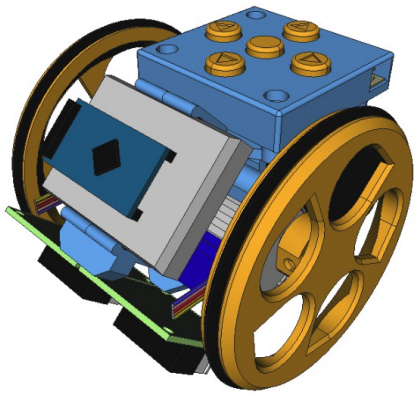
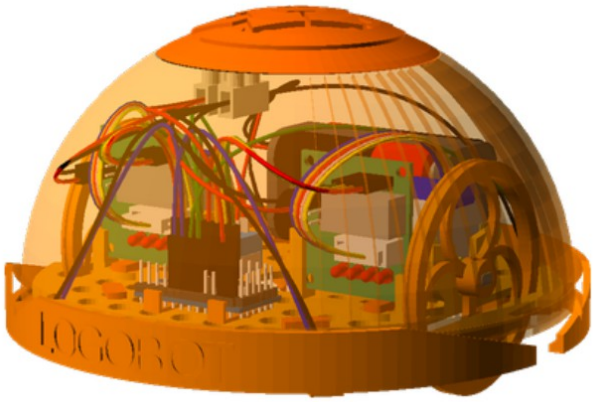
Institut Puig Castellar
Santa Coloma de Gramenet

PROJECTE ESCORNABOT I LOGOBOT 2N ASIX





GRIGORI ARAKELYAN
RAÚL PÉREZ MORALES
JORDI JIMÉNEZ SANTOS

©

INDEX:

1. INTRODUCCIÓ:

1.1 Context i justificació del projecte:

Necessitats a cobrir del projecte:

La necessitat principal en aquest projecte es fomentar l'aprenentatge dels més petits amb aquests bots, per tal d'introduir-los al món de la tecnologia d'una manera més entretinguda i didàctica.

Perquè fem aquest projecte:

Principalment hem escollit aquest tema de treball, per aprendre una mica com funciona el món de la robòtica i la programació d'Arduino.

Resultat a obtenir:

Dos robots, Escornabot i Beebot, completament operatius i funcionals.

1.2 Objectius del projecte:

En aquest projecte el nostre objectiu es realitzar la construcció de dos robots (Escornabot i Beebot). Aquests robots estan ideats per l'aprenentatge autodidacta dels més petits. Els dos robots estan basats en el projecte americà Beebot.

Abans de que tinguem tot, començarem a provar en la pàgina http://circuits.io per saber com funciona el codi d'Arduino.

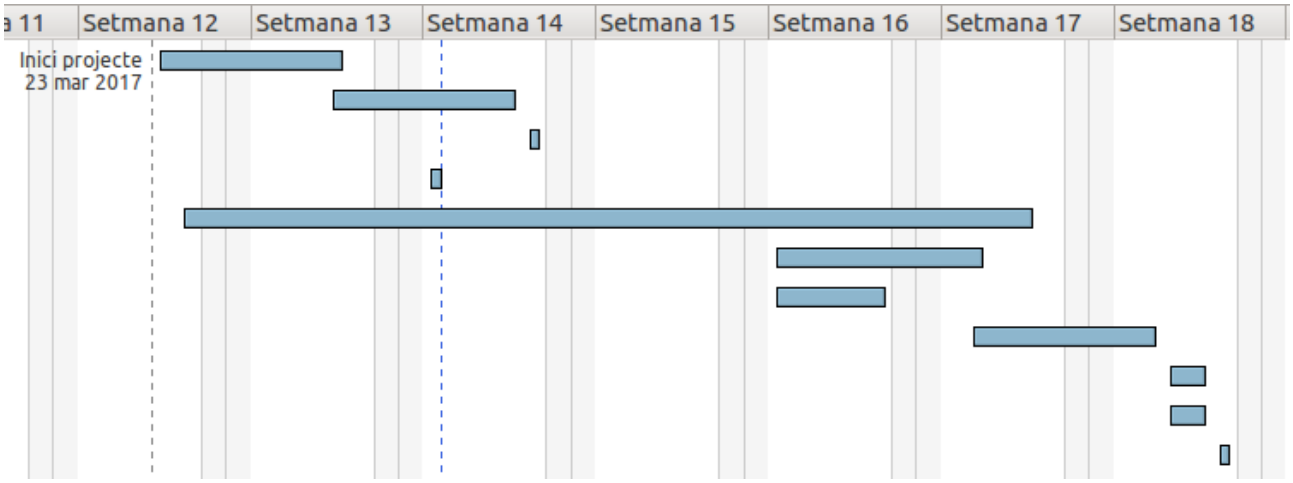
Una vegada tinguem tot el material necessari, farem el muntatge, després, passarem amb la programació en Arduino, anar provant que tot funcioni, perfeccionant el codi per a que els nostres robots puguin realitzar el nostre objectiu, després d'això, passarem al muntatge de la seva protecció en cadascun dels dos robots.

Per a que els robots tinguin un disseny més atractiu, imprimirem peces 3D per a que tinguin un fi més agradable i robust per a l'usuari final.

Una vegada tinguem els dos robots fets i programats, l'objectiu final serà fer una comparativa sobre quin Bot és millor, en els aspectes necessaris per a l'usuari final.

1.3 Planificació del projecte:

WBS	Nom	Feina
1	Brainstorming	6d
2	Documentació primera part de la memòria	6d
3	Entrega primera memòria	1d
4	Compra material 'Escornabot'	1d
5	Proves Arduino	25d
6	Programació codi	7d
7	Fabricació peces 3D	5d
8	Muntatge 'Escornabot 1.0'	6d
9	Carrega de codi	2d
10	Primeres proves	2d
11	Segona entrega	1d

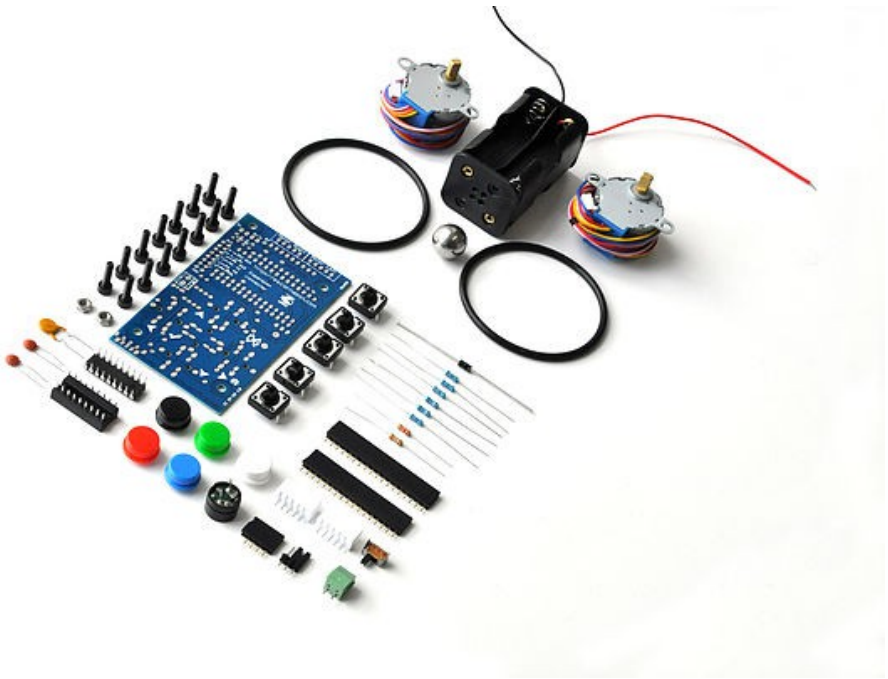


* Només hem assignat les hores, tenint en compte que realitzarem la construcció d'un dels robots.

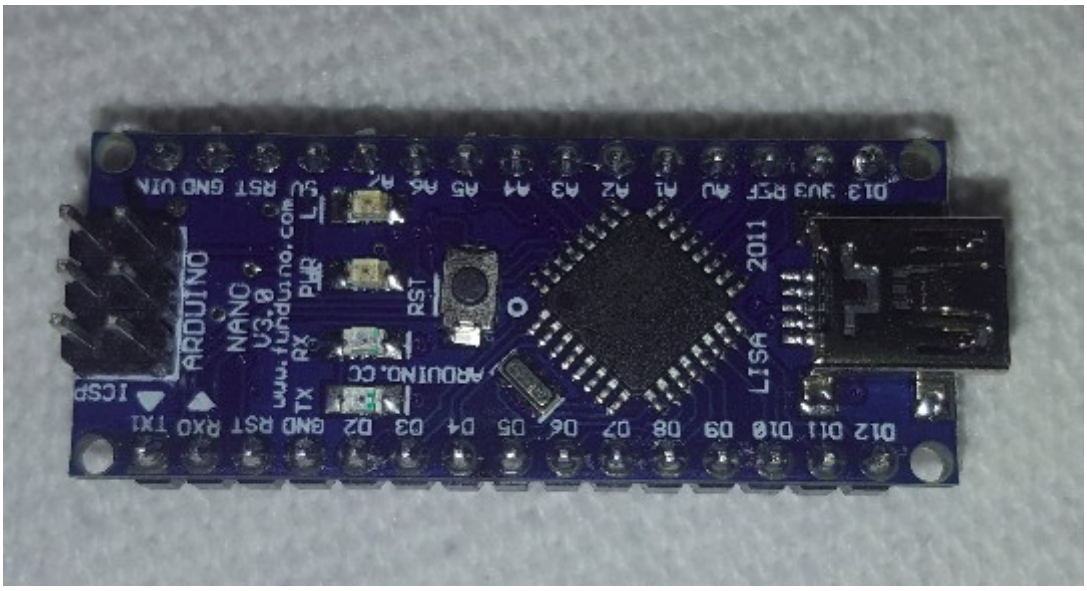
2 Parts de Escornabot:

Les peces necessàries per a muntar un Escornabot són les següents:

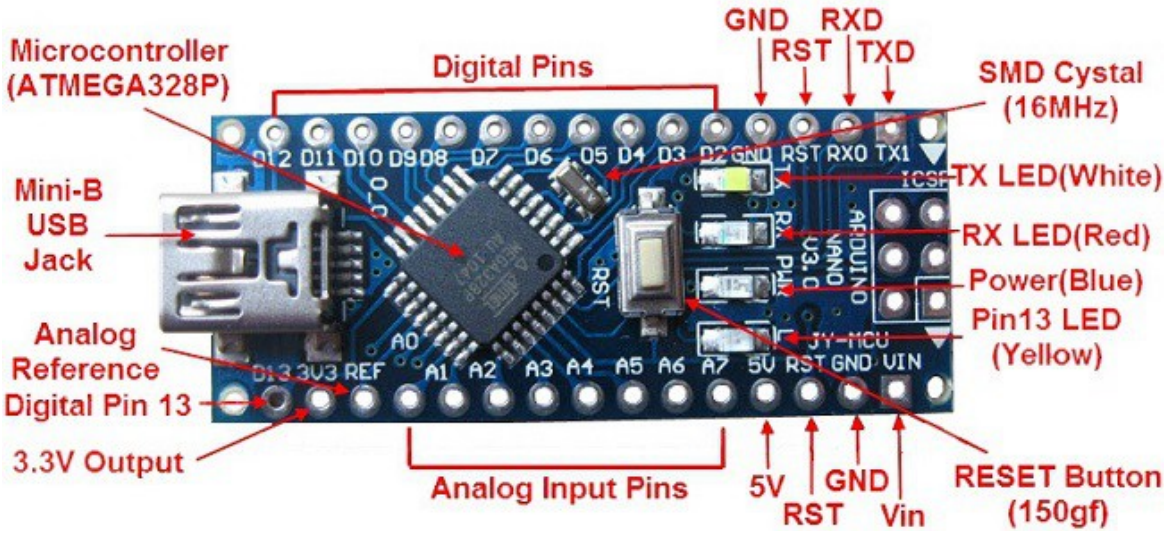
- Arduino Nano (programat)
- *PCBoard* (per soldar els components)
- 2 motors
- Porta bateries
- ULN2803 + sòcol de muntatge
- Tires de pins per Arduino
- Pal de 3 pins + pont
- 2 Connectors per a motor
- Connector d'alimentació
- Mini Interruptor BF3
- *Buzzer*
- Fusible rearma-ble 500 mA
- díode *Schottky*
- Resistències (100 Ω, 1 KΩ, 10 KΩ, 18 KΩ, 22 KΩ)
- Condensadors (100nF)
- 5 Polsadors + caps de colors
- Bola d'Acer
- Juntes tòriques per a les rodes
- Cargols necessària per al muntatge de les peces impreses 3D
- (Bluetooth opcional)



2.1 Arduino Nano:

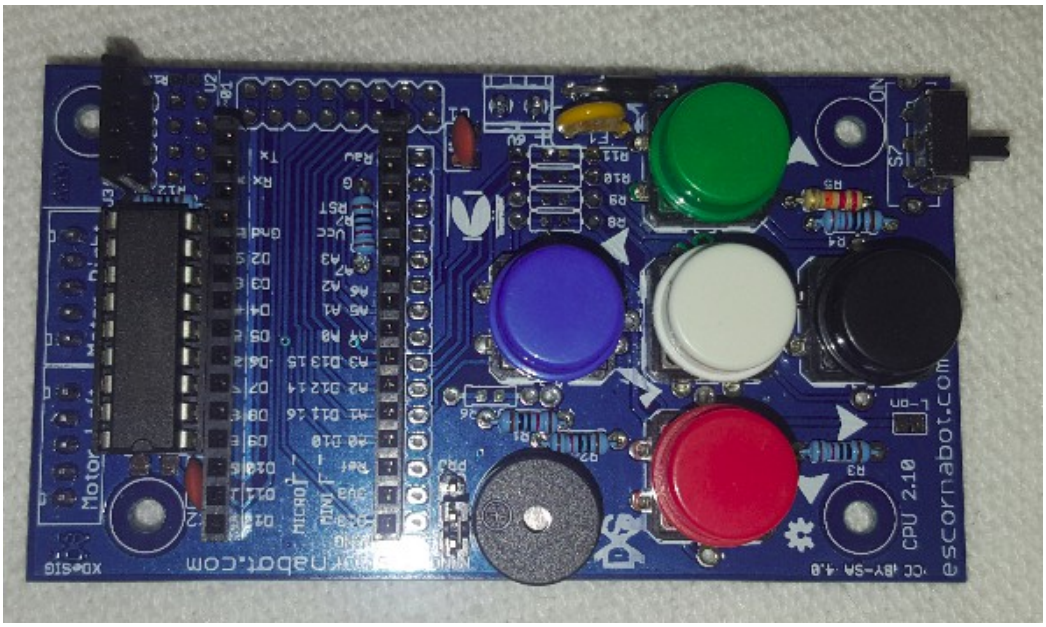


El Arduino Nano és una petita i completa placa basada en el ATmega328 o ATmega168 que s'utilitza connectant a una *protoboard o pcboard*. No posseeix connector per a alimentació externa, i funciona amb un cable USB *Mini-B* en comptes del cable estàndard.



