# DISSENY I CONSTRUCCIÓ D'UNA TAULA AUXILIAR PER A PERSONES AMB MOBILITAT REDUÏDA

Pol Gálvez
Escola Mestral
Tutor: Dani Urbano
Novembre 2017

# Índex

# 1. Introducció

L'objectiu d'aquest treball consisteix en el disseny i construcció d'una taula mòbil auxiliar controlada mitjançant un telèfon intel·ligent que permeti a les persones amb dificultats motrius parar i desparar la taula.

La taula mòbil que es pretén construir està formada per dos taulons de fusta sostinguts per una estructura que n'eleva un a l'alçada dels braços. La part electrònica està controlada per un *Arduino*, un microcontrolador de codi obert que permet fer accessible el disseny de circuits electrònics i que, per tant, es pot utilitzar per a desenvolupar objectes interactius autònoms.

El que em va motivar a fer aquest treball va ser pensar en un projecte o idea que contribuís a millorar i facilitar la vida a les persones amb handicaps motrius i, sobretot, que pogués tenir una utilitat pràctica per a aquest col·lectiu.

El treball està principalment dividit en quatre parts:

La primera part –la construcció de la taula- és la part en la qual s'explica la construcció de l'estructura de la taula mòbil. També s'explica el mètode d'ancoratge dels motors a l'estructura i a les rodes utilitzades.

En la segona part -el moviment de la taula- és on s'explica el funcionament dels motors, quins motors hem escollit i com es controlen amb l'*Arduino* i amb un *driver* específic.

En la tercera part -el mòdul de comunicació- és on s'explica com es fan interactuar un telèfon intel·ligent i un microcontrolador com l'*Arduino*, mitjançant *bluetooth*.

En l'última part -el codi i circuit complet- és on es mostra una combinació dels dos programes usats en les parts anteriors i l'addició de les parts necessàries perquè l'*Arduino* interpreti les dades en temps real i obtenir moviment de la taula immediat a distància.

## 2. Justificació del projecte

En començar el treball ens vàrem fixar dues condicions que havia de complir el nostre projecte. D'una banda, havia d'utilitzar com a element bàsic una placa de microcontrol *Arduino*, ja que aquest és un dispositiu que dona la possibilitat de fer gairebé qualsevol projecte de robòtica de manera senzilla i econòmicament accessible. D'altra banda, havia de ser un objecte útil que millorés la qualitat de vida de persones amb dificultats o handicaps motrius.

Gràcies a la Sara Suàrez, que desenvolupa part de la seva activitat en el món de l'esport adaptat, vàrem entrar en contacte amb el José Luís. El José Luís és un noi que va néixer amb una patologia que li impedeix moure les cames, tot i això viu sol i de manera autònoma. A més a més, és jugador de tennis en cadira de rodes. Ens va proposar dissenyar una taula mòbil auxiliar, no gaire gran, que li facilités el procés de parar i desparar la taula ràpidament i sense ajuda de ningú. Ens va semblar bastant bona idea i vàrem decidir escollir la proposta del José Luís com a projecte. Més endavant, la Sara ens va concertar una entrevista amb el Jose Luís per preguntar-li per les seves necessitats concretes, tenint en compte la seva situació personal, i també quines característiques creia que havia de tenir l'aparell.

En Jose Luís ens va explicar que ell té prou autonomia en la seva vida quotidiana per realitzar la majoria de tasques a casa tot sol, però si es donava la circumstància que venia algun convidat a casa, tot i poder cuinar l'àpat, li costava molt portar els plats, gots, paelles, olles o altres estris d'un lloc a un altre. Aquests estris, a més, acostumen a ser pesants, a estar calents i fan que aquest tipus d'activitat, que es fa amb normalitat per a gent sense problemes de mobilitat, per a persones com ell sigui ben incòmoda i, fins i tot, perillosa. Així que buscava una taula mòbil auxiliar que fos capaç de transportar aquest tipus d'objectes i que fos controlable a distància .

Li vàrem demanar que ens fes una descripció de casa seva, en concret del menjador i de la cuina, que són les zones per on la taula s'hauria de moure.

En el seu cas la cuina i el menjador estan de costat i no hi ha cap paret o porta que dificulti el pas d'un aparell d'aquesta mena.

La distància màxima que el dispositiu hauria de recórrer no és superior als 10 metres. La bateria hauria de durar unes 5 hores estant el mòbil en moviment per a fer-lo raonablement autònom.

La superfície útil de la taula hauria d'estar a l'alçada dels braços que, en el cas d'una cadira de rodes estàndard, és de 700 mil·límetres del terra.

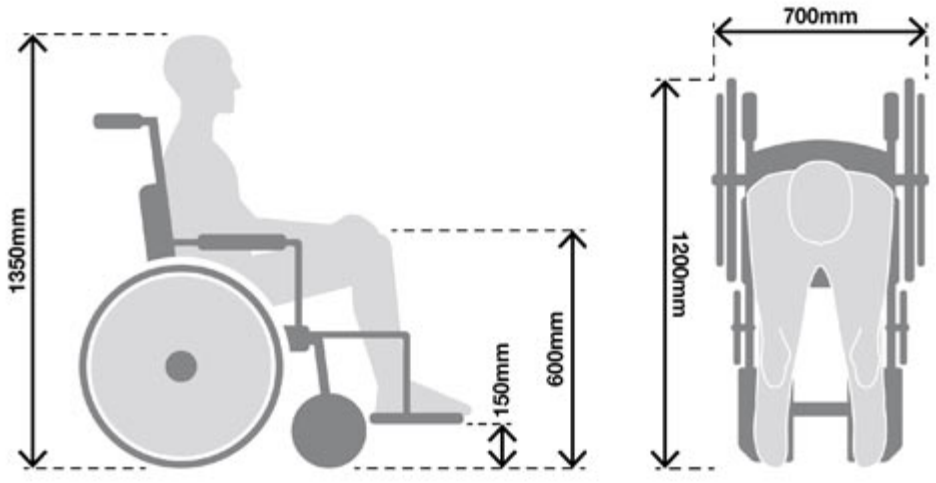


Fig. 1. Mides estàndard d'una cadira de rodes per a adults

# 3. Procés de disseny

## 3.1 Construcció de la taula

El primer que vàrem fer va ser realitzar amb *Draft Sight*, un programari de disseny CAD, un prototip de la taula que volíem construir.

