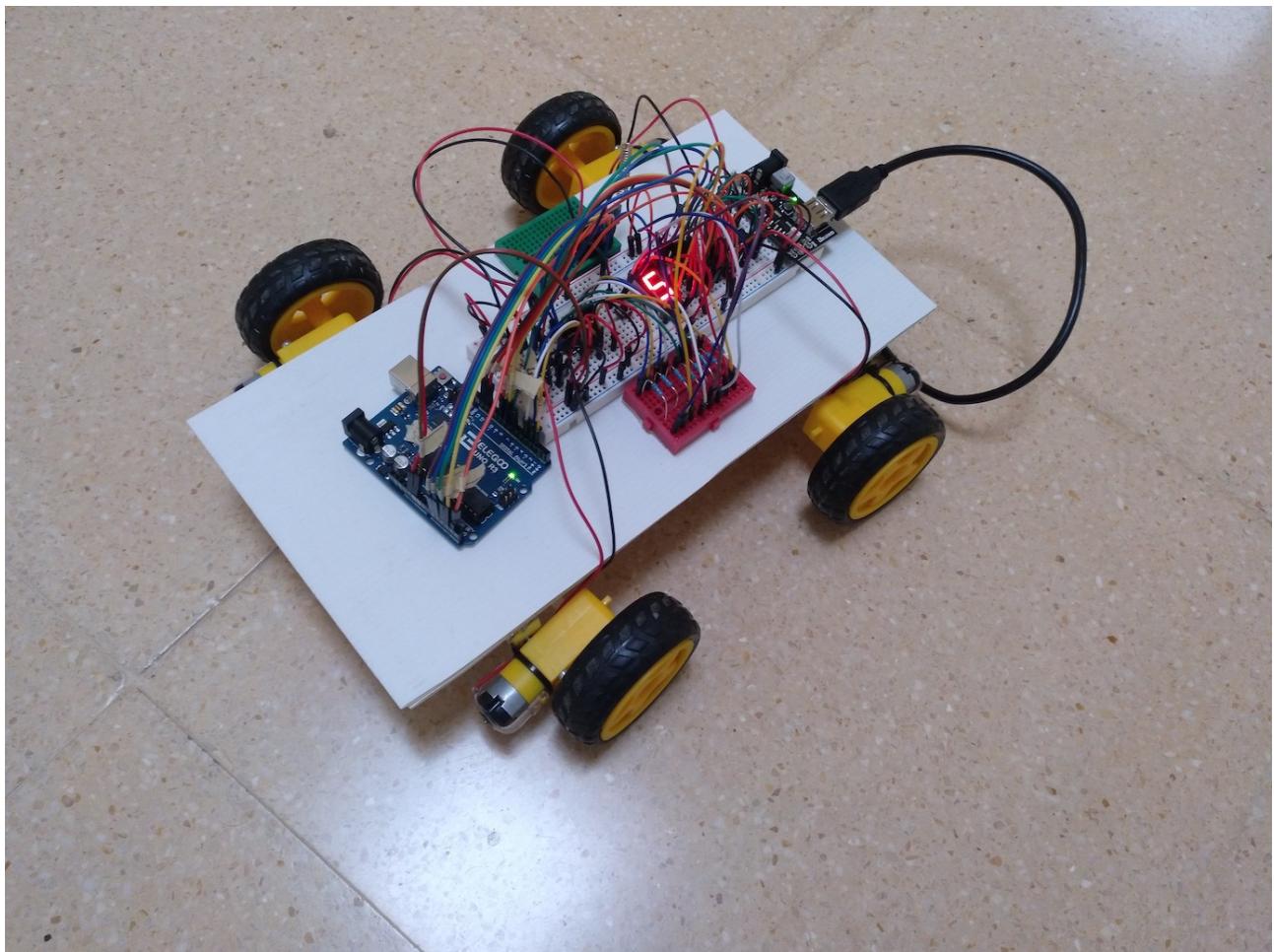


G1



**Jordi Borràs i Vivó
Curs Programació de plaques robòtiques
CIFO La Violeta
Novembre 2020**

0. Index

1. Introducció i objectius	p.3
2. Simulacions i primeres proves	p.4
3. Muntatge dels motors i xassís del vehicle	p.6
4. Diversos modes de conducció	p.9
5. Comandament per radiofreqüència	p.12
6. Millores i futures ampliacions	p. 20

1. Introducció i objectius

La present memòria recull la tasca del projecte de construcció d'un cotxe, en endavant G1, que he fabricat des de zero, tant el quant a la seva construcció física com a la programació necessària per al seu funcionament.

El projecte neix del curs de programació de plaques robòtiques desenvolupat al CIFO La Violeta en el mes de novembre de 2020.

El vehicle disposa de diferents "modes" de funcionament, i sensors que facilitaran la seva conducció, així com d'un sistema de control per radiofreqüència.

Els reptes que planteja el projecte són diversos i en destaco tres de principals: reducció de pins, problemes físics inherents a la construcció del cotxe i l'ús de nous sensors i actuadors no estudiats a classe.

Per un costat la necessitat de reduir el nombre de pins de la placa Arduino a usar. en el moment que incorpores diversos sensors i actuadors, cada un d'ells necessita l'ús de pins per enviar allò que mesura a la placa o rebre l'ordre que aquesta placa li envia. Això fa que de seguida s'ocupin tots els pins disponibles a la placa. En aquest sentit l'estalvi de pins és important.

Per a reduir l'ús de pins sobretot en els motors que mouran el cotxe, fem servir un xip desplaçador de registre 74H595 amb un sistema que detallo en l'apartat de simulacions, que permet fer anar els quatre motors de manera independent usant només quatre pins de la placa Arduino.

El segon repte ha estat el superar els desafiaments que suposava el disseny físic del cotxe, ja que s'ha hagut de treballar en reduir el pes que ha de moure el vehicle, en que les rodes estiguin el màxim d'equilibrades i paral·leles possibles.

