

Plantilla para la memoria del trabajo de IMD

Curso 2k14/2k15

Apellidos, nombre	Samuel Villaescusa Vinader (saviivi@inf.upv.es)
Titulación	Grado de Ingeniería informática
Fecha	Marzo de 2015



1	Resumen de las ideas clave	1
2	Introducción	2
3	Objetivos	2
4	Aplicación wiiZarra (explicamos su funcionamiento)	3
5	Johnny Chung Lee y wiiMotion.....	8
6	Aplicación wiiControlIMD y su funcionamiento.....	9
	6.1 Módulo calibrar.cpp.....	9
	6.2 Módulo conexionWii.cpp	12
	6.3 Módulo wiiControlIMD.cpp.....	14
7	Conclusión.....	19
8	Bibliografía.....	21
	8.1 Referencias de fuentes electrónicas:	21



Índice de figuras

Figura 1 . Ventana <i>build options</i> de CodeBlocks del proyecto wiiZarra...	4
Figura 2: Ventana Compiler de CodeBlocks del proyecto wiiZarra.....	5
Figura 3. Menú de la aplicación wiiZarra.....	6
Figura 4. Detectando mando en wiiZarra.....	6
Figura 5. Bolígrafo casero de infrarrojos (para mover y pulsar el ratón)....	7
Figura 6. Pantalla de calibrado del mando en wiiZarra.....	7
Figura 7: Código de pantalla de calibrado de wiiZarra.....	8
Figura 8: Código de compilado para el módulo calibrar.....	9
Figura 9: Fragmento de código del módulo calibrar.....	11
Figura 10: Fragmento de código del módulo calibrar para dibujar cuadrado.....	11
Figura 11: Programa calibrar ejecutado.....	12
Figura 12: Código de compilado para el módulo conexionWii.....	13
Figura 13: Código de reconocimiento del mando de la wii.....	13
Figura 14: Código de reconocimiento del mando de la wii.....	14
Figura 15: Código de compilado para el módulo wiiControllMD.....	15
Figura 16: Menú de la aplicación wiiControllMD.....	15
Figura 17: Reconocimiento del mando wiiControl.....	16
Figura 18: Menú para encendido de leds del mando wii.....	16
Figura 19: Mando de la wii sin luces leds encendidas.....	17
Figura 20: Mando de la wii con el led 1 encendido.....	17
Figura 21: Fichero de configuración del mando de la wii (infrarrojos).....	18
Figura 21: Fichero de configuración del mando de la wii (diapositivas)..	19
Figura 22: Fichero de configuración del mando de la wii (pong).....	19



UNIVERSIDAD
POLITECNICA
DE VALENCIA

Índice de tablas

Tabla 1: Objetivos del trabajo IMD.	1
--	---

1 Resumen de las ideas clave

La siguiente tabla presenta los objetivos clave que se van a intentar cumplir en la medida de lo posible en el presente trabajo de IMD, se desarrollaran los temas, explicando cómo se ha conseguido completar cada objetivo, así como problemas surgidos, mejoras etc.

Objetivos del trabajo de IMD
1. Conseguir poner en funcionamiento los proyectos wiiZarra, así como revisar su código y entender su funcionamiento.
2. Buscar información acerca de la librería wii de Johnny Chung Lee para iniciar la pequeña aplicación.
3. Conseguir que la "pequeña aplicación" reconozca el mando de la wii.
4. Explicar el código de reconocimiento de mando y probar a encender los leds del mismo.
5. Calibrar el mando de la wii en la "pequeña aplicación" de manera que el usuario busque las 4 esquinas de la pantalla.
6. Conseguir que el ratón del ordenador se mueva mediante el uso del mando de la wii.
7. Conseguir hacer <i>click</i> del ratón mediante el uso del mando de la wii.
8. Conseguir que se pasar transparencias haciendo uso del mando de la wii.
9. Crear un pequeño juego de una pelota y una raqueta (con Scratch).
10. Hacer que la raqueta se mueva mediante el uso del mando de la wii.

Tabla 1: Objetivos del trabajo IMD.

2 Introducción

El presente documento trata sobre el desarrollo de una pequeña aplicación, la cual va a permitir la interacción del mando de la wii mediante el uso de la librería cwiid para c y linux.

La aplicación consistirá en un pequeño menú, en el que mediante el uso de la consola se irá a cada una de las diferentes funcionalidades, como la de mover el puntero del ratón mediante el uso del mando, pasar transparencias del *power point*, jugar a un pequeño juego que desarrollaremos etc.

La base del trabajo consistirá en explicar el funcionamiento de wiiZarra, un programa por el cual se mueve el puntero del ratón mediante un puntero de infrarrojos, y explicar el código de reconocimiento del mando de la wii, así como alguna de las funcionalidades que permite la librería.

Por último cabe mencionar que se ha creado un pequeño juego con el lenguaje de programación Scratch en el cual se demuestra la usabilidad de cwiid permitiendo mover una de las raquetas con la cruceta del wiimotion.

3 Objetivos

El documento está estructurado en 10 objetivos clave, los cuales se van a ir describiendo a partir de este apartado.

1. El primer objetivo consiste en la recopilación de los proyectos wiiZarra [4], el cual fue desarrollado por un antiguo alumno, compilarlos, ejecutarlos y entender su funcionamiento.
2. Una vez se han entendido los anteriores proyectos y se tiene una base sobre la que trabajar, se procede a consultar las librerías de wii y los ejemplos de Johnny Chung Lee, para poder iniciar la "pequeña aplicación.
3. Cuando se tenga conocimiento suficiente sobre las librerías de wii, se procederá a iniciar la creación de la aplicación, con la primera funcionalidad de reconocimiento del mando de la wii.
4. Posteriormente, se procederá a conectarse al mismo y probar que funciona encendiendo los cuatro leds del mismo.
5. Se procederá a hacer uso de la funcionalidad de la que dispone cwiid, denominada wminput, la cual permite configurar el mando para hacer de ratón.
6. Una vez reconocido el ratón se le pedirá al usuario que calibre el mismo colocando el ratón en las 4 esquinas de la pantalla.
7. Una vez hecho esto el usuario podrá por ejemplo mover el ratón y hacer *click* en algún icono del ordenador.
8. Se dispondrá de una serie de botones, uno de ellos será el que permitirá pasar diapositivas de *power point* mediante el uso del mando (configurando wminput).
9. Por último y no menos importante se desarrollará un pequeño juego que consiste en una pelota que rebota sobre los bordes de la pantalla y una raqueta para golpearla (mediante el uso de *Scratch*)
10. Por último se dotará de movimiento a la raqueta con el uso del mando de la Wii.

