PROJECTE FINAL DE CARRERA

Control Remot del Robot Humanoide Kyosho AT01

(Remote Control for a Kyosho Manoi Humanoid Robot)

Estudis: Enginyeria Electrònica

Autor: Cristóbal Pio Garcia

Director/a: Sergi Bermejo

Any: 2015

Índex General

Agraïments

Agraeixo al tutor del projecte Sergi en primer lloc per la oportunitat que em va oferir per a realitzar el projecte final de carrera per les correccions que han ajudat a fer créixer i millorar la versió final del projecte i també per la paciència amb que ha tractat la meva manca de temps.

Agraeixo també l'ajuda de tota la meva família, no només gracies al seu suport econòmic dels primers anys, sinó també a la seva demostració diària de l'esforç, tot això m'ha permès arribar fins a la finalització del projecte.

Finalment només em quedar agrair a qui m'ha recolzat en el dia a dia, Gina, sense tu no hagués estat possible.

Resum del Projecte

Durant els últims anys han estat desenvolupats i comercialitzats múltiples dispositius personals que augmenten l'experiència de l'usuari quan participa amb l'entorn que l'envolta. Aquests dispositius han estat pensats per ser utilitzats en camps molt concrets com el cas del Wiimote o el Kinnect, ideats exclusivament dins del camp dels videojocs i d'altres han estat pensats per aplicacions en telefonia mòbil.

Tots aquests dispositius fan servir una sèrie d'eines com sensors, pantalles tàctils o botons, per saber de quina manera vol l'usuari interactuar amb el medi. Així, per exemple, amb un telèfon mòbil podem girar la pantalla per mostrar una foto en la pantalla de manera òptima en funció de la seva orientació espaial o amb el Wiimote podem mesurar un moviment a l'espai per tal de fer moure un element d'un videojoc. Aquests dispositius també ofereixen interfícies de comunicacions que permeten accedir a les dades –tant dels sensors com de les accions de l'usuari sobre botons o pantalles tàctils– que permeten potencialment realitzar qualsevol tractament que es vulgui realitzar sobre elles.

Paral·lelament, el món de la robòtica està cada dia més desenvolupat: els robots no només estan cada cop més integrats en la vida domèstica; resulta habitual disposar en les llars de robots que faciliten certes tasques quotidianes com ara els del tipus escombra o tallagespes. Addicionalment, dia a dia, s'estén més la cultura robòtica ja que el nombre de competicions entre robots creats o programats per usuaris és creixent, al igual que les empreses que es dediquen a la creació de robots programables que suporten diferents moviments i tenen ports generals d'entrada i sortida per la interacció amb l'entorn.

El projecte que presentem a continuació pretén aprendre sobre el dos mons descrits anteriorment –el dels robot si el dels dispositius mòbils– i tractar de fusionar-los. Per això, es descriurà i s'explicarà el funcionament de dos exponents d'ambdós mons: en primer lloc, un Kyosho Manoi que es tracta d'un robot humanoide desenvolupat per l'empresa Kondo i, posteriorment alguns dels dispositius d'interacció amb l'usuari disponibles al mercat. Per a possibilitar aquesta fusió s'ha desenvolupat un software que permetrà donar una alternativa al del fabricant per la interacció de l'usuari amb el robot, augmentant la seva experiència.

Resumen del Proyecto

Durante los últimos años se han desarrollado y comercializado múltiples dispositivos personales que aumentan la experiencia del usuario cuando interactúa con el entorno que lo rodea. Estos dispositivos han sido pensados y diseñados para ser utilizados en diferentes sectores, unos como es el caso de Wiimote o Kinnect para los videojuegos y otros como la telefonía móvil.

Todos estos dispositivos utilizan una serie de herramientas como sensores, pantallas táctiles o botones, para saber de qué manera quiere el usuario interactuar con el entorno; así en un teléfono móvil podemos saber cuál es su orientación para girar una foto adecuadamente o con el Wiimote podemos medir un movimiento en el espacio para realizar un movimiento en un elemento de un videojuego. Estos dispositivos además ofrecen interfaces de comunicaciones que permiten acceder a los datos tanto de los sensores como de las acciones que realiza el usuario sobre botones o pantallas táctiles y permiten potencialmente realizar cualquier tratamiento que se quiera realizar sobre ellos.

Paralelamente se han producido avances significativos en el desarrollo de los robots de manera que no sólo están cada vez más integrados en la vida doméstica –en la actualidad es habitual disponer de robots que, por ejemplo, barren o que cortan el césped –sino que, día a día, se extiende la cultura de la robótica. Un ejemplo de ello es la existencia de un creciente número de competiciones entre robots creados y/o programados por usuarios, ya que también son cada vez más las empresas que se dedican al desarrollo de robots que permiten a los usuarios programar sus movimientos y añadir sensores permitiendo su interacción con el entorno.

El proyecto que presentamos a continuación pretende aprender sobre estos dos sectores descritos anteriormente –el de los robots humanoides y el de los dispositivos móviles– y tratar de fusionarlos. Se describirá el funcionamiento de dos exponentes de estos sectores: en primer lugar, el denominado Kyosho Manoi, un robot humanoide desarrollado por la compañía Kondo y, posteriormente, algunos de los dispositivos de interacción con los usuarios disponibles en el mercado. Para posibilitar dicha fusión se ha desarrollado un software para la interacción del usuario con el robot que servirá de alternativa al proporcionado por el fabricante, permitiendo aumentar la experiencia del usuario en dicha interacción.

Abstract

Over the last years many personal devices for increasing users experience when it interacts with the environment have been developed and marketed. These devices have been designed and developed for being used in very different fields, e.g. Wiimote or Kinnect in videogames and other for more generic fields like mobile phones.

All these devices use a number of tools (sensors, touchable screens or buttons) that let to know how the user wants to interact with the environment; hereby we can change the orientation of the screen knowing which is the orientation of the device with a mobile phone or we can measure an spatial movement to move an element from a videogame with the Wiimote. These devices also offer some communication interfaces that allow acceding all sensor data and user actions on device to make any special process.

At the same time, robotics is getting growing day by day. Robots are more integrated in our daily life and, accordingly, it is very common nowadays the use of robots for domestic tasks like sweeping or mowing. Additionally, robotic culture extends; it is usual to see competitions between robots that have been developed or programmed robots by users. Consequently, more companies offer many kinds of robots that let users program their movements or add sensors to let robots interact with the environment.

The project described below is intended to learn from the two fields described above –the humanoid robots with mobile devices– and try to merge them. The performance of some of the elements in both fields will be described: on one hand, the so-called Kyosho Manoi, a humanoid robot developed by the Kondo Company and, on the other, some of the personal devices that can be found. A software tool has been developed for achieving this aim, which will offer an alternative to the use of the manufacturer control application and will improve the user experience with the robot interaction.

1. Introducció

1.1. Objectius

El principal objectiu d'aquest projecte final de carrera és obtenir les dades dels sensors o actuadors de diferents **capturadors de dades** i estudiar com fusionar-les amb un Robot Kyosho Manoi per tal s'hi puguin afegir noves funcionalitats.

Principalment el projecte té tres objectius:

1- L'estudien profunditat del funcionament del robot **Kyosho Manoi**, una revisió de la descripció física amb els elements que el conformen i la revisió del software ofert per el fabricant permetrà conèixer quines son les opcions que ofereix el robot en quant el moviment. Seguidament l'estudi d'alguns dels sensors que es poden connectar amb el robot permetrà conèixer com el robot pot intercanviar informació amb el medi i adaptar-s'hi. Finalment, respecte a l'estudi del robot, entendre la connexió entre el PC i el robot permetrà conèixer de quina manera s'envien comandes a la placa des de el PC i com respon la placa, així amb aquesta informació es podrà escriure el software per tal que es pugui utilitzar en altres plataformes.

2- L'estudi de cadascun dels capturadors de dades seleccionant, prenent especial atenció amb les dades que ens ofereixen i amb les interfícies que exporten per tal d'arribar a aquestes dades i com tractar aquestes dades per tal d'obtenir la orientació espaial del capturador de dades.

3- Fusionar les dades capturades amb el robot per tal d'afegir-hi noves funcionalitats.

1.2. Estructura de la memòria

La memòria ha estat estructurada en dos parts molt diferenciades, la primera part, més teòrica està centrada en l'estudi del hardware que permetrà descobrir quins son els elements principals que son utilitzats en el projecte, inclou:

- Els principals sensors i actuadors que seran utilitzats en el projecte.
- El robot **Kyosho Manoi,** es tractarà d'un intent de revisió profunda del robot i les seves capacitat. A partir del software del fabricant anirem desgranant una per una quines son totes les funcions que ofereix el robot i la seva placa de connexió **RCB3J**.
- Els capturadors de dades, amb els que estudiarem de quina manera es poden capturar les dades que ofereixen en funció de la seva interfície de comunicacions i a més de quina manera es podrien tractar aquestes dades per calcular posicions i conèixer com moure el robot.

Un cop vistos tots aquests elements hi ha una part més practica en la que descriurem de quina manera ha estat creat el programari bàsic que acompanya aquest projecte, que bàsicament es pot desgranar en:

- Descripció del mòdul de connexió amb el robot Kyosho Manoi
- Descripció del mòdul de connexió amb els capturadors de dades
- Descripció del mòdul de tractament de dades

Amb tota aquesta informació com a punt final es descriuran un parell d'aplicacions que han estat desenvolupades per tal de fusionar les possibilitats que ofereix el robot amb les dades que extraiem dels diferents capturadors de dades donant per finalitzat el projecte i esperant haver superat els objectius marcats.

2. Sensors i actuadors

Existeixen un gran numero de sensors que s'utilitzen en robots mòbils, alguns s'utilitzen per obtenir informació interna del robot, com la temperatura de l'electrònica o la velocitat de gir dels motors, altres més sofisticats s'utilitzen per obtenir informació del medi. Els sensors utilitzats en els robots es poden classificar en dos eixos funcionals, **propioreceptors/exteroreceptors** i actius/passius.

Es presenta a continuació una taula amb els sensors més utilitzats en robòtica[1] i la seva classificació:

Classificació General (us típic)	Sensor	PC o EC	Actiu o Passiu
Sensors Tàctils (detecció de proximitat o de contacte)	Interruptors de Contacte	EC	Passiu
	Barreres Òptiques	EC	Actiu
	Sensors de Proximitat	EC	Actiu
Sensors Motors (Velocitat de rodes i motors)	***Encoders*** òptics	PC	Actiu
	Encoders Magnètics	PC	Actiu
	Encoders Inductius	PC	Actiu
	Encoders Capacitius	PC	Actiu
Sensors de rumb (Orientació en relació a una referència fixa)	Acceleròmetres	EC	Passiu
	Giroscopis	PC	Passiu
	Brúixoles	EC	Actiu/Passiu
Balises Terrestres (Orientació dins una referència fixa)	GPS	EC	Actiu
	Balises Òptiques	EC	Actiu
	Balises ultrasòniques	EC	Actiu
	Balises Reflectives	EC	Actiu
Sensors de Moviment i Velocitat	Radar d'efecte Doppler	EC	Actiu
	So Doppler	EC	Actiu
Sensors de visió	CCDS - Càmeres	EC	Passiu

Taula 1. Tipus de Sensors.

Els sensors **propioreceptors** mesuren valors interns dels robot, per exemple la velocitat dels motor, la posició de les articulacions o la carrega de la bateria.

Els sensors **exteroreceptius** obtenen informació del medi amb el que el robot interactua, per exemple la distància, la intensitat de la llum o l'amplitud d'un so.

Els sensors **passius** mesuren energia provinent del medi, exemples d'aquest tipus de sensors son els micròfons, els CCDS de les càmeres o les sondes de temperatura.

Els sensors **actius** emeten energia al medi i aleshores mesuren la reacció del medi a aquesta energia. Com els sensors actius poden tenir una interacció més controlada amb el medi normalment son més eficients, per desgràcia també introdueixen més riscos, per exemple que l'energia emesa afecti a la mesura o que hi hagin interferències amb altres sensors[2].

De tots aquests sensors es veurà una referència ràpida als utilitzats en aquest projecte, bàsicament dels sensors que poden establir l'orientació d'un objecte respecte a una referència fixa.

2.1. Acceleròmetres

Els acceleròmetres son sensors que proporcionen una sortida proporcional a l'acceleració, vibració o xoc a la que estan sotmesos. Aquests sensors s'utilitzen en una gran varietat d'aplicacions, tant de recerca i desenvolupament com d'us diari. En la recerca és pot remarcar el seu ús en l'anàlisi modal o les proves de soroll, vibració i duresa, en els usos més prosaics els acceleròmetres son molt utilitzats en els coixins de seguretat i les alarmes de seguretat dels automòbils[3].

Existeixen molt tipus d'acceleròmetres en funció del principi físic aplicat per mesurar la força aplicada sobre el sensor:

- Acceleròmetres Piezoelèctrics
- Acceleròmetres Capacitius
- Acceleròmetres Piezoresistius

Sigui quin sigui l'efecte físic en el és basa l'acceleròmetre, sempre es pot imaginar com una caixa amb parets sensitives a la pressió amb una bola metàl·lica al mig.

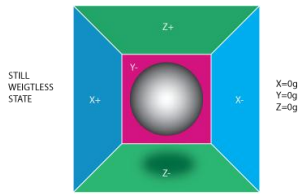


Figura 1. Dibuix simulat de l'interior d'un acceleròmetre sense força exterior aplicada.

L'acceleració i la força a la que està sotmès un objecte estan fortament relacionades, com ensenya la segona llei de Newton, això implica que qualsevol força que actuï sobre l'acceleròmetre serà mesurada i traduïda en dades, per exemple la Força de la gravetat, així un acceleròmetre en repòs sobre una superfície plana mesurarà un total de 9.8 m/s^2sobre l'eix perpendicular a l'acció de la força de la gravetat.

Quan cap força actua sobre la caixa la bola es manté flotant sense tocar cap de les parets que formen la caixa però si actua una força la bola pressionarà la paret en direcció a la força aplicada i ens generarà una senyal que podem utilitzar per calcular la força a la que està sotmès l'acceleròmetre.

En estat de repòs (sobre un pla) l'única força que actua sobre l'acceleròmetre és la força de la gravetat, així mantenint la caixa sobre el pla obtindrem la següent força:

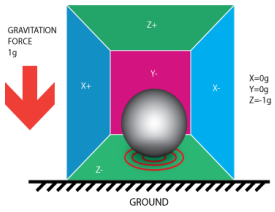


Figura 2. Dibuix simulat de l'interior d'un acceleròmetre sotmès a una força +g.

Que ens equivaldrà a un vector: (+g,0,0)

Així girant l'acceleròmetre sobre cadascun dels seus eixos s'obtenen diferents vectors, sobre l'eix y (0,+g,0) i sobre l'eix z (0,0,+g).

Principalment presenten les següents desavantatges:

- Molt soroll
- Son lents
- No podem destriar la força de la gravetat de les altres forces a las que estan sotmesos.

2.2. Giroscopis

Els giroscopis son sensors de rumb que preserven la seva orientació en relació a un eix de referència fixa, així proporcionen una mesura absoluta de la posició d'un sistema mòbil[4]. Bàsicament son sensors que mesuren la velocitat angular a la que estan sotmesos en algun dels tres eixos, integrant la velocitat de rotació és possible saber quina és la rotació aplicada sobre cadascun dels eixos i per tant conèixer quina és la rotació total a la que ha estat sotmesa el sensor.

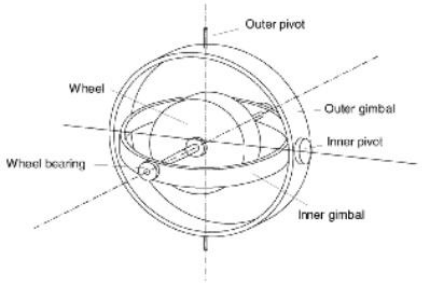


Figura 3. Giroscopi Mecànic.

Bàsicament existeixen 3 tipus de giroscopis, en funció de quin efecte utilitzen per tal de calcular la velocitat angular:

- Mecànics
- Vibratoris

- Òptics

Els utilitzats en la major part de la electrònica de consum son giroscopis vibratoris, per mida i preu de fabricació, son dispositius **MEMS** (Micro-machined Electro-Mechanical Systems) que utilitzen l'efecte Coriolis per tal de calcular quina es la velocitat de rotació.

Quan utilitzem un giroscopi es poden observar les següents fonts d'error:

- ***Bias***: És la sortida del giroscopi quan no pateix cap rotació
- Deriva del ***bias***: és la variació de la sortida del giroscopi quan no està sotmès a cap rotació, bàsicament aquesta deriva es deguda a la calor.

Com a conseqüència dels errors en moltes especificacions podem trobar referència a la mesura del ARW, aquesta propietat representa el soroll del giroscopi i té unes unitats de graus/(unitat de temps)$^{1/2}$, és pot entendre com la variació degut al soroll del càlcul de la orientació en estat estacionari. Idealment mantenint un giroscopi en estat estacionari en calcular la rotació a la que ha estat sotmès, amb la integració de la velocitat angular, la sortida es teòricament de 0º, ara bé si tingues per exemple un ARW de 1º/(1seg)$^{1/2}$ el resultat seria de d'1º passat un segon i 10º després de 100 segons.

2.3. Sensors Magnètics

Els sensors més moderns per a la mesura de la direcció d'un camp magnètic son els sensors d'efecte Hall i les brúixoles de interrupció de flux.

L'efecte Hall descriu el comportament del voltatge en un semiconductor quan aquest està en presencia d'un camp magnètic. Quan una corrent elèctrica constant s'aplica sobre el semiconductor apareixerà una diferencia de voltatge amb una direcció relacionada amb les línies de flux magnètic que travessen al semiconductor, a més el signe de la diferencia de voltatge indica la direcció del camp magnètic. Els sensors basats en l'efecte Hall son molt assequibles, per contra ofereixen una sèrie de desavantatges com poca resolució, errors de **bias** i la no linearitat del sensor, a més s'han d'afegir força elements per adaptar el senyal i això fa que és redueixi considerablement l'ample de banda del sensor.

Les brúixoles d'interrupció de flux és basen en un principi diferent, dues bobines petites s'enrotllen en nuclis de ferrita i es fixen perpendiculars entre si. Quan el corrent altern s'activa en ambdues bobines, el camp magnètic causa canvis en la fase en funció de la seva alineació relativa amb cada bobina. És pot calcular la direcció del camp magnètic de dues dimensions mesurant els dos desplaçaments de fase. Respecte els sensor d'efecte Hall, la brúixola de flux pot mesurar amb més precisió i resolució la força d'un camp magnètic però per contra es molt més car.

Mes enllà del tipus de sensor utilitzat, el pitjor inconvenient d'aquests sensors son les pertorbacions al camp magnètic que originen les estructures fetes per l'home com edificis, així com limitacions en l'ample de banda i la susceptibilitat a la vibració[5].

2.4. Actuadors

Un **servomotor** està format per un motor un reductor i un circuit de control, gracies a aquests elements és capaç de posicionar-se en qualsevol posició del seu rang d'operació i mantenir-se estable en aquesta posició. El control de la posició normalment es realitza

mitjançant la modulació per amplitud de polsos o ***pwm***, de l'anglès ***pulse width modulation***. Una senyal modulada per amplitud de polsos té una freqüència fixa i fa variar el seu cicle de treball, el cicle de treball indica l'ample relatiu de la part positiva en relació amb el període, es a dir:

$$D = \frac{\tau}{T} \qquad \text{Eq. 1}$$

On D és el cicle de treball, τ és el temps en que la funció és positiva i T el període del senyal.

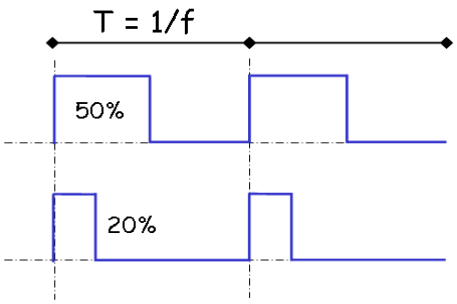


Figura 4. Senyal Modulada per Amplitud de Polsos.

Per tal de comprovar el funcionament d'un servomotor i simular el funcionament dels servomotors dels robots s'utilitza una placa ***Arduino*** que conté varis pins amb sortida ***pwm*** i a més conté un port USB per comunicar-se amb altres dispositius, utilitzant-lo com si es tractes d'un port sèrie.

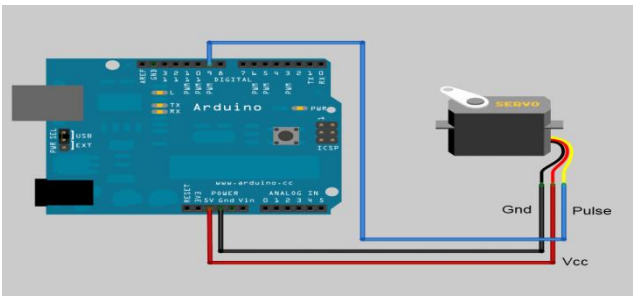


Figura 5. Connexió Arduino Servo.

En primer lloc s'analitza la sortida del **pwm** gracies a la utilització d'un potenciòmetre que ens modificarà la tensió d'entrada d'un pin del ***Arduino*** i amb això es pot calcular l'ample del pols de sortida del pwm, l'esquema és el següent:

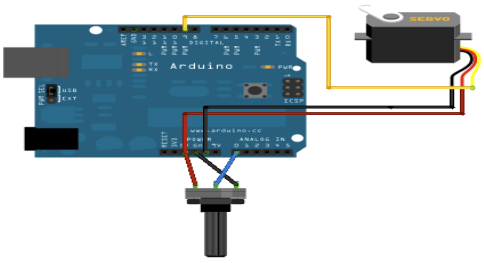


Figura 6. Connexió Arduino Servo amb potenciòmetre.

El programa introduït en la placa **Arduino** simplement llegeix l'entrada del pin 0 i escala la sortida del **pwm** en funció de la lectura, així per una entrada de 5 volts tindrem la sortida al 100% del **pwm** mentre que per 0 volts serà del 0% del **pwm**.

```
// Controlling a servo position using a potentiometer (variable resistor)
// by Michal Rinott <http://people.interaction-ivrea.it/m.rinott>
#include <Servo.h>
Servo myservo;   // create servo object to control a servo
int potpin = 0;   // analog pin used to connect the potentiometer
int val;      // variable to read the value from the analog pin
void setup()
        {
  myservo.attach(9);   // attaches the servo on pin 9 to the servo object
}

void loop()
        {
  val = analogRead(potpin); // reads the value of the potentiometer (value between 0 and 1023)
  val = map(val, 0, 1023, 0, 179); // scale it to use it with the servo (value between 0 and 180)
  myservo.write(val); // sets the servo position according to the scaled value
  delay(15); // waits for the servo to get there
}
```

Modificant el valors del potenciòmetre obtenim les següents sortides del pwm i les següents posicions del servomotor:

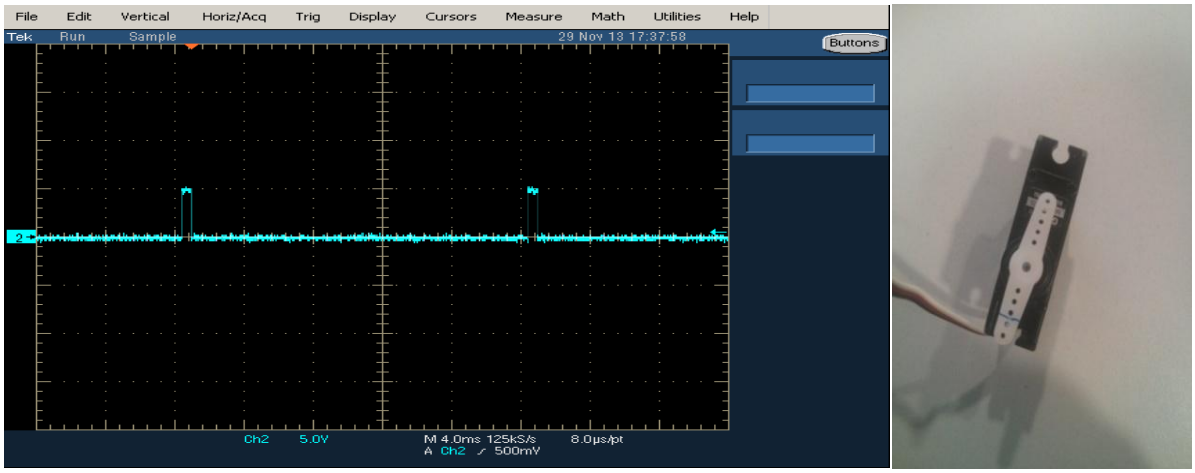


Figura 7. Sortida PWM amb el potenciòmetre al mínim.

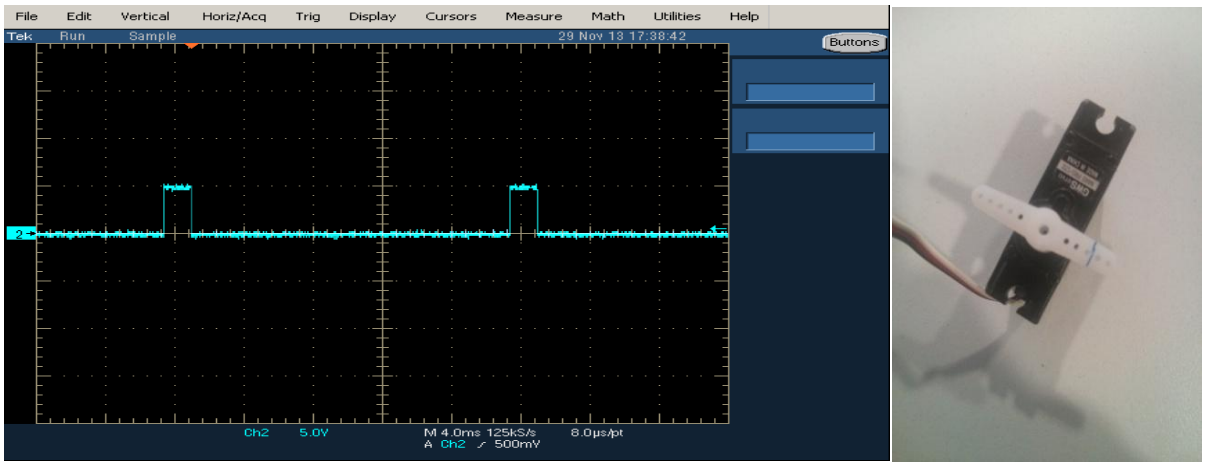


Figura 8. Sortida PWM amb el potenciòmetre al mig.

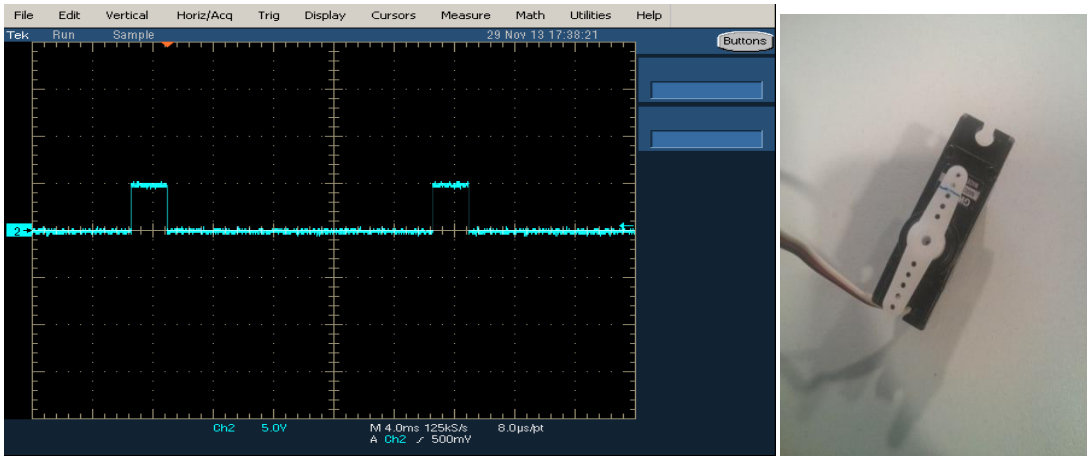


Figura 9. Sortida PWM amb el potenciòmetre al màxim.

Tal com és veu en les captures de l'oscil·loscopi la sortida del pwm te una freqüència de 50 Hz, amb la formula de l'equació 1.3 es pot calcular quin és el cicle de treball en cada cas:

Per el PWM mínim, la part positiva es de 0.8 ms, així el cicle de treball es de: 4%

Per el PWM mig, la part positiva es de 1.6 ms, així el cicle de treball es de: 8%

Per el PWM màxim, la part positiva es de 2.4 ms, així el cicle de treball es de: 12%

Es programa ara la placa Arduino per llegir valors del port sèrie i en funció del seu valor modificar la sortida del *pwm*:

```
#include <SoftwareSerial.h>
#include <Servo.h>
Servo myservo;

int pos = 0;
int motor = 0;

void setup()
{
  myservo.attach(9);   // attaches the servo on pin 9 to the servo object
  Serial.begin(115200);
  Serial.println("Moving to 0");
  myservo.write(pos);
}

void loop()
        {
         int nbits;
         int i;
         unsigned char data[2];
         unsigned int value;
        nbits = Serial.available();
        while(nbits<2)
         {
         nbits = Serial.available();
        }
        if(nbits)
            {
            Serial.println(nbits,DEC);
            Serial.println("Receiving Data!");
            data[0] = Serial.read();
```

```
            data[1] = Serial.read();
            value = data[0];
            Serial.println(value,DEC);
            Serial.println(value,DEC);
            value = (data[1]*256)+(data[0]);
            if(value>=0 and value<=180)
                {
         if (motor == 0) myservo.write(value);
        Serial.println("Moving to: ");
  Serial.println(value,DEC);
  }
 }
}
```

Amb aquest programa dins de la placa Arduino i utilitzant el mòdul **pyserial** de **Python** ens és molt fàcil programar una interfície en el PC que ens permeti controlar quina serà la posició del motor, en el següent exemple movem el motor fins a 90º, després fins a 180º i finalment el posem en la posició de 0º:

```
import struct
import time
comm = serial.Serial('/dev/ttyACM0',115200)
comm.write(struct.pack('h',90))
time.sleep(0.1)
comm.write(struct.pack('h',180))
time.sleep(0.1)
comm.write(struct.pack('h',0))
```

3. El Robot Kyosho Manoi

3.1. Introducció

La industria de la robòtica es una industria reeixida, actualment es poden veure braços robòtics capaços de moure's amb molta velocitat i amb una precisió enorme, la única mancança que tenen aquests robots és la falta de mobilitat.

Un robot mòbil necessita mecanismes que li permetin moure's lliurement en el medi, existeixen robots amb una gran varietat de maneres de moure's, caminar, saltar, córrer, lliscar, patinar, nedar, volar i per descomptat, rodar, la majoria d'aquests mecanismes de locomoció s'han inspirat en la natura, amb l'excepció de la roda de potència activa[6].

El moviment d'un robot depèn de moltes variables entre les que destaquen:

- Estabilitat
  - Numero i geometria dels punts de contacte
  - Centre de gravetat
  - Estabilitat dinàmica/estàtica
  - Inclinació del terreny
- Característiques del contacte:
  - Punt de contacte, mida i forma
  - Angle de contacte
  - Fricció
- Tipus de medi:
  - Estructura
  - medi, (per exemple: aigua, aire, terra dur…)

A més si es vol que sigui autònom ha de ser capaç de recollir dades del medi en el que es desenvolupa per tal de prendre decisions:

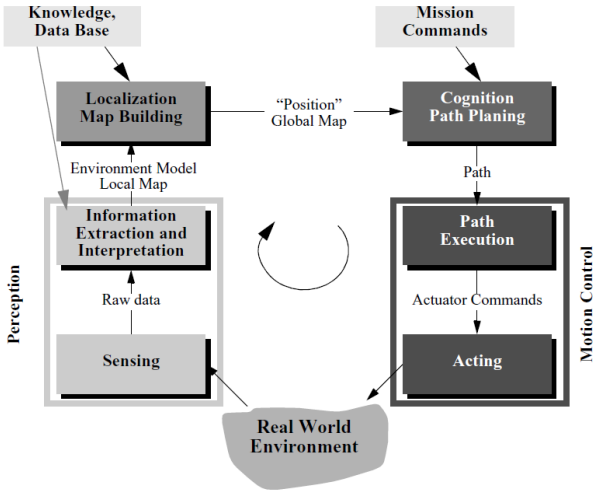


Figura 10. Esquema de blocs d'un robot mòbil autònom[7].

Tal com es pot veure a la figura anteriorun robot mòbilautònom interpretarà dades del medi captades amb els sensors i actuarà sobre ell mitjançant actuadors.

3.2. Informació bàsica

El Kyosho manoi es un robot mòbil bípede realitzat a una escala 1/5, conté 17 motors que li permeten realitzar moviments complexos en peus, cames i coll. Les seves característiques bàsiques son les següents:

Alçada	**340 mm**
Dimensions	340 x 204 [mm]
Pes	1.4kg
Motors	17

Taula 2 Característiques bàsiques At01

La targeta RCB-3 és el centre de control del moviment del robot, és llisten a continuació les seves característiques bàsiques:

Targeta de Control: RCB-3	
ICS	150 mm
Màxims Motors Controlables	24
Voltatge	9-12V
Dimensions	45x35 mm
Pes	12 g

Taula 3. Característiques RCB-3.

Figura 11. Targeta RCB3.

El moviment del robot es produeix mitjançant una sèrie de servomotors com el que mostra la figura 1, situats en determinades posicions que actuant sols o amb l'ajuda d'altres servomotors permet simular els moviments d'algunes articulacions humanes.

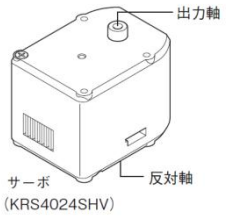


Figura 12. Detall Motor.

En total disposa de 17 servomotors, distribuïts tal com apareixen a la figura 13, cadascun d'aquests motors gira lògicament en una única direcció però ambdós sentits. Conté en total 3 motors dedicats al moviment dels braços (3 per braç) i 5 per cadascuna de les 2 cames, incloent el moviment dels malucs, els genolls i dels turmells, finalment hi ha un dedicat exclusivament per el moviment del coll. Tot i que racionalment es tendeix a ajuntar els motors en funció de l'extremitat a la que estan vinculats s'ha de ser conscient que el moviment d'un motor es independent de qualsevol altre.

Figura 13. Esquelet del Kyosho Manoi.

Les característiques de cadascun d'aquests servomotors es la següent:

Característiques Servo AT01	
Model del Servomotor	KRS-4024HV
Mesures	43. 32. 32.5 (mm)
Pes	52.5g
Voltatge d'operacions	9 - 12V
Parell de Forces	10.5kg. cm
Velocitat	0.17sec/60

Taula 4. Propietats Servomotor AT01.

