ✔ | ✔ |
| **EXTRA DEVICES** | | | | | |
| **Multiple Wiimote** | ✖ | ✔ | ? | ? | ✔ |
| **Nunchuk** | ✔ | ✖ | ? | ✔ | ✔ |
| **Balance Board** | ✔ | ✖ | ✔ | ✖ | ✔ |
| **MotionPlus** | ✔ | ✖ | ? | ✖ | ✔ |

After this comparison, it is concluded that GlovePie is the most complete tool, where it can be programmed all the parameters of the Wiimote.

It is also interesting, the Wiigee library, because, it can go beyond the features of the Wiimote, in case of having more controlling options than the buttons that it has. Training some gestures, using only the accelerometer, those gestures recognition can generate a desired action in HCI.

So, from now on, the research is going to be centered in developing with these two tools, to be able to control the same GUI.

## 3.2 GlovePIE

GlovePIE is a free Programmable Input Emulator, originally for the Essential Reality P5 Virtual Reality Glove, which now supports a huge range of input devices, especially the Nintendo Wiimote. It is used to map gestures, button presses, and other actions to keyboard keys, mouse input, or joystick buttons and axes.

### 3.2.1 Structure

The application programming interface (API), uses a simple language to map the Input of the Wiimote to any output available, such as, keyboard, mouse, sound. It also allows doing a logic application, for example, if only the B button is pressed, map the accelerometer coordinates.

The GUI allows the user to program its own, and it also has a tab where the user can choose the Output to emulate and map it to an Input (Fig 3.1).
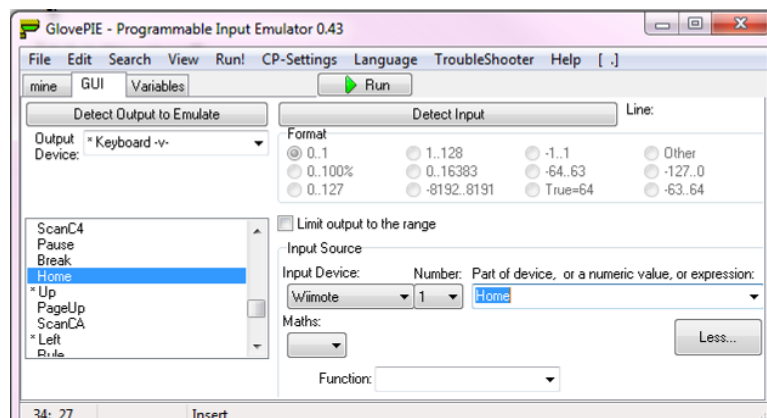


**Fig 3.1** Example of the GlovePie GUI

For example, to set some keyboard keys to a Wiimote button, the code in Table 3.2 is the one needed:

**Table 3.2** Setting keys to Wiimote buttons with GlovePIE

```
Key.Up = Wiimote.Up
Key.Down = Wiimote.Down
Key.A = Wiimote.A
Key.One = Wiimote.One
```

It is also possible to control the turn on or off of the four blue LED's individually, in the Wiimote, as shown in Table 3.3:

**Table 3.3** Controlling LED's with GlovePIE

```
Wiimote.Led1 = false
Wiimote.Led2 = true
Wiimote.Led3 = true
Wiimote.Led4 = false
```

## 3.2.2 Elements

As already has shown in 3.2.1, it is possible to control the buttons and the LED's of the Wiimote, in an easy way, but there are other parameters that need some more explanation.

### 3.2.2.1 Accelerometer

There are many ways to obtain information of the accelerometer in the 3 axes, taking into account the force, the acceleration in a specific axis, or the information of angle rotation of the sensor.

The force is applied in the three axes, X, Y and Z. The outputs are values that control the inclination of the different angles of the Wiimote and vary according to its position. This setting is programmed in a similar way as buttons, as in Table 3.4, except that we must take into account the possible values.

**Table 3.4** Raw Force in the 3 axes

```
Wiimote.RawForceX
Wiimote.RawForceY
Wiimote.RawForceZ
```

For acceleration, is programmed in the same way, as in Table 3.5, but the output is the relative acceleration of the Wiimote.

