Treball final de Màster

Master Universitari en Enginyeria Industrial

Disseny, programació i implementació d'un robot de dibuix amb Arduino

Autor:
Josep Maria Martí i Elias

Tutor:
Manel Velasco Garcia

Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial (ESAII)
Escola Tècnica Superior de Enginyeria Industrial de Barcelona

7 de juny de 2017

Resum

La idea principal d'aquest projecte és dissenyar, programar i construir un robot capaç de dibuixar sobre qualsevol paper, de qualsevol dimensió, la figura o traçada que l'usuari desitgi.

Per fer-ho, es vol dissenyar un robot de dues rodes motrius controlades per dos motors pas a pas, una roda boja per fer estable l'estructura, un suport per un punter (ja sigui un llapis, un retolador o qualsevol objecte que serveixi per dibuixar) situat al centre entre les dues rodes, una estructura de plàstic impresa amb tècniques d'impressió 3D i un microcontrolador *Arduino Uno* per tal de controlar tot el sistema i realitzar les comunicacions amb l'ordinador de l'usuari. Tota la programació serà realitzada a través de la plataforma de software lliure d'*Arduino*.

El primer objectiu serà trobar la manera de controlar els dos motors simultàniament per tal que segueixin la trajectòria desitjada. És molt important que aquest control sigui molt acurat ja que inicialment la posició del robot estarà controlada en llaç obert i per tant no es podran corregir el errors comesos i s'acumularan creant distorsió entre la trajectòria desitjada i la descrita pel propi robot. És per aquest motiu que s'utilitzaran motors pas a pas ja que en tot moment es pot saber la seva posició que vindrà donada pels passos produïts.

Paral·lelament s'ha de dissenyar l'estructura del robot per així estudiar el seu model cinemàtic i conèixer el seu moviment per tal de traduir-ho a impulsos del motor i així controlar el posicionament.

Un cop conegut el funcionament del robot i el seu moviment cal aconseguir dissenyar un lector de trajectòries que pugui comunicar-se amb el robot per tal que aquest realitzi el moviment desitjat sobre el paper. La idea inicial és aconseguir traduir la trajectòria desitjada a *G-code* amb el software lliure de dibuix i edició d'imatges *Inkscape*. A partir d'aquest codi, s'ha de decidir quina serà la manera adequada per comunicar l'ordinador amb l'*Arduino* per transmetre cada una de les ordres del codi i realitzar-les. Hi ha diferents opcions com poden ser mòduls *wifi* o *Bluetooth* o fins i tot a partir de memòries externes del tipus SD.

Més endavant i si el temps ho permet, es podrà dissenyar alguna manera per tancar el llaç de control de posició i així corregir errors que es puguin produir i fer el sistema més precís.

Agraïments

Índex

Índex de figures

Índex de taules

Símbols

TBR	Tritium Breeding Rate
ITER	Nuclear fusion reactor of first generation being building nowadays
WCSB	Water Coolant Solid Breeder
MCNP	Monte Carlo N-Particles code
FEEL	Fusion Energy Engineering Laboratory, ETSEIB, UPC
AINA	Fusion reactor safety code developed at the FEEL
DEMO	Nuclear fusion reactor of second generation being planned nowadays
IFERC	International Fusion Energy Research Center
Tally	Data results specification of the MCNP

Capítol 1

Introducció

Capítol 2

Funcionament

2.1 Motor pas a pas

2.1.1 Per què un motor pas a pas?

Els motors pas a pas o "steppers", són els motors ideals per controlar el moviment de les rodes del robot. En primer lloc ofereixen la possibilitat de rotar en les dues direccions, fet imprescindible per poder realitzar qualsevol trajectòria. La seva precisió angular és molt elevada, fet que permet coneixer en tot moment i amb exactitud quants graus ha girat el motor i per tant el moviment que han realitzat les rodes i, en definitiva, el robot. És un motor fàcil de controlar, amb el qual es pot minimitzar l'error comés gràcies a la seva excel·lent precisió, és pot controlar la seva velocitat i, per acabar, un altre avantatge que presenta és que quan el motor està parat, si rep corrent, ofereix una parell de resitència contra la rotació de l'eix, fet que evita que giri quan no ho ha de fer.

2.1.2 Funcionament

El funcionament del motor pas a pas es bassa en la transformació d'impulsos elèctrics en "steps" o passos del motor, que són rotacions angulars exactes de l'eix. Típicament poden presentar 12, 24, 72, 144, 180 o 200 passos per volta i això significa que per cada impuls elèctric la seva rotació és de 30, 15, 5, 2.5, 2 o 1.8 graus respectivament.

Presenta dues parts principals, l'estator, que esta format per bobines amb les quals s'indueix un camp magnètic al fer-hi passar corrent elèctric, i el rotor, que consisteix essencialment en un imant permanent rotatori, atret pel camp magnètic creat per l'estator.

Al fer passar corrent per una de les bobines de l'estator es crea una polaritat i per tant s'orienta el rotor en funció d'aquesta polaritat, ja que sempre buscarà l'estabilitat magnètica entre els seus pols. Quan canvia l'excitació de l'estator, canvien els pols magnètics i per tant el rotor es

reorienta. D'aquesta manera, al anar variant el pas del corrent per les bobines, es provoca un moviment circular del rotor molt precís, del qual es pot controlar la velocitat i el sentit de gir.

2.1.3 Tipus de motors pas a pas

En funció de la connexió i la forma d'excitació de les bobines, es pot diferencia entre dos tipus de motors pas a pas diferents, els unipolars i els bipolars.

2.1.3.1 Unipolars

El corrent que passa per les bobines de l'estator d'un motor pas a pas unipolar sempre va en el mateix sentit. Normalment tenen 5 o 6 cables de sortida, 6 si el cable d'alimentació comú de cada par de bobines va per separat i 5 si va junt. És més senzill de controlar ja que sempre es té el cable d'alimentació i cada pas es realitza al connectar a terra els altres cables amb un ordre determinat.

2.1.3.2 Bipolars

A diferencia dels unipolars aquests motors només presenten 4 cables de sortida, no tenen un comú d'alimentació. El seu funcionament es bassa en el canvi del sentit de circulació del corrent per les bobines en funció de la tensió, fet que provoca una diferència de polaritat que fa girar el rotor. Per tal de controlar-lo, és necessari utilitzar dos ponts H per poder invertir el sentit del corrent que passa per les dues bobines.

2.1.4 Modes de funcionamet

Existeixen diferents seqüencies per aconseguir el moviment del motor, que dependrà de quines bobines estan excitades, en el cas dels motors unipolars, i en la direcció del corrent en els bipolars.

2.1.4.1 Seqüencia d'una fase

Aquesta seqüencia consta de l'excitació d'una sola bobina fet que provoca un pas sencer del motor com es mostra a la imatge.

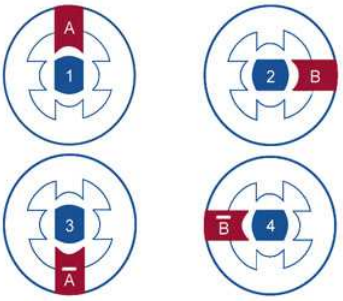

FIGURA 2.1: Sequencia d'una fase.

La seva expressió en forma de taula és la següent:

Pas	A	B	-A	-B
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1

TAULA 2.1: Seqüencia d' fase.

2.1.4.2 Seqüencia de dos fases

Igual que a la seqüencia anterior, s'aconsegueix un pas per cada un dels estats de la sèrie, però la gran diferència és que per aconseguir-ho s'exciten dues bobines alhora, fet que provoca que la força d'atracció augmenti i aconsegueixi un millor parell de resistència.

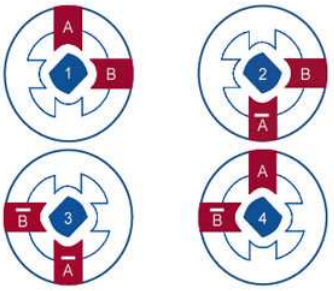

FIGURA 2.2: Sequencia de dos fases.

La seva expressió en forma de taula és la següent:

Pas	A	B	-A	-B
1	1	1	0	0
2	0	1	1	0
3	0	0	1	1
4	1	0	0	1

TAULA 2.2: Seqüencia de dos fases.

2.1.4.3 Seqüencia half step

Amb una combinació de les dues anteriors, aconseguim que per cada estat de la sèrie el motor realitzi un gir de mig pas, i per tant aconsegueixi un moviment més petit. El seu funcionament es basa en l'excitació d'una sola bobina i de dues alhora de forma seqüencial com es mostra a la figura.

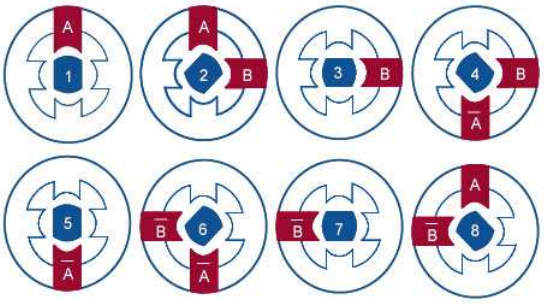

FIGURA 2.3: Sequencia half step.

La seva taula de funcionament és la següent:

Pas	A	B	-A	-B
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1
5	1	1	0	0
6	0	1	1	0
7	0	0	1	1
8	1	0	0	1

TAULA 2.3: Seqüencia mig pas (half step).

2.1.4.4 Microstepping

L'objectiu del microstepping és aconseguir un moviment de rotació més suau, reduint els moviments bruscos entre passos del motor, augmentant el nombre de passos. Per aconseguir-ho s'utilitza un controlador, en aquest cas el driver explicat a l'apartat (2.2), que idealment envia corrent en ones sinusoïdals enlloc d'impulsos. Es creen dos ones sinusoïdals desfasades 90º entre elles, una per cada bobina del motor, aconseguit així que quan a una bobina el corrent augmenta a l'altre decreix de la mateixa manera, creant un moviment perfecte del motor.

A la realitat aquestes ones no són sinusoïdals perfectes, sinó que són ones esglaonades. Això provoca un augment de passos per volta del motor, en aquest cas es pot aconseguir multiplicar els passos per 2, 4 o 8 assolint un màxim de 1600 passos per volta. A la figura (2.4) es pot observar un exemple d'ona encarregada del microstepping.