3.3. La placa RCB3J en detall

El cor de les funcionalitats del robot Manoi és la placa RCB3J:

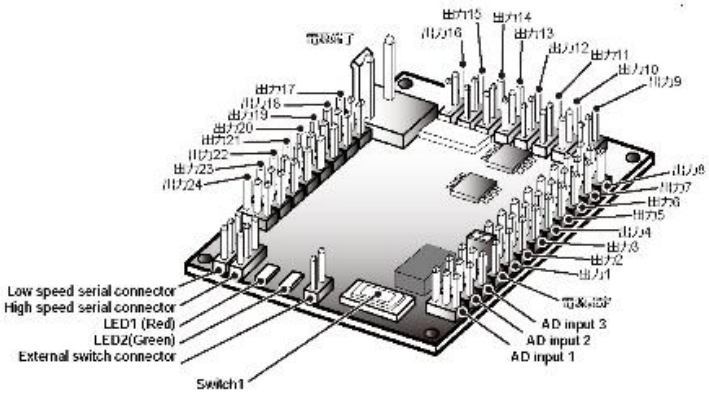


Figura 14. Placa RCB3J detall.

La placa disposa d'un total de 30 connexions, es poden diferenciar 2 tipus de connexions en funció del numero de pins que disposen, 29 entrades disposen de 3pins de connexió mentre que nomes n'hi ha dos que disposen únicament de 2 pins, una és tracta de l'entrada d'alimentació, es pot veure en detall en la següent figura:

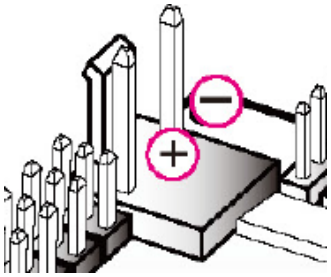


Figura 15. Connexió d'alimentació.

De les 29 entrades de 3 pins que la placa ***RCB3J*** conté24 son connexions de sortida per a servomotors, en el cas concret del ***Kyosho Manoi*** només son necessàries 17 d'aquestes connexions, de les 5 restants, 3 es corresponen a entrades analògiques que permetran l'ampliació de les capacitats del robot, connectant-hi diferents sensors, i les dues restants son connexions serials que permeten la programació de la placa, una es tracta d'una connexió d'alta velocitat per la connexió amb un PC, mentre que la segona es tracta d'una connexió de baixa velocitat que permet la connexió amb un control remot de radiofreqüència.

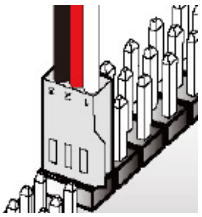


Figura 16. Connexió amb la placa.

Tal com es veu a la figura 16 la connexió entre els pins i els diferents elements és sempre igual, amb GND en el pin més extern dels 3, el del mig per a l'alimentació i finalment el pin interior per la senyal de control.

Per tal d'establir connexió amb un client, el robot ho realitzar mitjançant una clau USB que permet la conversió de dades USB a dades sèrie:

Figura 17. Detall clau USB de connexió.

3.3.1. El software de control de la placa

El robot ***Kyosho Manoi*** inclou un software que permet la comunicació directa amb la placa per tal de programar-la, aquest software s'anomena ***Heart to Heart 3*** i presenta la següent interfície gràfica:

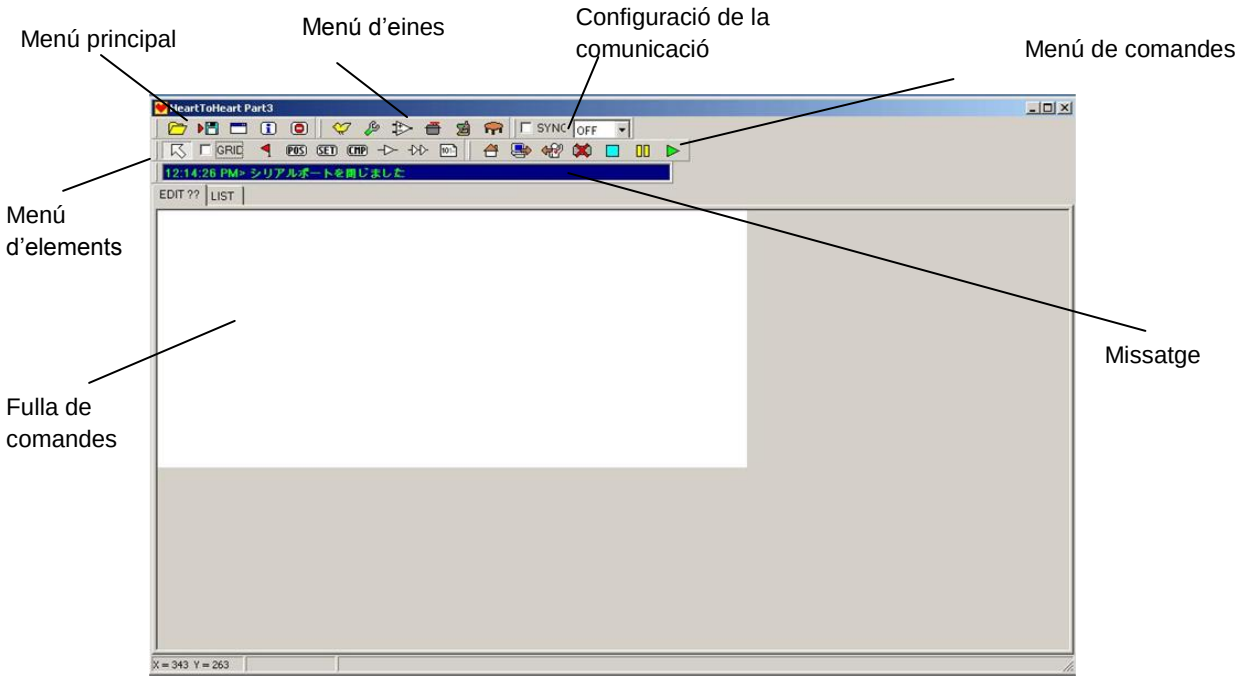
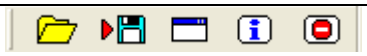
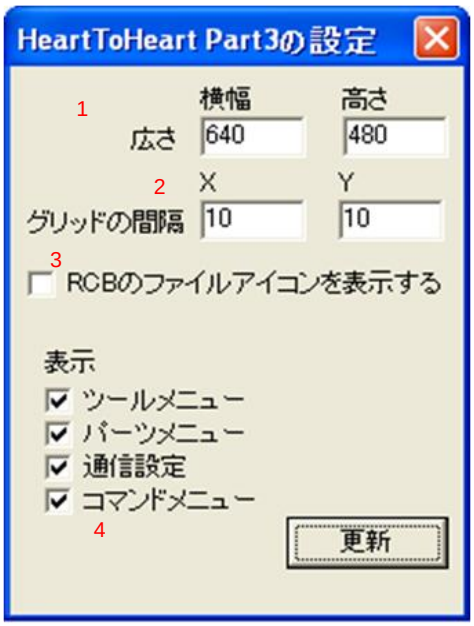


Figura 18. Pantalla principal HTH-3.

El funcionament bàsic del software és afegir sobre la fulla de comandes diferents elements

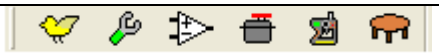
units entre si, fins un total de 255, aquests elements relacionats formaran el que anomenarem una **operació**. Aquestes **operacions** les podem guardar a la placa RCB3J on es disposa de fins a 80 posicions per guardar-les. Aquestes operacions guardades dins de la placa poden ser automàticament executades en funció de factors externs llegits per els sensors o poden ser executades manualment per l'usuari, des de el programari o un comandament a distància.

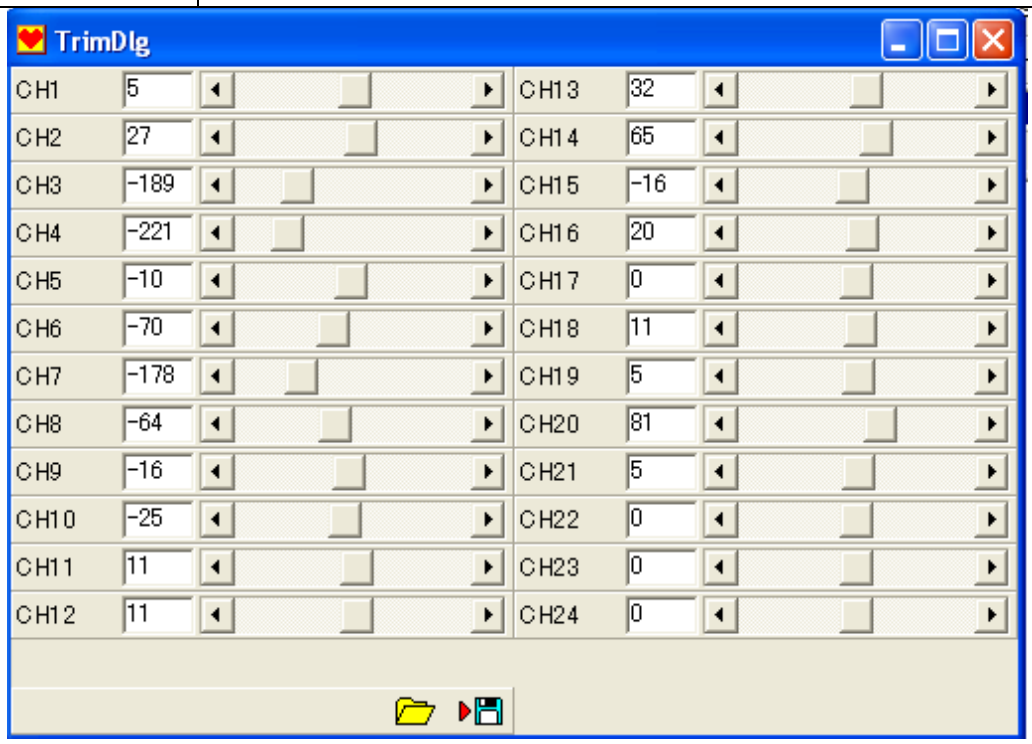
El menú del programa inclou totes aquestes funcionalitats resumides en les següents taules:

<table>
<tr><td colspan="2">Menú Principal:  </td></tr>
<tr><td>  </td><td>Carrega una nova sèrie de moviments guardats en un fitxer d'extensió RCB.</td></tr>
<tr><td>  </td><td>Guardar la llista de moviments en el disc dur</td></tr>
<tr><td>  </td><td>Obre una finestra que permet editar els paràmetres de la finestra principal</td></tr>
<tr><td colspan="2"> 

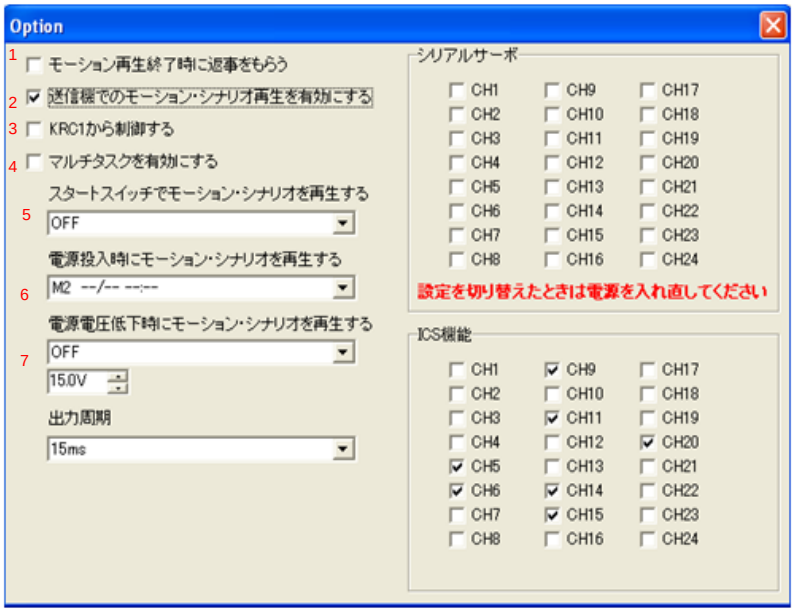
1: Estableix la dimensió de la fulla de comandes.
2: Estableix la resolució de la quadricula.
3: amb aquesta caixa marcada els fitxers de dades es mostraran en un fitxer especial.
4: Marcant o desmarcant cadascuna d'aquestes caixes, apareixeran o desapareixeran els menús de la pantalla principal.</td></tr>
<tr><td>  </td><td>Informació sobre el software Heart to Heart 3</td></tr>
<tr><td>  </td><td>Tancar la finestra</td></tr>
</table>

Taula 5. Menú Principal.

Menúd'eines:	
	Obre una finestra que permet definir les posicions inicials dels motors



	Obre una finestra que permet definir els paràmetres de la placa RCB3J



1: Enviar ACK quan una **operació** finalitza

2: Deixar que es puguin executar operacions des de el transmissor KRC1.

3: Es controla el robot des de un transmissor KRC1

4: Utilitzar Multi tasca.

5: Execució d'una **operació** quan es prem **start**

6: Selecció de la **operació** a executar quan s'encén el robot

7: Selecció de la **operació** a executar quan el robot disposa de poca bateria

	Obre una finestra que permet definir l'ús de les entrades analògiques AD1, AD2 i AD3

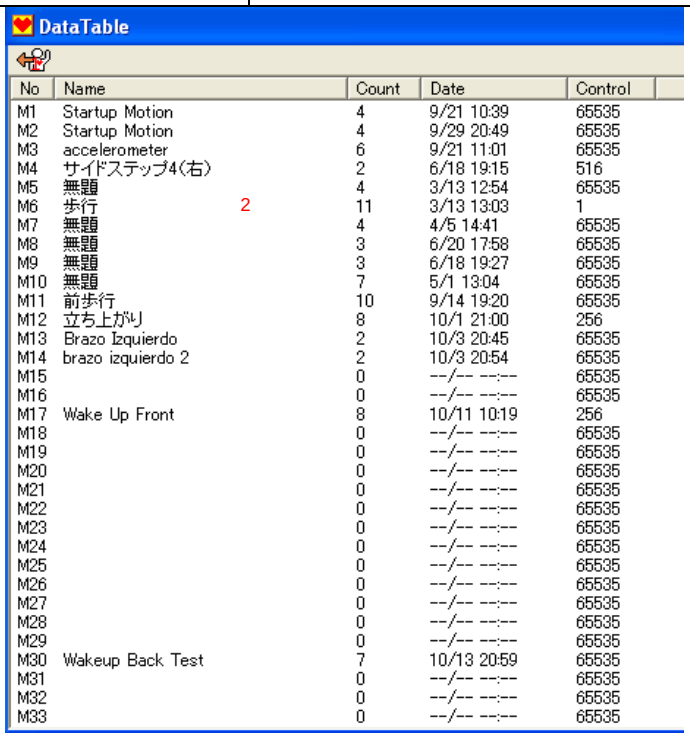
Taula 6. Menú Eines A.

<table>
<tr>
<td> 
</td>
<td>1: Selecció de cadascuna de les entrades analògiques, la placa RCB3J disposa de 3.

2: Aquest camp permet modificar l'offset de l'entrada analògica un cop ja digitalitzada

3: Amb aquestes caixes es selecciona la quantitat de la lectura analògica que modificarà la posició dels motors en temps real.

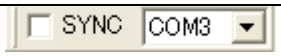
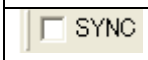
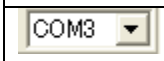
4: Permet la generació d'interrupcions quan les entrades analògiques superen una quantitat.

5: Panell es merament informatiu i mostra el valors analògiques de les tres entrades i el voltatge de la bateria.</td>
</tr>
<tr>
<td>  </td>
<td>S'obre una finestra que permet modificar les característiques dels servomotors del Robot.</td>
</tr>
<tr>
<td>  </td>
<td>S'obre una finestra que permet monitoritzar les dades rebudes i enviades mitjançant el radio transmissor.</td>
</tr>
<tr>
<td>  </td>
<td>Permet visualitzar totes els moviments i escenaris que tenim emmagatzemats dins de la placa RCB3J</td>
</tr>
<tr>
<td> 
</td>
<td>1: Amb aquest botó es traspassen tots els moviments guardats a la placa RCB3J a la taula 2.

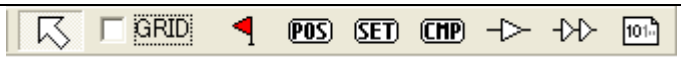
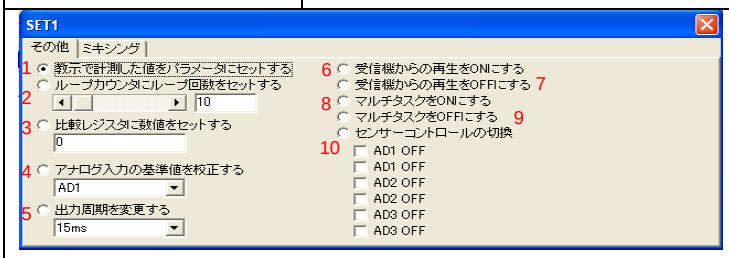
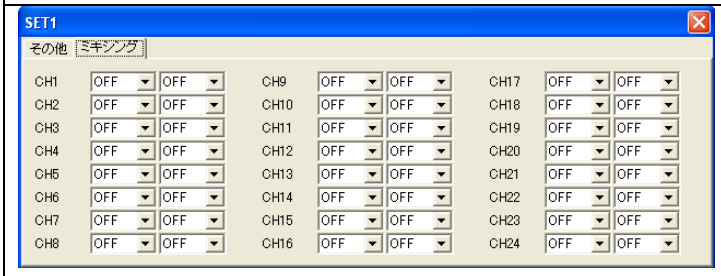
2: Es mostra en aquesta taula tota la informació sobre els moviments emmagatzemats a la taula. La informació és per ordre:

1. Numero on esta registrat
2. Nom del moviment
3. Numero de vegades que ha estat executat
4. Dia i hora de l'ultima execució.
5. Numero de control.</td>
</tr>
</table>

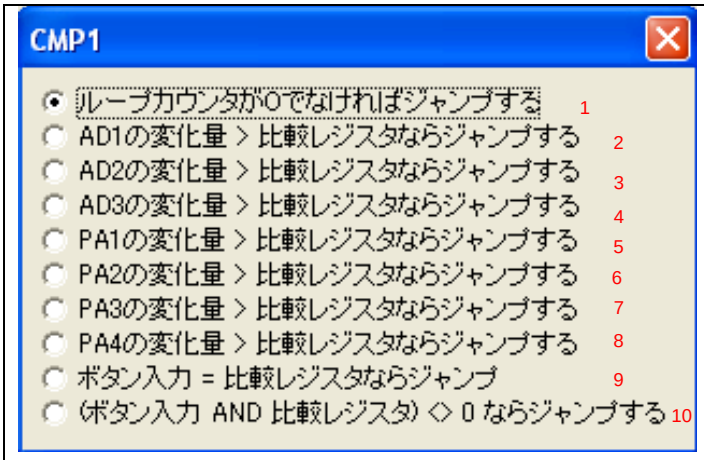
Taula 7. Menú d'eines B.

Configuració de la comunicació	
	Amb aquesta opció marcada els motors es mouran automàticament quan modifiquem el valor de la seva posició en el software
	Selecció del port sèrie on esta connectat el robot.

Taula 8. Configuració de la comunicació.

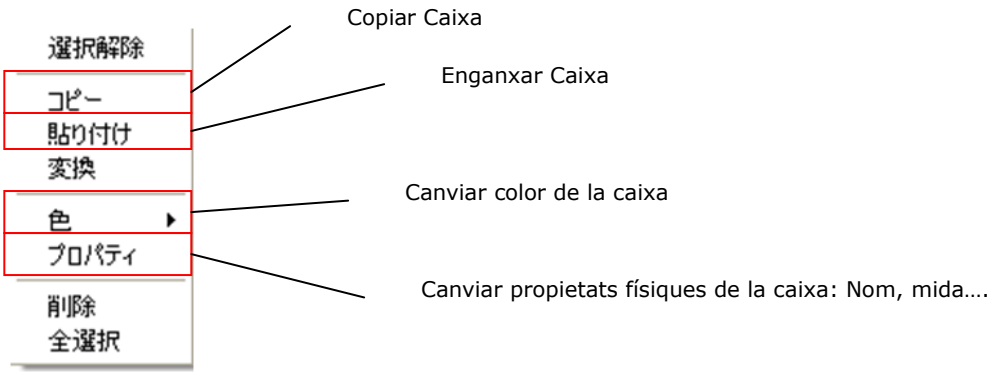
Menú d'elements:	
	Permet moure i selecciona els diferents elements de la fulla de comandes
	Si el marquem apareix una reixa sobre la fulla de comandes que permet una millor col·locació dels diferents elements.
	Indicador per marcar en quin punt comença en moviment de la fulla de comandes.
	Permet modificar les posicions dels motors per la definició d'un moviment.
	Obre un diàleg que permet modificar en calent diferents paràmetres entre postures, com les dades del **mixing** o el número de bucle
	1: No ho sabem 2: Comptador de bucle, estableix el valor inicial del valor d'un bucle 3: Número de valor del registre de comparació 4: El número de referència és el valor que tenim a les entrades analògiques, AD1, AD2 O AD3. 5: No ho sabem 6: Ignorar les senyals que venen del control remot 7: Acceptar les senyals que venen del control remot 8 i 9: No ho sabem 10: Anular el llançament de moviments per els canvis de les entrades digitals.
	L'altre pestanya permet canviar en calent les dades del **mixing** amb les senyals que llegim de les entrades analògiques, ja sigui dels sensors o del comandament remot.
	Obre un diàleg que permet realitzaren calent comparacions i en funció de la comparació realitzar una bifurcació.

Taula 9. Menú d'elements A.

	1: Salta quan el valor del registre del bucle no és zero 2: Salta quan el valor de AD1 es major que el registre de comparació 3: Salta quan el valor de AD2 es major que el registre de comparació 4: Salta quan el valor de AD3 es major que el registre de comparació 5: Salta quan el valor de PA1 es major que el registre de comparació 6: Salta quan el valor de PA2 es major que el registre de comparació 7: Salta quan el valor de PA3 es major que el registre de comparació 8: Salta quan el valor de PA4 es major que el registre de comparació 9: Salta quan el valor de botó d'entrada es igual al registre de comparació 10: Salta quan el valor de botó d'entrada es diferent al registre de comparació
	Uneix els diferents elements que formen part de l'escenari i marca també l'ordre de execució.
	Només es fica sobre els elements de comparació i indica quin camí prendre quan el resultat de la comparació es veritat.
	Compila el moviment de tal maneta que converteix tots els elements gràfics de la fulla de dades en ordres comprensibles per la placa RCB3-J

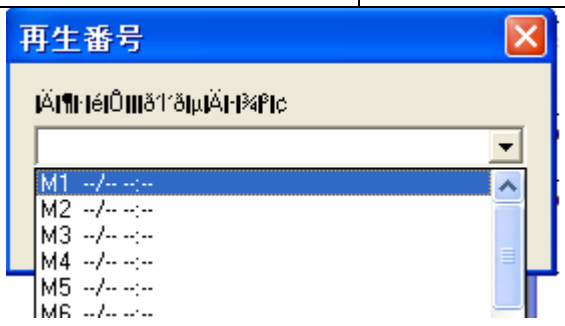
Taula 10. Menú d'elements B.

Fent clic amb el botó contextual sobre una caixa **pos**, **set** o **cmp** obtenim el següent menú:



Menú de comandes	
	Situa els servos del robot en la posició inicial
	Envia l'escenari cap a la placa indicant-li en quina posició volem guardar-lo, és seleccionat amb el següent quadre diàleg:

Taula 11. Menú de Comandes A.

<table>
<tr><td colspan="2"> 
</td></tr>
<tr><td>  </td><td>Carregar un moviment de la placa, és seleccionat amb el següent quadre diàleg:</td></tr>
<tr><td colspan="2"> 
</td></tr>
<tr><td>  </td><td>Esborrar un escenari guardat a la placa RCBJ3</td></tr>
<tr><td>  </td><td>Para l’execució de la operació en curs</td></tr>
<tr><td>  </td><td>Pausa l’execució de la operació en curs</td></tr>
<tr><td>  </td><td>Executa unaoperació que tenim guardat a la placa RCB3J, és seleccionat amb el següent quadre diàleg:</td></tr>
<tr><td colspan="2"> 
</td></tr>
</table>

Taula 12. Menú de Comandes B.

3.3.2. Us bàsic del software

3.3.2.1. Connectant el Robot amb un PC

Abans de començar a treballar amb el software el robot ha d'estar connectat a l'ordinador, el software ha d'estar en execució i finalment s'ha de connectar l'aplicació amb el robot, això es realitza seleccionant el port de comunicacions dins de l'aplicació:

Figura 19. Selecció port comunicacions.

Un cop seleccionat el port de comunicacions ha d'aparèixer el següent missatge, que indica que la connexió ha estat possible entre el PC i el Robot:

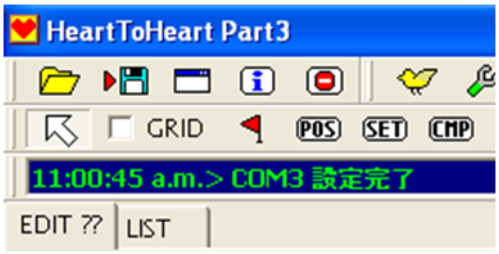


Figura 20. Detall Missatges.

3.3.2.2. Moviment dels motors

Amb la placa i el software sincronitzats es poden començar les primeres proves amb el robot, per realitzar simples moviments dels motors s'ha de marcar la caixa de ***sync*** per tal que les ordres dels motors que es demanen en el software siguin immediatament interpretades i realitzades a la placa, arrastrant sobre la zona de descripció de moviments la icona de ***pos***i tot seguit fent doble clic, apareixerà la següent pantalla:

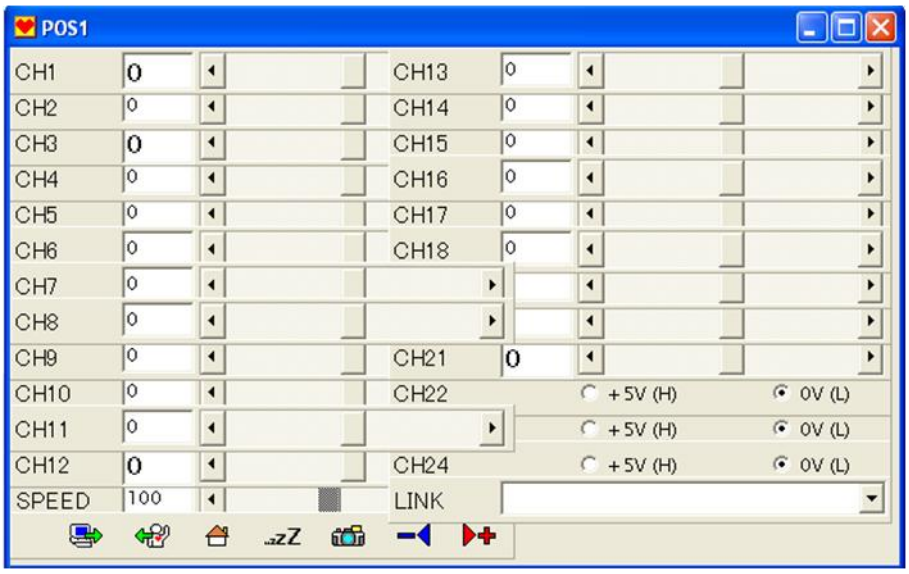


Figura 21. Control individual dels motors.

Cadascuna de les barres de desplaçament esta numerada amb un número de canal, així en modificar la posició de la barra és modificarà el valor d'aquell canal i per tant es produirà el moviment del servo que estigui connectat, en cas de moure un canal sense cap motor connectat no provocarà cap reacció.

3.3.2.3. Programar un moviment

Per tal de realitzar un moviment complex, per exemple caminar, simplement s'han d'enllaçar moviments simples, per això en la pantalla principal es disposen de diverses icones **pos** i s'enllacen amb l'eina de cablejat, per exemple:

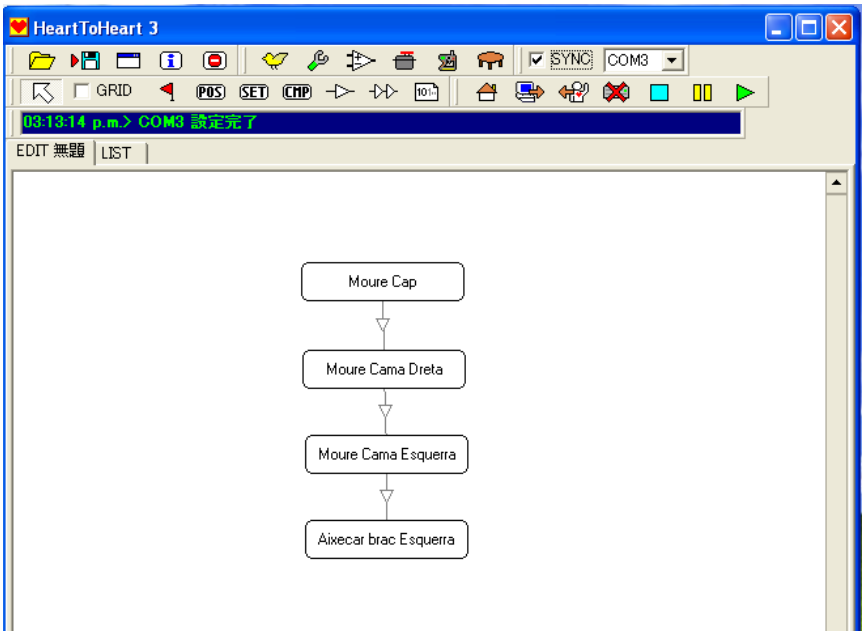


Figura 22. Exemple moviment senzill.

Fent doble clic a cadascuna de les posicions es poden modificar les posicions dels motors en aquell punt, ara és copia dins d'una posició de memòria amb l'eina de enviament cap al robot:  . Un cop emmagatzemada, es pot disposar del moviment, per exemple llançant-la amb el play  . Un cop s'executi el moviment, la placa RCB3-J mourà els motors en l'ordre que indiquem i la transició entre posició i posició farà que es generi el moviment.

3.3.3. Ús Avançat del software

El manoi com s'ha descrit abans disposa de varies extensions que permeten donar-li moltes més utilitats que les que ja s'han descrit del robot AT01, podem distingir dos tipus d'extensions:

- Extensions que permeten a l'usuari interactuar directament amb el robot.
- Extensions que permeten rebre informació al robot per prendre decisions autònomes.

En el primer grup distingim 2 extensions KRC-3 i KRR-1, que es tracta de un receptor i emissor de radio freqüència.

En el segon grup s'inclouen tots els sensors que es podem connectar als ports d'entrada analògics i permeten llançar moviments o modificar la posició d'un servomotor individual. Sensors disponibles n'hi han molts, per exemple la mateixa casa **Kondo** comercialitza els següents sensors:

- KRG-4: Giroscopis d'un eix
- RAS-2: Acceleròmetre
- Hrp-103: Sensor de distancia
- Hrp-101P: Sensor Tàctil

A continuació es descriu com es poden utilitzar algunes d'aquestes extensions.

3.3.3.1. Utilització d'un control remot

No ha estat possible obtenir un control remot per funcionar amb el robot, però ens fem ala idea de com podria funcionar. El control remot comercialitzat per ser utilitzat amb el robot manoi és el **KRC-3** i es comunica amb el robot amb un petit receptor anomenat KRR-1, aquest receptor s'ha de connectar amb l'entrada serial de baixa velocitat de que disposa la placa RCB3J. Un cop connectat l'extensió **KRR-1** únicament s'ha d'indicar a la pantalla de paràmetres que el robot utilitzarà un control remot **KRC** per tal de comunicar-se amb la placa.

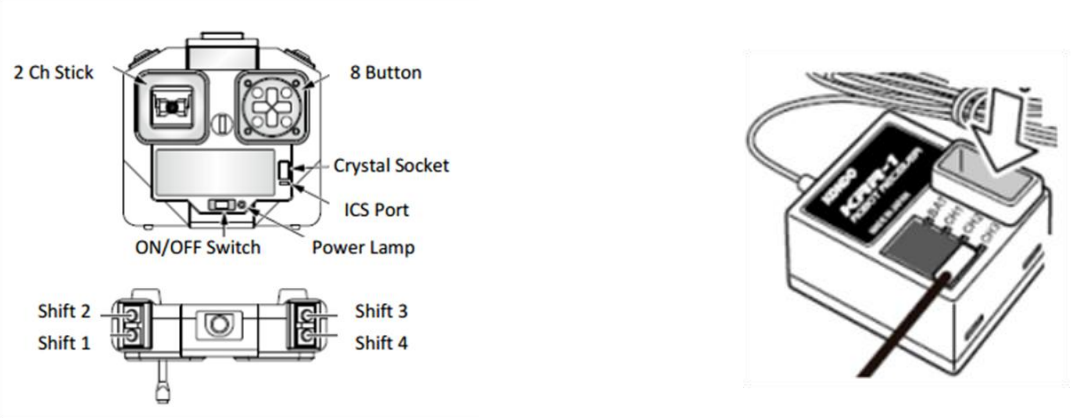


Figura 23. Esquema KRC-3 i KRR-1.

Amb el comandament es poden realitzar moltes operacions, des d'executar moviments gravats a la placa fins a modificar els valors individuals de cadascun dels servomotors. El parell **KRG-3 KRR-1** funciona bàsicament enviant per RF el valor del botó que l'usuari està prement, aquest valor es rebut al receptor i emmagatzemat en un registre especial de la placa RCB3J, de manera que la placa pot reaccionar en funció del valor d'aquest registre.

	RCB-3 Data input values				
	w/out shift	Shift 1	Shift 2	Shift 3	Shift 4
Neutral	0	512	1024	2048	4096
↑	1	513	1025	2049	4097
↓	2	514	1026	2050	4098
→	4	516	1028	2052	4100
↗	5	517	1029	2053	4101
↘	6	518	1030	2054	4102
←	8	520	1032	2056	4104
↙	9	521	1033	2057	4105
↖	10	522	1034	2058	4106
△	16	528	1040	2064	4112
×	32	544	1056	2080	4128
○	64	576	1088	2112	4160
↗	80	592	1104	2128	4176
↘	96	608	1120	2144	4192
□	256	768	1280	2304	4352
↙	272	784	1296	2320	4368
↖	288	800	1312	2336	4384

Taula 13. Relació Motor – Valor.

En aquest cas ens centrarem en poder executar moviments preexistents dins de la placa RCB3J, aquesta operació és molt fàcil de programar mitjançant el software que HTH3, accedint a la configuració dels moviments disponibles dins de la placa  , s'obté la següent pantalla:

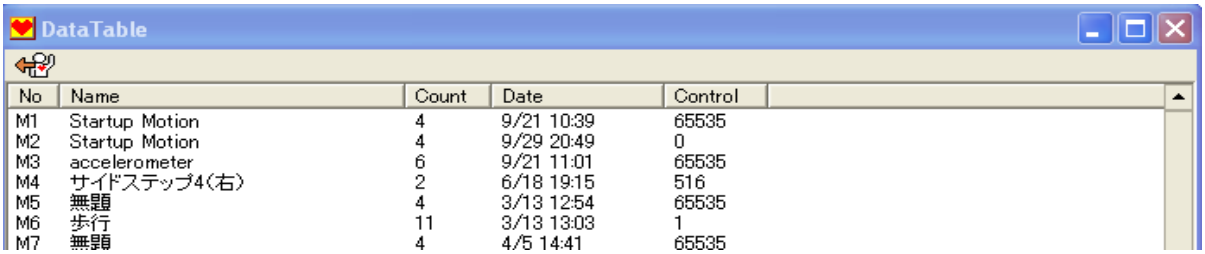


Figura 24. Detall càrrega dels moviments.