**Table 3.5** Relative Accelaration in the 3 axes

```
Wiimote.RelAccX
Wiimote.RelAccY
Wiimote.RelAccZ
```

I can be also control the angle each of the axes, like the Pitch, Yaw and Roll, as in Table 3.6. Where they have a 0 value when the Wiimote is laid down, the angles are positive or negative depending on the initial inclination.

**Table 3.6** Inclination in degrees of the 3 axes

```
Wiimote.Roll
Wiimote.Pitch
Wiimote.Yaw
```

It depends on the movement that is going to be reported, that is chosen between the three options.

## 3.3 Wiigee

Wiigee is an open-source gesture recognition library for accelerometer-based gestures specifically developed for the Nintendo Wiimote. It is implemented in Java and, thus, is platform-independent. Using a third-party Bluetooth library Wiigee allows defining and recognizing, the freely trained gestures.
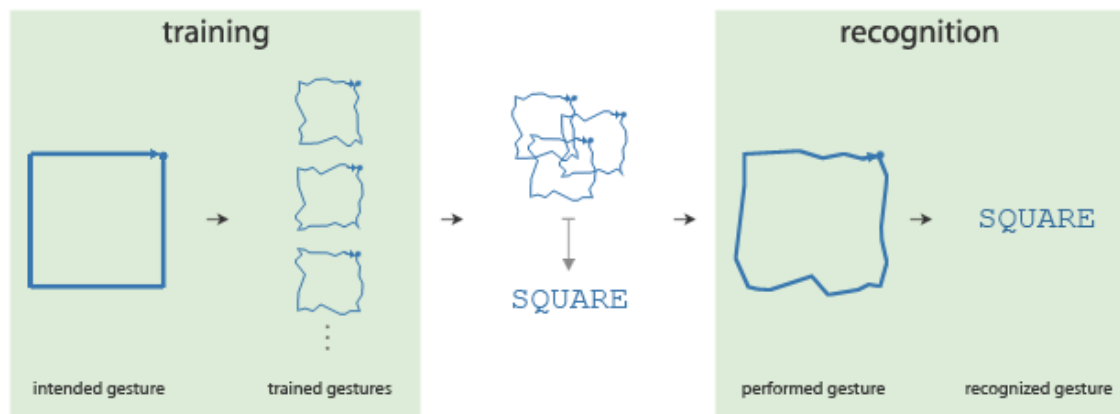


**Fig 3.2** Wiigee workflow [4]

### 3.3.2 Including to a Java project

First, it should be added the library Wiigee.jar to the classpath of the project. Afterwards, make at least its control-class accessible via a common import-command like the one in Table 3.7.

**Table 3.7** Wiigee library import

```
import org.wiigee.control.WiimoteWiigee
```

It is assumed that the Bluetooth library, corresponding to the using driver, is already added.

Then, is set up a Wiigee-object, as in Table 3.8.

**Table 3.8** Wiigee object

```
WiimoteWiigee wiigee = new WiimoteWiigee()
```

*3.3.2.2 Configure buttons action*

There are three buttons that have to be configured in order to control the flow of the training and the recognition process, shown in Fig 3.2.

- TrainButton: starts the recording process for a single gesture and has to be held down during recording.

- CloseGestureButton: if there are enough gestures recorded, it finishes the gesture recording session, and allows putting a name to the gesture recorded.

- RecognitionButton: starts the recognition of the movement done with the Wiimote, and when it is released compares the data of the movement obtained with the all the gestures of gestureset.

To configure the three buttons as in Table 3.9.

**Table 3.9** Default configuration of buttons

```
wiigee.setTrainButton(Wiimote.BUTTON_A);
wiigee.setCloseGestureButton(Wiimote.BUTTON_HOME);
wiigee.setRecognitionButton(Wiimote.BUTTON_B);
```

*3.3.2.3 Set up event listeners*

In order to be informed when a button on is pressed or to get pure motion acceleration data, the class has to implement the interface WiimoteListener. If it is wanted to be informed when a specific gesture occurred, the class has to implement the interface GestureListener. Let's assume that mygui is the class implementing these interfaces.