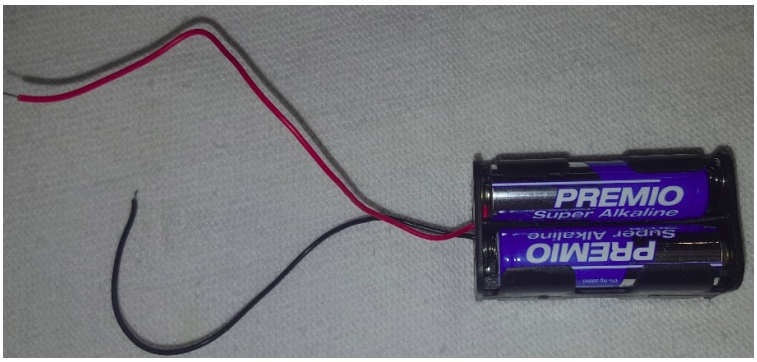
2.2 *PCBoard:*

La placa un cop soldades totes les peces queda de la següent manera:



2.3 Porta Bateries:

El porta bateries té dos cables, un positiu i un negatiu, es connecta a la part posterior de la *PcBoard*, es necessiten 4 piles alcalines per al funcionament del robot:

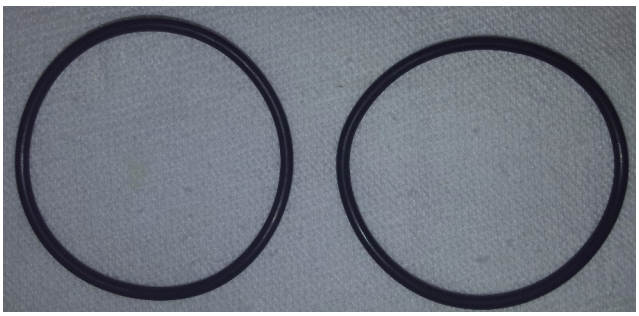


2.4 Cargols:

Es necessiten catorze cargols per cargolar les peces 3D entre altres coses, com els motors de les rodes o la mateixa *PcBoard:*

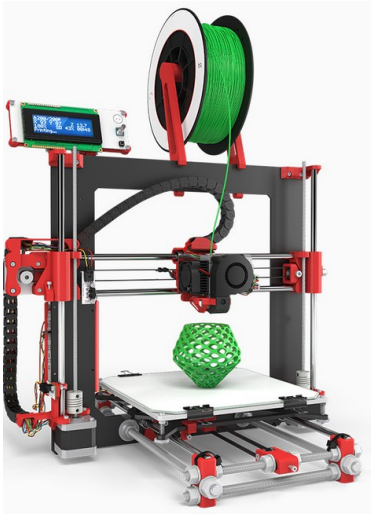
2.5 Gomes per rodes:

Es necessiten dues gomes per a les rodes 3D (posteriorment s'imprimiran amb la impressora) per aconseguir adherència a terra:



3 IMPRESSORA BQ HEPHESTOS PRUSA I3

Hem fet servir la impressora *Bq Hephestos Prusa i3* que ens ha prestat el centre per imprimir les peces 3D del nostre Escornabot:



La impressora 3D *Prusa i3 Hephestos* és un projecte lliure dissenyat i desenvolupat pel departament d'Innovació i Robòtica de *Bq*. *Hephestos* pren la base de la *Prusa i3* i afegeix diverses millores extretes d'altres impressores com la *PowerCode*, una curiositat que té la impressora és que ella mateixa ha fabricat les seves peces, amb la qual cosa una impressora podria fabricar les peces necessàries d'una altra impressora com aquesta.

3.1 Dimensions:

-Dimensions impressora: (x)460 x (y)370 x (z sense rotlle)510 (z amb rotlle)583 mm
-Dimensions àrea de impressió: (x)215 x (y)210 x (z)180 mm
-Dimensions caixa: (x)400 x (y)400 x (z)250mm

3.2 Observacions:

Les peces impreses no sempre s'imprimeixen exactament com ens els imaginem, de vegades, s'imprimeixen amb alguns defectes, en el nostre cas hem hagut de fer ús de papers de vidre entre altres eines per poder encaixar o adaptar les peces.

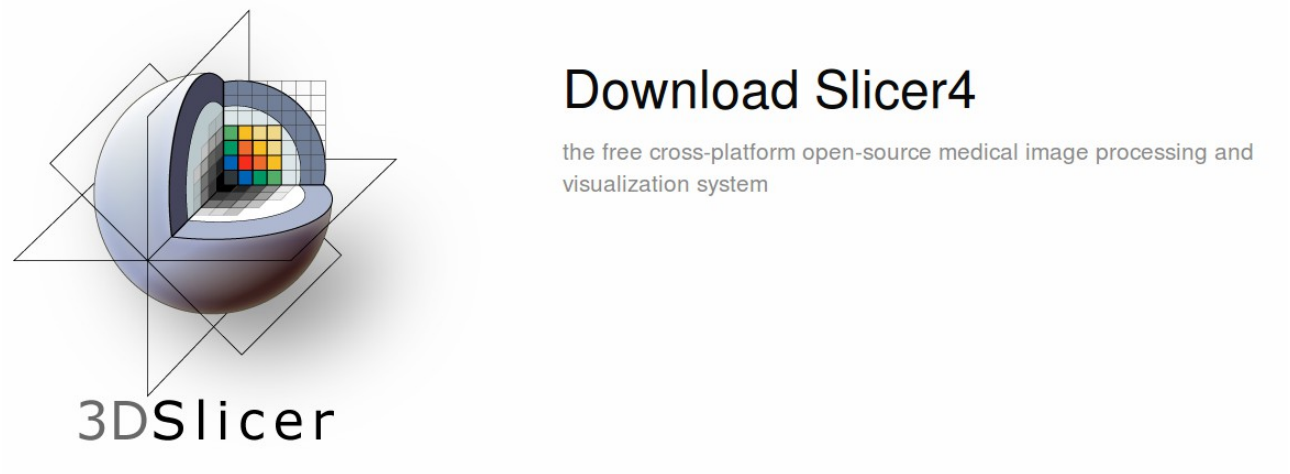
4 KIT 3D:

4.1 Conversió *'.stl'* a *'.gcode'*:

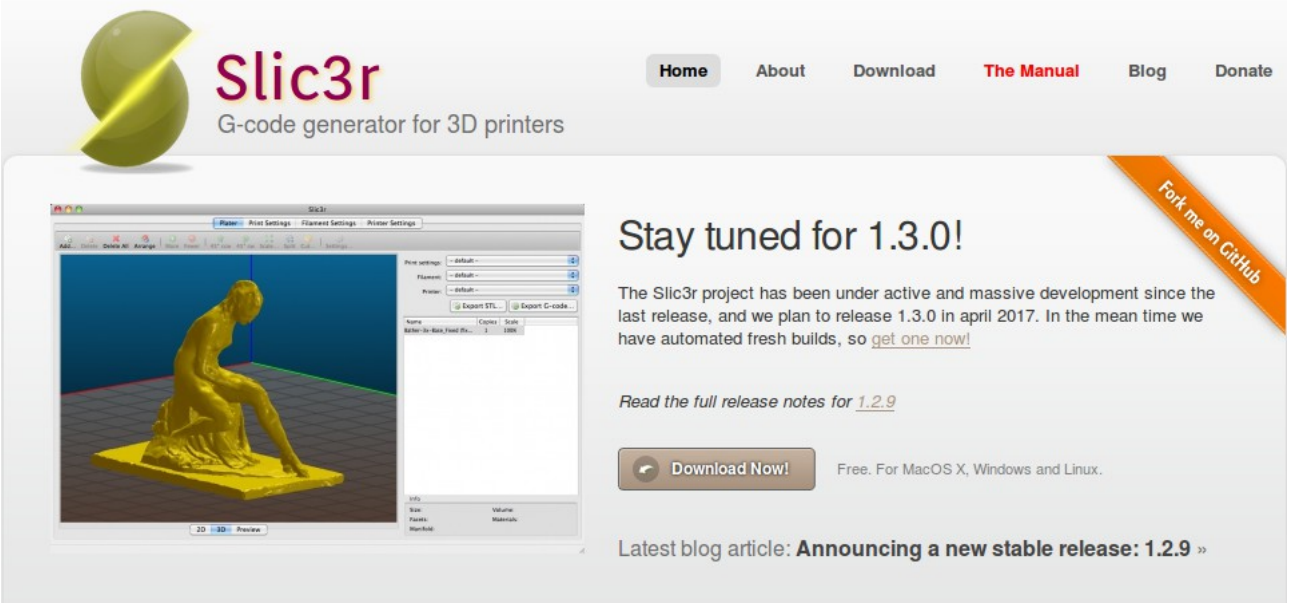
Abans de començar amb les peces 3d que imprimirem a la impressora, comentarem quines han estat les eines que hem utilitzat per passar els fitxers ***'.stl'*** a ***'.gcode'***.

G-code es el llenguatge que entén la impressora, està basada en les coordenades en les que treballara la impressora per a dur a terme el seu treball.

Primer vam trobar un programa anomenat ***'3DSlicer'***, que tractava amb una visualització 3D d'alta qualitat, però vam veure que hi havia un problema, que no convertia els nostres fitxers ***'.stl'*** a ***'.gcode'***.



Després, vam descarregar un dels programes de conversió que més ens sortien a *Google*, anomenat ***'Slic3r'***, que es un convertidor de fitxers ***'.stl'*** a ***'.gcode'***.

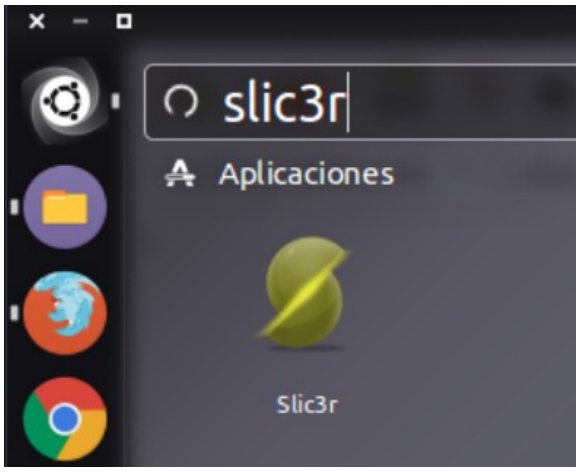


*Com es pot veure a la imatge, la seva interfície sembla molt simple, a la següent pàgina explicarem el seu funcionament.

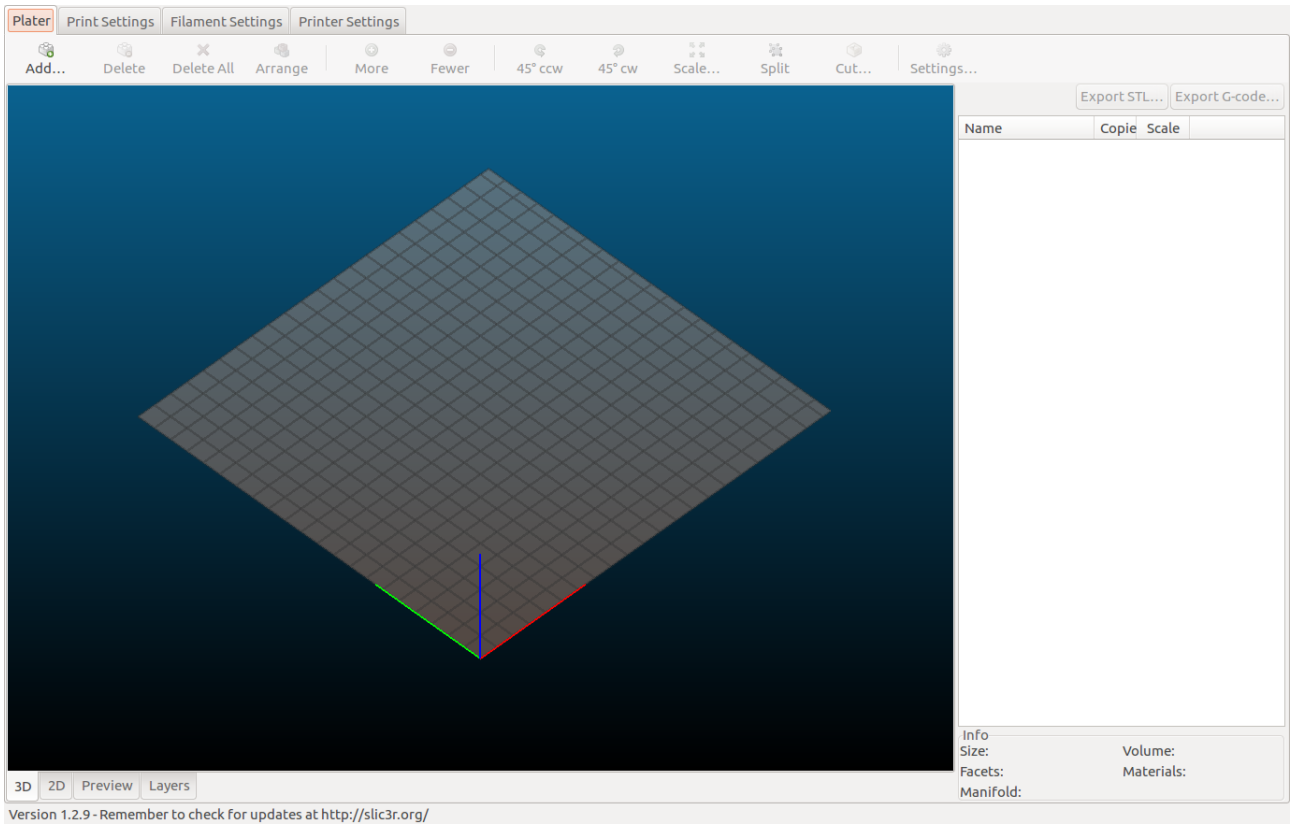
Nosaltres el que farem serà descarregar-nos el fitxer mitjançant el terminal, executant la següent comanda:

apt install slic3r

Una vegada el tinguem descarregat, ens sortirà amb la següent imatge com a icona:



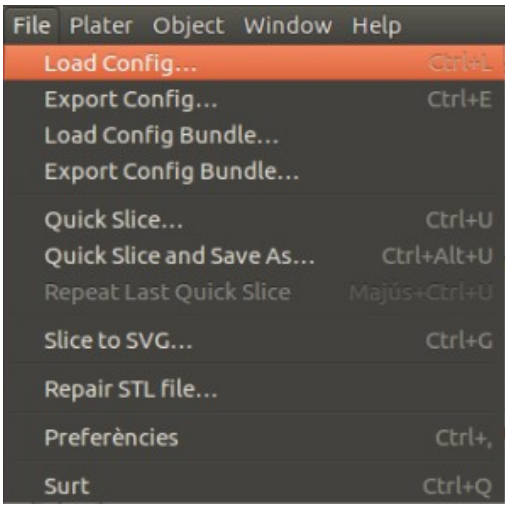
Obrim el programa i la interfície que veurem serà molt simple i intuïtiva:



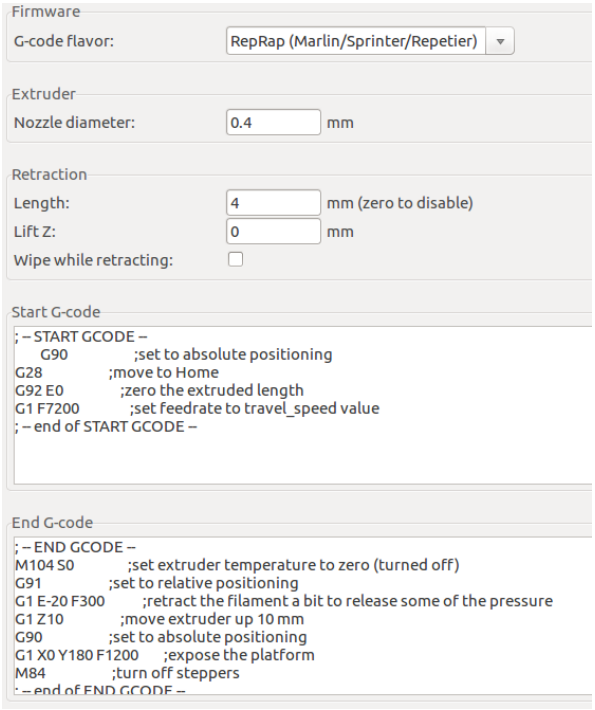
Com veiem a la imatge, a la part superior del programa veiem que tenim diferents opcions, la base de la nostra impressora 3d *(qual després explicarem com hem d'afegir la configuració de la impressora 3d)*, la configuració d'impressió, la configuració del fil amb el qual imprimeix la impressora 3d i la configuració de la impressora 3d.