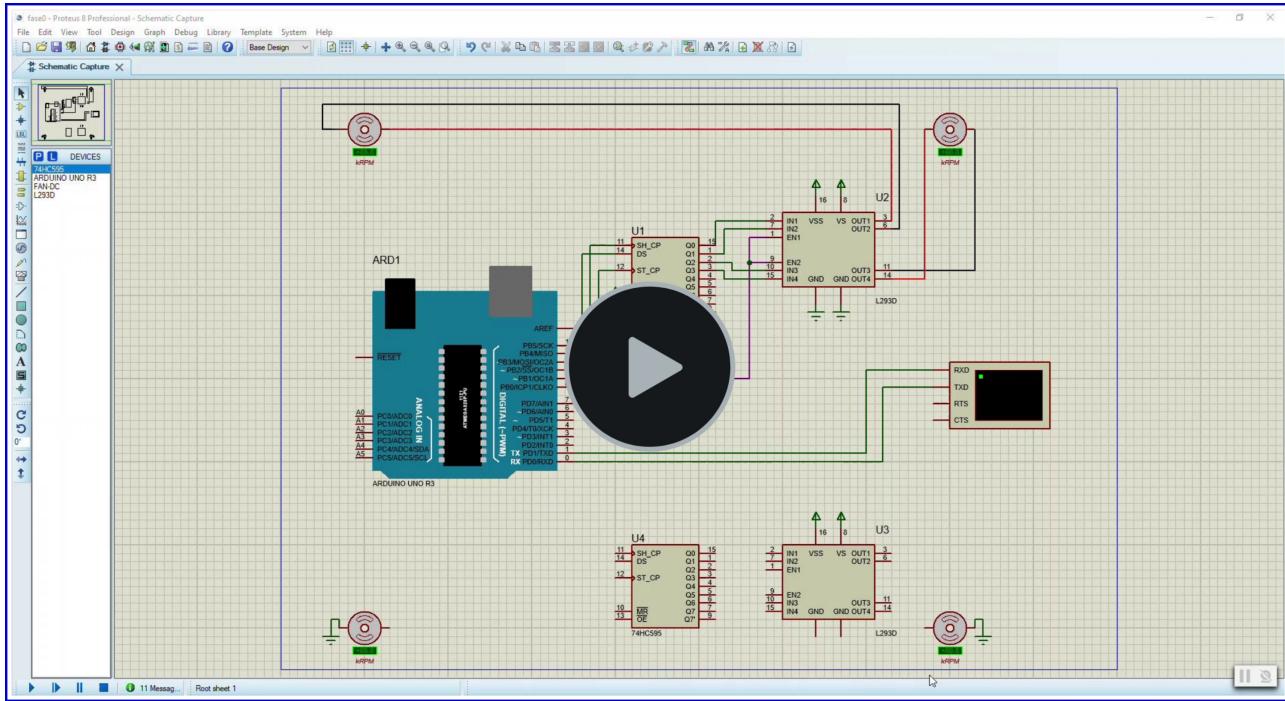
Finalment el tercer gran repte, ha estat la incorporació del modul de radiofreqüència, no treballat anteriorment i totalment desconegut, el que ha suposat un aprenentatge en quant al seu funcionament electrònic, com a la programació necessària per al seu correcte funcionament.

D'aquesta manera la següent memòria reflecteix de forma cronològica els procés de construcció del G1 i l'explicació de com s'ha anat superant les dificultats i reptes presentats durant el procés.

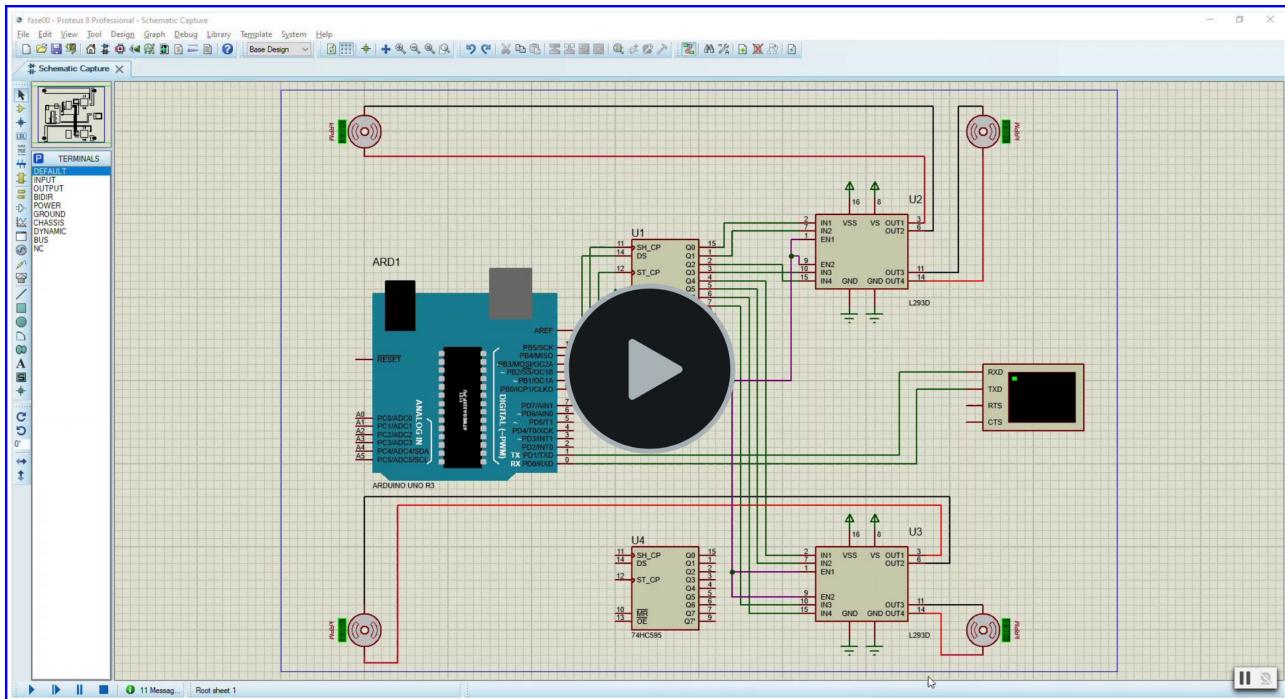
2. Simulacions i primeres línies de codi

Començo la simulació amb Proteus i Tinkercad, i les primeres línies de programació de la versió 0.

En aquesta primera simulació només funcionen els dos motors davanters:



Deixo la fase 0 en que li passo el moviment via variable i implemento la versió 00, en que li passaré per port sèrie el moviment que vull que faci el cotxe.



