Universitat de Girona
Escola Politècnica Superior

Treball final de grau

Estudi: Grau en Enginyeria Informàtica

Títol:

Disseny i desenvolupament d'un robot submarí amb una Raspberry Pi – R2B2

Document: Memòria

Alumne: Ferran Veciana i Buixeda

Tutor: Jordi Freixenet i Xavier Cufí

Departament: ATC – Arquitectura i Tecnologia de Computadors

Àrea: ATC – Arquitectura i Tecnologia de Computadors

Convocatòria (mes/any): 06/2020

Índex

Índex de figures

Índex de taules

Capítol 1 Introducció

Motivació

En aquest apartat descriuré quins van ser els motius per els quals vaig decidir fer aquest projecte.

L'any passat vaig cursar el 4t curs d'informàtica, en la branca de robòtica. Sempre m'havien apassionat molt els robots i però no va ser fins a l'assignatura obligatòria de Robòtica de tercer que em vaig adonar que era un tema en el que volia aprofundir i aprendre molt més. Per això a quart em vaig decantar per la branca de robòtica.

Durant l'assignatura de Sistemes Empotrats vaig dissenyar i construir juntament amb un company, un cotxe amb el microcontrolador arduino i sensors, que era controlat a través de moviments de la mà. Va ser una experiència molt enriquidora i divertida i per això volia que el meu TFG es decantés cap alguna cosa semblant.

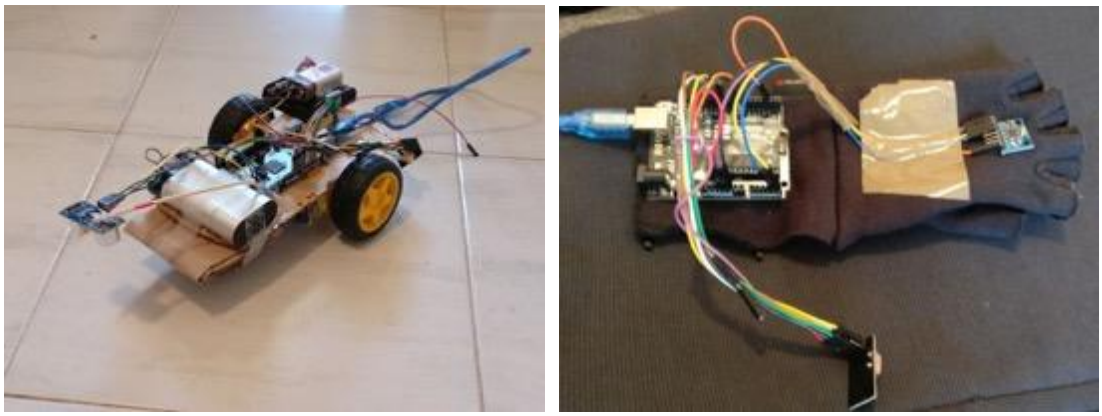


Fig. 1.1 Prototip final del HandBot desenvolupat a l'assignatura de Sistemes Empotrats

Buscant i pensant temes em vaig topar amb una proposta feta per els professors Jordi Freixenet i Xevi Cufí sobre el desenvolupament d'activitats de Disseny i Construcció d'un Robot Submarí teleoperat (ROV). Aquest projecte, tenia un doble sentit: el disseny i construcció del robot i poder preparar activitats per a nens i nenes amb ell. Durant dos estius he fet de monitor de lleure en un casal i sempre m'ha agradat molt la relació amb els nens i nenes. Aquestes són les principals raons per les quals vaig pensar que aquest projecte era perfecte per mi.

Objectius del projecte

Aquest TFG és la continuació del desenvolupament d'un robot submarí R2B2 dissenyat per poder-se imprimir en 3D, remotament operat i de baix cost. El projecte inicial va ser creat per un estudiant de màster a la universitat de Girona durant el curs 2017/2018. El resultat, tot i ser molt bo, era millorable i el projecte va quedar inacabat.

L'objectiu del meu projecte és redissenyar, millorar i construir el robot R2B2 perquè pugui ser treballat/programat i jugat per nens i nenes i d'aquesta manera apropar la robòtica als més joves.

El redisseny residirà principalment en canviar el "*core*" del robot d'un ESP32 a una Raspberry Pi per la seva potència, flexibilitat, fiabilitat, prestacions, preu i el seu bon reconeixement en l'àmbit de la informàtica i l'electrònica. Degut a l'augment del tamany del controlador, serà necessari també un canvi d'estructura del xassís, però també de l'electrònica i el software. Es vol afegir retransmissió i/o gravació de vídeo per tenir un control més bo i més divertit.

Per controlar el robot s'utilitzarà una app multiplataforma i/o a través d'un comandament físic com el de la PlayStation3/4 o creat de zero a partir d'un microcontrolador.

El model anterior va quedar inoperatiu, per tant, es parteix simplement del model i la documentació. El meu projecte final consistirà en redissenyar i reinventar el R2B2 afegint-hi les modificacions mencionades al paràgraf anterior.

En resum, els objectius principals que vam plantejar a l'inici del projecte van ser els següents:

- Redissenyar l'R2B2 seguint les premisses inicials
- Que sigui un robot de baix cost
- Que pugui ser replicat i construït
- Ha de poder transmetre vídeo en directe a una IP local.
- Ha de poder ser remotament operat i sense fils.
- Bateries i microcontrolador integrat.
- Crear o millorar una aplicació per poder controlar el robot
- S'ha de poder obrir amb facilitat per poder fer reparacions, canvis, millores, etc.
- Es recollirà material gràfic per documentar el projecte.
- Ha de ser robust
- Hem de tenir en compte l'experiència adquirida en el desenvolupament d'un projecte previ: Plataforma col·laborativa R2B2.

En la figura 1.2 es poden veure els robots R2B2 original (inspirat en el projecte SeaPerch), el R2B2nano (resultat del projecte de Pau Roura), el R2B2 Raspberry (resultat d'aquest TFG), i finalment el robot SPARUS dissenyat i desenvolupat per VICOROB.

Accessibilitat

Tot el codi i dissenys d'aquest projecte es troben penjats a GitHub (https://github.com/). GitHub és un servei de hosting de repositoris Git, el qual ofereix tota la funcionalitat de Git de control de revisió distribuït i administració de codi de la font així com afegint les seves característiques pròpies. Proporciona control d'accés i diverses característiques de col·laboració com bug tracking, administració de tasques, i wikis per cada projecte. S'utilitza principalment per a la creació de codi font de programes d'ordinador. Per poder descarregar els dissenys i programes s'ha de fer des del següent enllaç: https://github.com/ferrveciana/R2B2_RaspberryPi

Els dissenys 3D es troben dins la carpeta "Design", i dins la carpeta "Electronics", trobarem un esquema dels cablejats feta amb l'eina Frizing i dins la carpeta "Code", podem trobar tot el codi font del projecte, des de l'aplicació mòbil (Processing) fins al firmware del robot (Python).

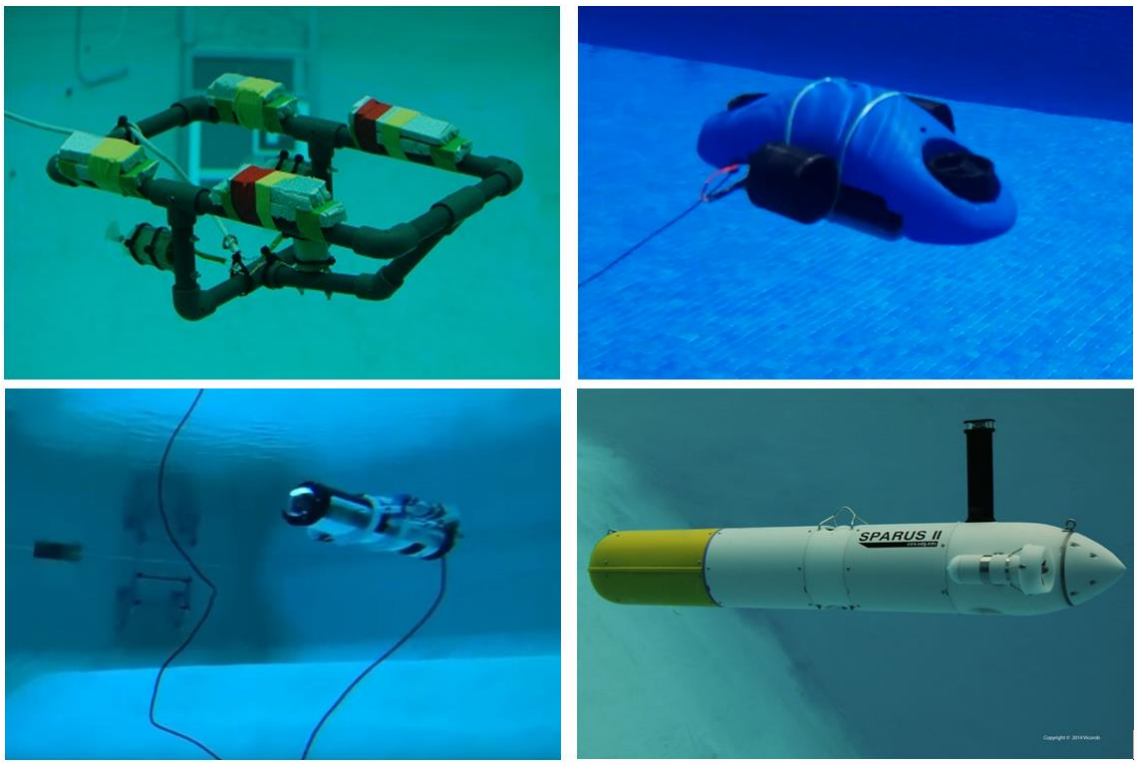


Fig. 1.2 R2B2 originial, R2B2 nano, R2B2 Raspberry i l'Sparus II AUV

Capítol 2 Disseny

Models de referència: R2B2 i R2B2 nano

El R2B2 és el nom d'un robot, i també el d'un projecte que neix de l'institut VICOROB fruit de la voluntat d'aproximar la cultura maker, el "do it yourself" (DIY) i "do it with others" (DIWO) per tal d'involucrar joves estudiants en enginyeria i ciència, i consisteix en dissenyar, construir i conduir un ROV (Remotely Operated Underwater Vehicle).

El R2B2 està inspirat en el Sea Perch del MIT (Massachusets Institute of Technology) que des del 2003 professors i estudiants treballen junts per crear les seves versions del ROVs i els utilitzen, per exemple, en missions de recerca per comprovar la qualitat de l'aigua. Aquests projectes obren la porta a molts estudiants per motivar-los a cursar carreres de robòtica, enginyeria, ciències maries, etc. Més informació sobre aquest projecte original es pot trobar a: https://www.seaperch.org/index

Des de l'UDG es fan tallers on estudiants d'entre 12 i 16 anys d'arreu de Catalunya, desenvolupen durant tres dies el seu propi R2B2 utilitzant material low-cost. Durant el taller s'enssenyen els principis matemàtics, físics i de robòtica necessaris per poder completar el projecte

Fig. 2.1 Model de l'R2B2 fet amb tubs de PVC

El meu projecte parteix d'un treball previ, el treball que va fer un exalumne d'aquesta universitat durant el curs 2017/2018. El treball de final de màster d'en Pau Roure consisteix en crear una versió de l'R2B2 dissenyat per poder-lo imprimir en 3D, que fos remotament operat i sense fils, i que tinguin un cost baix.

Fig. 2.2 El robot submarí R2B2nano ideat i desenvolupat per Pau Roura

El resultat va ser bo, i va aconseguir els objectius proposats, però era millorable. El robot anava lent, i al estar tot cobert amb resina per aconseguir l'estanqueïtat necessària, era molt difícil fer cap canvi sense haver de refer-lo tot de nou. Utilitzava un ESP32 (https://www.espressif.com/en/products/hardware/esp32/overview) com a microcontrolador i es recarregava de manera inalàmbrica a través d'una base de càrrega també impresa en 3D.

Evolució del disseny

En aquest apartat explicaré de manera resumida tota l'evolució del disseny: les idees inicials, com vaig arribar al primer model, el que funcionava i el que no, i com va evolucionar fins al prototipus final. Trobo necessari ressaltar que degut a vàries dificultats que em vaig anar trobant, vaig acabar construint dues versions. Molt del temps dedicat a aquest TFG va ser invertit en la primera versió, i és per això que he decidit incloure-la en el document. Ha estat un temps d'aprenentatge. Al llarg del projecte intentaré diferenciar-les al màxim per evitar confusions.

A mode de resum del disseny faig una petita presentació de tots els components que té la meva versió de l'R2B2. El ROV està dividit en parts bastant diferenciades: En primer lloc, tenim l'estructura exterior, que està formada per l'encapsulament, els suports per els motors, els quatre motors (i hèlixs), els cables, i l'antena amb la boia. Per altre banda, l'estructura interior consisteix en un microcontrolador, les bateries, els drivers de motor, la càmera, cables, i pesos per equilibrar-lo dins l'aigua. Tots aquests components estan explicats al llarg del treball.

OpenSCAD

Per el disseny 3D del robot he utilitzat una eina de codi lliure anomenada OpenSCAD. No és un editor interactiu sinó un compilador 3D basat en un llenguatge de descripció textual. Això pot dificultar-ne l'aprenentatge al principi però és molt útil ja que pots

parametritzar tots els valors. El que això permet és que si mai es canvia la mida del motor, simplement caldria modificar un valor i tota la resta del disseny s'adaptaria a aquest canvi.

Tots els dissenys són fets amb aquesta eina i estan penjats al GitHub del projecte.

Model 1

Estructura exterior

El més important alhora de construir un submarí és l'estancaitat. Això és imprescindible ja que a l'interior hi anirán el controlador, les bateries, etc. La idea inicial per la carcassa era posar els components electrònics dins un *tupper* i fer-hi forats per poder treure els cables. D'aquesta manera aconseguim una millora important respecte el model R2B2nano en el sentit que feia possible obrir la carcassa per arreglar o modificar qualsevol cosa en qualsevol moment. Tot i això, vam considerar que aquesta solució era poc acurada, ja que no assegurava una impermeabilitat del cent per cent. Per aquesta raó ens vam acabar decantant per utilitzar un recinte estanc de BlueRobotics, una reconeguda empresa americana que fa productes per robòtica submarina. Si bé aquesta decisió va augmentar el cost del projecte, es va aconseguir una carcassa per tota l'electrònica totalment hermètica i impermeable.

Al centre de robòtica submarina de Girona CIRS, ja havien utilitzat aquestes carcasses d'acrílic abans (termoplàstic rígid i transparent, veure figura 2.3), i va ser útil perquè vaig poder inspirar-me amb els seus models alhora de dissenyar el nou robot.