Després a la dreta veiem els dos tipus de conversió que ens ofereix el programa, a ***'.stl'*** i ***'.gcode'***.

El que hem fet ha sigut importar la configuració de la impressora 3d.



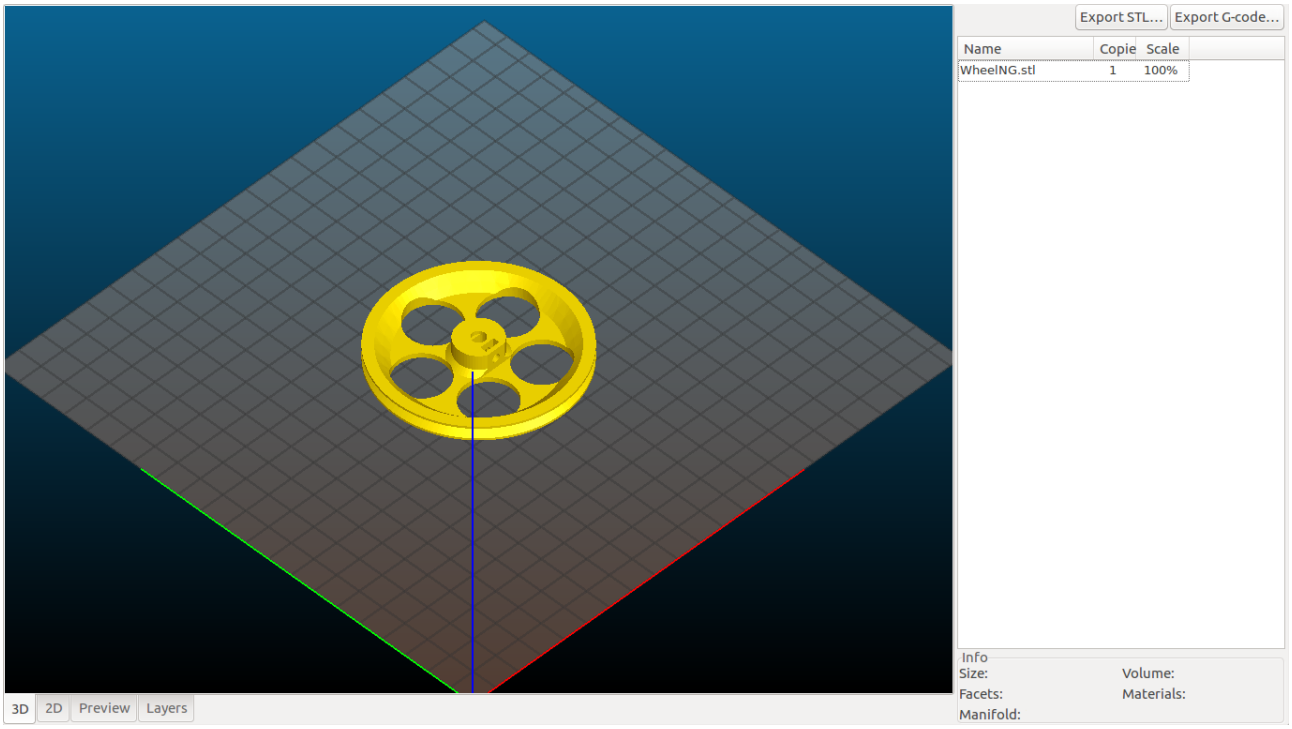
Una vegada vam clicar a ***'Load Config'*** vam escollir la nostra configuració de la impressora, que s'anomena ***'bq_hephestos.ini'***, fitxer de configuració el qual ens el va proporcionar el professor Oscar Torrente.



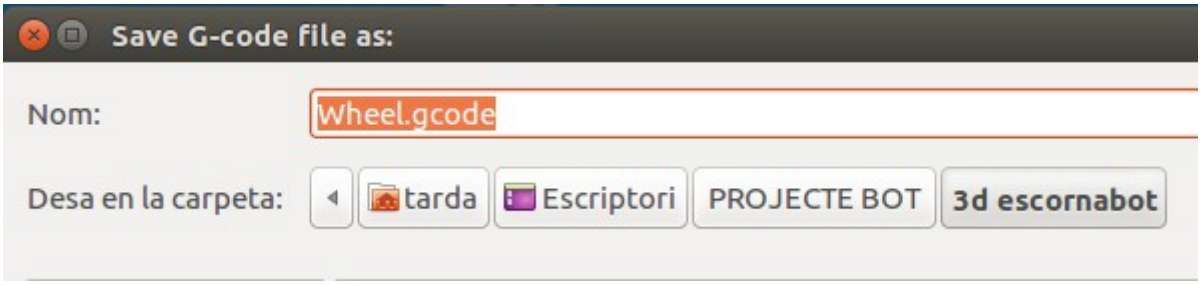
*Una vegada vam carregar la configuració de la impressora, el programa omple tota la configuració pertinent que necessita saber sobre la impressora que utilitzarem.

Després de carregar la configuració de la impressora, vam passar a exportar una peça 3d.

Primer, seleccionem el fitxer ***'.stl'*** que volem convertir a ***'.gcode'*** i el carreguem dins del programa *(hi han dos tipus de carrega de fitxers, un d'ells es arrossegar el fitxer des de la carpeta cap al programa, i l'altre, carregant-lo des de el mateix programa).*



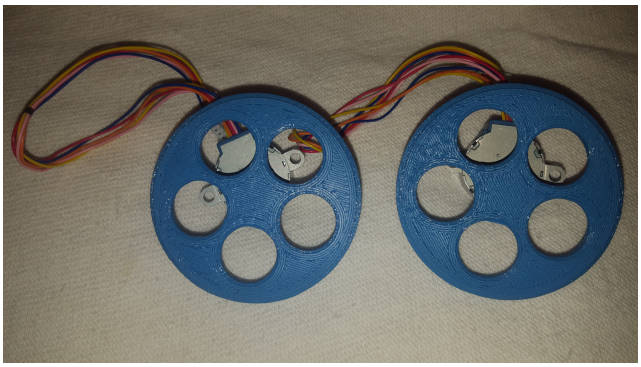
Quan tenim la peça 3d ja carregada, el que vam fer es clicar a ***'Export G-code...'*** que es troba a la part superior dreta del programa junt al botó ***'Export STL...'*** , quan fem clic per exportar-lo, el programa ens pregunta per el nom amb el qual volem guardar-lo i on, nosaltres per exemple, vam anomenar a aquest fitxer ***'.gcode' 'Wheel.gcode'***, per tal de trobar-lo d'una manera més ràpida a la impressora 3d en el menú d'impressió.



Per acabar, vam passar el fitxer ***'.gcode'*** a una targeta SD que ens va proporcionar el centre, per a que la impressora pugui imprimir les peces.

4.2 Rodes:

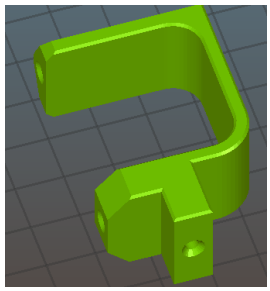
Les rodes 3D dissenyades prèviament, que com podem veure, tenen dos forats, un pel cargol i un altre per a la rosca. Com podem veure, les rodes tenen uns forats, ja que es molt important que quan fem els robots, els motors necessiten un pes molt petit per a que puguin moure's d'una manera més fàcil.



Un cop impreses, les rodes haurien de quedar de la següent forma, però, hem d'anar amb compte, ja que els forats de la rosca i el cargol poden ser impreses malament i hauríem de retocar-ho nosaltres manualment.

4.3 Suport porta bateries:

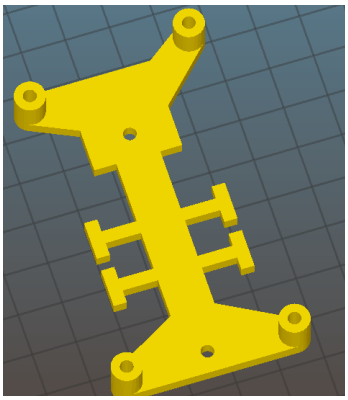
El disseny 3D del suport del porta bateries, que ens hem de fixar bé de que s'imprimiran en una posició òptima per a que la impressora no tingui cap problema a l'hora de fer els forats per als cargols i les parets per al porta bateries.

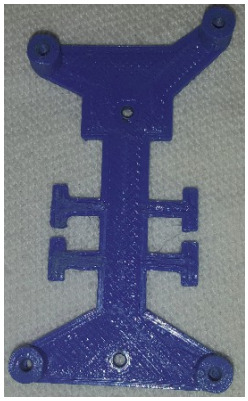


Un cop s'ha imprès el suport porta bateries, mirarem de que les parets s'hagin imprès bé, sinó, hauríem d'agafar un paper de vidre per fer que les parets siguin completament llises, ja que sinó el porta bateries no encaixaria bé i després pot donar problemes l'hora del muntatge.

4.4 Suport Placa:

Aquí veiem el disseny previ del suport de la placa, la qual es de les més importants, ja que aquesta peça s'encarrega d'unir la placa, el suport dels motors i el suport del porta bateries. També cal dir que té aquesta forma a la part mitja per a que puguem administrar els cables dels motors i els cables del porta bateries.

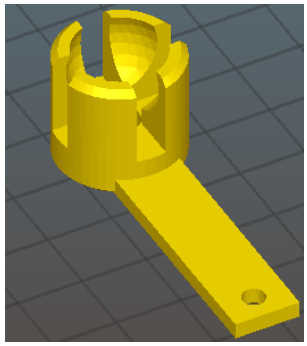




I aquí veiem la peça físicament, que conté sis forats per cargolar a la placa i al suport de la bola, mes dues patilles per cada costat per passar cables i no deixar-los a la vista.

4.5 Allotjament per a la bola:

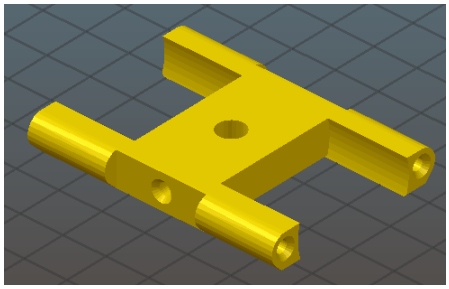
Aquest es el disseny 3D previ de l'allotjament per a la bola, però també es un dels suports que tenim per al porta bateries, ja que el suport del porta bateries té una zona sense tapar, i aquesta part oberta qui s'ocupa de tapar-la i acabar de subjectar el porta bateries es la de l'allotjament per a la bola.

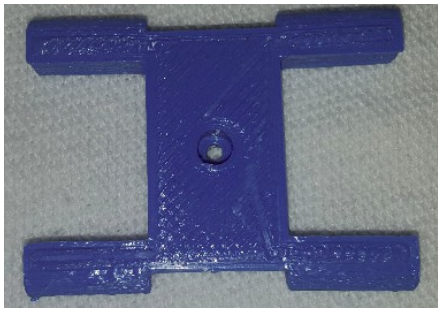


Un cop impresa la peça, aquesta peça serveix per subjectar la bola que farà que el nostre robot pugui rodar sense problemes d'equilibri, però hem d'anar amb compte, ja que una mala impressió de la zona de subjecció de la bola pot fer que hagi un mal comportament en el moviment del *Escornabot*.

4.6 Suport motors:

Aquesta es la peça 3D del suport dels motors, una de les peces mes importants que necessitarem a l'hora del seu perfecte moviment.





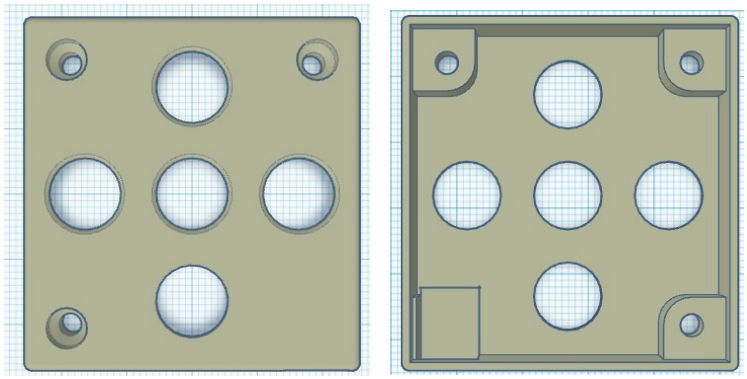
Aquí tenim la peça ja un cop imprès. Un dels punts més importants que hem de tenir en compte, son els forats impresos per als cargols, ja que pot ser que no es facin correctament, i cap la possibilitat de que necessitem un cargol mascle per a donar-li forma per a que els cargols entrin correctament.

5. Millores visuals:

Per al nostre robot hem dissenyat una carcassa per tapar la placa i deixar a la vista els botons.

5.1 Modificació peces 3d

Per a crear una carcassa pel nostre robot Escornabot, utilitzarem la primera part que cobreix només els botons, la qual podem trobar a la web *http://escornabot.com* on ja hem descarregat totes les altres peces 3d.
Utilitzaré l'eina web per a disseny de peces *https://www.tinkercad.com* .
Primer de tot necessitem tapar un dels forats dels cargols, ja que la distribució de la placa es diferent a la del disseny original.