Un cop carregats els diferents moviments que tenim en la placa, es pot fer doble clic sobre cadascun dels moviments, i així apareixerà el següent diàleg:

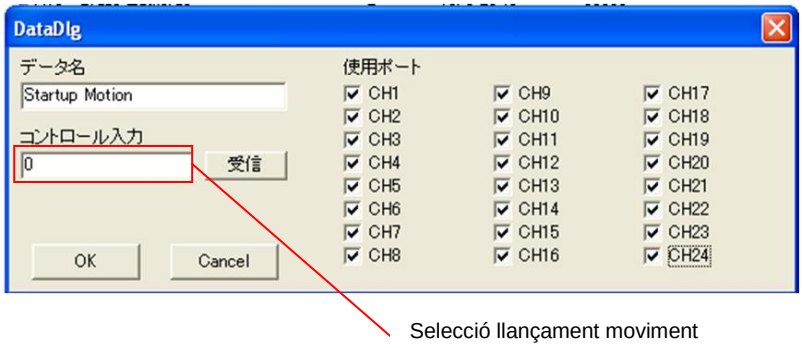


Figura 25. Diàleg de selecció del moviment.

Aquest diàleg permet seleccionar quin es el valor que s'ha de tenir dins el registre per tal que el moviment sigui executat, segons ens indica la taula 13 el valor que indica que hem premut la fletxa endavant és 1, per tant si volguéssim fer moure el robot endavant lo més lògic seria seleccionar el moviment de moure endavant i després introduir el valor 1 en aquesta casella. L'únic problema d'aquesta manera d'executar els moviments és que cada cop que es vulguin executar el moviments s'ha de prémer el botó, això pot ser molt emprenyador en cas de voler realitzar un moviment continu, fent que cada cop que el cicle

d'una pulsació finalitzi s'hagi de tornar a premé al botó, això es pot millorar editant el moviment i utilitzant les caixes de **SET** i **CMP.**

Tenint un moviment base com caminar endavant:

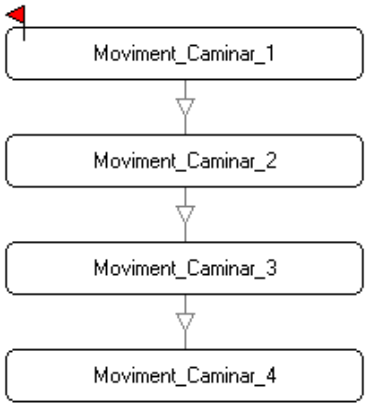


Figura 26. Esquema Moviment Caminar.

Disposant d'aquest moviment base es poden afegir caixes de SET i CMP per tal de fer el moviment continu:

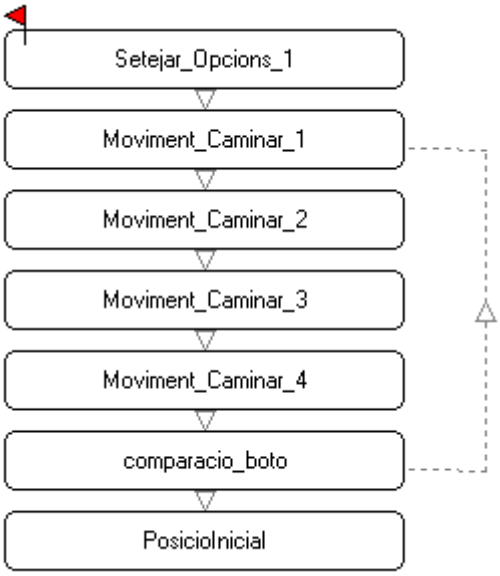
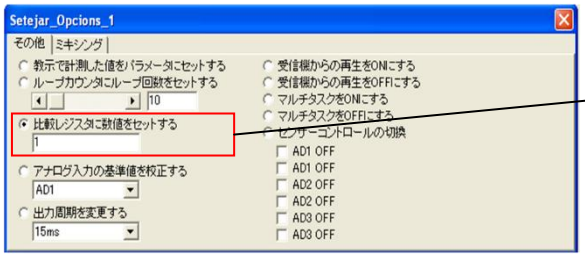
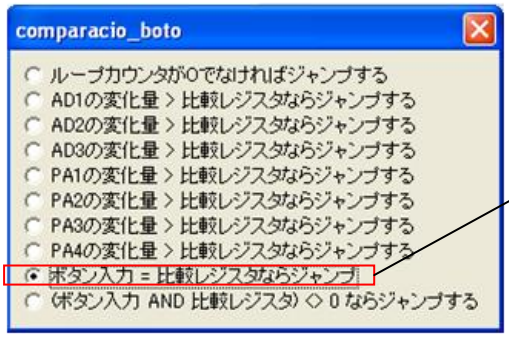


Figura 27. Caminar Endavant Continu.

Editant els valors de les caixes de **SET** s'estableixen les opcions per tal que el valor de comparació en el registre sigui igual al valor que s'ha programat per que s'executi aquest moviment, que com s'ha vist abans és 1 i editant la caixa de **CMP** és pot seleccionar quin és el valor que s'ha de mantenir per tal que es repeteixi el bucle de moviment.



Amb aquesta opció seleccionada, s'omple el valor del registre de comparació amb un 1 que és el corresponent a fletxa endavant.



En la comparació es selecciona aquesta opció per tal que es compari el valor del registre de comparació amb el botó que es rep del comandament a distància i en cas que siguin iguals es torni al inici del moviment.

3.3.3.2. Interrupcions a la placa RCB3J:

La placa RCB3J permet llançar moviments programats prèviament en funció de factors externs, per això es necessita en primer lloc un sensor que indiqui que s'ha produït un esdeveniment i programar la placa RCB3J per tan d'indicar-li quina acció ha de realitzar.

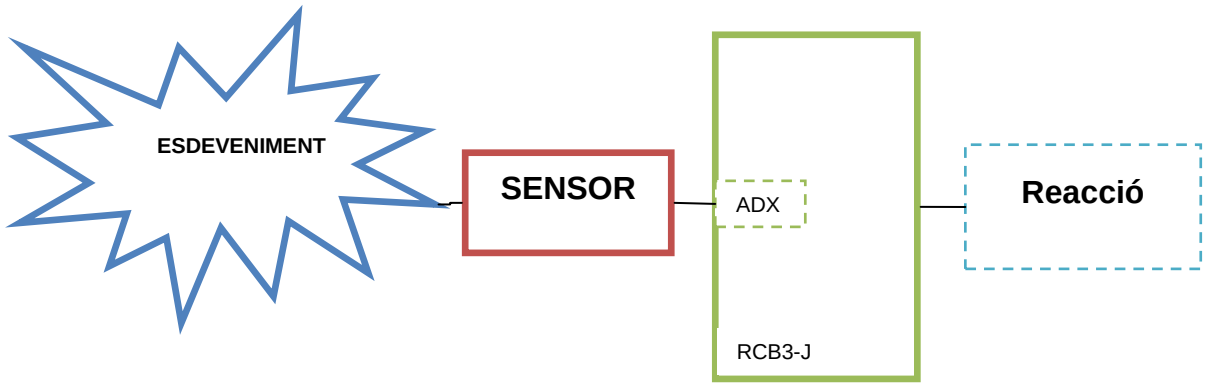


Figura 28. Llançament d'un moviment.

La placa disposa de 3 entrades analògiques i permet definir fins a dos reaccions en funció del valor mesurat per cadascuna de les entrades. És descriu a continuació un exemple per veure com funcionen i es programen les interrupcions dins la placa **RCB3J**.

Exemple: El robot s'aixeca sol

Per tal que el robot s'aixequi de manera autònoma la primera cosa que s'ha d'aconseguir lògicament és d'alguna manera que el robot ha caigut, això es possible gracies a l'ús d'un acceleròmetre que connectarem a una de les entrades analògiques de la placa.

Tal com s'ha descrit anteriorment existeix una solució comercial de Kyosho amb un acceleròmetre que pot ser directament connectada a la placa **RCB3J** anomenada **RAS-2**. Per desgràcia no ens ha estat possible accedir a un **RAS-2** per tant s'ha decidit dissenyar una placa exclusiva per aquesta funció. Després de buscar diferents solucions es decideix utilitzar una placa dissenyada per la plataforma Arduino que inclou un acceleròmetre d'**Analog Devices** l'**ADXL335**[1]**.**

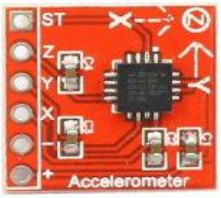


Figura 29. Foto ADXL335.

L'**ADXL335** es tracta d'un acceleròmetre de tres eixos que s'alimenta amb un voltatge entre 1.8 i 3.6 Volts i que detecta una força de ±3g. Les entrades analògiques de la placa disposen de la següent distribució:

Figura 30. Distribució dels pins d'una entrada analògica.

La VDD que s'obté des de la placa son de 5 volts, per tant s'ha de aconseguir adaptar-la a l'entrada requerida per el nostre sensor, per això es realitza una placa amb un regulador que passa de 5 a 3.3 Volts, aquí es pot veure quin és l'esquema final de tot el muntatge:

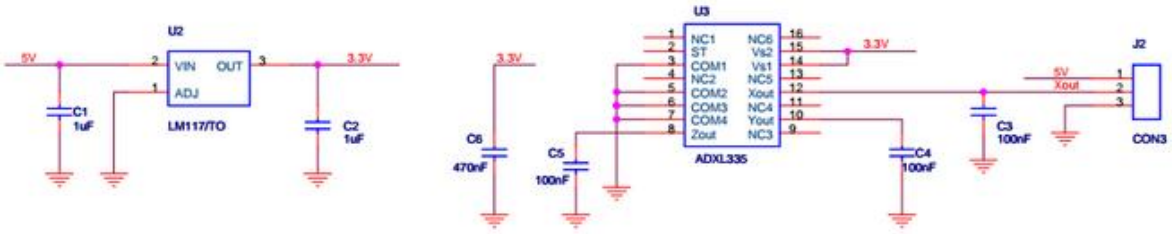


Figura 31. Esquema final ADXL335.

[1]Full d'especificacions disponible a documentació.

Amb la placa creada s'ha de situar en una posició que permeti distingir quan el robot ha caigut, lògicament la millor posició es fora de qualsevol extremitat, per tant només es pot situar sobre el tronc o sobre el cap, aquest exemple s'ha decidit capturar la direcció Z de l'acceleròmetre per tant ens és molt més fàcil col·locar-la sobre el tronc:

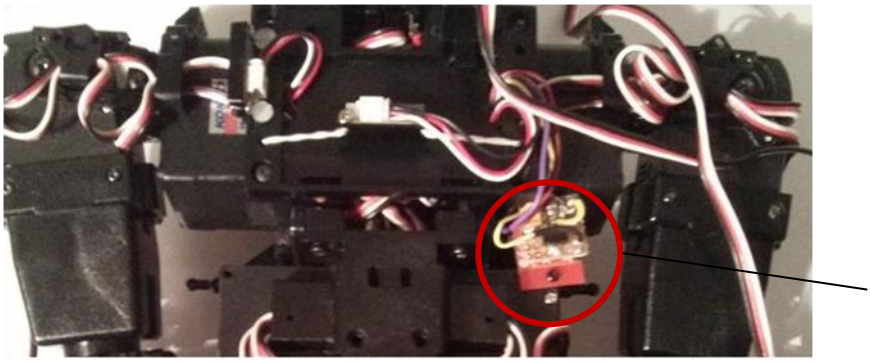


Figura 32. Detall Posició acceleròmetre.

Un cop la placa amb l'acceleròmetre està integrada sobre el robot només resta programar la placa RCB3-J per tal que llanci els moviments necessaris per aixecar-se un cop hagi caigut. Això es realitza mitjançant la utilitat de configuració dels ports analògics , s'obre el següent diàleg:

Figura 33. Control de les entrades analògiques.

Es selecciona la pestanya en funció de en quina de les 3 entrades analògiques està

connectat a l'acceleròmetre. Es pot veure a 7 el valor de cadascuna de les entrades analògiques, a 3 es mostra el mateix valor per l'entrada analògica seleccionada, però en aquest cas després de digitalitzar-lo mitjançant un convertidor analògic-digital.

En primer lloc s'ha d'obtenir el valor de referència quan el robot esta dret en una posició estable, aquest valor ha de ser zero i podem modificar-lo gracies a la barra 2 (la variació del valor es veu a la caixa 1), la suma de l'entrada analògica i l'offset que afegeix la barra se'ns mostra a la caixa 4.

Amb aquest valor de referència fixat a zero podem comprovar que modificant la inclinació vertical del robot hi ha una variació del valor del quadre de text referència amb el número 4, però sempre, mantenint el robot dret i estable el seu valor és zero (aproximadament).

Els valors que tenim indicats a 5.a i 6.a i que podem modificar amb les barres 5.b i 6.b respectivament permeten variar els límits que faran que la placa **RCB3J**inicií el numero de moviment que s'indica mitjançant el quadre desplegable 5.c i 6.c.

En el cas de voler que el robot s'aixequi sol per exemple i com es disposa de 2 interrupcions podem tenir dos rutines d'aixecament, en funció de quin és el sentit de caiguda del robot, de cara o d'esquena, així amb el robot estirat de cara i d'esquena s'apunta el valor de la caixa 4, aquests valor serà el que ficarem a les caixes 5.a i 6.a i li indicarem amb 5.c i 6.c el moviment corresponent a cadascuna de les necessitats, aixecar-se de cara o d'esquena.

De la mateixa manera que s'ha programat el robot per tal que reaccioni a una caiguda, es possible fer que reaccioni a altres accions en funció del sensor que connectem a les entrades analògiques, per exemple amb un sensor de proximitat fer que hagi de realitzar un gir a esquerra o dreta en funció de si troba un obstacle.

3.3.3.3. Barreja de Senyals

La barreja de senyals permet modificar la posició d'un servo en funció de les lectures dels ports analògics, això és realitza amb les entrades dels ports analògics o alternativament també es pot realitzar amb els valors que obtenim dels ***sticks*** analògics del comandament a distancia:

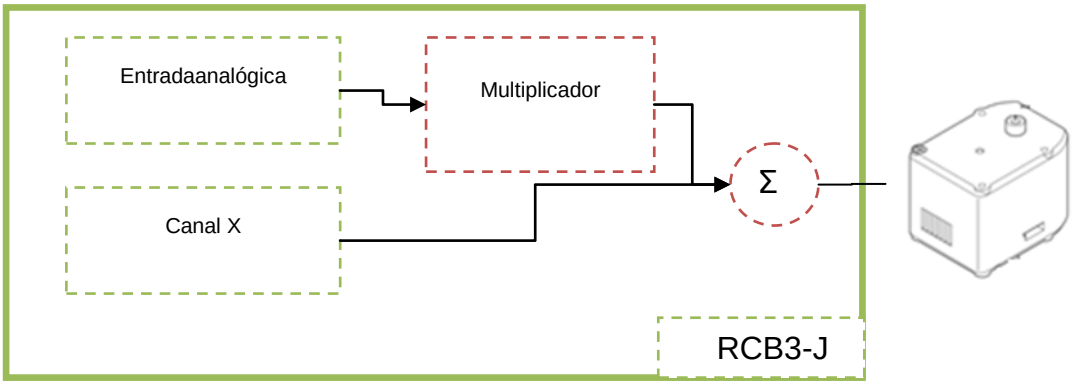


Figura 34. Barreja de senyals.

Tal com es veu en la figura la placa RCB3-J cadascuna de les entrades analògiques

disposa d'un multiplicador amb valor variable que permet graduar com serà la resposta de la posició del motor. No hi ha límit en la quantitat de motors que es poden controlar per cadascuna de les entrades analògiques, així una mateixa entrada analògica pot barrejar-se amb la posició de cadascun dels 17 servomotors, tot i que la lògica no ho aconsella.

Exemple: Guanyar estabilitat amb l'ajuda de giroscopis

Per guanyar estabilitat en el robot es necessària la utilització de giroscopis que barregin la seva sortida amb la sortida desitjada del sensor, de tal manera que les variacions de les lectures del giroscopi provoquin canvis en la posició dels servos, per això s'han de decidir els següents punts:

1. Quina quantitat de giroscopis es necessiten.
2. En quina posició s'han de col·locar els giroscopis.
3. Com s'ha de configurar la placa per tal d'utilitzar els giroscopis.

Els dos primers punts van de la mà, la placa RCB3-J té un màxim de 3 entrades analògiques, per tant el màxim de giroscopis que es podrien connectar serien 3, si a més s'avalua el moviment del robot, s'arriba a la conclusió que el robot en condicions normals només pot perdre l'estabilitat sobre 2 eixos, sobre l'eix X i sobre l'eix Y, per tant situant un giroscopi per captar possibles pertorbacions en cadascun d'aquests 2 eixos per compensar una possible inestabilitat seria suficient per fer el robot més estable.

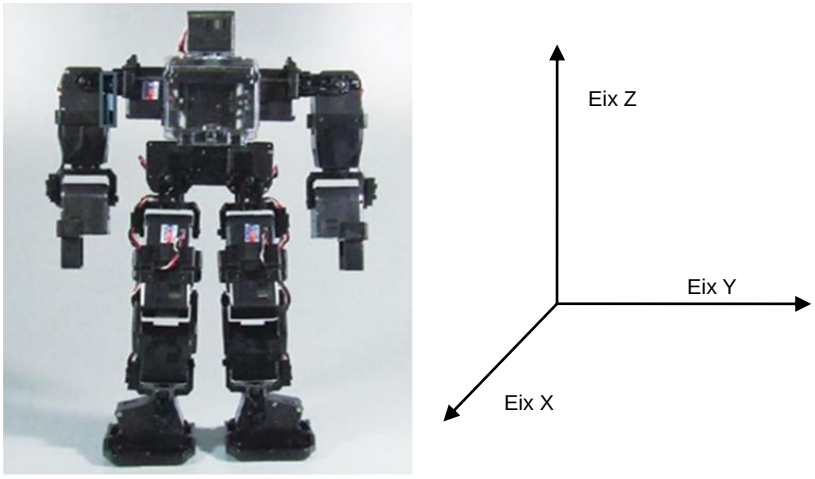


Figura 35. Eixos de moviment del robot.

És a dir s'han de situar els giroscopis per tal de compensar una possible caiguda amb una rotació sobre l'eix Y, que significarà una caiguda cap endavant o cap enrere o amb una rotació sobre l'eix Y que voldrà dir que la caiguda succeeix per un dels costats del robot, esquerra o dreta.

S'avaluen ara les especificacions del giroscopi KRG-4 per saber de quina manera s'han de situar sobre el robot per captar inestabilitats en qualsevol dels dos eixos. El KRG-4 és un giroscopi d'un únic eix així doncs serà molt important la forma en que es instal·lat, l'eix de funcionament del KRG-4 és mostra a següent figura:

Figura 36. Eix de Medició sensor KRG-4.

Coneixent l'angle de medició és disposen els giroscopis sobre el robot Manoi At01 de la manera[2] que indica la figura 37.

Un cop instal·lats els giroscopis s'han de definir quins son els servos que hauran de modificar la seva posició en funció de les lectures que es llegeixin dels sensors. Un anàlisis superficial del moviment d'un cos humà revela que en gran mesura l'estabilitat en caminares degut a les cames, tot i que els braços també ajuden. Les cames del **Kyosho Manoi** disposen d'un total de 10 servos, com es mostra en la figura 38, que s'hauran d'utilitzar per compensar les possibles inestabilitats.

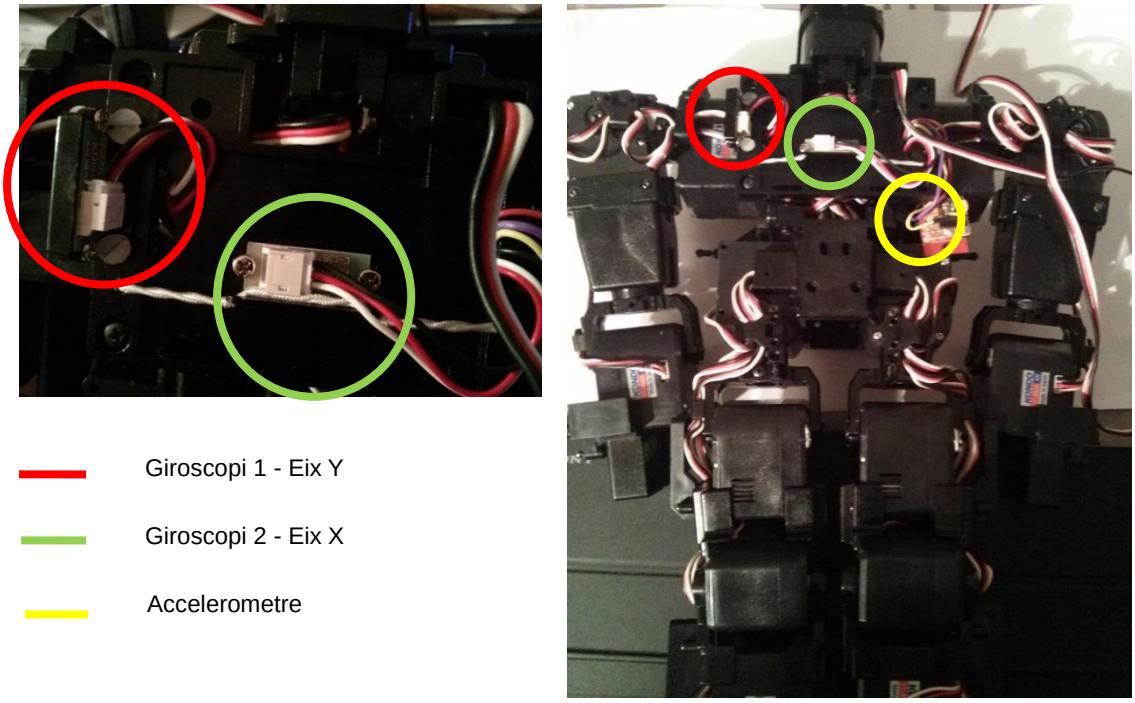


Figura 37. Vista detall i general de la instal·lació dels giroscopis.

[2]Aquesta instal·lació es temporal sobre l'esquelet del robot.

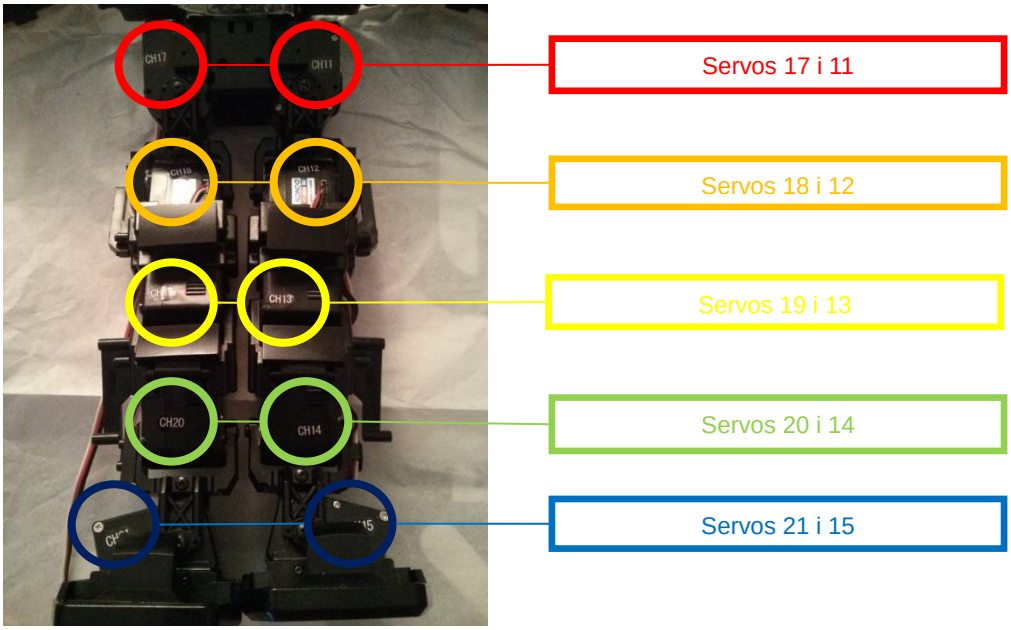


Figura 38. servos localitzats a la cama.

En la següent taula es mostra quin és l'eix de rotació de cada servomotor:

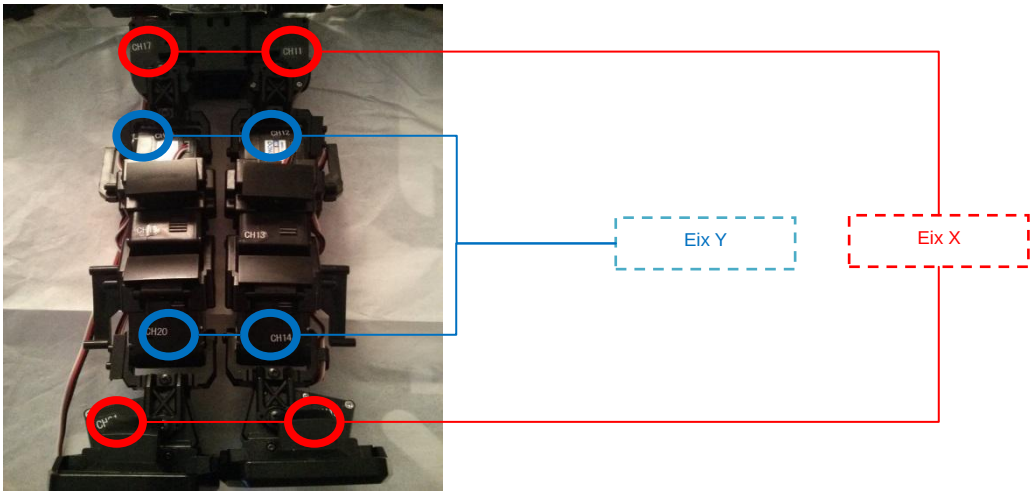


Figura 39. Relació Final Servos – Eixos.

Així s'emparellen els motors amb cadascun dels giroscopis d'acord amb el seu eix de rotació, és decideix utilitzar els servos 11-17 i 15-21 per guanyar estabilitat sobre l'eix X i el 12-18 i 14-20 per guanyar estabilitat sobre l'eix Y.

Servos	Eix de Rotació	Giroscopi
11 – 17	X	2
12 – 18	Y	1
13 – 19	Y	1
14 – 20	Y	1
15 – 21	X	2

Taula 14. Relació Numero de Servo- Eix de Rotació – Giroscopi.

Finalment s'ha de seleccionar quin multiplicador s'ha d'aplicar per cadascun dels servos mitjançant el programa de configuració de la placa. Per seleccionar aquest valor, primer es mostra en la figura 40 una aproximació de l'esquema de blocs que representa tota aquesta implementació:

Es tracta d'un mecanisme de control automàtic amb un control integral amb realimentació. És poden identificar cadascun dels elements que apareixen en el diagrama de blocs:

- **Referència**: És el valor de la referència que volem que es mantinguin el robot, aquest valor es 0 per definició, es vol que el robot mantingui el valor de la sortida dels giroscopis a 0, això voldrà dir que es manté en posició estable.
- **Ki**: És la constant del control integral, el valor d'aquesta constant es possible modificar-la dins la placa RCB3-J.(És pot identificar com al Multiplicador de la figura 31)
- **S1**: Funció de transferència d'un servomotor
- **S2**: Funció de transferència de tot el robot

El fet d'afegir el controlador integral fa que el sistema comenci a oscil·lar o fins i tot sigui no estable quan s'augmenta massa el valor de la variable Ki. El més pràctic en aquestes situacions per calcular el valor dels paràmetres es fer-ho empíricament fins veure que el robot és manté estable davant una pertorbació. En aquest cas i després de varies proves es decideix utilitzar els següents paràmetres:

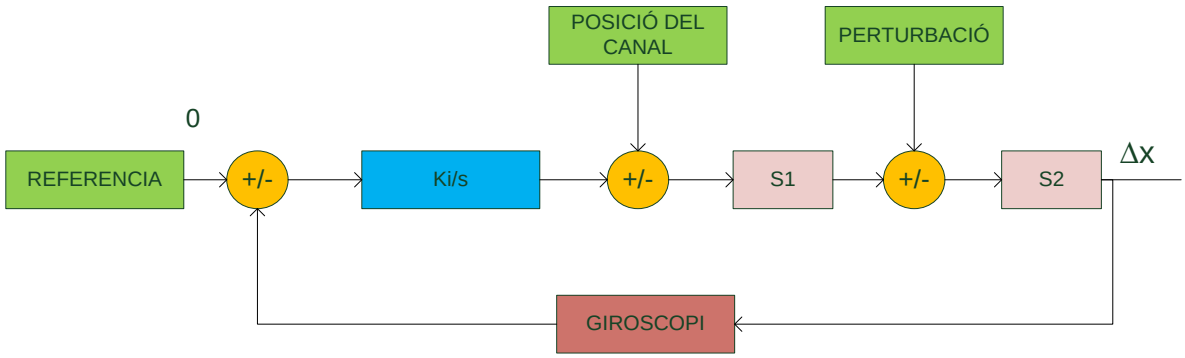


Figura 40. Esquema de blocs.

Numero de Servo	Giroscopi	Multiplicador
11-15	2	x-7
17-21	2	x-7
12-18	1	x-6
14-20	1	x12

Taula 15. Relació Multiplicador – Servo.

Mitjançant el programa es selecciona les propietats del ports analògics mitjançant  i seleccionem el multiplicador seleccionat per cadascun dels canals:

*Figura 41. Configuració **Mixing** placa RCB3-J.*

3.3.4. Comunicació amb la placa

Per les necessitat del projecte el software HtoH3 és insuficient principalment perquè una aplicació no es possible integrar-la dins d'un altre aplicació, per tant s'haurà de veure de quina manera s'estableix la comunicació entre el PC i el Robot per escriure una llibreria i poder-la utilitzar en altres projectes. L'aplicació està formada per les següents capes:

GUI or Graphic Interface
Communications Layer
USB Serial Driver

Taula 16. Capes de l'aplicació HtoH3.

Descartant la part de la interfície gràfica es pot veure que per tal de poder establir comunicació amb la placa s'ha d'entendre com funciona la capa de comunicacions i reescriure-la adaptada a les necessitats del nou programa, lògicament tampoc caldrà reescriure uns controladors sèrie-USB perquè amb una mica de sort ja es trobaran compilats per cada sistema operatiu.

El controlador utilitzat per establir una comunicació entre el PC i el robot és tal com s'ha descrit anteriorment, una comunicació sèrie sobre una connexió USB, mitjançant un analitzador de trameses pot analitzar l'intercanvii de missatges entre els dos objectes i per tant saber com desenvolupar la capa de comunicació, això però es tractaria d'una feina força intensa i per sort els desenvolupadors de la placa van publicar un document que descriu el protocol de comunicació entre el PC i la placa RCB3J[8].

Respecte les característiques de la comunicació sèrie, es tracta d'una comunicació sèrie amb les següents propietats:

Velocitat	115200 bps
Paritat	NONE
Stop Bit	1
Data	8 bits

Taula 17. Característiques comunicació Placa RCB3J.

El protocol de comunicació és molt senzill i únicament segueix 4 passos:

1.- En primer lloc el emissor envia el caràcter '0x0D' cap el robot

2.- El robot respon amb el caràcter '0x0D' per avisar que esta llest per rebre una comanda

3.- El PC envia la comanda cap al robot

4.- El robot respon a la comanda.

Cada comanda té un retorn concret que s'ha d'analitzar en cada cas. El control d'errors per les comandes és molt simple, es el total de la suma de cadascun dels caràcters que forma la trama enviada.

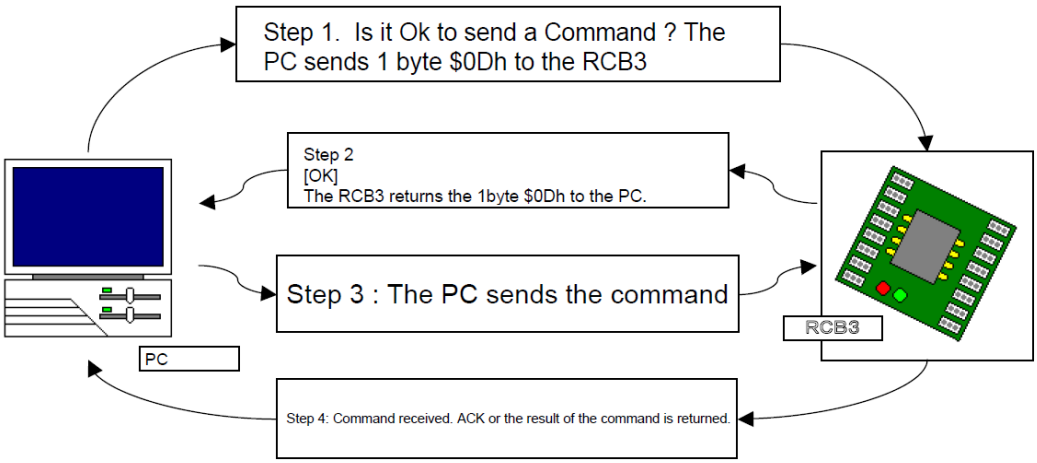


Figura 42. Comunicació entre el PC i la placa RCB3J.

Amb aquest protocol existeixen tota una seria de comandes que permeten realitzar totes

les operacions disponibles a la placa RCB3-J.

Totes les comandes contenen en primer lloc un 1 byte que indica la comanda que és vol executar, tot seguit una sèrie de bytes que son els arguments de la comanda i finalitzen amb la comprovació d'errors en la trama, la resposta de cada comanda no la podem estandarditzar doncs serà molt variable en funció de la comanda realitzada.

Tx:

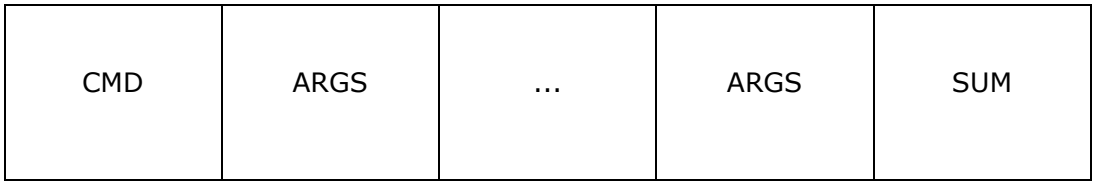


Figura 43. Trama de transmissió.

El control d'errors és realitza mitjançant la suma de tots els bytes que conformen la trama, excepte lògicament el mateix control d'errors, agafant sempre el byte més baix del resultat, en cas que passi de 1 byte.