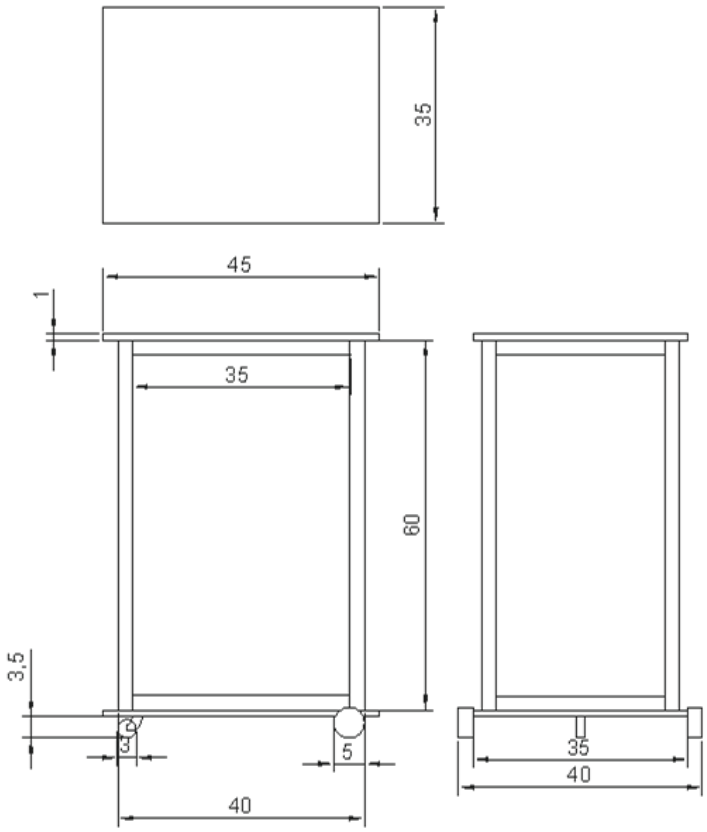


Fig 2. Planta alçat frontal i lateral
del disseny fer amb *DraftSight*

Aquest primer disseny consta de dos taulons de fusta, un per a la base i un altre per a la superfície de la taula. La base té dues rodes, una a cada part davantera dels laterals i una tercera roda, giratòria, al darrere. Per girar fa anar més ràpid la roda del costat contrari al qual vol girar i la del darrere segueix la direcció marcada per les frontals. D'altra banda, consta d'una estructura en forma d'ortoedre que aguanta el taulell superior a una certa alçada. Per fer aquesta estructura vàrem pensar a fer servir o bé tubs de PVC o bé un altre tipus de material que fos còmode de muntar com el *Mecalux*.

Un cop dissenyat el projecte, vàrem comprar un tauló de conglomerat de fusta per fer la base i la superfície de la taula. Aquest material és resistent, lleuger i econòmic i ,per tant, molt adient a les nostres necessitats. El tauló original era de 1200x600x10 mil·límetres, però el vàrem tallar en dues peces, una per a la base i l'altra per posar els estris a transportar, de 350 mil·límetres d'amplada i 450 mil·límetres de llargada.

Fig. 3 i 4.Procés de tall dels taulons

Amb aquestes mides la taula pot transportar gots, coberts i plats d'un sol trajecte i olles o utensilis més grans per separat. Un cop tallats els taulons, vàrem dissenyar l'estructura per mantenir la superfície on es col·loquen els utensilis de la llar a una alçada còmoda per a la interacció. Per construir l'estructura vàrem decidir usar *Slotted angles*, també coneguts com a Metalux, unes peces d'alumini en forma d'angle recte foradades pensades per fer prestatgeries.

Fig. 5 i 6. *Slotted* angles o *Metalux*

Vàrem valorar, basant-nos en l'alçada mitjana dels seients de les cadires de rodes, que el taulell superior havia d'estar a 60 centímetres de terra, a fi de situar-lo a una alçada inferior a la de la taula estàndard i superior al seient de la cadira. Les rodes eleven la base disposant-la a 5 centímetres de terra, per tant, vàrem demanar al Pol, l'encarregat del manteniment de l'escola, que ens tallés quatre peces de 390 mil·límetres, dues per a la part de baix i dues per a la part de dalt, així com quatre peces de 300 mil·límetres i quatre més de 550 mil·límetres per fer el suport horitzontal i vertical, respectivament.

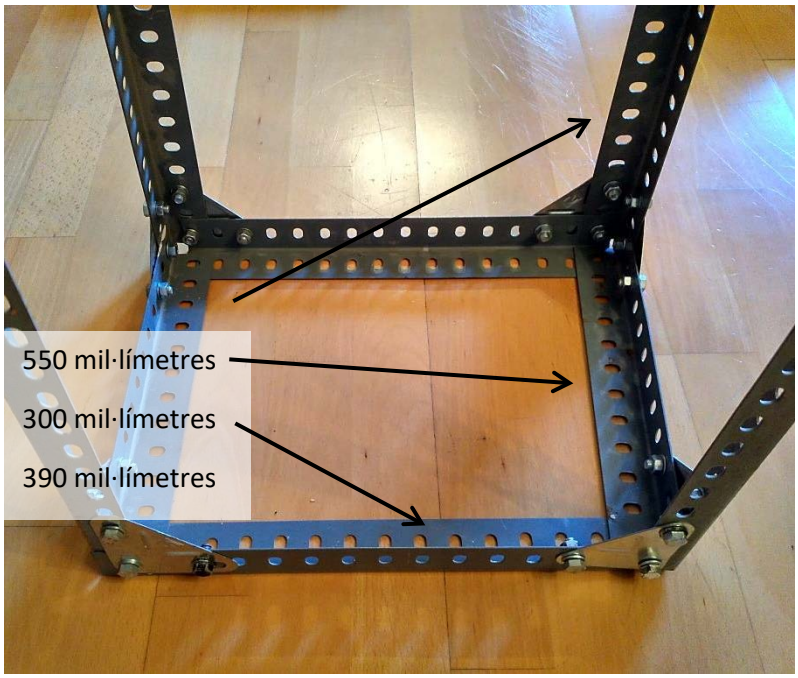


Fig. 7. Estructura de *Mecalux* amb mides

Un cop les dotze peces estaven tallades vàrem muntar-les formant un ortoedre. Vam fer servir cargols i uns triangles d'alumini amb tres punts d'ancoratge que proporcionen molta rigidesa a l'estructura.

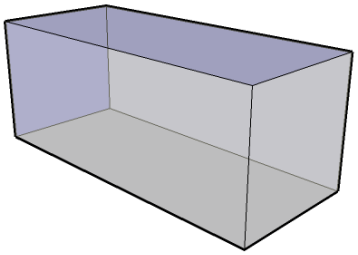

Fig. 8. Ortoedre

Fig. 9. Estructura de *Metalux* amb el tauló superior

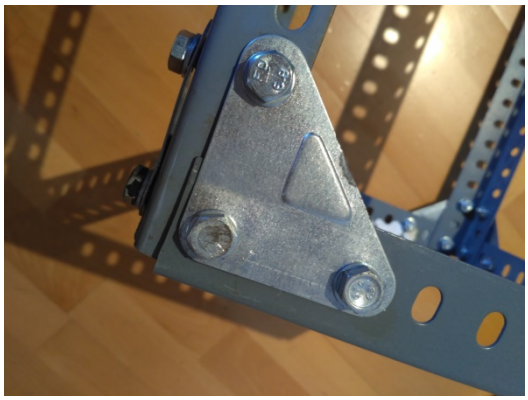

Fig. 10. Triangles d'alumini per unir el *Metalux*

Rere muntar l'ortoedre vàrem fer servir pintura en esprai de color negre sense brillantor per pintar l'estructura de *Mecalux*. Un cop sec vàrem fer quatre forats al tauló superior per passar quatre cargols i fixar-lo a l'estructura amb quatre volanderes i femelles.

Fig. 11 i 12. Procés de pintat del *Mecalux*

Després de finalitzar la part superior de la taula mòbil, vàrem continuar fent-ne la base. En aquell moment ens faltava pensar en un mètode per ancorar els dos motors a la base de la taula per posar-hi les rodes. Aleshores vam dissenyar un *bracket* amb *Tinkercad (un programari de modelatge en 3D).* El bracket és un suport per als motors que fa pressió als laterals per mantenir-