4 Aplicación wiiZarra (explicamos su funcionamiento)

Lo primero que se ha realizado es la instalación de las librerías de cwiid disponible en [2] y de gtkmm disponible en [1] para poder realizar el siguiente paso de compilación y comprobar el funcionamiento de wiiZarra.

Lo segundo que se ha realizado es la compilación de wiiZarra (el cual se puede encontrar en [4]), cabe decir que ha resultado muy costosa, ya que hay demasiadas variables a contemplar en la compilación del programa, por lo que se va a explicar paso a paso el proceso.

1. Se descargó la versión que se encuentra disponible en [4], y se intentaron seguir las instrucciones de compilación de las que ahí se dispone, a la hora de compilar el proyecto (mediante CodeBlocks [3]) las opciones eran insuficientes, en la *Figura 1* se pueden observar todas las opciones que se han añadido al compilador de CodeBlocks para conseguir poner en funcionamiento el programa:

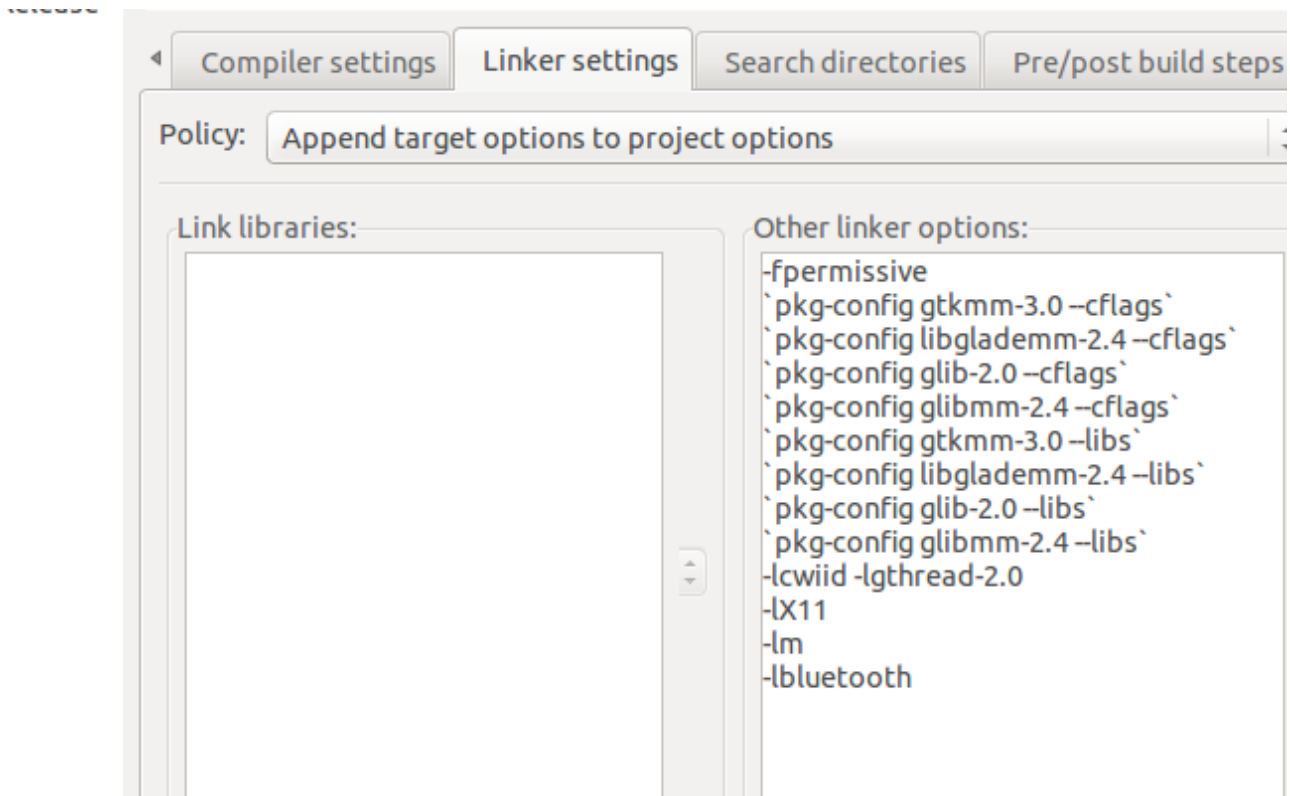


Figura 1: Ventana Build Options de CodeBlocks del proyecto wiiZarra.

además de una función necesaria que se necesitó añadir al compilador general de CodeBlocks que se muestra en la *Figura 2*:

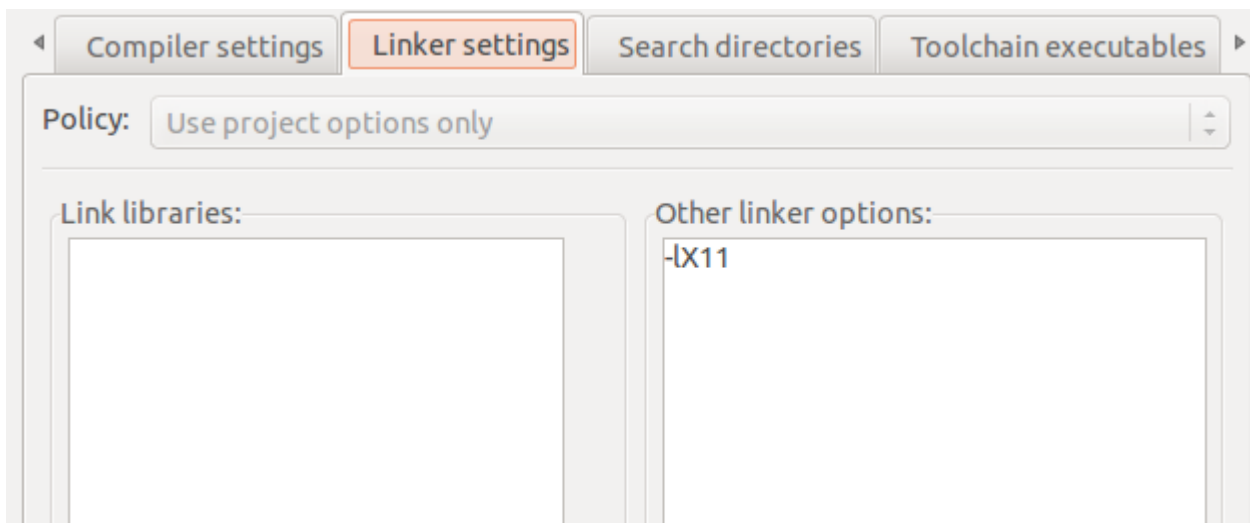


Figura 2: Ventana Compiler de CodeBlocks del proyecto wiiZarra.