A les següents taules s'ha disposat una descripció d'algunes de les comandes disponibles en la placa RCB3J:

RCB3J Versió			
Trama d'emissió:			
	CMD	CMD	0xFF
	SUM	SUM	Suma de Control
Trama de recepció:			
		FP1	1er byte Versió
		FP2	2on byte Versió
		...	
	SUM	SUM	Suma de Control

Taula 18. Trama de la versió de firmware.

Posició dels Motors:			
Trama d'emissió:			
	CMD	CMD	0xF0
	SUM	SUM	Suma de Control
Trama de recepció:			
	Posició	P1H	Posició Motor 1 byte Alt
		P1L	Posició Motor 1 byte Baix
		P2H	Posició Motor 2 byte Alt
		P2L	Posició Motor 2 byte Baix
		...	
		P24H	Posició Motor 24 byte Alt
		P24L	Posició Motor 24 byte Baix
	SUM	SUM	Suma de control

Taula 19. Trama Posició dels motors.

Executa un Moviment: (Gravat en la placa)			
Trama d'emissió:			
	CMD	CMD	0xF4
	ARGS	OPT	Opcions d'enviament
		NUM	Nº Moviment (0-79) o Escenari (80-84)
	SUM	SUM	Suma de Control
Trama de recepció:			
		ACK	0x06

Taula 20. Trama Executa un moviment.

Moviment dels Motors:			
Trama d'emissió:			
	CMD	CMD	0XFE
	ARGS		
		OPT	Opcions d'enviament
		PT	Motor que és vol moure (0x00 a 0x17)
		SPD	Velocitat del moviment
		PL, PH	Paràmetres dels moviments
	SUM	SUM	Suma de Control
Trama de recepció:			
		ACK	0x06

Taula 21. Trama Executa un Moviment.

Canviar dada del receptor			
Trama d'emissió:			
	CMD	CMD	0XDE
	ARGS	OPT	Opcions d'enviament
		RCV1	Sobreescriu dades
		RCV2	Byte alt de botó
		RCV3	Byte baix de botó
		RCV4	1 byte analògic
		RCV5	1 byte analògic
		RCV6	1 byte analògic
		RCV7	1 byte analògic
	SUM	SUM	Suma de Control
Trama de recepció:			
		ACK	0x06

Taula 22. Trama Canvi dades del receptor.

Setup dels interruptors de la placa			
Trama d'emissió:			
	CMD	CMD	0xF2
	ARGS	OPT	
		SWL	Opcions
		SWH	Opcions
	SUM	SUM	
Trama de recepció:			
		ACK	0x06

Taula 23. Setup del software.

En aquesta comanda SWL i SWH tenen el següent significat:

SWL	
0	Notificació periòdica del valor dels voltatges analògics. ON: 0 OFF: 1
1	Notificació de la finalització de l'execució d'un moviment. ON: 0 OFF: 1
2	Execució de moviment quan al robot li queda poca bateria. ON: 0 OFF: 1
3	Execució de moviment quan s'encén el robot. ON: 0 OFF: 1
4	Execució de moviment amb l'interruptor d'inici. ON: 0 OFF: 1
5	Habilitar el comandament a distància. ON: 0 OFF: 1
6	Mode ICS. ON: 0 OFF: 1
7	Sempre 0.

Taula 24. Byte Baix Configuració Switch.

SWH	
0	Sleep Mode. ON: 0 OFF: 1
1	Inhabilitar la interpolació amb el moviment ON: 0 OFF: 1
2	Rebre notificacions periòdiques del receptor ON: 0 OFF: 1
3	Setup de la connexió de dades serial. 0: Extended 1: Low Speed
4	Sempre 0
5-7	Sense significat en el nostre cas

Taula 25. Byte Alt configuració Switch.

4. Els capturadors de dades

4.1. Introducció

Els dispositius electrònics que s'han requerit per tal de considerar un dispositiu com a capturador de dades son:

- No sigui una solució feta a mida
- La connexió amb el receptor sigui sense fils.

Els elements de que han de disposar aquests capturadors de dades son:

- Botons
- Sensors

Amb aquestes especificacions les opcions comercials disponibles no son gaires, en primer lloc han de disposar d'una interfície de connexió sense fils, i a més hauran de disposar de botons i/o sensors. Els sensors disponibles haurien de ser sensors de rumb que ens donaran la orientació en relació a una referència fixa, per exemple un acceleròmetre, un giroscopi i un magnetòmetre, amb aquests prerequisits s'han seleccionat les següents solucions:

- ***Wiimote***
- **Telèfon mòbil**

4.1.1. Wiimote

El Wiimote es el controlador principal sense fils de la consola **Wii**. Està pensat per oferir a l'usuari tota una experiència lúdica gracies a la detecció de moviments que realitza. Físicament és pot veure que el Wiimote conte una sèrie de botons, un port d'expansió, un altaveu i un detector de infrarojos:

Figura 44. Wiimote.

Inclou els següents botons:

- Creu de control
- Botó A
- Botó B (en la part posterior)
- Botó -
- Botó +
- Botó home
- Botó 1
- Botó 2

4.1.2. Telèfon mòbil

Els actuals telèfons mòbils actuals també s'adeqüen a les solucions buscades, els models de telèfons mòbils que hi ha al mercat es molt gran però bàsicament qualsevol telèfon modern inclou com a mínim un acceleròmetre i a partir d'aquí es pot disposar a més d'un giroscopi i/o d'un magnetòmetre.

Figura 45. Telèfon Mòbil.

A més dels sensors en el telèfon mòbil es té la possibilitat de crear aplicacions i per tant es possible utilitzar la pantalla per programar botons. Normalment a més el telèfon incorpora no només la connexió per xarxa sinó que també com en el cas del Wiimote ofereix una connexió Bluetooth.

Veiem les principals característiques dels dos capturadors en una taula:

Característiques	Wiimote	Mòbil
Botons	Si	Si
Acceleròmetre	Si	Si
Gravetat	No	Si
Acceleració Lineal	No	Si
Giroscopi	Si(1)	Si(2)
Magnetòmetre	No	Si(2)
Programable	No	Si(2)
Bluetooth	Si	Si(2)
Network	No	Si(2)

Taula 26. Característiques Capturadors de Dades.

(1) Amb ampliació

(2) Segons el mòbil

4.1. Els Sensors dels Capturadors de dades

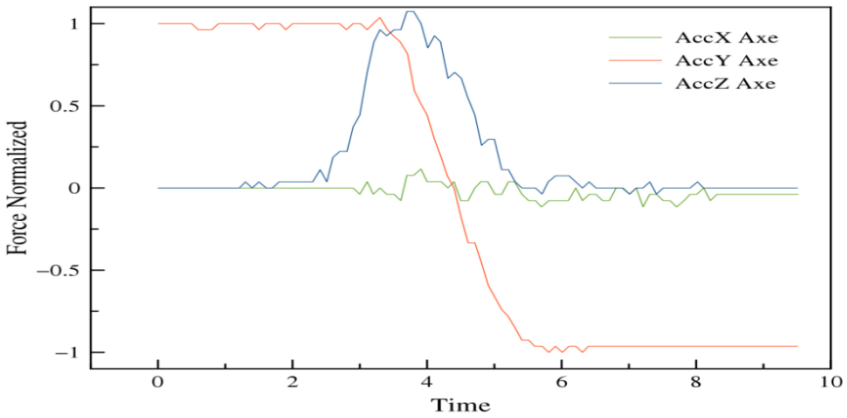
4.1.2. Acceleròmetre

Ambdós dispositius disposen de dos acceleròmetres:

- **Wiimote**: **ADXL330** manufacturat per Analog Devices.
- **Nexus4**: **InvensenseMPU-6050**.

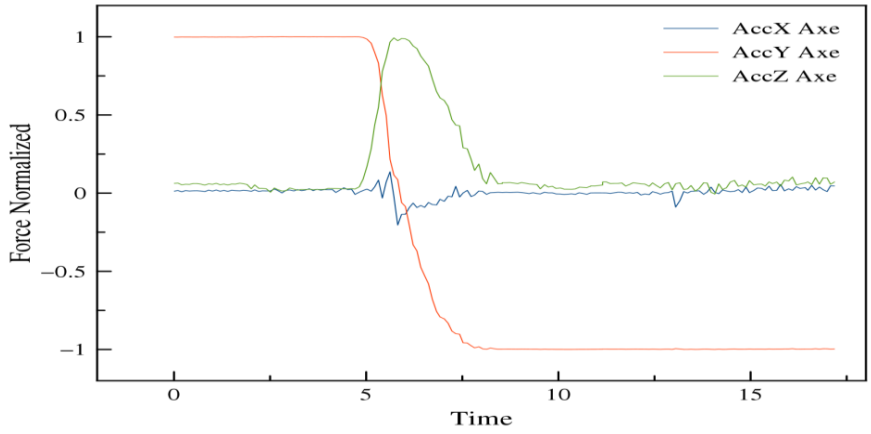
A continuació es poden veure algunes gràfiques amb les sortides que obtenim de l'acceleròmetre amb els valors normalitzats, és a dir: +g = 1, -g = -1.

Per exemple un moviment de (0,+1,0) a (0,-1,0) girant sobre l'eix X amb el Wiimote ens dona les següents gràfiques:



Gràfica 1. Sortida Acceleròmetre Wiimote.

S'observa com la variació de la lectura és correspon amb el moviment descrit, es pot veure que el moviment no ha estat especialment brusc, uns 180º en menys de 2 segons i tot i això podem observar clarament la quantitat de soroll que s'afegeix a la mesura. Amb el mateix moviment de (0,+1,0) a (0,-1,0) girant sobre l'eix Z en aquest cas amb el telèfon mòbil:



Gràfica 2. Sortida Acceleròmetre Mòbil.

Tot i que encara hi ha força soroll, aquest és molt menor.

Els sensors del Wiimote no discriminen la mesura de la gravetat en les lectures de l'acceleròmetre, així es pot fer una aproximació per tal de discriminar-la per sobre de la força provocada pel propi moviment, filtrant la senyal amb un filtre passabaixes, on les freqüències més baixes correspondran a canvis donats per la component de la gravetat mentre que els canvis més ràpids corresponen a la força del moviment.

Calculant la funció de transferència d'un filtre passa-baixes de primer ordre:

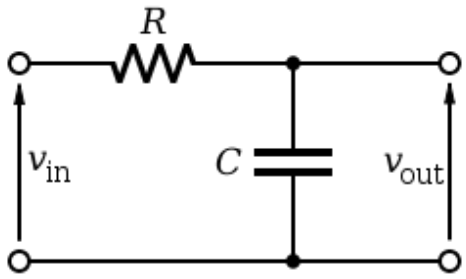


Figura 46. Circuit RC.

$$vin(t) - vout(t) = RC \cdot \frac{dvout}{dt} \quad \text{Eq.2}$$

$$xi - yi = RC \cdot \frac{y_i - y_{i-1}}{\Delta t} \quad \text{Eq. 3}$$

$$y_1 = x_i \cdot \left(\frac{\Delta t}{RC + \Delta T}\right) + y_{i-1}\left(\frac{RC}{RC + \Delta T}\right) \quad \text{Eq. 4}$$

En aquesta equació hi ha dues contribucions:

La contribució a la sortida per el valor actual:

$$x_i \cdot \left(\frac{\Delta t}{RC + \Delta T}\right)$$

I la contribució afegida per la inèrcia:

$$y_{i-1}\left(\frac{RC}{RC + \Delta T}\right) \quad \text{Eq. 5}$$

La equació anterior es pot reescriure com:

$$y_i = \alpha x_i + (1 - \alpha) y_{i-1} \quad \text{Eq. 6}$$

On:

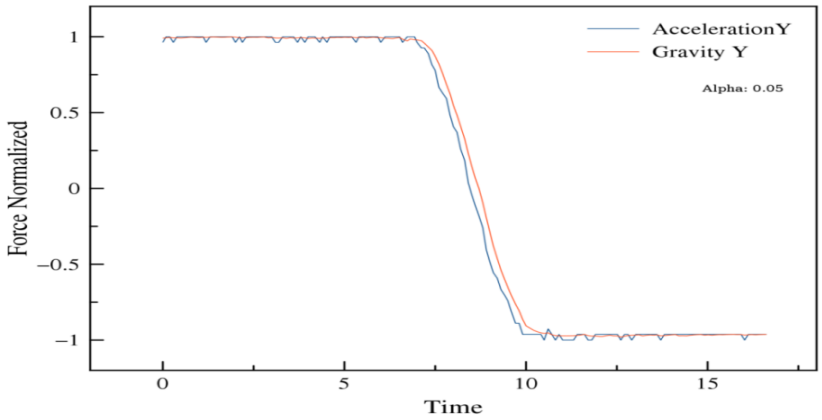
$$\alpha \cong \frac{\Delta T}{RC + \Delta T} \quad \text{Eq. 7}$$

Per tal de treure el valor de α s'ha de conèixer quina és la freqüència de mostreig i quina es la freqüència de tall del filtre amb el qual podrem treure la relació RC mitjançant la formula:

$$f_c = \frac{1}{2\pi \cdot RC} \quad \text{Eq. 8}$$

Els valors filtrats doncs es correspondran amb els valors de la gravetat.

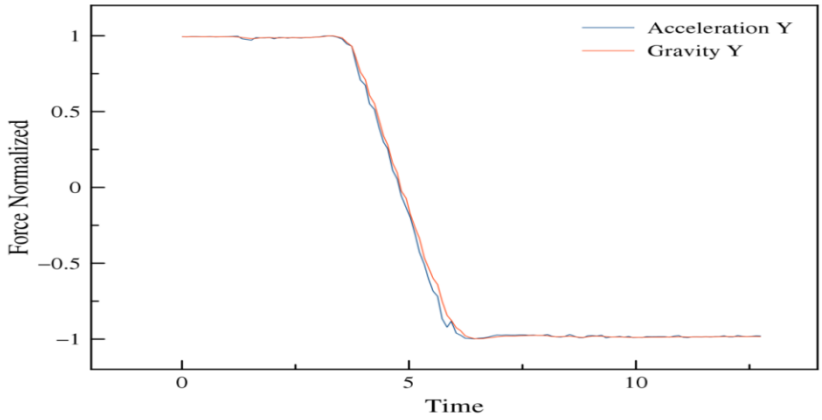
Disposant d'una mostra cada 20 ms i amb una freqüència de tall de 0.5Hz obtenim una α d' aproximadament 0.05; Veiem una gràfica amb la sortida de l'acceleròmetre de l'eix Y i el càlcul de la gravetat:



Gràfica 3. Acceleració VS Gravetat Wiimote.

Podem observar com molt del soroll que tenim l'aconseguim eliminar, per contra el filtre provoca que la senyal de gravetat sigui encara més lenta que la sortida de l'acceleròmetre.

Per altre banda podem comparar les dades de la nostre gravetat amb la que llegim del sensor de gravetat del telèfon mòbil i es pot comprovar que llegim en el mòbil es molt millor, a més que no afegeix cap retard:



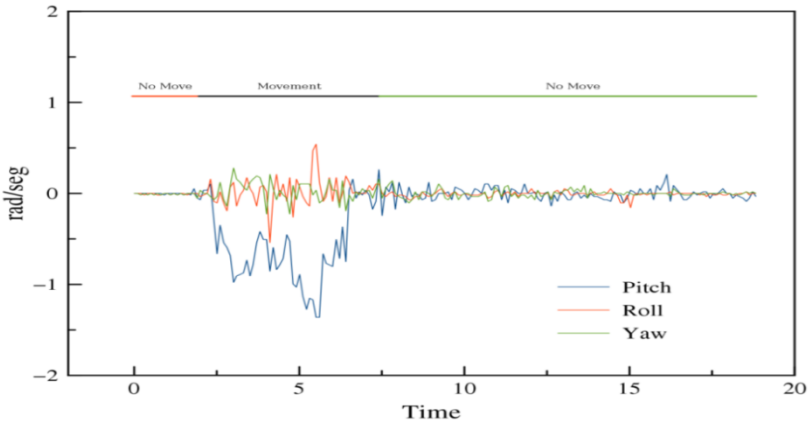
Gràfica 4. Acceleració VS Gravetat Mòbil.

4.1.3. Giroscopi

Els models dels giroscopis en els capturadors de dades seleccionats son:

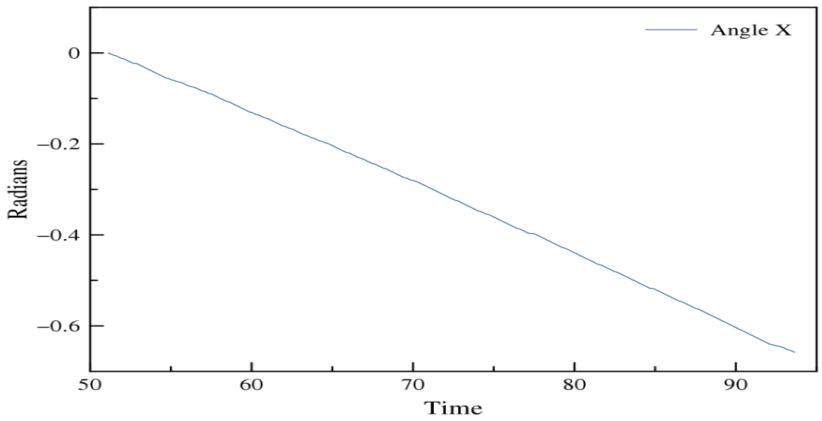
- **Wiimote**: El nom comercial és **Wiimotion** Plus i conté 2 giroscopis:
  - **IDG-600**per capcineig (pitch) i el balanceig(roll)
  - **Epcon ToyocomX3500W** per el càlcul de la guinyada (yaw).
- **Nexus 4**: **Invensense MPU-6050**

Observem a continuació la sortida de cadascun dels giroscopis, en el cas del Wiimote:
Es realitza un moviment igual a l'anterior des de (0,+1,0) a (0,-1,0) (gir sobre l'eix X):



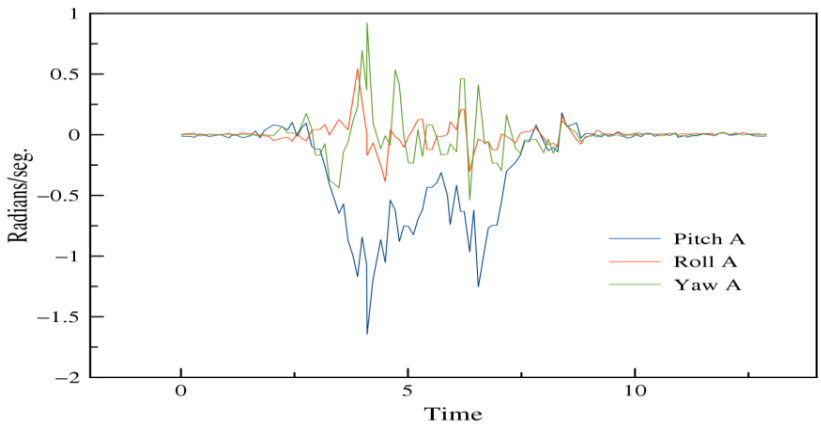
Gràfica 5. Sortida Giroscopi Wiimote.

S'han dividit les dades de la mesura en tres trams, on només es realitza una rotació sobre l'eix Y en el tram central. Mantenint el giroscopi fixa sobre una superfície e integrant el seu resultat es pot veure la gràfica del ARW, és a dir la variació de la lectura del giroscopi en repòs:



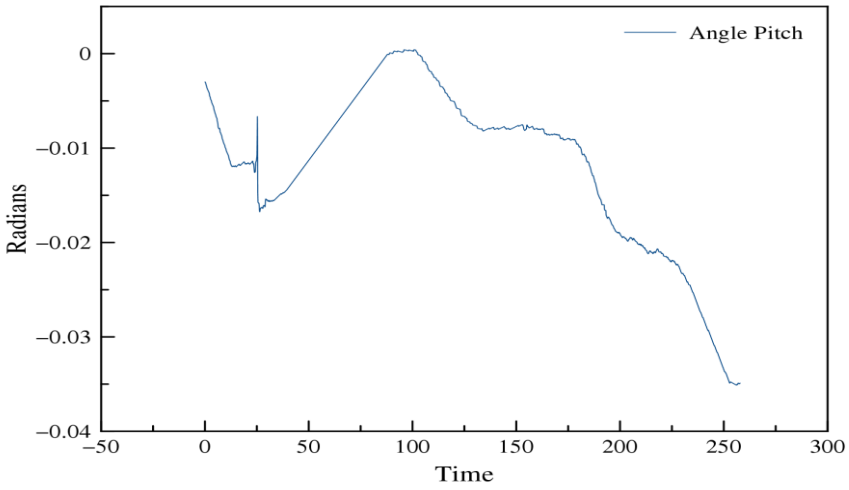
Gràfica 6. ARW Wiimote.

Llegint les dades del giroscopi del mòbil amb la mateixa rotació sobre l'eix X:



Gràfica 7. Sortida Giroscopi Mòbil.

La idea és la mateixa que abans el moviment sobre l'eix Y provoca un moviment majoritari sobre la variació de l'angle de l'eix Y,
També es pot analitzar l'**ARW** del giroscopi del mòbil:

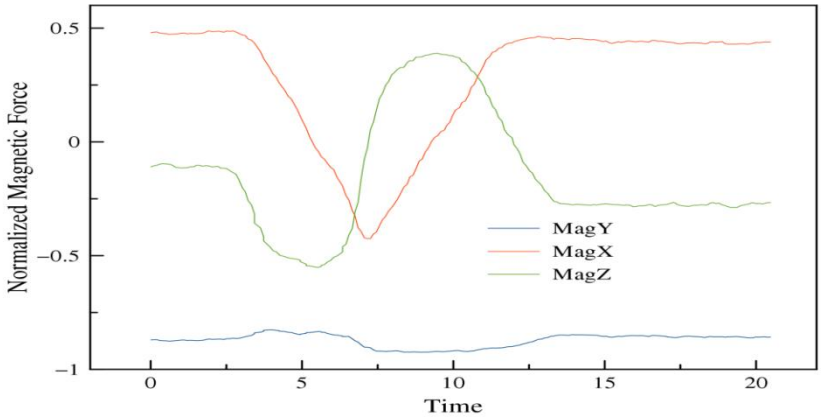


Gràfica 8. ARW Mòbil.

*L'**ARW*** del mòbil és molt millor que el que ofereix el giroscopi del Wiimote, amb les gràfiques es pot observar una deriva de 0.80º/segon per el Wiimote mentre que en el mòbil no arriba a 0.015º/seg.

4.1.4. Magnetòmetre

Els magnetòmetres son dispositius que mesuren la força i/o la direcció d'un camp magnètic, actualment son utilitzats en dispositius portàtils com mòbils i tableted per obtenir informació sobre la orientació del dispositiu. En el cas que ens ocupa només el telèfon mòbil disposa de magnetòmetre, en aquest cas un **AKM**, a continuació es poden veure les gràfiques de sortida del dispositiu amb una rotació de 360 graus sobre l'eix Y:



Gràfica 9. Sortida Magnetòmetre Mòbil.

5. Connexió amb la placa RCB3J

Amb anterioritat s'ha descrit el protocol de comunicació establert per tal de comunicar-se amb la placa RCB3J, en aquesta secció es descriurà la creació d'una llibreria per tal de plasmar aquest protocol i permetre la comunicació entre un client i la placa RCB3J, així les aplicacions desenvolupades podran ser compilades incloent la llibreria per tal de fer moure els robot.

5.1. Visió general

Un cop es connecta el robot a un port USB, en un S.O. Linux s'obté la següent informació en el log del sistema utilitzant **dmesg**:

```
[ 4069.097101] usb 2-1.2: new full-speed USB device number 6 using ehci-pci
[ 4069.197413] usb 2-1.2: New USB device found, idVendor=165c, idProduct=0002
[ 4069.197419] usb 2-1.2: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[ 4069.197423] usb 2-1.2: Product: SERIAL USB ADAPTER
[ 4069.197426] usb 2-1.2: Manufacturer: Kondo Kagaku
[ 4069.197430] usb 2-1.2: SerialNumber: KOP2O6G7
[ 4069.479586] usbcore: registered new interface driver usbserial
[ 4069.479605] usbcore: registered new interface driver usbserial_generic
[ 4069.479620] usbserial: USB Serial support registered for generic
[ 4069.505468] usbcore: registered new interface driver ftdi_sio
[ 4069.505497] usbserial: USB Serial support registered for FTDI USB Serial Device
[ 4069.505784] ftdi_sio 2-1.2:1.0: FTDI USB Serial Device converter detected
[ 4069.505853] usb 2-1.2: Detected FT232BM
[ 4069.505858] usb 2-1.2: Number of endpoints 2
[ 4069.505862] usb 2-1.2: Endpoint 1 MaxPacketSize 64
[ 4069.505866] usb 2-1.2: Endpoint 2 MaxPacketSize 64
[ 4069.505869] usb 2-1.2: Setting MaxPacketSize 64
[ 4069.506363] usb 2-1.2: FTDI USB Serial Device converter now attached to ttyUSB0
```

Es pot veure que la clau USB que es connecta amb el PC es tracta d'un convertidor USB-Serie, el circuit de control és el **FT232BM**, també es mostra informació molt interessant respecte al dispositiu connectat, per exemple quin és el número de sèrie i fabricant del dispositiu i també els diferents identificadors del dispositiu o a quin fitxer del sistema operatiu a estat associat.

5.2. Establir connexió

5.2.2. Connexió utilitzant el fitxer de dispositiu

Per tal d'utilitzar el dispositiu no es necessari l'ús de cap llibreria, un cop un nou dispositiu es connectat i reconegut el sistema operatiu Linux crea automàticament un fitxer que representa el dispositiu a la carpeta **/dev**, les lectures sobre aquest fitxer seran enviaments de dades cap al dispositiu i les lectures permetran obtenir dades des de el dispositiu, d'aquesta manera la llibreria escriu i llegeix dades sobre el fitxer que simbolitza el controlador dins el sistema operatiu.

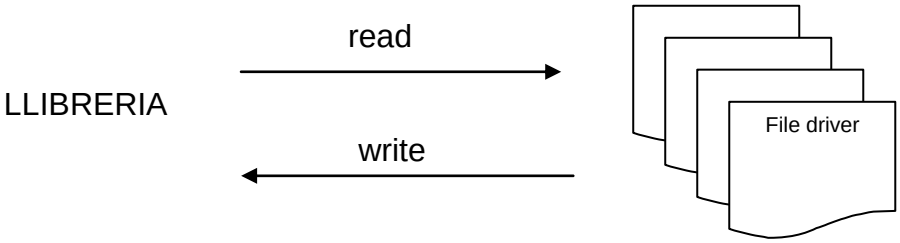


Figura 58. Ús dels controladors en l'aplicació.

El següent codi escrit en C, permet establir una connexió amb el dispositiu mitjançant el seu fitxer a /dev:

```
int fd;
//obrim el fitxer de connexió de lectura i escriptura
fd = open(devicename,O_RDWR);
struct termios tio;
memset(&tio,0,sizeof(tio));
//Opcions de connexió
tio.c_cflag = CS8 | CLOCAL | CREAD;
//Opcions de timeout
tio.c_cc[VTIME] = 0;
tio.c_cc[VMIN] = 1;
//Setejem la velocitat de connexió del port sèrie
cfsetispeed(&tio, baudrate);
cfsetospeed(&tio, baudrate);
tcsetattr(fd,TCSANOW,&tio);
```

Un cop la connexió el fitxer ha estat obert la connexió amb el dispositiu ja estat establerta, per tant ja es poden fer lectures i escriptures al fitxer-dispositiu de la següent manera:

Per escriure:

```
nbytesread = read(fd, *bytes, nbytes);
```

Per llegir:

```
nbyteswrite = write(fd, *buffer, nbytes);
```

Un cop finalitzat l'ús del dispositiu únicament s'ha de tancar la connexió amb el dispositiu, tancant el descriptor del fitxer:

```
close(fd);
```

5.2.3. Connexió programàticament

En comptes d'utilitzar el fitxer del dispositiu que crea el S.O. es poden utilitzar llibreries especifiques per a la comunicació amb diferents dispositius, com en aquest cas un dispositiu USB a sèrie FTDI. Una de les possibles solucions seria compilar la nostre llibreria

utilitzant els controladors del fabricant, per sort fent una cerca al repositori de Debian es pot trobar que també existeix una llibreria que permet realitzar operacions amb alguns dispositius FTDI, la llibreria **libftdi-dev**. La documentació i referències a aquesta llibreria es pot trobar a:

http://www.intra2net.com/en/developer/libftdi/index.php

Tal com s'explica a la pàgina web la llibreria permet establir comunicació amb dispositius FT232BM, FT245BM, FT2232C, FT2232D, FT245R, FT232H i FT230X , per tant no hi haurà cap problema en principi per comunicar-se amb el xip **FT232BM** que esta inclòs en la clau USB i que es el que realitza la traducció Sèrie-USB.

Com s'ha vist amb l'apartat anterior, per tal d'obrir una connexió amb el dispositiu s'hauran de realitzar les següents passes:

```
//init usb
ftdi_init(&ftdic);
//select first interface
ftdi_set_interface(&ftdic, INTERFACE_ANY);
// open usb device
ftdi_usb_open(&ftdic,0x165c, 0x0002);
// Set Connection Speed
ftdi_set_baudrate(&ftdic, 115200);
//Set Connection Properties
ftdi_set_line_property(&ftdic, BITS_8, STOP_BIT_1, EVEN);
```

Tal com es veu per obrir el dispositiu es necessiten dos números, aquests dos números identifiquen el venedor i el producte que s'estan buscant i per sort son fàcilment accessibles al log del sistema quan connectem la clau USB:

New USB device found, **idVendor=165c, idProduct=0002**

Un cop connectat es poden escriure i llegir dades del dispositiu:

```
nbytesread = ftdi_read_data(&ftdic,*bytes, nbytes);
nbyteswrite = ftdi_write_data(&ftdic,*buffer, nbytes);
```

Es pot trobar una relació complerta de totes les funcions de la llibreria disponible a:

http://www.intra2net.com/en/developer/libftdi/documentation/group__libftdi.html

5.3. Funcionalitat de la llibreria

La llibreria de connexió de moment ofereix les següents funcions per interactuar amb el robot tot i que és fàcilment ampliable amb noves comandes. És poden distingir diferents ordres en funció de la funcionalitat a que refereixen, així estan disponibles comandes que

permeten la connexió amb el robot i també obtenir informació de la placa RCB3J:

int serialInit()
Inicialització de la connexió entre la placa i el host
char* getVersion(void)
Retorna les dades de la versió del firmware de la placa RCB3J
bool isConnected(void)
Retorna si la connexió entre el robot ha estat establerta

Taula 27. Comandes de Connexió.

Abans de iniciar qualsevol interacció amb el robot s'ha de realitzar la comanda **serialInit** per tal d'inicialitzar la connexió amb la placa.

Altres modifiquen la posició dels motors del robot:

void moveAbsoluteMotor(unsigned short motoridentifier, float absolutepositionangle)
Mou el motor (motoridentifier) a la posició absoluta (absolutepositionangle). La informació de (absolutepositionangle) ha de ser en graus.
int moveRelativeMotor(unsigned short motoridentifier, float relativeposition)
Mou el motor (motoridentifier) a la posició relativa (relativeposition). La informació de (relativeposition) ha de ser en graus, aquest mètode per conèixer quina es la posició actual del motor, li pregunta a la placa.
int moveRelativeMotorNotAsking(unsigned short motoridentifier, float relativeposition)
Mou el motor (motoridentifier) a la posició relativa (relativeposition), la informació de relativeposition ha de ser en graus, aquest mètode particularment utilitza la pròpia llibreria per conèixer quina es la posició actual del motor, modificant el valor de la posició del motor cada cop que es fa un moviment
int getMotorPosition(int motoridentifier)
Retornala posició actual del motor (motoridentifier)

Taula 28. Comandes de moviment dels motors.

La placa representa la posició d'un motor amb 2 bytes, però a la llibreria se li passen les mesures en angles, així s'han de poder transformar aquests angles per això s'ha d'utilitzar una regla de tres, modificant la posició de 0º a 90º i mirant quina és la lectura que s'obté de cadascun dels motors amb la placa RCB3-J, sabent que la posició a 0º sempre serà de 16384:

Canal	Graus	Posició Placa
1	90º	16566
2	90º	16573
3	90º	16559
5	90º	16567
6	90º	16573
7	90º	16599
8	90º	16593
	...	

Taula 29. Relació angle - Valor placa.

Finalment altres comandes per el llançament de moviments:

int launchMovement(unsigned short)
Llança un moviment pregravat en la placa RCB3-J
int changeReceiverValue(unsigned short value)
Canvia el valor del receptor al valor indicat per (value). Aquest és el valor utilitzat a l'exemple anterior del comandament per el AT01
int setBlockingMode(bool mode)
Activa o desactiva el mode de bloqueig
int cancelMovement(void)
Cancel·la l'execució de l'actual moviment

Taula 30 Comandes de llançament de moviments

El mode de bloqueig es una opció que ha estat programada per tal d'evitar que es llancin noves comandes en cas que el robot estigui executant un moviment actualment, funciona de la següent manera:
En primer lloc es modificar les opcions de la placa enviant una comanda de modificació dels interruptors de la placa, especificada a la taula 23, tal com indica s'ha de ficar el bit dos del byte SWL a 1, a partir d'aquí un cop es llanci un moviment i aquest finalitzi s'enviarà una trama de finalització de moviment de la placa fins al host. Sabent això el moviments seguiran el següent diagrama de flux:

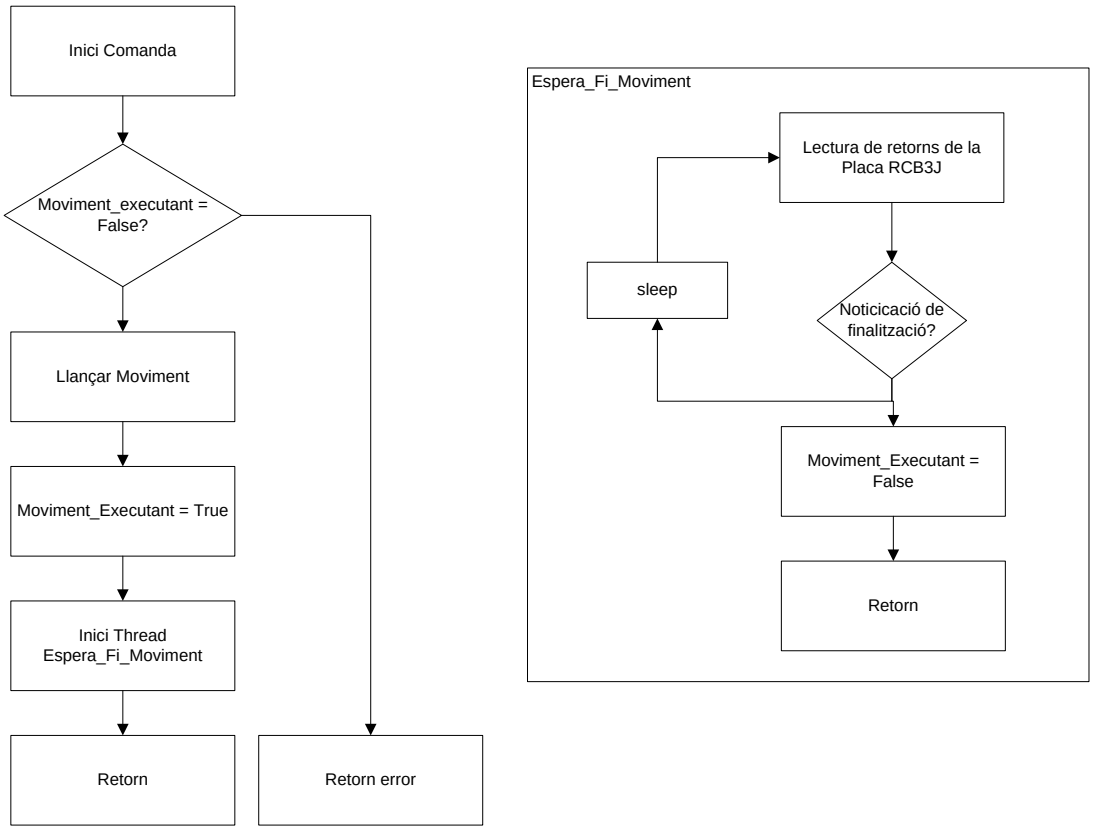


Figura 47. Diagrama flux en mode bloqueig.