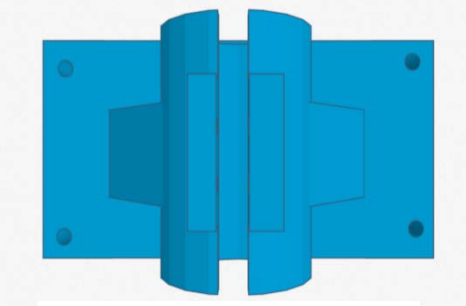

Fig. 13. Planta del *bracket*

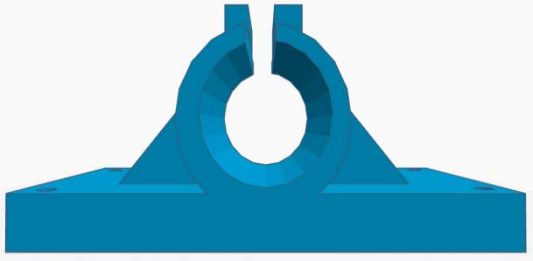

Fig. 14. Alçat frontal del *bracket*

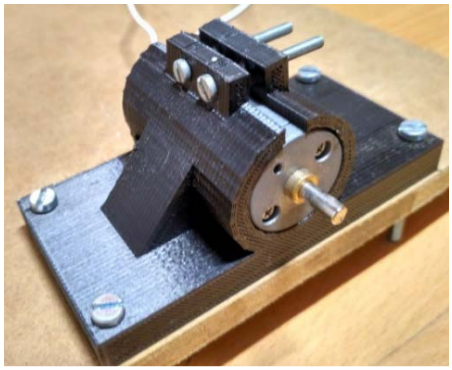

Fig. 15. Fotografia del bracket amb el motor

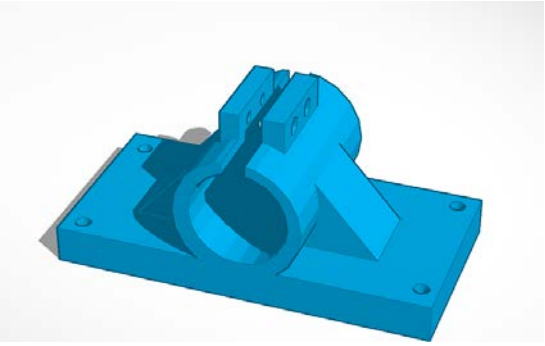

Fig. 16. Vista amb perspectiva del *bracket*

los a lloc gràcies als cargols que uneixen els dos costats del cilindre. A més, té quatre forats per poder fixar-lo a la base de la taula.

Un cop dissenyats els *brackets*, vam imprimir-los amb la impressora 3D i els vàrem muntar sobre la base amb els cargols. Aquests fixen l'eix dels motors a 10 centímetres de la part frontal del mòbil i estan alineats amb el límit del tauló. Vam agafar dues rodes de plàstic de cinc centímetres de diàmetre i les vam enganxar a l'eix del motor fent servir cola calenta.

La tercera roda, que té un coixinet en el punt d'ancoratge i que pot girar en qualsevol direcció, la vàrem unir a la base amb un cargol a la meitat de la taula (175 mil·límetres del costat) i a 50 mil·límetres de la cara anterior de la taula.

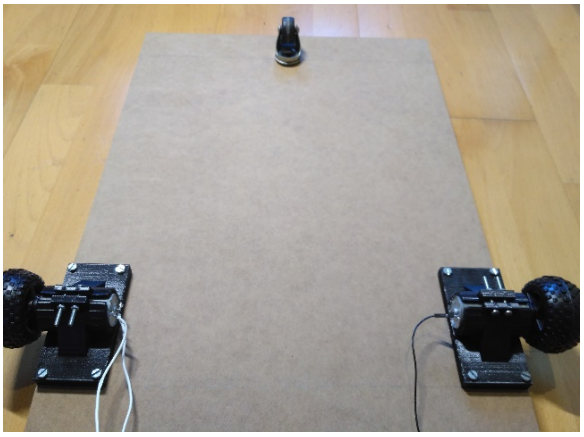

Fig. 17. Vista inferior de la base de la taula

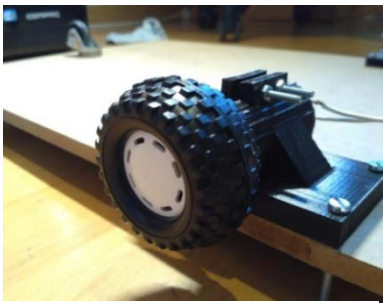

Fig. 18. Roda posterior

Fig. 19. Roda giratòria

Tot seguit, vàrem fer dos forats a l'estructura de *Mecalux* i vam unir-la a la base fent servir els cargols que mantenen el *bracket* ancorat. El resultat es el següent:

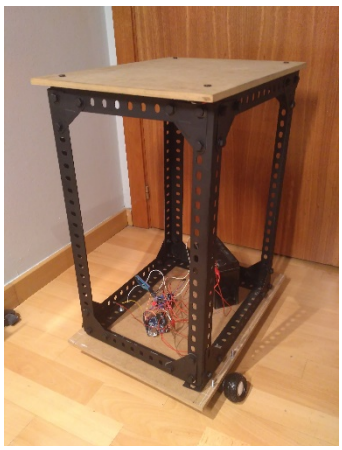

Fig. 21. Resultat del procés de construcció

Fig. 20. Realització forats a l'estructura de *Mecalux*

## 3.2 Moviment de la taula

Vàrem informar-nos sobre les possibles opcions que teníem per moure les rodes, em vam decantar pels motors de corrent continu a 12V, ja que són petits, capaços de fer moltes rotacions per minut i senzills de controlar amb l'Arduino i amb l'ajuda d'un driver.

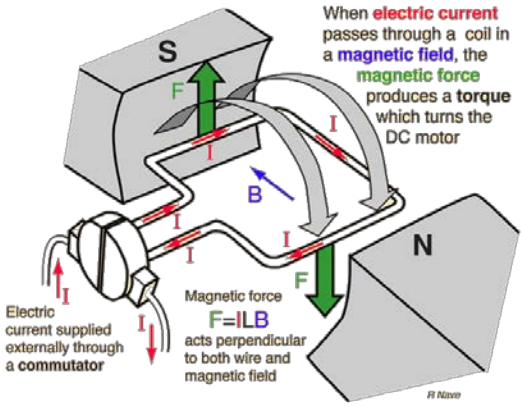


Fig. 21. Diagrama funcionament motor DC

Fig. 22. Motor

El funcionament d'aquests motors és el següent: passa corrent elèctric per la bobina que està situada entre dos imants els quals creen un camp magnètic. La força magnètica resultant de la circulació d'electrons crea un par motor que és el que fa rotar el motor.

Com que aquests motors tenen les propietats perfectes per als projectes en els quals es necessita velocitat de rotació i no un parell gran, cal acoblar al motor una caixa de canvis que aconsegueix bescanviar velocitat angular per parell motor. Actualment es venen motors que venen amb caixa de canvis incorporada, anomenats motors amb high torque que responen a la

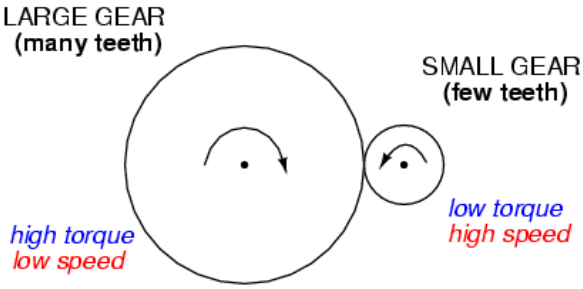


Fig. 23. Funcionament caixa de canvis

Fig. 24. Motor amb caixa de cavis (utilitzat en el projecte)

necessitat d'aquest projecte. Un cop considerades les opcions em vaig decidir per comprar dos motors de 12V a 100rpm que consumeixen entre 40 i 50mA sense càrrega.

Un cop escollits els motors, el següent pas era fer-los moure. Per fer això n'hi ha prou amb una placa de control *Arduino* UNO, els motors, un transistor i una font de corrent de 12V. Però aquest circuit només ens permetria decidir si el motor estava rotant o no, insuficient per aconseguir fer marxa enrere o girs fent anar les rodes en diferent direcció. El circuit que permet controlar la direcció del corrent d'electrons s'anomena pont en H i està format per quatre transistors que varien la direcció en què el corrent passa per la bobina i, per tant, fa girar el motor en diferents direccions.