Then, have to add this class as the appropriate listener to the wiigee-object via Table 3.10:

**Table 3.10** Class as a listener

```
wiigee.addWiimoteListener(mygui);
wiigee.addGestureListener(mygui);
```

# CHAPTER 4. Designing a GUI

As already mentioned, with the capabilities of the Wiimote will control a specific GUI, this is done just to justify that the device can be used to control any type of application.

To develop the GUI it is done with Processing 0, which is a Java based platform, which allows the visualization of an applet in the web and it is multiplatform.

## 4.1 Processing

Processing is an open source programming language and environment for people who want to program images, animation, and interactions.

The API is very simple, as shown in Fig 4.1, it use a similar language to Java. It is composed of some intuitive buttons, such as, play or stop the execution of the program, load, save and export a program. And creating a new class for the project with the button plus, that adds a new tab.
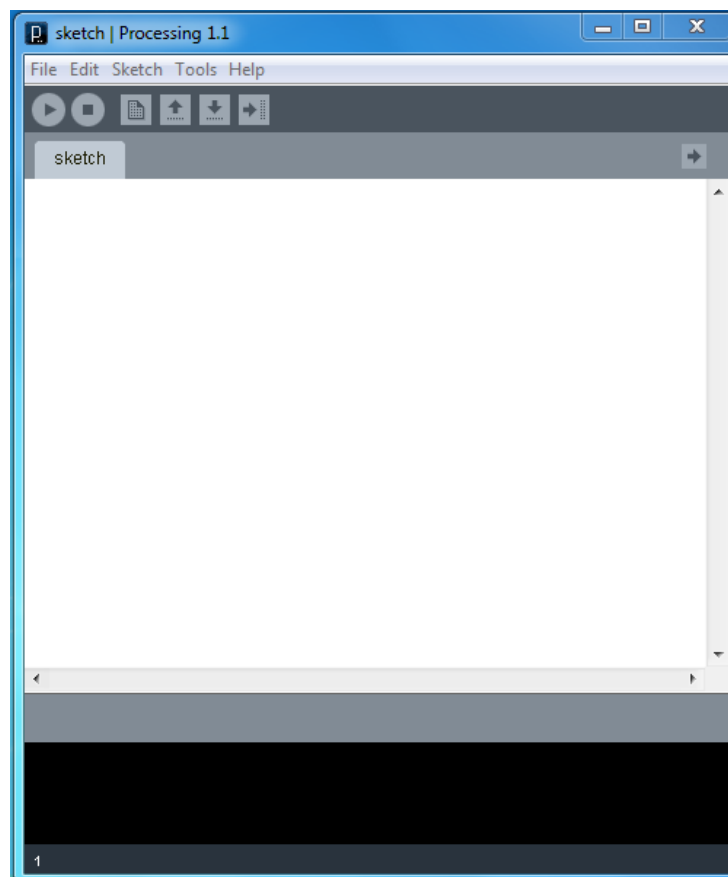


**Fig 4.1** Environment of Processing

## 4.1.2 Structure

Programming in Processing is quite simple once the structure is clear, there are some parts of the code that must appear.

### 4.1.2.1 Global variables

Are placed in the top of the main program, and they can be used in the rest of the parts. Variables are declared to a type and are initialized.

### 4.1.2.2 Setup

This part is called once the program is started, and should include the parts that are executed at first instance. Used to define properties such as like screen size, framerate, background color, the way the objects will draw, from the center or from one corner, etc.

Variables declared within *setup()* are not accessible within other functions. There can only be one *setup()* function for each program and it should not be called again after its initial execution, as in Table 4.1.

**Table 4.1** Inizialization of *setup()*

```
void setup(){
  size(1000, 700, P3D);      //canvas size and mode
  rectMode(CENTER);          //creates rectangles from the center
}
```

### 4.1.2.3 Draw

It is called directly after *setup()* and continuously executes the lines of code contained inside its block, as in Table 4.2, until the program is stopped. Is the function where most of the elements are drawn.