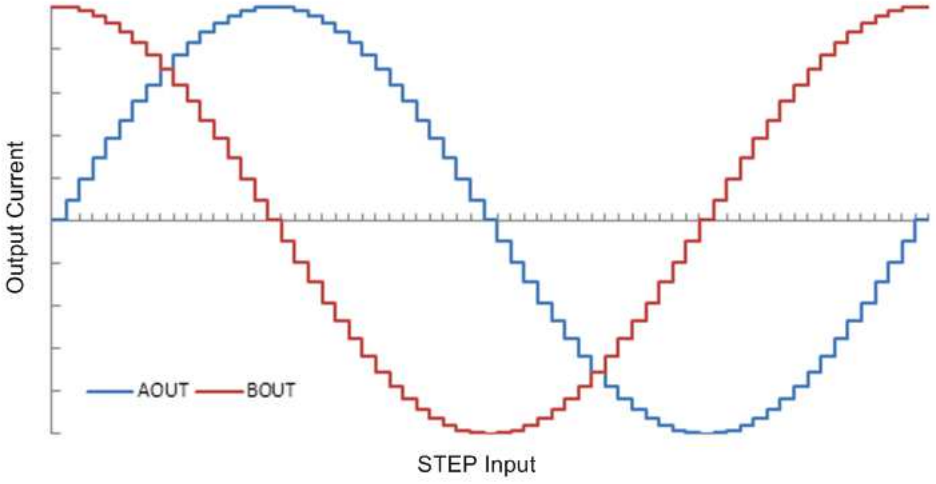

FIGURA 2.4: Ona sinusoïdal del microstepping.

2.1.5 Alternatives

Alhora de triar quins serien els motors adequats pel moviment del robot s'ha estudiat 3 opcions valides per fer-ho, el motor pas a pas, el motor de corrent continu i el servo-motor de rotació continua. S'ha escollit el motor pas a pas ja que el control de posició és bàsic pel funcionament del robot, i és l'únic que el pot assegurar sense la utilització d'un encoder i és el més fàcil de controlar. En els casos del motor de corrent continu i el servomotor de rotació continua, el seu funcionament es bassa en el control de la velocitat de rotació de l'eix segons la tensió o els impulsos del senyal de control respectivament, i per tant controlar la posició és molt més complicat.

2.1.6 Motor escollit: Nema 14

Per escollir un motor pas a pas s'han estudiat els diferents models de la família NEMA i s'ha escollit entre el NEMA14 i el NEMA17 ja que són els que presenten les prestacions idònies pel moviment que es vol realitzar. Finalment la decisió s'ha decantat cap al NEMA14 ja que és més petit i té menys pes, la qual cosa permet adaptar-lo millor al prototip i realitzar un model amb unes dimensions més petites i òptimes.

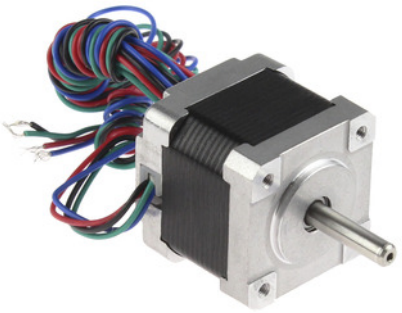

FIGURA 2.5: Motor NEMA14.

El NEMA14 és un motor pas a pas bipolar amb les següents característiques:

Tensió nominal/fase	2,7 V
Corrent nominal/fase	1.0 A
Resistència/fase	2,7 Ω
Par de retenció	1,4 Kg·cm
Número de cables	4
Pes	0,18 Kg
Àngle de pas	1,8º
Passos per volta	200

TAULA 2.4: Característiques del motor NEMA14.

2.2 Driver controlador del motor

2.2.1 Controlador: Easy Driver A3967

Els motors pas a pas bipolars necessiten un canvi de sentit en el flux de corrent de les bobines per aconseguir així la seqüencia de fases desitjada per realitzar el moviment. A part d'això, com que els motors estan controlats per un microcontrolador Arduino UNO, s'ha de controlar la intensitat del corrent ja que podria fer malbé la placa. És per aquesta raó que s'ha decidit utilitzar un controlador que es basa en dos ponts H, un per cada bobina del motor, un dispositiu que suporta el flux bidireccional de corrent inversa i permet canviar-lo de sentit.

Un pont H està format per 4 interruptors connectats com a la figura (2.6)que permet fer passar el corrent per la bobina en un sentit o en un altre en funció de quina combinació d'interruptors s'utilitza, evitant així treballar amb voltatges negatius.

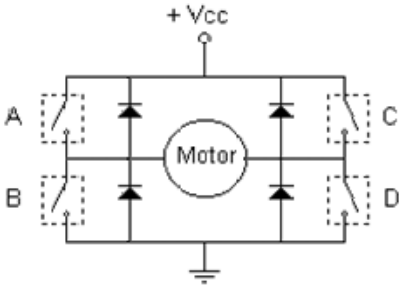

FIGURA 2.6: Connexió en pont H.

El seu funcionament és simple, i funciona per parelles d'interruptors que poden ser transistors bipolars, jfets, mosftes o alguna combinació entre ells. Si es tanquen els interruptors A i D el corrent circula en un sentit de la bobina i si al contrari els que es tanquen són B i C el corrent circula en l'altre sentit. Les combinacions A i B o C i D són perilloses pel circuit ja que crearien un curtcircuit. Els díodes són de protecció, ja que l'efecte inductiu de les bobines podria cremar el circuit un cop desconnectat.

2.2.1.1 Easy Driver A3967

Per facilitar el control dels motors pas a pas utilitzats, s'ha decidit utilitzar dos controladors Easy Driver A3967 que facilitarà el control i la programació dels motors i actuara com etapa de potència per no sobrecarregar l'Arduino amb la potencia requerida pel motor.

Com ja s'ha explicat, els motors pas a pas funcionen amb el canvi d'excitació de les bobines que el conformen, i cada parell de bobines te el seu parell de cables per fer-ho. La idea principal d'aquest driver o controlador és facilitar la feina i encarregar-se de l'alimentació dels motors, permetent així a l'usuari controlar el motor a través de només dos pins digitals de l'Arduino, l'STEP, que defineix un pas per cada pujada del pin i el DIR, que defineix el sentit de rotació del motor.

A la següent imatge es mostra la distribució de les diferents entrades amb les que compta el controlador:

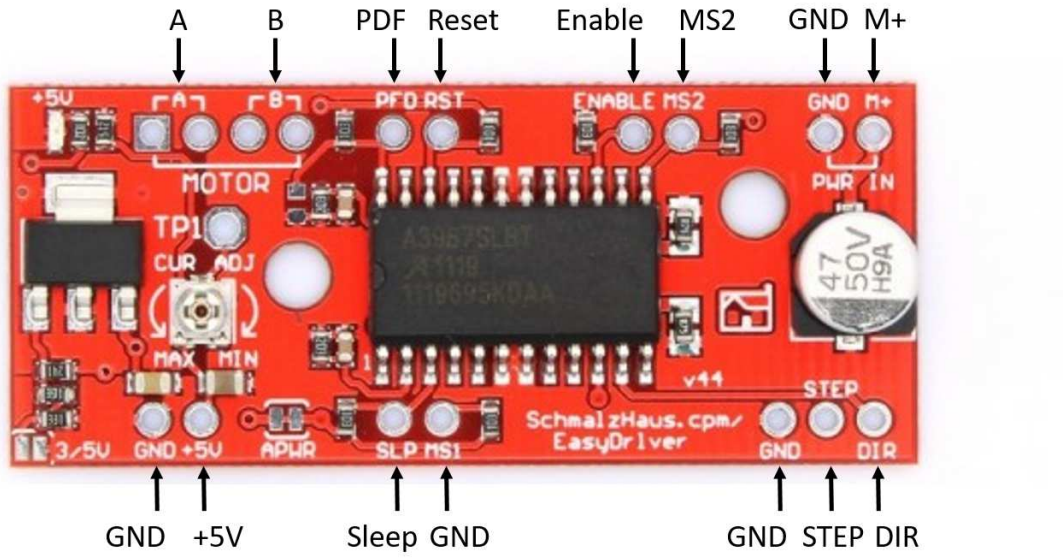

FIGURA 2.7: Pins del driver A3967.

Els pins principals i imprescindibles són els següents:

- A i B: Aquestes son les entrades dels 4 cables del motor, s'ha de connectar tenint en compte els dos parells de bobines diferents, una al als pins A i l'altre als pins B.
- GND: Hi ha tres pins amb aquest nom i són la connexió a terra necessària a qualsevol circuit electrònic.
- STEP: Connectat a un pin digital d'Arduino, es l'encarregat de realitzar els passos del motor. A cada pujada de 0 a 5V, ordena al motor realitzar un pas.
- DIR: Igual que l'STEP, va connectat a un pin digital d'Arduino (0-5V) i defineix segon el seu estat (high/low) la direcció del motor.
- M+: És l'entrada positiva de la font d'alimentació del motor i es recomana alimentar-la a un màxim de 12V. L'entrada negativa de la font es connecta al pin GND del costat.

La resta de pins permeten modificar el comportament del motor, però són opcionals:

- MS1 i MS2: Aquests pins són els encarregats del microstepping. Poden reduir l'angle del pas i per tant augmentar la precisió de moviment del robot ja que redueixen moviment per pas. Hi ha 4 opcions de funcionament en funció de la connexió d'aquests dos pins que poden ser 0V (low) o 5V (high). El pas normal o full-step s'aconsegueix amb la combinació low/low, el mig pas o half-step amb high/low, un quart de pas o quarter-step amb low/high i un vuitè de pas o eight-step amb high/high. Per defecte la connexió és high/high, i per tant divideix el pas per 8.
- Enable: És un input connectat com a low que elimina les sortides si es canvia a high.

- Reset: És un input connectat com a high que quan es canvia a low torna el controlador a la configuració inicial.
- Sleep: És un input connectat com a high que minimitza el consum dels motors quan aquests no s'utilitzen si es canvia a low.
- +5V: Output de 5V que es pot utilitzar per alimentar components que funcionin a baixa corrent.
- PDF: Aquest pin no s'utilitza, controla el mode de decadència del corrent de sortida.

2.3 Servomotor

2.3.1 Quina és la seva funció?

El servomotor és l'encarregat d'accionar el mecanisme per aixecar i baixar l'element de dibuix, en aquest cas el retolador, i, que per tant, permetrà a l'usuari decidir quins trams de la trajectòria formen part del dibuix i quins són moviments d'orientació i posicionament, els quals, gracies a aquest, no quedaran representats ja que no formen part del disseny. El funcionament del mecanisme d'elevació esta format pel servomotor que al rotar acciona una palanca unida al eix com a element terminal que, alhora, provoca un moviment vertical d'una placa plana unida al bolígraf que es desplaça per una guia.

2.3.2 Funcionament

Els servos són un tipus de motors molt utilitzats en la robòtica gràcies a la seva gran precisió angular en el posicionament, ja que permeten controlar la posició de l'eix en tot moment. Està dissenyat per posicionar-se en un rang determinat de posicions angulars (normalment de 0^o a 180^o), amb el qual es pot controlar el gir i la velocitat de rotació. Un cop realitzada la rotació de l'eix, el propi servo produeix un parell de resistència contra la rotació per mantenir estable la posició. També existeixen servos de rotació continua, però aquests no tenen control sobre la posició del mateix ja que, per tal de permetre la rotació continua s'elimina el potenciòmetre encarregat de tancar el llaç de control de la posició.