Un cop es demana el llançament d'un moviment, es comprova s'està llançant algun moviment, si és així es retorna el error corresponent, en cas contrari es llança el moviment

i un nou fil d'execució que s'encarrega de comprovar periòdicament, llegint les trames que envia la placa RCB3J al host si es que el moviment ha finalitzat, si es així indica que es poden continuar enviant nous moviments a la placa.
Finalment amb aquest mòdul s'han realitzat el ***bindings*** de **python** per facilitar el test, com a exemple d'un moviment absolut es presenta el següent codi:

Exemple Moviment Absolut[9]:

```
#Importar la llibreria
import ManoiConnection
#Instanciar la classe
manoi = ManoiConnection.ManoiConnection()
#inicialitzar la connexió amb el robot
manoi.serialInitialization()
#Moure el motor a la direcció desitjada
manoi.absoluteMoveMotor(14,15400);
#Imprimir el valor actual de la posició del motor
print "value: %x\n"%manoi.getMotorPosition(14)
```

6. Llibreria dels Capturadors de dades

Com s'ha realitzat amb la connexió amb el robot, amb els capturadors de dades també s'ha desenvolupat una llibreria que permet establir connexió, obtenir dades i estats dels diferents capturadors de dades.

6.1. Descripció de la llibreria

La llibreria del capturador de moviments s'encarrega de tractar amb els diferents capturadors de dades que estan disponibles, ara mateix hi han definits 2 tipus de capturadors de dades, **Wiimotes** i el telèfons mòbils, principalment aquest mòdul s'ocupa de les següents funcions:

- Oferir una interfície per treballar amb tots els capturadors de dades
- Oferir una interfície idèntica per cadascun dels tipus de capturadors de moviments
- Deserialitzar les dades que obtenim per cadascun dels canals.

El capturador de moviments té la següent estructura:

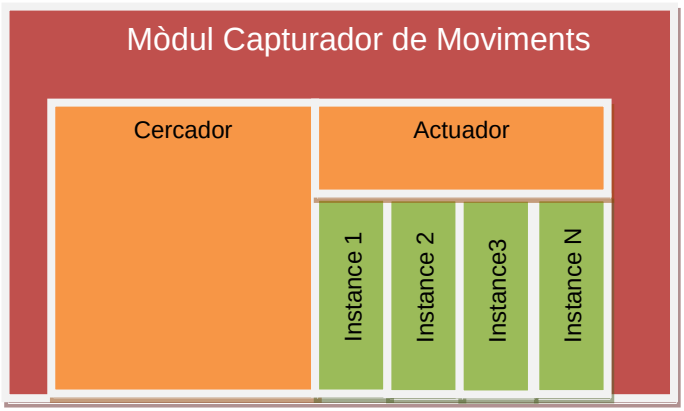


Figura 48. Blocs del Mòdul de captura de Moviments.

El capturador té dos vessants el **buscador** i l'**actuador**. El **buscador** serveix per iniciar una cerca sobre la interfície del Bluetooth o de xarxa i trobar tots els dispositius que puguin ser un capturador de dades, l'**actuador** un cop finalitzada la cerca crea una instancia local de cadascun dels dispositius trobats que permet interactua amb ell.

Cada capturador de dades pot treballar de dos maneres, activa o passivament, activament vol dir que pot enviar ordres que podran ser interpretades per executar accions, per contra passivament voldrà dir que només ofereixen la possibilitat de d'accedir als valors dels botons i els sensors.

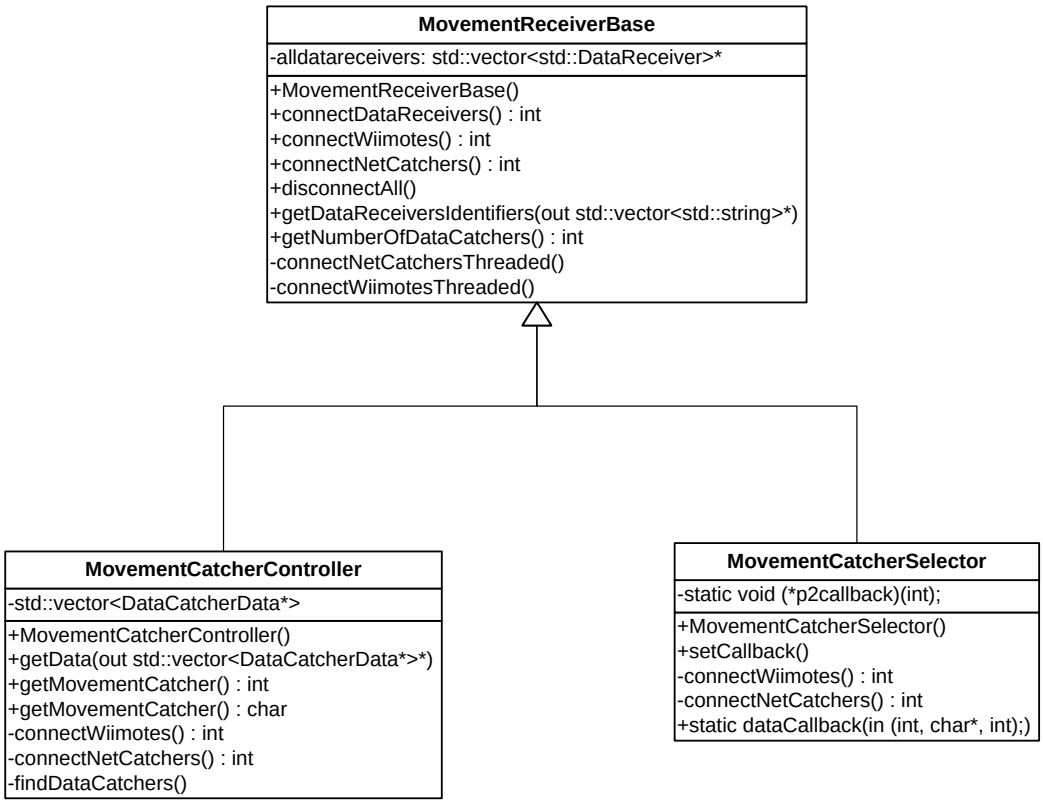


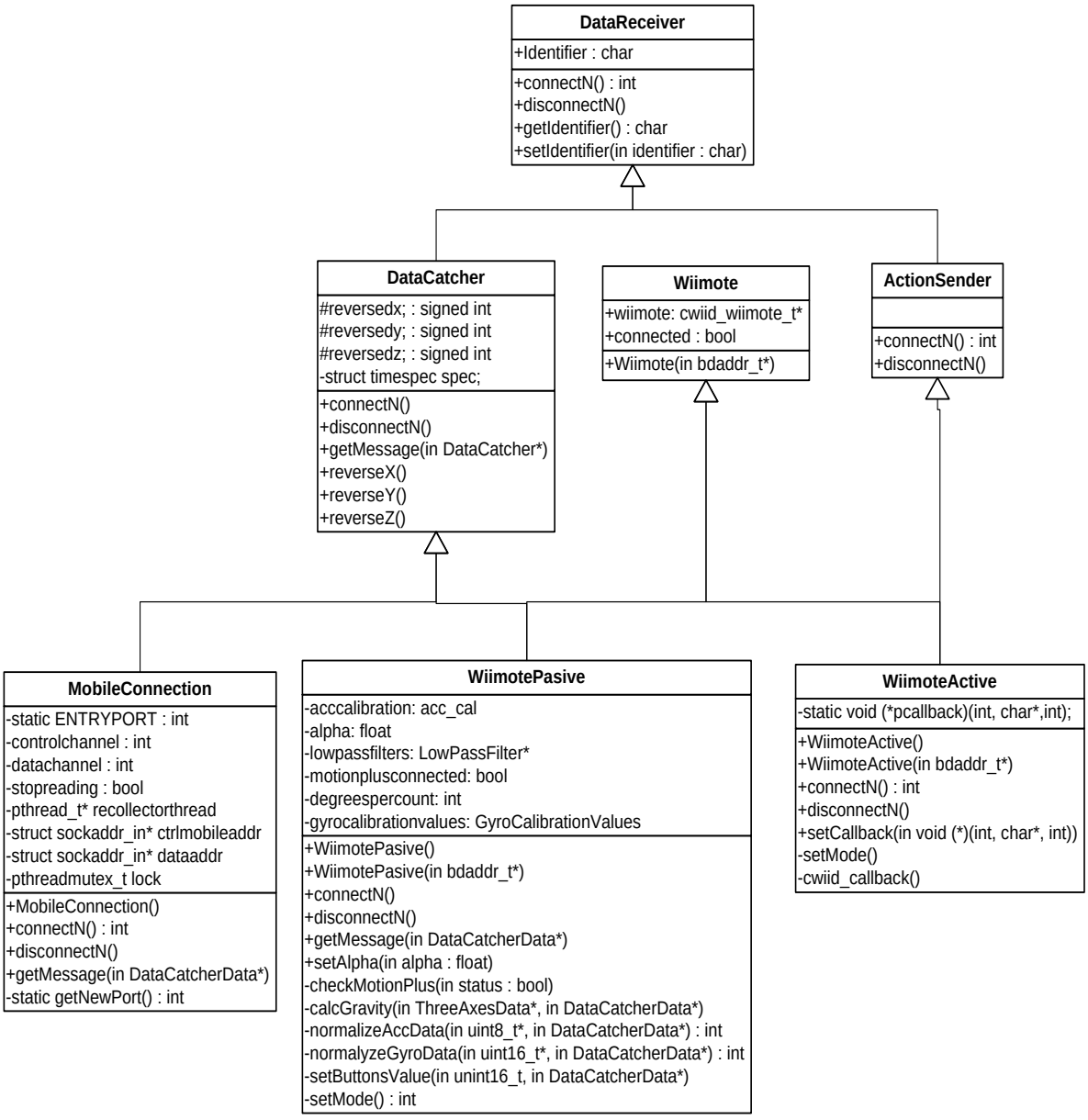
Figura 49 UML Manager dels Capturadors de Dades

El mòdul ofereix dos tipus de cercadors, els que busquen capturadors de dades passius anomenat **MovementCatcherController** i el que els busca de tipus actiu anomenat **MovementCatcherSelector**. Els dos funcionen d'una manera similar respecte al cercador, però un cop trobats els diferents capturadors de dades treballen de manera diferent, el primer permet accedir a cadascun dels capturadors de dades per tal d'obtenir les dades mentre que l'altre permet programar una funció ***callback*** que serà executada cada cop que es rebi un missatge d'un capturador de dades[3].

El cercador és capaç de buscar sobre les 2 interfícies, per la interfície Bluetooth s'utilitza el mòdul Select per fer una cerca asíncrona de tots els dispositius Bluetooth que estan al voltant, un cop finalitzada la cerca es filtren els resultats per destriar aquells que ja han estat prèviament trobats. El cercador de la interfície de xarxa, que s'utilitza per la connexió amb telèfons mòbils, per la seva part escolta sobre un port per identificar possibles dispositius d'enviament de dades durant uns segons, un cop finalitzada la cerca obre una connexió TCP amb el dispositiu, indicant-li que vol començar a rebre dades i un segon missatge indicant sobre quin port vol rebre les dades.

El diagrama UML dels capturadors de dades ens mostra com es relacionen les diferents classes que hi ha disponibles al mòdul,

[3]Per ara el **MovementCatcherSelector** només accepta un únic capturador de dades i de tipus Wiimote



Gràfica 10. Diagrama UML dels capturador de dades.

6.2. Connexió amb el Wiimote

La connexió entre el Wiimote i el PC servidor es realitza mitjançant Bluetooth que es una especificació mitjançant la qual es possible el intercanvi d'informació entre dos o més dispositius sense fils. Físicament Bluetooth utilitza la banda dels 2.4 GHz-2.48GHz tenint a la seva disposició un total de 79 canals amb un ample de banda de 1Mhz cadascun. La comunicació entre els dispositius es descentralitzada amb una estructura mestre-esclau on un dispositiu mestre pot comunicar-se amb fins a 7 esclaus formant el que es coneix com un ***piconet***. Un dispositiu Bluetooth té un identificador únic, semblant a la adreça física d'un dispositiu de xarxa, en el cas del Bluetooth conté un total de 48 bits, com per exemple: **00:03:C9:2D:B4:8F**

El model de capes de Bluetooth comparat amb el model de capes del model OSI el podem veure en el següent gràfic:

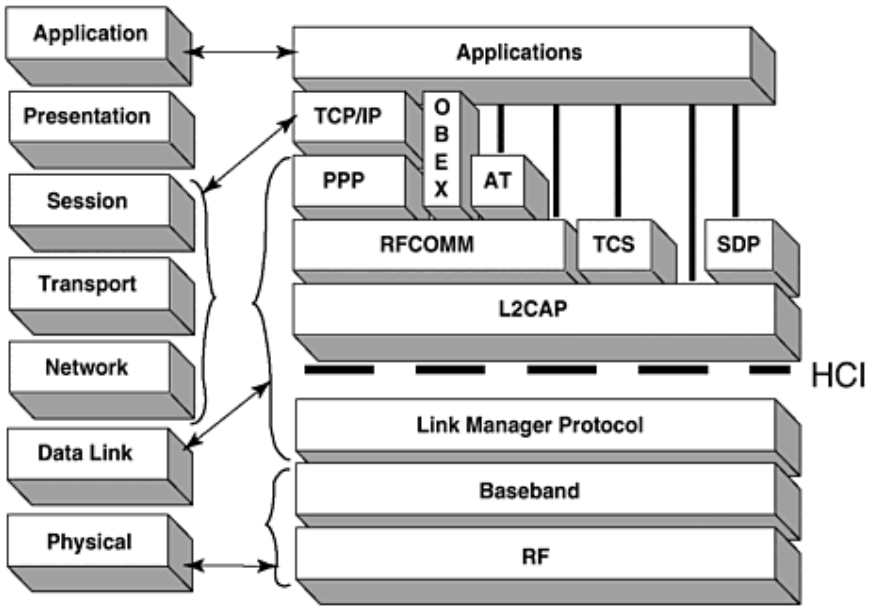


Figura 50. Capes del model de Bluetooth.

L'Stack de Bluetooth es utilitzat per el que s'anomenen perfils que no son més que una manera d'agrupar comportaments per unificar-los i fer-ne un ús i més eficient. Existeix una llarga llista de perfils que creix a mesura que va creixent l'especificació per exemple:

- Perfil de distribució avançada d'àudio(A2DP)
- Perfil de telefonia sense fils (CTP)
- Perfil de transferència de fitxers (FTP)
- Perfil de mans lliures (HFP)
- Perfil de dispositiu d'interfície humana (HID)
- Perfil d'auricular
- Perfil de port sèrie (SPP)
- Perfil de descobriment de serveis (SDAP)

El Wiimote utilitza dos perfils en primer lloc l'**HID** és a dir el de dispositius d'interfície humana i en segon lloc el SDP és a dir el de descobriment de dispositius. Un dispositiu HID (***Human Interface Device***) és aquell que interactua directament amb les persones, es podem trobar tot un ventall de dispositius que es troben dins aquesta categoria com per exemple:

- Dispositius d'adquisició de dades
- Dispositius de visualització
- Teclats, ratolí, joysticks...

Qualsevol innovació en el hardware, requereix sobrecarregar l'ús de dades d'un protocol existent o la creació d'un nou controlador i un nou protocol, això és una limitació molt gran i per això es va fer necessari la creació d'un controlador que fos suficientment flexible com

per permetre el seu ús amb tots aquests dispositius. D'aquí neix la necessitat i la creació del protocol **HID**, es tracta d'un protocol auto descriptiu, és a dir s'entreguen paquets que poden contenir molta varietat de dades i formats, i en el mateix protocol es descriuen la utilitat i format d'aquestes dades, així un sol controlador **HID** ens permet analitzar les dades i crear una associació dinàmica entre les dades d'entrada i sortida amb la funcionalitat de l'aplicació fent possible el seu ús amb molts tipus de dispositius diferents.

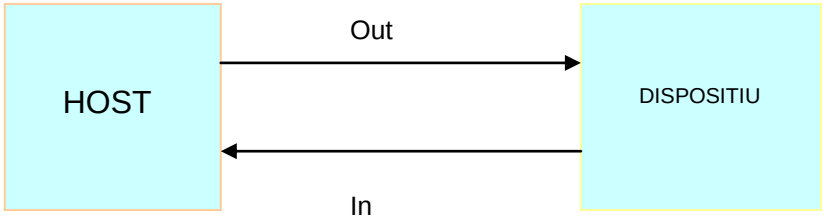


Figura 51. Direcció de les dades.

En el protocol HID existeixen 2 entitats, el host i el dispositiu. El dispositiu s'encarrega d'interactuar directament amb l'operador, mentre que el host es comunica amb el dispositiu reben les dades del dispositiu, en funció de les accions realitzades per l'operador, aquest a la vegada envia com a resposta dades al dispositiu, aquestes dades poden ser interpretades per l'operador que pot actuar en conseqüència a les dades rebudes. Donat que els dispositius HID interactuen amb persones no és necessari que el protocol tingui un ample de banda molt gran, així la màxima velocitat de transferència és de 64 Kb per segon.

La comunicació entre els dispositius es realitza mitjançant l'enviament del que s'anomena 'reports', existeixen tres tipus de paquets:

- Entrada

- Sortida

- De control

El dos primers son unidireccionals, i es diferencien per la direcció que tenen, els paquets d'entrada viatgen del dispositiu al host, mentre que els de sortida des del host cap al dispositiu, els paquets de control per la seva part son bidireccionals i serveixen per enviar dades de control, del host al dispositiu o en sentit contrari.

Un cop s'estableix la sincronització entre el host i el dispositiu, en primer lloc el Host rep del dispositiu el HID descriptor que conté tota la informació sobre els reports disponibles, aquesta informació es analitzada pel controlador del HID i genera una taula amb tota la informació de manera que sap exactament quin format tindran els reports que s'intercanviaran.

Exemple d'un HID descriptor:

Report descriptor d'un ratolí de 3 botons:

```
Usage Page (Generic Desktop),
        Usage (Mouse),
        Collection (Application),
                Usage (Pointer),
                Collection (Physical),
                        Report ID (0A), ;Make changes to report 0A
                        Usage (X), Usage (Y),
                        Logical Minimum (-127), ;Report data values range from -127
                        Logical Maximum (127), ;to 127
                        Report Size (8), Report Count (2),
                        Input (Data, Variable, Relative), ;Add 2 bytes of position data (X & Y)
                        Logical Minimum (0), ;Report data values range from -127
                        Logical Maximum (1), ;to 127
                        Report Count (3), Report Size (1),
                        Usage Page (Button Page),
                        Usage Minimum (1),
                        Usage Maximum (3),
                        Input (Data, Variable, Absolute), ;Add 3 bits (Button 1, 2 & 3) to report 0A
                        Report Size (5),
                        Input (Constant), ;Add 5 bits padding to byte align the report 0A
                End Collection,
        End Collection
```

Aquest HID descriptor ens defineix un format del report que seria el següent:

1 byte Report ID (0A)	1 byte Situació X	1 byte Situació Y	1 bit	1 bit	1 bit	5 bits a zero

Taula 31 Descriptor de un Mouse

Dins del mateix paquet de dades està inclòs el ***report ID***, ja que dins el mateix dispositiu es pot donar la necessitat de tenir més d'un ***report ID***.
A baix nivell el controlador ofereix tota una sèrie de funcions que permeten l'accés als reports, aquestes funcions són:

Get_Descriptor Request
Set_Descriptor Request
Get_Report Request
Set_Report Request
Get_Idle Request
Set_Idle Request
Get_Protocol Request
Set_Protocol Request

Desxifrant les dades del Wiimote

Amb les eines disponibles al Sistema Operatiu es possible establir comunicació amb dispositius Bluetooth, però abans de realitzar qualsevol intent de comunicació amb el Wiimote, es obligatori **ficar el Dispositiu en mode descobriment**, en el cas del Wiimote això s'aconsegueix prement a la vegada els botons 1 i 2 dels Wiimote.

Per realitzar una cerca de dispositius des de la consola s'utilitza l'executable **hcitool**:

Aquest ordre retorna tots els dispositius Bluetooth que tenim al voltant:

```
> hcitool scan
Scanning ...
        A8:F2:74:D6:77:82    Player One
        00:19:1D:AF:5A:C8    Nintendo RVL-CNT-01
```

En aquest cas es mostren 2 dispositius cadascun amb la seva adreça física i el seu nom associat. La ultima adreça 00:19:1D:AF:5A:C8, es tracta d'una direcció del dispositiu tipus que s'està buscant, com es pot veure hi ajuda molt el nom. Un cop ésté identificada l'adreça del dispositiu es podem enviar comandes al ***SDP***. Com per exemple obtenir tots el serveis que ofereix el dispositiu mitjançant ***browse***:

```
>sdptool browse 0:19:1D:AF:5A:C8
Browsing 00:19:1D:AF:5A:C8 ...
Service RecHandle: 0x0
Service Class ID List:
  "SDP Server" (0x1000)
Protocol Descriptor List:
  "L2CAP" (0x0100)
    PSM: 1
  "SDP" (0x0001)
Language Base Attr List:
  code_ISO639: 0x656e
  encoding:    0x6a
  base_offset: 0x100
Profile Descriptor List:
  "" (0x0100)
    Version: 0x0100

Service Name: Nintendo RVL-CNT-01
Service Description: Nintendo RVL-CNT-01
Service Provider: Nintendo
Service RecHandle: 0x10000

Service Class ID List:
  "Human Interface Device" (0x1124)
Protocol Descriptor List:
  "L2CAP" (0x0100)
    PSM: 17
  "HIDP" (0x0011)
Language Base Attr List:
  code_ISO639: 0x656e
  encoding:    0x6a
  base_offset: 0x100
Profile Descriptor List:
  "Human Interface Device" (0x1124)
    Version: 0x0100

Service RecHandle: 0x10001
Service Class ID List:
  "PnP Information" (0x1200)
Protocol Descriptor List:
  "L2CAP" (0x0100)
    PSM: 1
  "SDP" (0x0001)
Profile Descriptor List:
  "PnP Information" (0x1200)
    Version: 0x0100
```

El retorn de la comanda mostra que un Wiimote ofereix 3 serveis que son el ***SDP***, que correspon al ***Service Device Protocol***, el **HID** que correspon amb la ***Human InterfaceDevice*** i finalment un servei d'informació **PnP**, es a dir ***Plug and Play***. Lògicament el servei que més interessa és el HID que es el que porta les dades del

Wiimote.
Amb l'***sdptool*** s'ha vist força informació relacionada amb el Wiimote, però es pot veure més en profunditat quina és tota la que s'intercanvien els dos dispositius, això es pot realitzar fent peticions d'informació sobre el Wiimote i a la vegada utilitzar l'eina ***hcidump*** per veure el intercanvi d'informació entre els dos components, PC i Wiimote. Així en un terminal s'executa el ***dump*** del HCI:

```
> sudo hcidump -X -ihci1
HCI sniffer - Bluetooth packet analyzer ver 2.1
device: hci1 snap_len: 1028 filter: 0xffffffff
```

Aquí es quedarà bloquejat esperant comunicacions amb el dispositiu.
Ara en un altre terminal es realitza novament la navegació sobre el dispositiu per veure els serveis que ofereix, s'obté novament la mateixa informació que abans en aquest terminal, però al terminal on s'executa el ***hcidump*** apareix tota una sèrie d'informació, es pot veure a continuació un extracte del més interessant, concretament el que es refereix al servei del **HID**:

```
aid 0x0001 (SrvClassIDList)
< uuid-16 0x1124 (HID) >
                aid 0x0004 (ProtocolDescList)
<< uuid-16 0x0100 (L2CAP) uint 0x11 ><
                    uuid-16 0x0011 (HIDP) >>
                aid 0x0005 (BrwGrpList)
< uuid-16 0x1002 (PubBrwsGrp) >
                aid 0x0006 (LangBaseAttrIDList)
< uint 0x656e uint 0x6a uint 0x100 >
                aid 0x0009 (BTProfileDescList)
<< uuid-16 0x1124 (HID) uint 0x100 >>
                aid 0x000d (AdditionalProtocolDescLists)
<<< uuid-16 0x0100 (L2CAP) uint 0x13 >< uuid-16 0x0011(HIDP)>>>
                aid 0x0100 (SrvName)
                    str "Nintendo RVL-CNT-01"
                aid 0x0101 (SrvDesc)
                    str "Nintendo RVL-CNT-01"
                aid 0x0102 (ProviderName)
                    str "Nintendo"
                aid 0x0206 (unknown)
<< uint 0x22 str 05 01 09 05 a1 01 85 10 15 00 26 ff 00 75 08 95 01 06 00 ff 09 01 91
00 85 11 95 01 09 01 91 00 85 12 95 02 09 01 91 00 85 13 95 01 09 01 91 00 85 14 95
01 09 01 91 00 85 15 95 01 09 01 91 00 85 16 95 15 09 01 91 00 85 17 95 06 09 01 91
00 85 18 95 15 09 01 91 00 85 19 95 01 09 01 91 00 85 1a 95 01 09 01 91 00 85 20 95
06 09 01 81 00 85 21 95 15 09 01 81 00 85 22 95 04 09 01 81 00 85 30 95 02 09 01 81
00 85 31 95 05 09 01 81 00 85 32 95 0a 09 01 81 00 85 33 95 11 09 01 81 00 85 34 95
15 09 01 81 00 85 35 95 15 09 01 81 00 85 36 95 15 09 01 81 00 85 37 95 15 09 01 81
00 85 3d 95 15 09 01 81 00 85 3e 95 15 09 01 81 00 85 3f 95 15 09 01 81 00 c0 >>
```

Hi ha dos canals un el 0x11 i l'altre el 0x13, un es tracta del canal de control i l'altre el de informació, amb identificador 0x0206 s'obté tota una sèrie de bits, si consultem el perfil del HID veiem que el identificador 0x0206 es correspon amb la llista dels descriptors de l'HID. Si es formaten aquestes dades és pot veure que el Wiimote ofereix un total de 24

descriptors que van del 0x11 al 0x22 i del 0x30 al 0x3f. Comparant els valors amb els que es descriuen en el perfil de l'**HID** es pot veure el que una línia indica:

11	**95 01**	**09 01**	**91**
Report id	Grandària de les dates del report: 0x1	Ús	Direcció: Input

Taula 32. Reports HID Wiimote.

I fent aquest exercici per cadascun dels reports obtenim:

Tipus	**Report id**	**Grandària**
Output	0x11	1
Output	0x12	2
Output	0x13	1
Output	0x14	1
Output	0x15	1
Output	0x16	21
Output	0x17	6
Output	0x18	21
Output	0x19	1
Output	0x1a	1
Input	0x20	6
Input	0x21	21
Input	0x22	3
Input	0x30-0x3f	2-21

Taula 33. Reports HID Wiimote.

Amb els reports que ens ofereix el Wiimote s'intenta realitzar la connexió entre el Wiimote i el PC. Aquesta connexió es realitza mitjançant el dimoni ***hidd*** i a més controlant novament tota la comunicació mitjançant *hcidump*:

```
> sudo hidd --connect 00:19:1D:D6:A8:0D
```

(Nota: El Wiimote esta correctament connectat quan la fila de leds no deixi de fer pampallugues). En el terminal on s'ha llançat l'analitzador de paquets es pot veure com cada cop que es prem un botó apareix un report 0x30 0xHH 0xHH, on HH ens indica quin és el botó que s'està prement,

```
> ACL data: handle 11 flags 0x02 dlen 8
    L2CAP(d): cid 0x0041 len 4 [psm 19]
      HIDP: Data: Input report
      0000: 30 08 00                                    0..
> ACL data: handle 11 flags 0x02 dlen 8
    L2CAP(d): cid 0x0041 len 4 [psm 19]
      HIDP: Data: Input report
      0000: 30 00 00                                    0..
```

Així amb el report 0x30 s'informa d'un canvi d'estats en els botons. Per identificar els reports de sortida es pot utilitzar l'arxiu associat al dispositiu que està dins del directori

dev. Un cop connectats amb el dispositiu Wiimote llistem els dispositius que tenim a /**dev** i veiem que s'ha creat un fitxer nou amb el nom hidraw2, utilitzant la comanda **cat** amb el fitxer és pot veure que en prémer un botó del Wiimote s'obté una resposta que encara que illegible ens indica que efectivament el fitxer representa la comunicació amb el dispositiu. S'envia doncs un report amb id 0x11 i un altre byte:

```
> echo -en "\x11\xF1" > /dev/hidraw2
> echo -en "\x11\x01" > /dev/hidraw2
> echo -en "\x11\x00" > /dev/hidraw2
```

Comprovarem que el report 0x11 controla l'estat dels leds i el de la vibració del Wiimote. Amb tota aquesta informació i ficant fil a l'agulla es pot treure quina és la utilitat de cadascun dels reports:

Tipus	Report id	Grandària	Funció
Output	0x11	1	Control del Leds i vibració
Output	0x12	2	Canvi del tipus de report
Output	0x13	1	Habilitar càmera IR
Output	0x14	1	Habilitar altaveu
Output	0x15	1	Status Information Request
Output	0x16	21	Escriure a Memòria
Output	0x17	6	Llegir de Memòria
Output	0x18	21	Dades de l'altaveu
Output	0x19	1	Inhabilitar Altaveu
Output	0x1a	1	Habilitar càmera IR 2
Input	0x20	6	Informació de status
Input	0x21	21	Llegir de Memòria
Input	0x22	4	*ACK*
Input	0x30-0x3f	2-21	Data reports

Taula 34. Utilitat reports Wiimote.

Comunicació amb el Wiimote Programàticament

Treballant amb la consola ha esta possible comunicar-se e intercanviar dades amb el **Wiimote** i accedir i donar sentit als diferents reports que envia el **Wiimote**, però òbviament amb les utilitats que ens ofereix el Sistema Operatiu no es poden crear aplicacions, es necessita accedir a la API, aquesta ens la ofereix *l'**stack*** de Bluez.

1. Cerca de dispositius:

La cerca de dispositius utilitza la funcionalitat que ofereix la capa HCI de Bluetooth que només utilitza la capa física i la d'associació de dades[10].

Els adaptadors Bluetooth en un host son identificats mitjançant números enters començant per el 0, s'ha d'especificar per tant quin adaptador de Bluetooth es vol utilitzar, això es realitza mitjançant l'ordre ***hci_get_route*** que accepta com argument el numero enter que identifica el dispositiu. En cas que només es tingui un dispositiu en sistema o que ens

sigui indiferent quin és el que vol utilitzar, es possible passar NULL com a argument.
Aquesta funció ens torna un enter que representa el número de recurs del dispositiu, que podem passar com a argument de la funció *hci_open_device* per obrir el dispositiu.

```
dev_id = hci_get_route(NULL);
sock = hci_open_dev( dev_id );
if (dev_id < 0 || sock < 0)
    {
    perror("Error obrint el dispositiu");
    exit(1);
    }
```

Amb aquest codi ja es té disponible un ***socket*** obert per treballar sobre el dispositiu Bluetooth local i per tant poder-li enviar comandes, per exemple es pot obtenir la llista de dispositius que hi han al seu voltant mitjançant l'ordre **hci_inquiry**.

```
num_rsp = hci_inquiry(dev_id, len, max_rsp, NULL, &ii, flags);
```

Es tracta d'una de les poques ordres que no utilitza el ***socket*** obert, sinó que treballa encara amb l'identificador del dispositiu local, aquest ordre guarda i en el punter que se li ha passat com a argument guarda una llista de d'estructures que conté informació sobre cadascun dels dispositius que s'han trobat. L'estructura és la següent:

```
typedef struct {
    bdaddr_t     bdaddr;
    uint8_t      pscan_rep_mode;
    uint8_t      pscan_period_mode;
    uint8_t      pscan_mode;
    uint8_t      dev_class[3];
    uint16_t     clock_offset;
} __attribute__ ((packed)) inquiry_info;
```

Així es pot saber quina és la direcció de cadascun dels dispositius trobats, per tal d'obtenir la adreça en forma de cadena de caràcters s'utilitza la funció**bs2str** :

```
ba2str(&(ii+i)->bdaddr, (aux->address));
```

Finalment amb el *socket* al dispositiu local i l'adreça de cada dispositiu es pot obtenir quin és el nom de cadascun dels dispositius trobats mitjançant ***hci_read_remote_name.***

2. Comunicació amb el Wiimote:

Un cop trobats els dispositius que hi ha al voltant es poden discriminar quin d'aquests es correspon amb un Wiimote gracies al seu nom, i amb l'adreça iniciar una comunicació. Consultant el dispositiu Wiimote amb l'eina **sdptool** i analitzant les dades amb hcidump s'ha vist que els dos canals del Wiimote és corresponen amb el 0x11 per control i el 0x13 per interrupcions, així per tal d'iniciar la comunicació amb el Wiimote necessitarem crear

dos ***sockets*** un per cada canal, aquests dos ***sockets*** permeten realitzar les funcions bàsiques d'enviar i rebre dades del dispositiu. Amb aquest mètode s'està accedint a una comunicació de baix nivell des de la capa de L2CAP.

El següent codi correspon amb la creació del **socket** de control i la seva posterior connexió.
Creació del **socket**:

```
remote_addr.l2_family = AF_BLUETOOTH;
str2ba(result->address,&address);
remote_addr.l2_bdaddr = address;
remote_addr.l2_psm = htobs(0x11);
if ((ctl_socket = socket(AF_BLUETOOTH, SOCK_SEQPACKET, BTPROTO_L2CAP)) == -1)
    {
    printf("Socket creation error (control socket)");
}
if (connect(ctl_socket, (struct sockaddr *)&remote_addr, sizeof remote_addr))
     {
    printf("Socket connect error (control channel)");
}
```

Posteriorment es realitza la lectura del ***socket*** mitjançant:

```
ret = recv (int_socket, &data, sizeof(data),0);
```

Es tracta d'una funció bloquejant que esperarà a rebre dades pel canal per continuar amb l'execució.