**Table 4.2** Initialization of *draw()*

```
void draw() {
  statements
}
```

### 4.1.2.4 Functions

Functions can be created to define some actions, or whatever other usage. There are two types, with no value return, that are declared in the way shown in Table 4.3:

**Table 4.3** Definition of functions

```
void Function_name {
   statemnts
}
```

Or if there is a return value of the function, are declared with the type retuned, boolean, integer, as in Table 4.4

**Table 4.4** Definition of functions with a return value

```
type Function_name {
   statements
}
```

There are other functions with an input feedback that allows to know when a key in the keyboard is pressed, or when is done an action with the mouse.


*4.1.2.5 Class*

A class is a composite of data and methods (functions) which may be instantiated as objects. The class can be declared in the main tab after the *draw(),* or it can be added a tab to the main one, declare the class like Table 4.5.

**Table 4.5** Definition of a class

```
class ClassName {
   statements
}
```


## 4.1.3 Elements

There are many elements that can be drawn with processing, rectangles, boxes, circles, spheres, ellipses, lines, curves, text, etc and most of them use the same way to declare them.

Before to draw any type of element, its properties are initialized, such as color, stroke, translation, rotation, etc. and then it is declared the element, as in Table 4.6.

**Table 4.6** Common properties to elements

```
translate(width/2, height/2), 0);
fill(color(230, 30, 50));
noStroke();
box(80);
```

The result is a red box, with a size of 80x80x80 pixels, without stroke, placed in the middle of the canvas.

# CHAPTER 5. Demonstration

In this chapter is explained, how all that has been explained in previous chapter, is shown in a real case, with a GUI done to interact with the Wiimote, in both cases using Wiimote's features, and using trained gestures to interact.

## 5.1 The GUI

In Fig 5.1, it is shown the aspect of the GUI developed. It has a three boxes system, to allow the user to orient in the 3 axes. The red one is the master of the system; it has a blue one which is below, and the green one which is behind it. All of them move at the same time, and with the same reference.
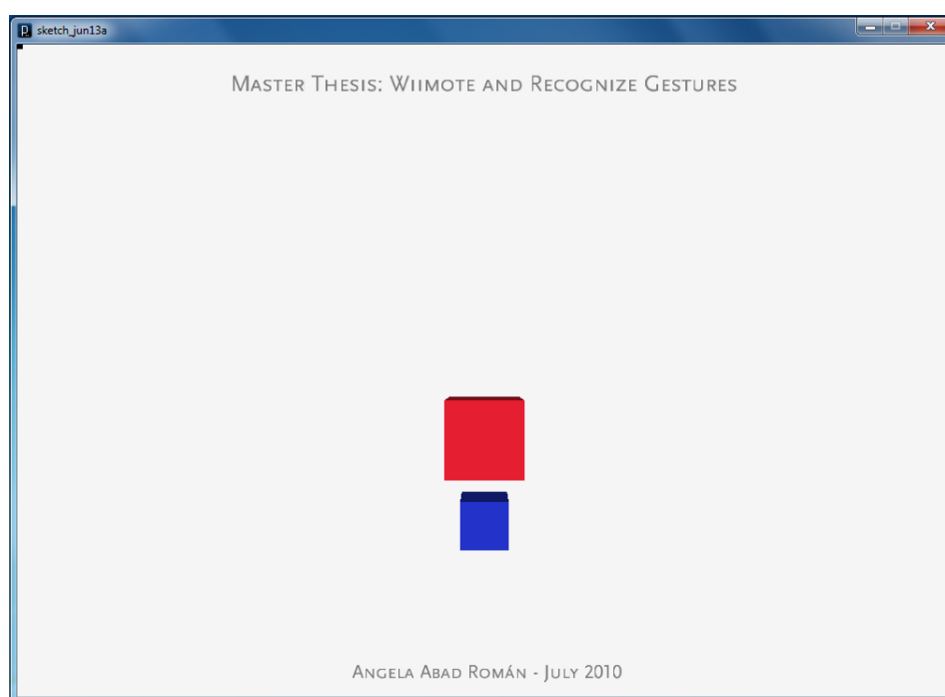


**Fig 5.1** Aspect of the GUI designed

The rest of the canvas can be used as a blackboard to paint on it, in a black color.