La composició d'un servomotor es bassa, normalment, en 4 elements fonamentals:

- Motor de corrent continua: És l'encarregat de moure el motor, produeix una rotació de l'eix al aplicar-hi un corrent i es pot canviar el sentit de rotació aplicant el corrent invers.
- Tren d'engranatges: Són un conjunt d'engranatges reductors que actuen per reduir la velocitat de sortida de l'eix del motor de corrent continua i brindar un parell més elevat per mantenir estable la posició.

- Potenciòmetre: Es col·loca a la sortida de l'eix del motor de corrent continua per controlar la posició angular i així tancar el llaç de control. És pot eliminar i convertir el servo en un servo de rotació continua, però llavors es perd el control de la posició ja que el llaç de control quedaria obert.

- Circuit de control: Circuit electrònic encarregat de tancar el llaç de control comparant la senyal d'entrada amb la senyal de sortida mesurat pel potenciòmetre. Compara aquestes dues senyals per calcular l'error i corregir-lo, per tant un servo motor és un sistema de llaç tancat com el que es mostra a la figura.

La connexió del servo és realitza a partir dels 3 cables que presenta, dos cables d'alimentació (positiu i negatiu) connectats a la font d'alimentació entre 4,8 i 6 V i un senyal de control per definir la posició mitjançant un senyal modulat per amplada de pols (PWM).

Mitjançant un microcontrolador, en el aquest cas un Arduino UNO, s'envia un senyal de polsos quadrats (PWM) al motor i, depenent de l'amplada del propi pols, el circuit de control posiciona l'eix a l'angle desitjat. Per la majoria de servos el senyal de control és el mateix, amb polsos quadrats d'entre 15 i 25 mil·lisegons de cicle de treball i l'angle depèn de l'amplitud del pols seguint l'exemple de la imatge.

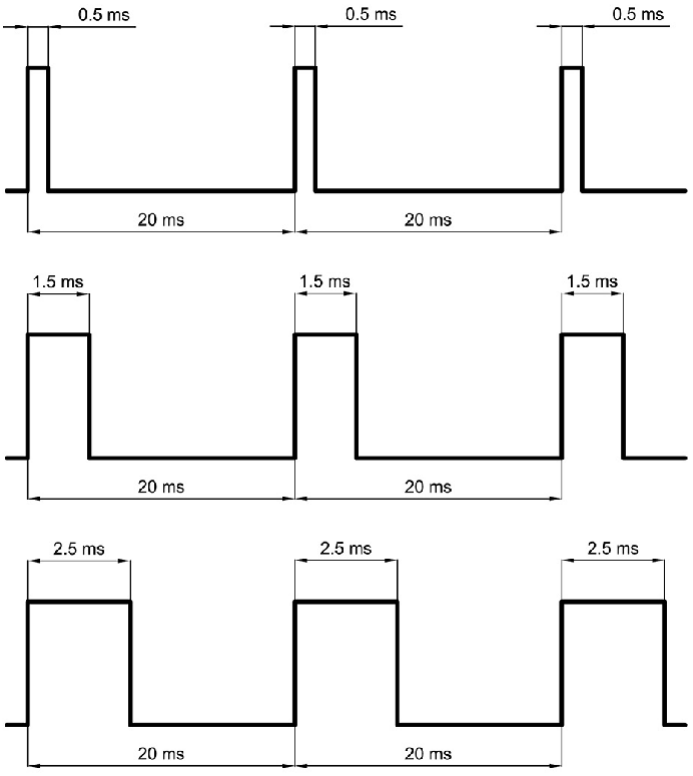

FIGURA 2.8: Exemples de senyal d'entrada del servomotor.

Amb la rotació del propi motor es posiciona el potenciòmetre permeten d'aquesta manera controlar i corregir l'error, per tant tancar el llaç de control entre el senyal d'entrada i el senyal de sortida llegit com la posició del potenciòmetre.

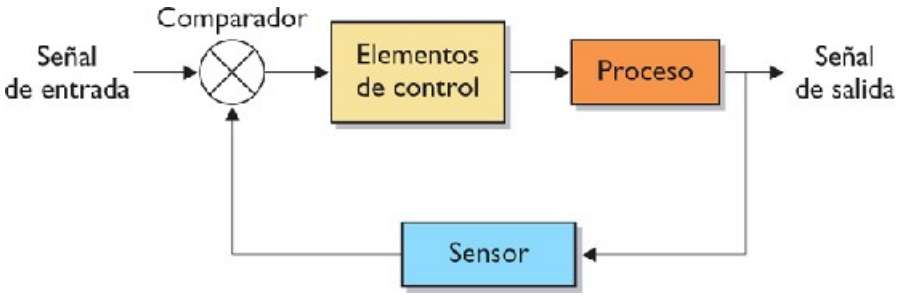

FIGURA 2.9: Llaç de control del servomotor. (provisional)

Existeixen 2 tipus diferents de servomotors, els analògics i els digitals. Els digitals tenen millors característiques, millor resolució, resposta més ràpida, una acceleració més suau durant el moviment i un parell més elevat en estat estacionari. Però, per altra banda, tenen un consum més elevat ja que treballen a una freqüència deu vegades majors a la dels analògics i el seu preu és molt més elevat.

2.4 Bateria

L'objectiu del projecte era dissenyar un robot autònom, i per aquesta raó s'ha decidit incorporar una bateria que subministra l'energia necessària als dos motors així com al microcontrolador Arduino Uno. Després de les proves inicials es va decidir separar les fonts dels dos motors per evitar d'aquesta manera un mal repartiment de la potencia i complir en tot moment amb les exigències energètiques d'ambdós motors.

S'ha optat per escollir una bateria externa recarregable de la marca Proda de 10000 mAh per dos motius, en primer lloc perquè ofereix dues sortides de corrent de 5V a 1 i 2A respectivament i pot per tant alimentar els dos motors per separat, i perquè s'havia comprat abans de començar el projecte i no calia adquirir-ne una altra.

És una bateria de polímer d'ions de liti (LiPo) que permet emmagatzemar una gran càrrega d'energia i per tant permetrà operacions de llarga durada del prototip. L'avantatge que presenta és, com ja s'ha explicat, que té dos canals amb els quals és possible alimentar tots els components del robot des d'una mateixa font d'alimentació. Així doncs, el canal de sortida de 5V i 2A s'encarrega d'alimentar el motor dret del robot i el microcontrolador Arduino UNO (que tanmateix el propi Arduino subministra el corrent necessari pel funcionament del mòdul Bluetooth), mentre que el de 5V i 1A alimenta el motor esquerra i el servo. També conté una petita pantalla per mostrar la càrrega actual de la bateria i que així l'usuari sàpiga en tot moment si cal recarregar-la. Per altre banda, el desavantatge que presenta és el seu elevat pes (170 grams),

tot i que després de les proves inicial es va comprovar que els motors eren capaços de moure aquesta càrrega sense problemes.

FIGURA 2.10: Imatge de la bateria utilitzada.

2.5 Retolador

Com a element de dibuix s'ha escollit un retolador Edding 1200. És un retolador que es pot compra a qualsevol papereria, amb una amplia gamma de colors i un funcionament excel·lent. Les principals característiques que el fan perfecte per aquesta funció són el seu perfil circular de diàmetre constant que permet el lliscament dins de la guia de plàstic fixe al xassís del robot i que la seva tinta s'asseca de forma immediata després de pintar el paper, evitant així que les rodes puguin fer malbé el dibuix si passen per sobre el traçat i arroseguen la tinta si encara no s'ha assecat. Amb el seu propi pes més el del suport que l'aguanta és suficient per dibuixar la línia sobre el paper i no ofereix resistència per fricció, i per tant no afecta negativament a la trajectòria del robot.

Com s'explicarà més endavant a l'apartat (3.3), s'ha creat un suport per tal d'aixecar-lo quan la trajectòria del robot sigui de posicionament i la seva posició és al centre de l'eix coaxial als eixos dels dos motors, facilitant d'aquesta manera el control de la seva trajectòria.

FIGURA 2.11: Imatge de la bateria utilitzada.

Capítol 3

Disseny

En aquest apartat es descriu el disseny del robot, la creació de l'estructura i la distribució dels components, explicant el perquè d'aquesta distribució i les permeses que s'ha tingut en compte alhora de dissenyar les diferents peces en Solid Works per tal d'imprimir-les posteriorment a l'aula Rep Rap de la universitat.

3.1 Xassís

El xassís és el cos del robot, incorpora totes les altres peces i el seu disseny és una de les parts fonamentals del treball. És l'encarregat de centrar els eixos dels motors i fixar la seva posició, d'incorporar la bateria, el microcontrolador, els divers i tot el circuit elèctric i d'assegurar la seva posició fixe durant el moviment del robot.

L'objectiu era dissenyar un xassís amb les dimensions més petites possibles, i aquestes van venir donades directament per la mida dels dos motors. S'ha dissenyat amb una amplada de X tenint en compte que cada motor necessita com a base de suport un rectangle de X x X mm i a l'espai entre mig dels dos cal adaptar el suport del retolador. Per tal de no posar unes limitacions molt restrictives, s'ha decidit realitzar un forat per l'element de dibuix de XX mm de diàmetre per tal de poder adaptar el suport a les mides del retolador o estri escollit. És fonamental que aquest forat estigui centrat entre els dos motors i amb un eix perpendicular i coincident amb els eixos dels motors que han d'estar alineats. D'aquesta manera s'aconsegueix l'amplada mínima, ja que la resta d'elements són de dimensions més reduïdes i per tant s'han posicionat de manera centrada per mantenir centrat el centre de masses.

Per tal de fixar els motors s'han incorporat uns suports verticals que fixen amb cargols la cara frontal del motor (on neix l'eix) evitant així que pugui desplaçar-se i mantenint d'aquesta manera els dos motors sempre alineats. Aquest suports es van imprimir incorporats directament al xassís, per minimitzar els possibles errors de muntatge o descentratge. De la mateixa manera, aprofitant aquests suports i per optimitzar l'espai, s'ha incorporat una petita part plana horitzontal sobre els motors per instal·lar-hi els drivers dels mateixos i així mantenir-ho junt i

facilitant el reconeixement del driver corresponent a cada motor. Cal destacar que aquest suports no tanquen la part superior de tal manera que es muntar posar i desmuntar els motors de manera més còmode sense, per exemple, desmuntar el suport del retolador.