I es pot realitzar una escriptura sobre el **socket** mitjançant la funció:

```
ret = write (int_socket, &data, sizeof(data));
```

Amb aquestes funcions es pot realitzar una comunicació complerta entre el Host i el Wiimote, en la carpeta d'exemples hi ha varies mostres de programes escrits en C per tal de veure com funciona la comunicació amb el Wiimote, com a representants de la lectura i l'escriptura de paquets hi ha els exemples **ComBasic** i **Turnlights**, els dos utilitzen una petita llibreria de comunicacions anomenada ***wiimotelib***.

a. Rebre paquets:

L'executable ComBasic[11] realitza un exemple bàsic que busca tots els dispositius disponibles, i en cas que siguin Wiimotes obre una comunicació amb ells i és queda a l'espera de rebre paquets. Com el report sol·licitat al iniciar una comunicació és el 0x30 enviarà un report per cada cop que hi hagi un canvi en l'estat d'un botó.

b. Enviar paquets:

L'executable TurnLights[12] realitza un exemple basic d'escriptura cap al Wiimote, concretament controlem l'encesa i l'apaga't dels leds del **Wiimote**

La Llibreria CWIID

S'ha decidit delegar l'accés als **Wiimotes** a una llibreria que ja ha estat testejada i depurada, es tracta de la llibreria ***CWiid***. ***CWiid***[13] és un projecte realitzat en C en codi lliure que utilitza *l'**stack*** Bluez.

L'obtenció de dades del **Wiimote** és pot realitzar bàsicament de dues maneres diferents activament o mitjançant *callback,* Aquests modes s'activen o desactiven mitjançant les funcions**Enable** i **Disable** que accepten els següents arguments:

- **FLAG_CONTINUOUS**: Habilita els informes continus del Wiimote
- **FLAG_MESG_IFC**: Habilita la interfície basada en missatges
- **FLAG_NONBLOCK:** Permet que get_mesg() no és bloquegi esperant un canvi en l'estat del Wiimote.

Es vol fer treballar els capturadors de dades en dos modes, el mode actiu i el passiu, per tant s'han d'habilitar correctament les opcions dels Wiimote. En el cas de voler que treballi en mode passiu es reben és contínuament quin és l'estat del Wiimote i els valors dels seus sensors s'activarà la funció **FLAG_CONTINUOUS**:

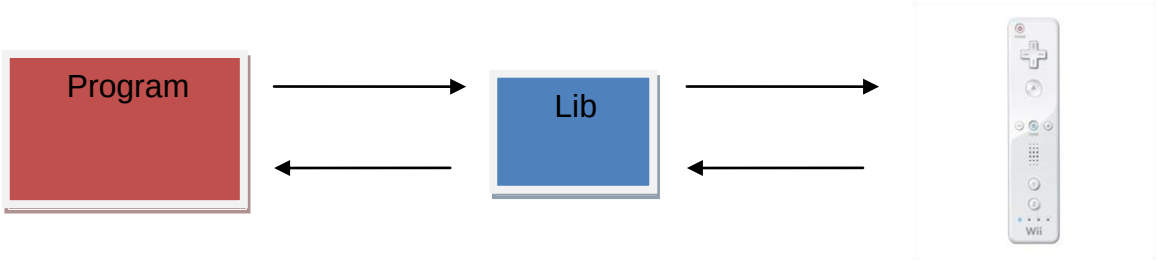


Figura 52. Mètode Passiu.

L'accés mitjançant **FLAG_MESG_IFC** ofereix un accés passiu, els canvis en l'estat del Wiimote ens provocaran la crida d'una funció de ***callback*** que permet programar quina serà la utilitat que li volem donar a les dades rebudes.

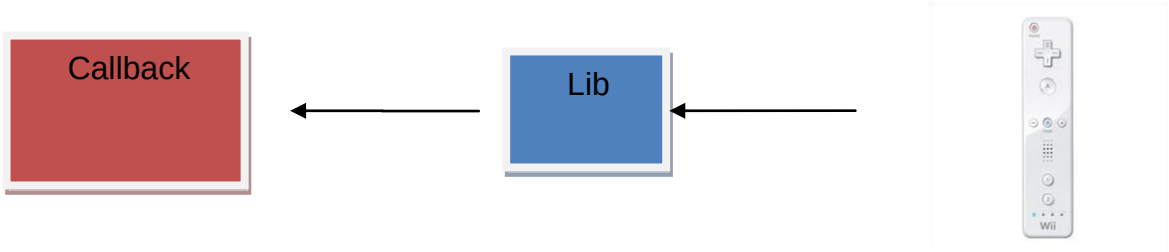


Figura 53. Mètode Actiu.

Condicionament de les dades:

Mitjançant la llibreria **cwiid** les dades que s'obtenen son en brut, es a dir s'haurà de fer un tractament a les dades per tal d'adaptar-les a les necessitats de les capes superiors.

Acceleròmetre:

Respecte a les dades corresponents a l'acceleròmetre les dades venen en una estructura

amb tres variables, cadascuna de les posicions de la llista correspon a un eix de coordenades.

Una mostra d'exemple dels valors retornats per l'acceleròmetre seria la següent:(130,133,160)

- 130 es correspon amb el valor de la coordenada X
- 133 es correspon amb el valor de la coordenada Y
- 160 es correspon amb el valor de la coordenada Z

Els valors de calibratge de l'acceleròmetre també son accessibles mitjançant la funció **get_acc_cal** si s'executa la funció per un Wiimote concret s'obté:

Dades de calibratge:
([131, 132, 133], [157, 159, 160])

Les dades de calibratge son dos valors, el valor quan no tenim cap força aplicada sobre el sensor i el valor quan tenim una força +g. Per cada cas:

X	en Repòs	+g
	131	157
Y	en Repòs	+g
	132	159
Z	en Repòs	+g
	133	160

Taula 35. Valors Acceleròmetre Wiimote.

Amb aquests valors és possible normalitzar els valors obtinguts:

$$ValorNormalitzat = \frac{ValorObtingut - Valorenrepòs}{Valorg - Valorenrepòs} \quad \text{Eq. 9}$$

S'obté 0 per quan sobre l'eix no actuï cap força i 1 quan la força aplicada sigui la de la gravetat 9,8m/s^2 amb el signe contrari de la força aplicada a l'eix.

Giroscopi:

Mantenint el giroscopi en repòs s'obtenen les següents dades:

Pitch:8322
Yaw: 7950
Roll: 8175

Aquests valors degut a les mancances del giroscopi son aproximats i a més varien al llarg del temps encara que mantinguem el Wiimote en repòs. Per tal de caracteritzat el giroscopi i saber com son les dades que es reben s'ha d'utilitzar una eina externa per calibrar-lo, en aquest cas s'utilitza un disc de soldador que permet controlar la velocitat de gir.

Figura 54. Caracterització del giroscopi.

En fer girar el Wiimote sobre cadascun dels eixos s'obté una lectura acurada gracies a la taula de gir i també una lectura del giroscopi mitjançant el Bluetooth, amb un gir de 60º/segon s'obtenen els següents valors:

Pitch: 9280 comptes

Roll: 9450 comptes

Yaw: 9460 comptes

En repòs s'obtenen en aquest moment els següents resultats:

Pitch: 8170 comptes

Roll: 8300 comptes

Yaw: 7940 comptes

Amb una simple regla de 3 es pot calcular que cada 20 comptes son 1º.

$$Pitch = \frac{(9300 - 1180)comptes}{60º} = 20 \cdot \frac{comptes}{º} \qquad \text{Eq. 10}$$

6.3. Connexió amb el telèfon mòbil

Per la connexió entre el PC i el telèfon mòbil no s'utilitzarà la interfície Bluetooth, sinó la interfície de xarxa. És tracta d'una connexió que disposa de les següents avantatges respecte al Bluetooth:

- més ample de banda
- cobreix major distància

(Cap d'aquestes avantatges es significativa, però també estem més familiaritzats amb

programar en un entorn de xarxa i això es definitiu a l'hora de decidir-nos.)
El mode de establir la connexió és equivalent a l'anterior, així el dispositiu també entrarà en mode descoberta, i això permetrà al servidor establir una connexió de control per enviar comandes de control al dispositiu i un altre connexió per rebre les dades.

Programació del client:

El client que vam plantejar havia de complir les següents condicions:

- Seguir l'esquema dels dispositius Bluetooth de mode cerca + connexió.
- Disposar d'una canal de control i un altre de dades
- Poder enviar les dades d'una manera regular controlable
- Poder disposar de diferents interfícies d'enviament de dades
- Poder enviar des de diferents dispositius

Amb aquestes especificacions, la màquina d'estats del client serà la següent:

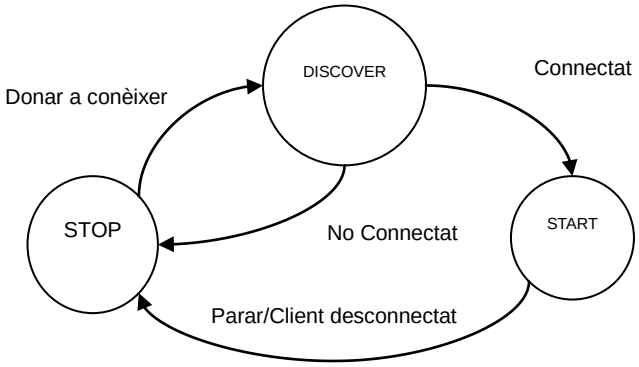


Figura 55. Estats Aplicació Mòbil.

I l'esquema del client és el següent:

Comunicacions:

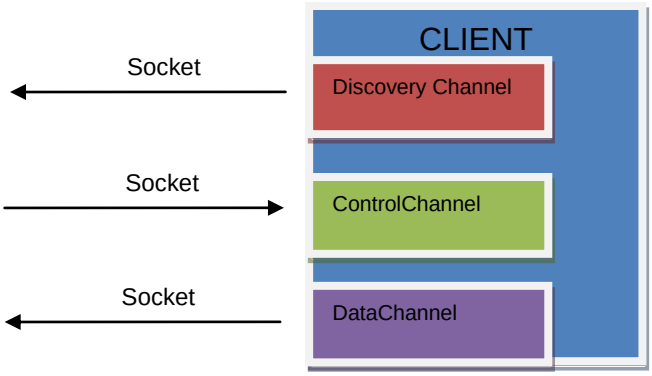


Figura 56. Sockets en Aplicació Mòbil.

Bàsicament un ***socket*** permet la comunicació entre dos programes utilitzant una interfície de xarxa gracies a 3 elements, una direcció IP, un número de port i un protocol de

transport. La direcció IP reconeix a l'ordinador, el número de port està assignat a la aplicació i finalment el protocol de transport indica quin tipus de comunicació estabiliran. Cadascun dels canals del client disposa de 3*sockets* cadascun amb una missió especifica:

- Discovery Channel Socket:

La finalitat d'aquest ***socket*** es que emeti sobre tota la xarxa local a la que està connectat un missatge de que està disponible un nou dispositiu capturador de dades mòbil, en aquest tipus de paquet anomenats paquet de ***broadcast***, no han d'estar enviats a cap adreça en concret però si a un port, el programa servidor serà l'encarregat de llegir sobre un port concret per saber si hi han nous capturadors de dades. Aquestes emissions lògicament no fa falta que estiguin orientades a connexió i per tant utilitzaran el protocol de transport UDP per l'enviament de les dades.

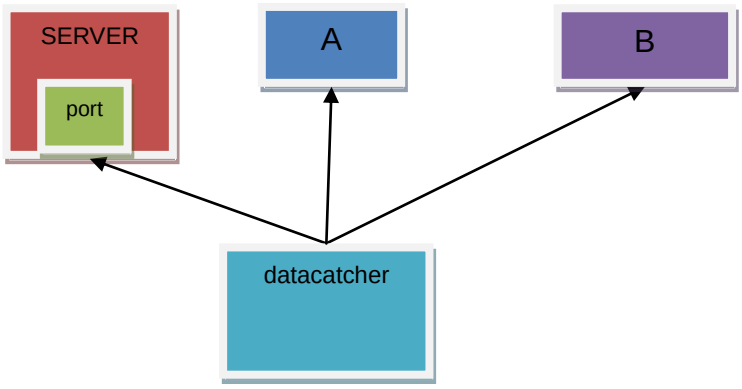


Figura 57. Descobriment de dispositius Mòbils.

- *Control Channel Socket:*

El canal de control s'ocupa de rebre ordres des de el servidor per saber quina acció es vol realitzar. Lògicament en aquest cas es necessita assegurar que l'ordre que s'envia és rebuda correctament per el capturador de dades, per tant es necessari un ***socket*** orientat a connexió. Les ordres que és poden rebre son: **INICIAR/FINALITZAR LA TRANSMISSIÓ de dades,** també sobre aquest canal es reben altres dades necessàries per establir la comunicació, per exemple el ***handshaking*** que permetrà establir quins seran els paràmetres de la comunicació i sobre quin port volem que s'iniciï l'enviament de dades, això es imprescindible donat que si es reben les dades de més d'un dispositiu ha de ser possible que cadascun d'ells enviï dades a ports diferents.

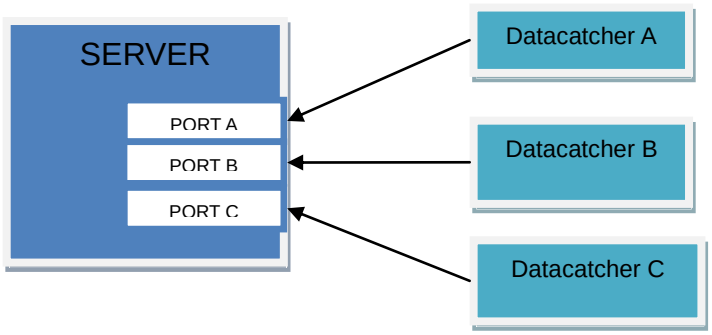


Figura 58. Distribució de ports.

Sobre aquest canal enviarem les dades dels sensors del mòbil. Per aquest canal realitzar l'enviament de dades amb un protocol de transport orientat a connexió no sembla mala idea, això assegura rebre les dades en ordre i sense perdre cap dels paquets enviats, però un temps de latència massa alt o un canal amb errors el protocol TCP provoca perdre el control del flux de les dades per el fet d'haver de validar cadascun dels paquets que s'envien o el reenviament de paquets que no es reben correctament, per tant per evitar aquests problemes és millor utilitzar un protocol no orientat a connexió tot i conèixer les mancances d'aquests tipus de protocols, com la pèrdua de paquets o rebre'ls de forma desordenada.

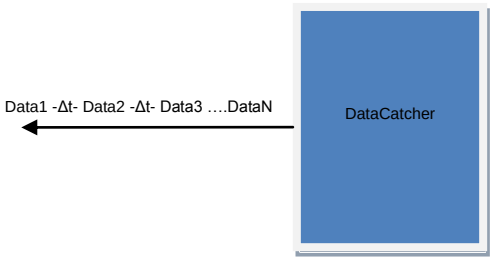


Figura 59. Enviament de Dades.

Programació del Client Android:

El client **Android** està realitzat amb temporitzadors per el control del flux de dades, cada x mil.lisegons un temporitzador s'aixeca i realitza la tasca d'enviar les dades per el canal corresponent. Cal tenir en compte que Android té varies restriccions en la seva programació, per exemple:

- No permet la lectura directa del valor dels sensors sinó que funciona per callback
- No permet l'execució de funcions bloquejants en el fil principal d'execució

Aquestes dos restriccions justifiquen algunes solucions adoptades, que encara que siguin una mica rebuscades, han de complir les exigències del sistema operatiu on s'executen.

esquemàticament està realitzat:

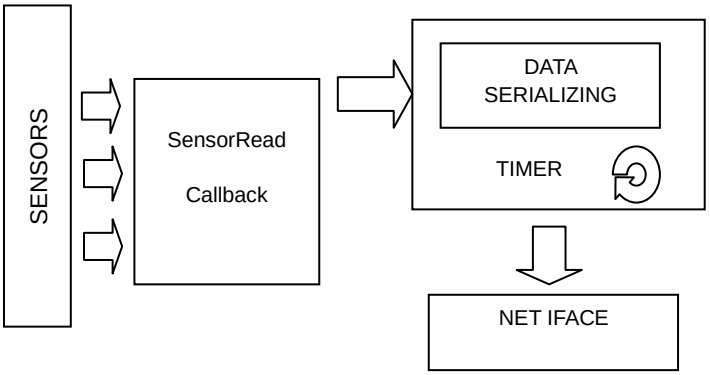


Figura 60. Diagrama de blocs de la Aplicació mòbil.

Es tracta d'un objecte que utilitza els objectes associats a sensors que ofereix **Android**, per

cada canvi en el valor d'un sensor executa un *callback* que serialitza les dades rebudes, paral·lelament tenim un temporitzador que s'aixeca cada x mil.lisegons recull les dades serialitzades i l'envia al servidor en forma de datagrama. D'aquesta manera podem programar la interfície de sortida per tal que el temporitzador s'executi cada certs mil.lisegons i poder controlar el flux d'enviament de paquets.

Tractament de les dades:

Per tal d'uniformitzar els capturadors de dades hem d'intentar que les dades que envien tots els capturadors sigui igual en unitats, s'ha vist abans que un cop tractades les dades amb el Wiimote es rebem amb les següents unitats:

Sensor	Unitats	Rang Vàlid
Acceleròmetre	No Units	[-1:1]
Giroscopi	rad/seg	--

Per la seva part, el mòbil envia la següent informació i en les següents unitats:

Sensor	Unitats	Rang Vàlid
Acceleròmetre	m/s^2	--
Giroscopi	rad/seg	--
Magnetòmetre	uT	--

El giroscopi, com es veu a les taules envia les dades en les mateixes unitats, per tant no s'ha de realitzar cap tipus de tractament sobre les dades, en canvi si que s'haurà de tractar les dades que obtenim de l'acceleròmetre, per aconseguir-ho simplement es tracta de normalitzar el vector, aquesta operació és realitza simplement dividint el valor llegit per el mòdul del vector:

$$\mathrm{Vaccn} = \frac{\mathrm{Vacc[n]}}{|\mathrm{Vacc}|} = \frac{\mathrm{Vacc[n]}}{\sqrt{V_{accx}^2 + V_{accy}^2 + V_{accz}^2}} \qquad \text{Eq. 11}$$

En el cas del magnetòmetre també interessa obtenir un rang com en el cas de l'acceleròmetre, per tant també normalitzarem el valor de la sortida:

$$\mathrm{Vmagn} = \frac{\mathrm{Vmag[n]}}{|\mathrm{Vmag}|} = \frac{\mathrm{Vmag[n]}}{\sqrt{V_{magx}^2 + V_{magy}^2 + V_{magz}^2}} \qquad \text{Eq. 12}$$

D'aquesta manera obtenim les sortides de l'acceleròmetre i el giroscopi amb valors:

Sensor	Unitats	Rang Vàlid
Acceleròmetre	No units	[-1:1]
Giroscopi	rad/seg	--
Magnetòmetre	No units	[-1:1]

Taula 36. Valors Sensors Mòbil.

Serialització de les dades:

Per un socket només es poden enviar dades sense format, això vol dir que l'únic que podem enviar son cadenes de bytes que és donen sentit als extrems de la comunicació, això no suposa cap problema quan es vols envia dades molt simples, per exemple enters o cadenes de caràcters, la cosa és complica quan el que es vol enviar son classes o dades estructurades i encara es complica més si es vol enviar classes entre diferents llenguatges de programació. Es coneix com a serialització de les dades el pas de dades estructurades o objectes a una cadena de caràcters de tal manera que pugui ser tractada en una corrent de dades, ja sigui un fitxer de dades o una connexió, aquesta serialització ha de permetre també la desserialització, es a dir que a partir de les dades sigui possible la reconstrucció de l'objecte. Existeixen mols serialitzadors de dades disponibles, alguns menys en cas que es vulgui que tinguin suport en varies plataformes a la vegada. Els tres més utilitzats i multi llenguatge son Json, XML i protobuf, les característiques bàsiques de cadascun son:

Json

- Editable i llegible
- Analitzable sense esquema
- Suport ampli dels navegadors XML

XML
- Editable i llegible
- Analitzable sense esquema
- Moltes eines de suport (xsd, xslt, sax, dom, etc)

Protobuf

- Densitat d'informació molt alta per quantitat de dades
- Molt difícil d'analitzar sense l'esquema
- Velocitat de processament molt alta
- No està creat per interactuar amb els usuaris

Per velocitat de transmissió es decideix l'ús de Protobuf, aquest es un mecanisme per la serialització de dades desenvolupat per Google, admet serialització i deserialització de dades entre tres plataformes Java, Python i C++, partint d'un fitxer de text anomenat fitxer proto, on es descriu la estructura de les dades, així compilant-lo en cadascuna de les plataformes requerides obtenim uns fitxer que permeten la serialització-deserialització de cadascun dels missatges.

En aquest cas el fitxer protobuf serà:

```
package projectpfc;
option java_package = "com.pfc.serializer";
option java_outer_classname = "RAWData";
message DataReaded
{
repeated SensorReads sensor = 1;
}
message SensorReads
  {
  enum Sensortype{
       GYROSCOPE = 0;
       ACCELEROMETER = 1;
       MAGNETOMETER = 2;
       LINEARACC = 3;
       GRAVITY = 4;
       ORIENTATION = 5;
  }
  required double x = 1;
  required double y = 2;
  required double z = 3;
  required Sensortype type = 4;
}
```

Aquí indica que s'enviarà un missatge ***datareaded*** que estarà format per un o varis missatges SensorReads que cadascun portarà la informació de la lectura del sensor que li correspon.

7. Posició Espaial dels Capturadors

En aquesta secció s'explicarà com es poden combinar les dades que s'extreuen dels capturadors de dades mitjançant els sensors per tal de conèixer quina és la posició espaial en que està situat el capturador de dades.

En general qualsevol objecte en un espai 3D disposa de 6 graus de llibertat:

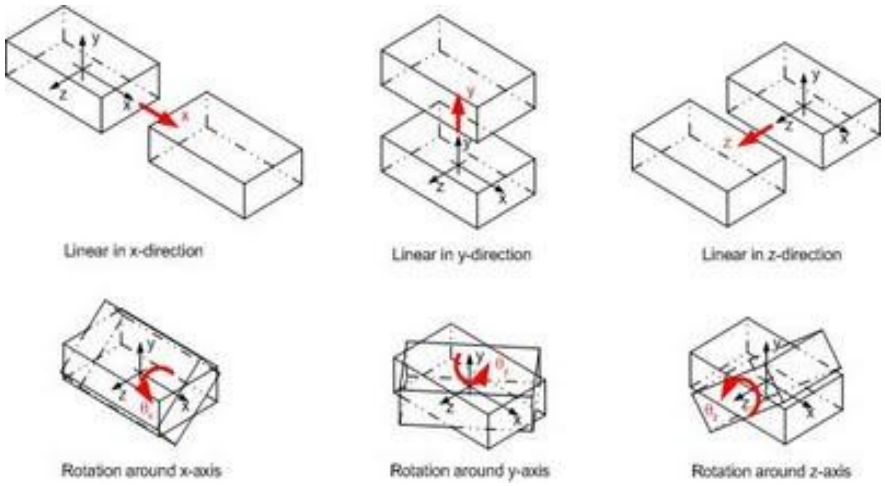


Figura 61. Moviments d'un objecte en 3D.

En el cas dels capturadors de dades cadascuna de les tres rotacions que podem realitzar estan definides amb un nom en funció de quin és l'eix de gir, així un gir sobre l'eix X, l'anomenarem ***Roll***, una rotació sobre l'eix Y l'anomenarem ***Pitch*** i finalment una rotació sobre l'eix z ***Yaw***, per exemple en el cas del **Wiimote**.

Cada capturador de dades té definit el seu propi eix de referència:

Figura 62. Eixos de Coordenades.

Tal com s'ha vist anteriorment és disposa de dos tipus de capturadors de dades el **Wiimote** i un telèfon mòbil **Nexus 4**, el Wiimote presenta la limitació de no disposar d'un magnetòmetre, ja es veurà com afecta als càlculs.

7.1. Conèixer la posició d'un únic angle

S'ha de calcular quina es la rotació que ha realitzat el capturador de dades sobre un únic eix ja sigui ***pitch, yaw*** o ***roll***. El càlcul de l'angle de rotació sobre l'eix Z, amb un moviment en vertical, es veu gràficament en la següent imatge.

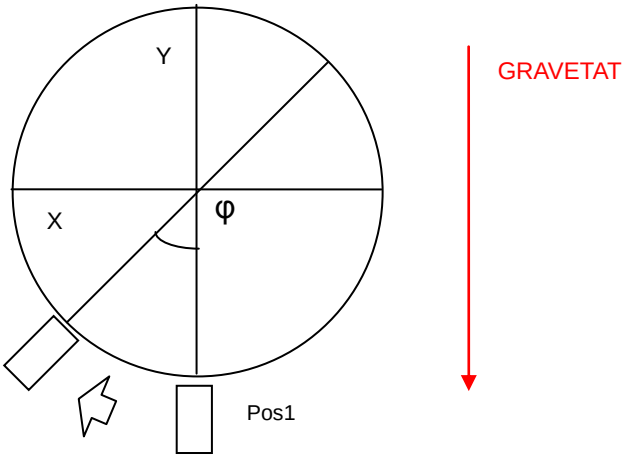


Figura 63. Moviment d'un Objecte en un únic Eix.

En moure el Wiimote des de la posició 1 a la posició 2 hi haurà un canvi en l'angle φ, que és correspondrà en aquest cas concret amb una rotació sobre l'eix Z, si es mesura quin es l'angle es pot saber quin és la posició del capturador de dades en la circumferència.

Calcular l'angle amb l'acceleròmetre:

La força de la gravetat aplicada sobre cada eix de l'acceleròmetre en repòs variarà únicament en funció de la posició espaial del sensor, així si les llegeix directament l'acceleròmetre en repòs, podrem obtenir la posició actual.
Tenint com a referència l'eix sobre el que actuaran les forces:

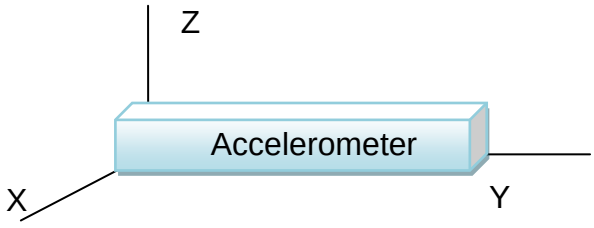


Figura 64. Eixos de l'acceleròmetre.

Degut a la força de la gravetat aplicada sobre l'acceleròmetre s'obtindrà un vector diferent en funció de la seva orientació espaial, si es manté el Wiimote quiet sobre una superfície plana, per exemple, s'obté un vector (0,0,1) es a dir l'únic eix que es veurà afectat per la gravetat serà l'eix Z del Wiimote que mesurarà un total de +g. Un moviment en vertical o en lateral o de rotació del Wiimote provocarà un canvi en aquest vector, així en el cas d'una

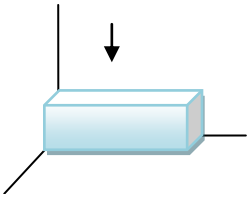


Figura 65. Força Gravetat Aplicada.

rotació de 90º provocarà un canvi en el vector a (1,0,0) i mentre que la rotació el vertical una de (0,1,0). Amb aquests canvis del vector que es donen podem efectivament calcular quina és la orientació que té el Wiimote.

Per tal de calcular els angles s'haurà d'aplicar trigonometria bàsica, en primer lloc es calcula l'angle Φ,

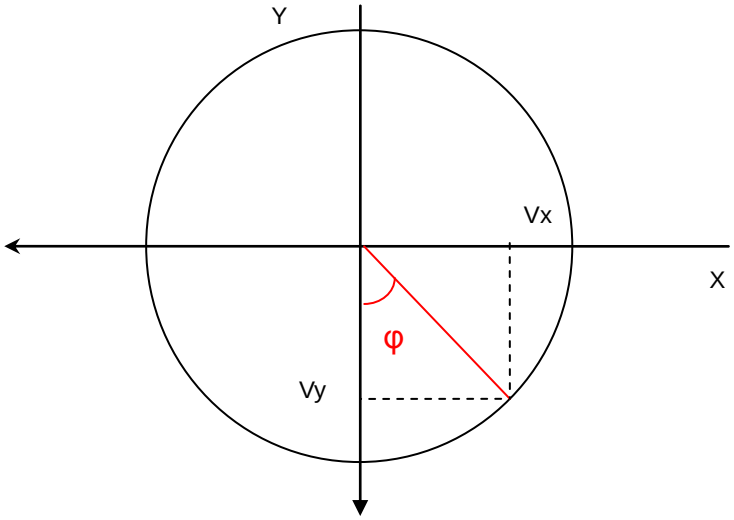


Figura 66. Angle Calculat Moviment d'un Eix.

Tal com veiem en la figura 67, l'angle φ es pot calcular gracies a Vx i de Vy mitjançant les relacions trigonomètriques del triangle rectangle, se sap que:

$$\sin\varphi = \frac{Vy}{\|V\|} \qquad \text{Eq. 13}$$

$$\cos\varphi = \frac{Vx}{\|V\|} \qquad \text{Eq. 14}$$

Mentre el capturador de dades es mantingui en estat estacionari el mòdul del vector que es llegeix de l'acceleròmetre serà sempre 1.

$$|\vec{v}| = \sqrt{x^2 + y^2 + z^2} = 1 \qquad \text{Eq. 15}$$

Així aïllant el valor de φ d'aquestes dues relacions (eq. 12) i (eq. 13) es suficient per conèixer quin es l'angle que forma el capturador amb el terra i per tant quina es la posició espaial del capturador de dades. El problema d'aquestes equacions es que només funcionen per 2 quadrants, si es vol augmentar el número de quadrant es pot disposar de la funció atan2, que té en compte el signe per saber a quin quadrant.
Així dividint les dues equacions:

$$\frac{\sin\varphi}{\cos\varphi} = \frac{\frac{Vy}{\|V\|}}{\frac{Vx}{\|V\|}} \qquad \text{Eq. 16}$$

I per tant:

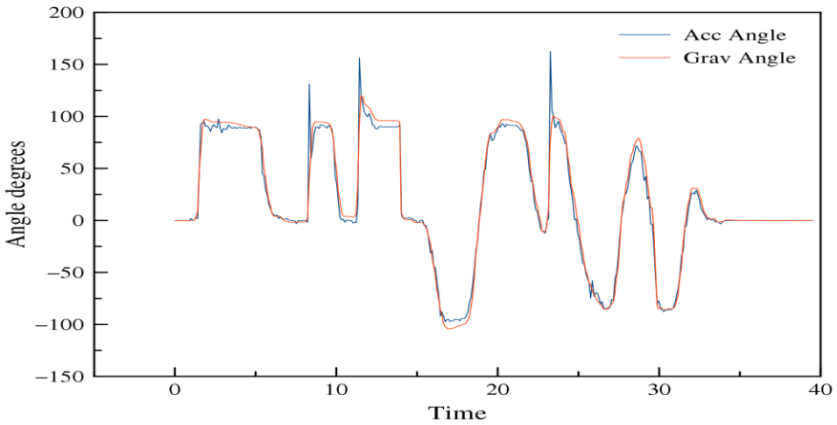
$$\tan\varphi = \frac{Vy}{Vx} \qquad \text{Eq. 17}$$

Es necessari modificar la funció **atan** per obtenir l'angle per atan2, aquesta és una funció amb dos arguments que permet obtenir el quadrant sobre el que treballen gracies als valors i els signes dels dos arguments, de la manera que ens indica la figura 68:

$$\operatorname{atan2}(y,x) = \begin{cases} \arctan\left(\frac{y}{x}\right) & x > 0 \\ \arctan\left(\frac{y}{x}\right) + \pi & y \geq 0, x < 0 \\ \arctan\left(\frac{y}{x}\right) - \pi & y < 0, x < 0 \\ +\frac{\pi}{2} & y > 0, x = 0 \\ -\frac{\pi}{2} & y < 0, x = 0 \\ \text{undefined} & y = 0, x = 0 \end{cases}$$

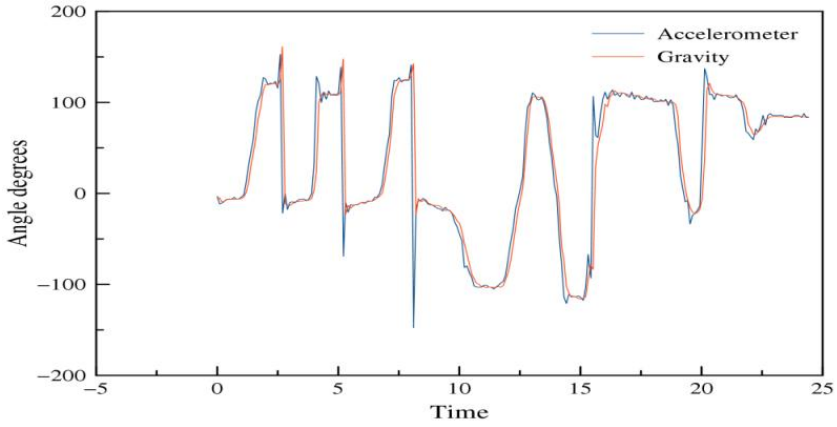
Figura 67. Formula de atan2.

Les lectures associades a l'acceleròmetre tenen molt soroll per tant seria molt millor utilitzar les dades del sensor de gravetat, veiem les dues lectures dels dos sensors contraposades en el cas del mòbil en primer lloc:



Gràfica 11. Càlcul de l'angle amb Acceleròmetre i Gravetat (Mòbil).

Es pot observar perfectament a partir de la gràfica que els càlculs de l'angle realitzats amb les dades de gravetat eliminen tot el soroll de l'acceleròmetre i permeten realitzar un càlcul correcte de l'angle associat a la rotació de l'eix. A continuació és mostren les mateixes gràfiques utilitzant les dades del Wiimote filtrades per aconseguir la gravetat:



Gràfica 12. Càlcul de l'angle amb Acceleròmetre i Gravetat (Wiimote).

Es pot observar a la gràfica com el filtre passabaixes actua correctament i filtre el soroll de freqüències més altes de totes maneres alguns pics provocats per moviments bruscos no els acaba de filtrar correctament i per tant no es gaire satisfactòria la solució del Wiimote.

Calcular l'angle amb el giroscopi:
La posició actual del capturador a partir de la lectura del giroscopi es pot realitzar a partir de lectures que del giroscopi i la seva integració donat que:

$$pos = \int vel \cdot dt \qquad \text{Eq. 18}$$

Amb varies rotacions entre -90º a 90º sobre l'eix X amb el Wiimote obtenim:

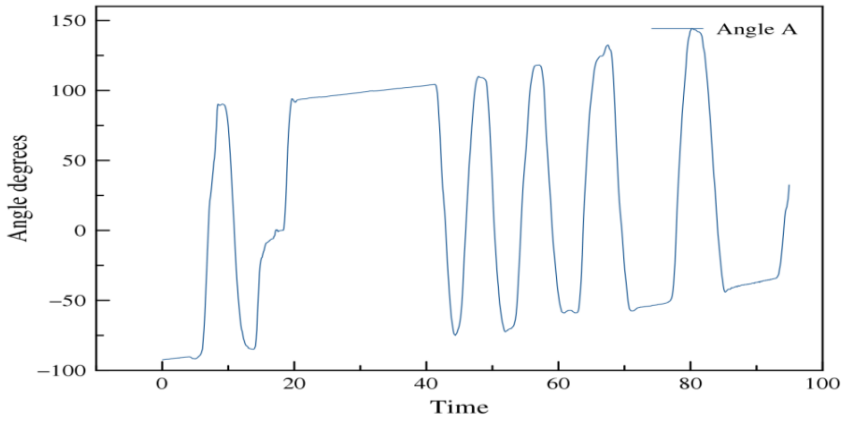


Figura 68. Càlcul de l'angle amb el giroscopi.