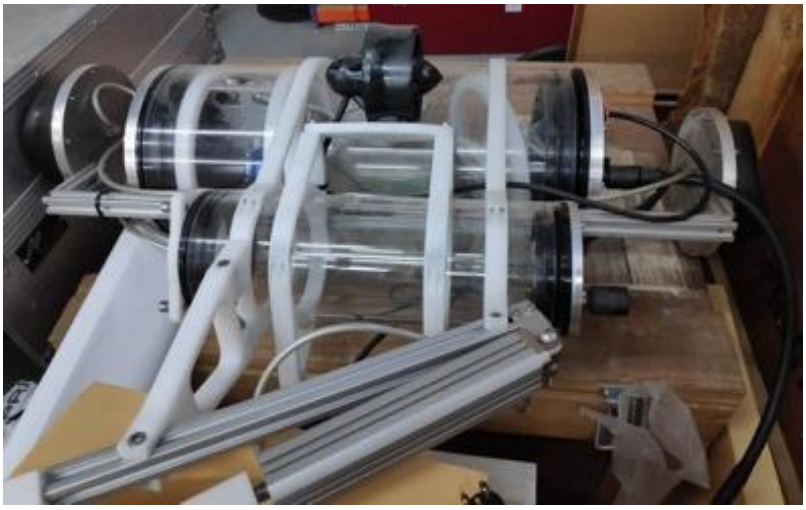


Fig. 2.3 Estructura submarina de proves creada amb material de BlueRobotics al CIRS

El primer model que vaig dissenyar tenia quatre puntes. En cada una hi aniria un motor, dos en direcció horitzontal i dos més orientats verticalment. Estarien col·locats de tal manera que el flux d'aigua no fos interceptat per el cos del robot. El model tenia varis

problemes: Fora de l'aigua era difícil de col·locar sense trencar res, els motors s'havien d'enganxar d'alguna manera sense que les hèlixs toquessin ni el cos del robot ni l'esquelet exterior. (Fig. 2.4)

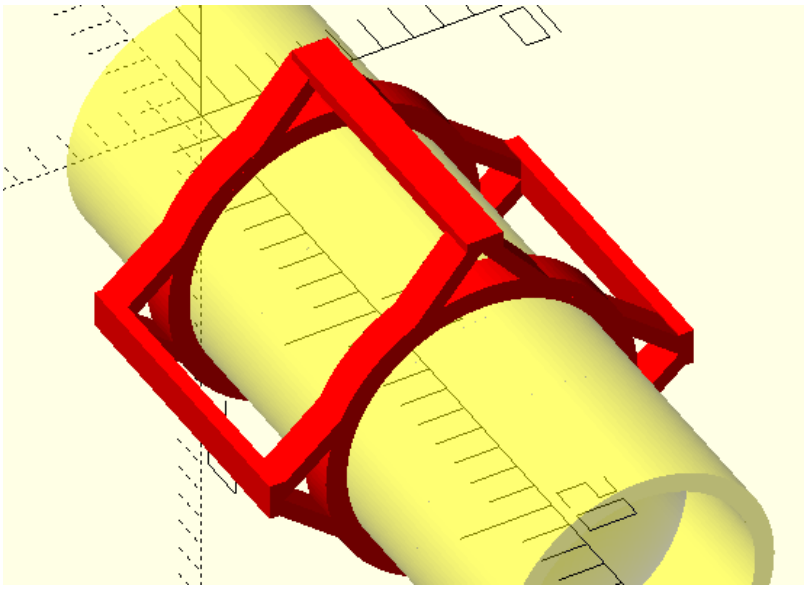


Fig. 2.4 Primer model 3D

Una idea posterior va ser posar-li una mena d'ales o aletes al robot. On els motors verticals anirien en un forat al mig i els horitzontals, col·locats a la punta de l'aleta. D'aquesta manera el flux d'aigua no quedava pràcticament interromput, era fàcil de col·locar, i els motors eren fàcilment intercanviables en cas que algun s'espatllés. (Fig. 2.5)

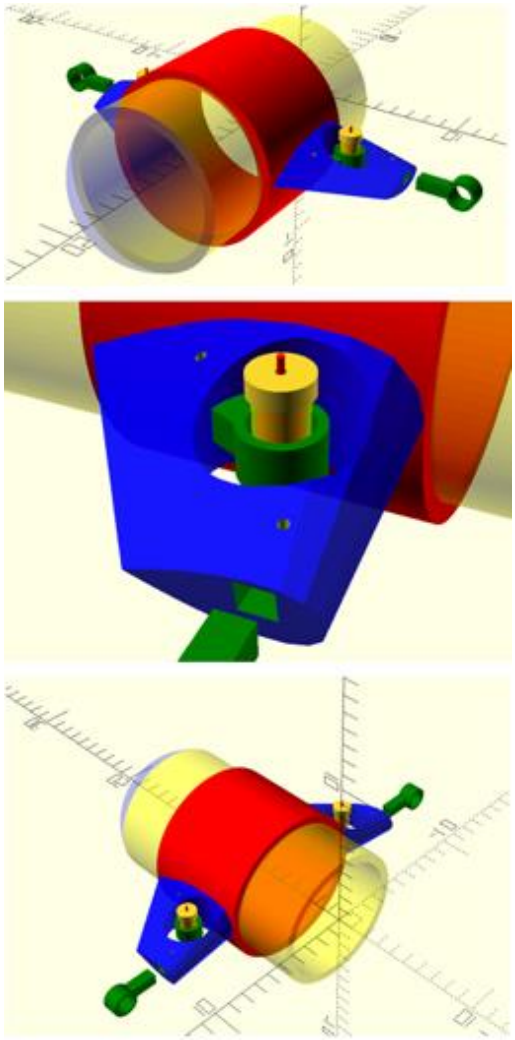


Fig. 2.5 Disseny amb aletes del ROV

Vam poder consultar i contrastar les idees d'aquests dissenys amb varis investigadors del CIRS, i el disseny va agradar bastant, tot i que els va preocupar el tema de resistència dels materials, ja que la intenció inicial era imprimir-ho tot en 3D. La impressió 3D actual, al estar feta a làmines ofereix poca resistència, a més, moltes de les peces tenien mides rondant al 1cm de diàmetre; per tant, hi havia una alta probabilitat a que alguna de les peces es trenqués durant l'assemblatge o durant la utilització del robot.

Ens van recomanar que utilitzéssim peces amb un plàstic molt més resistent i tallades amb una fresadora. Vam utilitzar la fresadora que es troba al CIRS.

Volíem mantenir el disseny de les ales és per això que vam fer una mena d'esquelet del disseny anterior. Cal tenir en compte que una fresadora no té tanta flexibilitat alhora de fer peces, ja que un dels costats de la peça sempre serà pla (costat que està de cares a la taula mentre es fresa).

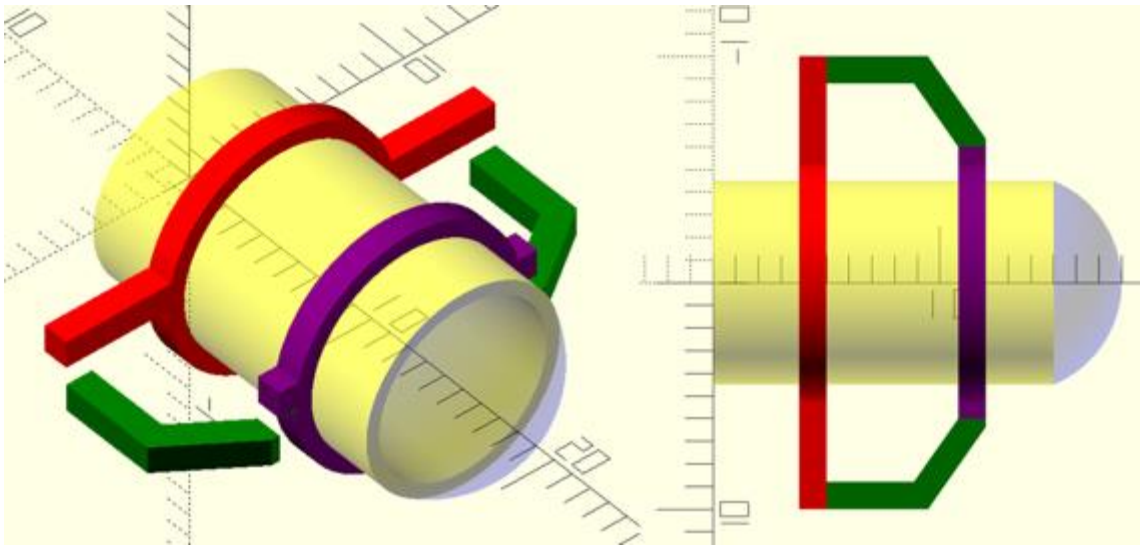


Fig. 2.6 Disseny més simple, pensat per fer-lo amb una fresadora

El disseny de la figura 2.6 estava format per tres peces i els motors anirien subjectats amb brides. Més endavant, però, per millorar el flux d'aigua, vam posar dues peces més per aguantar els motors verticals.

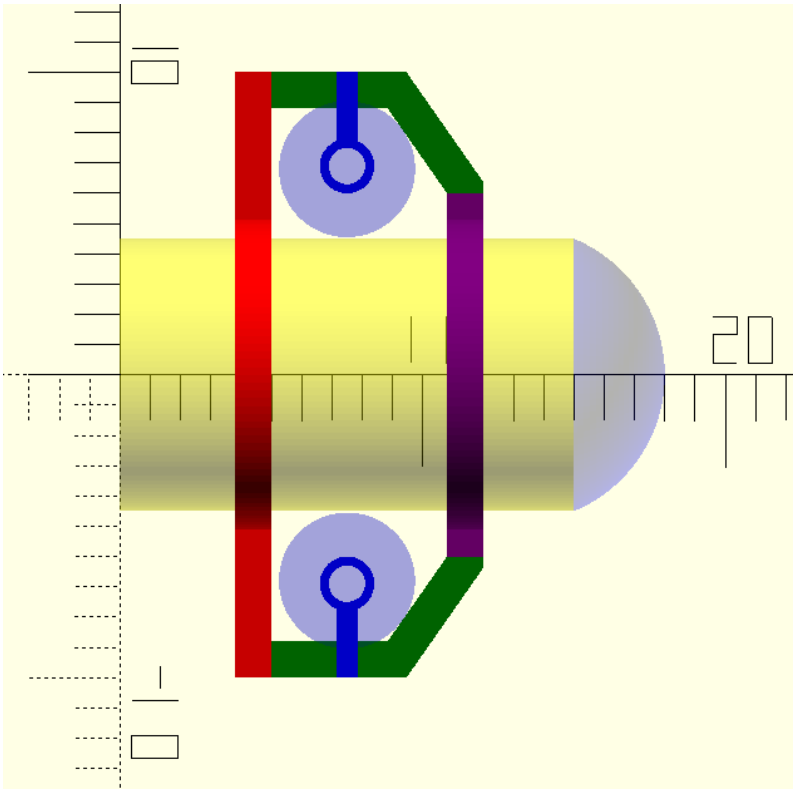


Fig. 2.7 Vista en planta del disseny, on l'àrea blau clar representa el diàmetre de les hèlixs.

Per poder enganxar la peça blava és necessari rebaixar la part coincident de la peça verda. D'aquesta manera les dues parts encaixaran, i un sol cargol mantindrà els motors immòbils.

Es va utilitzar la fresadora del CIRS. Algunes de les modificacions de les peces van ser fetes directament en el programa de la fresadora. Ja que, al acceptar només formats de fitxer 2D, tots els rebaixats del material s'havien de parametritzar *in situ*. (Fig. 2.8)

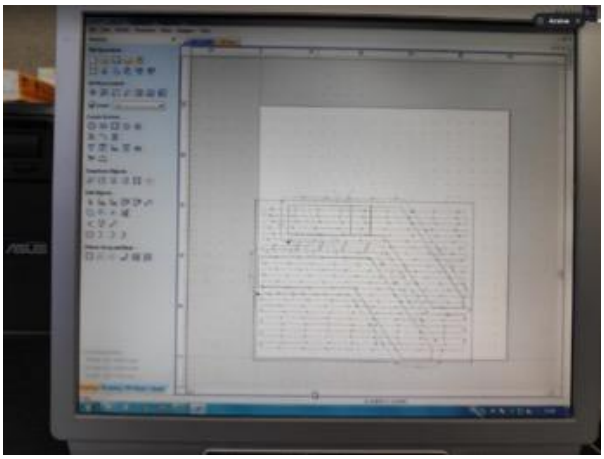


Fig. 2.8 Foto del control de la fresadora

Estructura interior

L'estructura interna del tub d'acrílic també va anar evolucionant, al igual que l'estructura exterior, a mesura que avançava el projecte.

La idea inicial era tallar el tub d'acrílic de manera que quedés tot comprimit i utilitzéssim el mínim espai possible. Després d'una petita reunió amb investigadors del CIRS, vam decidir que no era necessari, ja que la fricció que feia el submarí al avançar endavant era la mateixa independentment de la llargada del tub. A més, el pes i l'espai de les bateries dificultava molt l'assemblatge i flotabilitat.

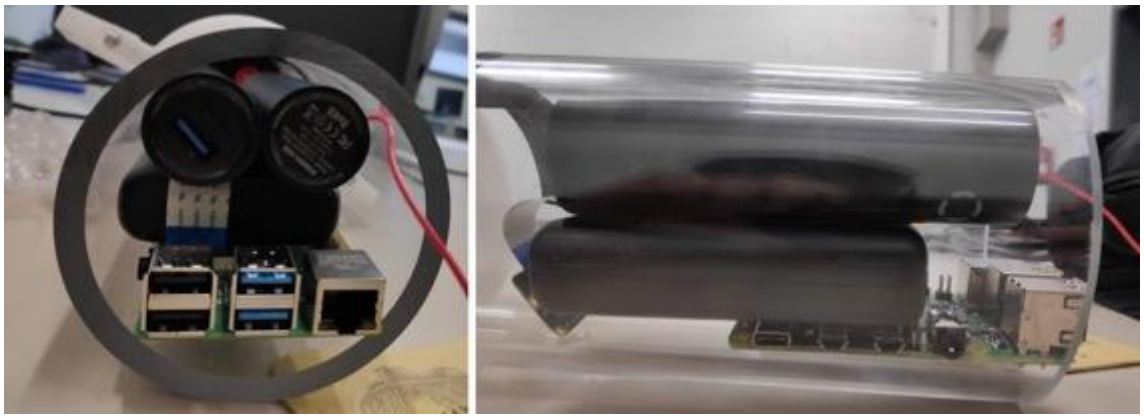


Fig. 2.9 Aspecte inicial de l'Interior del ROV

Un altre de les característiques que volíem per el nostre R2B2 és que fos fàcil de treure i posar els components de l'interior per poder fer modificacions, recàrrega de bateries,

extreure'n dades, etc. BlueRobotics ven a la seva pàgina web una safata on posar tota l'electrònica i collar-la, però buscàvem alguna solució més econòmica.