2. Para acceder a la ventana de la Figura 1, hay que pulsar con el botón derecho sobre el proyecto una vez se ha cargado en el entorno de desarrollo CodeBlocks, y pulsar en Build Options del menú desplegable, una vez en esa ventana tendremos que ir a *Linker settings*, y en "*Other linker options*" colocar todas las primitivas que necesitamos añadir al compilador del proyecto, hay que tener en cuenta ciertas primitivas, `-cwiid` se añade para poder compilar la parte de la librería para el uso de *wiimotion*, `-lblueetooth` para el reconocimiento de dicho dispositivo en el proyecto y las demás librerías son para reconocer *gtkmm* para desarrollar interfaces en sistemas *gnome*.
3. A la etiqueta `-fpermissive` le vamos a dedicar algo más de explicación, ya que no es necesario incluirla en todas las máquinas, puesto que se incluye para evitar un *warning* que evita la compilación del proyecto, y es debido a una variable de la librería *cwiid* que nos permite reconocer cualquier dispositivo que conectemos al bluetooth. Dicho problema viene comentado en el api de *cwiid* disponible en [2], en la cual se comenta que BlueZ tiene un comportamiento bizarro a la hora de implementar la variable `BDADDR_ANY` (la cual se encarga de reconocer cualquier dispositivo que se conecte al bluetooth) y se recomienda no pasar directamente dicha variable al método `wiimote_connect`, ya que se puede cambiar el valor de la variable.

Por último, una vez explicado el paso de compilación, pasaremos a comentar la funcionalidad de *wiiZarra* así como algunas partes de código que nos parecen interesantes para su entendimiento. Lo primero que nos encontramos al arrancar la aplicación es el menú, donde se pueden observar tres botones: el primer botón (Conectar) permite el reconocimiento del mando de la wii, el segundo (Calibrar) permite calibrar el mando una vez éste ha sido reconocido y el último (Activar) permite el uso del bolígrafo de infrarrojos para simular el movimiento del ratón en el pc, podemos observar la composición del menú en la Figura 3.



Figura 3: Menú de la aplicación wiiZarra.

Al pulsar en conectar se pedirá al usuario que pulse los botones 1 y 2 del mando de la wii, los cuales permiten la conexión del mando a cualquier dispositivo *bluetooth* que detecte, se esperan unos segundos y cuando la aplicación nos valide, se desbloquearán los botones "Calibrar" y "Activar" como se puede observar en la Figura 4:



Figura 4: Detectando mando en wiiZarra.

Si pulsamos en el botón "Calibrar" aparecerá una ventana negra con cuatro cuadrados, cada uno en una esquina de la misma. Uno de ellos se encuentra parpadeando, para calibrar el mando no hay más que hacer uso de un dispositivo infrarrojo (preferente mente un bolígrafo de infrarrojos, aunque se ha hecho uso de una vela al no tener disponible dicho artilugio) y mover el ratón al cuadrado parpadeante hasta que éste deje de parpadear, en la *Figura 6* se puede observar la composición de la ventana de calibrado y en la *Figura 5* el ejemplo disponible en [4] de cómo construir nuestro propio bolígrafo de infrarrojos:

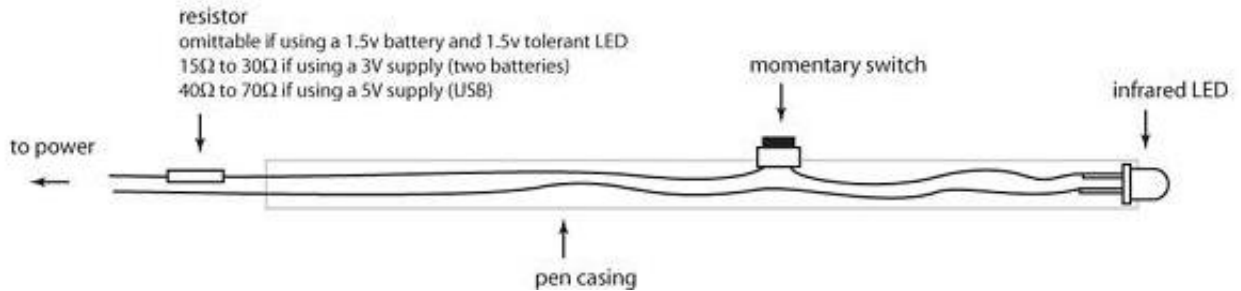


Figura 5: Bolígrafo casero de infrarrojos (para mover y pulsar el ratón).



Figura 6: Pantalla de calibrado del mando en wiiZarra.

Se va a proceder a explicar algunas partes del código referentes a la parte de calibrado del mando, puesto que éste consiste en poner el LED infrarrojo en el cuadrado parpadeante y esperar a que la circunferencia verde se rellene por completo (mientras mantienes el infrarrojo activo) y se active el siguiente cuadrado, básicamente en cada iteración del calibrado se llama a la siguiente función que se muestra en la *Figura 7*:



```
MENSAJE_DEBUG(2, "Ventana de calibrado: empezando ventana gtk");
Gtk::Main gtk_kit(0, 0);

MENSAJE_DEBUG(2, "Ventana de calibrado: cursor hilo wii");
empezar_thread_wiicursor(datos_thread);
sigc::connection redibujar_conexion = Glib::signal_timeout().connect(sigc::mem_fun(*this, &VentanaCalibracion::redibujar_conexion), 1000);

MENSAJE_DEBUG(2, "Ventana de calibrado: ventana de calibracion");
gtk_kit.run(*gtkmm_ventana);

MENSAJE_DEBUG(2, "Ventana de calibrado: calibrado acabado");
poner_estado_led(datos_thread.wiimotes.front().wiimote, WIIMOTE_LED_CONNECTED);
redibujar_conexion.disconnect();

bool const funciona = (datos_calibrado.punto_activo == WIIMOTE_NUM_PUNTOS_CALIBRADOS);
if (funciona)
{
    MENSAJE_DEBUG(1, "Calibrado funcionó, devolviendo los valores");
    p_wii = datos_calibrado.p_wii;
}
```

Figura 7: Código de pantalla de calibrado de wiiZarra.

Si nos fijamos en `sigc::connection redibujar_conexion`, cada vez que se rellena el círculo verde, se ponen los leds en estado conectado, se vuelve a redibujar los cuadrados para poner el siguiente a parpadear, y se pasan los valores obtenidos en el calibrado por la función en la variable `WIIMOTE_NUM_PUNTOS_CALIBRADOS`.

La forma en la que el mando se conecta a través del bluetooth con el ordenador la explicaremos en el apartado 6, ya que ese código se ha desarrollado para el trabajo.