Prova a tinkercad: <https://www.tinkercad.com/things/9yvLgN1cFRG>

El codi usat en aquesta fase en destaquem el funcionament bàsic:

```
// variable que dirà en quina direcció aniran els motors  
int camin = 153;
```

aquest número és un valor que obtinc de passar a integer el valor byte que ens diu la direcció de cada un dels quatre motors seguint la següent taula:

Direcció	Motors davanters				Motors trasers				suma
	esquerra		dreta		esquerra		dreta		
	dir. a	dir. b	dir. a	dir. b	dir. a	dir. b	dir. a	dir. b	
endavant	1	0	1	0	1	0	1	0	170
gir dret	1	0	0	1	1	0	0	1	153
gir esq.	0	1	1	0	0	1	1	0	102
enrere	0	1	0	1	0	1	0	1	85
a sumar	128	64	32	16	8	4	2	1	

```
// funció que ordena el moviment dels motors usant un 74HC595  
  
void moume(byte data){  
    //Serial.println(data, BIN);  
    digitalWrite(actmt, HIGH);  
    digitalWrite(latch1, LOW);  
    for (int i = 0; i < 8; i++){  
        updateShiftRegister(data);  
    }  
    digitalWrite(latch1, HIGH);  
}
```

I en el loop l'executo mitjançant

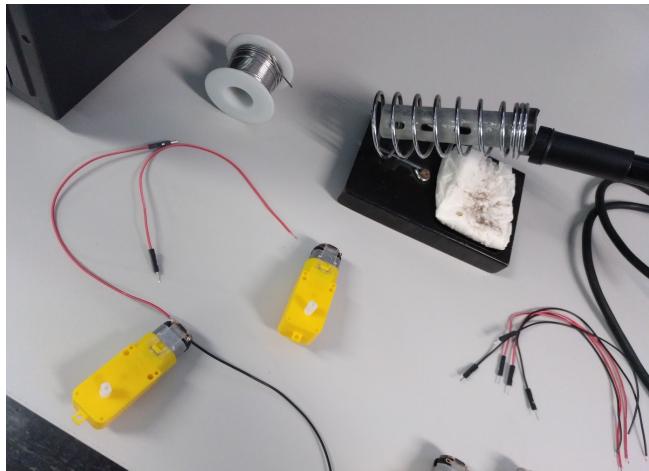
```
moume(camin);
```

Per a recollir el valor rebut per port sèrie substituïm l'ordre anterior pel següent codi:

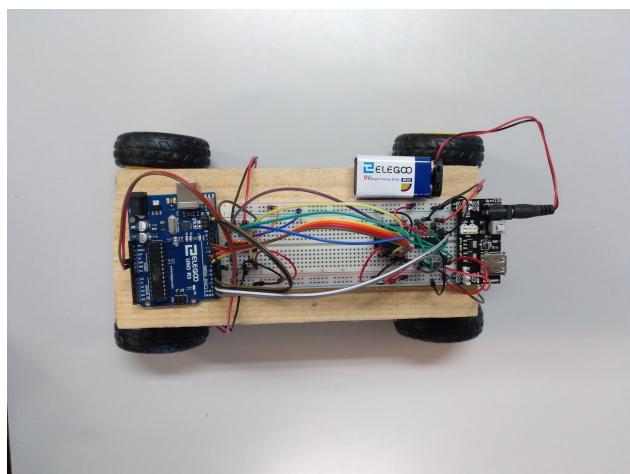
```
if (Serial.available() > 0){  
    data = Serial.parseInt();  
    Serial.println("rebut!");  
    Serial.print(data);  
    Serial.print(": ");  
    Serial.println(data, BIN);  
    if (data==0) {  
    } else {  
        moume(data);  
    }  
}
```

3. Muntatge dels motors i xassís del vehicle

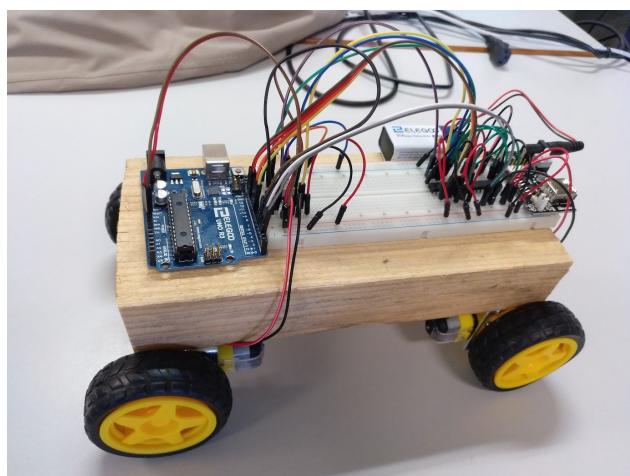
Un cop fets els test en el simulador inicio el muntatge físic del vehicle:



Soldadura dels cables dels motors



Vista superior del vehicle



Vista lateral del cotxe

Després de fer el muntatge físic del G1, el funcionament dels motors no és òptim. Sembla no haver-hi prou potència. Descarto fer sortir els quatre pins d'activació dels motors d'un sol pin d'Arduino (9) mitjançant un pont, i els faig sortir per quatre pins diferents (2 a 5).

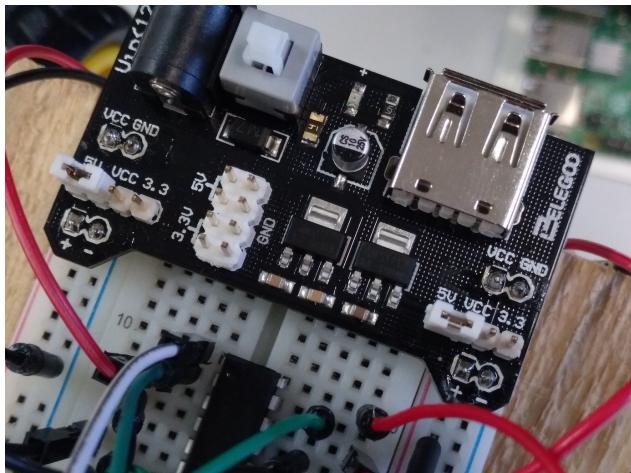
Amb la connexió a la xarxa elèctrica li costa molt d'avançar i hi ha una roda (darrera esquerra) que no roda bé, s'encalla.

Amb la pila de 9V no és capaç ni de tan sols moure una sola roda.

Mesurem el consum d'un sol motor amb un amperímetre. 100 miliampers, descartaré moure'l amb la pila i hi aportaré una bateria auxiliar de mòbil que té 4000mAh de capacitat i una sortida de 5V i 2A.

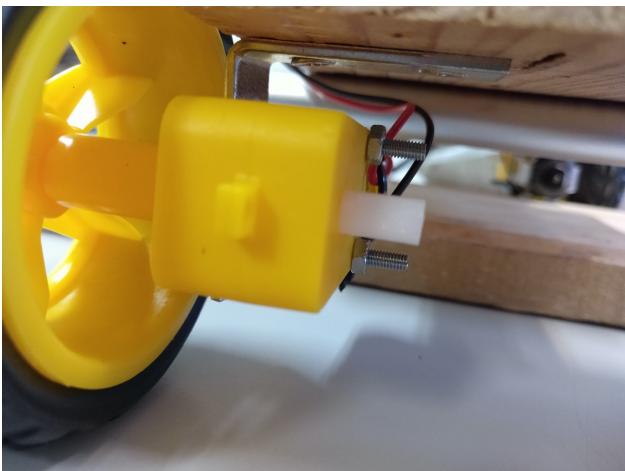
Tot i així el funcionament dels motors no és el desitjat i investigo més respecte aquests problemes amb els següents resultats:

Descobreixo que el selector de voltatge del costat dret de la placa d'alimentació auxiliar estava posat a 3.3V, de manera que part de la placa de proves (d'on s'alimentava entre altres l'Arduino) s'estava alimentant a 3.3V, el que donava problemes d'alimentació.