Fig. 2.10 Captura de pantalla de la web BlueRobotics

Una de les idees que vam tenir va ser posar una cartolina per tot l'interior, i posar-hi tota l'electrònica a sobre. D'aquesta manera, per treure o posar tots els components dins el recipient només caldria estirar la cartolina. Els avantatges eren el preu, el pes, que ocupava poc i a més, en cas que entrés aigua, l'absorbiria i serviria per detectar el problema. (Fig. 2.13)

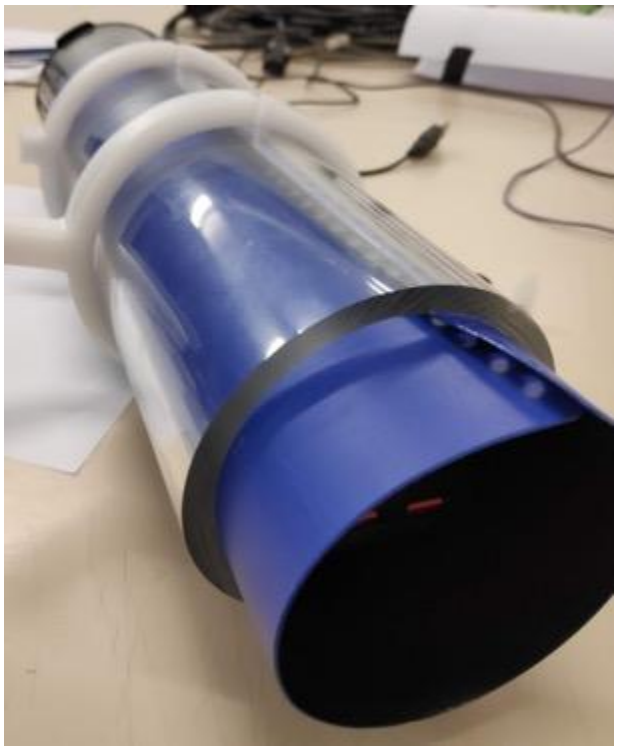


Fig. 2.11 Prova utilitzant una cartolina (en aquest cas de plàstic)

Finalment, però, vam decantar-nos a usar una planxa de metall on hi reposarien tots els components de manera totalment rígida, formant un sol bloc. Era una solució molt més simple i elegant, que afegia una mica més de pes, el qual ja necessitava per tal d'aconseguir flotabilitat zero. Alhora d'estructurar l'interior era necessari també

distribuir bé els pesos. En alguns punts, fins i tot, vaig afegir llast (plom) per tal poder equilibrar el robot. (Fig. 2.14)

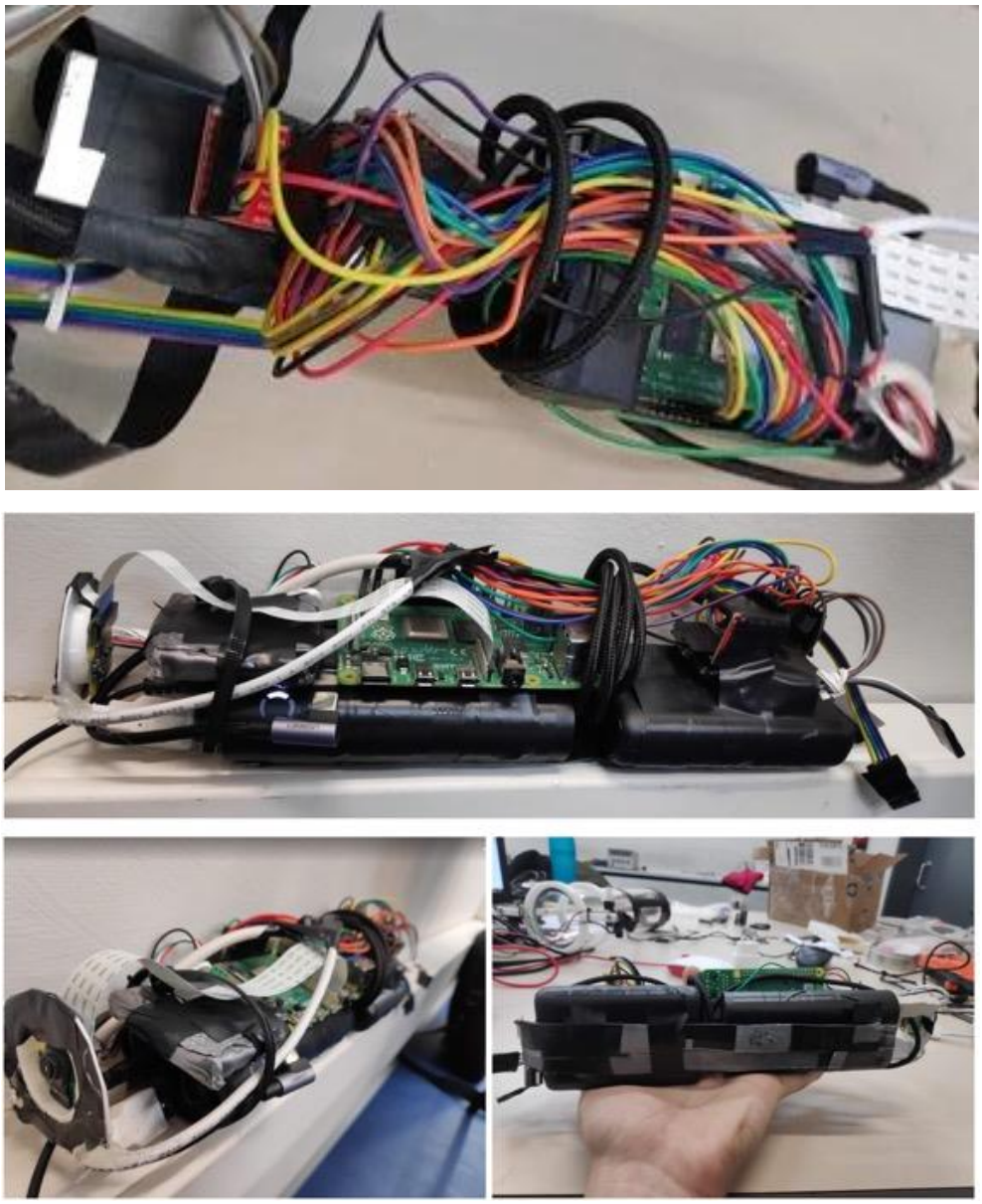


Fig. 2.12 Interior finalitzat en el model 1

A partir d'aquest punt tot el disseny anava basat en motors petits de drone de menys de 5 volts. Eren els motors que havia utilitzat el projecte R2B2nano. Són motors que mostren un comportament molt bo en l'aire, però com veurem més endavant, no gaire bo a sota aigua. El disseny de unes hèlixs bones va ser complicat i es va allargar (Veure Anex 1 – Avantatges i desavantatges eix motor). En tenir-les fetes i col·locades vam provar el seu rendiment fora l'aigua, i dins l'aigua. El resultat va ser una mica decepcionant ja que, al ser les hèlixs tant grans, el motor es quedava sense gaire força dins l'aigua. Tot i que funcionaven, el robot es movia molt a poc a poc. Va ser en aquest

punt en que vam decidir que era necessari un canvi en el disseny, i es pensa en una segona versió del prototipus.

El model 1 és tot el que hem vist fins ara en aquest apartat. A continuació explicaré el disseny i canvis que vaig dur a terme per poder seguir endavant amb el projecte i aconseguir un millor comportament del robot.

Fig. 2.13 Resultat final en la Versió 1

Model 2

Estructura exterior

Es van canviar els motors per els que s'utilitzen en la versió de tubs de PVC de l'R2B2. Són motors de 12-24V que sabíem segur que funcionaven bé dins l'aigua. També vam utilitzar dos motors que havia comprat per Ebay i que declaraven ser imparmeables. D'aquesta manera investigàvem també noves possibilitats. Els suports pel motor que havia fresat van quedar inoperatius. Es va fer un redisseny de l'esquelet, adaptat als nous motors. Tot i això, no es va arribar a dur a terme ja que, abans de fer-lo "bonic", volia aconseguir un bon funcionament del robot.

Fig. 2.14 Motors resistents a l'aigua d'Ebay (esq) - Motors encapsulats utilitzats per l'R2B2 (dret)

Posar en funcionament aquests motors era més senzill ja que la mida de les hèlixs era molt més adequada, l'únic inconvenient era que les bateries utilitzades per fer anar motors a 5 V no eren suficients per fer anar aquests altres.

Per aquesta raó vam decantar-nos per unes bateries Li-Po. Són les més utilitzades en projectes de radio control i hi ha una gran varietat de potències i capacitats. Com havia pogut comprovar en els primers tests sota l'aigua, el pes no era un problema, ja que per poder-lo equilibrar i aconseguir flotabilitat zero, vaig haver d'afegir pesos dins el robot. Per això vaig buscar la bateria 3S (11.1V) amb més capacitat que pogués cabre dins el cilindre.

Una bateria de 6000mhA duraria aproximadament unes 6 hores en funcionament constant ja que el consum estàndard (2 motors al 80 %) és de +-1 A. De totes maneres, no és bo descarregar totalment una bateria LiPo ja que es podria fer malbé. La bateria per la Raspberry també duraria, més o menys, un temps semblant aquest.

Estructura interior

Internament el que va canviar en la versió 2 van ser les bateries per els motors. Enlloc d'utilitzar dues powerbanks 5V 2.4A, es va utilitzar una bateria LiPo de 6000mah 11.1V (3S). En aquest sentit, per tant, l'estructura interior no va canviar massa.

Fig. 2.15 Connexions amb la bateria LiPo

Capítol 3 Disseny Hardware

Components

En aquest apartat es presenten tots els components electrònics utilitzats al llarg del projecte per la construcció del robot. També s'explica, en algun cas, les raons per les quals s'ha decidit utilitzar-lo.

RaspberryPi 4

El *core* del robot vas ser un dels canvis principals que vam dur a terme respecte la versió de l'R2B2 nano. Volíem més potència i flexibilitat. També volíem poder instal·lar una càmera i la RaspberryPi oferia l'opció més senzilla. A més, després d'una reunió amb en Pau Roura, vam concloure que si volíem acabar el projecte amb 3 mesos seria el millor (era el pla inicial). L'altre opció requeria d'imprimir les plaques PCB (*P*rinted *C*ircuit *B*oard), i això segons l'experiència d'en Pau en el seu projecte, va comportar que el temps per dissenyar, enviar i rebre-les plaques a la universitat fos de com a mínim un mes.

El Raspberry Pi [5] és un ordinador monoplaca o SBC (*Single-Board Computer*) de baix cost desenvolupat en el Regne Unit per la Fundació Raspberry Pi. L'objectiu principal d'aquest disseny és estimular l'ensenyament de les ciències de la computació, però també s'ha popularitzat com a plataforma per a dissenys d'aficionats i per a usos informàtics generals. Els primers models es van començar a comercialitzar el febrer de 2012. El darrer model, Raspberry Pi 4, va sortir el juny del 2019 i compta amb un processador de quatre nuclis, 802.11ac Wi-Fi, Bluetooth 5, etc. S'alimenta a través de un USB-C.

El driver dels motors

Per poder comandar els motors des de la raspberryPi ens fan falta *drivers*. En el nostre cas les operacions que volem fer és arrancar i parar cadascun dels motors, i també poder girar cap un costat i cap a l'altre, independentment. A més, a diferència del R2B2 on els motors funcionen amb "tot o res", en el nou robot volem que l'usuari pugui especificar la potència dels motors amb un PWM (P*ulse-Width Modulation*). TB6612FNG[7] és un *driver* IC (*Integrated Circuit*) per motors de corrent continua amb output transistor en LD MOS estructura amb low ON-resistor. Té dues senyals d'entrada, IN1 i IN2, es poden triar entre els quatre modes com CW (agulles del rellotge), CCW (contrarellotge), fre curt i aturada.

Bateria per la Raspberry Pi 4

La Raspberry Pi 4 s'alimenta a través d'un USB-C i, idealment, necessita 15.3Watts de potència (5.1V / 3.0A). Amb la bateria Charmast Mini PwerBank 10400mAh veure Figura 3.1)[11] aconseguim aquesta potència i una autonomia de aproximadament 5-6 hores.

Fig. 3.1 Charmast Mini PowerBank 10400mAh

Antena

La Raspberry Pi 4 ja porta un mòdul WiFi incorporat, però dins l'aigua, les ones electromagnètiques queden atenuades molt ràpidament, i és per aquesta raó que es necessita un cable fins la superfície per portar el senyal WiFi. En aquest sentit, ens plantegem que el robot porti una boia. L'antena consistirà d'un cable allargador USB[11] que sortirà del robot i que anirà fins a la boia, que flotarà en la superfície de l'aigua. A la boia, connectada al cable, hi haurà l'adaptador WiFi (veure Figura 3.2)[11] i que farà de punt de connexió entre el robot i un mòbil, que ens permetrà controlar el robot remotament. Com que el model 4 de la Raspberry Pi no permet soldar una antena externa, vaig buscar una solució alternativa. Comprant un mòdul WiFi extern, que es connecta per USB, i un allargador USB mascle-femella, podia portar aquesta senyal fàcilment fora l'aigua.

Fig. 3.2 Antena WiFi per USB

Fig. 3.3 Allargador USB, mascle-femella

Motors del robot

Al començament del projecte vam pensar en utilitzar motors de drone, però que més tard van demostrar que no tenien un comportament bo a sota aigua. Van ser la nostra primera opció i durant el TFG vaig treballar-hi per poder millorar-ne el comportament. Al principi teníem clar que en el robot d'en Pau no havien funcionat correctament (el robot també es movia lentament) per les hèlixs. En el R2B2 les hèlix es van imprimir amb impressores 3D seguint models no estandards, ja que es podia fàcilment triar el nombre de pales de les hèlix, també la forma, i altres paràmetres, però no havíem consultat cap expert en nàutica. Per això la nostra hipòtesi de partida va ser utilitzar els motors de drone, que eren low cost i amb un comportament molt bo, i unes hèlix comercials, com les que utilitzem en el R2B2 original. Les característiques dels motors són els següents:

Fig. 3.4 8.5x20mm Brushed motor 15,000KV

Una de les coses que vaig buscar alhora de triar motors va ser que tingues els KV baixos. KV es refereix al nombre de revolucions per minut quan només s'hi aplica 1 Volt. Com més alt sigui aquest valor, més ràpid anirà però no tindrà ni tanta acceleració ni tant de parell com un amb menys KVs. Aquest model té 15.000KV.

Com que aquests motors de dron petit no van ser prou potents per desplaçar-se dins l'aigua, vam canviar a uns de més potència: el model N2738-125 [13]. Amb una tensió nominal de 12-24V DC i consum de corrent mitja de 0.35 A i una força de torsió (parell) de 5N mm, són els utilitzats per crear els R2B2's originals. En aquest cas, però, no són alimentats per cable a través d'una font d'alimentació.

Fig. 3.5 Motor N2738-125, de 12-24V

Bateries pels motors

En el primer model, per alimentar els motors de dron vam utilitzar dues bateries POWERADD EnergyCell de 5000mAh[14] col·locades en sèrie.

Fig. 3.6 POWERADD EnergyCell PowerBank 5000mAh

En la segona versió del robot va ser necessari un canvi de motors per uns de més voltatge (12V). Aquest canvi resultar en un canvi en l'alimentació. Es va utilitzar la bateria Li-Po Li-Po 3S, 50C, 11.1V, 6000mAh[15] de la figura 3.7.

Fig. 3.7 Bateria Li-Po 3S, 50C, 11.1V, 6000mAh

Les bateries LiPo són les més utilitzades en projectes RC. A diferència de les típiques bateries NiMH, les LiPo poden tenir capacitat més alta, pesen menys, i velocitat de descarrega és molt més elevada. Això últim permet que puguin alimentar a varis motors sense perdre potència molt més fàcilment. Tenen una vida útil més baixa, només 150-250 cicles, són sensibles i es necessita atenció i cura alhora de carregar, descarregar i emmagatzemar.

Aquestes bateries estan formades per cèl·lules. Cada cèl·lula té un voltatge nominal de 3.7V. Per obtenir una bateria de 11.1V és necessari posar 3 cèl·lules en sèrie (3S). Alhora de carregar s'ha de vigilar que totes les cèl·lules tinguin el mateix voltatge. Això ajuda

en el rendiment de la bateria, tot i que també és crucial per raons de seguretat. És per això que és pràcticament obligatori utilitzar un carregador de balanceig. Aquest s'encarrega de carregar les bateries i mantenir-ne l'estabilitat.

El voltatge per cèl·lula de 3.7v no és la càrrega màxima. Una cèl·lula totalment carregada té 4.2V. Si el voltatge baixa de 3.0V, la cèl·lula deixa de ser estable i el seu ús seria perillós.

Més informació sobre bateries LiPo: https://rogershobbycenter.com/lipoguide [6]

Cameres

Inicialment la càmera utilitzada era el mòdul V2 de 8MP[16], una càmera standard amb un connector especial per raspberrys. Més endavant, per fer un experiment de seguiment d'un objecte de color amb el ROV, vaig decidir canviar de càmera i utilitzar-ne una amb gran angular[17].