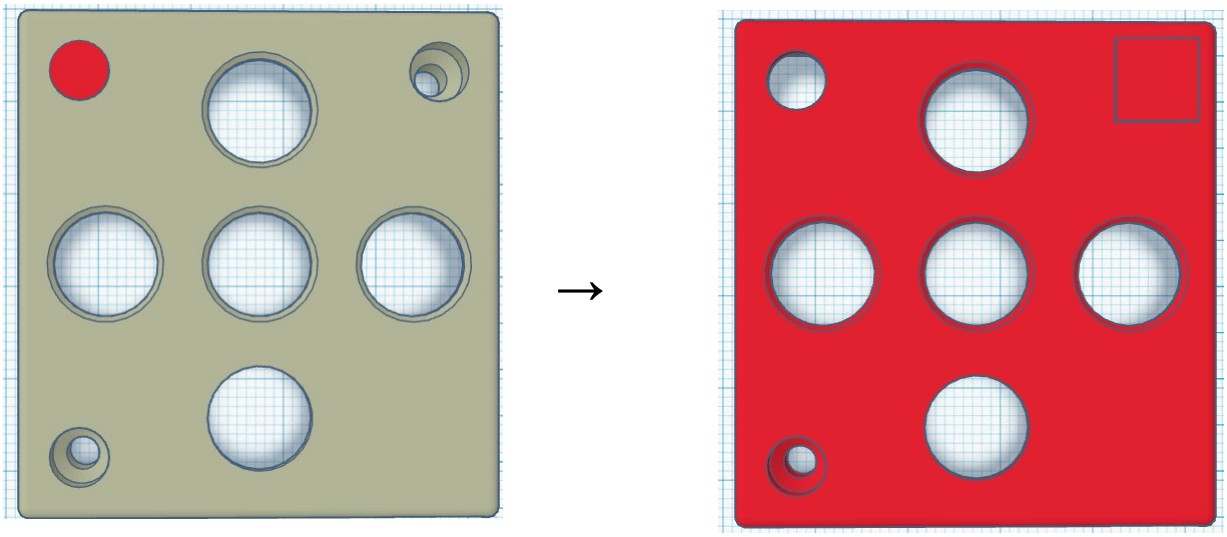


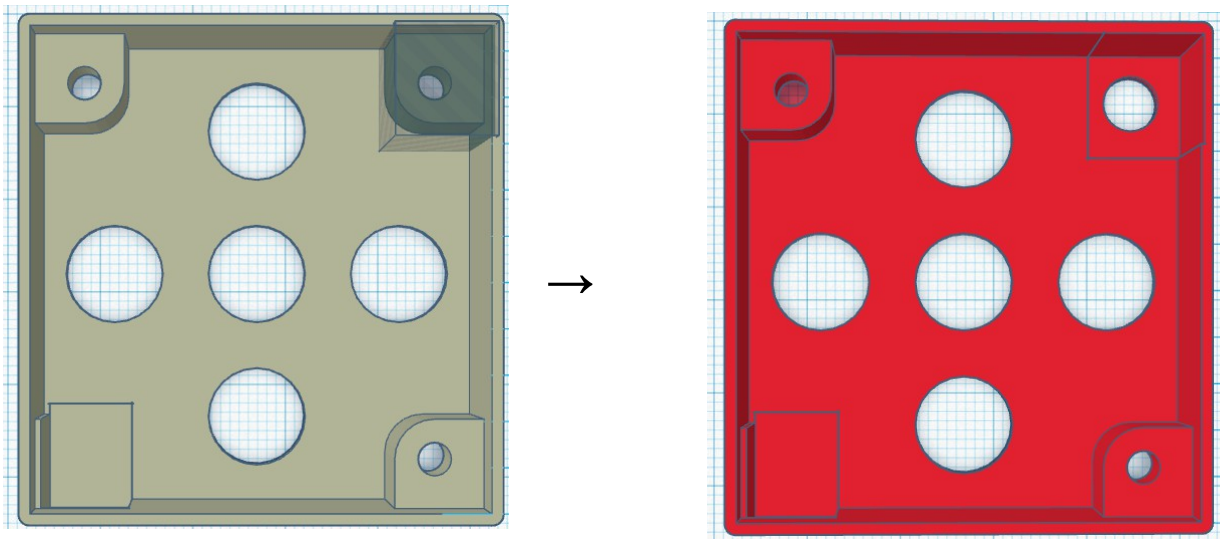
*disseny original de la carcassa. (per dalt i per abaix)

Primer de tot seleccionem la peça geomètrica bàsica, el cub. Li donem les mesures pertinents i la afegim a la peça original.

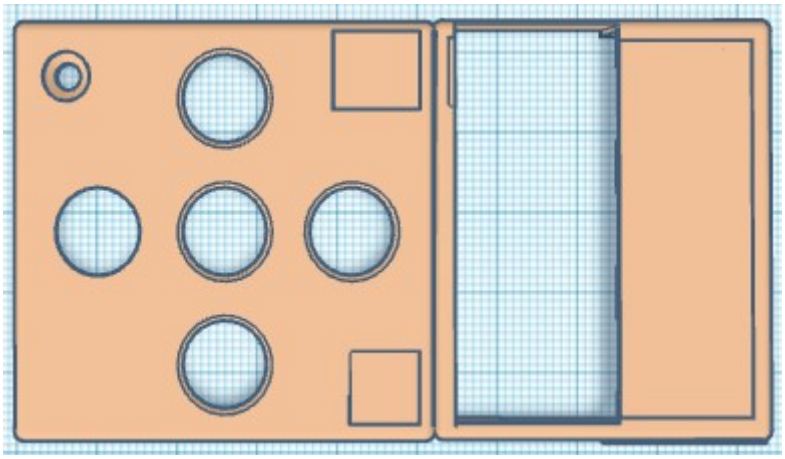
Aquesta eina web, ens permet afegir i retirar trossos de les peces originals, seleccionant si volem fer servir una peça solida o buida, i seleccionant la opció de fer un grup.

Un cop ja tenim les mesures correctes, l'afegim a la peça i fem un grup amb totes elles, de manera que quedi com un únic disseny:





Ja tenim la primera part de la carcassa, que tapara els botons, ara hem de realitzar la creació de la segona part on intentarem deixar la part del Arduino a la vista. Utilitzarem la primera part de la carcassa per realitzar el disseny de un mode més còmode.
El resultat final es el següent:



*prova de carcassa sencera nº1

6. Instal·lació programa Arduino:

Per carregar o editar el codi es necessita l'aplicació de Arduino.

Per descarregar aquest programa hem accedit a la pagina oficial:

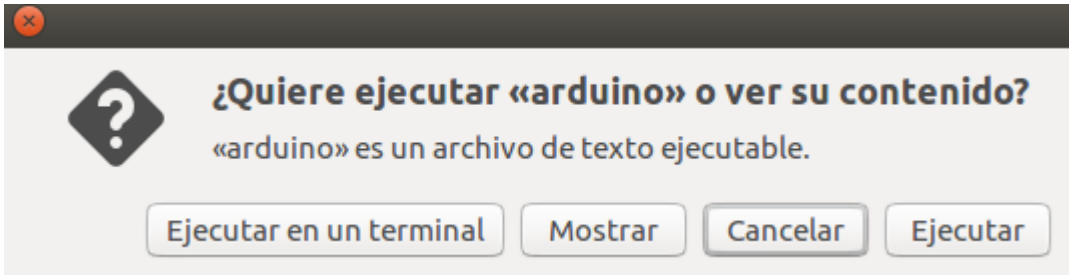
https://www.arduino.cc/en/Main/Software

Un cop descarregat el fitxer l'hem descomprimit, però per iniciar l'aplicació hem trobat un problema, al executar *nautilus* en lloc d'iniciar l'aplicació ens mostra el codi:

→ → →

```
#!/usr/bin/env bash

APPDIR="$(dirname -- "$(readlink -f -- "${0}")" )"

for LIB in \
    "$APPDIR"/java/lib/rt.jar \
    "$APPDIR"/java/lib/tools.jar \
    "$APPDIR"/lib/*.jar \
    ;
do
    CLASSPATH="${CLASSPATH}:${LIB}"
done
export CLASSPATH
```

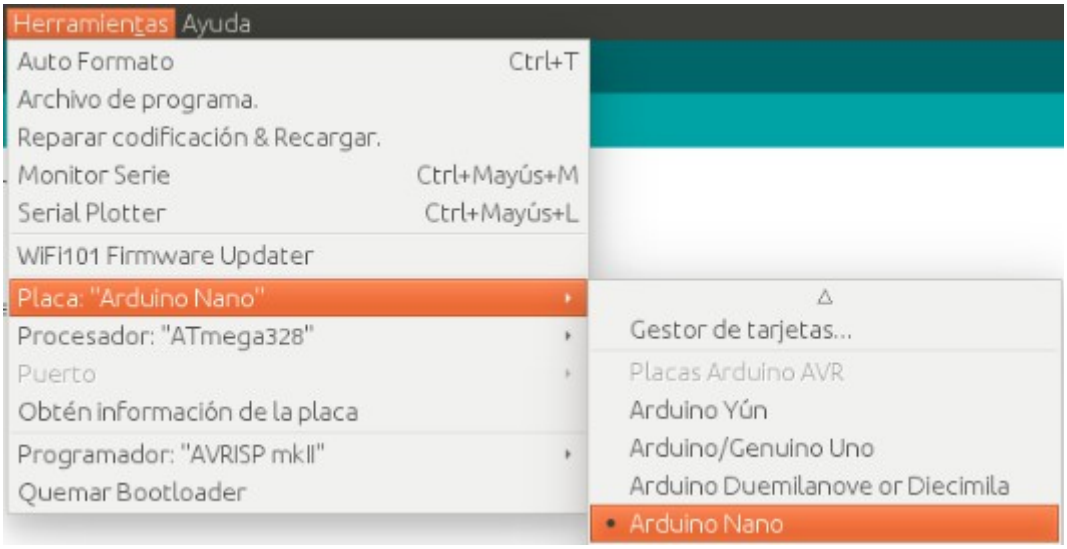
Per solucionar aquest problema el que hem de fer es anar a ***Edita*** → ***Preferencias*** → ***Comportamiento*** i a continuació a l'apartat ***Archivos de texto ejecutables***, marquem on diu ***Preguntar cada vez***, d'aquest mode cada cop que executem el fitxer ens farà la pregunta següent:



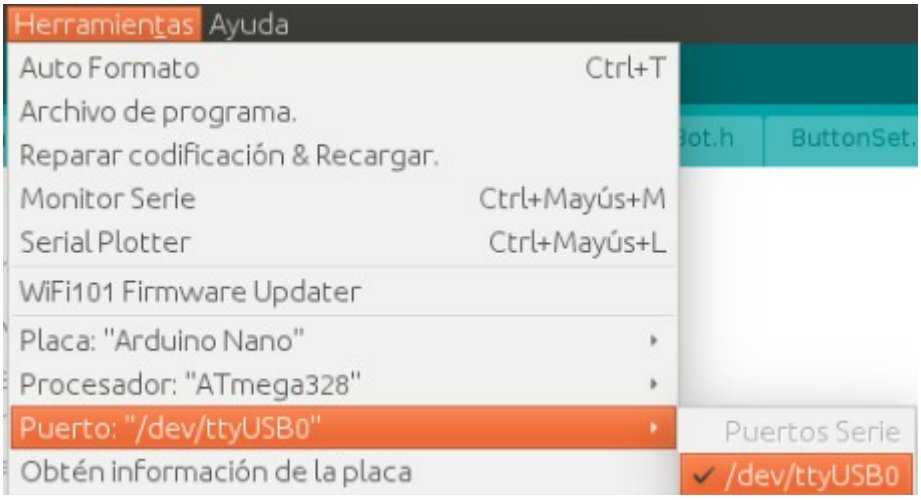
A altres versions de *nautilus* aquest problema ha desaparegut, però si el tenim, la solució es molt simple, com ja hem vist.

6.1 Programa Arduino:

Abans de carregar el codi al nostre Escornabot hem hagut d'indicar a l'aplicació quina placa Arduino estem utilitzant i el port des d'on carregarem el codi. Per indicar el tipus de placa utilitzada fem:
Herramientas → Placa → Arduino Nano



Pel port es igual de senzill:
Herramientas → Puerto → <port_utilitzat>



Per ultim, en un sistema Linux, no tots els usuaris tenen permís per gestionar els ports. Només els usuaris que formin part del grup *dialout* gaudeixen d'aquest permís. El nostre usuari, si no es *root* no serà dins d'aquest grup, així que haurem de afegir-lo de manera manual:

```
EASYNOTE:~$ sudo usermod -a -G dialout grigori
```

Només executar aquesta comanda per terminal ja hem donat permís al nostre usuari per poder enviar dades pels ports a la nostra placa Arduino.

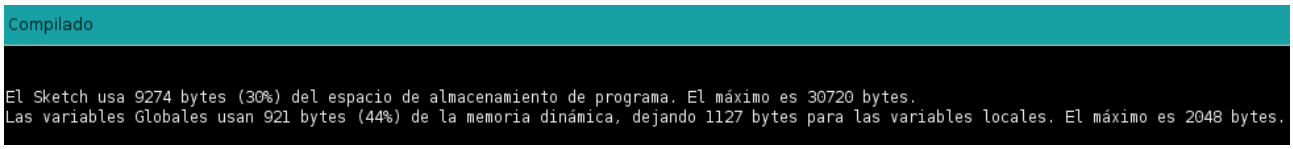
7. Codi Escornabot:

```
////////////////////////////////////////////////////////////////////////
///// configuració general
////////////////////////////////////////////////////////////////////////

// engine to use
#define ENGINE_TYPE_STEPPERS

// botó configurat per a el seu us (entrada analògica(ANALOG), entrada digital(DIGITAL))
#define BUTTONS_ANALOG
//#define BUTTONS_DIGITAL

// mil·lisegons després botó es considerat com pressionat
#define BUTTON_MIN_PRESSED 30

// mil·lisegons que han de passar per que un botó es consideri llarga pulsació
#define BUTTON_LONG_PRESSED 1000

// posar en fals per afegir moviments dessprés de la execució del programa
#define PROGRAM_RESET_ALWAYS true

// store configuration and program within internal EEPROM
#define USE_PERSISTENT_MEMORY true

// cuants moviments seguits por fer l'escornabot
#define MOVE_LIMIT 100

// mil·lisegons per al botó de pausa
#define PAUSE_MOVE_MILLIS 500

// temps que triga en posar-se en moviment (mil·lisegons)
#define DELAY_BEFORE_GO 500

// pausa que fa en cada moviment, abans de realitzar un altre (mil·lisegons)
#define AFTER_MOVEMENT_PAUSE 0

////////////////////////////////////////////////////////////////////////
///// Conifguració dels motors
////////////////////////////////////////////////////////////////////////

#ifdef ENGINE_TYPE_STEPPERS

// stepper pin setup (digital outputs)
#define STEPPERS_MOTOR_RIGHT_IN1 5
#define STEPPERS_MOTOR_RIGHT_IN2 4
#define STEPPERS_MOTOR_RIGHT_IN3 3
#define STEPPERS_MOTOR_RIGHT_IN4 2
#define STEPPERS_MOTOR_LEFT_IN1 9
#define STEPPERS_MOTOR_LEFT_IN2 8
#define STEPPERS_MOTOR_LEFT_IN3 7
#define STEPPERS_MOTOR_LEFT_IN4 6
```