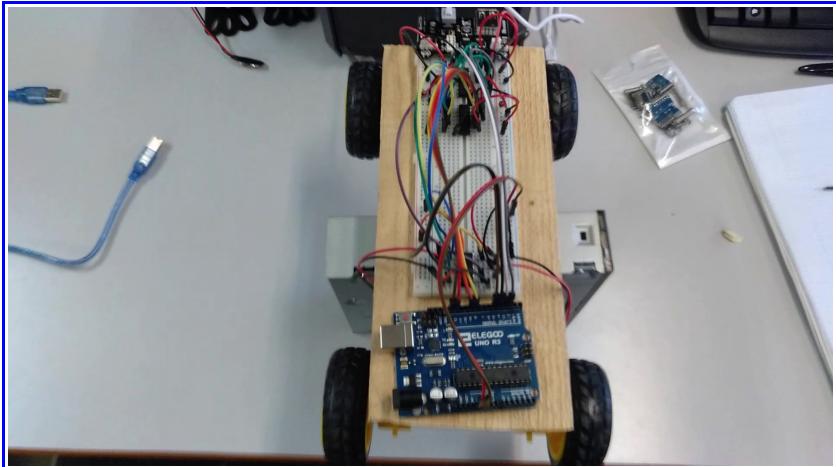


A l'esquerra imatge de la placa d'alimentació i a la dreta detall del selector que vaig haver de canviar.

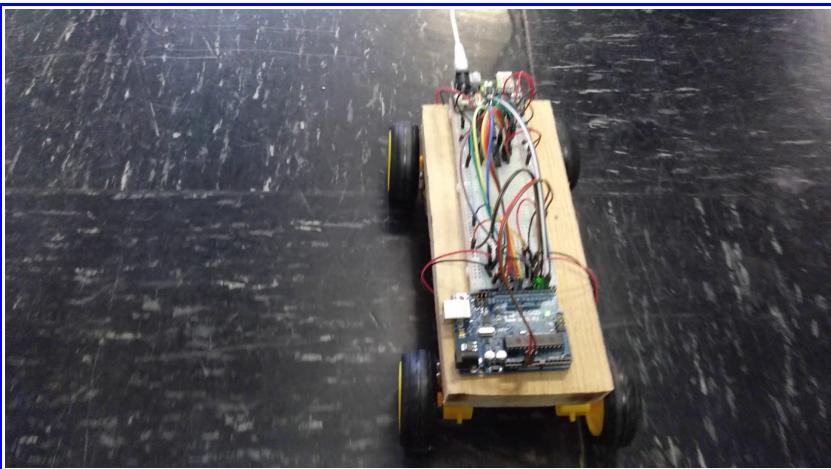
D'altra banda el motor del darrera esquerra que no girava prou bé, descobreixo que una excessiva pressió dels cargols feia que el frenés. Afluixó una mica els cargols i ara ja roda amb normalitat.



Detalls dels cargols que pressionaven massa el motor.



Vídeo del funcionament a la taula de proves (click per reproduir)



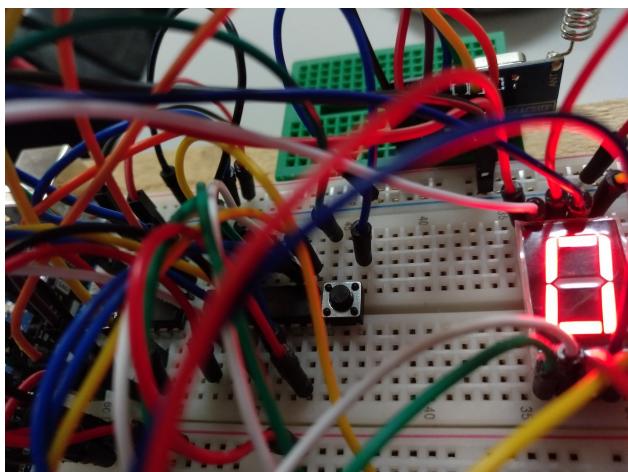
Vídeo del funcionament a terra (click per reproduir)

4. Diversos modes de conducció

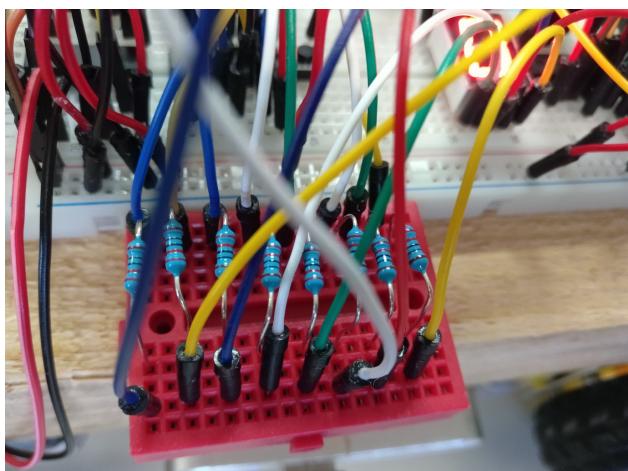
Incorporo la fase 1, en que incloç un botó que em permetrà modificar la direcció del cotxe i un display que em mostri quin valor de direcció està funcionant.

En aquest punt del projecte, el G1 tindrà diferents modes de conducció que seleccionaré prement el botó:

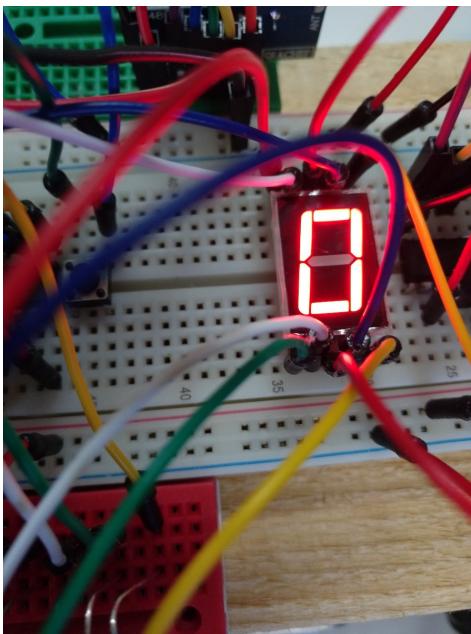
- Mode 0: el mode inicial quan encenc l'alimentació, el 0, el cotxe restarà parat
- Mode 1: el vehicle anirà endavant i els quatre motors avançaran endavant.
- Mode 2: el vehicle anirà enrere i els quatre motors avançaran enrere.
- Mode 3: el cotxe girarà a la dreta, els motors de l'esquerra avancen i els de la dreta retrocedeixen.
- Mode 4: el cotxe girarà a l'esquerra, els motors de l'esquerra retrocedeixen i els de la dreta avancen.
- Mode 5: el cotxe espera ordres a través del comandament de radiofreqüència.
- Mode 6: el cotxe fa allò que li diem a través del port sèrie.