The GUI can be used in case of not having the Wiimote, it is possible to control it with the keyboard, to move the boxes (WSAD) and dragging the mouse, to paint in the canvas.

## 5.2 Using GlovePIE

Then, it is explained how to interact with the GUI with what has been developed controlling the Wiimote with GlovePie. It is able to move the boxes and also painting.

## 5.2.1 Moving the elements

For moving the boxes, it is take into account the information of the accelerometer, as shown in Fig 5.2, Fig 5.3, Fig 5.4 and Fig 5.5, the inclination of the Wiimote is translated in the movement of the boxes. To avoid not wanted movements in the boxes, while holding the Wiimote, it is compulsory to press button B in the Wiimote, while making the movements,.



**Fig 5.2** Moving the Wiimote forwards
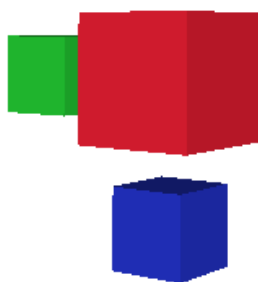


**Fig 5.3** Moving the Wiimote backwards



**Fig 5.4** Rolling the Wiimote to the right
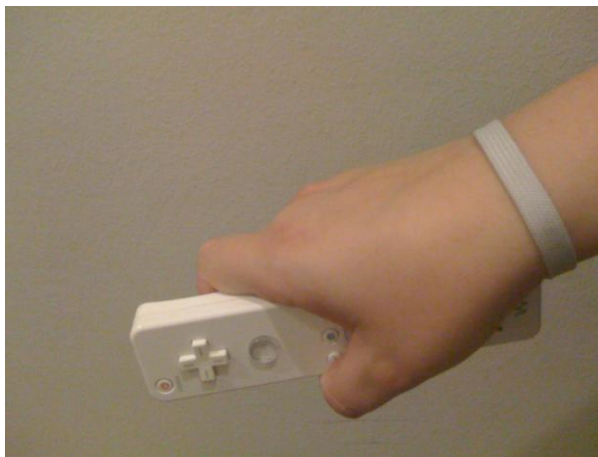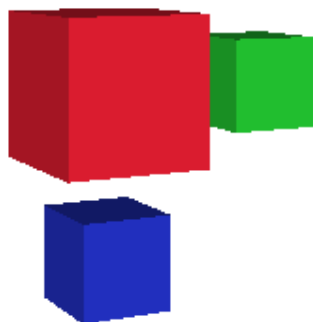
**Fig 5.5** Rolling the Wiimote to the left

So, moving the Wiimote in all directions, the moving elements can be seen from any angle.

It can also be seen the boxes system closer or further, with the + and - buttons in the Wiimote.

## 5.2.2 Painting

To paint in all the canvas, as shown in Fig 5.6, is used the infrared sensor information of the Wiimote, as explained in 2.2.2.1, it is needed the sensor bar, to have the two infrared points of reference. To paint, the Wiimote must point to the screen, and have the A button pressed.
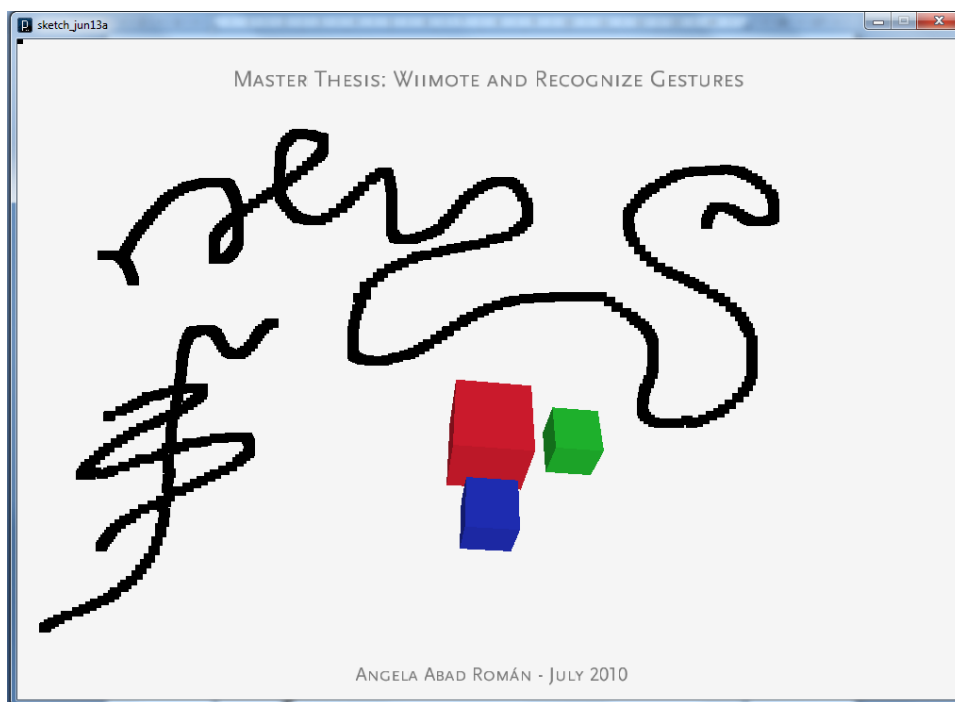