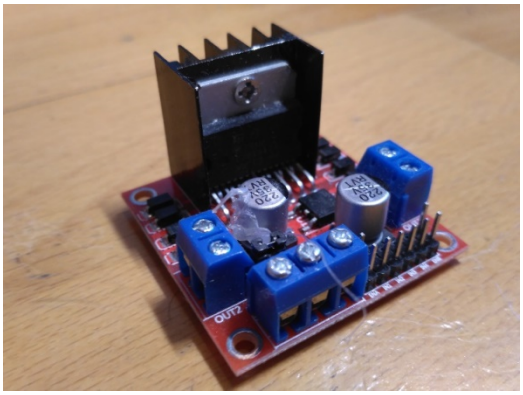

Fig. 25. Driver per a *Arduino*

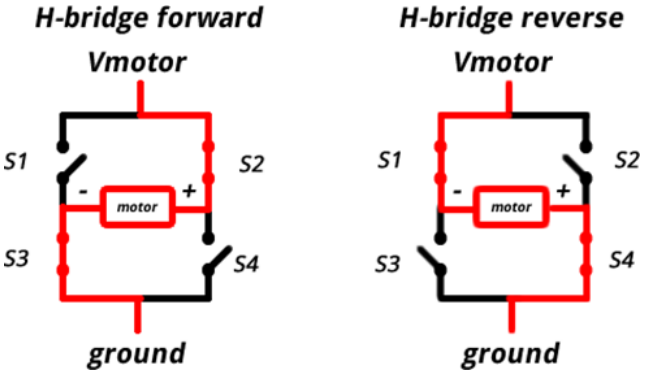


Fig. 26. Diagrama funcionament H-bridge

El pont va incorporat en un component que anomenen driver o controlador. Ens vàrem decantar pel L298N, un driver que a part de tenir els dos ponts en h també permet controlar la velocitat dels motors variant el voltatge que arriba als motors respecte a un valor que s'indica des d'un dels pins amb corrent regulable de l'Arduino (de 0 a 255). A més, aquest driver és molt adient perquè a més de les funcionalitats anteriors també té un pin de 5V de sortida que és perfecte per donar corrent a l'Arduino sense haver de tenir un circuit específic que variï el voltatge de 12V a 5V.

L'estructura del circuit per al control dels motors queda així:

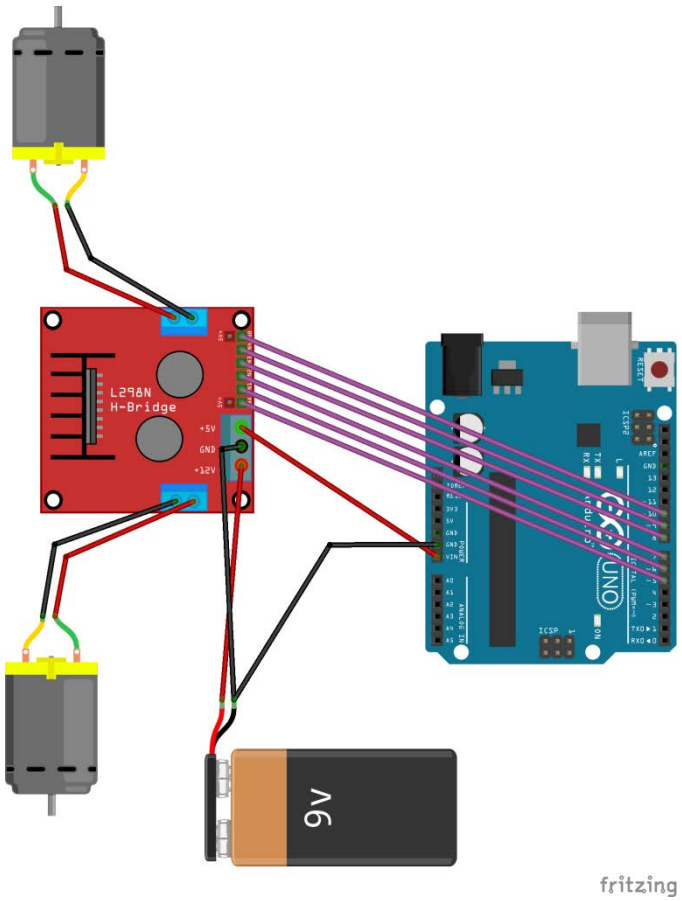


Fig 27. Diagrama del circuit dels motors

De la bateria de 12V,en aquest cas utilitzem una bateria de trepant, surten tres cables, un que va fins a l'entrada de 12V al *driver*, i dos que van al *ground* de l'*Arduino* i al del *driver*.

Del controlador surten 4 cables cap als motors, dos per a cada motor. L'ordre dels cables és indiferent, ja que la direcció de gir la podem canviar des de l'*Arduino*. Del *driver* surt un altre cable de la sortida de 5V fins al pin *In* de l'*Arduino* que serveix per donar corrent a l'*Arduino*. Per últim, de l'Arduino surten 6 cables que són els que comuniquen al *driver* què ha de fer. Dels pins D5 i D10 van respectivament a ENA i ENB. Aquests són els pins amb els quals direm al *driver* (amb un valor de rang entre 0 i 255) el voltatge que ha de proporcionar als motors i, en conseqüència, la velocitat. D'altra banda, els pins D6, D7, D8 i D9 estan connectats respectivament a IN1, IN2, IN3 i IN4. Amb aquests cables podem variar les direccions de cada motor accionant els transistors que el formen.

Després d'assegurar-nos que les connexions eren correctes, vàrem fer un programa per a moure els motors:

En primer lloc, vam declarar 6 variables amb un valor corresponent al seu pin de l'*Arduino* amb què es controla el *driver*, dos per a la velocitat (ENA i ENB) i quatre per a la direcció (IN1, IN2, IN3 i IN4).

Un cop tots els pins declarats i dins de la funció *set-up* vam declarar els sis pins com a sortida de dades (output).

```
//motor one
int enA = 5;
int in1 = 6;
int in2 = 7;
// motor two
int enB = 10;
int in3 = 8;
int in4 = 9;

 void setup() {
// put your setup code here, to run once:
pinMode(enA, OUTPUT);
pinMode(enB, OUTPUT);
pinMode(in1, OUTPUT);
pinMode(in2, OUTPUT);
pinMode(in3, OUTPUT);
pinMode(in4, OUTPUT);
```

Fet això ja podem controlar els motors únicament enviant senyals als pins declarats anteriorment. En primer lloc, cal especificar la direcció dels motors que es defineix seguint la següent taula:

| ***In1*** | ***In2*** | ***In3*** | ***In4*** | ***Direcció Motor L*** | ***Direcció Motor R*** | ***Direcció Taula*** |
|---|---|---|---|---|---|---|
| *High* | *Low* | *High* | *Low* | *Endavant* | *Endavant* | *Endavant* |
| *Low* | *High* | *Low* | *High* | *Enrere* | *Enrere* | *Enrere* |
| *High* | *Low* | *Low* | *High* | *Endavant* | *Enrere* | *Dreta* |
| *Low* | *High* | *High* | *Low* | *Enrere* | *Endavant* | *Esquerra* |
| Cal destacar que el tipus de gir realitzat fent anar un motor endavant i l'altre enrere és sobre el propi eix de l'aparell i només es realitzarà en el cas que l'usuari mogui el controlador digital a un extrem de la zona útil, com ja explicarem més endavant. | | | | | | |

Per exemple,

```
digitalWrite(in1, HIGH);
digitalWrite(in2, LOW);
digitalWrite(in3, HIGH);
digitalWrite(in4,LOW);
```

En aquest cas seguint la taula anterior podem saber que els dos motors giraran endavant. En segon lloc, cal especificar amb un valor d'entre zero i dos-cents cinquanta-cinc el voltatge que s'envia als motors, per exemple:

```
analogWrite(enA, 250);
analogWrite(enB, 250);
```

En aquest cas els dos motors rebran dos-cents sobre dos-cents cinquanta-cinc del voltatge total.

$$\frac{200}{255} = 0{,}78$$

$$12V \cdot \frac{200}{255} = 9{,}41V$$

Per últim, cal afegir al codi una espera de tres segons i després definir la velocitat dels dos motors com a zero perquè s'aturin.

```
delay(3000);

digitalWrite(in1, HIGH);
digitalWrite(in2, LOW);
analogWrite(enA, 0);
digitalWrite(in3, LOW);
digitalWrite(in4, HIGH);
analogWrite(enB, 0);
}
```

Per tant, en penjar a l'*Arduino* aquest codi, el dispositiu farà anar les dues rodes endavant a una velocitat de 200 respecte a 255, essent 255 la velocitat a 12v, esperarà tres segons i després aturarà els motors.