Fig. 3.8 Mòdul V2 8MP - Mòdul amb lents d'ull de peix

El mòdul d'ull de peix permetia un rang de visió molt més gran i feia molt més senzilla la tasca de fer tracking d'un objecte. A l'apartat de Tests de seguiment d'un objecte s'explica en més detall les avantatges i desavantatges de les dues càmeres.

Fig. 3.9 Visió des del mòdul V2 8MP – Usant el mòdul amb lents d'ull de peix

Esquema de connexions del robot

L'esquema de cablejat entre els components electrònics final del ROV és el següent:

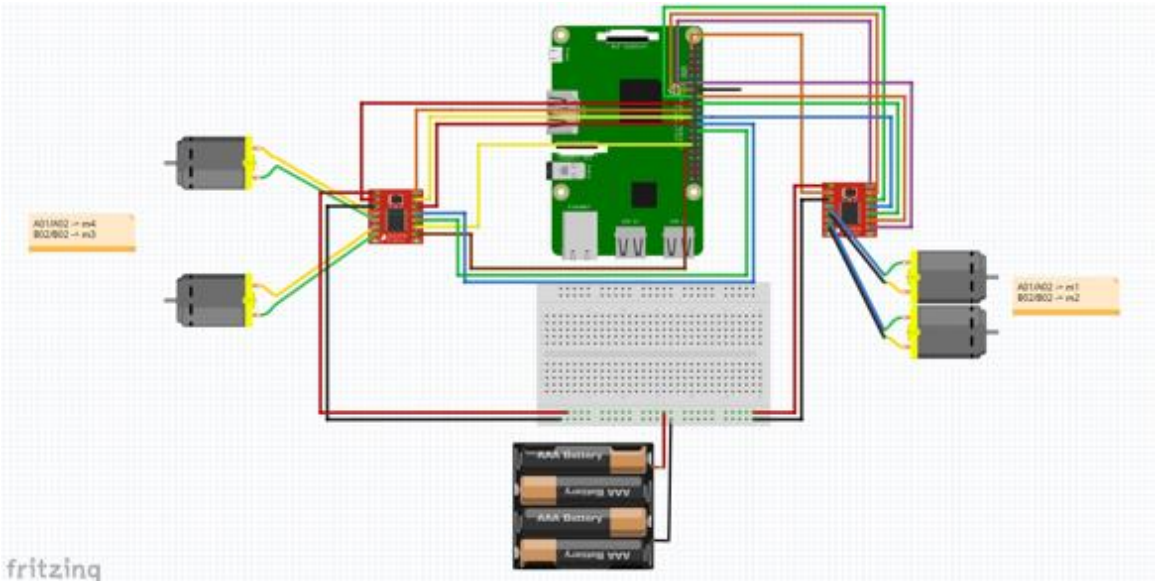


Fig. 3.10 Esquema del cablejat electrònic fet amb l'eina Fritzing

La placa de color verd gran és la RaspberryPi 4; en vermell, més petit, els dos motorDrivers TB6612FNG; els motors (de color gris), i la bateria pels motors (representada amb quatre piles alcalines). L'esquema es troba també penjat a GitHub.

Capítol 4 Disseny i desenvolupament de software

En el següent apartat explicaré amb detall els entorns de programació, el protocol de comunicació utilitzat i el seu funcionament, les noves funcionalitats programades respecte la versió de l'R2B2 nano, un petit resum de com les he implementat i també comentaré alguns dels problemes més importants que m'he trobat alhora de desenvolupar el projecte.

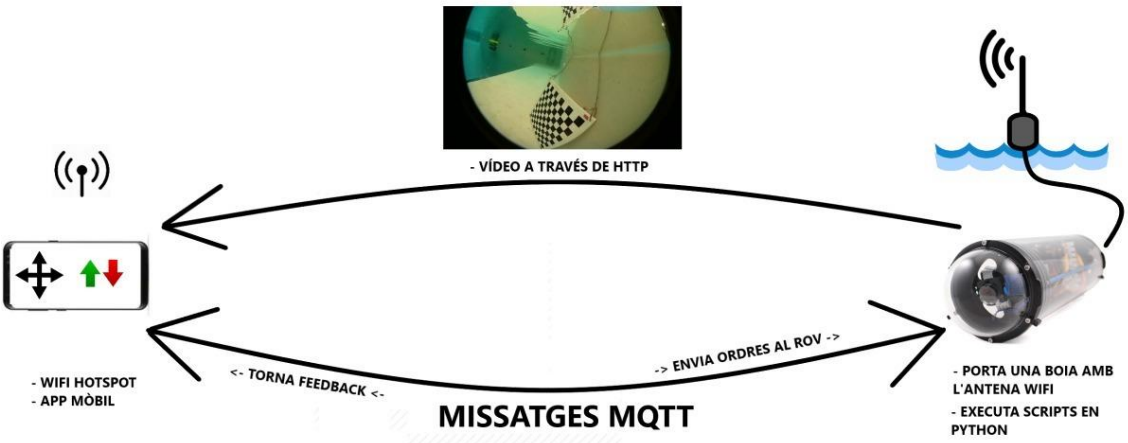


Fig. 4.1 Esquema general del fluxe de dades i els softwares que intervenen en el control del robot Raspberry Pi – R2B2

Entorn (RaspberryPi i IDEs)

Un dels grans canvis respecte a les altres versions de R2B2 va ser posar com a microcontrolador del robot, una RaspberryPi 4. El llenguatge de programació que vaig decidir utilitzar va ser el Python. Els motius van ser varis, en primer lloc, Python és el llenguatge oficial de la Raspberry Pi. Això implica que un gran percentatge dels exemples i documentació que trobem per internet estan fetes en aquest llenguatge. En segon lloc, és un llenguatge que s'està posant molt de moda i l'havíem utilitzat molt poc a la carrera, i per tant, m'interessava aprendre'n. Un dels principis que he mantingut al llarg de tot el projecte és no refer el que ja s'havia fet en la versió de l'R2B2 nano, i que pogués servir pel nou robot. L'R2B2 nano utilitzava l'Arduino IDE, i per tant, poca cosa es pot reutilitzar, però per tal de mantenir el codi de l'App mòbil, calia mantenir el protocol de comunicació MQTT. Python disposa de totes les llibreries necessàries per fer-ho, a més d'altres pel control de motors, execució de comandes Linux i Powershell, etc. Per tant el llenguatge em donava tot el suport necessari.

Com a IDE (*Integrated Development Environment*) he utilitzat el Geany, un editor de text de codi lliure.

Per el desenvolupament de l'app he utilitzat el ProcessingIDE 3. Processing és un llenguatge de programació i entorn de desenvolupament integrat de codi obert basat en Java. És de fàcil utilització i té suport per Android.

La raó principal per la qual vaig decidir utilitzar aquesta aplicació va ser perquè era la que s'havia utilitzat en l'R2B2 nano i així es podia reutilitzar bona part del codi. A més, per crear una aplicació Android senzilla i executable, és molt pràctic.

L'única cosa que s'ha de fer és instal·lar el mode Android, clicant el desplegable de dalt a la dreta on posa Java. Cliquem "*Add Mode*..." i en el cercador que ens apareix busquem "*Android Mode*" i l'instalem. A més necessitarem també les llibreries de MQTT i de *IpCapture*. Per fer-ho anem al menú superior i seleccionem *Sketch -> Import Library -> Add Library.* Igual que amb els modes, la busquem, la seleccionem i la instal·lem.

En el cas de la llibreria de *IpCapture* serà necessari substituir el *.jar* que es troba dins la carpeta (a Windows): Documents\Processing\libraries\IPCapture\library, per el que hi ha al GitHub. Es poden veure més detalls en l'apartat de càmera streaming.

Protocol de comunicació MQTT

MQTT (*Message Queuing Telemetry Transport*) és un protocol de missatgeria de publicació / subscripció, extremadament senzill i lleuger, dissenyat per a dispositius restringits i xarxes amb poc ampla de banda, alta latència o poc fiables. Els principis de disseny són minimitzar l'amplada de banda de xarxa i els requeriments de recursos del dispositiu, alhora que s'intenta assegurar la fiabilitat i cert grau de lliurament. Aquests principis també resulten ideals en el món emergent de IoT (*Internet of Things*).

Disposa de qualitat del servei QoS 2 i per tant s'assegura que el missatge sigui rebut. En quan a seguretat, disposa de transport SSL/TLS i autentificació per usuari i contrasenya o mitjançant un certificat.

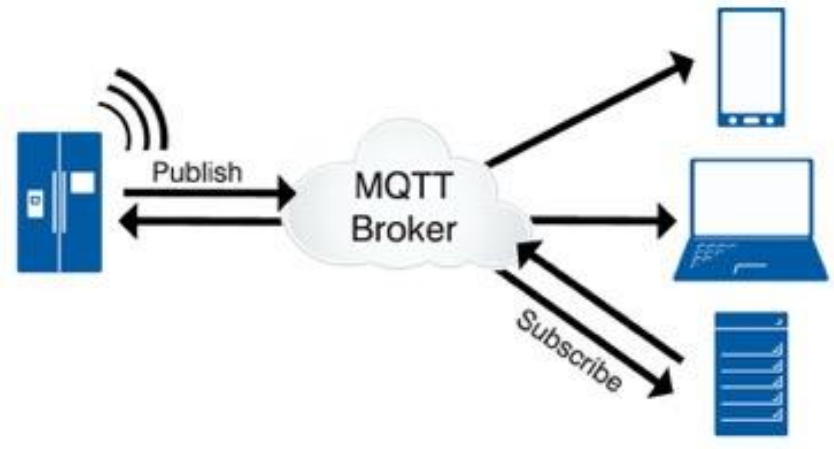


Fig. 4.2 Esquema del protocol MQTT

Per entendre el funcionament d'aquest protocol primer cal identificar-ne les parts. MQTT té un *Broker* que actua com a oficina de correus. MQTT no fa servir adreces sinó tòpics, i tothom que vulgui rebre missatges s'haurà de subscriure a un tòpic.

Els clients es poden subscriure a un o varis tòpics i també podran publicar a qualsevol tòpic. El sistema de tòpics té una estructura d'arbre, és a dir, que et pots subscriure a "esports/pilota" i rebries missatges de "esports/pilota/futbol, esports/pilota/basquet, etc.

En el meu cas, el broker es troba a la Raspberry Pi 4. Tant l'aplicació mòbil com la Raspberry seran clients d'aquest broker, i per tant, podran publicar i rebre(subscriure's) missatges.

El protocol té molta flexibilitat i permet afegir per exemple, un altre ordinador/microcontrolador, que executi un altre script i enviï el resultat o les accions a fer a l'R2B2 directament. Això pot ser molt útil durant els testos de noves funcionalitats, per exemple, el seguiment d'un peix, ja que vaig poder executar el programa des de un portàtil on veia a temps real el que veia el robot, la segmentació de color i la acció resultant.

Exemple d'un "*set up*" bàsic del protocol i funcions més utilitzades:

```
client = mqtt.Client() #creem nova instància
client.connect("192.168.1.45", port=1883, keepalive=60) #connectem al broker
client.on_message = on_message #definim funció on podem tractar cada missatge rebut
client.on_connect = on_connect

client.subscribe("r2b2/7") #per subscriure'ns en un tòpic
client.publish("r2b2/7/sensors","Missatge") #enviem "Missatge" a tothom que estigui
subscrit al topic

client.loop_start() #evitem que el programa acabi i podem seguir reben i enviant
missatges.
```

Definició de la funció executada quan ens connectem.

```
def on_connect(client, userdata, flags, rc):
    client.subscribe("r2b2/7") #d'aquesta manera si perdéssim la connexió en algun
moment al reconnectar-nos ens tornaríem a subscriure.
```

Definició de la funció per tractar els missatges rebuts:

```
def on_message(client, userdata, msg):
    print(msg.topic+" "+str(msg.payload)) #imprimim el missatge rebut
```

Llista de missatges MQTT programats i utilitzats:

Missatge	Utilitat
"mup "+pwr	Robot va amunt amb potència "pwr"
"dwn "+pwr	Robot va avall amb potència "pwr"
"rt+ "+pwr	Robot gira a la dreta amb potència "pwr"
"rt- "+pwr	Robot gira a la l'esquerre amb potència "pwr"
"mFF "+pwr	Robot va endavant amb potència "pwr"
"mBB "+pwr	Robot va endarrere amb potència "pwr"
"mRB "+pwr	Robot va endarrere dreta amb potència "pwr"
"mRF "+pwr	Robot va endavant dreta amb potència "pwr"
"mLB "+pwr	Robot va endarrere esquerra amb potència "pwr"
"mLF "+pwr	Robot va cap endavant esquerra amb potència "pwr"
"stop"	Parem tots els motors
"ping"	Comprovem que el robot segueix connectat
"getTemp"	Obtenim la temperatura de la cpu de la Raspberry
"pwrOff"	Apaguem la Raspberry
"rec"	Comencem la gravació
"recStop"	Aturem la gravació
"pic"	Fem una fotografia.
"flw"	Comencem l'script de seguiment del peix vermell.
"flwStop"	Aturem el seguiment del peix vermell.

Taula 4.1 Llista de missatges MQTT programats i utilitzats

Funcionalitats

En aquest apartat s'explicen totes les funcionalitats que s'han programat al llarg del projecte.

Video i imatge

Una de les millores principals hem implantat en la nova versió de l'R2B2 és afegir retransmissió de vídeo a temps real i així obtenir un control del vehicle més bo i una experiència de pilotatge molt més divertida. A continuació descriurem diferents aspectes relacionats amb el vídeo, com ara, com ho hem dissenyat i implementat, els

problemes que he tingut per transmetre la senyal de vídeo al mòbil, i les opcions de gravar (guardar el vídeo) i fer fotos.

Retransmetre vídeo

El primer pas per poder streamejar vídeo és connectar i activar el mòdul de càmera de la Raspberry Pi. Totes les versions d'aquest microcontrolador tenen un connector CSI (*Camera Serial Interface*) per poder connectar-la fàcilment.

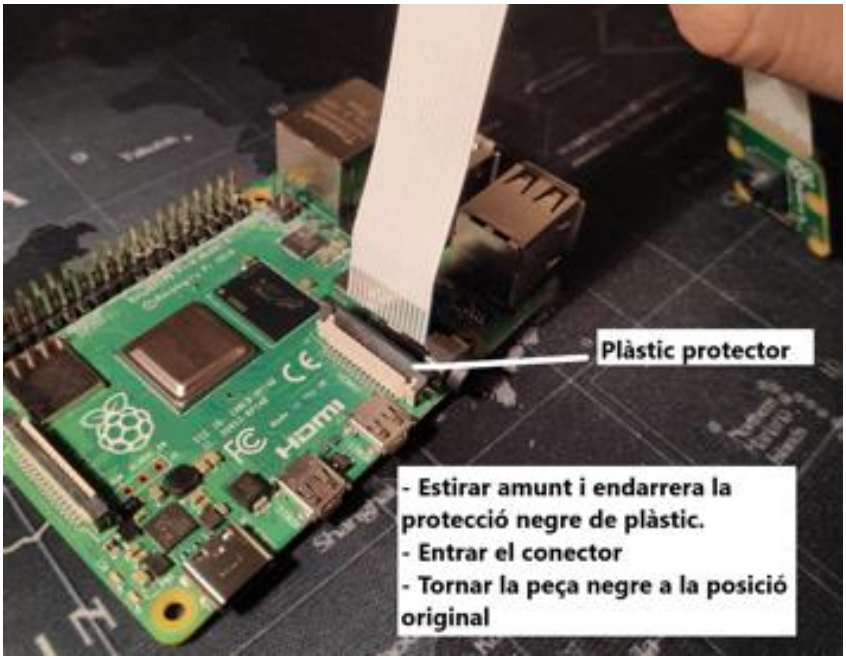


Fig. 4.3 Com connectar el mòdul de càmera a la Raspberry Pi 4

Per activar-la, només caldrà anar exectuar la comanda *sudo raspi-config*, obrir *Interfacing Options*, seleccionar *camera* i *enable camera*. A continuació, fem *reboot* de la màquina i ja tindrem el mòdul de càmera funcionant.