Imatge del botó selector del mode de funcionament



Detall de la connexió del display, usant un 74HC595 i vuit sortides als vuit segments.



Detall del display que mostra el "mode de conducció"

En aquest punt incorporo un integer que emmagatzemara els modes de conducció

```
int conducc[10] = {
    0,      //mode 0 apagat
    170,   //mode 1 tot endavant
    85,    //mode 2 tot enrera
    153,   //mode 3 gira dreta
    102,   //mode 4 gira esquerra
    0,     //mode 5 incorporarà radiofreqüència
    0,     //mode 6 rebrem per port sèrie
    0,     //mode 7 mode "buit"
    0,     //mode 8 mode "buit"
    0     //mode 9 mode "buit"
};
```

I en la funció loop, hi posem una escolta al botó que cada cop que el pressionem modificarà el mode de conducció:

```
Serial.print("Presionat! ");
if (mode==9){
    mode = 0;
} else {
    mode++;
}
Serial.println(mode);
escriu7digis(mode);
moume(conducc[mode]);
```

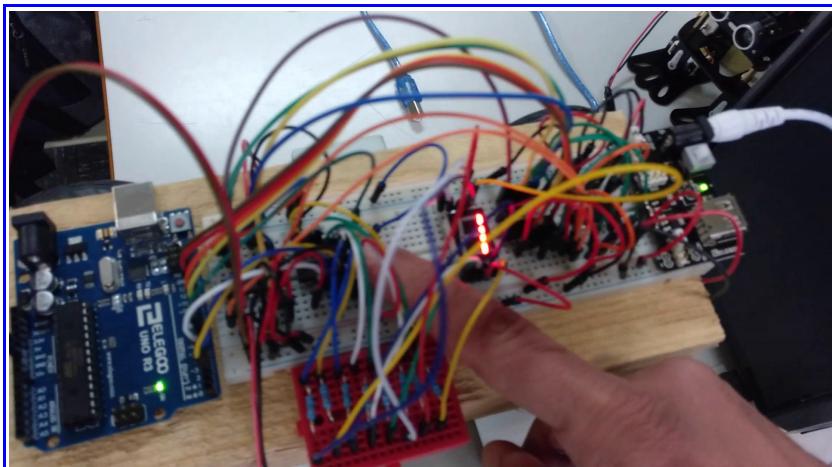
i una funció que mostrerà al display el mode en funcionament:

```
void escriu7digis(byte digit) {
    Serial.print("escriu: ");
    Serial.println(digit);
    digitalWrite(latch2, LOW);
    shiftOut(dataP2, clock2, LSBFIRST, d7digits[digit]);
```

```
        digitalWrite(latch2, HIGH);  
    }
```

Per a fer funcionar el display estem usant novament un chip de desplaçament de registre 74HC595 amb el que amb només tres pins de l'arduino podem escriure 8 valors corresponents als 8 segments del display.

També incorporo la bateria que em permet fer-lo autònom.



Vídeo del funcionament dels modes de conducció (click per reproduir)

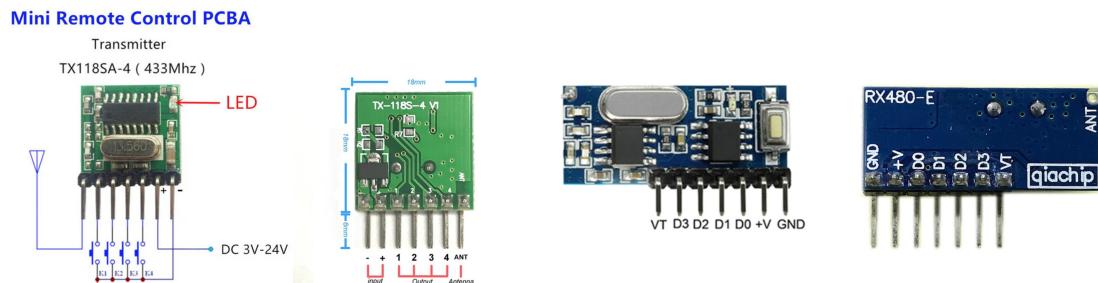


Vídeo del funcionament amb bateria a terra (click per reproduir)

5. Comandament per radiofreqüència

Arribats a aquest punt, i donat el poc temps de què disposem per a la realització del projecte, descarto la idea inicial del sensor de distància que aturi el vehicle i començó a treballar amb el comandament per radiofreqüència. El canvi de planificació és degut a que en el curs ja hem treballat amb el sensor de distància i en canvi no hem treballat amb radiofreqüència, el que resultarà un repte que m'aportarà més coneixements.

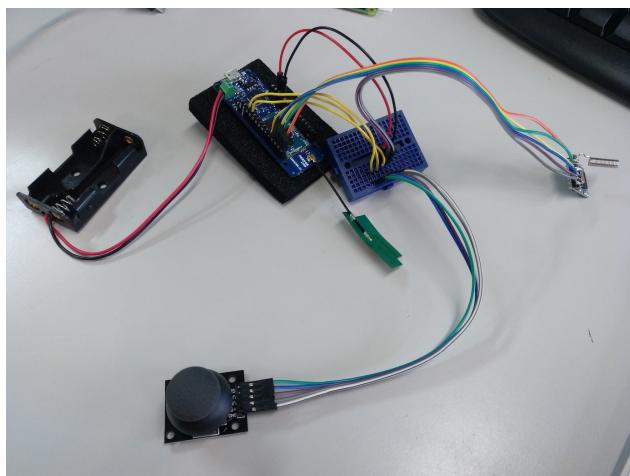
L'emissor triat és el «Qiachip TX118SA-4 ASK RF Transmitter Module Remote Control 1527 Encoding For Arduino Module DIY 433MHz» i el receptor el «QIACHIP RX480E 433Mhz Wireless Remote Control Switch 4CH RF Relay 1527 Encoding Learning Module For Light Receiver Diy Kit»



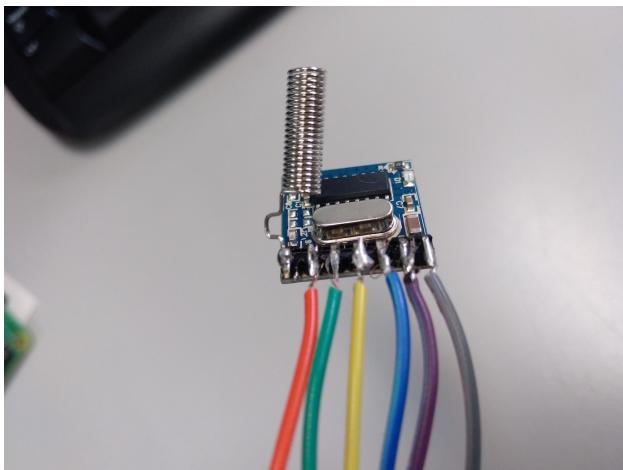
Imatges del mòdul de radiofreqüència i el seu pinnatge proporcionades pel fabricant.