5 Johnny Chung Lee y wiiMotion

En esta sección del trabajo vamos a hablar sobre los proyectos desarrollados por Johnny Chung Lee, disponibles en su blog de proyectos en [6], ya que hemos consultado dicha información para recoger ideas y realizar la aplicación para la asignatura, sobretodo me he basado en los 2 primeros proyectos disponibles en la página, "*Tracking your fingers with the wiiMote*" y "*Low-Cost Multi-point Interactive Whiteboards Using the Wiimote*".

El primero de los proyectos se centra en detectar el movimiento de los dedos haciendo uso del mando de la wii como sensor para captar éstos, para poder captar el movimiento se hace uso de un array de LED y una tapa refractiva, la cual con ayuda del mando de la wii permite captar el movimiento de los dedos como se puede observar en el video

Al no disponer del array de leds, el programa no detecta el movimiento de los dedos de la mano, pero en el video que hay disponible se ve claramente la funcionalidad del mismo.

El segundo de los proyectos es muy similar al programa que hemos comentado en el **punto 4** (wiiZarra), ya que se utiliza un bolígrafo infrarrojo fabricado por Johnny para mover el ratón utilizando para ello el mando de la wii que previamente se ha conectado de la misma manera que se realiza en nuestro trabajo, solo que para mover el ratón se utiliza el cursor infrarrojo del led, lo que permite unos movimientos más suaves y precisos, la forma de utilizar el puntero de infrarrojos la explicaremos en el punto 6 con más detalle.

El proyecto solo está disponible para windows de momento, pero hay total disponibilidad del código fuente y Johnny propone que alguien intente hacer el porte a Linux/Mac.

Por último el proyecto tiene un modo multitouch, que permite varios punteros a la vez y que se asimila a una *tablet*, la cual puede llegar a detectar los diez dedos de la mano (en este caso son bolígrafos infrarrojos, para que el mando de la wii los pueda detectar), el programa es capaz de detectar 4 bolígrafos infrarrojos al mismo tiempo.

No me ha sido posible ejecutar el programa, puesto que su compatibilidad es con Windows xp service pack 2 y se requiere de varias herramientas para poner en funcionamiento el bluetooth con el mando, pero en el video que acompaña la descripción del proyecto se puede observar completamente la funcionalidad del mismo.

Cabe decir que para el desarrollo de la aplicación del trabajo, he obtenido algunas partes del código de la pantalla multitouch, y del proyecto wiiZarra, ya que era muy similar a la funcionalidad que pretendía para el mismo, por lo que me ha servido en gran medida para iniciar la aplicación.



6 Aplicación wiiControllMD y su funcionamiento

En esta sección del trabajo se hablará sobre cómo se ha desarrollado la pequeña aplicación "wiiControllMD" y porqué se han hecho las cosas, y más que eso para qué, cabe decir que se ha desarrollado en el entorno linux, por lo que todo lo que se describe a continuación está basado en su funcionamiento en éste.

Lo primero que se va a comentar es todo lo necesario para poner en funcionamiento el programa así como la manera de compilar cada uno de los proyectos que se han creado, comentar que el proyecto completo lo forman tres ejecutables, que son "calibrar", "conexionWii" y "wiiControllMD".

Para compilar el primero de los ejecutables, son necesarias las librerías de OpenGL [9] que se pueden encontrar en el enlace que hay a pie de página ¹, ya que dicho ejecutable se basa en crear una especie de pantalla para comprobar el correcto funcionamiento del mando (similar a la pantalla de calibración del proyecto wiiZarra). El segundo de los ejecutables necesita de la librería cwiid para poder ser ejecutado, la cual se encuentra disponible en [2].

El último de los ejecutables no necesita de más librerías que las de c y c++, ya que simplemente se encarga de conectar con los diferentes módulos desarrollados.

Se va a pasar a explicar el funcionamiento de los módulos y las partes de código necesarias para entenderlos. El primero de los módulos "calibrar" se centra en crear una ventana que permite comprobar el funcionamiento del mando como ratón del ordenador, simplemente son cuatro cuadrados, uno en cada esquina de la pantalla, los cuales se encuentran en color blanco, el programa se encarga de que el usuario haga uso del mando de la wii haciendo click en cada uno de los cuadrados haciendo que estos se pongan en color rojo, cuando tenga los cuatro cuadrados en color rojo se termina la comprobación. El programa también devuelve las coordenadas de pulsado del ratón y el cuadrado pulsado por consola a modo de *feedback*, para futuros desarrollos sería interesante ampliar el contenido de dicho módulo y que diera más información y comprobara más usos del mando.

6.1 Módulo calibrar.cpp

Lo primero que se realiza para que funcione el módulo es su compilación, la cual se puede observar en la *Figura 8*, hay que tener en cuenta que hay que añadir a la línea de compilación las opciones para OpenGL y glut.

```
gcc calibrar.cpp -o calibrar -lglut -lGLU -lGL
```

Figura 8: Código de compilado para el módulo calibrar.

Una vez compilado podemos entrar a explicar cómo se realiza la detección de una pulsación dentro de uno de los cuadrados y el paso de coordenadas de los mismos, en la *Figura 9* se puede observar dicho fragmento de código:

```

if (state == GLUT_DOWN) {
    printf("Coordenada X --> %d\n", mouseX);
    printf("Coordenada Y --> %d\n", mouseY);
    if (button == GLUT_LEFT_BUTTON) {

        if(mouseX >= 50 && mouseX <= 100 && mouseY >= 37 && mouseY <= 70) {
            printf("Cuadrado izquierdo superior pulsado\n");
            glutDisplayFunc(cambiarColorIS);

        } else if(mouseX >= 900 && mouseX <= 950 && mouseY >= 37 && mouseY <= 70) {
            printf("Cuadrado derecho superior pulsado\n");
            cambiarColorDS();

        } else if(mouseX >= 50 && mouseX <= 100 && mouseY >= 630 && mouseY <= 665 ) {
            printf("Cuadrado izquierdo inferior pulsado\n");
            cambiarColorII();

        } else if(mouseX >= 900 && mouseX <= 950 && mouseY >= 630 && mouseY <= 665 ) {
            printf("Cuadrado derecho inferior pulsado\n");
            cambiarColorDI();

        }

    }

}

```

Figura 9: Fragmento de código del módulo calibrar.

Ya que pretendemos realizar una especie de comprobación de las correctas coordenadas del ratón, el programa se encarga de comprobar que cuando se haga click mediante el uso del mando de la wii, se encuentre entre ciertas coordenadas (previamente definidas) donde se encuentran posicionados los cuadrados, si el click se produce dentro de dichas coordenadas, el cuadrado cambiará de color (pasará de ser blanco a ser rojo), de éste modo podemos probar las opciones que nos ofrece el mando de la wii con un sencillo programa.