El protocol utilitzat per retransmetre vídeo a temps real és HTTP, és el més versàtil i utilitzat per aquest tipus de tasques i és fiable. Hi ha moltes maneres per fer-ho i existeix software que a més d'streamejar des de la Raspberry Pi, implementa software de seguretat amb detecció de moviment, gravació de vídeo, possibilitat de fer fotos, editar la imatge de sortida, etc. Nosaltres però, per poder agafar el vídeo i reproduir-lo en un telèfon mòbil necessitem el vídeo MJPG tot sol (format important).

Després de testejar vàries llibreries, la que va donar millors resultats i s'adaptava millor a les meves necessitats és *mjpg-streamer*. És un software lliure va ser desenvolupat originalment per Tom Stöveken. https://github.com/jacksonliam/mjpg-streamer

Passos per la instal·lació i execució de l'stream:

```
# Update & Install Tools
sudo apt-get update -y
sudo apt-get upgrade -y
sudo apt-get install build-essential imagemagick libv4l-dev
libjpeg-dev cmake -y

# Clone Repo
git clone https://github.com/jacksonliam/mjpg-streamer.git

# Inside the cloned Repo
cd mjpg-streamer/mjpg-streamer-experimental
make
sudo make install

# Run
/usr/local/bin/mjpg_streamer -i "input_uvc.so -r 1280x720 -d
/dev/video0 -f 30" -o "output_http.so -p 8080 -w
/usr/local/share/mjpg-streamer/www"
```

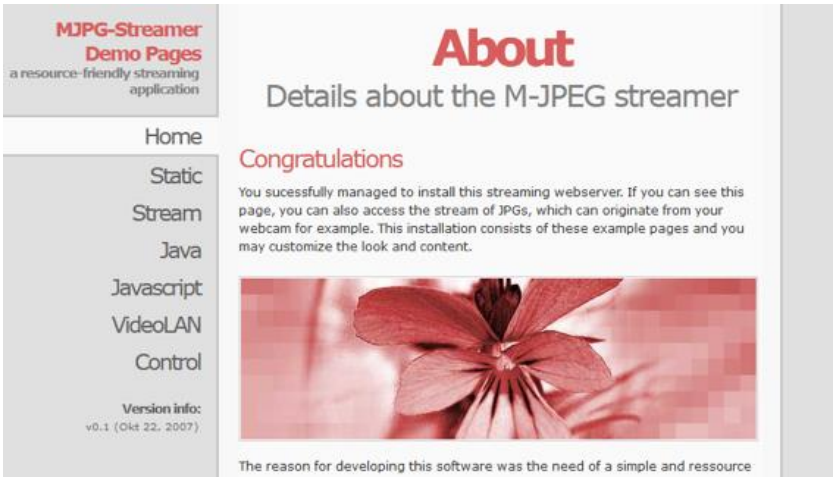


Fig. 4.4 MJPG-Streamer web - Demo

Per accedir-hi cal posar a la línia de comandes de qualsevol navegador:

http://IP:port

La IP serà la de la nostra Raspberry Pi i el port, el definim a la comanda quan executem el programa després del flag '-p'. MJPG-streamer té té una gran varietat d'opcións de configuració: el framerate, la resolució, el contrast, la brillantor, saturació, estabilització de vídeo, rotació, etc.

La interfície del software permet, afegint el text */?action=stream* al final de l'enllaç, obtenir el fil de vídeo net i, per tant, ja podrem agafar-lo des de la nostra aplicació i

reproduir-lo a temps real. A més, mjpg-streamer té altres funcionalitats com per exemple, permet fer i obtenir captures del vídeo.

Per agafar el vídeo en HTTP i reproduir-lo a l'aplicació, vaig utilitzar la llibreria IPCapture. És la única llibreria existent per ProcessingIDE que llegeix i visualitza vídeo a través d'una IP, el format de vídeo ha de ser MJPEG. És una llibreria creada per Stefano Baldan[18] al 2013, al 2016 va rebre la última actualització i va perdre part del suport.

Durant el procés de desenvolupament de l'aplicació em vaig trobar un *bug* en la llibreria. Aquest feia que l'aplicació "petés" si la resolució en la que retransmetia el vídeo era diferent a la finestra del telèfon mòbil. És a dir, no era capaç de fer un reescalat de la imatge. Aquest error passava només en el Android mode, altrament feia un reescalat correcte.

Com ja he dit abans es va perdre gran part del suport de la llibreria al ser abandonada pel seu autor, per arreglar aquest error, vaig haver de modificar un fitxer de la llibreria IPCapture. Per fer-ho però no va ser senzill ja que per compilar i crear el *.jar* necessari per crear de nou la llibreria havia de crear el projecte. Eclipse IDE té suport per Processing però des de fa un parell d'anys va treure el suport a Android. Això va fer que no pogués re-compilar tot el projecte. Per sort, el fitxer que vaig haver de modificar, no necessitava importar cap llibreria Android així que vaig crear el .class amb el fitxer modificat i el vaig substituir de la llibreria original.

Finalment va funcionar i per tant l'aplicació podia rebre vídeo en qualsevol resolució i aquesta seria reescalada a la resolució del telèfon mòbil. Això també donava la opció a retransmetre a menys qualitat si la cobertura no era prou bona.

Per solucionar el problema vaig participar en varis fòrums. En aquest post l'usuari @GWAK es va trobar en un problema semblant al meu.

https://discourse.processing.org/t/is-the-ipcapture-library-not-usable-not-android-mode/11315/21

Gravar vídeo

Aquesta funcionalitat permet poder començar i aturar la gravació des de la mateixa aplicació de mòbil. Per fer-ho s'envia un missatge mqtt per començar i un per aturar la gravació i són rebuts pel firmware del R2B2. La llibreria utilitzada és ffmpeg.

El problema principal va ser que el procés de gravar s'ha d'executar en paral·lel, altrament, l'aplicació queda bloquejada i no pot rebre més missatges.

Per solucionar-ho vaig decidir crear un altre client de MQTT que s'executava en un altre programa de python que estava subscrit només als missatges de vídeo. D'aquesta

manera quan es rebia el missatge de començar a grabar s'executava el següent Shell Script:

```
# [Varables]
source_stram="http://localhost:8181/?action=stream"
destination_directory="/home/pi/Videos"
destination_file="vid_$(date +'%Y%m%d_%H%M%S').mpeg"

# Recored Stream w/ ffmpeg
ffmpeg -re -i "${source_stram}" -
q:v 10 "${destination_directory}/${destination_file}"
```

Per aturar el vídeo busquem el procés i el matem amb la comanda ***sudo killall ffmpeg***. El vídeo es guarda en la memòria de la RaspberryPi, dins la carpeta /home/pi/Videos amb el nom vid_yyyymmddhhmmss.

Fer fotos

Per poder fer fotos utilitzarem la llibreria de python *urllib*, que serveix per obtenir diferents recursos d'internet. En el nostre cas, imatges guardades per MJPG-streamer. La url http://ip:port/?action=snapshot ens donarà tant sols una imatge *.jpg* que podrem agafar i guardar a la memòria de la RaspberryPi cada vegada que rebi el missatge MQTT 'pic'.

Comanda:

```
now = datetime.now()
dt_string = now.strftime("pic_%d%m%Y_%H%M%S")#pic_ddmmyyyy_hhmmss
urllib.urlretrieve("http://"+ipR2B2+":8181/?action=snapshot", "/home/pi/Pictu
res/"+dt_string+".jpg")
```

Resolució escalable

Una de les millores que vaig desenvolupar en el meu projecte respecte a la versió de l'ESP32, era el tema de l'escalabilitat de l'aplicació android. Els mòbils tenen resolucions diferents depenent del model, i aquestes resolucions tendeixen a augmentar cada vegada més.

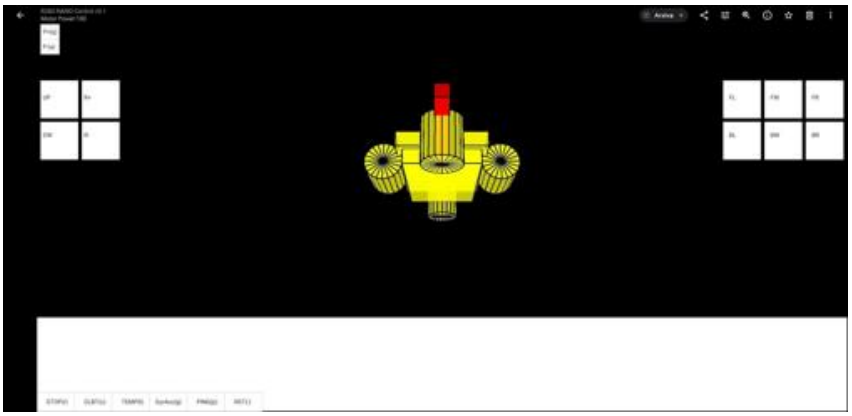


Fig. 4.5 Aplicació R2B2 nano sense reescalat

És per això que fer una aplicació que sigui reescalable a qualsevol resolució és important. En aquest sentit cal que tant la imatge de la càmera com els botons i els textos s'adeqüin a la resolució de la pantalla de l'aparell utilitzat.

És important parametritzar tots aquests valors ja que per exemple, per fer un botó o un text, en el processing, és necessari determinar la seva posició (en coordenades cartesianes) visual i l'àrea de "*Touch*", per separat. En llenguatges o eines de programació més modernes com Android Studio, això ja es faria automàticament.

Control motors

El software de control dels motors s'ha de dissenyar i implementar en Python directament a la RaspberryP. A continuació explicaré breument el funcionament del codi.

Primer de tot definim els pins utilitzats en la Raspberry Pi. Són tot pins de sortida.

```
#1st motor Driver
GPIO.setup(12, GPIO.OUT)    # PWMA  19
GPIO.setup(18, GPIO.OUT)    # AIN2  21
GPIO.setup(16, GPIO.OUT)    # AIN1  23
GPIO.setup(22, GPIO.OUT)    # STBY  24
GPIO.setup(15, GPIO.OUT)    # BIN1  29
GPIO.setup(13, GPIO.OUT)    # BIN2  26
GPIO.setup(11, GPIO.OUT)    # PWMB  31

#2nd motor Driver
GPIO.setup(19, GPIO.OUT)    # PWMA 19
GPIO.setup(23, GPIO.OUT)    # AIN2 21
GPIO.setup(21, GPIO.OUT)    # AIN1 23
GPIO.setup(24, GPIO.OUT)    # STBY 24
GPIO.setup(29, GPIO.OUT)    # BIN1 29
GPIO.setup(26, GPIO.OUT)    # BIN2 26
GPIO.setup(31, GPIO.OUT)    # PWMB 31
```

A la inicialització del motors crearem quatre canals PWM amb una freqüència de 100 (recomanada per als TB6612FNG) i ho assignarem als motors.

```
    pwmFreq=100
    pwma1 = GPIO.PWM(12, pwmFreq)
    pwmb1 = GPIO.PWM(11, pwmFreq)
    pwma2 = GPIO.PWM(19, pwmFreq)
    pwmb2 = GPIO.PWM(31, pwmFreq)
```

Utilitzem .start per definir la potència inicial (voltatge) dels motors. Varia de 0 a 100.

```
    pwma1.start(50)
    pwmb1.start(50)
    pwma2.start(50)
    pwmb2.start(50)
```

Per inicialitzar una altre vegada o aturar els motors, posem tots els pins a zero.

```
def iniMotors():
    GPIO.output(12, GPIO.LOW)
    GPIO.output(18, GPIO.LOW)
    GPIO.output(16, GPIO.LOW)
    GPIO.output(22, GPIO.LOW)
    GPIO.output(15, GPIO.LOW)
    GPIO.output(13, GPIO.LOW)
    GPIO.output(11, GPIO.LOW)

    GPIO.output(19, GPIO.LOW)
    GPIO.output(23, GPIO.LOW)
    GPIO.output(21, GPIO.LOW)
    GPIO.output(24, GPIO.LOW)
    GPIO.output(29, GPIO.LOW)
    GPIO.output(26, GPIO.LOW)
    GPIO.output(31, GPIO.LOW)
```

Cada “motor driver” controla a dos motors, d’aquests en podem controlar la velocitat (voltatge subministrat) i la direcció (CW o CCW).

```
def runMotor(motor, spd, direction):
    GPIO.output(22, GPIO.HIGH);
    in1 = GPIO.HIGH
    in2 = GPIO.LOW
    if(direction == 1):
        in1 = GPIO.LOW
        in2 = GPIO.HIGH

    if(motor == 0):
        GPIO.output(16, in1)
```

```
        GPIO.output(18, in2)
    elif(motor == 1):
        GPIO.output(15, in1)
        GPIO.output(13, in2)
```

Temperatura

Entre els components electrònics del robot disposem d'un sensor de temperatura a la CPU de la RaspberryPi. Ens va semblar interessant poder llegir aquest valor des de l'aplicació mòbil per saber si s'escalfa massa i causa un fenomen anomenat "thermal throttling", que reduiria la potència del nostre processador.

El funcionament consisteix en enviar un missatge MQTT demanant la temperatura (des de la app), llegir-la, i publicar un altre missatge (des del bot) amb la temperatura. Al rebre-la, serà mostrada a la part superior dreta de l'aplicació.

Codi python per comprovar la temperatura del nostre processador és el següent:

```
def measure_temp(client):
    global topic
    temp = os.popen('vcgencmd measure_temp').readline()
    client.publish(topic,(temp.replace("temp=","").replace("'C\n","")))
```

Comportament autònom: seguiment d'un objecte de color

Gràcies a la càmera i la potència de la Raspberry Pi 4, obra la porta a moltes possibilitats. Una d'elles, la utilització de tècniques de visió per computador. La idea d'aquest apartat del projecte és que puguem posar el robot a l'aigua, i en qualsevol moment, des de l'app mòbil, executar un script que faci que, quan el robot vegi un objecte "interessant" com un objecte d'un color predeterminat, el robot vagi autònomament cap aquest objecte.

L'algorisme que vaig desenvolupar dóna consignes als motors depenent de la posició de l'objecte respecte la càmera. Per detectar la posició de l'objecte de color s'utilitza la tècnica de CCL Connected-component labeling[9].

L'algorisme està programat en python per temes de comoditat i comunicació (ja que tot el codi està fet amb Python). Quan es prem el botó de "follow" de l'app mòbil, s'executa l'script de seguiment i quan es torna a prémer, es mata el procediment directament.

Estructura del codi:

```
#Ens connectem al servidor MQTT per tal de poder donar ordres als motors.
#obrim stream de video de la ip local
while True:
    #llegim un frame i en reduim la mida per alleujar-ne el cost
    #apliquem un filtre HSV per tal de deixar passar només un color
    #binaritzem la imatge
    #apliquem CCL
    #seleccionem el bloc més gran i busquem el centre de masses
    #dividim la imatge en 9 blocs i el posicionem dins un d'ells
    #enviem la ordre als motors
    #si l'objecte és més gran que un threshold (està aprop), no moure's
```

Segmentació per color

El rang de colors més conegut i utilitzat és el RGB (*Red, Green, Blue*),. En visió per computador i processat d'imatge, però, utilitzar aquest model de color no és adequat. Per explicar perquè és millor utilitzar un filtre HSV perquè per exemple, si volguéssim detectar una senyal d'stop buscaríem el color vermell, en RGB (255, 0, 0). Tot i això aquest valor de RGB és extremadament específic i la possibilitat de trobar vermell "pur" en una imatge no artificial és molt difícil. Es podria buscar un rang de vermells però això no solucionaria el problema ja que en realitat aquest vermell és una combinació dels 3 colors primaris.