La longitud del xassís ha de ser la suficient per col·locar-hi els motors, el servomotor encarregat d'aixecar el retolador, la bateria, el microcontrolador Arduino i la roda davantera. D'aquesta manera, el xassís final te una llargada de XXX mm i la següent distribució:

En primer lloc es posicionen els motors en un espai de XX mm (fixació?). Seguidament hi ha un espai reservat al suport fixe del bolígraf que s'ha dissenyat com una peça diferent per tal de poder adaptar-la a altres utillatges sense haver de modificar el xassís per complet. Disposa d'una base per enganxar el servomotor fixant la seva alçada de manera que l'eix del servo es mantingui a la mateixa alçada que l'extrem del suport fixe, per tal d'exercir la força per aixecar el retolador sobre el suport mòbil des de la posició horitzontal per gaudir del recorregut màxim del moviment si fos necessari.

En següent lloc s'ha dissenyat un espai reservat per la bateria el qual compte amb un petit compartiment que manté la bateria en posició vertical i fixe per evitar que caigui o pugui interrompre el funcionament del robot. Aquest compartiment té una mida de XxX amb tres parets de Xmm d'alçada i la paret posterior de Xmm que s'aprofita per fer de base de l'Arduino. Es pot observar que les mides són uns mil·límetres més grans que la pròpia bateria ja que s'ha deixat l'espai suficient per tal de poder posar els cargols per fixar l'Arduino, és per aquesta raó que s'ha deixat un petit espai entre les parets laterals i aquesta per poder manipular els cargols amb la bateria fixada si fos necessari. S'ha decidit instal·lar la bateria al punt mig del robot per distribuir la massa al llarg del xassís i evitant que el centre de masses quedi a la línia de l'utillatge de dibuix per evitar que el robot pugui bolcar en algun punt de la trajectòria degut a algun moviment concret.

L'Arduino, com s'acaba d'explicar es col·loca a la paret vertical del compartiment de la bateria de manera que queda fixat verticalment, optimitzant així els espais. S'han dissenyat els forats per tal de col·locar l'entrada del port serial a la part superior fent possible d'aquesta manera la programació de la placa un cop s'ha implementat tot el prototip, ja que sinó s'hauria de desmontar cada cop que es volgues programar. Al lateral de la placa s'han instal·lat dues barres de corrent de protoboard per tal de poder alimentar des de la mateixa sortida de la font un motor conjuntament amb la placa i l'altre motor amb el servomotor.

Per acabar, a la part davantera s'ha deixat l'espai lliure suficient per instal·lar-hi una roda boja que actuarà com a punt de suport per estabilitzar el robot.

3.2 Guia del retolador

Aquest suport esta dissenyat per funcionar com a guia pel retolador, de manera que el seu desplaçament sigui vertical i centrat per minimitzar l'error de descentratge del mateix. Consta

d'una part plana en forma de T la qual anirà fixada al xassís amb 3 cargols i un tub vertical que farà de guia pel retolador. Aquest tub està expressament dissenyat per treballar amb un retolador Edding 1200 (presentat a l'apartat (2.5)) i s'ha dissenyat de manera independent del xassís per així poder fer-la a mida de l'estri que es vulgui utilitzar sense haver de tornar a imprimir tot el cos del robot.

A part de la guia per l'estri de dibuix, també compta amb un petit suport per tal d'enganxar el servomotor de manera que l'eix d'aquest coincideixi amb l'alçada de la guia i aconseguir que el moviment del servo pugui ser màxim, des de la posició horitzontal amb l'estri actiu fins a una posició mitja amb l'estri aixecat i per tant, sense actuar. No és més que un petit graó a la part dreta del suport de les mides del servo i l'alçada de Xmm per aconseguir la posició desitjada del servo, que s'ha fixat directament amb cinta adhesiva de doble cara ja que el seu esforç no és massa elevat i queda el fixe perfectament sense haver de foradar el suport imprès en 3D.

3.3 Mecanisme d'elevació del retolador

Consta d'una petita placa fixada a certa altura del retolador que actua com a superfície per tal de poder transmetre el moviment circular del servo en el moviment vertical de l'utillatge. L'alçada a la qual s'ha de col·locar ve donada per la guia del moviment fixe al xassís, assegurant aquesta manera que quan el servomotor està en posició de dibuix aquest suport es recolzi a la cara plana superior de la guia i el retolador estigui en contacte amb el paper.

S'ha dissenyat amb una part més allargada per tal d'assegurar el contacte i l'acció del servomotor al aixecar el braç, i al altre costat s'ha deixat un espai buit i dues parets verticals foradades per tal de fixar l'estri amb l'ajuda d'un cargol un cop ajustada l'altura a la qual s'ha de fixar, mantenint el servo en posició activa, el retolador tocant al paper i el suport recolzat a la cara superior de la guia. L'eix vertical serveix per tal d'assegurar la perpendicularitat de l'estri amb la base del suport.

3.4 Roda boja davantera

Com a tercer punt de suport i per estabilitzar el robot, s'ha dissenyat una roda boja de manera que no intervingui en el moviment del robot, permetent el lliscament en qualsevol direcció. Consta de dues parts, el cos imprès en 3D i una bola de vidre que gira lliurament a l'interior. El cos assegura el contacte amb la bola en només 4 punts d'una circumferència paral·lela a la base fent així que el lliscament sigui més fàcil i minimitzant la força de fricció. També assegura que la bola es mantingui dins de la cavitat quan s'aixeca el robot fet que provoca que la bola entri a pressió. La cara superior és una cara plana per facilitar la impressió 3D de la peça amb dos forats per fixar-la amb cargols al xassís i un forat per on es pot observar la bola per poder treure-la aplicant una petita pressió.

L'alçada de la roda, junt amb les rodes motrius, són les encarregades fixar l'alçada del robot, de manera que s'han dissenyat conjuntament per tal d'assegurar que la base sigui paral·lela al paper i mantenir l'eix del retolador vertical i centrat.

3.5 Rodes motrius

Les rodes són una part fonamental del disseny, ja que són les encarregades del moviment del propi robot. Hi ha quatre aspectes bàsics per a la seva construcció: minimitzar la superfície de contacte amb el paper, assegurar la perpendicularitat i concentricitat amb l'eix, evitar el lliscament perquè giri de manera solidaria al eix i optimitzar el diàmetre per aconseguir un pas més precís disminuint al màxim el diàmetre.

Per minimitzar la superfície de contacte amb el paper, s'ha decidit utilitzar una junta tòrica de goma d'1,5mm de diàmetre, que actua de la mateixa manera que els neumàtics d'un cotxe, evitant el lliscament amb el paper i, al ser circular, es minimitza el contacte fent-lo puntual o gairebé puntual. Així s'evita que un major contacte de la roda pugui actuar en contra del moviment del robot creant forces de fricció que evitin la rotació del robot sobre el punt de contacte. Amb una superfície puntual també es millora la precisió dels moviments circulars, ja que es calculen a partir de la distància entre el retolador i el punt de contacte, i per tant, com més petit sigui aquest últim, més exacte i constant serà aquesta distància durant el moviment radial de la roda. Per tal de mantenir el neumàtic a la roda, s'ha imprès directament amb un petit conducte amb la forma de mig neumàtic aconseguint així que encaixin.

En la pròpia impressió de la roda, s'ha afegit un eix transversal del mateix diàmetre que l'eix del motor per assegurar-ne d'aquesta manera la perpendicularitat i concentricitat. És un fet clau per assegurar el correcte funcionament, perquè, d'igual manera que al punt anterior, s'ha d'assegurar que la distancia entre les rodes sigui sempre la mateixa al llarg de tots els punts de la rotació de les mateixes, ja que és així com s'ha calculat la trajectòria.

Un altre element del disseny són les dues petites parets verticals foradades que surten de l'eix i el forat que es pot observar al cos de la roda. Això permet que amb un cargol i una femella es pugui fixar la roda al eix ja que es crea una deformació ínfima que redueix el forat per on passa l'eix contra el propi eix fent que quedin fixats sense alterar la forma de la roda. El forat a la pròpia roda és d'aquestes dimensions per tal de permetre la manipulació dels cargols del motor sense la necessitat de treure la roda.

Per acabar, el diàmetre de la roda s'ha intentat minimitzar de manera que es disminueix l'avanç de la roda per cada pas, augmentant així la precisió. Això s'ha realitzat tenint en compte que la distancia entre l'eix del motors i la base més una distància prudencial de XXXmm per poder col·locar els cargols necessaris per fixar els suports de l'utillatge de dibuix i la roda boja. D'aquesta manera s'han dissenyat unes rodes de XXXmm que sumat als neumàtics de goma

3.6 Cargols, femelles i tubs auxiliars

Per l'assemblatge de tot el prototip s'han utilitzat cargols de mètrica M3 i les femelles adients per tal de fixar-los. S'han utilitzat un total de 14 cargols M3x10mm de llargada, 3 per fixar el suport del retolador, 3 per fixar la placa Arduino al xassís i 8 per fixar els motors als suports del xassís, 4 per cada motor. Per altre banda, s'han emprat 4 cargols M3x30mm per fixar els divers (2 per cadascun) i 2 més per la roda boja davantera, que es podia fixar amb cargols més curts, però s'ha decidit per aquests ja que estaven en disposició. Tots els cargols porten una femella per tal de poder fer fixe la unió, i, en el cas dels motors, perquè es necessitava separar els cargols dels cargols de tancament posteriors del motor amb els quals comparteixen eix, i amb una femella s'ha escurçat la llargada de 10mm ja que eren els cargols dels quals es disposava.

Per altre banda, s'han dissenyat i imprés en 3D uns tubs de plàstic, per tal d'elevar tant els divers com la placa Arduino. En primer lloc s'han aixecat els divers XXXmm respecte als seus suports per permetre d'aquesta manera la correcta connexió dels cables que necessita la qual es fa per la part inferior del mateix. Per altra banda, el microcontrolador s'ha separat XXXmm respecte la paret per així evitar el contacte dels pins soldats i evitar possibles sobreescalfament.

Capítol 4

Programació

En aquest capítol es presenta la programació del robot per tal d'aconseguir el moviment desitjat. Aquesta programació es basa principalment en dos grans blocs que actuen conjuntament, el microcontrolador Arduino i l'ordinador programat amb Python. La comunicació entre aquests es realitza a través d'una connexió Bluetooth que permet la transmissió bidireccional entre ambdós a través del port serial. Aquesta connexió s'ha aconseguit gràcies al mòdul HC-05 connectat a l'Arduino que també es detallarà a continuació.

4.1 Principis de funcionament

L'objectiu final del projecte és el de dibuixar una figura amb el robot, i per aconseguir-ho s'ha de passar per diferents etapes. En primer lloc s'ha de crear la figura que es vol dibuixar, es recomana fer-ho a partir del programa de software lliure Inkscape ja que és una eina de dibuix polivalent i té l'opció de generar el GCode (que s'explicarà més endavant) associat a tal figura en un arxiu de text, tot i que també es pot realitzar a partir de l'aplicació creada en aquest projecte. Amb la figura dibuixada s'ha de crear l'arxiu GCode per representar-la, que defineix la trajectòria que ha de seguir el robot. Un cop obtinguda la trajectoria es processa amb el programa creat amb Python, l'arxiu RobotMoveBT.py, que a partir d'aquest fitxer de text calcula la trajectòria que han de seguir les rodes i ho tradueix a passos dels motors. A partir d'aquí, el propi programa s'encarrega d'enviar les comandes corresponents a l'Arduino del robot que ho tradueix en els moviments dels motors, tant del servo com de les rodes. Aquesta comunicació es realitza per Bluetooth i es fa comanda per comanda, per la qual cosa els dos programes seran executats durant tot el recorregut del robot.