Com s'ha vist anteriorment els giroscopis tenen un offset que canvia amb la temperatura això fa que no pugui ser compensable, a més la integració d'aquest offset provoca aquesta deriva en la lectura de la posició amb el giroscopi, tal com veiem a la gràfica amb aquest resultat ens resulta impossible fer un seguiment de la posició del motor només amb el giroscopi.

En el cas des capturadors de dades seleccionats els dos ofereixen una solució força acceptable utilitzant únicament l'acceleròmetre, això es degut a que els dos realitzen

tractament de dades per eliminar el soroll, normalment però llegint la sortida d'un acceleròmetre la lectura és molt sorollosa tot i que a llarg termini en mode estacionari ens dona una lectura precisa de la posició del dispositiu. El giroscopi per la seva part té una lectura bona a curt termini però falla per culpa de la deriva, es a dir a llarg termini, per sort veiem que els errors dels dos sensors son complementaris i per tant el que s'acostuma a fer en aquests casos és treure avantatge de les bones característiques de cadascun d'ells per tal de fer un seguiment molt millor de la posició del dispositiu, això és el que es coneix com a **fusió de sensors**.

Per tal de fusionar les dades del giroscopi i l'acceleròmetre es disposen de varis mètodes, la forma més eficient és el càlcul e implementació d'un filtre **Kalman**, per desgràcia les matemàtiques associades al càlcul d'un filtre de **Kalman** son força complexes, però per sort existeix una alternativa, un filtre complementari que ofereix uns resultats no gaire pitjors que el filtre Kalman i a més el càlcul de paràmetres per la seva implementació és molt més senzilla.

La idea del filtre complementari[14] es utilitzar un doble filtre sobre les posicions que és calculen amb les dades del giroscopi i amb les dades de l'acceleròmetre, el primer filtre és de tipus passabaixes aplicat sobre les dades de l'acceleròmetre, el segon es tracta d'un filtre passaaltes aplicat sobre les dades de la posició que obtenim del giroscopi:

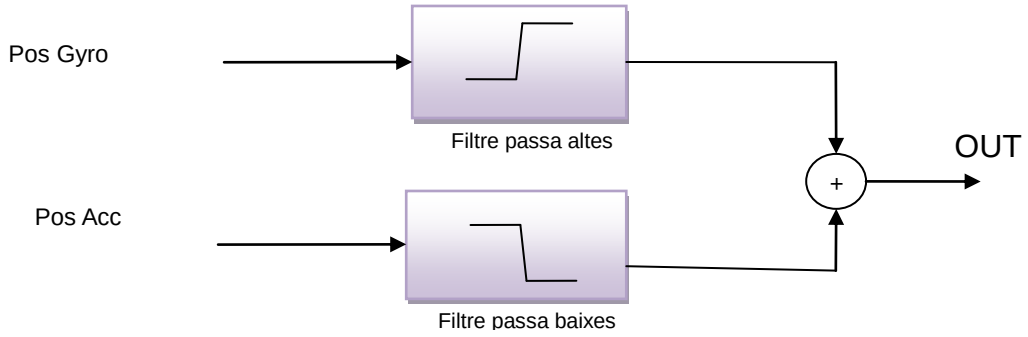


Figura 69. Blocs Filtre Complementari.

La condició per tal d'aplicar correctament el filtre complementari és que en qualsevol instant de temps només un filtre dels dos s'estigui aplicant, la idea del filtre és aprofitar les lectures ràpides del giroscopi per fer estimacions a curt termini, a llarg termini però s'utilitzen les lectures de l'acceleròmetre que son més fiables. Les funcions de transferència de cadascun dels filtres:

Filtre passa baixes:

$$y[i] := \alpha * x[i] + (1-\alpha) * y[i-1] \qquad \text{Eq. 19}$$

Filtre Passa Altes:

$$y[i] := \alpha * y[i-1] + \alpha * (x[i] - x[i-1]) \qquad \text{Eq. 20}$$

Si es combinen els dos filtres en una sola sortida s'obté la funció de transferència del filtre complementari, sabent que el filtre passabaixes s'aplica sobre les entrades de la posició del giroscopi i les del filtre passabaixes sobre les dades de l'acceleròmetre s'obté:

$$y[i] := \alpha * \left(y[i-1] + \left(x_{gyro}[i] - x_{gyro}[i-1]\right)\right) + (1-\alpha) * x_{acc} \qquad \text{Eq. 21}$$

La funció de transferència del filtre en diagrama de blocs:

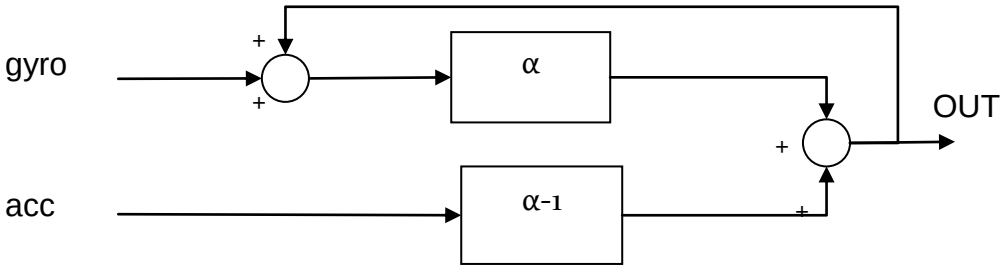


Figura 70. Diagrama de Blocs del Filtre Complementari.

En aquesta equació es veu el terme **α**, aquest terme ens indica quina és la relació de la sortida entre el giroscopi i la sortida de l'acceleròmetre, un **α** major indica un major pes del giroscopi a la sortida així com un **α** menor indica més pes de l'acceleròmetre, lògicament aquest paràmetre és una de les variables més importants a calcular.

El valor d'**α** es pot calcular a partir de les equacions anteriors:

$$\tau = \frac{\alpha \cdot dt}{1-\alpha} \rightarrow \alpha = \frac{\tau}{dt - \tau} \qquad \text{Eq. 22}$$

El valor de τ ens indica quin és en temps el que triga el filtre en donar validesa a les dades de l'acceleròmetre, per exemple es seleccionen un valor de τ de 150 mil.lisegons, això son moviments força curts en temps, un cop fixat aquest valor es poden calcular quins son els valors de **α** a partir de d'un determinar temps de mostreig, així utilitzant un temps de mostreig de 5ms obtenim un α de 0.965 i amb aquests valors per el Wiimote obtenim la següent gràfica:

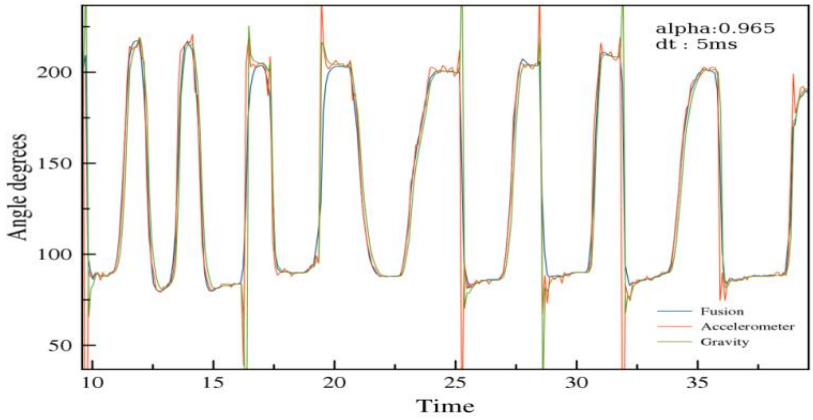


Figura 71. Càlcul angle filtre complementari VS Gravetat VS Acceleròmetre.

El moviments emprats per la realització de la gràfica anterior son aproximadament del temps requerit i es pot veure que el seguiment del filtre de fusió segueix correctament el moviment del capturador i a més elimina tot el soroll de l'acceleròmetre i corregeix la deriva del giroscopi. Si es preveu realitzar moviments més suaus es possible relaxar el valor de τ i això comportarà relaxar el temps de mostreig.

De la mateixa manera que hem calculat la rotació sobre l'eix Z també ens és possible calcular un altre angle, sempre que la rotació provoqui un canvi en la mesura de l'acceleròmetre.

7.2. Conèixer la posició de més d'un angle

Per descriure la posició d'un objecte i quina ha estat la rotació a la que ha estat sotmès en cadascun del angles disponibles en un espai tridimensional, **pitch, yaw i roll** és disposa d'un eina matemàtica que son les matrius de rotació.

Una matriu de rotació en un espai tridimensional està composada per 3 vectors unitaris, ortogonals e independents entre si, que indiquen quina és la relació que hi ha entre la rotació de l'objecte i un altre eix de coordenades, així es possible portar punts d'un eix de coordenades a un altre i conèixer les noves posicions dels punts en funció de l'eix que es vol utilitzar.

La rotació és el canvi d'orientació d'un objecte de forma que un punt o una línia és manté fixa, aquesta línia es coneguda com l'eix de rotació. Fem una aproximació a les matrius de rotació realitzant en primer lloc els càlculs de rotacions en un espai de 2 dimensions, veient de quina manera una matriu descriu l'angle de rotació realitzat i com a partir d'un punt inicial es pot calcular la nova posició del punt aplicant sobre ell la matriu de rotació.

Es defineix una matriu de rotació de l'angle θ en un pla com:

$$M = \begin{matrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{matrix} \qquad \text{Eq. 23}$$

On es defineix θ com la rotació aplicada el punt, tal com es veu a la figura 74:

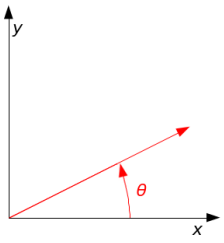


Figura 72. Rotació d'un punt en 2D.

Es pot comprovar fàcilment que si tenim un punt P = (x,y) en el que realitzem una rotació de θ graus la seva nova situació serà:

$$P' = M * P \qquad \text{Eq. 24}$$

$$x' = x \cdot \cos\theta - y \cdot \sin\theta \qquad \text{Eq. 25}$$

$$y' = x \cdot \cos\theta + y \cdot \sin\theta \quad \text{Eq. 26}$$

En el cas d'una matriu de rotació en 2D només es disposa d'un únic eix de rotació possible, en canvi per un espai tridimensional es disposa d'un total de 3 matrius de rotació una per cada eix, això afegeix força dificultat als càlculs:

$$Rx(\theta) = \begin{matrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{matrix} \quad \text{Eq. 27}$$

$$Ry(\theta) = \begin{matrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{matrix} \quad \text{Eq. 28}$$

$$Rz(\theta) = \begin{matrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{matrix} \quad \text{Eq. 29}$$

Cadascuna de les rotacions que s'apliquen sobre un eix tenen associat un angle, els tres angles son els anomenats angles d'**Euler**, com s'ha especificat abans la rotació sobre l'eix X coneguda o també coneguda com a **pitch**, la rotació sobre l'eix Yo **roll** i finalment d'inclinació sobre l'eix Z o **yaw**.

La rotació total aplicada en una composició ZYX sobre un objecte es pot obtenir multiplicant cadascuna de les rotacions sobre cadascun dels eixos, així la rotació resultant sobre un objecte serà la multiplicació de les tres matrius resultants:

$$R_{TOTAL} = R_X \cdot R_Y \cdot R_Z \quad \text{Eq. 30}$$

On l'ordre de les multiplicacions ens indica l'ordre d'aplicació de les rotacions. Aquesta rotació total es el que coneixem com **Matriu de Rotació en 3D.**

I així com abans el punt després de la rotació serà:

$$P' = M * PR_{TOTAL} = R_X \cdot R_Y \cdot R_Z \quad \text{Eq. 31}$$

Lògicament a l'hora de calcular la rotació total s'ha de tenir en compte que la multiplicació de rotacions no compleix la propietat commutativa, per exemple en una rotació sobre l'eix X e Y:

$$R_X \cdot R_Y \neq R_Y \cdot R_X \quad \text{Eq. 32}$$

Això es molt lògic, donat que l'ordre de les rotacions afecta en la posició final de l'objecte, per exemple no és el mateix rotar sobre l'eix X 90º i després 90º sobre l'eix Y que fer-ho a l'inrevés.

Així coneixent quina és la matriu de rotació es pot conèixer de quina manera ha rotat un objecte com s'ha vist fins ara. La pregunta que sorgeix a continuació és: **de quina manera es podria obtenir una matriu de rotació a partir dels sensors de que disposa un capturador de dades?**, Cada capturador de dades disposa dels següents sensors:

Capturador de Dades:	Sensors Disponibles:
Wiimote	Acceleròmetre, Giroscopi
Telèfon Mòbil	Acceleròmetre, Giroscopi, Magnetòmetre

Taula 37. Sensors disponibles per capturador de dades.

Utilitzant únicament el giroscopi és possible obtenir quina és la matriu de rotació a la que ha estat sotmès el capturador de dades. Un giroscopi retorna la velocitat de rotació sobre un eix, en el cas dels capturadors de dades son giroscopis de 3 eixos i per tant es possible obtenir-la rotació sobre cadascun dels aplicant les següents equacions:

$$Angle\_Roll = Lectura\_GiroscopiX * dt \quad \text{Eq. 33}$$

$$Angle\_Pitch = Lectura\_GiroscopiY * dt \quad \text{Eq. 34}$$

$$Angle\_Yaw = Lectura\_GiroscopiZ * dt \quad \text{Eq. 35}$$

A partir d'aquí substituint l'angle **Pitch**, l'angle **Roll** i l'angle **Yaw** a cadascun de les matrius de rotació corresponents i multiplicant les tres matrius s'obté la rotació total durant ***dt***, aquesta rotació es va acumulant per obtenir la matriu de rotació total.

$$Rt = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi & \cos\varphi \end{pmatrix} * \begin{pmatrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{pmatrix} * \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \text{Eq. 36}$$

Amb la matriu de rotació Rt es pot conèixer la rotació dels capturadors de dades, per desgracia el càlcul d'aquesta matriu hereta els problemes de les mesures dels giroscopis, el principal la deriva que provocarà falsejar la lectura dels angles i per tant que hi hagi una modificació de la rotació encara que és mantingui sense moviment el capturador. S'intentarà doncs realitzar el càlcul de la matriu de rotació a partir de dels altres sensors disponibles.

Es fàcilment imaginable que únicament amb un acceleròmetre no es pot obtenir una matriu de rotació donat que no es possible mesurar forces que actuïn perpendiculars a la força de gravetat, això vol dir que en cap cas serà possible capturar amb l'acceleròmetre certes rotacions:

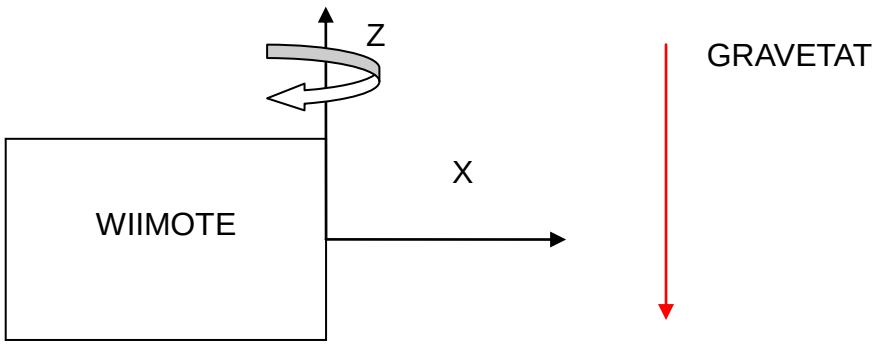


Figura 73. Força perpendicular a la gravetat.

Tal com és veu en la figura anterior un moviment de rotació sobre l'eix Z és provocat per una força perpendicular a la gravetat, aquesta força tot i resultar en una rotació no provoca cap canvi en la sortida de l'acceleròmetre, per tal de captar aquesta rotació es necessària l'ús d'un sensor que canviï en funció d'una força aplicada perpendicularment a la força de gravetat, aquest sensor per sort no és altre que el magnetòmetre.

La matriu de rotació del capturador de dades son 3 vectors unitaris, ortogonals e independents entre ells que indiquen quina és la rotació a la que ha estat sotmesa el propi capturador de dades, per tant son aquests 3 vectors els que haurem d'aconseguir per obtenir la matriu.

En primer lloc observem el sensor acceleròmetre, la sortida d'un acceleròmetre de tres eixos és un vector que apunta cap al centre de la terra en relació a la posició que ocupa el capturador de dades en l'espai es representa en aquesta senzilla transició:

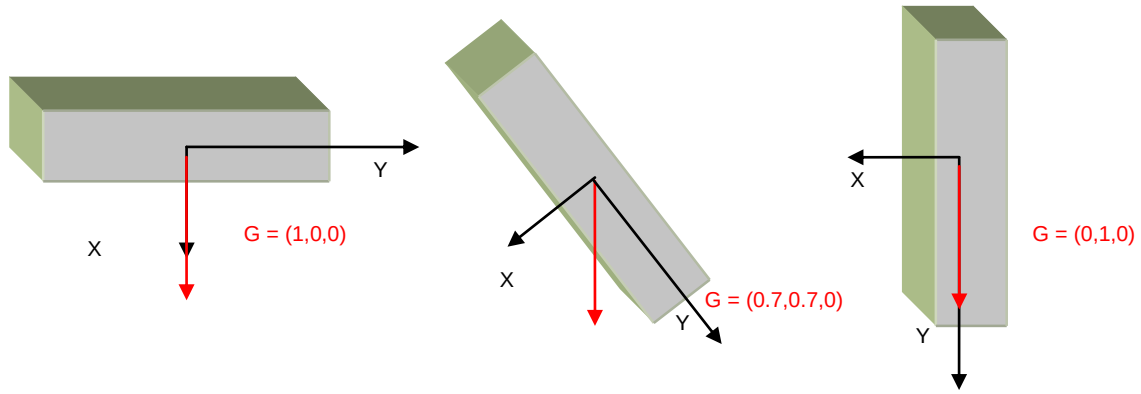


Figura 74. Vector de Gravetat.

Amb la informació que s'extrau del sensor acceleròmetre ja està disponible el primer vector del que serà la matriu de rotació.

El segon vector es pot extraure de la informació que obtenim del magnetòmetre, de la mateixa manera que l'acceleròmetre ens dona un vector que apunta cap al centre de la terra, la sortida del magnetòmetre es pot interpretar com un vector que ens apunta cap al nord magnètic, amb la informació d'aquest vector es possible calcular un vector perpendicular al vector de l'acceleròmetre i a la vegada perpendicular al Nord magnètic, això es realitza amb producte creuat dels dos vector:

$$E = uG \; X \; uB \qquad \text{Eq. 37}$$

Així s'obté un nou vector perpendicular als dos, aquest serà el segon vector.

Finalment es necessita un últim vector que sigui perpendicular als dos anteriors, per tant només es tracta de realitzar un nou producte creuat, aquest cop entre el vector de l'acceleròmetre i el nou vector calculat anteriorment:

$$N = uE \; X \; uG = (uG \; X \; uB) \; X \; uG \qquad \text{Eq. 38}$$

Ja tenim disponibles els eixos del nostre sistema de coordenades: G, E i N. Amb aquests

tres vectors podem crear-nos una matriu de rotació:

$$M = \begin{matrix} Nx & Ny & Nz \\ Gx & Gy & Gz \\ Ex & Ey & Ez \end{matrix} \qquad \text{Eq. 39}$$

Aquesta Matriu permetrà traslladar punts i vectors referencials al sistema de referència del capturador de dades a punts fixes referencials al punt de referència de l'usuari. Per sort també es pot realitzar el càlcul a la inversa, es a dir portar un punt o vector referència en el sistema de coordenades de l'usuari al sistema de referència del capturador, això és possible utilitzant la matriu inversa de la matriu calculada anteriorment.

Com ha succeït anteriorment però aquesta matriu de rotació continua heretant els problemes de les lectures dels sensors amb la que es construeix i en aquest cas no és més que el soroll de les lectures.

Novament com en el cas del càlcul d'un únic un angle tenim dos lectures, una bona a curt termini però dolenta a llarg termini com és el cas de la lectura amb el giroscopi i un altre bona a llarg termini (o en repòs) però molt sorollosa en moviment, per tant s'ha de tornar a realitzar la fusió dels dos sensors amb un filtre complementari, tal com s'indicava en el càlcul d'un únic angle el filtre complementari és:

$$y[i] := \alpha * \Big(y[i-1] + \big(x_{gyro}[i] - x_{gyro}[i-1]\big)\Big) + (1-\alpha) * x_{acc} \qquad \text{Eq. 40}$$

En el cas concret d'utilitzar matrius de rotació:

$$RMatrix[i] := \alpha * (RGyro^{*} RMatrix[i-1]) + (1-\alpha) * Racc - mag \qquad \text{Eq. 41}$$

Matemàticament les dues equacions son quasi idèntiques, les diferencies que es poden observar son els canvis de sumar els angles per multiplicacions de matrius, això es necessari perquè per fer acumular rotacions i veure la rotació final aplicada l'operació és la multiplicació en comptes de la suma.

En cas de no disposar d'un magnetòmetre com el cas del Wiimote que és pot realitzar, l'únic que ens queda és utilitzar l'acceleròmetre de manera creativa. Sabent que l'acceleròmetre de 3 eixos té un vector de 3 components, mentre l'acceleròmetre es mantingui en estat estacionari el mòdul del vector que llegim serà sempre 1.

$$|\vec{v}| = \sqrt{x^2 + y^2 + z^2} = 1$$

Per tant una possible aproximació per solucionar el problema seria utilitzar aquesta propietat per saber quan un capturador de dades està en moviment i començar a realitzar la integració de les dades del giroscopi en aquell moment. El diagrama de flux de l'algoritme en aquest cas seria:

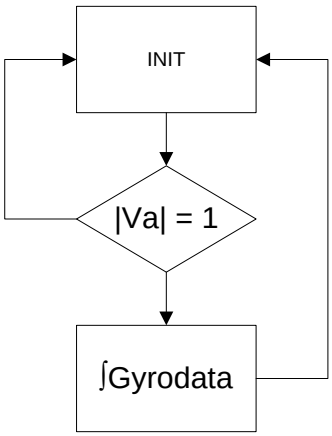


Figura 75 Diagrama de flux

8. Aplicacions

Amb les dos llibreries descrites anteriorment, la del control del robot i la del control dels capturadors de dades podem començar a treballar en diferents aplicacions que uneixin els dos mòduls, en aquest apartat presentarem totes les que s'han desenvolupat. Moltes d'aquestes aplicacions utilitzen eines que poden ser compartides entre elles i per futures aplicacions per tant s'han organitzat moltes d'aquestes eines en forma de llibreria estàtica.

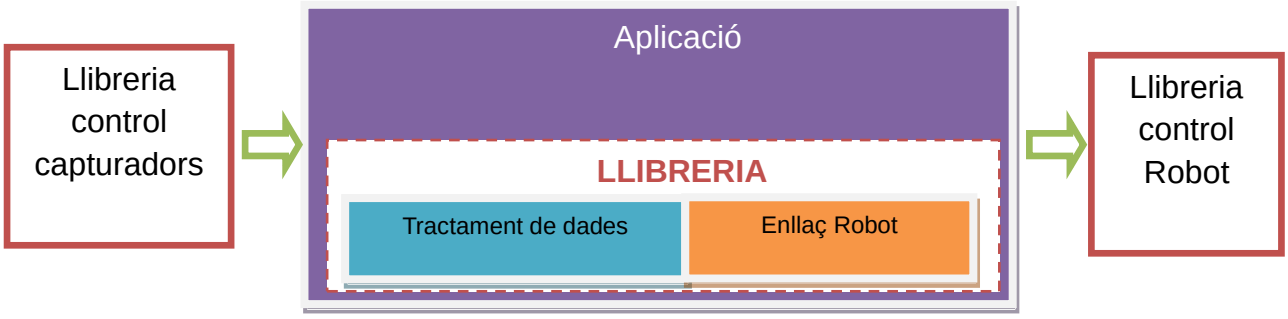


Figura 76. Diagrama blocs Aplicació.

8.1. La llibreria d'eines

La llibreria esta plantejada i construïda de forma modular, de tal manera que cadascun dels mòduls serà el més independent possible dels altres, això ajudarà a possibles futures millores del desenvolupament i a la comprensió global del codi.

La llibreriaconté els següents mòduls que tenen el següent ús:

Mòdul:	Funció
Filtre de dades	Transformar les dades del capturador en moviment
Enllaç Robot	Prepara les ordres per enviar a la placa RCB3-J del robot

Taula 38. Mòduls Disponibles.

8.1.2. El Mòdul de filtratge de dades

El cor d'aquest mòdul és el càlcul de la matriu de rotació calculada a partir de les dades dels capturador de dades, amb aquesta matriu de rotació es possible realitzar totes les eines que és vulguin desenvolupar, actualment el mòdul disposa d'una única eina:

- Càlcul del moviment

Amb el càlcul de la matriu de rotació s'inclou el càlcul de les tres matrius de rotació que s'han descrit anteriorment, la realitzada amb el giroscopis, o matriu dinàmica, la calculada a partir de l'acceleròmetre i el magnetòmetre o matriu estàtica i finalment la fusió de les dues matrius és a dir la matriu complementaria.

8.1.2.1. Càlcul del moviment de l'usuari

Aquesta eina permet calcular quin moviment general està realitzant el usuari, d'aquesta

manera es podria detectar si l'usuari esta caminant cap endavant, caminant lateralment, estirant-se o aixecant-se, per exemple.

Per desenvolupar aquesta idea es disposa d'un braçalet que permet integrar el capturador de dades sobre l'usuari, d'aquesta manera es poden rebre les dades quan l'usuari està en moviment, analitzar-les i obtenir un resultat sobre quin és el moviment que s'està realitzant.

En aquests moments és vol que la llibreria pugui distingir els següents moviments:

- Passa Endavant
- Passa Enrere
- Passa lateral Esquerra
- Passa lateral Dreta

I les següents accions:

- Estirar-se
- Aixecar-se
- Gir 90º Esquerra
- Gir 90º Dret

Detecció de moviments:

Per a la realització de totes les proves, per una limitació en la quantitat de moviments seguits que es poden realitzar, principalment degut a l'espai, l'adquisició de dades és realitza amb cinc moviments seguits del mateix tipus, i encara que no sigui una mostra molt representativa permetrà treure conclusions sobre les possibilitats de realitzar un algoritme de detecció del moviment realitzat.

1era aproximació per la detecció de moviments:

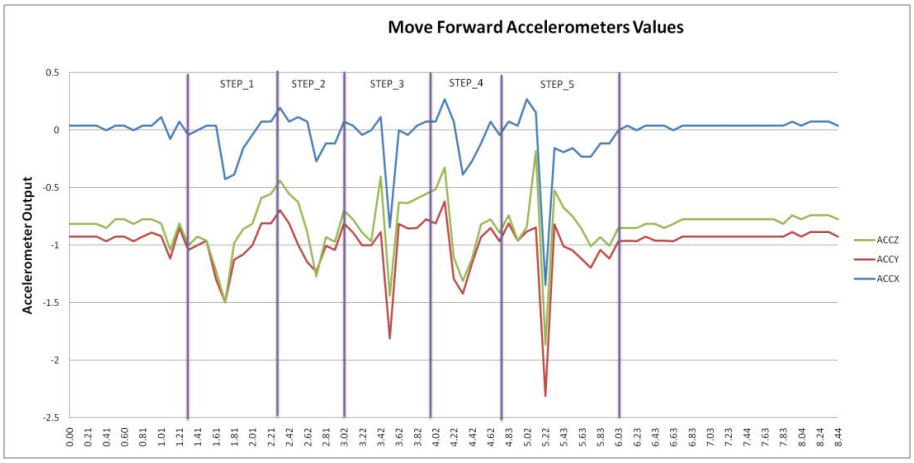
Prerequisits:

- Únicament amb l'ús de la matriu estàtica.
- Col·locació del capturador de dades: Utilitzant únicament la matriu estàtica es disposa el capturador de dades sobre l'usuari d'una manera que provoqui un canvi en l'acceleròmetre quan hi ha un moviment en principi es decideix per comoditat gracies a l'ús d'una canellera penjar-ho sobre el braç.

Anàlisi de les dades:

És perfectament normal que una persona faci un moviment realitzi un balanceig dels braços que ajudi a l'estabilitat, es aquest balanceig que es tracta de quantificar i analitzar mitjançant variacions en les lectures de l'acceleròmetre, per tal de conèixer quin és el moviment realitzat per l'usuari.

Amb la realització del test de 5 moviments endavant obtenim la següent sortida dels acceleròmetres:

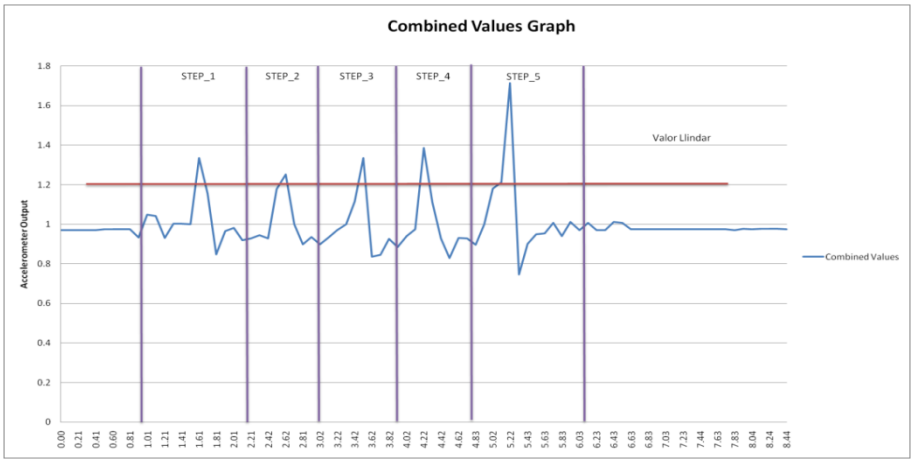


Gràfica 13. Sortida dels Acceleròmetres passa endavant.

Es pot veure a la gràfica com es produeixen canvis significatius en la lectura dels acceleròmetres quan és produeix alguna passa de l'usuari, la combinació dels tres valors de cadascun dels eixos de l'acceleròmetre es calcula a partir de l'equació:

$$combined\_values = \sqrt{Accx^2 + Accy^2 + Accz^2}$$

El valor de sortida dels valors combinats tindrà un valor de 1 sempre que el capturador de dades es mantingui quiet, en quant es produeixi un moviment es produirà un desequilibri en els valors de l'acceleròmetre que provocarà valors diferents a 1, aprofitant això es perfectament possible detectar quines son les passes que ha realitzat l'usuari mitjançant les lectures que obtenim dels valors combinats de l'acceleròmetre,



Gràfica 14. valors Combinats de l'Acceleròmetre.

Analitzant els canvis de la gràfica dels valors de l'acceleròmetre combinats es pot definir una **passa realitzada per l'usuari com un canvi de direcció en el valor si succeeix passat un determinat llindar**, com també es pot veure a la gràfica.

Basant-se en això es realitza el següent algoritme per la detecció de dades:

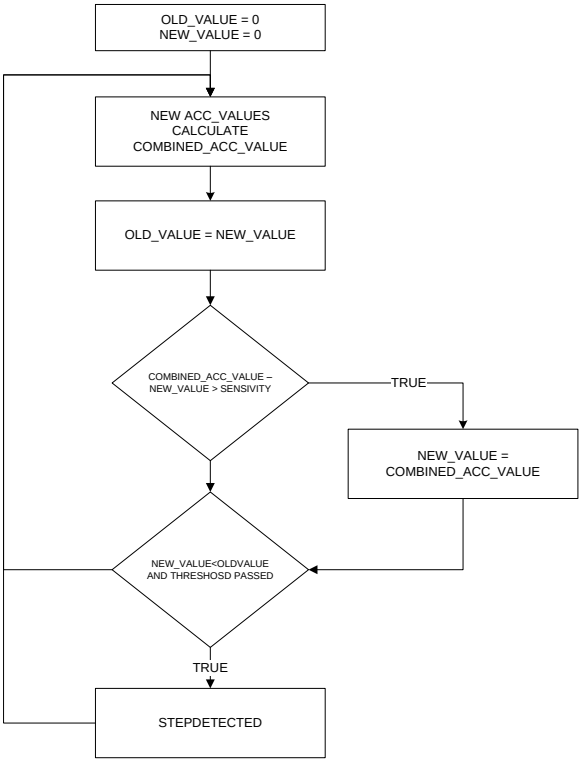


Figura 77. Algoritme de detecció de passes.

Aquest algoritme està basat en el document[15], s'han realitzat algunes modificacions per tal d'adaptar-lo en aquesta configuració. Bàsicament hi ha dos trets diferencials, en primer lloc l'algoritme del document es basa en les dades d'un únic eix de l'acceleròmetre, en aquest cas en analitzar les dades de l'acceleròmetre s'ha preferit utilitzar les dades combinades dels 3 eixos donat que l'anàlisi en un únic eix potser no simptomàtic que l'usuari hagi realitzat una passa. Finalment s'ha realitzat el nivell del llindar en el cas de l'algoritme del document es calcula dinàmicament cada 50 mostres, en aquest cas hem decidit utilitzar un llindar fixa que permet perdre menys passes tot i que el fa molt menys flexible que la solució del document.

S'ha trobat un possible primer algoritme que detectaria si l'usuari a realitzat una passa endavant, però a priori i mirant les dades no podríem distingir si l'usuari ha realitzat la passa enrere donat que en el valor de les dades combinades no tenim cap tipus d'informació sobre quin és la direcció del moviment, a més un anàlisi de les dades de cadascun dels eixos per separat tampoc ajuden degut al soroll dels acceleròmetres.
Quan es realitza el moviment lateral però no sembla natural el moviment de balanceig dels braços que l'usuari realitza quan es mou endavant per tant concloem que no és possible tampoc la detecció d'un moviment lateral cap a cap sentit amb el capturador de dades sobre el braç.

Així utilitzant les dades de l'acceleròmetre amb el capturador de dades sobre el braç s'obté la següent taula:

MOVIMENT	CAPACITAT DE DETECCIÓ
Pas Endavant	X
Pas Enrere	O
Pas lateral Esquerra	O
Pas lateral Dreta	O

Taula 39. Detecció moviments aproximació 1.