Canviar els valors dels 3 colors primaris suposaria massa esforç i encara podríem obtenir resultats no molt bons.

A diferència del RGB, el model de color HSV utilitza els paràmetres *Hue* (matís), *Saturation* (saturació) i *Value* (valor) per definir un color. D'aquesta manera modificant el rang de valors de matís podem capturar tots els vermells.

El color vermell queda dividit entre els valors més baixos de *Hue* i els valors més alts, és per això que per fer-ne el rang necessitem sumar les dos màscares.

Valors utilitzats per segmentar el color vermell amb l'ajuda de la llibreria de python cv2:

```
    lower_color = np.array([0,100,100])
    upper_color = np.array([10,255,255])
    lower_color2 = np.array([170,100,100])
    upper_color2 = np.array([179,255,255])
```

Algorisme CCL (Connected-component labeling)

L'algorisme en qüestió és usat en visió per computador per detectar regions connectades en imatges binàries. A través d'aquest algorisme podem identificar aquests grups de núvols de punts i etiquetar-los. Cada grup de píxels diferent tindrà una etiqueta

diferent. Els núvols de píxels es poden considerar junts si es toquen per un dels 4 costats (*4-connectivity*), o per un dels 4 costats o 4 cantonades (*8-connectivity*).

Mentre anem recorrent la imatge (d'esquerra a dreta), per cada píxel, obrim una finestra de 3x3 on mirem els píxels anteriors al del centre (esquerra i dalt). En 4-*connectivity* només n'haurem de mirar dos, en 8-*connectivity*, 4.

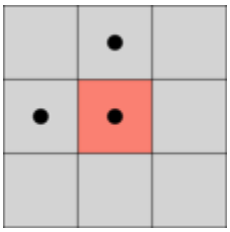


Fig. 4.6 4-connectivity

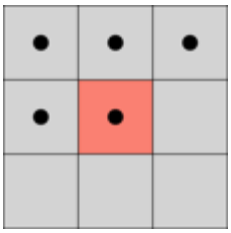


Fig. 4.7 8-connectivity

El pseudocodi de l'algorisme CCL és el següent:

```
algorithm TwoPass(data) is
   linked = []
   labels = structure with dimensions of data, initialized with the value of Background

   First pass

   for row in data do
      for column in row do
         if data[row][column] is not Background then

            neighbors = connected elements with the current element's value

            if neighbors is empty then
               linked[NextLabel] = set containing NextLabel
               labels[row][column] = NextLabel
               NextLabel += 1

            else

               Find the smallest label
```

```
            L = neighbors labels
            labels[row][column] = min(L)
            for label in L do
               linked[label] = union(linked[label], L)

Second pass

for row in data do
   for column in row do
      if data[row][column] is not Background then
         labels[row][column] = find(labels[row][column])

return labels
```

Aquest vídeo ensenya gràficament el pas a pas de l'algorisme i ajuda molt a entendre'l:

https://www.youtube.com/watch?v=ticZclUYy88

Capítol 5 Desenvolupament del nou robot

Carcassa

Al estar tot comprat a BlueRobotics no va ser necessari fer gaire enginyeria. El muntatge va consistir en ajuntar les següents peces:

Fig. 5.1 Cast Acrylic Tube (3'' Series)

Fig. 5.2 O-Ring Flange

Fig. 5.3 Dome End Cap (3'' Series)

Fig. 5.4 Aluminium End Cap with 4 holes (3'' Series)

Fig. 5.5 Enclosure Vent and Plug

Fig. 5.6 M10 Cable Penetrator (6mm Cable)

El resultat és el següent:

Fig. 5.7 Carcassa blue robotics muntada

Motors Model 1

A continuació explicaré el procés d'encapsulament i ensemblatge de motors i hèlix, que com podreu comprovar no va ser fàcil. La idea era poder encapsular motors de drone de manera que fossin resistents i capassos de funcionar sota l'aigua. A més va caldre idear un sistema per subjectar l'hèlix a l'eix del motor. Per tot aquest procés vam intentar seguir els passos que es fan servir per estancar i encapsular els motors en el robot R2B2 original. Per la primera implementació vaig utilitzar els següents materials i eines:

Material

- 4x motorPrint.stl
- Vaselina
- Cola Tèrmica
- *Self Bonding Electrical Tape*

Eines

- Llima per netejar les peces impreses
- Pistola de cola tèrmica

Muntatge

Per fer el muntatge, el primer que vam fer va ser imprimir 4 carcasses i quatre taps per els motors i polir-ne les imperfeccions amb una llima. Es poden trobar els models que vaig utilitzar en el github. En el meu cas, vaig utilitzar la impressora 3D del CIRS, però lògicament es pot utilitzar qualsevol impressora. El filament utilitzat va ser ABS, perquè és el més resistent i per tant, també el que ens donava més garanties per aconseguir l'estancaitat.

Fig. 5.8 Procés i resultat de l'impresió en 3D dels recipients

Una vegada tinguem tots els materials ja podrem començar. Primer de tot taparem amb cola tèrmica els dos forats de la part superior del motor. A continuació, recobrirem el cos del motor amb cinta aïllant i hi posarem una capa de vaselina. Tot seguit posem la carcassa que hem imprès de tal manera que només sobresurti l'eix del motor.

La idea de la vaselina i la cinta aïllant és de cobrir el 100% de l'espai on hi ha el motor i treure'n tot l'aire.

Fig. 5.9 Motor dins l'encapsulament, preparat per posar el tap

El següent pas consistirà en emplenar la part inferior del motor amb cola tèrmica. És recomanable donar un parell de voltes als dos cables que surten del motor abans d'emplenar-ho de cola. D'aquesta manera dificultem encara més l'accés a l'aigua.

Fig. 5.10 Tap col·locat i ensiliconat

Abans de col·locar la tapa emplenarem de vaselina l'eix, de manera que, quan hi posem la tapa a sobre, sobresurti vaselina per tots costats. L'objectiu és emplenar totes les cavitats d'aire que puguin quedar de vaselina i d'aquesta manera, evitar que semplenin d'aigua. Una vegada el tap estigui el seu lloc, netegem la vaselina sobrant i segellem amb cola tèrmica.

Per últim, amb la cinta aïllant especial (*self bonding electrical tape*), recobrirem tot el motor.

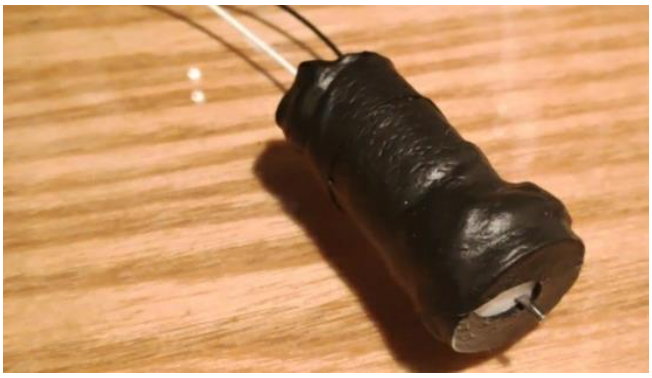


Fig. 5.11 Motor totalment aïllat

Hèlixs

L'ensemblatge de les hèlixs als eixos dels motors ha estat un petit repte en aquest TFG. A l'apèndix d'aquest projecte hi ha un apartat sobre el desenvolupament de les hèlixs. A continuació que ensenyaré com construir-les. El material necessari és el següent:

Material

- 4x hèlixs
- Plàstic fresadora
- 6x M3 cargol
- 6x M3 rosca

Eines

- Taladre i broques de mètric 3
- Fresadora
- Serra/llima per netejar les peces fresades

Muntatge

Amb la peça cilíndrica del mateix diàmetre que el motor dissenyada, en un costat hi fem un forat de 0.9 mm i en l'altre un forat de mètric 3.

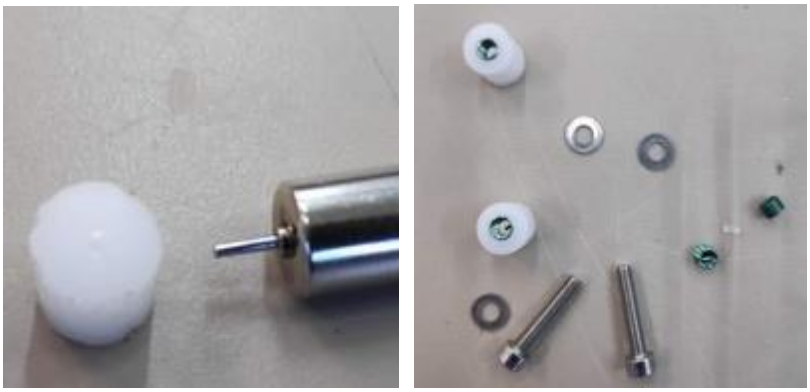


Fig. 5.12 Motor amb l'acoblament per l'eix (esq.) - Acoblament amb l'helicoil instalat (dreta)

Hi posem la rosca del cargol, el cargol i unes femelles per distribuir la força més equitativament per tot el cilindre.

Fig. 5.13 Hèlixs acabades

Els motors de la versió 2 es van fer seguint el mateix procediment. Amb l'única diferència de que l'encapsulament eren pots de plàstic prèviament comprats.

Ensemblatge dels motors al robot

A continuació explicaré com connectar els quatre motors, com fer passar el cable a través del penetrator fins a dins del robot, i com crear un connector per connectar i desconectar fàcilment els motors.

Material

- Silicona
- Cables (amb pins i sense)
- Motors estancs
- Carcassa amb el xassís
- Estany
- Brides
- Cinta aïllant (normal i la "*Self bonding electrical tape*")

Eines

- Pistola de cola tèrmica
- Soldador

Muntatge

Per connectar els motors als drivers cal tenir en compte que estem connectant uns components que van a l'exterior (motors) amb uns que van a l'interior del xasís (*drivers*). Per tant, hem de tenir en compte que al obrir el robot, els cables estaran enganxats a la tapa. Per facilitar-ne l'obertura, he utilitzat uns cables amb pins (normalment utilitzats per connexions amb GPIO) i una mica de cinta, i així idear quelcom semblant a un connector. Per saber com connectar-lo, he etiquetat amb un número (una enganxina) cadascun dels diferents fils que van a cada motor.

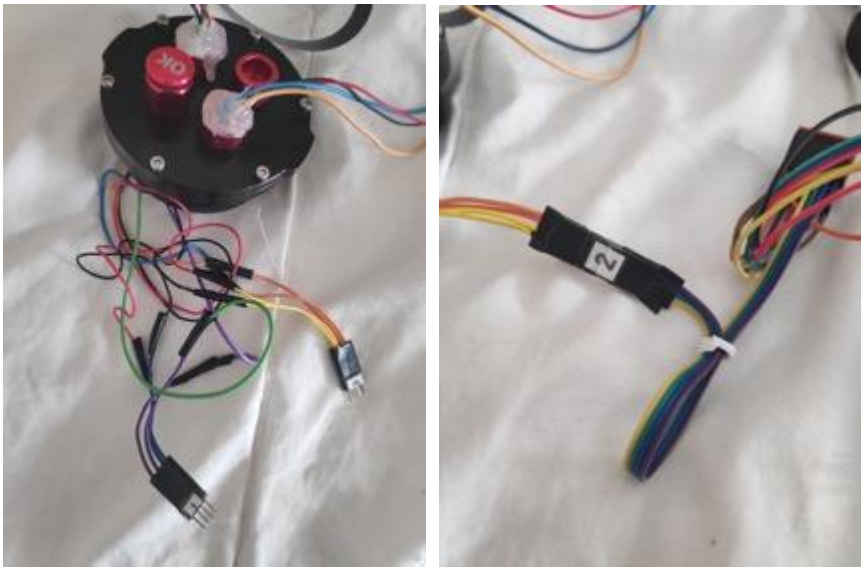


Fig. 5.14 Connexions DIY per els motors

Per ajuntar els motors amb els connectors, pelarem els cables i els soldarem amb una mica d'estany. Com que estaràn fora l'aigua, necessitem aillar-los. Primer posarem una capa de cola tèrmica. Per assegurar, els unim i els recobrim amb cinta aillant especial (*self bondig electrical tape*).

Fig. 5.15 Soldadures per els motors

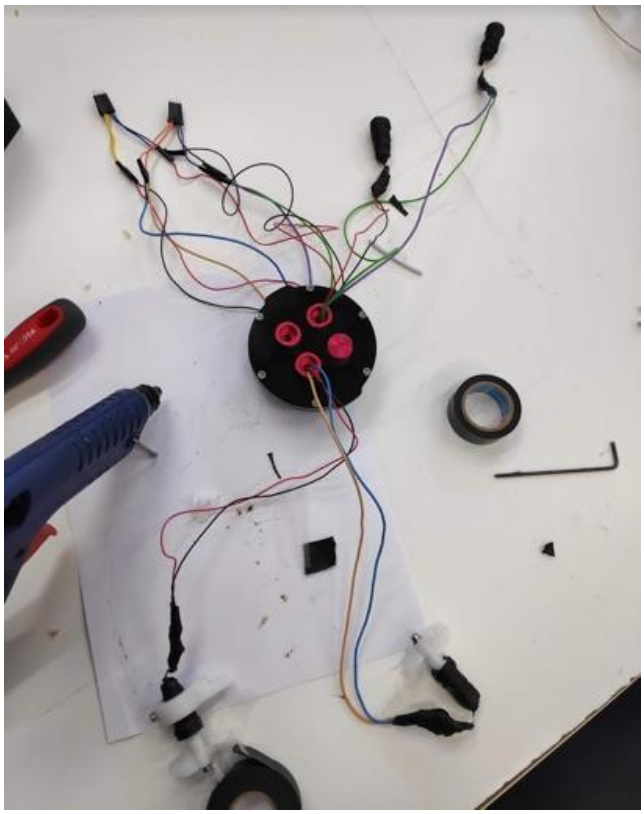


Fig. 5.16 Tots els motors connectats

Per aconseguir impermeabilitat en els *penetradors* (taps amb forat pels cables) caldrà taponar-los amb algun material. En el nostre cas hem utilitzat silicona. Els cables, els col·loquem amb l'ajuda de brides de tal manera que empipin el menys possible. Els motors laterals també aniran enganxats amb brides.

Fig. 5.17 Motors connectats i xassis muntat a l'encapsulament

Xassís

El muntatge del xassís és un dels més tediosos ja que cal fresar i ajuntar bastantes peçes. A continuació mostraré i explicaré el procediment que he seguit per construir el xassís. Els materials i eines necessàries són les següents:

Material

- 1x RedChasis
- 1x GreenChasis
- 1x PurpleChasis
- 2x MotorHolders
- Plàstic
- 6x M3 cargol
- 6x M3 rosca

Eines

- Taladre i broques de mètric 3 (veure Apèndix – Eines)
- Fresadora
- Serra/llima per netejar les peces fresades

Muntatge

Primer de tot necessitarem passar a format *.dxf* (2D) els models en 3D del xassís, ja que la fresadora tindrà sempre una cara totalment plana, a l'altre haurem de donar-li els paràmetres d'alçada tenint en compte el gruix del plàstic inicial. Si la peça de plàstic és massa gruixuda, s'haurà de rebaixar tota la peça, altrament només caldrà tallar.

Fig. 5.18 Procés de fresa d'una peça

Una vegada fresada la peça caldrà separar-la de la resta del plàstic i llimar-ne les imperfeccions.