DIAGRAMA FLUX

4.2 GCode: Com es calcula la trajectoria?

4.2.1 Que és el GCode?

El GCode és el llenguatge de programació més utilitzat a l'actualitat pel control de màquines CNC de control numèric com ara torns, fresadores i impressores 3D. Tot i que existeixen diferents dialectes del mateix codi, tots segueixen unes normes bàsiques que el fan entenedor. La seva funció és descriure el comportament de la màquina, "què" farà i "com" ho farà, controlant la trajectòria, les velocitats de tall i avanç i el posicionament de l'eina, entre d'altres.

L'estructura del llenguatge és molt simple: cada línia de codi representa una nova instrucció o bloc i per tant es llegeix per línies. Hi ha diferents tipus d'ordres i per diferenciar-les es segueix una combinació de lletres i números. Sempre segueix la mateixa estructura, cada ordre es representa com una lletra seguida d'un número i dins una mateixa línia es poden combinar més d'una ordre. La lletra diferencia entre les diferents comandes com, per exemple, la lletra "G" s'utilitza per comandes de moviments, la "M" per comandes miscel·lànies auxiliars o de configuració i "X", "Y" i "Z" com a posició absoluta o incremental del punts d'aplicació.

4.2.2 Comandes principals

Les comandes més utilitzades, i les que s'utilitzaran principalment en el treball, són:

- G00: Posicionament ràpid. Comanda que, acompanyada amb la posició final X Y Z, posiciona l'eina de forma ràpida fins a la posició indicada. De totes les comandes que es descriuen en aquest apartat, és l'única en que leina no treballa, només es mou. Exemple: *G00 X0 Y100 Z0.*
- G01: Moviment en línia recta. L'eina realitza un moviment rectilini fins a la posició indicada. També es pot definir la velocitat de treball amb la comanda F. Exemple: *G01 X100 Y100 Z0 F100.* Per altre banda, també s'utilitza aquesta comanda per rotar l'eina i corregir la seva orientació. Per fer-ho s'acompanya la comanda G01 amb la lletra A acompanyada de la direcció desitjada en radians, per exemple: *G01 A3.141592.*
- G02 i G03: Arc de circumferència. Moviment circular en sentit horari (G02) o antihorari (G03). La velocitat serà la mateixa que s'hagi definit anteriorment o bé es pot tornar a assignar amb la mateixa comanda F. El moviment es pot definir pel radi amb la comanda R o pel centre de la circumferència amb les comandes I, J i K, sent aquestes tres les coordenades en els eixos X, Y i Z del centre. Exemple: *G02 X200 Y 100 Z0 I150 J0.* Tot i que les coordenades del punt final siguin coordenades absolutes, les del centre acostumen a ser incrementals respecte al punt d'origen.
- M3: Inici de programa. Així s'indica l'inici del programa.
- M30: Fi del programa. Amb aquesta comanda es tanca el programa.

Existeixen moltes més ordres, però aquestes són les principals i les que utilitza l'escriptor de codi del programa Inkscape que és el que es vol utilitzar en aquest cas. També són les únique programades per ser llegides pel robot, així que són les úniques a tenir en compte.

4.3 Inkscape: Com es crea el GCode?

4.3.1 Inkscape

Inkscape és un editor de gràfics vectorial de codi lliure i gratuït que permet crear il·lustracions, formes, gràfics, dibuixos i molt més en format vectorial .SVG (Scalable Vector Graphics). La avantatge que té és que crea un arxiu que conte aquesta figura de manera vectorial, i això ens ajuda a crear a partir del dibuix un codi basat en GCode que s'utilitzarà per guiar el robot i definir la seva trajectòria. Aquest codi es pot guardar com un arxiu de text *.txt* i serà aquest el que utilitzarà el robot per executar el programa.

Es recomana utilitzar la versió del programa 0.91, ja que l'actualització 0.92 té un error amb la creació dels punts d'orientació i no es respecta l'escala desitjada.

FIGURA 4.1: Logo del programa Inkscape.

4.3.2 Com crear l'arxiu GCode a partir d'Inkscape

Aquí es mostraran els passos a seguir per tal de passar un dibuix d'Inkscape a G-code amb l'extensió pròpia del programa Gcodetools.

- Realitzar el dibuix: El primer pas és realitzar el dibuix que es vol enviar al robot perquè aquest el dibuixi. És important definir les mides del "paper" d'Inkscape per tenir una idea de les dimensions desitjades del dibuix. Això es fa des de la pestanya "Archivo/Propiedades del documento". A partir d'aquí ja es pot dibuixar el que es vulgui des de el propi Inkscape. La colocació del dibuix al full serà la desitjada per l'usuari, tenint en compte que l'origen de coordenades serà sempre l'exterem inferior esquerra.

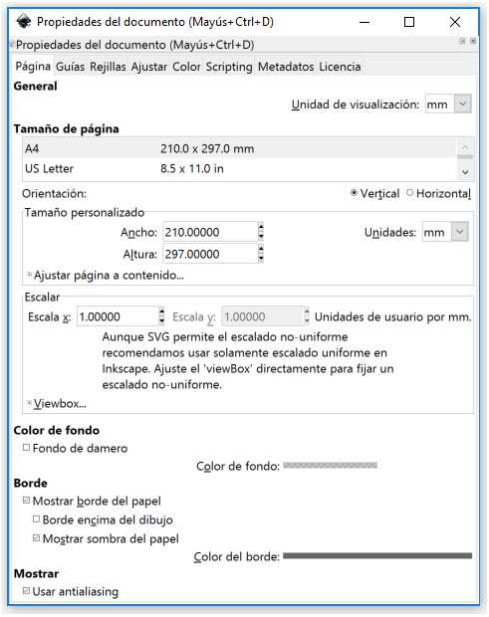

FIGURA 4.2: Pestanya de "Propiedades del documento".

- Convertir l'objecte en un trajecte: Amb l'objecte ja dibuixat, el següent pas és convertir-lo a una trajectoria, vectoritzar el dibuix en vectors rectes o arcs de circumferència per crear després el codi GCode de la figura. Primer cal seleccionar l'opció de "Desvio dinámico" del menú i després seleccionar al menú "Objeto" la pestanya "Objeto a trayecto".

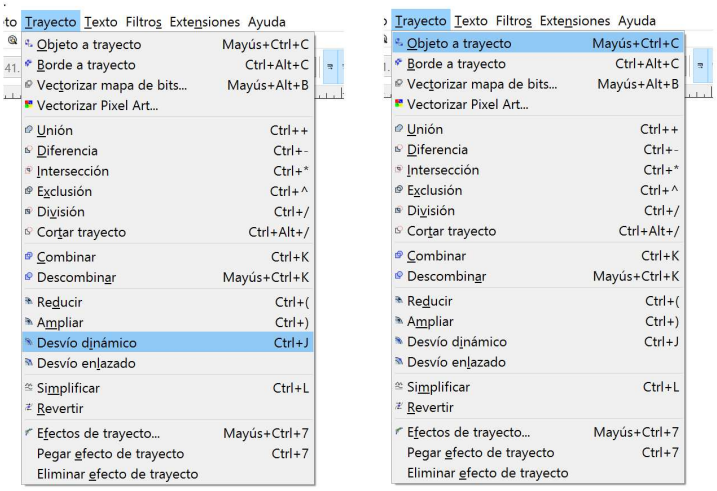

FIGURA 4.3: Opcions a escollir per convertirl'objecte en trajecte.

- Crear els punts de referencia: A partir d'aquí ja es treballarà amb l'extensió Gcodetools. Fent click a l'opció "Extensiones/Gcodetools/Puntos de referencia..." es podrà veure que apareixen, a les cantonades, les referencies (0,0,0) a la cantonada inferior esquerra com a origen de coordenades, que voldrà dir que s'han creat correctament. Això indica que l'origen del dibuix, el punt on situarem el robot inicialment, serà la cantonada inferior esquerra del paper on es vulgui fer el dibuix, com s'ha explicat anteriorment.

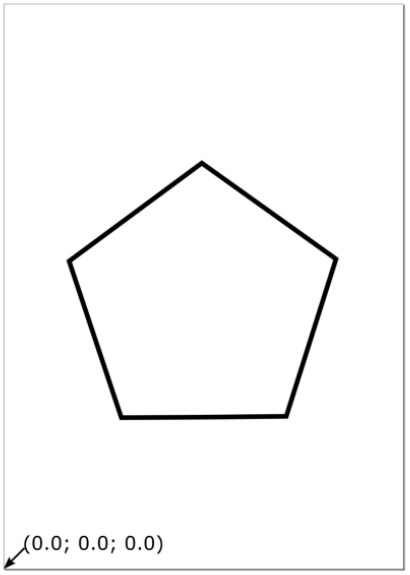

FIGURA 4.4: Punt de referència creat correctament.

- Seleccionar l'eina de treball: L'Inkscape té diferents eines de treball per generar el GCode i cadascuna es caracteritza per una forma diferent de treballar, fet que cs'associa en un canvi en el codi GCode. Per aquesta aplicació, l'eina més interessant i que millor s'adapta és el "Cuchillo tangencial", que a part de la trajectòria del punter també té en compte l'orientació de l'eina, en aquest cas del robot, fet que donarà peu a un codi més fàcil de llegir pel robot, que s'estalviara calcular aquesta orientació. Per seleccionar-la, només cal dirigir-se a la pestanya "Extensiones/Gcodetools/ Biblioteca de herramientas..." y seleccionar l'opció desitjada. Un cop seleccionada s'obre un recuadre verd en el que es poden modificar els paràmetres de l'eina amb l'eina d'edició de text.

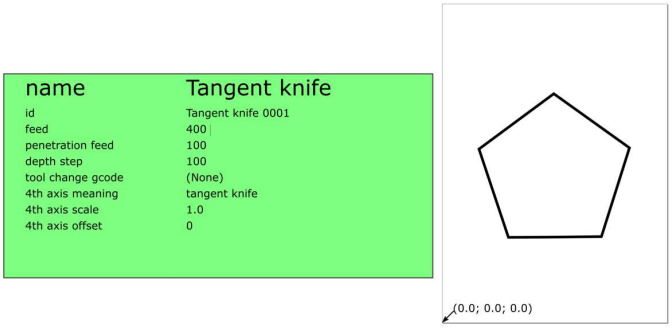

FIGURA 4.5: Quadre de text per la configuració de l'eina "Tangent knife".