**Fig 5.6** Painting in the canvas while pointing at screen

## 5.3 Using Wiigee

As done before with GlovePIE, in this section is going to be explain how to control the GUI with the Wiigee library.

It has been defined some gestures to control GUI, this gestures has been trained and saved, in a gesture set done by the library Wiigee.

### 5.3.1 Moving elements

To move the boxes some gestures have been defined, in Fig 5.7, Fig 5.8, Fig 5.9 and Fig 5.10, the arrows explain the gestures. They start in one point, is done the movement and return to the start point.



**Fig 5.7** Moving forward



**Fig 5.8** Moving backward

**Fig 5.9** Rolling right



**Fig 5.10** Rolling left

To make a zoom in and out, the gesture defined is making a circle, as shown in Fig 5.11 and Fig 5.12, it starts from the bottom and the action depends on the way that is moved clockwise or counterclockwise.



**Fig 5.11** Zoom in

**Fig 5.12** Zoom out

## 5.3.2 Gesture usability

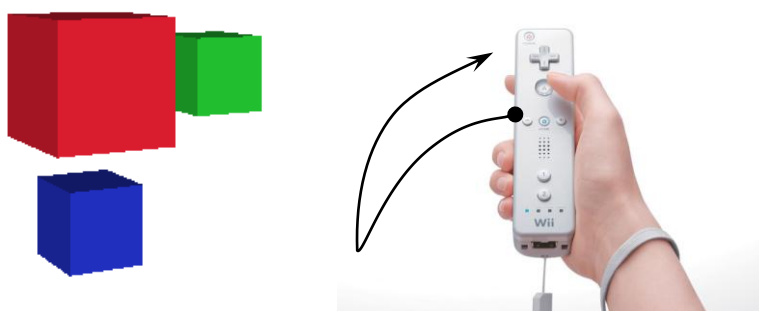It is necessary to make a usability test of the gestures trained, just to know if those gestures are easily recognized if another person, not the one who trained them, can make them with the same probability of success.

To make a usability test it is necessary to have at least two people more to be able to draw some conclusions of the results of the test and compare it with the result of the person who trained the gestures.

It is given to the some users Fig 5.7, Fig 5.8, Fig 5.9, Fig 5.10, Fig 5.11 and Fig 5.12, and they have to reproduce each one of the gestures 10 times, to have an accurate result. In the Table 5.1, we can see the results, and how many times the gesture is well recognized. The user 0 is the one that trained the gestures.