Soldo l'antena i els pins de l'emissor, ja que els pins són molt curts i no queda ben agafat el cable ni es fixa bé a una placa de connexions.

Aquest emissor el connectem a una placa MKR FOX 1200 com a prova, en el seu muntatge final usarem un AZDelivery Nano, una placa similar a l'Arduino nano. Aquesta placa s'usarà com a emissor del comandament a distància. També li incorporem un joystick que ens servirà per guiar el cotxe i un adaptador de bateries AA per tal que sigui independent de la corrent elèctrica.



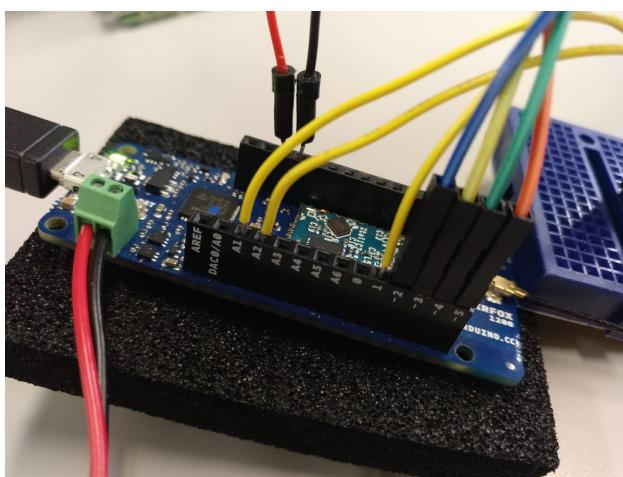
Visió general del muntatge del comandament a distància



Detall de les connexions del mòdul emissor de radiofreqüència

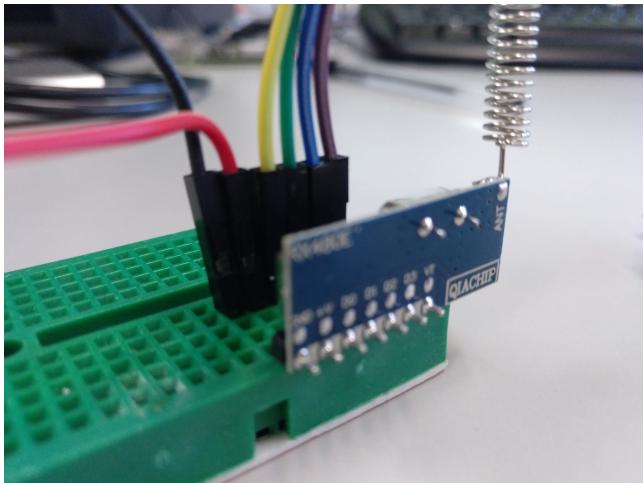


Detall del joystick

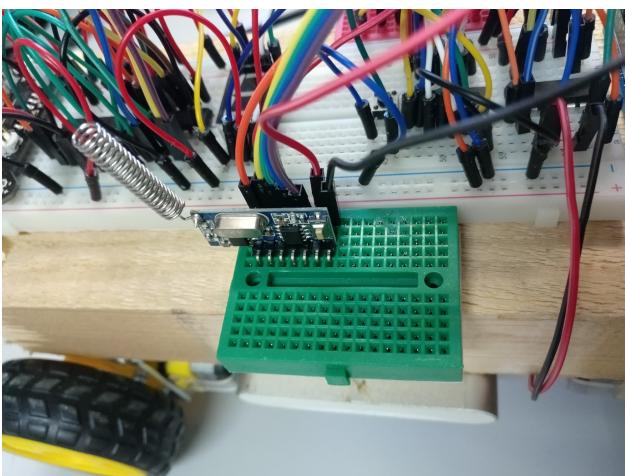


Detall de les connexions a l'arduino MKR FOX 1200

També soldo l'antena del mòdul receptor de radiofreqüència que incorporo en una mini placa de connexions annexa a la placa general per falta d'espai.



Detall de la connexió del mòdul receptor



Detall del muntatge al xassís del vehicle

Seguint les instruccions del fabricant emparerello l'emissor i el receptor.

1. Esborrar les dades existents al receptor pressionant 8 cops el botó del mòdul receptor, el led parpallejarà set vegades.
2. Prémer el botó una, dues o tres vegades dependent de mode que volguem (veure més endavant), el led es queda fixe.
3. Prémer qualsevol botó de l'emissor, el led del receptor parpallejarà tres vegades. L'emparellament s'ha completat.

Modes d'emparellament: (pressionant un, dos o tres vegades el botó en el pas 2)

1. Un cop: Mode momentani.
2. Dos cops: Mode fixe de cada un dels quatre canals.
3. Tres cops: Un cop s'activa un canal queda actiu fins que no s'activi un altre.

Un cop fet això tinc problemes per enviar/rebre dades ja que funciona al revés de l'esperat, està sempre en alt i quan envio un missatge es canvia a baix. Ho soluciono declarant-los com a HIGH i llançant l'esdeveniment quan es posin en LOW, de manera similar a com funcionen els botons. En el fons el funcionament dels pins del mòdul emissor és igual a la dels botons.

Hi ha quatre pins o canals, i un cinquè, el VT, que registra quan hi ha un canvi de senyal en qualsevol dels quatre canals, i per tant quan s'està enviant una comunicació.

D'aquesta manera el funcionament és similar a un comandament a distància d'una porta amb quatre botons. Aquest funcionament em suposarà unes limitacions de "hardware" que hauré de tenir en compte d'ara endavant.