## 3.3 Mòdul de comunicació

El primer que vàrem fer rere tenir preparada la taula mòbil, però sense la possibilitat de controlar-lo a distància va ser informar-nos sobre les diferents tecnologies de transmissió de dades per decidir quina era la que més s'adaptava a les nostres necessitats. Els dos principals candidats eren *Bluetooth* i *Wi-Fi*, però vam preferit utilitzar *Bluetooth* perquè la informació que el mòbil envia és molt poca. Tot i que la distància màxima entre els dos dispositius és de 10 metres com a màxim i això podria semblar un problema, no ho és. Els recorreguts més freqüents per un dispositiu d'aquest tipus com, per exemple, el menjador - cuina no són de més de deu metres en la majoria d'habitatges. Un cop escollida la tecnologia, havia d'escollir el mòdul, ja que a diferència dels telèfons intel·ligents l'*Arduino* no té *Bluetooth* de fàbrica, sinó que s'ha de fer servir un mòdul extern.

En aquest cas vam escollir la família de mòduls *Bluetooth* HC, ja que són petits, eficients, no gaire complexos i barats. Dins d'aquest tipus vaig escollir el HC-05 que, a diferència del HC-06, és més configurable. Aquests mòduls intercanvien dades a distàncies curtes i operen a 2.4GHz.

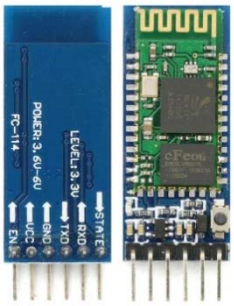

Fig. 28. Mòdul Bluetooth HC-05

També cal comentar que aquests mòduls tenen dos modes d'operació, el *Command mode*, on mitjançant ordres AT es pot configurar el dispositiu, i el "Data mode", on el mòdul intercanvia dades amb un altre dispositiu *Bluetooth*.

Un cop vàrem adquirir el mòdul vam descarregar la documentació del fabricant i vaig fer les connexions necessàries entre els dos dispositius: Del "pin" RX al digital 4, del pin TX al digital 2, de GND a GND i de 5V o VCC a 5V de l'*Arduino*.

Un cop connectat vaig programar un senzill codi per poder enviar i rebre informació al mòdul des de l'ordinador i, per tant, poder fer proves còmodament amb l'aplicació mòbil. A la primera línia, incloc la llibreria *Software Serial* per crear un nou serial fent servir uns altres pins en comptes del 0 i 1, ja que aquests són els que fa servir l'ordinador per comunicar-se amb l'*Arduino* i si els utilitzes per transferir dades de dos dispositius diferents

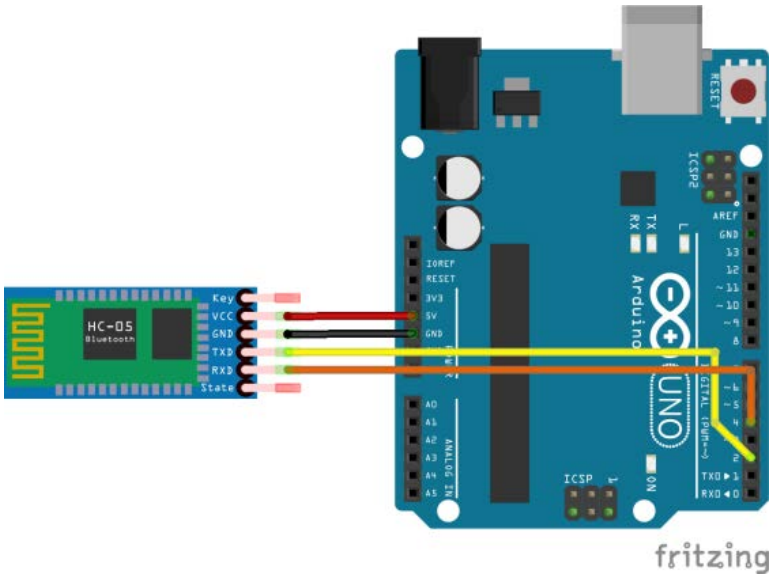


Fig. 29. Diagrama de connexions amb el mòdul *Bluetooth*

resulta que no funciona ni el mòdul ni l'ordinador. A la segona línia declaro un nou *serial* amb els pins 4 i 2.

```
#include <SoftwareSerial.h>
SoftwareSerial BT(4, 2 ); // RX, TX
```

En el *set-up* declaro la velocitat de transmissió de dades dels dos serials, i faig que s'escrigui *Enter AT commands*. La velocitat de transmissió és un paràmetre que es mesura en bauds (número de caràcters per segon a la transmissió).

```
void setup() {
  BT.begin(9600);
  Serial.begin(9600)
  Serial.println("Preparat"); }
```

El *loop* principal simplement llegeix les dades del serial BT, les escriu al serial original i a la inversa. D'aquesta manera, puc rebre i enviar dades de

l'ordinador a l'*Arduino* sense interferir en la comunicació entre aquest i el mòdul *Bluetooth*.

```
void loop() {
  if (BT.available()){
   Serial.write(BT.read()); }
  if ( Serial.available()){
   BT.write(Serial.read()); }}
```

Un cop pujat el programa a la placa vaig engegar el mòdul en mode Command mode, ja que era el primer cop que es fa servir i s'ha de configurar com el necessitava. Per fer això, simplement cal mantenir el botó del mòdul premut abans d'engegar-lo i deixar-lo anar uns segons després que la llum vermella s'il·lumini. Un cop amb el programa correcte i dins del command mode fent servir la taula de AT commands vaig definir el nom a HC-05, el rol a esclau, i la velocitat de transmissió a 9600 bauds.

**Basic AT commands**

| Command | Return | Parameter | Description |
|---|---|---|---|
| AT | OK | None | Test |
| AT+VERSION? | +VERSION:<Param> OK | Param: Version number | Get the soft version |
| AT+ORGL | OK | None | Restore default status |
| AT+ADDR? | +ADDR: <Param> OK | Param: Bluetooth address | Get module Bluetooth address |
| AT+NAME=<Param> | OK | Param: Bluetooth device name | Set device's name |
| AT+NAME? | +NAME:<Param> OK | Param: Bluetooth device name | Inquire device's name |
| AT+ROLE=<Param> | OK | Param:0=Slave role; 1=Master role; 2=Slave-Loop role | Set module role |
| AT+ ROLE? | + ROLE:<Param> | Param:0=Slave role; 1=Master role; 2=Slave-Loop role | Inquire module role |
| AT+UART=<Param>,<Param2>,<Param3> | OK | Param1: baud rate ( bits/s); Param2: stop bit; Param3: parity bit | Set serial parameter |
| AT+ UART? | +UART=<Param>,<Param2>,<Param3> OK | Param1: baud rate ( bits/s); Param2: stop bit; Param3: parity bit | Inquire serial parameter |

Fig 30. Taula amb ordres AT

Un cop feta la configuració i després de reiniciar el mòdul fent servir una aplicació anomenada HC05 *Bluetooth* terminal, vaig provar que tot funcionés correctament enviant un missatge de prova des del mòbil a l'*Arduino* i al revés.