A continuación en la Figura 10, se muestra un pequeño fragmento de creación de uno de los cuadrados (en concreto el que se encuentra en la esquina izquierda superior de la pantalla:

```

/*
 *Cuadrado situado en la esquina superior izquierda.
 */
glBegin(GL_POLYGON);
    glVertex3f (0.05, 0.95, 0.0);
    glVertex3f (0.10, 0.95, 0.0);
    glVertex3f (0.10, 0.90, 0.0);
    glVertex3f (0.05, 0.90, 0.0);
glEnd();

```

Figura 10: Fragmento de código del módulo calibrar para dibujar cuadrado.

Simplemente se realiza un polígono de cuatro vértices, los cuales se colocan en las coordenadas izquierda superior de la pantalla (para más información sobre construcción de polígonos con



UNIVERSIDAD
POLITECNICA
DE VALENCIA

OpenGL consultar [8]). Por último mostramos el resultado de la ejecución del programa en la siguiente imagen, *Figura 11*:

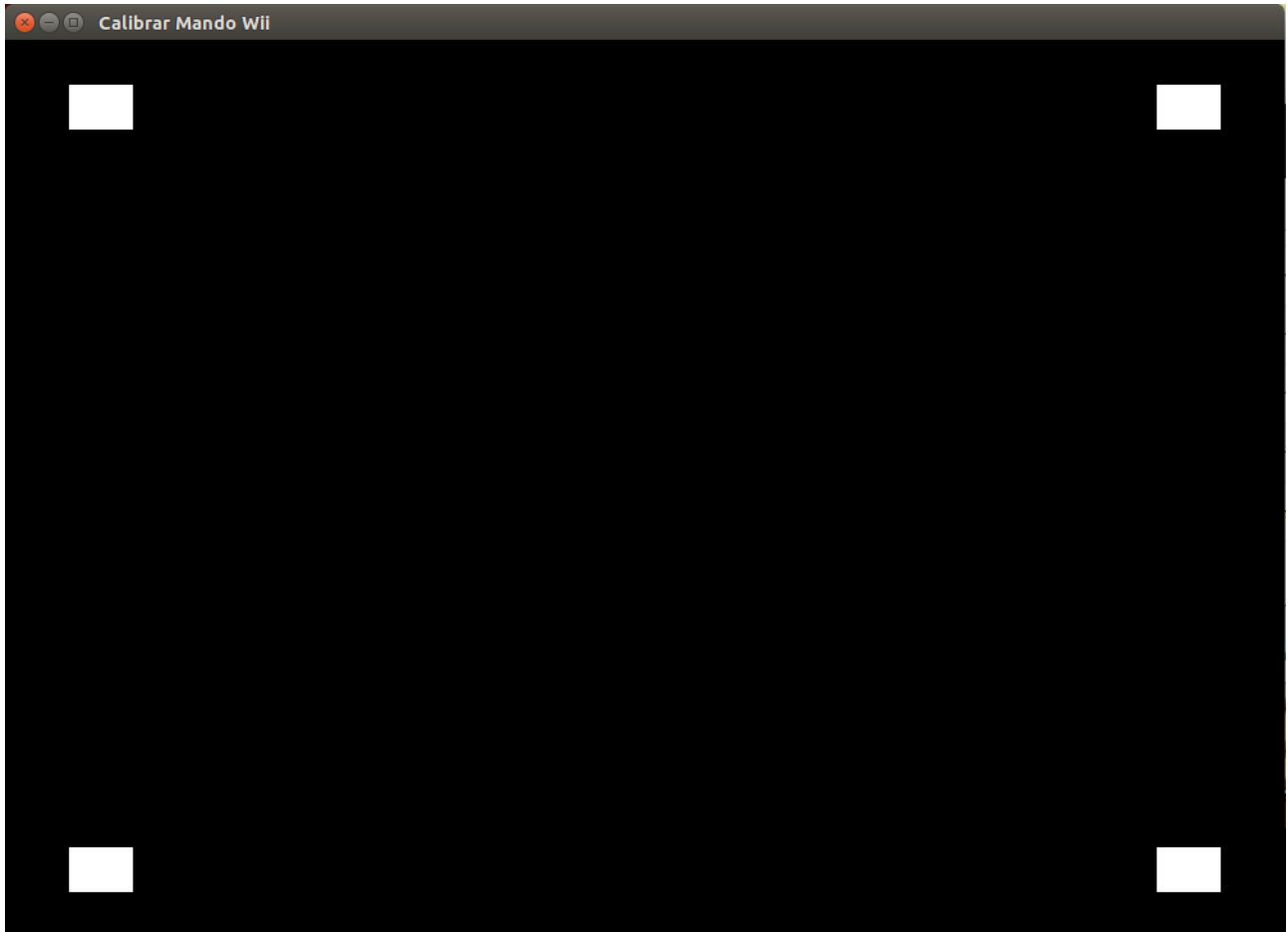


Figura 11: Programa calibrar ejecutado.

6.2 Módulo conexionWii.cpp

El siguiente de los módulos requiere del uso de las librerías de cwiid para su compilación, ya que se basa en conectarse al mando de la wii, y comprobar la correcta conexión encendiendo cada uno de los cuatro leds que dispone el mando, y al escribir `exit`, cerrar la conexión con el mismo, en la Figura 12 se puede observar la línea de compilación del módulo "conexionWii":

```
gcc conexionWii.cpp -o conexionWii -lcwiid -fpermissive -lbluetooth
```

Figura 12: Código de compilado para el módulo conexionWii.

Hay que tener en cuenta que como se explicó en el **punto 4**, `-fpermissive` es necesario por el trato especial que se le da a la variable `BDADDR_ANY`, y `-lcwiid` junto a `-lbluetooth` son las librerías necesarias para el reconocimiento del mando de la wii.

Una vez compilado se puede pasar a explicar algunas partes del código, ya que es interesante para entender el funcionamiento y el trato que se le da en el ordenador al módulo bluetooth que utilizamos para reconocer el mando. Lo primero a tener en cuenta es el siguiente fragmento de código que vamos a mostrar en la Figura 13, en el que se realiza todo lo referente a la conexión con el mando de la wii:

```
/* Si se le pasa directamente la dirección del mando se conecta. */
if (argc > 1) {
    str2ba(argv[1], &bdaddr);
}
else {
    bdaddr = *BDADDR_ANY;
}

/* Función por la cual nos conectamos al dispositivo wii */
printf("Put Wiimote in discoverable mode now (press 1+2)...\\n");
if (!(wiimote = cwiid_open(&bdaddr, 0))) {
    fprintf(stderr, "Unable to connect to wiimote\\n");
    return -1;
}
```

Figura 13: Código de reconocimiento del mando de la wii.