- Crear la trajectòria de G-code: Per acabar, ja només caldrà crear l'arxiu amb el GCode. Per fer-ho, s'utilitza l'opció "Extensiones/Gcodetools/Trayecto a GCode..." on es pot modificar el nom i la direcció on es guardarà l'arxiu a la pestanya "Preferencias". Escriurem el nom de l'arxiu acabat amb l'extensió *.txt* per després poder-lo llegir. Tornant a la pestanya

"Trayecto a GCode", fent click a "Aplicar" es crearà aquest arxiu y estarà llest per ser llegit amb l'Arduino.

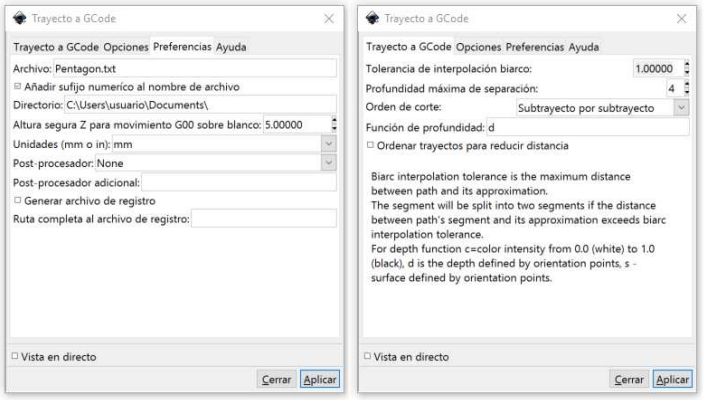

FIGURA 4.6: Menus necessaris per crear la trajectòria.

És important tancar les finestres sempre que acabem amb la operació abans de fer la següent ja que pot comportar errors.

4.4 Python: Com es processa el GCode?

Amb el llenguatge de programació Python s'ha creat un programa capaç de llegir un arxiu .txt creat amb Inkscape i traduir-ho a moviments del robot. En aquest apartat s'explicarà que és el codi Python i el funcionament del programa principal i totes les seves funcions.

4.4.1 Introducció al Python

El python es un llenguatge de programació lliure (Open Source) que té com a objectiu mantenir la simplicitat i ser un codi fàcil de llegir, entendre i d'aprendre. Al ser un codi lliure està en constant creixement ja que tothom pot aportar-hi els seus coneixements i la comunitat de programadors Python és enorme, per la qual cosa és fàcil d'apendre a partir de llibres, manuals o tutorials online.

Aquest llenguatge és el llenguatge aprés a la universitat durant el grau i és per això que s'ha decidit utilitzar-lo. S'ha instal·lat la versió de Python 2.7 directament des de la seva pàgina web. Al següents apartats s'explicarà el funcionament del programa i el seu codi.

FIGURA 4.7: Logo del programa Python.

4.4.2 El programa

El programa encarregat de moure el robot es troba dins l'arxiu "RobotMoveBT.py" que es pot trobar a l'annex XXX i el seu funcionament es basa en diferents funcions. L'objectiu del programa és llegir l'arxiu GCode creat, traduir cadascuna de les ordres en passos del motor i la posició del servo per tal d'enviar-ho després a l'Arduino.

En primer lloc, al executar l'arxiu s'importen les llibreries "math" que ens permet fer operacions matemàtiques complexes, "serial" per tal de comunicar l'ordinador i l'Arduino i "time" per tal de programar temps d'espera per una bona comunicació. Tot seguit s'inicia la comunicació amb el Robot mitjançant la següent comanda:

```
import math, serial, time
arduino=serial.Serial('COM4', 9600)
```

Notar que el port 'COM4' és el port que ocupa la connexió Bluetooth, i que per tant s'ha de configurar abans d'executar-lo si s'utilitza des d'un altre ordinador. A partir d'aquí es defineixen una sèrie de variables globals que serviran per tot el programa, les quals s'utilitzaran en les altres funcions. Defineixen els paràmetres bàsics del Robot i la seva configuració.

```
StepsVolta=1600  #passos per realitzar una volta completa del motor
x0 = 0.0  #coordenades absolutes que ocupa el robot
y0 = 0.0  #coordenades absolutes que ocupa el robot
pasdreta=0  #posicio absoluta del motor dret en pasos
pasesquerra=0  #posicio absoluta del motor dret en pasos
DiamRoda=50.9295  #diametre de la roda en mm
RadiRoda=DiamRoda/2.0  #radi de la roda en mm
distDreta=60.0  #distancia en mm entre el centre de l'eix (punt del boli) i la
                                    roda dreta
distEsquerra=61.5  #distancia en mm entre el centre de l'eix (punt del boli) i
                                    la roda esquerra
pas= math.pi *DiamRoda/StepsVolta  #mm recorreguts per pas del motor
direccio0=math.pi/2  #angle inicial del robot a 90 graus
tempsLectura=0.01  #temps per llegir l'Arduino
```

A partir d'aquí s'han creat diferents funcions que es poden separar en dos grans apartats: Funcions de moviment del robot i funcions de lectura.

4.4.3 Funcions de moviment del robot

S'han definit les funcions bàsiques per tal d'actuar amb les diferents comandes de GCode que poden aparèixer, G00, G01, G02, G03 i G01 de rotació. L'objectiu d'aquestes és calcular els pasos que ha de realitzar cada motor per tal de seguir el moviment definit.

4.4.3.1 Funció G00(x_1, y_1)

Aquesta funció realitza un moviment rectilini fins al punt definit per les coordenades d'entrada x_1 i y_1 amb el retolador aixecat, de tal manera que aquest moviment no quedarà representat en el dibuix final, és un moviment de posicionament.

En el cas de l'ordre G00, el primer que cal fer és rotar el robot per tal de posicionar-lo en direcció al punt de destí (x_1, y_1). És en la única operació que cal fer-ho, ja que, al utilitzar l'eina "Cuchillo tangencial" d'Inkscape , sempre que hi ha un moviment amb el retolador en posició activa primer es realitza una ordre G01 AX que rota el robot fins a la direcció (X) adequada per començar el moviment, però al tenir el retolador aixecat el propi Inkscape entén que es pot moure fins a les coordenades destí de qualsevol manera, però com que el moviment del robot és empre en una única direcció, cal rotar-lo per tal d'apuntar cap al punt final i realitzar un moviment rectilini. Aquest càlcul es fa utilitzant l'arctangent del valor de $\frac{\Delta y}{\Delta x}$ amb el següent codi:

```
if (dx != 0):
  direccio = math.atan(dy/dx)
  if (dx < 0 and dy >= 0):
    direccio = direccio + math.pi
  elif (dx < 0 and dy < 0):
    direccio = direccio - math.pi
  apuntar(direccio)
else:
  if (dy > 0):
    direccio = math.pi/2.0
  elif (dy<0):
    direccio = -math.pi/2.0
  else:
    direccio=direccio0
  apuntar(direccio)
direccio0=direccio
```

Com es pot observar, si la direcció és vertical ($\pm\frac{\pi}{2}$) l'arctangent és infinit i no es pot calcular, per això s'estudia apart. Per altre banda, si l'angle pertany al segon i tercer quadrant, cal sumar i restar pi respectivament per aconseguir l'angle desitjat, ja que els angles resultants de l'operació arctangent estan compresos sempre a l'interval $[-\frac{\pi}{2}, \frac{\pi}{2}]$.

Per calcular els passos que ha de fer el motor només cal calcular la distància que hi ha entre el punt inicial i el final, i dividint aquesta distància pel número de mil·límetres recorreguts per cada pas de la roda s'obtenen el número de passos a realitzar, arrodonit a l'enter més proper.

$$Distancia = \sqrt{\Delta x^2 + \Delta y^2} \tag{4.1}$$

$$Pas = \frac{\pi \cdot Diametre\ de\ la\ roda}{Steps/volta} \tag{4.2}$$

$$Steps = \frac{Distancia}{Pas} \tag{4.3}$$

Un cop acabats els càlculs es corregeixen els valors de la posició actual (x_0, y_0), el de la direcció del robot $(direccio_0)$ i la posició absoluta en passos dels motors, la qual s'enviarà a l'Arduino.

Per acabar s'estableix la connexió amb l'Arduino. En primer lloc s'envia una cadena de valors amb el següent format: *"0, passos del motor dret, passos del motor esquerra,"*. El primer valor indicarà la posició del servo motor, i al ser *0* s'està indicant que ha d'aixecar el retolador per realitzar aquest moviment. Cal acabar la cadena amb una coma *","* ja que és així com s'ha programat l'Arduino. Un cop enviada, el programa entra en un bucle fins que no rep de l'Arduino la paraula *"Ready"* senyal que ja es pot enviar una nova ordre. Això s'aconsegueix mitjançant el següent codi:

```
text='0'+','+str(pasdreta)+','+str(pasesquerra)+','
arduino.write(text)
robot=1
while robot==1:
  if arduino.inWaiting()>0:
    st=arduino.readline().strip()
    time.sleep(tempsLectura)
  if st=='Ready':
    robot=0
```

4.4.3.2 Funció G01(x_1, y_1)

El funcionament d'aquesta funció és exactament igual a la funció G00, ja que també és un moviment rectilini, però aquest cop el retolador sí que ha de traçar una línia. La única diferencia entre aquestes és que en el cas de l'ordre G01 no cal redireccionar el robot per apuntar al punt de destí, perquè el propi Inkscape crearà prèviament una ordre especifica per fer-ho, que s'explicarà a l'apartat 4.4.3.5. Per altre banda, en aquest cas, el primer valor que s'envia a l'Arduino serà un *1* ja que la posició del servo ha de ser posició activa amb la qual el retolador està dibuixant.

4.4.3.3 Funció G02(x_1, y_1, x_C, y_C, $direccio_1$)

Aquesta funció te l'objectiu de dibuixar un arc de circumferència en sentit horari. Les entrades de la mateixa són les coordenades absolutes del punt de destí (x_0, y_0), les coordenades relatives del centre de gir (x_C, y_C) respecte el punt inicial i la direcció amb la qual acabarà el moviment.

Per tal de calcular els passos necessaris de cada motor per realitzar el moviment primer es calcularà el radi de gir aprofitant que es coneix la posició relativa del centre de la següent manera:

$$Radi\ de\ gir = \sqrt{x_C^2 + y_C^2} \tag{4.4}$$

Després es realitza un canvi de coordenades per tal de definir les coordenades del centre en el punt *(0,0)* de la següent manera:

Primer es calculen les coordenades absolutes del centre:

$$\begin{aligned} x_C &= x_C + x_0 \\ y_C &= y_C + y_0 \end{aligned} \tag{4.5}$$

I després es fa el canvi de coordenades:

$$\begin{aligned} x_0 &= x_0 - x_C \\ y_0 &= y_0 - y_C \\ x_1 &= x_1 - x_C \\ y_1 &= y_1 - y_C \end{aligned} \tag{4.6}$$

Amb les noves coordenades el primer pas és calcular l'angle de circumferència que separa els dos punts, ja que, com que el retolador està alineat amb les rodes, els tres punts de contacte posterior del robot (rodes i retolador) traçaran la mateixa trajectòria amb un radi diferent, i per tant aquest angle serà el mateix. Aquest es calcula amb les següents línies de codi:

```
angle0=math.atan2(y0,x0)
angle1=math.atan2(y1,x1)
angle=angle0-angle1
if angle<0:
angle=2*math.pi+angle
```

La funció *math.atan2(y,x)* retorna la direcció d'aquest punt, i, restant les direccions del punt inicial i el punt final, es pot calcular l'angle entre ells. Si l'angle resultant és menor a *0* es converteix al seu respectiu angle en positiu per mantenir-se sempre a l'interval $[0, 2\pi]$.