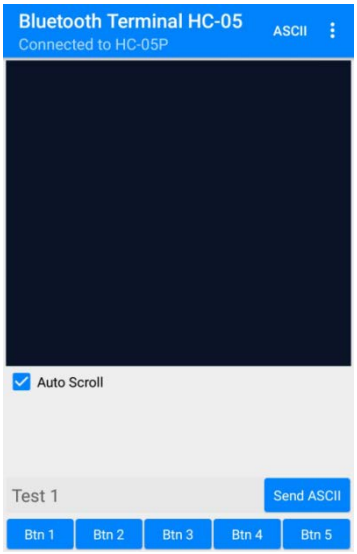


Fig. 32. Aplicació móvil des d'on s'envia el

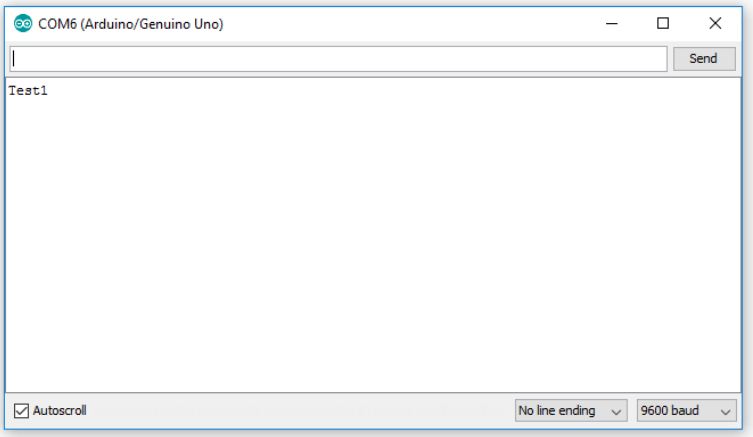


Fig. 31. Serial de l'Arduino on es rep el

Un cop els motors i el *Bluetooth* estaven preparats em vaig posar a buscar una aplicació que em permetés controlar els motors de manera intuïtiva. Per fer aquesta funció vaig escollir *Joystick Bluetooth Commander*, desenvolupada per *kas_dev* i que es pot descarregar gratuïtament des de *Google Play*.

Aquesta aplicació conté una palanca de control virtual (*Joystick*) molt simple a la part dreta de la pantalla, formada bàsicament per un cercle petit dins d'un cercle més gran pel qual es pot desplaçar.

La posició del centre del cercle petit respecte al del gran són dues dades en forma de coordenades que l'aplicació mostra a la cantonada inferior esquerra, aquests dos valors van de zero a cent i permet enviar-les per *Bluetooth* a un altre dispositiu. Aquestes són les dues dades que l'*Arduino* després processarà per calcular la velocitat i direcció dels motors.

D'altra banda, l'aplicació també conté uns botons a la part esquerra de la pantalla configurables per enviar un missatge a mesura que han estat premuts o deixats anar. Aquests botons es fan servir per configurar alguns paràmetre al nostre gust: la velocitat mínima i màxima, la velocitat de gir...

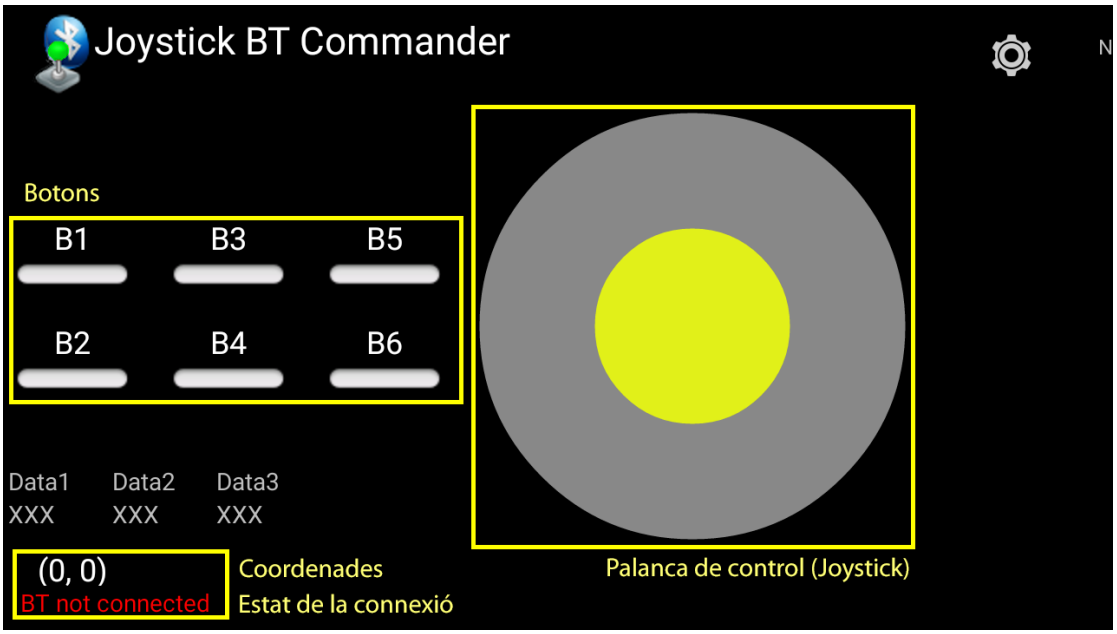


Fig. 33. Aplicació *Joystick Bluetooth Commander*

Tot i que la configuració és bastant bàsica hi ha alguns paràmetres que és millor canviar-los per aconseguir una millor resposta de la taula. El primer és canviar el *Joystick constraint* de *circle* a *box*. Això el que fa és canviar la forma de la zona límit per on el cercle petit es mou; de cercle a quadrat, d'aquesta manera serà més fàcil calcular la velocitat dels motors.

### 3.4 Codi i circuit final

Un cop escollida l'aplicació vàrem unir totes les parts de codi.

La idea general de què fa el codi és la següent: En primer lloc, el telèfon intel·ligent envia per *Bluetooth* al mòdul la posició del centre del cercle petit respecte al del gran. Un cop al mòdul, transmet les dades a l'*Arduino* que les guarda en forma de variable com a *joyY* (posicó sobre l'eix vertical) i *joyX* (posició sobre l'eix horitzontal). Un cop les dades estan a l'*Arduino* aquest les fa passar per diferents condicionals on fa diferents càlculs i successivament n'extreuen la velocitat i l'ordena al *driver* .

En primer lloc, incloem la llibreria *software Serial*, i definim com a variable de nom *STX* (*Start of text*) 0x02 i com a *ETX* (*End of text*) 0x03. Aquestes dues variables són caràcters de control i ens serveixen per a marcar l'inici i el final d'una transmissió de dades. Després definim com a SLOW 750ms i com a FAST 250, aquests són els intervals entre els quals farem que el programa oscil·li per variar la quantitat de dades per segon que es transmeten.

```
#include "SoftwareSerial.h"
#define   STX       0x02
#define   ETX       0x03
#define   ledPin    13
#define   SLOW       750            // (ms)
#define   FAST       250
```

Vam declarar com a pins del serial virtual el dos i el tres i vàrem crear diverses variables: una per emmagatzemar els bytes que rep l'*Arduino*, una altra per emmagatzemar el primer byte que s'envia al mòbil, una altra per guardar els anteriors, una altra per a definir l'interval en què s'envien dades i una altra per guardar l'estat de la connexió.

```
SoftwareSerial mySerial(2,3);                    // pin 2=TX pin 3=RX
byte cmd[8] = {0, 0, 0, 0, 0, 0, 0, 0};          // bytes rebuts
byte buttonStatus = 0;                           // primer byte cap al Mòbil
long previousMillis = 0;              // emagatzema button status
long sendInterval = SLOW;      // interval entre transmissió Buttons status
String displayStatus = "xxxx";
```