Per fer els forats on hi posarem la rosca i el cargol per poder unir les peces:

El primer pas serà agafar la broca de mètric 3 i fer el forat.

Fig. 5.19 Foradant la peça amb mètirc3 per posar-hi l'helicoil

A continuació amb una altre broca especial (nº3), farem la rosca (helicoil) dins el plàstic.

Tot seguit col·locarem la rosca de metall dins una altre broca (nº2) i l'entrarem amb l'ajuda del taladre a l'interior de la peça.

Fig. 5.20 Ús de l'helicoil

A l'altre peça que vulguem unir, només caldrà fer-li el forat amb la broca de mètric 3. Per acabar ja podrem col·locar el cargol. Per temes estètics i d'aerodinamicitat també

podem avellanar (nº5) o fer més gran el forat (nº4) per tal de que el cap del cargol quedi dins el plàstic.

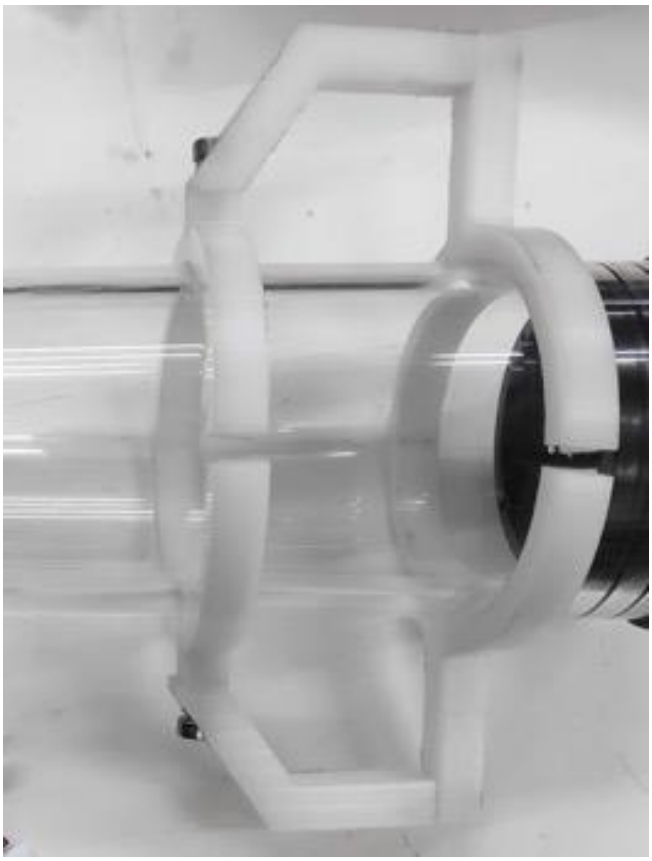


Fig. 5.21 Resultat de l'estructura de plàstic exterior

Per poder aguantar els motors verticals (de la versió 1) i centrar-los al mig de l'ala, afegim els dos motor holders. Fresem les peces i, també amb la fresadora, rebaixem en 6 mm, 8mm de la peça "greenChasis". En el cas dels motors més grossos del model 2 vaig enganxar-los utilitzant brides. Tot i això, a través dels dissenys d'openSCAD es podria modificar fàcilment el diàmetre del forat del *motor holder* per poder fer-li cabre.

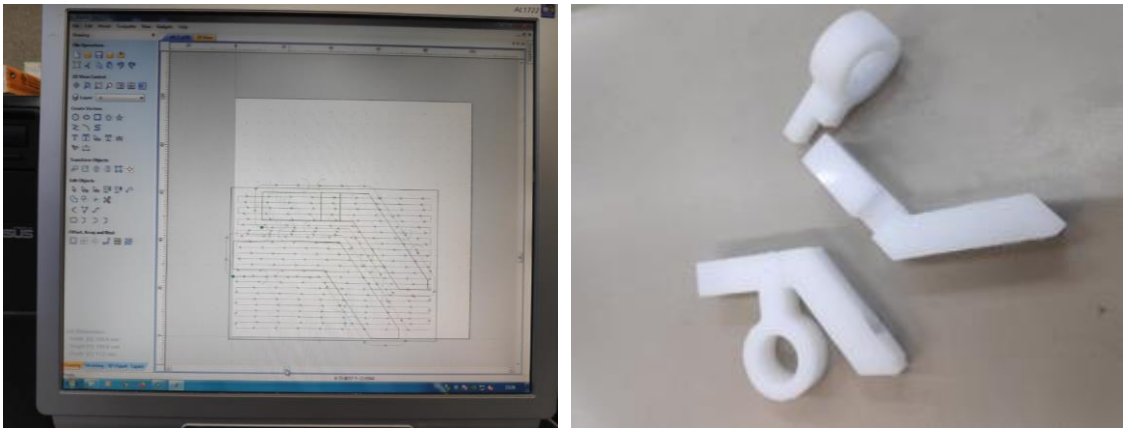


Fig. 5.22 Afegint els suports per els motors

Per subjectar les peces al cos del robot, farem igual que amb la resta del xassís: forats de 3mm amb rosca. Per subjectar els motors al motorHolder, utilitzarem les rosques i un cargol curt de M3.

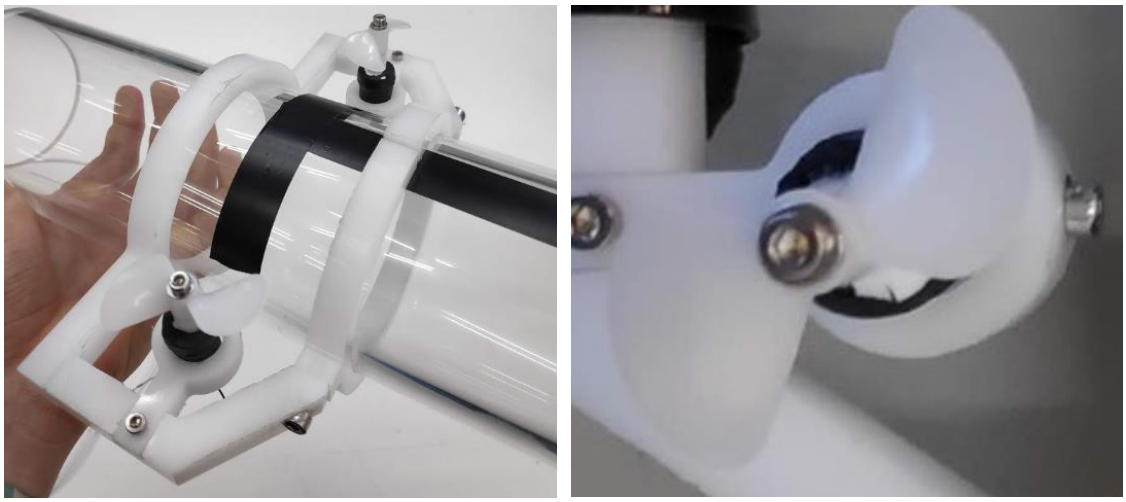


Fig. 5.23 Resultat final exterior, amb els motors inclosos

Boia

El muntatge de la boia és més aviat senzill. El que necessitem és una base de espuma FOAM prou gran per poder aguantar i protegit el módul WiFi que hi haurà a sobre. El material necessari és el següent:

Material

- Espuma FOAM
- Loctite
- Cinta aïllant
- Allargador USB

Eines

- Cúter
- Paper de vidre
- Llimes
- Trepant

Muntatge

La creació de la boia té poca part tècnica. Amb un cúter fem la forma que vulguem que tingui amb espuma de FOAM, i amb l'ajuda de paper de vidre i llimes, li acabem de donar la forma i allisem els costats. Amb un trepant, fem un forat al mig de la boia, i hi passem el cable USB femella.

Una vegada el cap hagi travessat la boia, taponem el forat amb Loctite i cinta aïllant per els dos costats i deixem assecar.

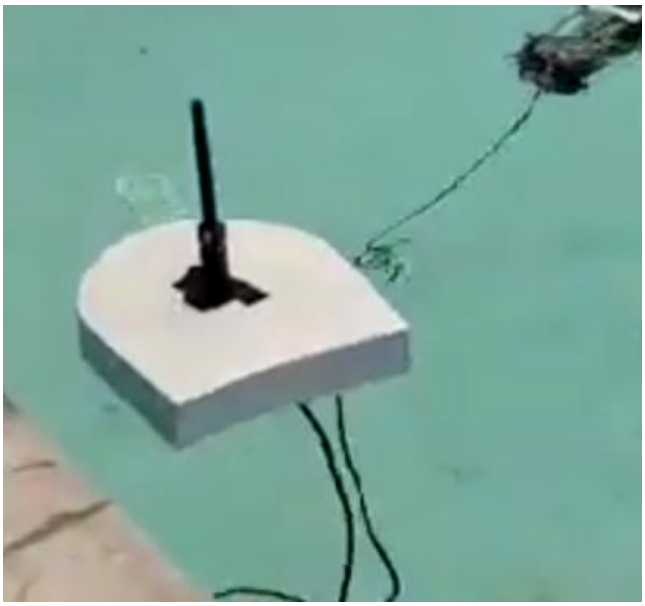


Fig. 5.24 Boia flotant amb el mòdul WiFi enganxat

Capítol 6 Proves i resultats finals

Tests del funcionament bàsic

Experiment 1

Aquest va ser dels testos inicials que vaig dur a terme. Em va servir per testejar el funcionament dels motors i per decidir la connexió de les bateries: per saber si amb una en feia prou, si les connectava en sèrie, en paral·lel, una per motor i driver, etc. En aquesta etapa de tests no només va ser anar provant sinó també informant-me a través d'internet. Una de les coses que vaig aprendre, va ser que utilitzar la mateixa font d'alimentació per alimentar el controlador (Raspberry Pi) i els motors era una mala idea, ja que una pujada de potència dels motors podria causar una baixada de tensió a la Raspberry Pi i fer que es reiniciés. (8 de Novembre de 2019)

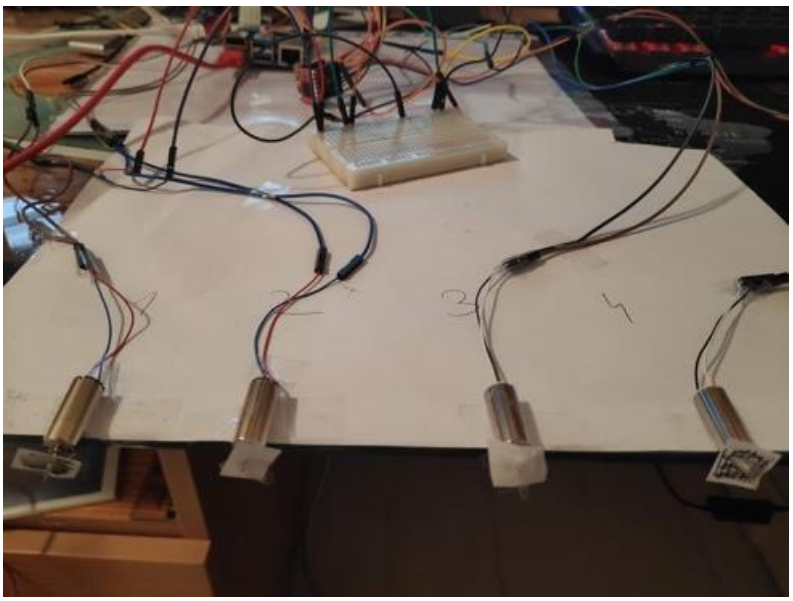


Fig. 6.1 Primers testos de funcionament i potència dels motors

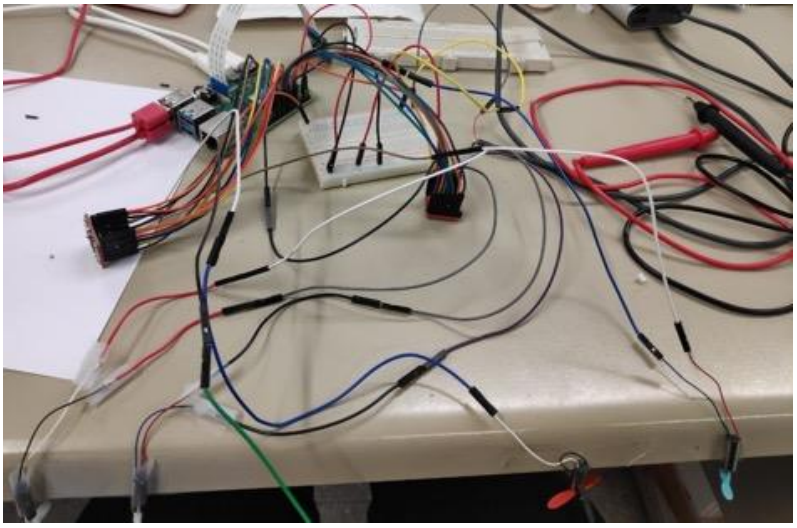


Fig. 6.2 Tests en el laboratori de Visió

Experiment 2

Una vegada vaig tenir tots els cables dels motors soldats, aïllats i amb els connectors a l'interior del recipient era moment de fer la primera prova d'aigua. I així comprovar la impermeabilitat de les juntes, els taps, i els recobriments de cola tèrmica que havia fet per taponar els "cables penetrators". Per comprovar si entrava aigua vam posar paper de vàter als dos costats; en cas que es mullessin sabríem per quin costat hi havia fuites. (20 de desembre de 2019)

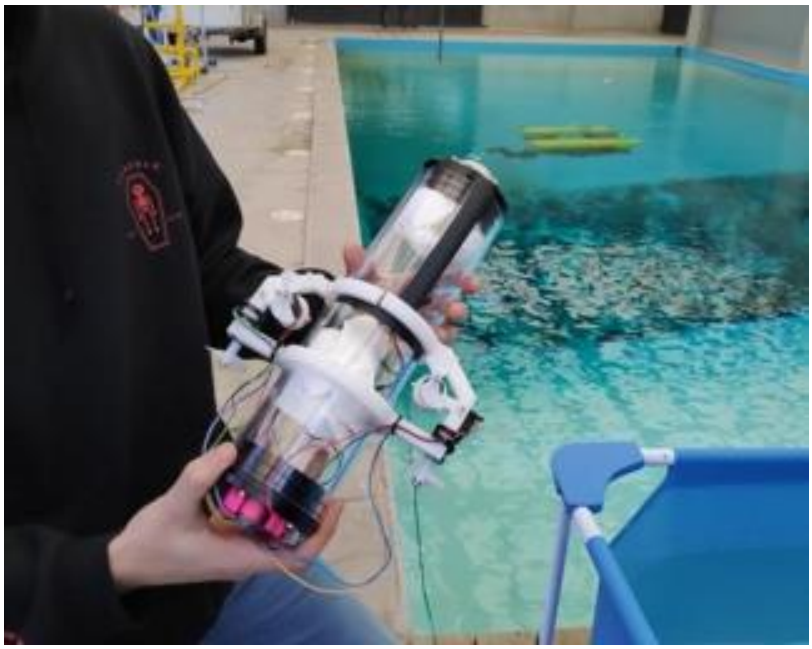


Fig. 6.3 Primer test sota l'aigua al CIRS

Experiment 3

Després de comprovar que el recipient era estanc, ja podia posar els components electrònics dins i passar a la següent fase: equilibrar el robot. (8 de Gener de 2020)

Fig. 6.4 Buscant l'equilibri i flotabilitat zero

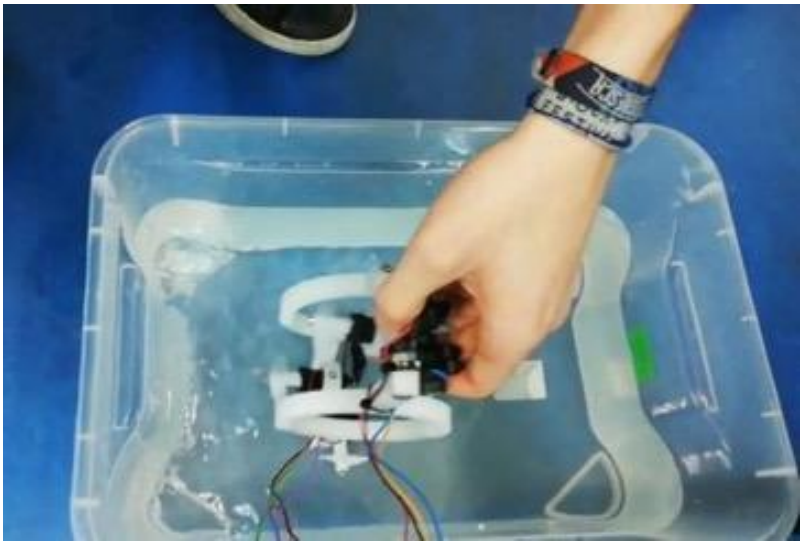
En la primera versió del robot, per aconseguir l'equilibri va ser necessari afegir bastants pesos. Com que treballava paral·lelament en varis apartats del projecte, no vaig tenir les hèlixs llestes fins després d'haver fet l'estabilització del robot.