Podemos observar en la Figura 13 el funcionamiento del código de reconocimiento del mando de la wii, simplemente lo que se hace es comprobar si se pasa algún argumento por consola (la dirección MAC del mando de la wii, que más tarde explicaremos como obtenerla), si no se le da ninguna entonces utilizamos la variable del módulo -bluetooth `BDADDR_ANY` la cual a través del dispositivo bluetooth conectado al ordenador, busca cualquier otro dispositivo conectado a él, y lo guardamos en la variable `bdaddr`.

Una vez detectado el dispositivo, se comprueba que no existe otra conexión abierta, y se le da una dirección a la función `cwiid_open`, la cual se encarga de conectarse al mando de la wii, una vez conectados se obtiene un menú en el cual se pueden encender las luces led del mando y por último podemos salir del programa escribiendo `exit`, en la Figura 14 se puede observar el código de encendido de las luces led:



```
while (!exit) {  
    switch (getchar()) {  
        case '1':  
            toggle_bit(led_state, CWIID_LED1_ON);  
            set_led_state(wiimote, led_state);  
            break;  
        case '2':  
            toggle_bit(led_state, CWIID_LED2_ON);  
            set_led_state(wiimote, led_state);  
            break;  
        case '3':  
            toggle_bit(led_state, CWIID_LED3_ON);  
            set_led_state(wiimote, led_state);  
            break;  
        case '4':  
            toggle_bit(led_state, CWIID_LED4_ON);  
            set_led_state(wiimote, led_state);  
            break;  
        case '5':  
            exit = -1;  
            break;  
        default: break;  
    }  
}
```

Figura 14: Código de reconocimiento del mando de la wii.

Una vez conectado al mando de la wii, pulsando uno, dos, tres o cuatro en el teclado encenderemos cada una de las luces del mando de la wii mediante el uso de la función `set_led_state` de la librería `cwiid`, la cual se encarga del manejo de las luces led del mando, simplemente pasando como parámetros la dirección del mando y el estado de los led del mando.

6.3 Módulo wiiControlIMD.cpp

Por último se va a explicar el módulo central del desarrollo, el cual se encarga de dirigirnos a cada una de las funcionalidades desarrolladas para la asignatura, para compilar dicho programa simplemente es necesario tener la librerías de c y c++ para linux, y escribir la siguiente línea de compilación en la consola, disponible en la *Figura 15*:

```
gcc wiiControlIMD.cpp llamadaswii.h llamadaswii.cpp -o wiiControlIMD
```

Figura 15: Código de compilado para el módulo wiiControlIMD.

Una vez compilado pasamos a explicar la funcionalidad y algunas partes del código, como se puede observar en la *Figura 16*, el programa simplemente se basa en un menú, el cual nos dirige a las 6 funcionalidades que se han desarrollado, las cuales se van a explicar con detalle a continuación:

```
printf("\n***APLICACIÓN wiiControlIMD by Samuel Villaescusa Vinader IMD***\n");

printf("\n***MENÚ***\n");
printf("1--> Reconocer mando de la wii\n");
printf("2--> Probar el funcionamiento del mando mediante leds\n");
printf("3--> Usar el mando de la wii para controlar el ratón\n");
printf("4--> Probar el funcionamiento del mando con un pequeño juego(calibrar)\n");
printf("5--> Pasar transparencias con el mando de la wii\n");
printf("6--> Juego Pong(Desarrollado mediante Scratch)\n");

printf("Elija una opción(pulse 9 para salir):\n");

scanf("%d", &opcion);

menu(opcion);
```

Figura 16: Menú de la aplicación wiiControlIMD.

Antes de comenzar la explicación de cada una de las seis funcionalidades, cabe comentar que la aplicación se pretendía desarrollar mediante el uso de gtkmm bajo una interfaz de usuario, pero debido al poco tiempo disponible y al poco conocimiento sobre la librería, se terminó decidiendo el realizarlo bajo un menú básico de consola, sin más que añadir pasamos a explicar cada una de las funcionalidades:

1. La primera opción disponible del programa es la de reconocer los dispositivos bluetooth que hay dentro del radio del pen que se dispone para la detección (BELKIN bluetooth), para ello se hace uso de un módulo de la librería cwiid para reconocer dispositivos bluetooth, **hcitool scan**, con el que obtendremos la dirección MAC de nuestro dispositivo wii, en la *Figura 17* podemos observar el resultado de seleccionar la primera funcionalidad:



```
***MENÚ***
1--> Reconocer mando de la wii
2--> Probar el funcionamiento del mando mediante leds
3--> Usar el mando de la wii para controlar el ratón
4--> Probar el funcionamiento del mando con un pequeño juego(calibrar)
5--> Pasar transparencias con el mando de la wii
6--> Juego Pong(Desarrollado mediante Scratch)
Elija una opción(pulse 9 para salir):
1
Scanning ...
00:1E:35:18:9D:C9          Nintendo RVL-CNT-01
```

Figura 17: Reconocimiento del mando wiiControl.

2. La segunda de las opciones permite realizar la conexión con el mando de la wii como se ha descrito en la sección 6.2 del módulo conexionWii, y una vez conectados permite comprobar su funcionamiento con un sencillo menú que se muestra en la Figura 18 el cual permite encender las luces leds del mando de la wii, para ello se muestra una imagen del mando de la wii una vez conectado, con las luces leds apagadas en la Figura 19, y el mando wii con la luz 1 del led encendida en la Figura 20 (una vez se elige la opción 1 del menú de encendido de los leds):

```
***MENÚ***
1--> Reconocer mando de la wii
2--> Probar el funcionamiento del mando mediante leds
3--> Usar el mando de la wii para controlar el ratón
4--> Probar el funcionamiento del mando con un pequeño juego(calibrar)
5--> Pasar transparencias con el mando de la wii
6--> Juego Pong(Desarrollado mediante Scratch)
Elija una opción(pulse 9 para salir):
2
Put Wiimote in discoverable mode now (press 1+2)...
1: toggle LED 1
2: toggle LED 2
3: toggle LED 3
4: toggle LED 4
5: exit
```

Figura 18: Menú para encendido de leds del mando wii.



Figura 19: Mando de la wii sin luces leds encendidas.