2ona aproximació per la detecció de moviments:

Prerequisits:

- l'Ús de la matriu estàtica i complementaria.
- Per a mirar de detectar més moviments s'intentarà modificar la posició del capturador de dades, analitzant els moviments que es volen capturar es pot observar que tots estan relacionats amb la posició dels peus, per tant és raonable pensar que si el capturador de dades estar sobre les cames de l'usuari es podran obtenir dades més exactes sobre quin és el moviment que l'usuari està realitzant, per altre banda també interessa que la posició del capturador provoqui la reacció més gran possible, així es decideix canviar la posició del capturador de dades del braç al turmell del peu de referència de l'usuari, aquest peu de referència és el que es mou en primer lloc quan s'inicia una passa.

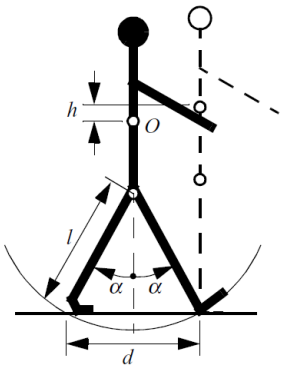


Figura 78. Caminar endavant/Enrere.

Anàlisi de les dades:

Tal com es veu es por veure existeix una diferencia en l'angle que fa la cama respecte al terra i es pot saber quin és el sentit del moviment de l'usuari. El capturador de dades esta situat de tal manera que les dades que es reben en estàtic, és a dir amb l'usuari quiet i de peu amb les cames rectes i en paral·lel, idealment per l'acceleròmetre siguin:

AccX: 0 AccY: ±1 AccZ: 0

A més per conèixer exactament la posició del capturador es defineix que el moviment de

la cama cap endavant - enrere es realitzi en una rotació sobre l'eix Z, això voldrà dir que en aquesta posició la matriu ens donarà la següent informació:

Vx0, Vx1, Vx2 Informació de rotació

Vy0, Vy1, Vy2 Informació de translació

Vz0, Vz1, Vz2 Informació de rotació

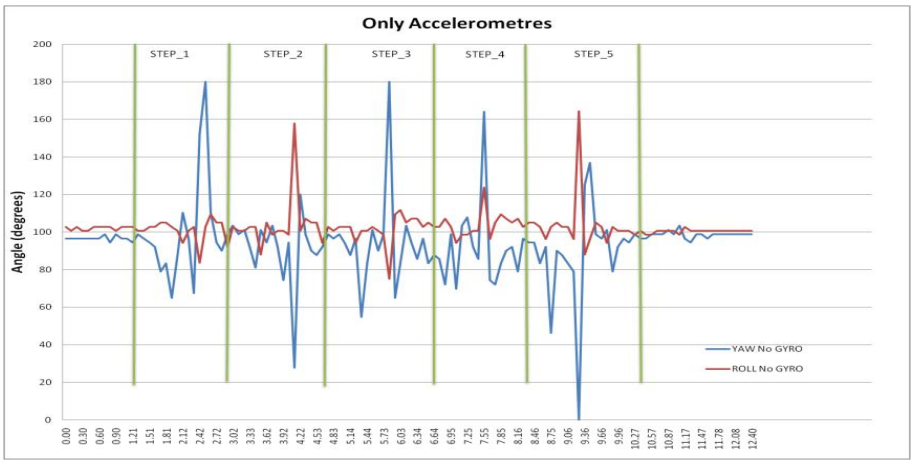
Sabent això es pot establir les equacions per conèixer el moviment:

$$\varphi = \text{asin}(Vy1) \quad \text{Eq. 42}$$

$$\varphi = \text{acos}(Vy0) \quad \text{Eq. 43}$$

$$\varphi = \text{atan2}(\frac{Vy1}{Vy0}) \quad \text{Eq. 44}$$

S'obtenen els següents resultats:



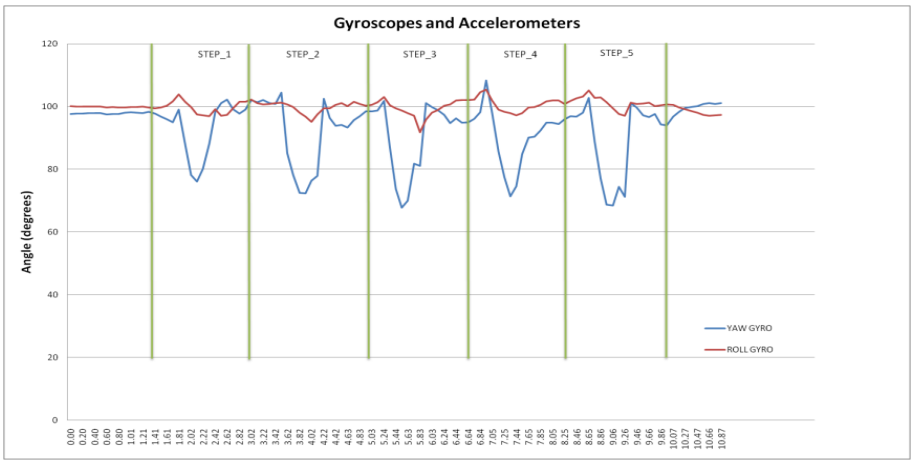
Gràfica 15. Moviment endavant amb Capturador al Peu.

Com és pot veure en la gràfica apareix un nou problema que és el soroll quan és realitza el moviment que fa molt difícil fins i tot la detecció de les passes endavant.
Així per culpa del soroll no es possible distingir ni el sentit ni la direcció del moviment,

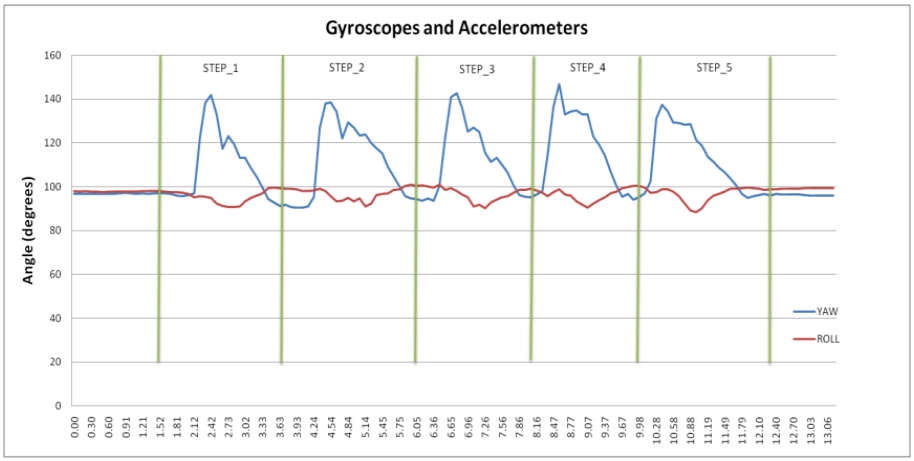
MOVIMENT	CAPACITAT DE DETECCIÓ
Pas Endavant	X
Pas Enrere	X
Pas lateral Esquerra	X
Pas lateral Dreta	X

Taula 40. Detecció moviments aproximació 2.

És passa a continuació a realitzar els mateixos càlculs amb la matriu complementaria, en teoria el fet d'utilitzar aquesta matriu ens hauria de reduir considerablement el soroll,



Gràfica 16. Moviment Endavant amb Capturador al Peu (filtre complementari).



Gràfica 17. Moviment Endavant amb Capturador al Peu (filtre complementari).

No només es veu que les lectures no tenen soroll i ens permetran conèixer la direcció de la passa sinó que tal com es veu a les gràfiques podem distingir quin és el sentit gracies a que l'angle té una canvi diferent en funció del sentit del moviment de l'usuari.

En les gràfiques anteriors també s'ha representat el valor de rotació sobre l'eix X, que també coneixem com a ***Roll***, aquest angle és pot utilitzar per saber si s'està realitzant un moviment lateral. Les equacions per trobar aquest angle, agafant les dades de la matriu de rotació, son:

$$\alpha = asin(Vy1) \quad \text{Eq. 45}$$

$$\alpha = acos(Vy2) \quad \text{Eq. 46}$$

$$\alpha = atan2(\frac{Vy1}{Vy2}) \quad \text{Eq. 47}$$

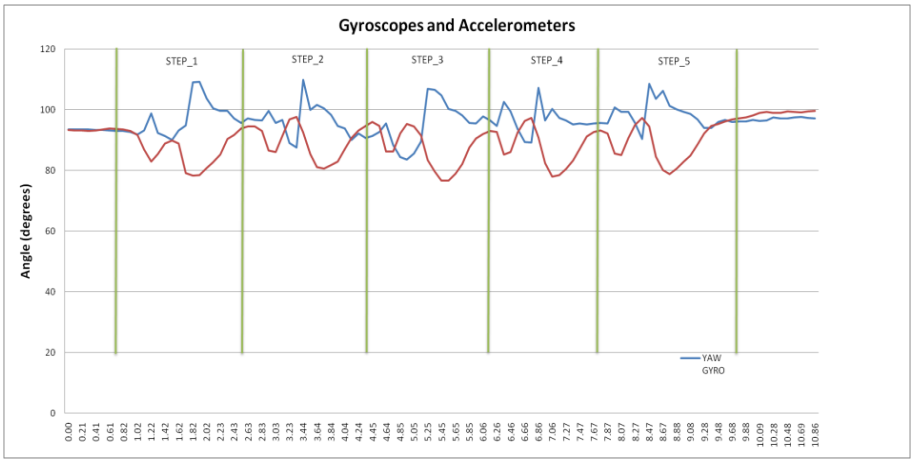
Es vol captar els dos moviments a la vegada, per tant es comprova a continuació que passa

quan es realitzen els dos moviments a la vegada, lògicament les dades de cada moviment no han d'interferir en l'altre moviment, per tant s'han de seleccionar variables independents en cada moviment, per tant:

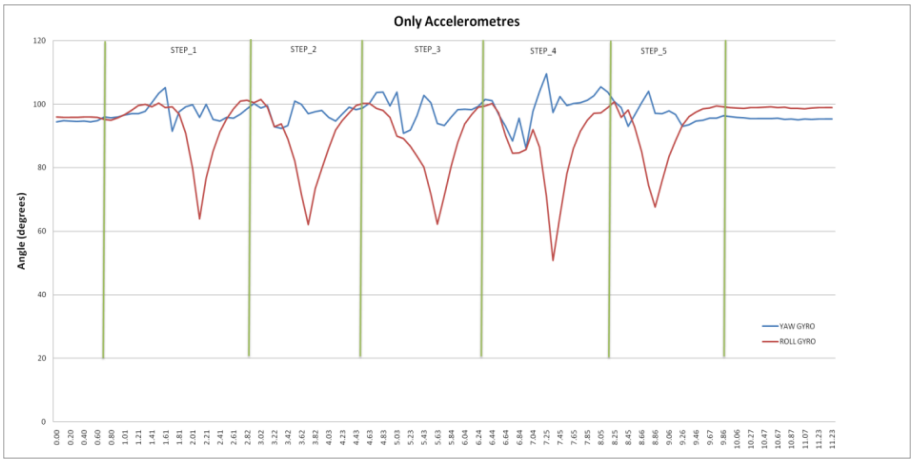
Moviment Frontal: $\varphi = \text{acos}(Vy0)$

Moviment Lateral: $\alpha = \text{acos}(Vy2)$

Com és pot veure en les gràfiques un moviment endavant- enrere no modifica l'angle de Roll, però si es realitza un moviment lateral, el filtre complementari de detectar-ho i donar la correcta direcció del moviment com veiem a les següents gràfiques:



Gràfica 18. Moviment Lateral Dret amb Capturador al Peu (filtre complementari).



Gràfica 19. Moviment Esquerra amb Capturador al Peu (filtre complementari).

És perfectament possible detectar quina és la direcció dels moviments, donat que cada moviment modificà un angle diferent o en qualsevol cas el modifica d'una manera més notable, segurament el més difícil d'aconseguir serà distingir entre quin és sentit (esquerra o dret) en els moviments laterals, d'aquesta manera amb la matriu complementaria es poden distingir els següents moviments:

MOVIMENT	CAPACITAT DE DETECCIÓ
Pas Endavant	O
Pas Enrere	O
Pas lateral Esquerra o Dret	O

Taula 41. Detecció moviments aproximació 3.

Detecció d'accions:

Tal com s'ha vist abans estan disponibles les següents accions:

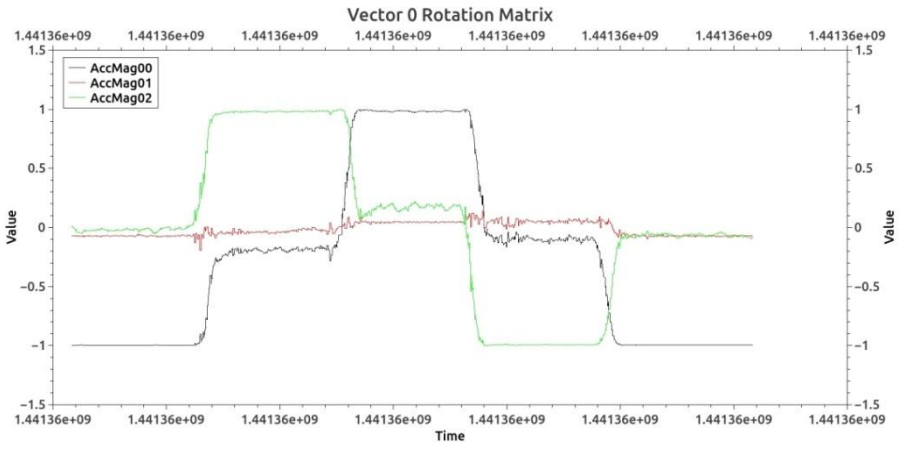
- Estirar-se
- Aixecar-se
- Gir 90º Esquerra
- Gir 90º Dret

És disposa el capturador de dades sobre el turmell i situat de tal manera que les dades que es reben en estàtic, és a dir amb l'usuari quiet i de peu amb les cames rectes i en paral·lel, siguin:

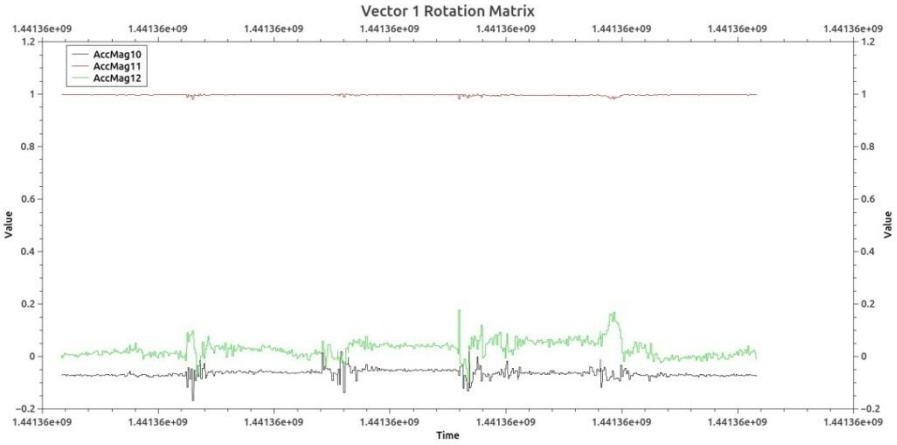
AccX: 0 AccY: ±1 AccZ: 0

Per detectar que l'usuari s'ha estirat al terra s'haurà d'esperar que el valor Vy0 de la matriu de rotació tingui aproximadament un valor de ±1 i per saber que si s'ha tornat a aixecar aquest mateix valor ha de ser aproximadament 0. Alternativament es podria fer un seguiment sobre la variable Vy1 i esperar a que tingues un valor de 0 per saber si l'usuari s'ha estirar i un valor de ±1 per saber si s'ha tornat a aixecar.

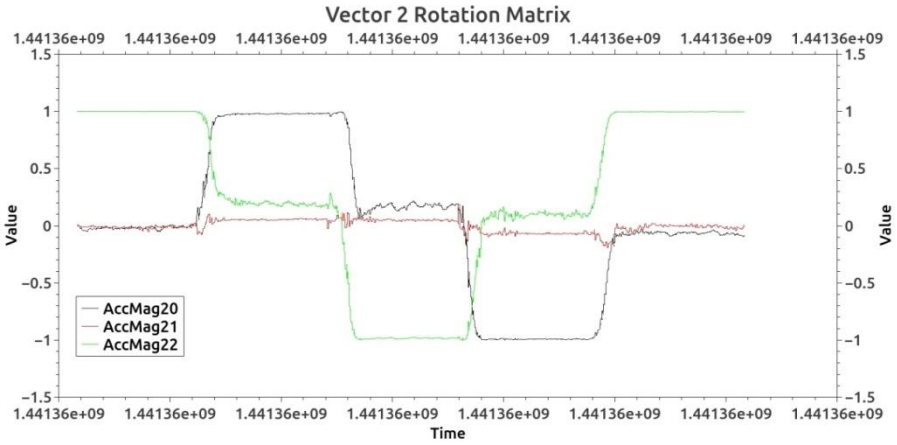
Per capturar un gir sobre si mateix de l'usuari els vectors que s'hauran de consultar de la matriu de rotació son el Vx o el Vz, que son els que variaran amb l'actual disposició del capturador de dades, les dades que obtenim per cadascun dels paràmetres de la matriu en cas de realitzar un gir complert del capturador de dades es poden comprovar en les següents gràfiques:



Gràfica 20. vector Vx Matriu de Rotació.



Gràfica 21. Vector Vy Matriu de Rotació.



Gràfica 22. Vector Vz Matriu de Rotació.

Tal com es pot veure a les gràfiques i s'ha indicat abans el vector Vy no dona cap informació, aquest és el vector de l'acceleròmetre i com al moviment no l'afecta la força de la gravetat el vector és manté inalterable, els altres dos vectors si poden ser utilitzats per conèixer quina és la orientació de l'usuari. Tant si s'utilitzen el vector Vx com el vector Vz es poden utilitzar les següents equacions per conèixer l'actual orientació:

$$\alpha = asin\ (Vx0) \qquad \text{Eq. 48}$$

$$\alpha = acos\ (Vx2) \qquad \text{Eq. 49}$$

$$\alpha = atan2\ (\frac{Vx0}{Vx2}) \qquad \text{Eq. 50}$$

8.1.3. El mòdul de control del robot

La idea d'aquest mòdul es donar eines per treballar amb el robot, principalment disposa d'un control d'emmagatzematge de moviments. Aquesta eina permet llançar moviments i en cas que el robot estigui ocupat llançant un moviment esperar a que estigui lliure per anar llançant els moviments que estan acumulats.

Mètode	Descripció
Initialize	Inicialitza la connexió amb el robot e inicialitza el fil d'execució de moviments.
launchMovement	Afegeix una nou punter a una instancia de ***Movement*** a la cua, aquest instància conté informació sobre quin és el moviment que es vol llançar.

Taula 42. Mètodes disponibles.

A continuació un esquema de com funciona el fil d'execucions:

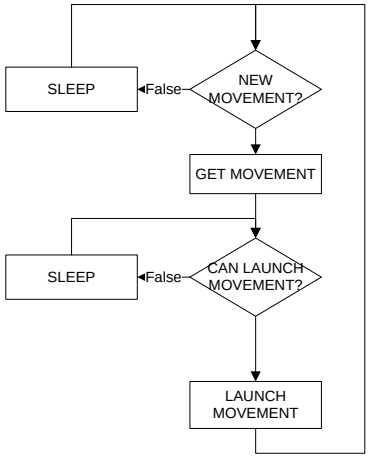


Figura 79. Esquema fil llançament de moviments.

8.2. Software utilitzat

Per tal de establir una comunicació amb els capturadors de dades seleccionats es necessari disposar de ***Bluetooth***, a més en el cas del telèfon mòbil, també és possible comunicar-se mitjançant una interfície de xarxa TPC/IP. Les interfícies de xarxa per TCP/IP estan disponibles a tots els sistemes operatius, l'***stack*** de Bluetooth també està disponible en molts S.O. però per sort el projecte està realitzat sobre un SO Linux on a partir de la versió de **kernel** 2.4.6. ofereix de sèrie un ***stack*** Bluetooth anomenat ***Bluez***, i a més disposa d'una sèrie d'eines que ens permetran atacar l'***stack*** del Bluetooth directament, com:

- **hcitool**: permet l'enviament de comandes HCI a traves del dispositiu Bluetooth local
- **hcidump**: permet visualitzar les dades que passen a través de la capa de HCI
- **sdtools**: Permet fer peticions sobre el protocol SDP.

Aquestes eines ens permetran obtenir informació per aprendre quin és el funcionament en general de qualsevol capturador que es connecti mitjançant l'***stack*** de Bluetooth i en particular del Wiimote.
Respecte a la comunicació amb el robot s'han de tenir instal·lats els controladors per comunicar-se amb els xips ftdi que passen de USB a port serial. Finalment per tal de treballar amb les matrius s'ha utilitzat una llibreria matemàtica de C++ anomenada **armadillo**.

8.3. Descripció d'aplicacions

S'han realitzat 2 aplicacions per simular l'ús de la llibreria, per utilitzar el capturadors de dades en els dos modes disponibles actiu i passiu.

8.3.2. Llançament de moviments

Aquesta és un aplicació de consola que enllaça cadascun dels botons dels capturadors de dades amb els moviments gravats dins de la placa de RCB3J, es podria pensar que és una mena de comandament a distància, d'aquesta manera quan es prem el boto endavant es selecciona quin és el moviment que es vol executar. Tal com es veu per la descripció els capturadors de dades treballaran en mode Actiu.

El diagrama de flux de l'aplicació:

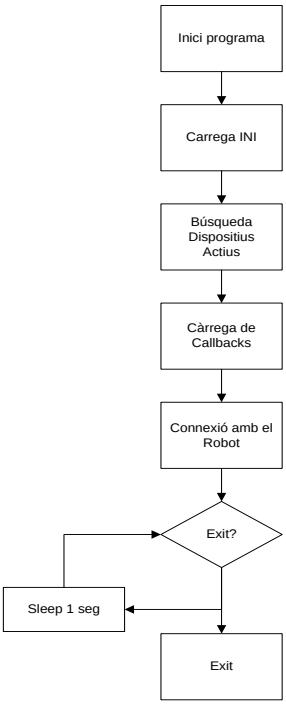


Figura 80. Diagrama de flux de l'aplicació de llançar moviments.

Per no tenir l'enllaç dels botons amb els moviments ***hardcodejats***, s'ha decidit utilitzar un fitxer .ini per guardar la configuració, així no s'ha de compilar l'aplicació per cada canvi que es decideixi realitzar en la relació botó - moviment, la càrrega d'aquest ini és el primer que realitza l'aplicació, un cop finalitzada, realitza la cerca i la configuració dels capturadors de dades en mode actiu, indicant quin és el ***callback*** que s'executarà cada vegada que es premi un botó, finalment estableix la connexió amb el robot, a partir d'aquí l'aplicació es manté esperant que l'usuari decideixi finalitzar l'aplicació i dona el control total al **Wiimote**, que executarà el callback per cada botó premut.

8.3.3. Repetició de moviments

La última aplicació realitzada enllaça els valors de moviment que son capturats per els

capturadors de dades, mitjançant la eina de càlcul de moviment i passar-li a la eina de llançament de moviments.

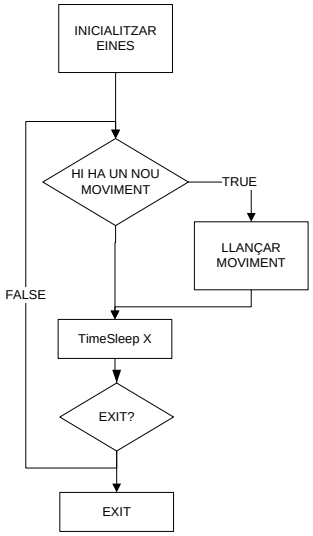
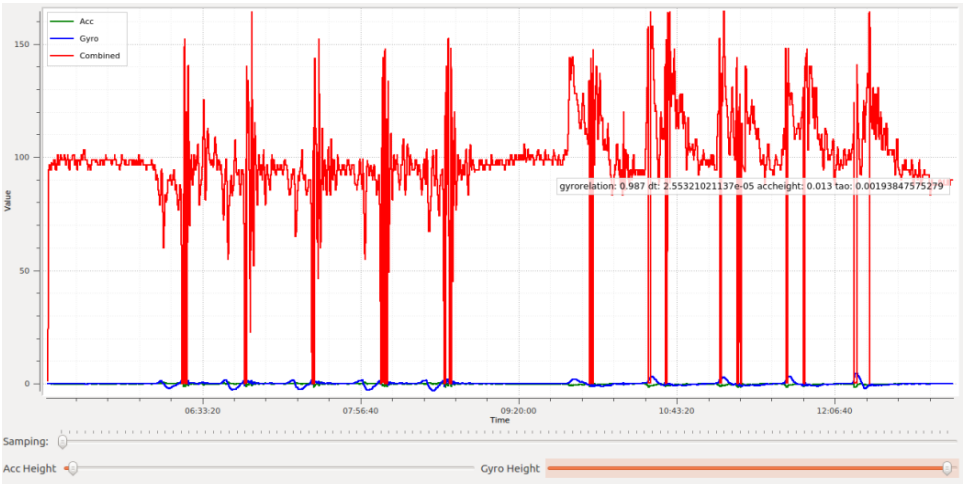


Figura 81. Diagrama aplicació llançament de moviments.

Tal com es veu en el diagrama s'ha d'indicar quin és el temps entre les lectures i el llançament del moviment, el valor d'aquesta variable és particularment important donat que influeix directament en el funcionament del filtre complementari i en com és produeix la fusió de les dades entre l'acceleròmetre i el giroscopi. Per conèixer quin és aquest valor disposem d'un petit programa que permet modificar els valors del filtre α i dt amb dades de 5 passes endavant i 5 passes enrere i així comprovar com varia la sortida del filtre complementari, les dades de mostra s'han agafat amb un dt lo mínim possible per tenir marge per calcular els paràmetres del filtre[4].

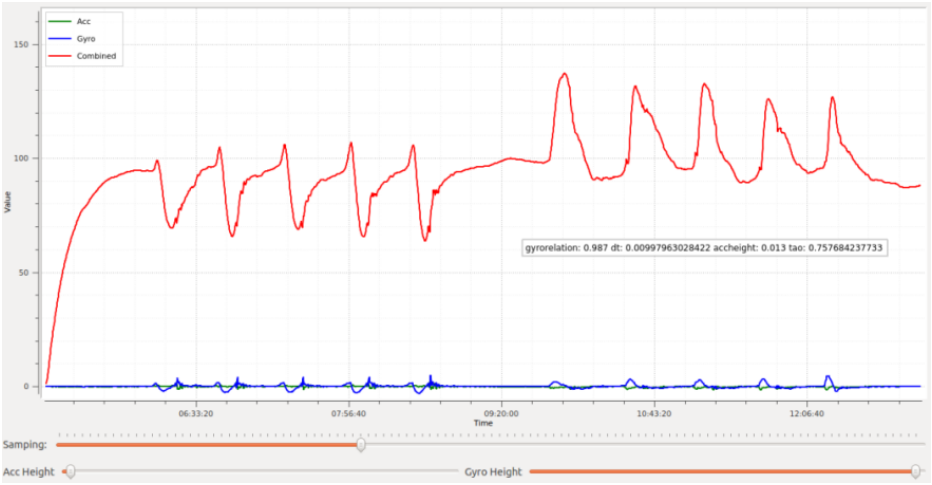
El gràfic amb les dades originals és i amb una α de 0.987 i una dt de 2.5e-05:



Gràfica 23. Sortida Filtre Complementari tao=0.0019.

Es pot veure com hi ha molt de soroll degut a que el temps entre mostres és massa petit, això fa que l'acceleròmetre trigui molt poc en començar a actuar i tingui massa pes en el filtre complementari, pujant el temps de mostreig fins a 20 ms:

[4]Dades disponibles a /data/complementary filter data



Gràfica 24. Sortida filtre complementari tao 0.75.

Tal com es pot veure en la última gràfica amb un temps de mostreig de 20 ms obtenim una sortida correctament filtrada, a més si veiem el càlcul de τ en el segon cas s'obté sobre uns 1.16 segons que s'entén és el que aproximadament ha durat una passa i per tant el temps a partir del qual és vol que la sortida de l'acceleròmetre es doni per bona.

9. Conclusions i Treball Futur

Per complir aquest llarg objectiu s'ha hagut de conèixer les possibilitats que ofereix el robot Kyosho Manoi i de quina manera es pot establir comunicació per tal que realitzi accions, conèixer també les possibilitats de sensors i actuadors que ens ofereixen els capturadors de dades seleccionats i aprendre'n a extreure les dades, a partir d'aquestes dades també s'han hagut realitzar els càlculs per tal d'obtenir informació que era necessària en aquest cas la posició en 3D dels capturadors de dades i finalment s'ha culminat el projecte fusionant la posició espaial del capturador amb el llançament d'ordres cap al robot. Durant el camí s'han pogut assolir coneixement associats a la programació, als filtres digitals i a les rotacions dels objectes en espais tridimensionals que esperem s'hagin pogut plasmar en idees pel lector, ja sigui en la memòria o en el codi font adjunt al projecte.

Tal com s'ha vist el projecte ha estat dividit en varies parts, i així també ha estat organitzat el codi, partint d'aquesta divisió, es pot fàcilment pensar en quines millores es podrien afegir per cada mòdul.

Respecte el robot, la base de la comunicació entre un host i el robot no ofereix errors de comunicacions, però la llibreria de control que s'ha desenvolupat per sobre només conté algunes de les funcions que s'han cregut necessàries i encara queden moltes per escriure, a més no només existeix el robot Kyosho Manoi al mercat, per tant una possible tasca seria actualitzar i homogeneïtzar l'accés al robot, de tal manera que des de les capes més altes fos indistingible quin tipus de robot humanoide estem controlant.

Respecte als capturadors de dades, donat que la capa ja esta prou homogeneïtzada simplement es podrien afegir nous tipus de controladors que complissin amb els patrons de disseny descrits en el codi. On és tindria més feina seria en la implementació de nous filtres per tractar les dades que s'obtenen dels capturadors de dades i conèixer quina és la posició espaial del capturador de dades, en aquest cas s'ha limitat a descriure un filtre complementari per la combinació de les dades de l'acceleròmetre i el giroscopi, però sabem que existeixen filtres molt més complexos però també més eficients per tractar les dades dels sensors, com un filtre Kalman.

Per últim, el codi es perfectament compatible amb plaques computadores tipus ***Raspberry***, per tant seria possible utilitzar aquesta placa com a host i muntar-la sobre el Kyosho Manoi d'aquesta manera seria possible treure tots els cables USB i això donaria moltes més possibilitats al robot en quant a moviments.

10. Referències

[1] SIEGWART ILLAH R., NOURBAKHSH ROLAND, Introduction to Autonomous Mobile Robots, p.91

[2] SIEGWART ILLAH R., NOURBAKHSH ROLAND, Introduction to Autonomous Mobile Robots, p.89

[3] Jon S. Wilson, Sensor Technology Handbook p.150

[4] SIEGWART ILLAH R., NOURBAKHSH Roland Introduction to Autonomous Mobile Robots, p.100

[5] SIEGWART Illah R., NOURBAKHSH Roland Introduction to Autonomous Mobile Robots, p.98

[6] SIEGWART Illah R., NOURBAKHSH Roland Introduction to Autonomous Mobile Robots, p.13

[7] SIEGWART Illah R., NOURBAKHSH Roland Introduction to Autonomous Mobile Robots, p.10

[8] rcb3commandref3.pdf, Commands ReferencesKONDO KAGAKU CO., LTD (disponible a docs_pfc\documentació\Robot)

[9] Exemple complert disponible a: Exemples\Python Code Examples\Robot\MoveAbsoluteMotor.py

[10] Exemple complert disponible a docs_pfc\Exemples\C code Examples\DevSearch.c

[11] Exemple disponible a docs_pfc\Exemples\C code Examples\ComBasic

[12] Exemple disponible a docs_pfc\Exemples\C code Examples\TurnLights

[13] D. Smith. (2010, September2). Cwiid Library and Trac Available: http://abstrakraft.org/cwiid

[14] Shane Colton, "filter.pdf," June 2007. (Disponible at docs_pfc\documentació\Altres)

[15] Neil Zhao, pedometer.pdf, June 2010. (Disponible at docs_pfc\documentació\Altres)

[16] SIEGWART Illah R., NOURBAKHSH Roland Introduction to Autonomous Mobile Robots, p.15

Altre material consultat:

[17] ALBERT HUANG, LARRY RUDOLPH, "Bluetooth for Programmers"

[18] MARK LUTZ, "Learning Python", O'Reilly, 2009

[19] ZIGURD MEDNIEKS, LAIRD DORNIN, G. BLAKE MEIKE, MASUMI NAKAMURA,

[20] Programming Android: Java Programming for the New Generation of Mobile Devices,O'Reilly, 2012

[21] JASMIN BLANCHETTE, MARK SUMMERFIELD, C++ GUI Programming with Qt4, Prentice Hall Open Source Software Development, 2008

[22] BJARNE STROUSTRUP, Programming: Principles and Practice Using C++, Addison Wesley, 2014

Manuals: (Disponibles a la carpeta de documents)

[23] ManoiEnglish.pdf, 1/5 scale atlethe humanoid Manoi AT01,Instruction manual 2006KONDO KAGAKU CO.,LTD (disponible a docs_pfc\documentació\Robot)

[24] HtH3operationManual.pdf, RCB-3 Application Software HeartToHeart3 Instruction manual 2005KONDO KAGAKU CO.,LTD (disponible a docs_pfc\documentació\Robot)

[25] KRG-4.pdf, Modifications guide No.08J27-1, KONDO KAGAKU CO.,LTD (disponible a docs_pfc\documentació\Robot)

[26] Acceleration sensor RAS-1 / RAS-2 for KHR-1HV 2008.01 Ver1.0, KONDO KAGAKU CO.,LTD(disponible a docs_pfc\documentació\Robot)

[27] KRC-3AD Wireless ControllerInstruction Manual, translated by Kumotec

[28] ADXL335full de dades (disponible a docs_pfc\documentació\Robot)

Pàgines web:

[29] Unknown, (19 Setembre 2015)Wiimote Wiki and Library http://wiibrew.org/wiki/Wiimote/Library

[30] Unknown, https://developers.google.com/protocol-buffers/

[31] Unknown, http://www.intra2net.com/en/developer/libftdi/index.php

Apèndix

Apèndix A: Connectar un dispositiu Android al PC

La connexió amb un dispositiu mòbil via WIFI té el problema que per realitzar-se és necessària tota una infraestructura prèvia, a no ser que s'utilitzi una xarxa ad-hoc, per desgràcia per defecte un dispositiu Android no permet la seva connexió amb aquest tipus de xarxa, per tal d'aconseguir que un dispositiu Android pugui establir connexió amb un PC sense infraestructura, s'haurà de transformar en un punt d'accés sense fils, per això s'ha decidit utilitzar l'aplicació ap-hotspot, realitzant les següents passes:

```
$ sudo su -
$ add-apt-repository ppa:nilarimogard/webupd8
$ aptitude update
$ aptitude install ap-hotspot
$ ap-hotspot configure
$ ap-hotspot start
```

Durant la configuració únicament s'haurà de seleccionar la interfície de xarxa, el SSID de la xarxa i la paraula de pas per tal de realitzar la connexió.