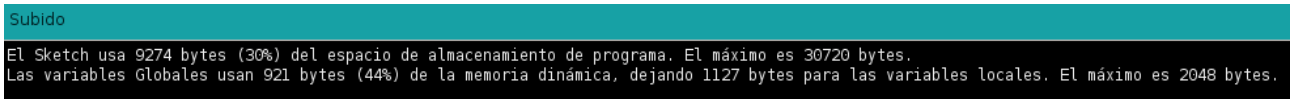
```
////////////////////////////////////////////////////////////////////////
///// Button set analog
////////////////////////////////////////////////////////////////////////

#ifdef BUTTONS_ANALOG

#define BS_ANALOG_WIRES 2
//#define BS_ANALOG_WIRES 3

// Button set pin setup (analog input)
#define BS_ANALOG_PIN A7

// valors d'entrada per cada tecla premuda (0 si le botó no existeix)
#define BS_ANALOG_VALUE_UP 512
#define BS_ANALOG_VALUE_RIGHT 683
#define BS_ANALOG_VALUE_DOWN 769
#define BS_ANALOG_VALUE_LEFT 860
#define BS_ANALOG_VALUE_GO 810
#define BS_ANALOG_VALUE_RESET 0

#endif // BUTTONS_ANALOG



////////////////////////////////////////////////////////////////////////
///// Button set Bluetooth
////////////////////////////////////////////////////////////////////////

#ifdef USE_BLUETOOTH

// Port per defecte de l'Arduino:Serial
#define BLUETOOTH_SERIAL    Serial
//#define BLUETOOTH_SERIAL    Serial1
//#define BLUETOOTH_SERIAL    Serial2
//#define BLUETOOTH_SERIAL    Serial3

#endif // USE_BLUETOOTH
```

7.1 Carrega de codi:

Un cop realitzat aquest pas l'únic que queda es pujar el codi, des de l'aplicació Arduino obrim el codi i verifiquem que es pot realitzar la compilació i que no hi ha cap error de síntesi i a continuació realitzar la pujada de codi, per fer-ho només cal ***verificar*** (Ctrl + O) i a continuació ***subir:***



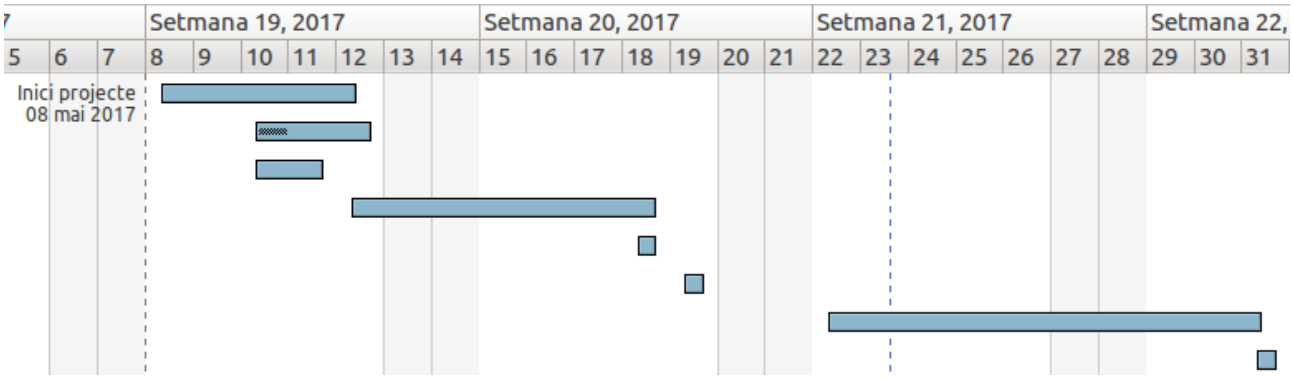
* El codi ha sigut verificat i compilat amb èxit.



* El codi ha sigut pujat amb èxit a la placa.

8. Planificació de la segona part del projecte:

WBS	Nom
1	Investigació logobot
2	Impressió peces 3D
3	Compra de material
4	Proves Arduino
5	Montatge LogoBot
6	Carrega de codi
7	Proves finals
8	Entrega Final



8.1 Tipus de Logobot:

Existeixen diverses versions de **Logobot**:

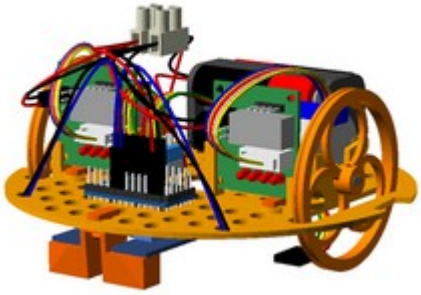
8.2 Logobot Bàsic

Aquesta versió de Logobot no té cap extra com a sensors de so o algun sensor cercador.
Aquesta versió conté unes instruccions prèviament programades així que, executa les instruccions un cop que entra corrent a la placa, simplement segueix els mateixos moviments cop i un altre.

8.3 Logobot IR Seeker

La versió IR *Seeker* té una diferència respecte al Logobot bàsic, aquest robot porta un sensor de moviment, detecta els moviments, les distàncies i executa els passos necessaris. Per exemple si fem moviments amb la mà a una certa distància, el robot detectarà la distància / moviment i s'acostarà, una vegada que s'acosta, si acostem la nostra mà al sensor el que farà és que el robot és allunyi o retrocedeixi.

8.4 Logobot Line Follower



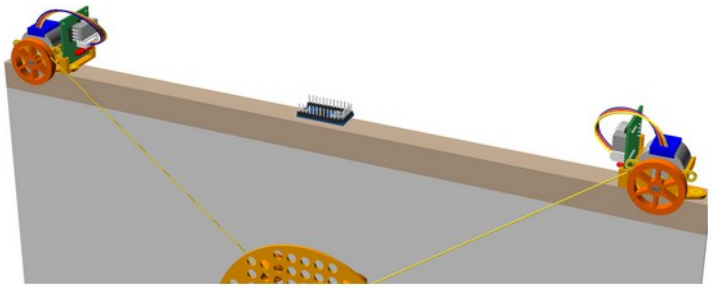
També anomenat segueix-línies, com el propi nom ho indica aquest robot el que fa és, seguir les línies que tracem nosaltres.
Porta un sensor a la part de baix que detecta les línies i segueix les línies fins a arribar al seu destí:

8.5 Logobot Scribbler

Aquest robot s'assembla bastant al Logobot Bàsic, ja que executa les instruccions prèviament programades, però es diferencien que aquest en comptes d'executar passos aleatoris, porta un bolígraf o retolador integrat al mig i escriu paraules segons els moviments que realitza.

8.6 Logobot Polar Graph

Aquesta versió és la que més es diferencia dels altres robots pel que fa al disseny, les peces estan separades uns dels altres, els motors dirigeixen la base a l'alçada i al lloc que toqui. La funció d'aquest robot és dibuixar / escriure en vertical, els motors fan politja amb la base i l'acompanyen per dibuixar sobre del paper, la base portaria un bolígraf i uns punts de suport per poder equilibrar-se sobre del paper.



8.7 Logobot with control

Nosaltres hem decidit anar mes enllà i hem construït un Logobot amb un comandament, **Logobot with control**, ja que no existeix cap versió semblant.
La principal funció del nostre robot és, que li passem una sèrie d'instruccions / moviments per mitjà del teclat numèric i que el robot les vagi executant en el moment que indiquem inici.

8.8 Tecles:

Les tecles per al maneig són les següents:

- Tecla numero **2** Cap endavant.
- Tecla numero **4** cap a l'esquerra.
- Tecla numero **8** cap enrere.
- Tecla numero **6** cap a esquerra.
- Tecla numero **5** per iniciar els moviments.

9. Arduino UNO:

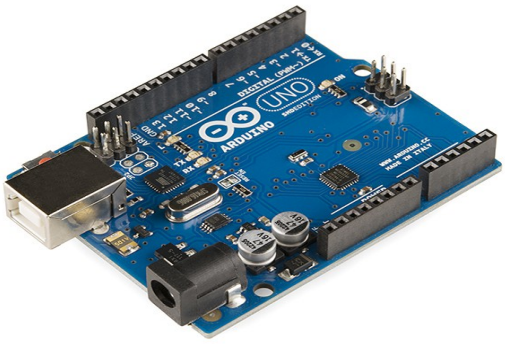
Arduino UNO es l'ultima versió de la placa, existeixen dues variacions, la Arduino UNO convencional i la Arduino UNO SMD. L'única diferencia entre les dues es el tipus de microcontrolador que porten.

· La Arduino UNO convencional porta un microcontrolador ATmega en format DIP
· La Arduino UNO SMD porta un microcontrolador en format SMD

La placa UNO s'utilitza per al desenvolupament del microcontrolador AVR ATmega 328 i pot rebre alimentació mitjançant la connexió de la placa a un ordinador a traves d'un cable USB o be mitjançant una font d'alimentació externa, com podria ser un petit transformador o, per exemple una pila de 9V, aquesta a sigut la nostra elecció.

9.1 Característiques de la placa Arduino UNO:

· ATmega 16U2 programat com un convertidor USB a serie
· 14 contactes de E/S digitals (6 es poden utilitzar com sortides de modulació d'ample impulsos)
· 6 entrades analògiques
· Ressonador ceràmic de 16 MHz
· Connexió USB
· Connector jack d'alimentació
· Connector ICSP
· Botó reset



9.2 Entrades i sortides:

Cadascun dels 14 pins digitals es poden utilitzar com una entrada o sortida. Funcionen a 5V, cada pin pot subministrar fins a 40mA. La intensitat màxima d'entrada també es de 40mA.

Arduino també disposa de 6 pins d'entrada analògics que traslladen les senyals a un convertidor analògic/digital de 10 bits.

9.3 Pins especials d'entrada i sortida:

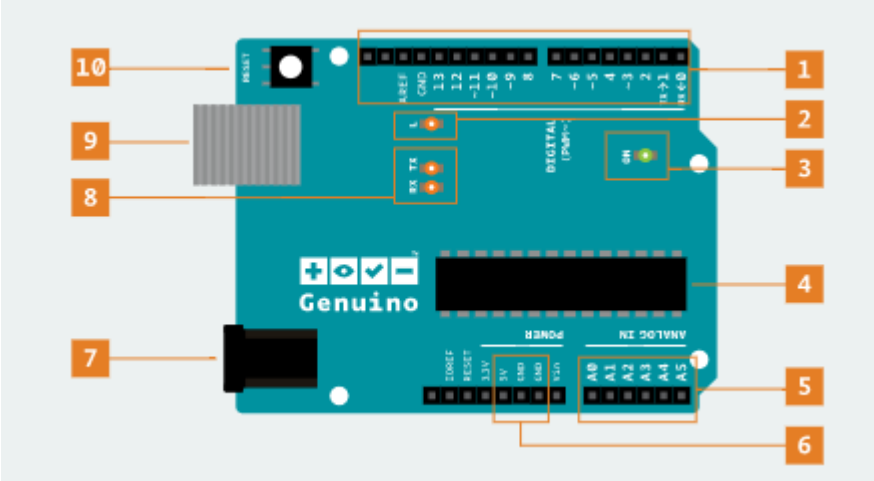
RX i TX: s'utilitzen per transmissions de serie de senyals TTL.

Interrupcions externes: els pins 2 i 3 són configurats per generar una interrupció en l'ATmega. Les interrupcions poden activar-se quan es troba un valor baix en aquestes entrades i amb flancs de pujada o baixada de l'entrada.

SPI: els pins 10, 11, 12 i 13 poden ser utilitzats per dur a terme comunicacions SPI, que permeten traslladar la informació *'full duplex'* en un entorn de *Master/Slave*.

9.4 Parts de la placa:

Aquí podem veure les totes les parts de la placa Arduino UNO convencional anteriorment esmentades.



1. Pins digitals
2. Led del pin 13
3. Led d'energia
4. Microcontrolador ATmega
5. Pins analògics
6. Pins GND i 5V
7. Font d'alimentació externa
8. Led TX i RX
9. Connexió USB
10. Botó de reset

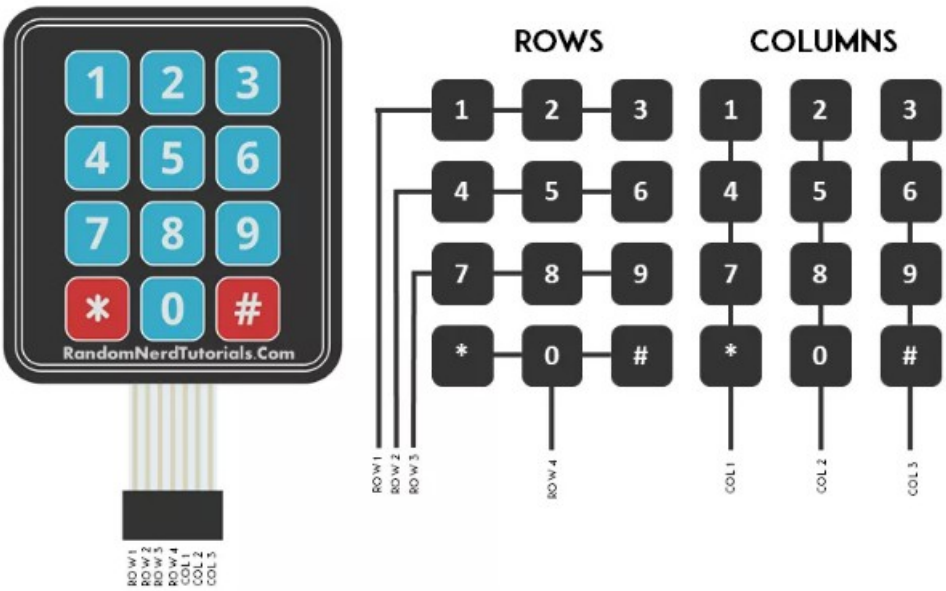
10. Keypad:

El *Keypad* li permet interactuar amb el microcontrolador. Es poden reutilitzar teclats d'antics telèfons o es poden comprar-los en la majoria de les botiga d'electrònica per un preu molt econòmic. En el nostre cas hem decidit comprar el teclat numèric. Hi ha una gran varietat de teclats i mides. Les mides més comuns consisteixen en 3 × 4 i 4 × 4 i es pot obtenir amb teclats amb paraules, lletres i números escrits en les tecles:

*el nostre cas hem escollit la versió de 3 × 4.