Aquest tros de codi és idèntic al que he utilitzat en el programa per provar els motors, on es declaren els pins als quals es connectarà el *driver*. Té quatre variables més per més tard poder fer càlculs necessaris per dictar la velocitat a partir del controlador del mòbil.

```
//motor one
int enA = 5;
int in1 = 6;
int in2 = 7;
// motor two
int enB = 10;
int in3 = 8;
int in4 = 9;
int speedA = 0;
int speedB = 0;
int speedA2 = 0;
int speedB2 = 0;
```

Dins la funció set-up es declaren els dos serials a 9600 bauds, i tots els pins que s'estan fent servir com a sortida (*output*). D'altra banda, hi ha un *while* que es manté llegint el *serial* virtual sempre que aquest estigui disponible.

```
void setup()  {
 Serial.begin(9600);
 mySerial.begin(9600);
 pinMode(ledPin, OUTPUT);
 pinMode(enA, OUTPUT);
 pinMode(enB, OUTPUT);
 pinMode(in1, OUTPUT);
 pinMode(in2, OUTPUT);
 pinMode(in3, OUTPUT);
 pinMode(in4, OUTPUT);
 while(mySerial.available())  mySerial.read();
}
```

La funció *loop* es dedica a llegir tota l'estona la informació del *serial virtual* i a separar els missatges d'entre ells buscant els principis de text (*STX*) i els finals de text (*ETX*). Si els troba, els emmagatzema a la variable cmd en format ASCII i executa la funció *Joystickxy.*

```
void loop() {
 if(mySerial.available())  {
   delay(2);
   cmd[0] =  mySerial.read();
   if(cmd[0] == STX)  {   //busca inici
     int i=1;
     while(mySerial.available())  {       //mentre virtual dispnible
       delay(1);
       cmd[i] = mySerial.read();
       if(cmd[i]>127 || i>7)                 break;     // error de comunicació

     if(i==7)          Joystickxy(cmd);     // cada 6 bytes
   } }}
```

La funció Joystickxy s'encarrega de transformar la representació en ASCII de les coordenades que rep l'*Arduino* via *Bluetooth* en un valor amb el qual l'*Arduino* pugui fer operacions (*Integer*).

```
void Joystickxy(byte data[8])    {
 int joyX = (data[1]-48)*100 + (data[2]-48)*10 + (data[3]-48);
 int joyY = (data[4]-48)*100 + (data[5]-48)*10 + (data[6]-48);
 joyX = joyX - 200;
 joyY = joyY - 200;
 if(joyX<-100 || joyX>100 || joyY<-100 || joyY>100)     return;
```

A partir d'aquí el codi es dedica a interpretar les dades rebudes per a prendre decisions en funció a això. En el primer condicional la y que és igual a zero desactiva els dos motors, això passa quan l'usuari deixa anar el joystick.

```
 if (joyY == 0){
   digitalWrite(in1, LOW);
   digitalWrite(in2, LOW);
   digitalWrite(in3, LOW);
   digitalWrite(in4, LOW);  }
```

El segon condicional defineix la direcció dels motors cap endavant si *joyY* és superior a 0 i a més defineix la velocitat del motor A (*speedA*) com a *joyY* x 2.5, ja que l'alçada màxima del joystick és de 100 i el valor màxim a enviar és de 255. D'altra banda, defineix la velocitat del motor B com a (*joyY**2.55)*(1 - *joyX**0.01) perquè en la mesura en què *joyX* augmenta*, el valor del* parèntesi dret queda més proper a zero, i això produeix que, en fer la multiplicació dels dos parèntesis, el valor de *speedB* (velocitat motor B) sigui més baix.

```
if (joyY > 0){
   digitalWrite(in1, LOW);
   digitalWrite(in2, HIGH);
   digitalWrite(in3, HIGH);
   digitalWrite(in4, LOW);
```

```
  if (joyX > 0){
   speedA = joyY*2.5;
   speedB = (joyY*2.5)*(1 - joyX*0.01);
  }
  if (joyX < 0){
   speedA = (joyY*2.5)*(1 - (joyX*0.01*-1));
   speedB = joyY*2.5; }
```

D'altra banda, si *joyY* és més gran que zero però *joyX* és menor que zero passa el mateix que en el cas anterior però a la inversa; *SpeedA* es redueix, en la mesura que *joyX* augmenta. Aquest augment és relatiu, ja que en realitat estem treballant amb coordenades a la X negativa, per això la multiplicació per -1 dins el parèntesi dret.

```
if (joyY<0){
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
if (joyX > 0){
   speedA = joyY*2.5*-1;
   speedB = (joyY*2.5*-1)*(1 - joyX*0.01);
  }
   if (joyX < 0){
   speedA = (joyY*2.5*-1)*(1 - (joyX*0.01*-1));
   speedB = joyY*2.5*-1;
}}
```

L'única diferència entre la marxa endavant i la marxa enrere és el senyal que dono als transistors que controlen la direcció dels electrons a la bobina. Així que canviant només el senyal que es dona als transistors les rodes actuen de la mateixa manera però en una altra direcció.

Per últim, al final del *loop* hi ha el següent codi que senzillament diu la velocitat als motors i escriu en el serial les variables per saber en tot moment què està fent el circuit

```
analogWrite(enA, speedA);
analogWrite(enB, speedB);
Serial.print("Joystick pos:  ");
Serial.print(joyX);
Serial.print(", ");
Serial.println(joyY);
Serial.println("Speed A: ");
Serial.println(speedA);
Serial.println("Speed B: ");
Serial.println(speedB
```

Així, doncs, el funcionament de l'aparell complet és el següent:

1. A l'aplicació l'usuari defineix cap a on vol que es mogui la taula fent ús del controlador virtual.

2. L'aplicació envia per *Bluetooth* dues coordenades, de -100 a 100 cadascuna, que són la posició del dit dins del *joystick*.
3. L'*Arduino* rep les dades i les transforma en un tipus de dades amb les quals es pugui operar.

4. Darrerament el microcontrolador passa les dades pels condicionals, i fa les operacions corresponents a cada cas d'on n'extreu la velocitat en forma d'un valor d'entre 0 i 255 per a cada motor.

Posant per exemple que les dues dades rebudes són JoyX=50 i JoyY=90, com que la velocitat del motor esquerra (speedA) ve donada per joyY*2.5 seria 225, i com que la velocitat del motor dret (speedB) ve donada per (joyY*2.5)*(1 - joyX*0.01) seria 112.5. Per tant a l'anar el motor esquerra més ràpid que el dret la taula giraria cap  la dreta
5. Per últim l'*Arduino* dicta al *driver* aquests valors que successivament són convertits en voltatge a proporcionar als motors.

El circuit complet és la unió del circuit dels motors amb el circuit del *Bluetooth*. La bateria utilitzada és la mateixa que anteriorment, la d'un trepant.