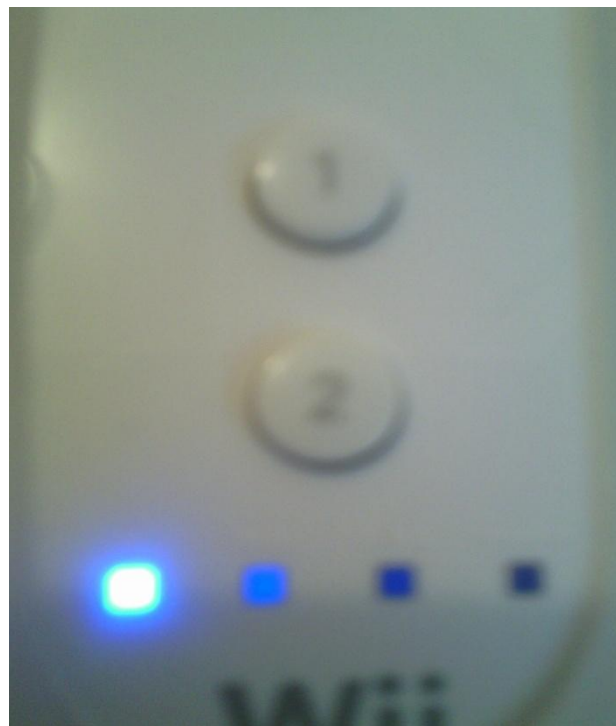


Figura 20: Mando de la wii con el led 1 encendido.



3. La tercera de las opciones del menú permite conectar el mando mediante la librería `cwiid` al ordenador, y mediante el uso del módulo `wminput` permite usar el acelerómetro para mover el mando de la wii y usar el botón A para hacer click izquierdo con el ratón y el botón B para hacer click derecho.

El módulo `wminput` se puede ejecutar por defecto como `sudo wminput`, pero también permite el paso de parámetros de un fichero `.conf`, en el cual se indica la funcionalidad que se desea proporcionar al mando de la wii, de la siguiente manera:

`sudo wminput -c` y el nombre del archivo de configuración.

Hay que decir que para que la librería detecte los archivos de configuración es necesario crear un directorio denominado `.CWiid`, en el cual se guardarán los archivos de configuración.

En la *Figura 21* se puede observar un ejemplo de archivo de configuración, en el que se le dice al módulo que deseamos detectar el acelerómetro del mando como movimiento del ratón (es necesaria una fuente de infrarrojos para probarlo), el botón A como click izquierdo y el botón B como click derecho:

```
# wiimote(Comando)   Tecla Pulsada

include buttons      #Se carga la configuración de los botones.

Plugin.ir_ptr.X = ABS_X   #Eje de las X, necesario sensor infrarrojos.
Plugin.ir_ptr.Y = ABS_Y   #Eje de las Y

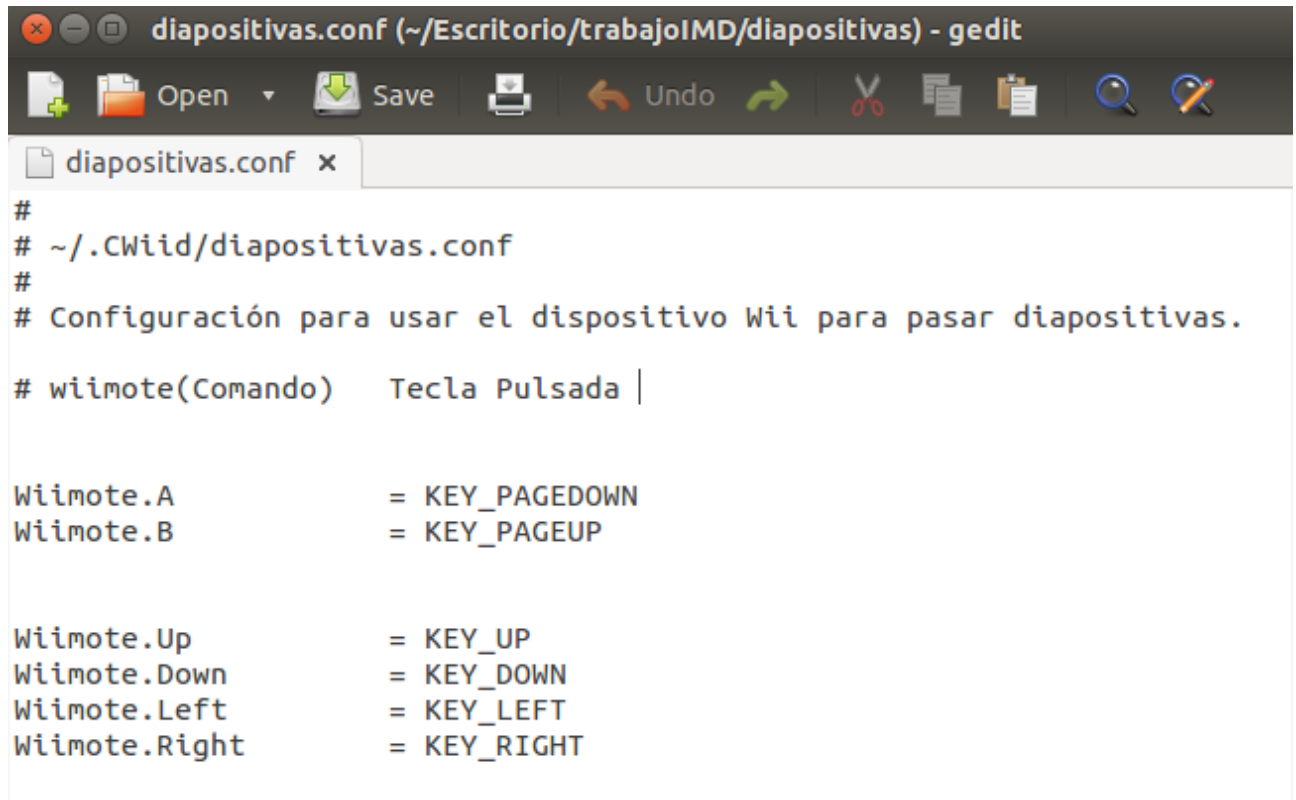
Wiimote.A           = BTN_LEFT
Wiimote.B           = BTN_RIGHT
```

Figura 21: Fichero de configuración del mando de la wii (infrarrojos).

Se han usado dos velas para simular los sensores infrarrojos, una a cada lado del ordenador, la calidad con la que se recogen los movimientos es bastante pobre, por lo que se recomienda el uso de un bolígrafo infrarrojo, ya que la potencia concentrada en la pantalla permite un mejor uso del módulo.

Centrándonos en el contenido del fichero de configuración, si nos fijamos en la línea `Plugin.ir_ptr`, podemos ver que se trata de la función encargada del manejo del acelerómetro para detectar el movimiento del mando mediante un sensor de infrarrojos, y un poco más abajo se encuentra `Wiimote.A` y `B`, que son las encargadas del manejo de los botones A y B del wiiMote (mando de la wii), podemos hacer uso de cualquiera de los botones mediante `Wiimote.(botón deseado)`.