10.1 Com funciona:

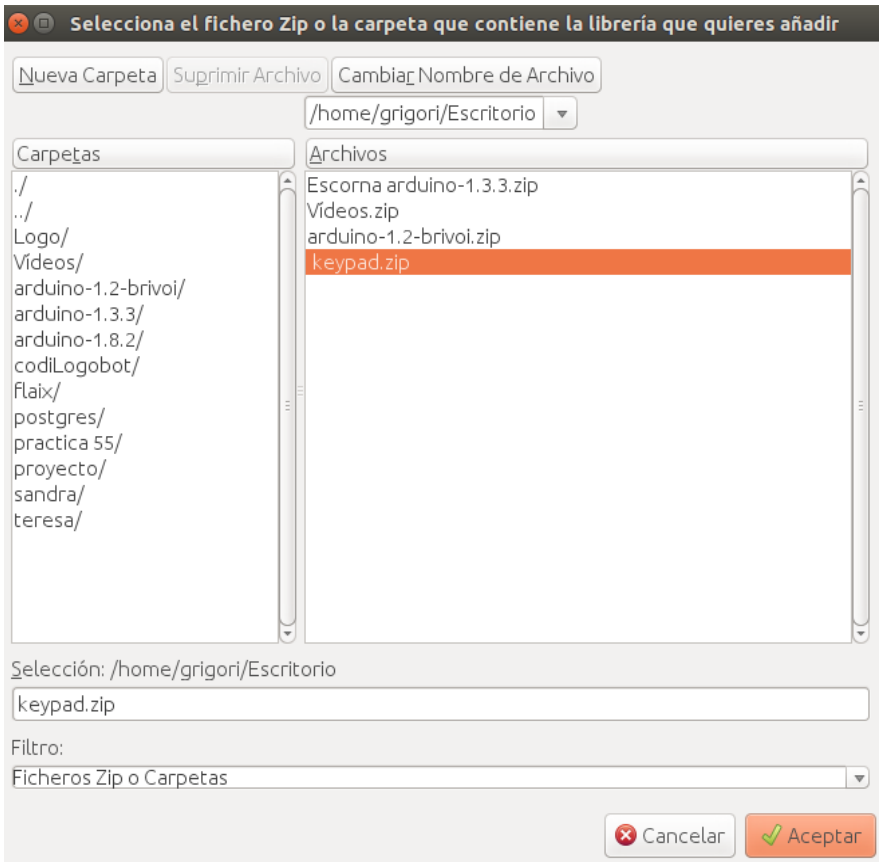
Un teclat de membrana està constituïda per files i columnes. Cada tecla està assignada a una determinada fila i columna.
En un teclat de 12 botons que té 4 files i 3 columnes. La primera clau seria establir un vincle entre la fila 1 i la columna 1.



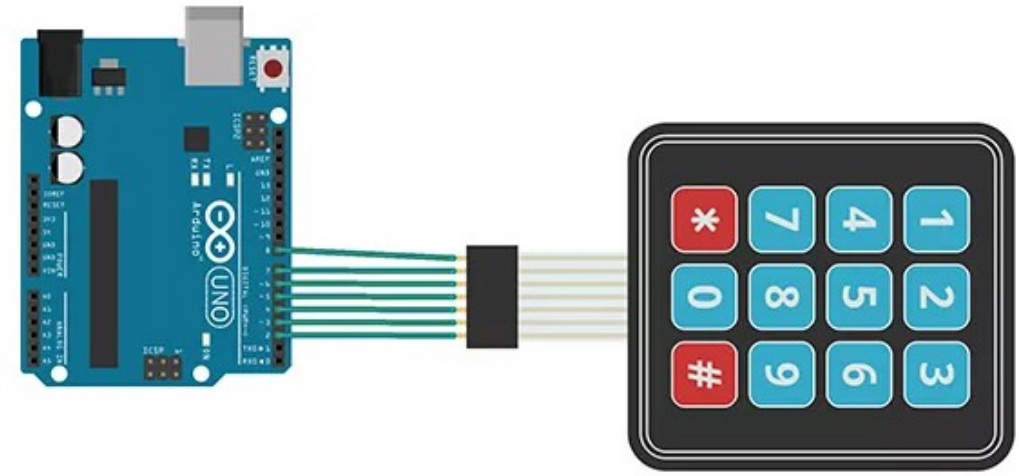
Un cop hem entès com funciona un teclat 3 × 4, el següent pas perquè funcioni aquest teclat és descarregar la llibreria del teclat, aquesta llibreria no ve per defecte en el programa de Arduino, simplement la podem descarregar des de la pàgina oficial de Arduino en format ".zip":

http://playground.arduino.cc/code/keypad#Download

A continuació, un cop hem descarregada la llibreria, per importar-la a nostre Arduino, fem click a **"Programa"** tot seguit **"Incloure llibreria"** després **"Afegir llibreria .zip"** i per acabar, seleccionem la ruta del fitxer descarregat:



Després d'incloure la llibreria, el següent pas seria connectar el *Keypad* als pins del nostre Arduino UNO: nosaltres hem optat per connectar-lo a partir del pin 3 fins al pin 8, deixant lliures els pins 1 y 2.



11. Motors pas a pas *(Steppers)*:

Els motors anomenats S*teppers* o pas a pas, son dispositius econòmics i capaços de girar una quantitat de graus depenent de les indicacions que rebin. Tenen un altre avantatge i es que son relativament precisos parlant del posicionament.

Existeixen dos tipus de motors S*teppers*, els unipolars i els bipolars.

11.1.1 Unipolars:

Per diferenciar un motor bipolar d'un unipolar, només hem d'observar el numero de cables que contenen:

Els motors unipolars contenen 5 o 6 cables de sortida, depenent de la seva connexió interna. Aquest tipus de motor es caracteritza per ser molt simple de controlar. Utilitzen un cable comú a la font d'alimentació i posteriorment es col·loquen les altres línies a terra en un ordre especific per generar cada passa.

Si el motor te 6 cables es perquè cada par de bobines tenen un comú separat de la resta, si pel contrari tenen 5 vol dir que les quatre bobines tenen un pol comú. Una curiositat, es que els motors unipolars de 6 cables es poden utilitzar com un motor bipolar, amb la condició de que els cables comuns s'han de deixar sense utilitzar.

11.1.2 Bipolars:

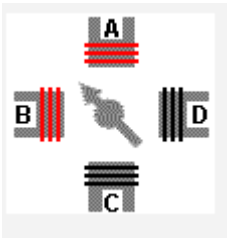
Els motors bipolars generalment tenen 4 cables de sortida, dos per cada bobina. Necessiten certs trucs per ser controlats, degut a que requereixen del canvi de direcció del flux de corrent a traves de les bobines en la seqüencia apropiada per realitzar un moviment.

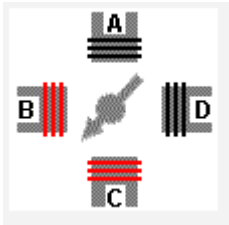
11.2 Moviment:

Per gestionar el tant per cent que volem que tingui moviment, hem d'excitar les bobines del nostre motor i així poder moure'l de manera continua. En el nostre cas, es tracta d'un motor de 4 fases i podem fer que es mogui de 3 maneres diferents:

Excitant dues bobines a l'hora (manera recomanada pel fabricant):

Passos	Bobina A	Bobina B	Bobina C	Bobina D
1	ON	ON	OFF	OFF
2	OFF	ON	ON	OFF
3	OFF	OFF	ON	ON
4	ON	OFF	OFF	ON

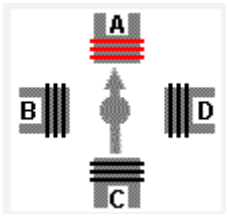


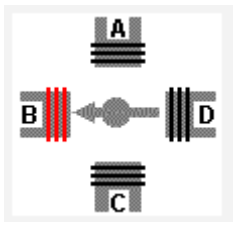


*Amb aquesta combinació tindríem bona velocitat però un alt consum.

Excitant només una bobina cada cop:

Passos	Bobina A	Bobina B	Bobina C	Bobina D
1	ON	OFF	OFF	OFF
2	OFF	ON	OFF	OFF
3	OFF	OFF	ON	OFF
4	OFF	OFF	OFF	ON

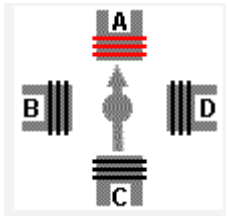


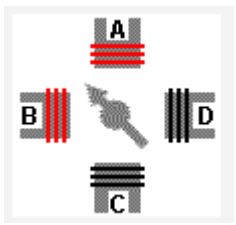


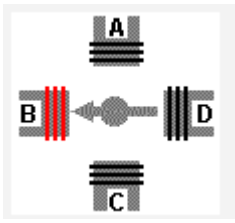
*Baix consum, ja que només activa un bobina.

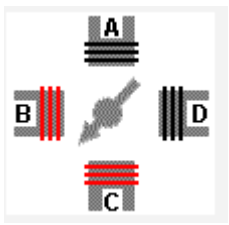
Per últim donem passos a mitges

Passos	Bobina A	Bobina B	Bobina C	Bobina D
1	ON	OFF	OFF	OFF
2	ON	ON	OFF	OFF
3	OFF	ON	OFF	OFF
4	OFF	ON	ON	OFF
5	OFF	OFF	ON	OFF
6	OFF	OFF	ON	ON
7	OFF	OFF	OFF	ON
8	ON	OFF	OFF	ON









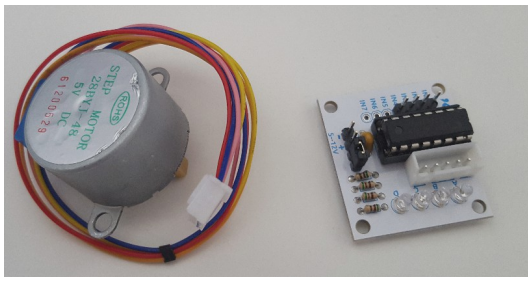
*El moviment es mes suau i lent que amb els mètodes anteriors i el consum intermedi.

11.3 Característiques del motor unipolar:

· Tensió nominal entre 5V i 12V
· 4 fases
· Resistencia 50 Ω
· Consum de 55 mA
· 8 passos per volta
· Reductora d' 1/64

11.4 Connexió motor *steppers:*

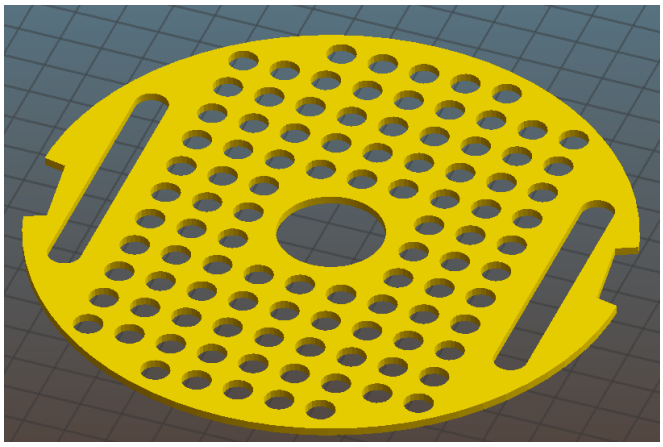
El consum d'un motor *stepper* unipolar es molt baix, tan sols 55 mA, dins del rang que el nostre USB proporciona a la placa Arduino UNO i l'alimentarem mitjançant una placa que li acompanya. Normalment utilitzen un integrat del tipus *ULN2003A* que es un *array* de transistors *Darlington*, que suporta fins a 500 mA i que ja disposa d'un connector pel motor i uns pins (IN1, ..., IN4) per connectar-lo al nostre Arduino.



12. Peces Logobot

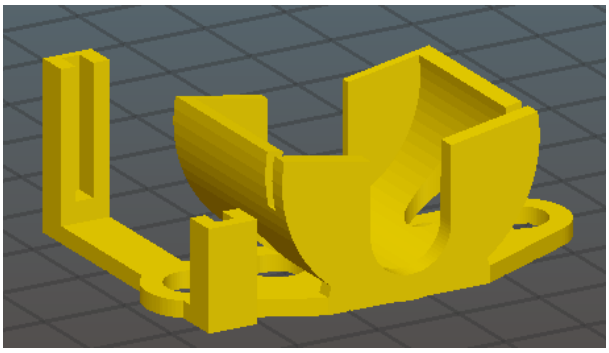
12.1 Base Logobot:

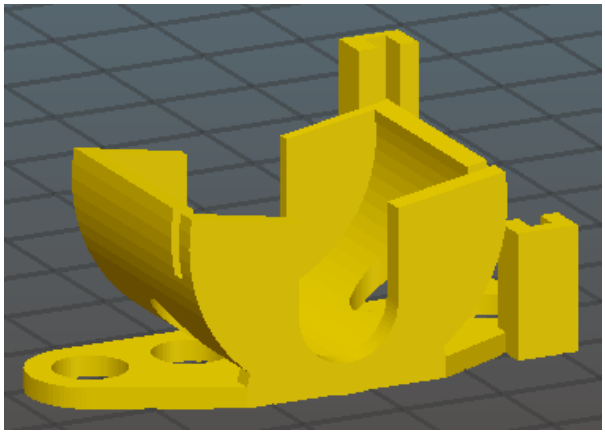
Per començar amb el muntatge del Logobot, necessitarem primer la base, que serà on anirà la placa Arduino, motors, bateria i la resta de peces 3D subjectades. Com podem veure a la imatge, la base té unes esquerdes, on posarem la cúpula per a que tingui una subjecció. El motiu del perquè n'hi han molts forats a la base és per a que quan vulguem subjectar qualsevol peça, tinguem els forats per a poder posar brides, cargols, o pinces. Una de les raons per les quals també estan els forats es per a que el Logobot no tingui tant de pes.



12.2 Suport dels motos:

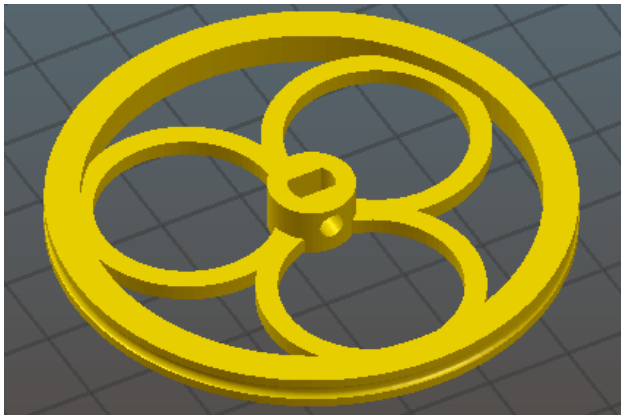
A continuació, la següent peça que necessitarem seran els suports dels motors. Cal dir que cada motor està dissenyat de diferent forma per a que tingui un disseny interior més polit i simètric, per a que els cables no ens donin tants problemes.



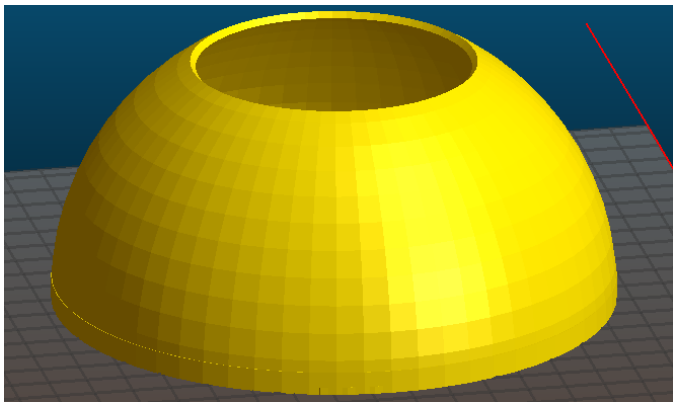


12.3 Rodes:

Una vegada tinguem muntats els motors als seus respectius suports, passarem a muntar les rodes als motors, les quals necessitem que tinguin la mida perfecta, ja que sinó podrien tocar amb la base i el Logobot tindria problemes de mobilitat.