Un cop calculat l'angle s'utilitza les següents equacions per calcular la distancia que han de recorre cadascuna de les rodes tenint en compte la variació de radi:

$$\begin{aligned} Dist.\ recorregut\ R.D. &= Angle * (Radi\ de\ gir - Dist.\ R.D.\ a\ C) \\ Dist.\ recorregut\ R.E. &= Angle * (Radi\ de\ gir - Dist.\ R.E.\ a\ C) \end{aligned} \tag{4.7}$$

Essent (*Dist. R.D. a C*) i (*Dist. R.E. a C*) les distancies entre les rodes (*R.D.* dreta o *R.E.* esquerra) fins al centre (C).

Amb el valor de les distàncies i utilitzant l'equació (4.3) es calculen els passos a recorre per cada motor, s'actualitzen les dades i s'envia la comanda a l'Arduino de la mateix manera que s'ha explicat amb la funció G00, però amb la posició del servo igual a *1* ja que el retolador ha d'estar actiu.

4.4.3.4 Funció G03(x_1, y_1, x_C, y_C, $direccio_1$)

El funcionament és exactament igual al G02, però en aquest cas el gir serà en sentit antihorari. La única diferencia es troba en el càlcul de l'angle i en la variació del radi de gir, que es canvien en el codi de la següent manera:

```
angle=angle1-angle0
...
distanciaD = angle * (RadiGirC+distDreat)
distanciaE = angle * (RadiGirC-distEsquerra)
```

4.4.3.5 Funció apuntar($direccio_1$)

Aquesta és la funció encarregada de la rotació del robot. S'utilitza quan l'ordre del GCode és G01 AX on X és la direcció a la qual ha d'apuntar el robot i dins de la funció G00 per tal de redireccinar-lo.

El càlcul de passos consisteix primer en la decisió del sentit de gir per tal de saber per quin és el camí més curt, que es troba amb les següents línies de codi:

```
gir=direccio0-direccio1
absgir=abs(gir)
if (absgir > math.pi):
  if (gir<0):
    gir= 2.0 * math.pi + gir
  else:
    gir= -2.0 * math.pi + gir
```

Amb la qual, si el valor de la variable *gir*, que és l'angle a girar pel robot, és negatiu girarà en sentit antihorari, i sinó en sentit horari. Aquest angle mai serà superior a π.

El càlcul de passos es basa en l'equació (4.3) de l'apartat G00. En aquest cas, el valor de la posició del servo és *0*, ja que s'ha decidit així per tal de no mantenir el contacte del retolador amb el paper i evitar així taques de tinta als punts de gir.

4.5 Arduino: Qui mou el robot?

Arduino es presenta com una plataforma de codi lliure (Open-Source) basat en un hardware i un software de fàcil utilització i accessible a tothom. Permet crear prototips a partir de la lectura d'entrades (inputs) com podrien per exemple ser un sensor de temperatura, l'acció d'un polsador o una comunicació des de l'ordinador, i convertir-ho en una acció de sortida (output) com podria ser el moviment d'un motor, enviar un missatge a un ordinador o qualsevol cosa que l'usuari pugui programar. La programació de la placa es realitza mitjançant el software (IDE) lliure de la marca basat en Processing i en el seu propi llenguatge de programació Arduino basat en Wiring.

Arduino neix a la ciutat piemontesa d'Ivrea al nord d'Itàlia al Ivrea Interaction Design Institute com un projecte pels estudiants de l'institut de mans del professor Massimo Banzi. Va començar com una eina de prototipatge ràpid utilitzada per la introducció a l'electrònica i la programació dels estudiants sense experiència. Però el projecte va ser molt ben acollit per la comunitat internacional convertint-se així en una eina molt utilitzada arreu del món per principiants i experts. Al llarg dels anys ha evolucionat molt per tal de brindar noves opcions als usuaris així com una col·laboració amb Google per aconseguir comunicar directament la placa amb el sistema operatiu Android o l'adaptació de noves plàques per treballar amb aplicacions d'IoT (Internet of things) o Wearables. Totes les plaques d la companyia són Open Source igual que el software que es manté en constant creixement gràcies a l'aportació dels usuaris.

FIGURA 4.8: Logo d'Arduino.

4.5.1 Arduino UNO

Per aquest projecte s'ha utilitzat el model Arduino UNO. Aquest model es basa en un microcontrolador ATmega328P amb memòria flash de 32Kb programable des de el software Arduino. Presenta 14 pins digitals d'entrada/sortida, 6 dels quals es poden programar com a sortida amb

modulació de pols PWM, i 6 entrades analògiques disponibles per l'usuari, un oscil·lador de quars de 16MHz, una connexió USB per connectar-lo a l'ordinador, connexió ICSP-6 per programar-la, una entrada de corrent de 5V i un polsador de reinici.

És una placa bàsica, la primera que va sortir al mercat, ideal per a projectes petits i la iniciació en el món Arduino.

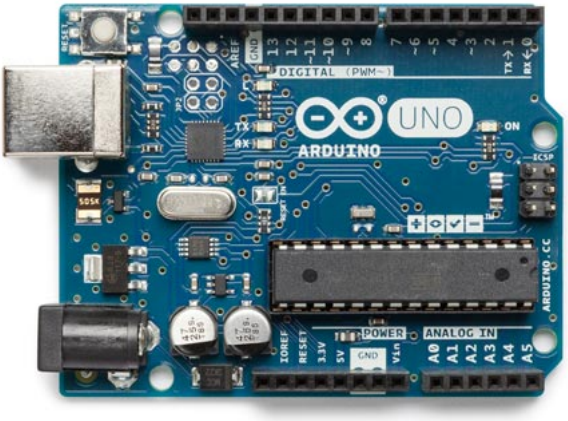

FIGURA 4.9: Placa Arduino UNO.

4.5.2 Connexió

La connexió de l'Arduino és molt important, ja que es on es connecten tots els elements del robot i l'encarregada de que tot funcioni. Seguidament es presentarà la connexió dels diferents elements.

En primer lloc els motors es connecten a partir dels divers i ocupen 4 entrades de la placa cadascun. Les dues primeres són els pins de Step i Direcció del controlador, els encarregats de definir la quantitat i la direcció dels passos del motor, que es connecten als pins digitals 8 i 9 (direcció i Step) en el cas del motor dret i 12 i 13 (direcció i Step) en el cas del motor esquerra. Els altres dos pins que ocupen cada motor a l'Arduino són els anomenats MS1 i MS2 que, com s'ha explicat a l'apartat explicatiu del driver, s'encarreguen de controlar el microstepping (reduir l'angle de pas i per tant augmentar el nombre de passos). Aquest s'han connectat als pins digitals 2 i 3 per la roda dreta i 4 i 5 per l'esquerra. S'ha programat de manera que la sortida estigui activa (HIGH) i per tant rebi 5V, la qual cos es traduirà en un augment de 8 vegades els passos inicials (200 passos per volta) fins 1600 passos per volta. Aquesta configuració permet augmentar considerablement la precisió de la traçada. Els pins de GND de referència dels divers s'han connectat a les entrades GND de l'Arduino per assegura un mateix connector a terra per tot el disseny. Per acabar els pins d'alimentació (5V i GND) s'han connectat directament a la sortida de la font amb les dues fileres de pins de la protoboard.

Per altre banda, el senyal de control del servomotor es connecta al pin 6 amb sortida PWM de la placa per tal de definir la posició de l'eix en funció del pols de l'entrada. Els cables de

voltatge es connecten directament a les tires de pins de la protoboard a la sortida de la bateria compartida amb el motor esquerra. S'evita així connectar-ho a la sortida de 5V de l'Arduino per repartir el consum entre les dues fonts, com ja s'ha explicat a l'apartat de la bateria.

Per acabar, es connecta el mòdul HC-05. Els senyals TX i RX, que es detallaran més endavant, actuen sobre les entrades digitals 10 i 11 respectivament, mentre que les entrades de corrent es connecten a les sortides de 3,3V i GND de l'Arduino.

Un cop fetes totes les connexions amb els diferents components, s'alimenta l'Arduino directament des de la font compartida amb el motor dret utilitzant les entrades Vin i GND de la pròpia placa.

Per tal de re-programar el microcontrolador des de l'ordinador, es pot utilitzar l'entrada del port serial USB. És per aquesta raó que el mòdul Bluetooth es connecta als pins 10 i 11 i no als pins 0 i 1 reservats pels canals de lectura i escriptura, ja que si es connectes aquí s'hauria de desconnectar cada cop que es volgués carregar un programa a la placa ja que es podrien crear interferències al estar accedint al mateix canal per dues bandes diferents alhora.