**Table 5.1** Usability of gestures defined

| Gestures | User 0 | User 1 | User 2 |
|---|---|---|---|
| Forward | 90% | 60% | 90% |
| Backward | 100% | 100% | 90% |
| Roll right | 100% | 100% | 100% |
| Roll left | 90% | 100% | 100% |
| Zooms In (circle m) | 40% | 10% | 10% |
| Zoom Out (circle n) | 90% | 90% | 70% |

Users searched for the test have been different generational age and opposite relation to technology, in order to get results that can be extrapolated to a large proportion of the population:

- User 1 profile is a 24-year-old woman, who knows the use of the Wiimote in Wii games, related to technology.
- User 2 profile is a 45-year-old woman, who has never played with a Wiimote before, is not very familiar to technology

Taking into account the results obtained in Table 5.1, and the user profiles, one of the most important aspects is that almost all the gestures are recognized when the three users interpret them at nearly the 100% of the times.

Sometimes the gesture is not understood 100%, and is confused with another gesture. For example in the case of the Forward gesture, is often confused with Backward gesture, which movements are opposite. It is observed that many times depends on the way to handle the Wiimote at the start of the gesture, sometimes with a relax position as pointing down.

The normal position which has been establish for the use of the Wiimote is inclined some degrees pointing up, which is the comfortable way to handle it when playing games, so it is exported the position of handle to this test.

The most remarkable point is the error of detection, in the three cases, of the Zoom In gesture, which is confused more or less in the 90% of cases with the Zoom Out gesture. This happens because it is not well detected the direction of the circles in the air, and it is interpret in most cases the Zoom Out. The times when Zoom Out gesture is not well recognized, it is always confused with the Zoom In gesture. Maybe, there is not enough gesture data to distinguish between them.

To conclude, the library is able to understand the gestures of anyone in most of the cases with at least a 90% of success, it does not matter if is the person that trained the gestures or another person not related to the platform, just looking at the draws where each gesture is explained.

The error obtained with the Zoom In and Zoom Out gestures, is due to a not well trained gestures, so there is not enough data stored.

This last point shows that it is important to know the minimum number of repetitions of the gesture during training is the optimum. In the specifications of the library is advised to repeat the gesture from 5 to 10 times when training, when more repetitions the system become more accurate in the detection.

In practice the number of times you have to repeat a gesture during training, for further positive recognition, depends entirely on the type of gesture. Thus cannot establish a minimum number of training, with all gestures, and therefore it is advisable to follow the recommendation of the library, and repeat the gesture for 10 times or even 15.

# CHAPTER 6. Conclusions

In this last chapter, it is resumed all the things done in this master thesis, is done a critical review of the tools used. Is taken into account the environmental issues involved. And finally, some future work is proposed.

## 6.1 Overview

As a result of this master thesis has explained the concepts related, revealed the features of the Wiimote, and its possibilities. It has been made a deep s search and comparison of the existing libraries that can control the Wiimote as a human computer interaction.

From the comparison are obtained the most complete libraries, the first one, GlovePIE, which is the one with more features configurable, the other was different from the rest, Wiigee, which is able to detect gestures.

Both libraries are used in a graphical user interface design for this thesis, and design to the use with the Wiimote. This GUI is developed with the Processing tool, once it is completely developed, the libraries are used to communicate the Wiimote with the interface, in case of GlovePIE accelerometer data, buttons, and IR cam is used, and with Wiigee is has been fixed some specific gestures.

After defining the gestures of Wiigee, some usability test are done with three users, to evaluate whether the recognition of the gestures will be universal, for any kind of people that would like to use it. Some conclusions are obtained, and mostly all the gestures are recognized with a 90% of success or more. Unless, two similar gestures, Zoom In and Out, which are easily confused with each other.

This confusion of gestures, leads us to conclude that we cannot fix a minimum number of training repetitions and therefore have to follow the recommendations of the library repeat from at least 10 to 15 times.

All the objectives set out in section 1.3 have been met satisfactorily.

## 6.2 Review

After all the research done, both libraries are well prepared for any kind of use of human computer interactions. Below is an overview of each one of them.

### 6.2.1 GlovePIE

It is only available for Windows OS, and is very easy to develop with this library, in scripting format, for many configurable inputs and Wiimote peripherals, such as Nunchuk, BalanceBoard, WiiMotion Plus, Guitar Hero, Rock Band.

Using accelerometer data, the response is natural, when moving Wiimote, is what the elements in the GUI move also. Also, accurate pointing response when using the Sensor Bar, with Infrared Light.