12.4 Cúpula:



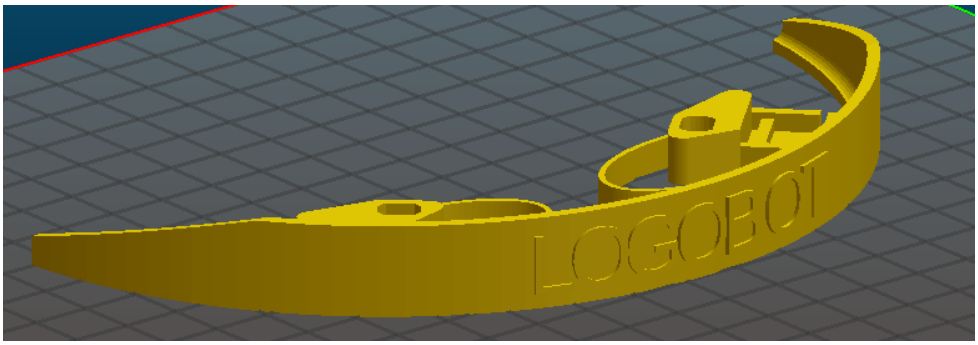
Per a la protecció del nostre Logobot, el més important és la cúpula, ja que el Logobot té molts cables a la vista, i per a l'ús de l'usuari pot ser un problema. Aquesta peça no necessita cap tipus de cargol o brida, perquè té una pestanya la qual hem ajuntat amb l'esquerda de la base. També cap la opció de fer una cúpula cilíndrica, però en el nostre cas és semi-esfèrica.

12.5 Tapa de la cúpula:

Com hem vist, la cúpula té un forat degut a si volem afegir un bolígraf per a traçar una línia o un speaker, per a altres variants de Logobot, però nosaltres hem fet aquesta tapa perquè el nostre Logobot no complirà amb aquestes necessitats.

12.6 Bumper frontal:

Amb el *bumper* frontal, l'objectiu que tenim amb aquesta peça es merament estètica, per a que no es vegin irregularitat per davall de la base i de pas posem el nom del Logobot.



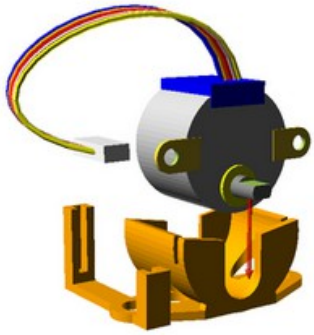
13. Muntatge:

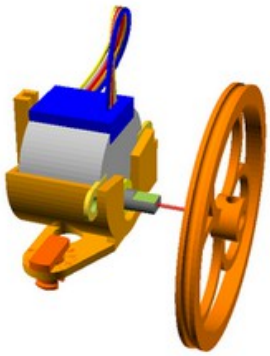
Després d'imprimir les peces del nostre Logobot, hem seguit els següents passos per muntar el nostre robot:

13.1 Muntatge de motors:

Per muntar els motors farem servir el suport imprès amb anterioritat, els passos a seguir són els següents:

Encaixem el motor en el suport de la manera com observem en la imatge.
El suport del motor no solament serveix per a subjectar els motors, també porta un suport per encaixar la placa que controla el motor, més endavant veurem com encaixar-ho.
Farem això amb els dos motors, amb el dret i amb l'esquerre.

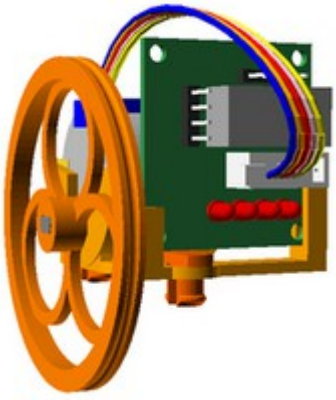




El següent pas després d'encaixar els motors en el seu suport seria encaixar les rodes en el motor:

A l'encaixar les rodes hem tingut alguns problemes per la forma que s'havien imprès les peces, per encaixar les rodes hem hagut de llimar per dins, per poder adaptar a la mida necessari.

El següent pas, una vegada que tinguem el motor amb la roda en el seu suport, seria afegir el controlador del motor.
El controlador es col·loca en la part posterior del suport del motor de la següent manera:



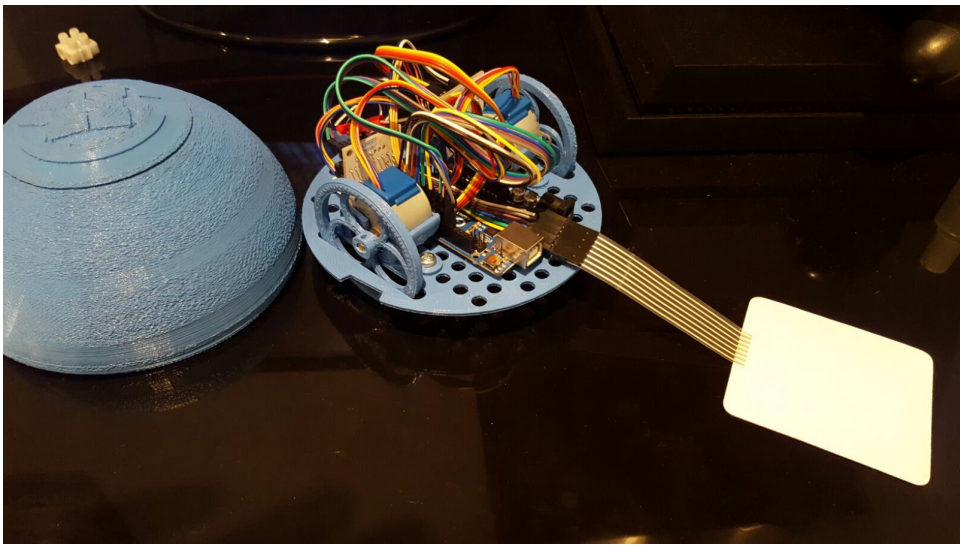
Seguint aquests passos muntarem les rodes amb els seus motors i les seves plaques respectivament al seu sòcol.

13.2 Unió amb la base:

Com veiem en la imatge la placa Arduino UNO va en el centre de la base i per darrere tenim la pila de 9V, als costats tenim les rodes amb els seus suports. Per connectar les rodes i donar-los corrent hem fet servir dos regletes.

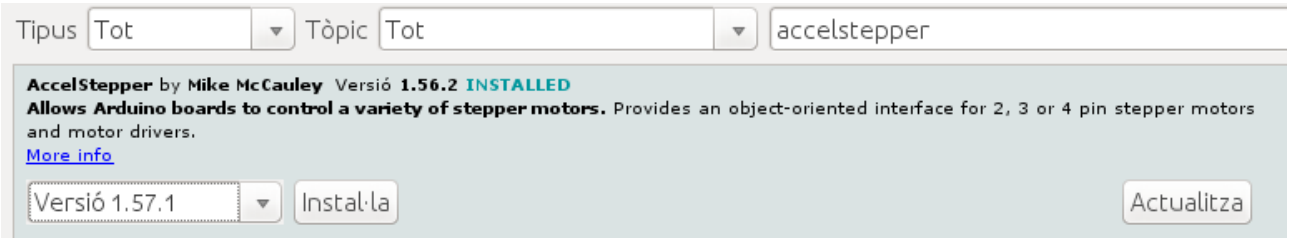
Per unir les rodes amb els seus suports a la base, hem decidit utilitzar 4 cargols i 4 femelles, en comptes d'usar peces fets de plàstic, amb els cargols i les femelles un dels avantatges que tenim és que, són bastant més resistents i més fàcils a l'hora de muntar.

Per unir la placa Arduino i la pila a la base hem decidit fer servir velcro, així podem extreure la pila i la placa amb major facilitat en el cas que sigui necessari.
Per últim al costat de la pila hem introduït un petit interruptor per encendre o apagar el robot, per unir l'interruptor a la base hem fos plàstic amb l'ajuda d'un soldador al voltant de la base, així aconseguim tenir una major subjecció a la base.



14. Codi Logobot:

Per moure dos motors com si fossin només un, hem d'afegir la llibreria *AccelStepper*, per instal·lar-la solament fem click en **"Programa"** tot seguit **"Incloure llibreria"** i després **"Gestiona les llibreries"**.
Ens sortira una pàgina com la següent on només hem d'escriure el nom de la llibreria desitjada i fer click a instal·lar.




```
//////////////////////////////////////////
/////////////// LIBRERIAS ///////////////
//////////////////////////////////////////

#include <Keypad.h>
#include <AccelStepper.h>
#include <MultiStepper.h>
#define STEP 8
```

```
//////////////////////////////////////////////////
/////////////// MOTORES STEPPER ///////////////
//////////////////////////////////////////////////

// Definimos que pin utilizara cada motor stepper

#define pinMotor1 10 //Motor nº1  IN1
#define pinMotor2 11 //Motor nº1  IN3
#define pinMotor3 12 //Motor nº1  IN2
#define pinMotor4 13 //Motor nº1  IN4

#define pinMotor5 A1 //Motor nº2  IN1
#define pinMotor6 A2 //Motor nº2  IN3
#define pinMotor7 A3 //Motor nº2  IN2
#define pinMotor8 A4 //Motor nº2  IN4

// Este array determina la posicion donde se guarda el stepper en el objeto MultiStepper

long position [2];
```

```
  ////////////////////////////////////////
  ///////////////// KEYPAD ///////////////
  ////////////////////////////////////////

  // Filas y columnas que tiene nuestro keypad

  const byte filas = 4;
  const byte columnas = 3;

  // Guardamos en un array los pines de la placa que utilizara el keypad

  byte pinFilas[] = { 8, 7, 6, 5 };
  byte pinColumnas[] = { 4, 3, 2 };
  byte i = 0;

  // Guardamos los valores que tiene el keypad por cada boton pulsado

  char botones [filas] [columnas] =
  {
    {'1','2','3'},
    {'4','5','6'},
    {'7','8','9'},
    {'*','0','#'}
  };

  // Definimos el array donde se guardan el maximo de movimientos(120)

  char movimientos [120];


 ////////////////////////////////////////////////
 ///////////////// CONSTRUCTORES ////////////////
 ////////////////////////////////////////////////

 Keypad keypad1 = Keypad(makeKeymap(botones), pinFilas, pinColumnas, filas, columnas);
 AccelStepper stepper1(8,10,12,11,13);
 AccelStepper stepper2(8,A1,A3,A2,A4);
 MultiStepper multi;

////////////////////////////////////////
/////////////// FUNCIONES //////////////
////////////////////////////////////////

// Funcion principal

void setup() {
  // Inicializamos el puerto serie para realizar las pruebas con el keypad
  Serial.begin(9600);
  // Velocidad maxima que alcanzara nuestro motor
  stepper1.setMaxSpeed(1000);
  // Velocidad a la que acelerara nuestro motor
  stepper1.setAcceleration(100);
  // Velocidad con la que inicia nuestro motor
  stepper1.setSpeed(100);
  stepper2.setMaxSpeed(1000);
  stepper2.setAcceleration(100);
  stepper2.setSpeed(100);
  // Añadimos ambos motores al constructor MultiStepper
  multi.addStepper(stepper1);
  multi.addStepper(stepper2);

}
```

```
 // Funcion ir hacia delante

 void delante(char movimientos [], int x){
   movimientos[x] = ' ';
   // Cuantos pasos dan los motores
   position[0] = position[0] +3700;
   position[1] = position[1] -3700;
   multi.moveTo(position);
   multi.runSpeedToPosition();
 }

 // Funcion ir hacia atras

 void atras(char movimientos [], int x){
   movimientos[x] = ' ';
   position[0] = position[0] -3700;
   position[1] = position[1] +3700;
   multi.moveTo(position);
   multi.runSpeedToPosition();
 }

 // Funcion girar hacia la derecha

 void derecha(char movimientos [], int x){
   movimientos[x] = ' ';
   //position[0] = position[0] -4095;
   position[1] = position[1] -3895;
   multi.moveTo(position);
   multi.runSpeedToPosition();
 }

 // Funcion girar hacia la izquierda

 void izquierda(char movimientos [], int x){
   movimientos[x] = ' ';
   position[0] = position[0] +3895;
   //position[1] = position[1] +4095;
   multi.moveTo(position);
   multi.runSpeedToPosition();
 }

// Funcion ejecutar movimientos

void mover(char movimientos[], int x){

  byte numElemArray = sizeof(movimientos) / sizeof(char);

    // Añadimos cada movimiento introducido, hasta llegar al maximo
    for(int x = 0; x < 120; x++){
      if(movimientos[x] == '2') { delante(movimientos,x); }
      if(movimientos[x] == '8') { atras(movimientos,x); }
      if(movimientos[x] == '6') { derecha(movimientos,x); }
      if(movimientos[x] == '4') { izquierda(movimientos,x); }
    }
    // Reiniciamos el array para volver a introducir nuevos movimientos
    i = 0;
}
```

```
void loop(){

  // Almacenamos en la variable el valor de la tecla pulsada

  char key = keypad1.getKey();

  int x;

  // Escribimos en pantalla la tecla pulsada

  if ( key != NO_KEY){
    Serial.println(key);
  }

  if ( key == '2'){
    movimientos[i] = key;
      i = i+1;
  }
  if ( key == '4'){
    movimientos[i] = key;
      i = i+1;
  }
  if ( key == '6'){
    movimientos[i] = key;
      i = i+1;
  }
  if (key == '8'){
    movimientos[i] = key;
      i = i+1;
  }
  if (key == '5'){
    mover(movimientos,x);
  }
}
```

15. Coses a tenir en compte:

15.1 Bateries

El consum de les bateries es bastant alt, hem d'anar amb compte perquè al esgotar-se el suficient el robot no realitza els moviments al 100%.

15.2 Preus

Els preus varien considerablement segons si els comprem via internet o via tenda física.

15.2.1 Escornabot:

Les peces d'aquest bot s'han comprat via internet, el preu sumant el transport i els impostos arriba al total de 52,40€, es poden adquirir mes barates però amb la condició de soldar les peces després (resistències, botons, etc), al nostre cas hem optat per demanar les peces soldades.

15.2.2 Logobot:

Les peces del Logobot las hem comprat via tenda física.