4. La cuarta de las opciones es la que se encarga de abrir el programa de calibrado para el mando de la wii que se ha presentado en el punto 6.1, y que no comentaremos ya que en dicho apartado viene definida toda su funcionalidad.
5. La quinta opción permite darle un uso al mando de la wii para pasar las transparencias de cualquiera de los programas dedicados a esto (como puede ser PowerPoint), para ello hace uso del módulo wminput una vez más pero en ésta ocasión utiliza el fichero de configuración diapositivas.conf, el cual se puede observar en la Figura 22:



```
#
# ~/.CWiid/diapositivas.conf
#
# Configuración para usar el dispositivo Wii para pasar diapositivas.
# wiimote(Comando) Tecla Pulsada |

Wiimote.A          = KEY_PAGEDOWN
Wiimote.B          = KEY_PAGEUP

Wiimote.Up         = KEY_UP
Wiimote.Down       = KEY_DOWN
Wiimote.Left       = KEY_LEFT
Wiimote.Right      = KEY_RIGHT
```

Figura 21: Fichero de configuración del mando de la wii (diapositivas).

6. La sexta y última de las opciones del menú permite al usuario probar la funcionalidad del módulo mediante el fichero de configuración pong.conf que se puede observar en la Figura 22, para jugar a un sencillo juego que se ha desarrollado con la plataforma Scratch [10] y gracias al uso del mando de la wii podemos manejar una de las raquetas con la cruceta del mando (el juego se encuentra disponible en el enlace [11]).

```
# wiimote(Comando) Tecla Pulsada

Wiimote.Up         = KEY_UP
Wiimote.Down       = KEY_DOWN
```

Figura 22: Fichero de configuración del mando de la wii (pong).



7 Conclusión

Para concluir cabe decir que se han alcanzado todos los objetivos propuestos, aunque se disponía de muy poco tiempo y algunos puntos que ahora se enumeraran se podrían haber desarrollado con más detenimiento, aunque es perfecto para plantear posibles mejoras futuras.

A continuación se van a enumerar los objetivos propuestos para el proyecto y cómo se han conseguido completar satisfactoriamente:

1. El primero de los objetivos consistía en compilar el proyecto wiiZarra de un alumno de la asignatura, y explicar su funcionamiento. Se ha cumplido el objetivo, mejorando la explicación que se ofrecía para compilar el programa y explicando la parte que nos ha servido de ejemplo para calibrar el mando.
2. El segundo objetivo consistía en buscar información acerca de los proyectos desarrollados por Johnny Chung Lee, que se pueden encontrar en [6] y explicarlos un poco, se ha completado buscando información en su blog y observando los videos de ejemplo.
3. El tercero de los objetivos era conseguir que el ordenador reconociera el mando de la wii a través de un dispositivo bluetooth, el cual se ha logrado gracias al uso de la librería cwiid y de la función hcitool scan.
4. El cuarto objetivo que consistía en conectar y probar el mando de la wii a través del pc y de la librería cwiid, se ha conseguido haciendo uso del api de la misma y de las funciones encontradas en los proyectos de Johnny y wiiZarra.
5. El quinto de los objetivos que consistía en crear una pequeña pantalla para probar y calibrar el mando de la wii, se ha completado con el desarrollo de un programa, el cual se ha descrito en el apartado 6.1 de la memoria.
6. El sexto pedía al usuario que se conectara mediante el uso del mando de la wii, y para probar el correcto funcionamiento se encienden los cuatro leds del mando, se han alcanzado a resolver mediante el uso de la librería cwiid y las funciones de los leds.
7. El séptimo y octavo de los objetivos consistía en conseguir poner en movimiento el ratón del ordenador mediante el uso del mando de la wii, se ha conseguido mediante el uso del módulo wminput y los ficheros de configuración (se ha conseguido mover el cursor mediante el uso de una fuente infrarroja con velas, y el uso de la configuración estándar del módulo) descritas en el apartado 6.3, en el punto 3.
8. El noveno objetivo consistía en pasar las transparencias mediante el uso del mando de la wii, se ha conseguido resolver mediante el uso de los botones A y B, y el uso de la cruceta. Aunque en un principio se pretendía resolver con un movimiento del mando, al no ser lo suficientemente preciso no se ha conseguido.
9. El último objetivo consistía en desarrollar un pequeño juego para probar la usabilidad del mando de la wii a través del pc (Pong), al principio se pretendía dar movimiento a una de las paletas del juego mediante el uso del acelerómetro, pero debido a la mala calidad de referencia de la señal, ya que se han usado velas, se ha decidido configurar el archivo mediante el uso de la cruceta del mando de la wii.

Para terminar decir que me ha resultado muy interesante la investigación sobre el uso de la wii, y aunque se disponía de muy poco tiempo se han alcanzado todos los objetivos dispuestos en la memoria, quedan abiertos el módulo de "calibrar.c" para mejorar su funcionalidad y conseguir que el mando sea lo suficientemente preciso en su uso del acelerómetro para poder darle mayor funcionalidad.



8 Bibliografía

8.1 Referencias de fuentes electrónicas:

- [1] Gtkmm. Librería que permite el desarrollo de aplicaciones con una interfaz de usuario para linux. Disponible en: <http://www.gtkmm.org/en/>
- [2] Cwiid. Librería para el desarrollo de aplicaciones compatibles con wiiMotion para el lengua C en linux. API disponible en: <http://abstrakraft.org/cwiid/wiki/libcwiid> y descarga de la librería en <http://abstrakraft.org/cwiid/wiki/libcwiid>
- [3] The Code::Blocks Team. IDE (Entorno de desarrollo) para desarrollar en varios lenguajes, tanto para Windows, Linux y Mac. Disponible en: <http://www.codeblocks.org/downloads/26>
- [4] Javier Martí Monforte. Aplicación wiiZarra para linux, consultado 20/03/2015. Disponible en: <http://web-sisop.disca.upv.es/~fsmm/projectes/2k9-2k10/wiiZarra/trabajo.xml>
- [5] Javier Martí Monforte. Aplicación wiiDrums para Visual Studio, consultado 20/03/2015. Disponible en: <http://web-sisop.disca.upv.es/~fsmm/projectes/2k9-2k10/wiiZarra/trabajo2k8.xml>
- [6] Johnny Chung Lee. Proyectos sobre el uso de wiiMotion, consultado 06/04/2015. Disponible en: <http://johnnylee.net/projects/wii/>
- [7] Johnny Chung Lee. Página de currículum y proyectos, consultado 08/04/2015. Disponible en: <http://johnnylee.net/>
- [8] OpenGL. Muchas fuentes de consulta para trabajar con OpenGL. Disponible en: <https://www.opengl.org/documentation/>
- [9] OpenGL. Getting Started. Disponible en: https://www.opengl.org/wiki/Getting_Started
- [10] Scratch. Página de registro y consulta de proyectos. Disponible en <https://scratch.mit.edu/>