El flux de treball de l'aparell emissor/receptor serà el següent:

1. Faig un moviment amb el Joystick.
2. Com el Joystick és analògic, aquest envia un parell de valors x i y d'entre 0 i 1023. El centre del Joystick coincideix amb el punt 512 d'ambdos eixos. Al pujar el Joystick cap amunt l'eix y canvia a 0 i al baixar, a 1023. D'igual manera funciona el eix X.
3. Un cop rebo els valors analogics del Joystick els analitzo per detectar quan hi ha hagut moviment i en quina direcció.
4. Mapejo aquests valors per transformar-lo en un senyal digital que enviaré a un dels quatre canals del comandament de radiofreqüència, segons el valor.
5. D'aquesta manera quan situo el Joystick en posició endavant activo el canal 1, cap enrere el canal 2, cap a la dreta el canal 3 i cap a l'esquerra el canal 4.
6. El receptor rep aquests senyals i detecta quan hi ha un canvi, si hi ha un canvi dispara l'ordre de moure's en la direcció seleccionada.

Hem de fer ara un codi a part per al mòdul emissor, que serà el comandament a distància. Aquest comandament, com ja hem comentat, utilitza una altra placa, l'MKR FOX 1200, que tindrà la seva pròpria programació independent de l'Arduino UNO del cotxe.

Per poder detectar els moviments del joystick el que faig és fer una quadricula de tres per tres sobre l'eix x i y del joystick. Si els eixos se situen del 0 al 1023, els rangs divisoris es trobaran en els valors 250 i 750.

	0	250	750	1023
0	sense us	endavant	sense us	
250	esquerra	centre	dreta	
750	sense us	enrere	sense us	
1023				

D'aquesta manera, per exemple, si el valor x es troba entre els valors 250 i 750 i l'eix y més petit de 250, estarem donant l'ordre d'activar o parar el moviment endavant

El codi que ho farà és com segueix:

```
if((lx>250 and lx<750) and ly < 250){ // endavant  
    estat = 1;  
} else if(lx > 750 and (ly>250 and ly<750)){ // dreta  
    estat = 3;  
} else if((lx>250 and lx<750) and ly > 750){ // enrera  
    estat = 2;  
} else if(lx < 250 and (ly>250 and ly<750)){ // esquerra  
    estat = 4;
```

```

    estat = 4;
} else {
    estat = 0;
}

```

D'aquesta manera en una variable "estat" emmagatzemo en quina posició està el joystick.

Paral·lelament estic llegint aquest valor i en el cas que algun d'aquests estats canviï executo l'ordre de canviar l'estat del pin corresponent.

Per detectar els canvis d'estat emmagatzemo l'estat anterior (eaj) i el comparo amb l'estat actual (estat).

```

if(estat != eaj){
    Serial.print("estat anterior: ");
    Serial.print(eaj);
    Serial.print(" estat actual: ");
    Serial.println(estat);
    switch(estat){
        case 1:
            premboto(04);
            break;
        case 2:
            premboto(03);
            break;
        case 3:
            premboto(01);
            break;
        case 4:
            premboto(02);
            break;
        default:
            switch(eaj){
                case 1:
                    premboto(04);
                    break;
                case 2:
                    premboto(03);
                    break;
                case 3:
                    premboto(01);
                    break;
                case 4:
                    premboto(02);
                    break;
                default:
                    break;
            }
            break;
    }
    eaj=estat;
}

```

El segon switch anidat el que fa és detectar quan torna a 0 (centre o cantonades) i torna a canviar l'estat del darrer pin activat, de manera que al pujar activaria el pin d'avançar i al deixar anar i tornar al centre desactivaria aquest pin l'aturaria.

La funció premboto el que fa és posar en alt tots els pins i posar en baix el pin que jo li dic, esperar 10 milisegons i tornar a posar en alt el que jo li he dit, de manera que enviarà un senyal com si s'hagués premut un botó en aquell canal, un senyal que rebrà el receptor.

```
void premboto(int bt){  
    Serial.println("Premut botó "+String(bt));  
    digitalWrite(o1, HIGH);  
    digitalWrite(o2, HIGH);  
    digitalWrite(o3, HIGH);  
    digitalWrite(o4, HIGH);  
    digitalWrite(bt, LOW);  
    digitalWrite(LED_BUILTIN, HIGH);  
    delay(10);  
    digitalWrite(bt, HIGH);  
    digitalWrite(LED_BUILTIN, LOW);  
}
```

La resta de programació es farà en el codi del cotxe que incorpora el receptor:

El primer que fem es escoltar el pin vt, que detecta quan s'ha rebut un senyal qualsevol. Si hi ha un canvi respecte a l'estat anterior (ear) activarà el procés de revisió:

```
val0 = digitalRead(vt);  
if(val0 != ear){  
    if (val0 == HIGH){ //detecto flanc ascendent (premo)  
        revisa();  
    }  
}  
ear = val0; //actualitzo estat per al proper loop
```

La funció revisa el que fa és emmagatzemar en un array (esta) el estat actual dels quatre pins

```
esta[0] = digitalRead(i1);  
esta[1] = digitalRead(i2);  
esta[2] = digitalRead(i3);  
esta[3] = digitalRead(i4);
```

i tot seguit la compara amb el darrer estat dels pins registrat. D'aquesta manera podem saber quin pin ha rebut senyal i moure'ns en la direcció elegida.

```
boolean canvis = comparaArrays(esta,estb);
```

```
boolean comparaArrays(int array1[],int array2[]){  
    if(array1[0] != array2[0]){  
        canviara=1;  
        return(true);  
    }  
    if(array1[1] != array2[1]){  
        canviara=2;  
        return(true);  
    }  
    if(array1[2] != array2[2]){  
        canviara=3;
```

```

        return(true);
    }
    if(array1[3] != array2[3]){
        canviara=4;
        return(true);
    }
    return(false);
}

```

Per tant si el pin que s'ha modificat és el pin 1 vol dir que volem moure'ns endavant i si el cotxe està parat activarem el moviment endavant i si ja està movent-se l'aturarem.

```

if(canvis==true){
    Serial.println("canvia: "+String(cambiara));
    if (esta[cambiara-1]==1){
        Serial.print("Activa "+String(cambiara));
        moume(conducc[cambiara]);
    } else {
        Serial.print("Atura ");
        moume(conducc[0]);
    }
}

```

Aquest flux no funcionava correctament ja que quan donava l'ordre d'una direcció, el primer cop no la detectava correctament, i detectava la darrera direcció marcada, de manera que havia una espècie de buffer.

Per exemple si donava l'ordre d'anar endavant, després tornava a donar l'ordre d'anar endavant per aturar-lo, i després donava l'ordre de girar dreta, el primer cop rebia l'ordre d'anar endavant i no girava a la dreta, el que feia la conducció impossible.