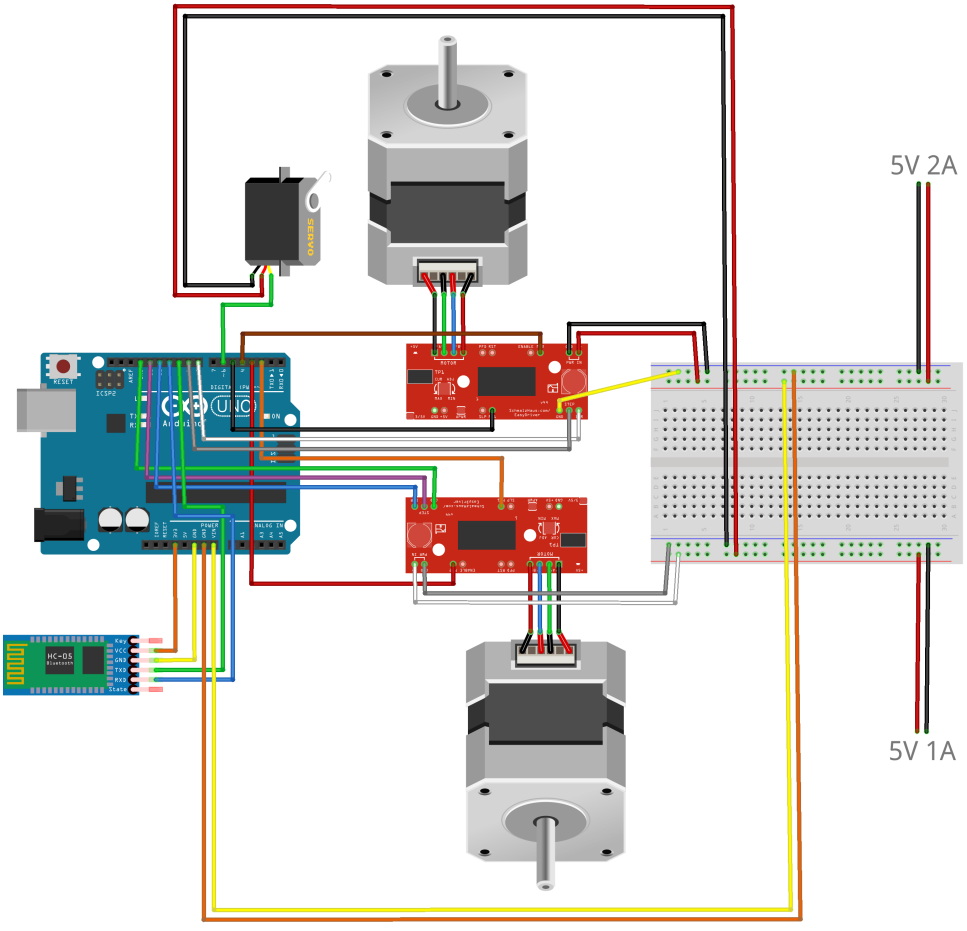

FIGURA 4.10: Connexió dels components.

4.5.3 Programa Robot.ino

S'ha creat un programa genèric amb la funció de llegir les ordres enviades des de l'ordinador. S'ha decidit fer d'aquesta manera per evitar la reprogramació constant de la placa a cada trajectòria i aconseguir així el control total del robot des de l'ordinador.

La dinàmica de comunicació consisteix en rebre un missatge des de l'ordinador que conte 3 valors separats pel caràcter ',' definit com el caràcter de separació. Aquests valors són: en primer lloc la posició del servo, que serà 1 si aquest ha d'estar dibuixant i 0 si s'ha d'aixecar. Els dos següents valors són la posició absoluta dels motors mesurada en passos, en primer lloc de la roda dreta i per acabar de l'esquerra.

Tot seguit es fa una descripció del programa per entendre el seu funcionament. El programa complet està present a l'annex XXX:

4.5.3.1 Llibreries utilitzades

El primer que cal fer és importar les llibreries que s'utilitzaran al executar el programa amb les següents comandes:

```
#include <AccelStepper.h>
#include <MultiStepper.h>
#include <Servo.h>
#include <SoftwareSerial.h>
```

La primera llibreria AccelStepper permet enviar els senyals de control al driver per tal de moure cadascun dels motors, definint la velocitat i els passos que ha de realitzar. És una llibreria molt funcional pel control de motors pas a pas amb Arduino, tot i que no és possible moure els dos motors simultàniament amb aquesta llibreria. És per això que s'utilitza la llibreria MultiStepper, que permet agrupar elements per tal de moure'ls alhora. Per fer-ho, com es veurà més endavant, es defineixen les posicions dels dos motors i aquests realitzaran un moviment constant controlant la velocitat de manera que finalitzaran el seu recorregut amb el mateix increment de temps, permetent així les trajectòries circulars. Aquestes dues llibreries no venen incorporades al entorn Arduino, però són igualment de codi lliure i es poden descarregar de la xarxa.

Les llibreries Servo i SoftwareSerial sí que s'inclouen en la descarrega inicial del software Arduino i per tant no cal descarregar-les d'internet. La primera s'utilitza per controlar la posició del servomotor, mentre que la segona és l'encarregada d'establir la comunicació serial amb l'ordinador a partir del mòdul de Bluetooth.

4.5.3.2 Configuració

Un cop incorporades les llibreries cal inicialitzar tots els paràmetres necessaris:

- Definició dels motors:

```
AccelStepper RodaDreta(1,9,8);
AccelStepper RodaEsquerra(1,13,12);
MultiStepper Robot;

int VelocitatMax=500;
long posicio[2];
int pasdreta=0;
int pasesquerra=0;
```

Primerament es defineixen els motors per separat com a objectes de la classe AccelStepper, on s'indica el tipus de control del motor (en aquest cas l'1 equival a control d'un motor bipolar a través d'un driver) i els pins als quals es connecten els pins d'Step i Direcció de cadascun, entrades 9 i 8 pel motor dret i 13 i 12 per l'esquerra. Es defineix també l'objecte Robot de la classe MultiStepper al qual més endevant s'hi afegiran els dos motors. Després es defineixen constants que s'utilitzaran per definir els moviments, com la velocitat màxima en steps/segon, una matriu amb la posició dels dos motors i la variable es definirà la posició de cada motor com el número absolut de passos que ha fet cada motor.

Amb els objectes ja definits, dins del bucle de configuració inicial (void setup()) es defineixen els pins 2, 3, 4 i 5 com a sortides actives per activar així els senyals MS1 i 2 dels drivers activant d'aquesta manera el microstepping i multiplicant per 8 el nombre de passos del motor. Tot seguit s'afegeixen els dos motors de la classe AccelStepper al Robot que definirà el moviment conjunt dels dos motors i s'estableix la velocitat màxima dels dos motors.

```
pinMode(2, OUTPUT);
pinMode(3,OUTPUT);
digitalWrite(2,HIGH);
digitalWrite(3,HIGH);
pinMode(4, OUTPUT);
pinMode(5,OUTPUT);
digitalWrite(4,HIGH);
digitalWrite(5,HIGH);

Robot.addStepper(RodaDreta);
Robot.addStepper(RodaEsquerra);

RodaDreta.setMaxSpeed(VelocitatMax);
RodaEsquerra.setMaxSpeed(VelocitatMax);
```

- Definició del Servo:

```
Servo boli;
int up=0;
int down=50;
```

Aquí es defineix l'objecte Boli de la classe Servo el qual controlara la posició del servomotor. S'han definit les dues posicions que pot assolir, a 0 graus quan està aixecat i a 50 si està dibuixant. Al bucle de configuració es defineix el pin 6 com a sortida pel servomotor i la posició inicial es fixa en 0, per tant amb el retolador aixecat.

```
Boli.attach(6);
Boli.write(0);
```

- SoftwareSerial:

```
SoftwareSerial bluetooth(10,11);
```

Per acabar es defineix l'objecte Bluetooth de la classe SoftwreSerial que es connecta amb el pin 10 com RX i l'11 com TX de recepció i transmissió de dades. Aquesta connexió s'inicia al bucle de configuració amb la comanda:

```
bluetooth.begin(9600);
```

La comunicació es realitza a 9600 bps (bits per segon) ja que aquesta és la configuració per defecte del mòdul Bluetooth.

4.5.3.3 Bucle principal

Aquest programa es bassa en dos funcions que es criden seqüencialment, de manera que al acabar l'execució de la primera s'executi instantàniament la segona. Aquestes són la funció getSerialData() que, com el nom indica, s'encarrega de rebre i emmagatzemar les dades rebudes per Bluetooth i la funció processData() que processa les dades presses i provoca l'acció dels motors seguint l'ordre rebut. Al acabar s'envia un missatge conforme la placa està llesta per rebre una nova ordre.

```
void loop() {
  getSerialData();
  delay(1);
  processData();
}
```

- getSerialData():

  El funcionament d'aquesta funció és molt senzill. El primer que avalua és si s'ha rebut un missatge per part de l'ordinador amb la funció bluetooth.available(). Si s'ha rebut es guarda com un String a la variable x, i es crea una variable buff que servirà per emmagatzemar caràcters.

```
  void getSerialData(){
    if(bluetooth.available() > 0) {
      String x = bluetooth.readString();
      String buff="";
```

  Tot seguit es recorre l'string rebut i guardat a la variable x caràcter a caràcter per tal de separar-la per les comes, ja que aquest es el caràcter definit com a separador pel programa. Aquesta funció for que recorre la cadena emmagatzema els caràcter llegits a la variable buff fins a trobar el caràcter de separació ',' i els guarda a la matriu text en la posició cnt on cnt és un comptador que augmenta en una unitat cada cop que es guarda una valor a la matriu text. D'aquesta manera s'inicia en cnt=0, es guarda el primer valor quan està complet, s'incrementa cnt=1 i es guarda el segon valor complet, es torna a incrementar cnt=2 per emmagatzemar el tercer i últim valor i es reinicia el valor del comptador cnt abans d'acabar la el for.

```
      for (int i=0; i<x.length();i++){
        String caracter="";
        caracter=caracter+x[i];
        if (caracter!=","){
          buff=buff+x[i];
        }
        else {
          int y=buff.toInt();
          text[cnt]=y;
          buff="";
          if (cnt<3){
            cnt+=1;
          }
          if (cnt==3){
            cnt=0;
          }
```

```
        }
      }
      Rebut=true;
      delay(1);
    }
  }
```

Per acabar, es canvia el valor de la variable booleana Rebut a True, de manera que així pugui començar a treballar la funció processData() un cop llegit tot el missatge.

- processData():

  Un cop rebut i llegit el missatge només cal processar aquesta informació i enviar les ordres adients als motors amb la següent funció:

```
void processData(){
  if (Rebut==true){
    if (text[0]==0 and Boli.read()!=up){
      Boli.write(up);
      delay(300);
    }
    else if (text[0]==1 and Boli.read()!=down){
      Boli.write(down);
      delay(300);
    }
    posicio[0]=-text[1];
    posicio[1]=text[2];
    Robot.moveTo(posicio);
    Robot.runSpeedToPosition();
    bluetooth.println("Ready");
    Rebut=false;
  }
}
```

En primer lloc, només s'executa la funció si la variable Rebut val True, i per tant s'ha rebut un missatge complert. Un cop dins la funció es llegeixen els 3 valors de la matriu text, de manera que, si el primer valor (text[0]) val 0 i el retolador està en posició de dibuix s'aixeca, i si val 1 i esta aixecat es baixa. Es pot observar que després de l'acció del servo s'atura el programa 300 ms per tal d'assegurar la posició del retolador abans de començar amb el nou moviment. A partir d'aquí, es guarden els dos valors en una matriu que conté la posició dels motors, tenint en compte que es canvia el signe de la posició del motor dret per tal d'aconseguir que girin en el mateix sentit , ja que un motor està situat a la inversa de l'altre. Aquesta matriu s'envia als divers amb les ordres moveTo() i runSpeedToPosition() de la classe MultiStepper que realitzaran els moviments esperats dels motors.

Per acabar s'envia un missatge de retorn a l'ordinador amb la paraula "Ready" per tal de continuar amb l'execució de la següent ordre.

4.6 Bluetooth: Com es comunique l'ordinador i el robot?

4.7 Aplicació: Com el controlo?

Capítol 5

Conclusions i treballs futurs

Bibliografia