Cables plans → **6,39€**

Motors *steppers* 5V → **9.92€**

Plaques controladores → **4.13€**

Connector 2.1mm → **1.24€**

Pila de 9V → **2€**

Commutador 1C → **1.15€**

Keypad 3x4 → **3.31€**

Placa Arduino UNO →**25€**

TOTAL ---------- **53.15€**

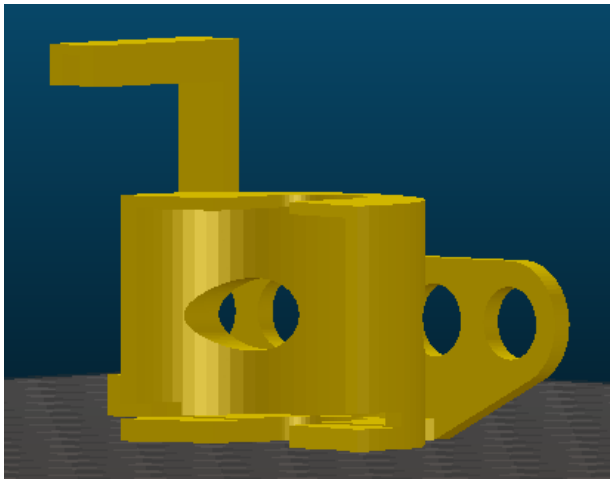
15.3 App Mòbil

Vam intentar fer una aplicació per a Android amb l'objectiu de poder moure el Escornabot a distància des del nostre *Smartphone*, però per falta de temps no es va poder realitzar del tot aquesta part del projecte, vam decidir donar prioritat a altres parts mes importants.

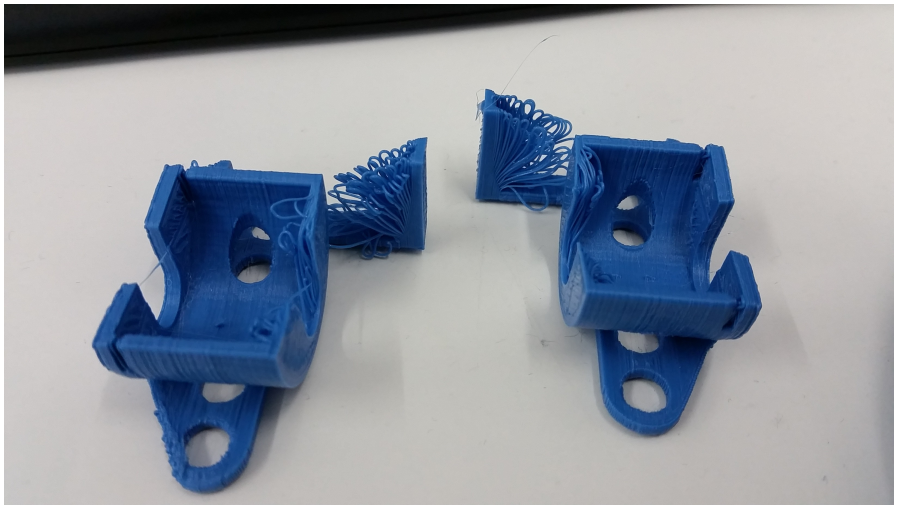
16. Gestió d’errors:

16.1 Errors Logobot

Amb el Logobot vam tenir una sèrie de problemes, a l’hora d’imprimir les peces 3d dels motors, ja que quan importàvem la peça 3d al Slic3r, el resultat que veiem era aquest:

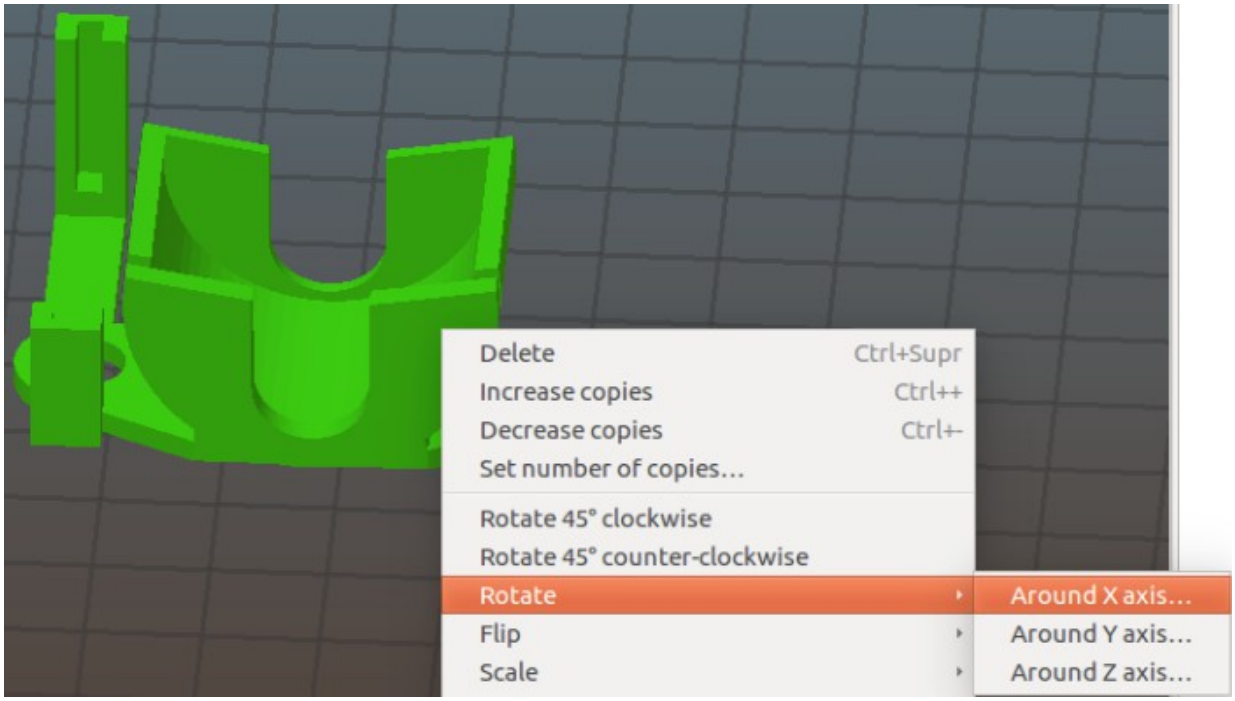


Una vegada vam passar aquesta peça 3d a la impressora, una vegada va finalitzar la seva impressió, ens vam trobar amb el següent resultat:

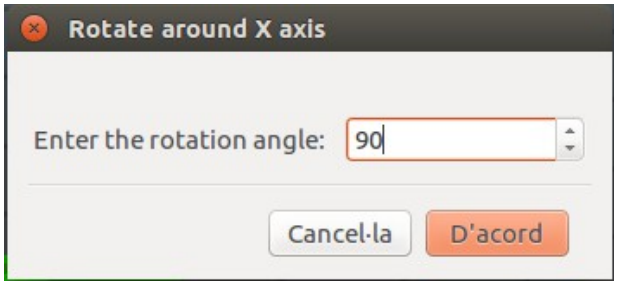


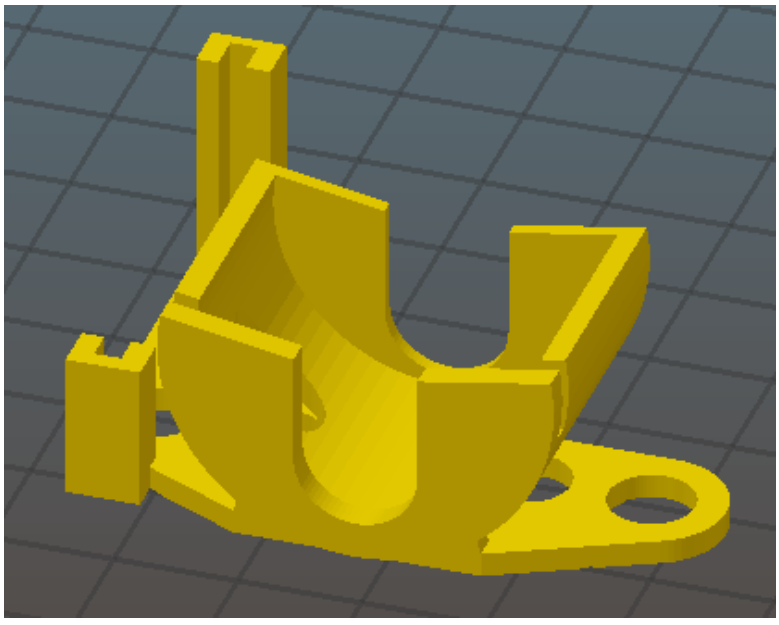
Com es pot veure a la imatge, la impressora 3d quan tenia que imprimir zones on la peça no tenia cap base per a poder imprimir, el filament va caure, per tant, vam tenir que modificar la base de la peça per a que quan l’imprimim no donés cap problema. Per poder realitzar això, vam tenir que anar-nos al programa Slic3r i investigar una miqueta per a poder realitzar la modificació de la peça per a poder posar la peça en una zona en que no doni problemes a l’hora d’imprimir peces flotants.

Una vegada estem al Slic3r, carreguem la peça 3d i fem clic dret a la mateixa, ***Rotate > Around X axis*** i se'ns obrirà una finestra en la qual el programa ens preguntarà quants graus vols girar la peça 3d sobre el paràmetre X.

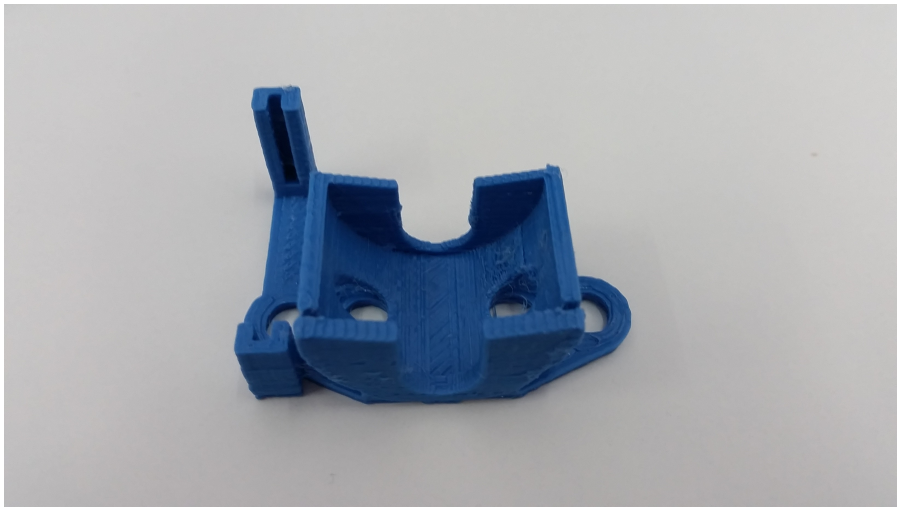


Una vegada se'ns obre la finestra, el que nosaltres especificarem seran els graus que volem que giri, en el nostre cas, només volem que giri 90 graus, per a que la peça quedi de la següent manera:

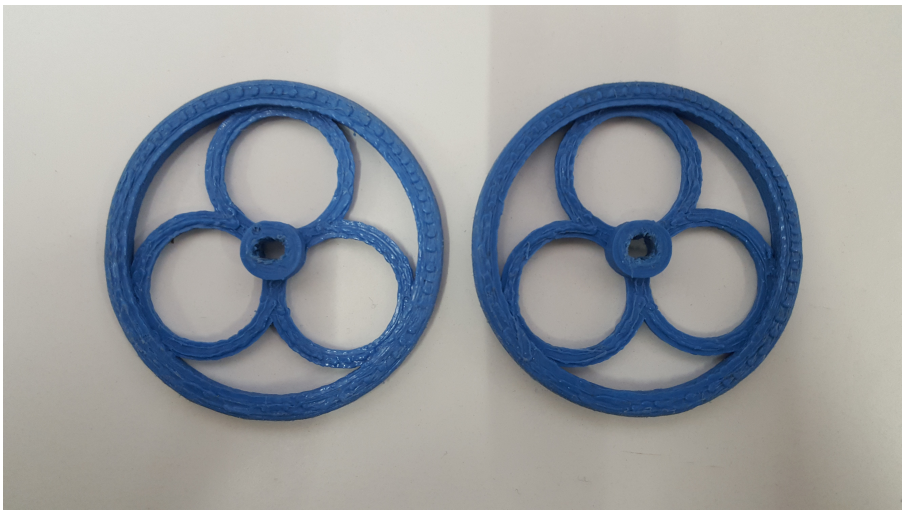




Una vegada vam portar la peça 3d a la impressora, el resultat que vam tenir finalment va ser aquest:



A part dels problemes anteriors, vam tenir problemes amb les rodes i la cúpula, per la seva grandària original en les rodes. El disseny original de les rodes és massa gran i freguen amb la part interior de la cúpula i a l'hora de rodar es quedava encallat. La solució ha estat dissenyar dues rodes noves però mes petites, concretament 2 mil·límetres, d'aquesta manera podem posar gomes per a que el robot no derrapi i rellisqui amb més facilitat.



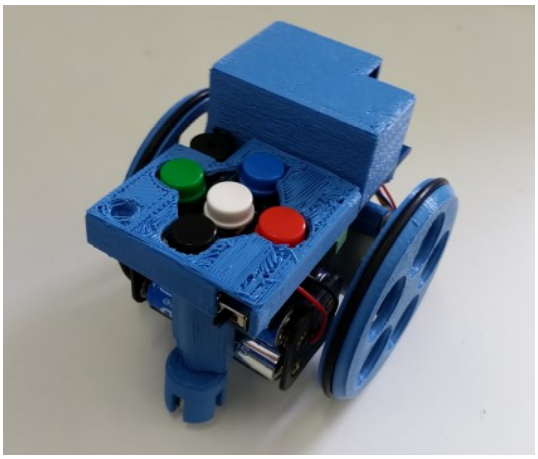
16.2 Errors Escornabot:

Com a extra, vam crear unes peces per a utilitzar-les a mode de carcassa. Després de fer una espècie de test amb cada peça, al cap d'unes 2-3 impressions, vam tenir les peces ja enllestides.

Un dels contratemps a l'hora d'obtenir les peces va ser que no les podíem fer totes d'una sola peça ja que tenen un desnivell i ho vam tenir que fer en dues peces.

Un altre problema, va ser, que a l'hora de la impressió, les peces no quedaven bé totalment, tenien molts desperfectes i havien problemes a l'hora de la seva estabilitat i duresa.

Una vegada totes les peces van estar enllestides i col·locades, el resultat va ser el següent:



17. Bibliografia:

17.1 Curs Pràctic de Formació sobre Arduino:

Autor: Oscar Torrente Artero.
Nº de pagines: 588.
Enquadernació: Tapa blanda
Editorial: RC LIBROS (SC LIBRO)
Llengua: CASTELLÀ
ISBN: 9788494072505

https://www.wikipedia.org

17.2 Escornabot:

https://github.com/escornabot/3dmodel/blob/master/Brivoi/README.md

https://groups.google.com/group/escornabot_users/attach/4f3da8e6cc19b/escornacpu2-bracket_motors_down.stl?part=0.1&authuser=0&view=1

http://www.pipop.es/escornabot-soldado

17.3 Logobot:

https://github.com/swindonmakers/LogoBot/tree/master/hardware/printedparts/stl

http://rawgit.com/swindonmakers/LogoBot/master/hardware/docs/index.htm

https://github.com/swindonmakers/snhack.github.io/wiki/LogoBot