### 6.2.2 Wiigee

As is a Java library it is supported by any OS. This library allows using gestures as a way of not using buttons or accelerometer info as data processing, just as a recognized gesture, so there are infinite possibilities and actions to associate.

But the gestures are not always well recognized; it depends on the set of gestures, and if there is some similar, so the library can get confused.

It allows saving gestures, by training it repeated times and then store it with the desired name, making a gesture set, which can be loaded in the next usage.


### 6.2.3 Conclusion

Both libraries are very mature, but Wiigee has the disadvantage that gestures are not well recognized at a 100% of success, so there is a margin of error which is quite remarkable to the user, when training gestures they must be very different in order to distinguish between them. And also the user needs to learn how the gestures are done, because are not intuitive at all.

I think that GlovePIE is more intuitive, for developing, and also is easier the learning of the movements of usage with the interface.


## 6.3 Environmental considerations

The Wiimote, the sensor bar and the computer used for development, meet the essential requirements of European Directive 1999/5/EC, in establishing guidelines in relation to the components used in manufacturing the device and the testing process that have been submitted.

At the end of their useful life each one of these devices, have to be deposited in containers of electrical and electronic waste, for recycling

The use of batteries as power supply for the Wiimote is harmful to health and for the environment, if they are not well recycled. Batteries contain heavy metal materials that are harmful for health. Over time, the layer that protects the battery decompose easily, and if they are leaved in a inappropriate place, such as soil, waste leak and can contaminate the water, with what that entails. Therefore it is advisable to use rechargeable batteries or direct connection to power source, allowing longer duration, and prevents the replacement of a large number of batteries.

## 6.4 Future Work

In this last section, some future work is proposed in case of continuing with the all the research and achievements done in this master thesis.

The use of Wii peripherals can be very interesting, for example the use of Wii BalanceBoard, and that the user is able to move through an interface with the movement of its whole body, distributing the weight in the board.

Design a complex GUI with more elements and more types of interaction depending on the device used, such as Wiimote, Nunchuk or BalanceBoard, or the way of moving the device. It can be included color changing, audio sources, etc.

Make a more reliable study of the gestures usability, it is needed a large number of people, 20 or 30, with many different profiles and with a wide range of ages, from children to elderly.

It could be interesting to make another type of usability test, to ask the users to make a gesture for each action, as they seem, and then extract for each case which is the most common gesture, so it will be intuitive to people, and adopt it in the gesture set used.

# References

[1]     Wii at Nintendo (2010). Retrieved February 2010, from http://www.nintendo.com/wii

[2]      Dentro del Wiimote del Nintendo Wii (2010). Retrieved March 2010, from http://www.cristalab.com/blog/dentro-del-wiimote-del-nintendo-wii-c37627l

[3]     Managed Library for Nintendo's Wiimote (2010). Retrieved March 2010, from http://wiimotelib.codeplex.com/

[4]     Wiigee - a Java-based gesture recognition library for the Wii remote (2010). Retrieved February 2010, from http://www.wiigee.org/

[5]     Pywwi – wiki robotics (2010). Retrieved March 2010, from http://www.iearobotics.com/wiki/index.php?title=Pywii

[6]     DarwiinRemote: Project Web Hosting, (2010) Retrieved March 2010, from http://darwiin-remote.sourceforge.net/

[7]     GlovePIE$^{TM}$ – Programmable Input Emulator (2010), Retrieved April 2010, from http://glovepie.org/

[8]     Processing web page (2010). Retrieved May 2010, from http://processing.org/

[9]     T. Schlömer, B. Poppinga, N. Henze and S. Boll, "Gesture Recognition with a Wii Controller" In: Proceedings of the 2nd International Conference on Tangible and Embedded Interaction. Bonn, Germany. February 2008. Unofficial Draft.

[10]    B. Poppinga, "Beschleunigungsbasierte 3D-Gestenerkennung mit dem Wii-Controller" at University of Oldenburg, Germany. September 2007.

[11]    Johnny Chung Lee - Projects - Wii (2010). Retrieved February 2010, from http://johnnylee.net/projects/wii/