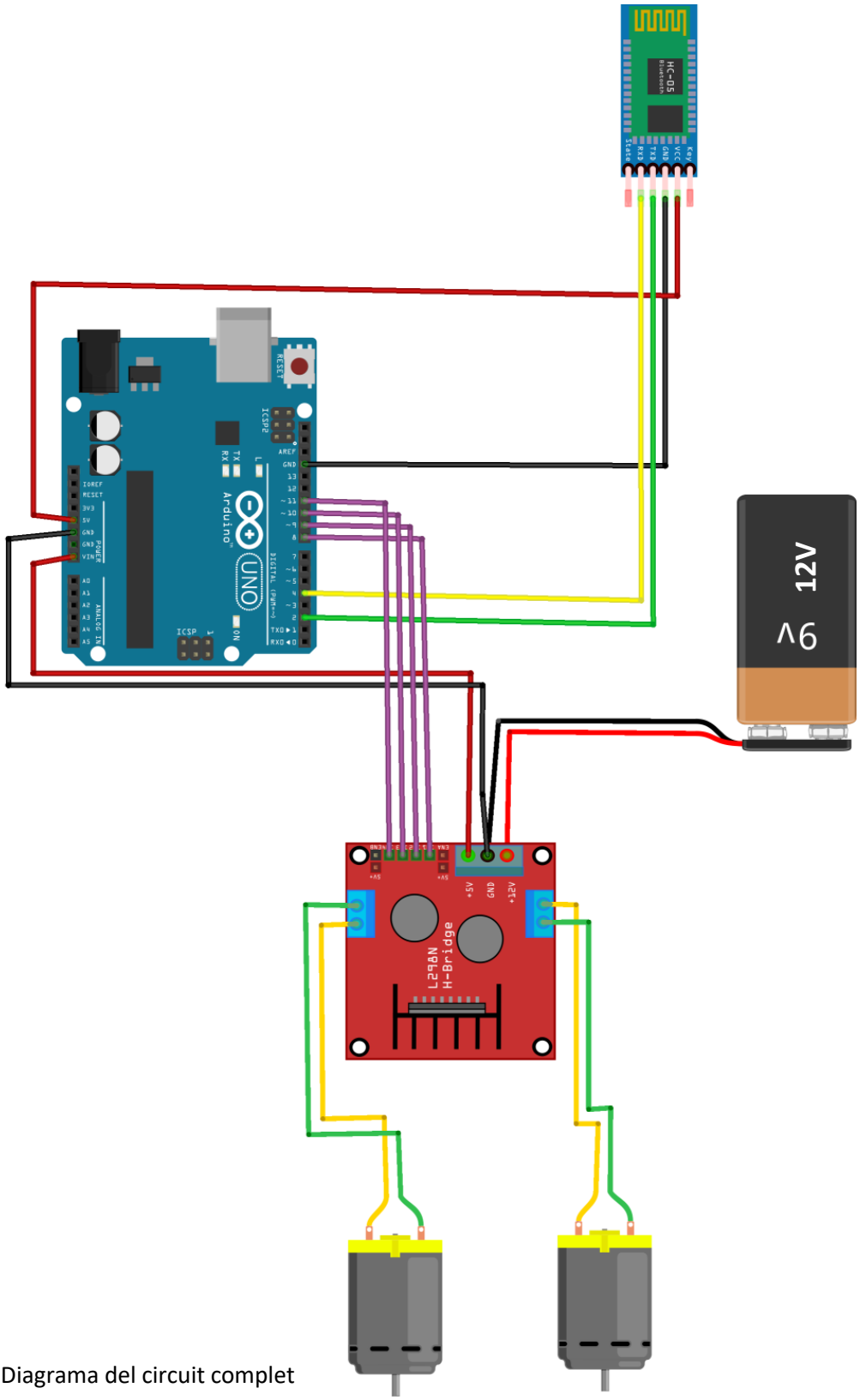


Fig 34. Diagrama del circuit complet

# Conclusions

En la realització d'aquest treball s'ha contribuït a la disminució d'una dificultat en l'àmbit de les persones amb handicaps motrius, en concret en el transport d'objectes a la casa.

En començar el projecte, partíem de coneixements bàsics d'*Arduino* i el coneixement d'una dificultat amb què es trobaven les persones amb mobilitat reduïda. Un cop acabat el projecte, hem arribat a aproximar a aquest col·lectiu la possibilitat de transportar utensilis a la llar amb més facilitat i de manera més segura. D'altra banda, també s'ha après en l'àmbit de la programació, la robòtica i la tecnologia, claus per a aconseguir l'autonomia de l'aparell construït.

S'ha arribat a construir un dispositiu amb el qual ens podem fer una clara idea de la gran utilitat de l'aparell per a les persones amb mobilitat reduïda. Tot i això, la taula mòbil s'allunya de poder ser emprada a les cases amb la fidelitat i la seguretat adients.

Durant la realització d'aquest treball ens hem trobat amb diferents obstacles majoritàriament en el procés de construcció del circuit. Els obstacles han estat en gran part resolts, excepte en el cas del mòdul *Bluetooth* que a vegades falla, suposem que per una irregularitat en l'amperatge o un problema de fàbrica.

En l'àmbit de la construcció de l'estructura, però, hi ha diferents aspectes que no s'han pogut millorar com, per exemple, l'excessiu pes de l'estructura (és un objecte un pèl pesant i poc manejable); la manca d'estil de l'aparell en general (faltaria l'assessorament d'un dissenyador de mobiliari d'interiors), la no cobertura del circuit elèctric, cosa que pot comportar problemes de seguretat, la poca fiabilitat de les rodes(s'agafen poc al terra i són molt petites) i la bateria és molt pesant.

No hem pogut resoldre els problemes per una limitació de temps, principalment. El pressupost utilitzat, però, ha estat molt adient en tot moment per mantenir el robot econòmicament assequible a l'usuari.

Aquest treball es podria prosseguir per dos camins. El primer, és en l'àmbit de la comercialització, de la publicitat i màrqueting del producte realitzat. El

segon, és centrar-se en l'eficiència de l'aparell en l'ús diari, és a dir, que veritablement sigui útil calibrant el gir, la velocitat, fent-lo segur i autònom.

Com a consideració personal, aquest projecte ha estat una molt bona manera de practicar i ampliar els meus coneixements tant en l'àmbit de la robòtica com en el de la tecnologia.

# Bibliografia

Kas_dev. Joystick bluetooth Commander a Google Play [en línea].<https://play.google.com/store/apps/details?id=org.projectproto.btjoystick&hl=en>. [Consulta: 9 de Juny de 2017]

Kas_dev. Joystick bluetooth Commander a arduino.cc [en línea].< http://forum.arduino.cc/index.php?topic=173246.0>. [Consulta: 9 de Juny de 2017]

Arduino and Bluetooth [en línia.]<http://www.instructables.com/id/Arduino-AND-Bluetooth-HC-05-Connecting-easily/>[Consulta: 3 de Maig de 2017]

HC-05 Bluetooth for Arduino[en línia.]< http://howtomechatronics.com/tutorials/arduino/arduino-and-hc-05-bluetooth-module-tutorial/>[Consulta: 5 de Maig de 2017]

Módulo Bluetooth HC-05 [en línia.]< https://www.prometec.net/bt-hc05/>[Consulta: 1 de Maig de 2017]

BlueTooth HC05-HC06 Modules How To [en línia.]< https://arduino-info.wikispaces.com/BlueTooth-HC05-HC06-Modules-How-To>[Consulta: 1 de Maig de 2017]

HC-05 Bluetooth Module [en línia.]< https://www.gme.cz/data/attachments/dsh.772-148.1.pdf>[Consulta: 8 de Maig de 2017]

How to use the L298 Motor Driver [en línia.]< http://www.instructables.com/id/How-to-use-the-L298-Motor-Driver-Module-Arduino-Tu/>[Consulta: 10 de Juny de 2017]

Dc motor control tutorial with L298N h-bridge [en línia.]< http://howtomechatronics.com/tutorials/arduino/arduino-dc-motor-control-tutorial-l298n-pwm-h-bridge/ >[Consulta: 10 de Juny de 2017]

Fritzing(software suat per a fer els diagrames) [en línia< http://fritzing.org/home/>[Consulta: 14 de Setembre de 2017]

# Agraïments

Voldria agrair al meu tutor, Dani Urbano, la seva constància i dedicació a l'hora de conduir el meu treball i per haver-me esperonat durant tots aquests mesos. També m'agradaria agrair la col·laboració de la Sara Suàrez i el José Luis en el projecte , ja que, sense ells aquest no hagués estat possible dur-lo a terme.