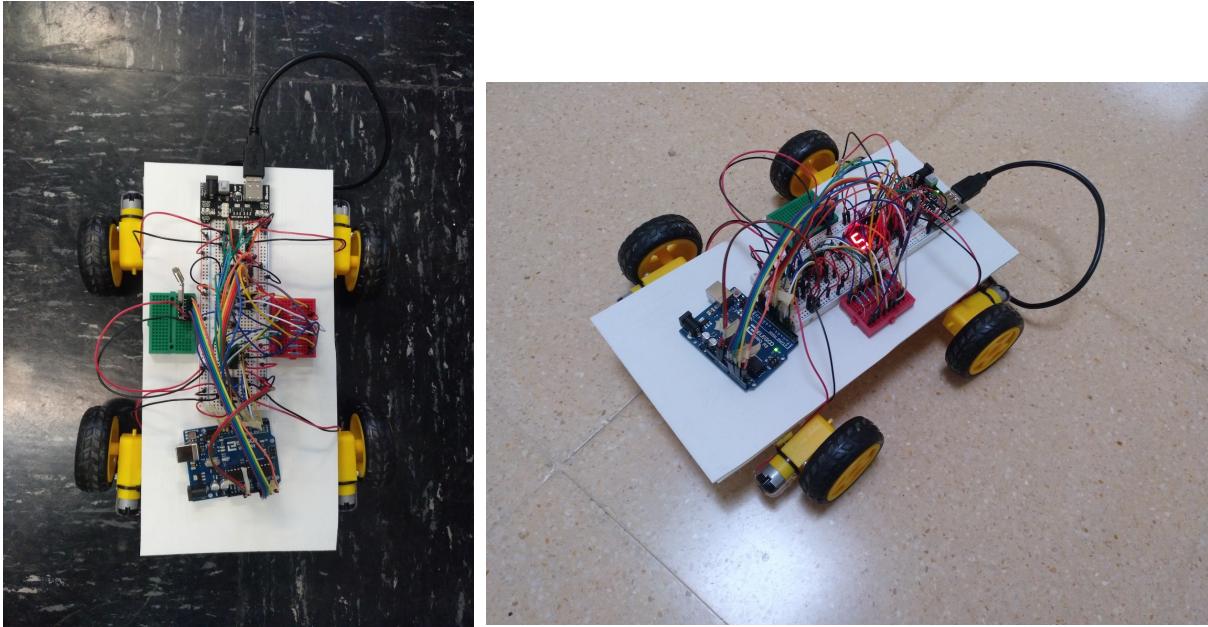
La causa d'aquest malfuncionament es devia a que en el moment de revisar, quan emmagatzema l'estat actual dels pins per comparar-lo posteriorment amb l'anterior, la funció s'activa de manera instantànea i no dona temps a que el pin hagi canviat d'estat, per tant la lectura no és correcta. La solució al problema és incorporar un delay de 20 milisegons a l'inici de la funció, que és imperceptible en quan a la resposta del vehicle, però genera prou temps per tal que la lectura del pins sigui la correcta.

Paral·lelament, el cotxe continuava patint problemes de tracció, sobretot en els girs, ja que no era capaç de girar correctament. Aquest problema era físic degut al pes de la fusta, de manera que torno a modificar el xassís del vehicle, canviant la fusta per un polipropilé.



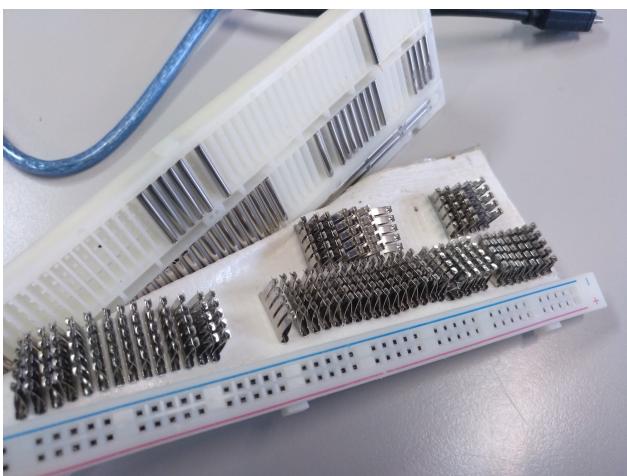
Comparació dels tres xassís del GI



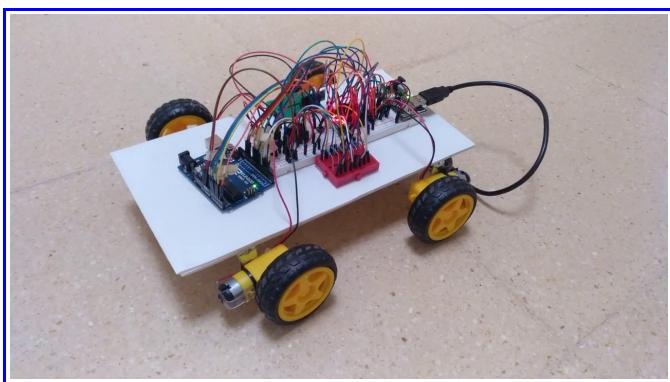


Vista del vehicle un cop muntat el tercer xassís

En el procés de trasllat dels components la placa de connexions es va trencar, el que va fer necessari substituir-la per una de nova i tornar a realitzar el muntatge de tots els components i cables del vehicle.



Protoboard trencada on es pot apreciar el sistema intern de connexions de la mateixa.



Vídeo final del funcionament (clic per reproduir)

6. Millors i futures ampliacions

Un cop acabada la construcció del G1, i la seva programació, funcionant de manera correcta. Dono per tancada la versió 1, i comencem a pensar possibles millors per a futures versions del cotxe. la G1.1 o fins i tot la G2.

La primera millora passa per una modificació del xassís del vehicle que permeti, mitjançant una superfície alternativa al polipropilè un suport prou rígid per tal que es pugui subjectar les rodes perfectament paral·leles i alineades, però alhora prou lleuger per que no penalitzi la seva conducció.

La següent millora consistiria en la incorporació d'un botó en el comandament a distància que fes la mateixa funció que el botó inclòs al G1, per poder seleccionar el mode de conducció de manera remota i no haver de prémer el botó que hi ha enmig de la placa de connexions.

D'altra banda la incorporació d'un sensor de distància que permeti un mode de conducció "autònoma" pel qual el cotxe es mogui endavant fins detectar un obstacle, moment en el que s'aturarà i girarà per no col·lisionar i evitar-lo.

El fet que el cotxe sigui de tracció a les quatre rodes i el gir el faci mitjançant el desplaçament endavant de les rodes d'un costat i el desplaçament enrere de les rodes de l'altre costat, permet un gir perfecte sobre el seu propi eix, el que permet una conducció en "quadrícula", pel que podria incorporar-se un nou mode de conducció pel qual sobre una quadrícula dibuixada a terra, li poguéssim enviar una posició i que el cotxe anés a la posició desitjada.