Experiment 4

Tant bon punt vaig tenir les hèlixs acabades, vaig començar a fer els testos dels motors dins l'aigua i em vaig adonar que la potència que aportaven no seria suficient.

En els següent vídeo es pot veure un dels testos dels motors de la versió 1. Agafant-los amb la mà vaig poder notar la força que feien i ja em va semblar que no seria suficient. (10 de gener de 2020)

- https://youtu.be/mSqvUc8tz4c



Experiment 5

En aquest altre vídeo veiem el funcionament dels motors amb tot l'R2B2 dins l'aigua. L'enregistrament mostra com submergeixo el robot dins l'aigua i faig tests de motor. La versió no estava polida però ja es veia que la força que feien els motors no era suficient. (10 de gener de 2020)

- https://youtu.be/zjbgNVf6tBs

A partir d'aquest experiment entrem en els testos de la versió 2 del R2B2 Raspberry Pi.

Experiment 6

Test complet però agafant-lo amb la mà (4 de Febrer de 2020). En aquest test comprovem que tot funcioni correctament dins l'aigua. També serveix per veure la força dels motors. En el vídeo es veu com submergeixo el robot dins la piscina del CIRS i faig tests amb els motors mentre aguanto el robot amb la mà.

- https://youtu.be/UZixkQNRLew

Experiment 7

Test amb corda (4 de Febrer de 2020). En aquest test es posa tot el robot dins l'aigua amb control lliure. Al ser el primer test on no s'està agafant el robot amb la mà, vam posar una corda de seguretat per no perdre'l.

- https://www.youtube.com/watch?v=RXXrLQeSBlo

Experiment 8

Test on el robot anava lliure, és a dir, sense lligar-lo amb cap corda ni fil per no perdre'l. (6 de febrer de 2020). Els resultats ja començaven a ser molt bons. En l'enregistrament es veu el ROV navegant per la piscina del CIRS. En alguns moments no acaba d'anar del tot recte, s'havia de fer una petita redistribució de pesos per arreglar-ho.

- https://youtu.be/ANKcAMHNRUk

Experiment 9

Durant aquests tests, vam tenir la sort de poder gravar des de sota l'aigua a la piscina que es troba al CIRS, dins el Parc Científic i Tecnològic de Girona. (6 de Febrer de 2020). Els vídeos mostren des de diferents perspectives aquest experiment.

- https://youtu.be/irv4gB_E2LU
- https://youtu.be/YjrH_nTIA6c
- https://youtu.be/YbpUgBr7xsY (visió des del R2B2 del test amb velocitat x4)

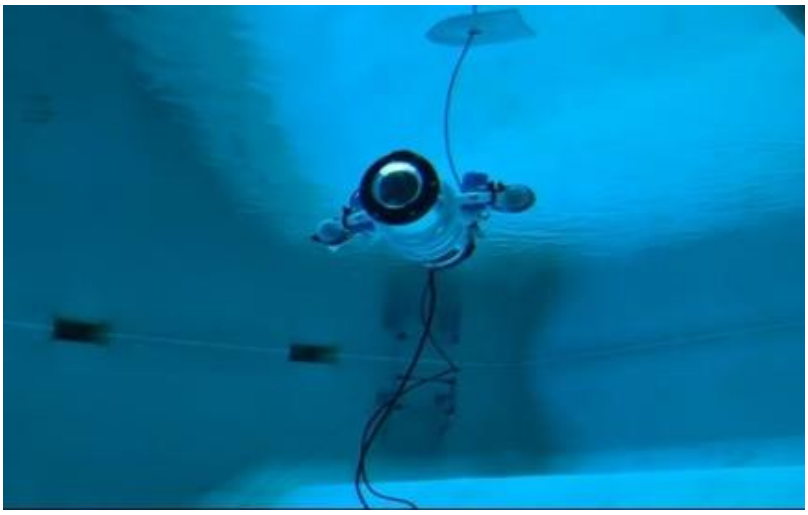


Fig. 5.5. Foto feta sota l'aigua, a la piscina del CIRS

Tests de seguiment d'un objecte

Els resultats obtinguts eren molt bons, però, després d'aconseguir una navegació fluida utilitzant els controls del mòbil el que volíem a continuació era que el robot pogués seguir un objecte de color d'una manera autònoma.

Experiment 10

Per poder fer els testos més ben documentats, vaig executar l'script de seguiment des d'un portàtil, que feia els calcula i enviava les ordres al robot. D'aquesta manera podia mostrar fàcilment una pantalla on es veies el que veia el robot (a part del mòbil), i la segmentació a temps real que feia l'algorisme. (18 de Febrer de 2020). El vídeo de continuació mostra el primer experiment que vaig fer pel seguiment d'un objecte. Es poden veure dues vistes, la del robot, i una des de l'exterior de la piscina.

- https://youtu.be/F0c2vC839XA

Fig. 6.5 Primers tests de seguiment efectuats amb una pilota vermella enganxada a un pal

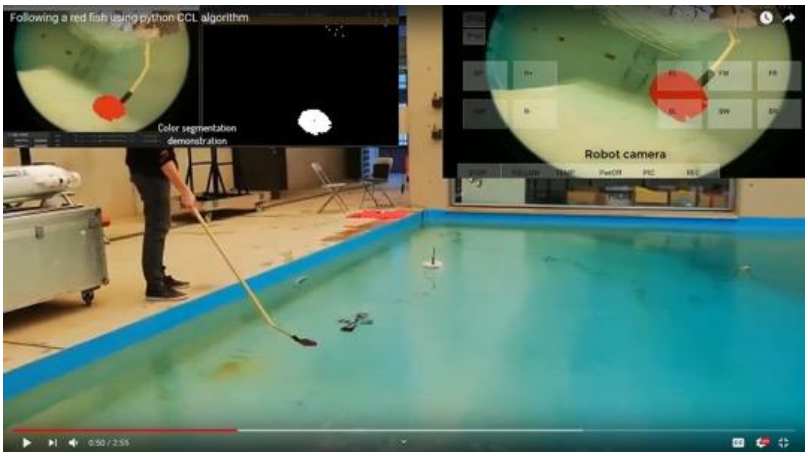
Experiment 11

Va ser en aquest punt que vaig decidir canviar el mòdul de càmera standard per un amb lents d'ull de peix. Aquest mòdul de càmera feia el "tracking" a l'objecte molt més senzill, ja que amb els seus 160 graus de visió (normalment les càmeres el tenen de 72 graus) era pràcticament impossible que el perdés. A més, amb les velocitats ajustades, el seguiment era molt més fluid i eficaç.

Per fer la demo final, vaig desenvolupar també un peix de color vermell. D'aquesta manera el seguiment tenia una pinta més immersiva. (20 de febrer de 2020). El test final consistia en posar el robot a l'aigua i executar la rutina follow. Tot seguit agafàvem el peix vermell (Figura 6.8) enganxat en un pal i fèiem un recorregut per la piscina del CIRS. En el recorregut fèiem fer al robot tot tipus de moviments: pujar, baixar, girar i parar.

Fig. 6.6 Peix vermell per la demostració del seguiment

En el següent vídeo es pot veure la càmera del robot, la segmentació que fa l'algorisme CCL i un vídeo fet per un company des de fora la piscina, mentre es duia a terme el test.



https://youtu.be/p0Jb0TmXWtc

Capítol 7 Guia d'ús

Posada a punt

Per tal de tenir el robot apunt cal primer de tot comprovar que les dues bateries del submarí estan carregades. La powerbank de 10400mAh per la Raspberry és pot recarregar amb pràcticament qualsevol carregador de mòbil, ja sigui mini-USB o USB-C. Per altre banda, la bateria LiPo per els motors necessita més cura alhora de carregar-la:

El primer que cal fer és connectar el positiu i negatiu 4mm banana connector al balancejador de càrrega. Tot seguit ja podrem connectar l'altre costat del cable a l'R2B2.

És important seguir aquest ordre ja que si ho féssim al revés, quedarien els dos pols de la bateria penjant i, al tocar-se el faria un curtcircuit i faríem malbé tota la bateria.

A continuació podem connectar els 4 pins que surten de la bateria als sockets de balanceig del carregador.

Fig. 7.1 Connexions al balencejador

Les connexions en la figura 6.1. són les següents: en vermell i negre grans tenim els connectors banana de 4mm. En cables més prims de color vermell/blau/groc/negre són els de balanceig.

Engeguem el balancejador connectant-lo a la corrent. Per carregar la nostra bateria seleccionarem el mode "CHARGE". Aquest mode és el més ràpid per carregar però cal anar amb molt de compte i hem de vigilar ja que aquest mode no comprova que totes les cèl·lules tinguin el mateix voltatge. Tècnicament podem carregar a un amperatge de 6 Ampers ja que la gran majoria de bateries LiPo tenen un rati de càrrega de 1C és a dir: si la bateria és de 3000mAh -> 3A, si és de 4500mAh -> 4.5A. El carregador però, només ens permet fins a 5 Ampers. En quant a voltatge, el fixarem a 11.1V.

Per equilibrar les cel·les, en cas que una tingués més voltatge que l'altre, existeix el mode de càrrega "BALANCE". Aquest mode carrega molt més a poc a poc, ja que per tal de

mantenir un voltatge igual entre elles, el que fa és descarregar constantment la que està per sobre de la resta.

Sabrem que la bateria està carregada quan el voltatge total estigui als 12.6 V o al voltatge per cel·la, als 4.2V.

Una vegada la bateria estigui carregada, desconnectarem seguint l'ordre invers de connectar.

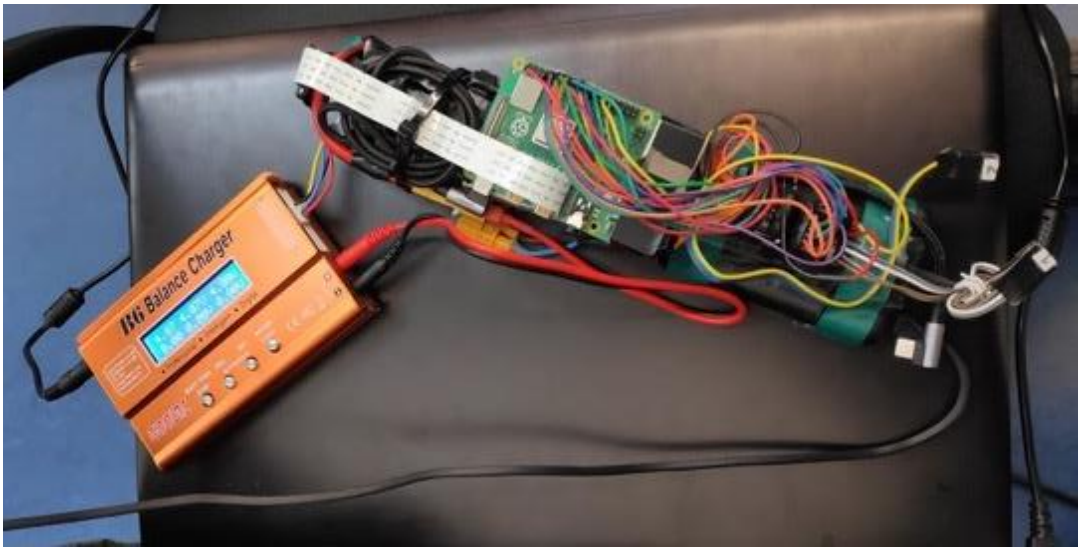


Fig. 7.2 Càrrega simultània de les dues bateries

Si volem que el R2B2 es connecti automàticament, cal configurar el WiFi. Això ho podem fer fàcilment des de la versió desktop de la Raspberry Pi. És recomanable utilitzar el hotspot portàtil del nostre mòbil. En cas que per alguna raó no ho poguéssim fer, sempre podríem utilitzar un router qualsevol. D'aquesta manera, el mínim necessari per poder fer funcionar el robot és un smartphone amb l'app instal·lada i el submarí. L'adreça IP definida a l'aplicació mòbil és 192.168.43.227 per tant, caldrà definir des de la Raspberry que es connecti sempre amb aquesta IP estàtica. També podríem canviar-la, però això voldria dir també haver de canviar el codi de la app.

Abans de tancar el contenidor haurem d'endollar tant la Raspberry com la bateria pels motors, a més de connectar l'USB de l'antena WiFi. A més, és recomanable posar una mica de paper de vàter o de cuina entre el tap i tota l'electrònica per poder-nos adonar ràpidament en el cas d'alguna fuita d'aigua petita, i també per identificar-la i poder-la corregir en la següent immersió.

Funcionament

Tenint ja el WiFi portàtil obert i el robot engegat amb l'antena i els motors connectats, podem obrir l'aplicació mòbil i ja podrem veure el que veu la càmera de l'R2B2. A més, la temperatura de la CPU de la Raspberry s'anirà actualitzant cada segon. A partir d'aquí, i abans de posar-lo dins l'aigua, farem un petit test dels quatre motors prement, per

exemple, “up”, “down”, “forward”, “backward”. Si tot funciona bé, ens assegurem una altre vegada que estigui ben tancat l’encapsulament i ja el podrem posar a l’aigua. Una vegada dins l’aigua, comprovem que estigui ben equilibrat i tirem també la boia amb l’antena a dins l’aigua. A partir d’aquest moment, amb l’app del mòbil, ja podrem controlar el robot totalment.

Recordar que l’antena té una llargada d’uns 2.5 metres, i que per tant, no podrem baixar a més d’aquesta profunditat.

Els controls de la app són els següents:

Missatge	**Utilitat**
P+(q)	Pujem la potència dels motors en un 10%
P-(a)	Baixem la potència dels motors en un 10%
UP	Ens movem va cap amunt
DW	Ens movem cap avall
R+	Girem a la dreta
R-	Girem a l’esquerra
FL	Ens movem endavant girant lleugerament a l’esquerra
FW	Ens movem endavant
FR	Ens movem endavant girant lleugerament a la dreta
BL	Ens movem endarrera girant lleugerament a l’esquerra
BW	Ens movem endarrera
BR	Ens movem endarrera girant lleugerament a la dreta
STOP	Aturem tots els motors
FOLLOW	Comencem l’script de seguiment del peix vermell.
	Aturem l’script de seguiment del peix vermell.
TEMP	Actualizem la temperatura
PwrOff	Apaguem la Raspberry Pi
PIC	Fem una fotografia.
REC	Comencem a gravar
	Parem de gravar

Taula 7.1 Controls per l'aplicació mòbil

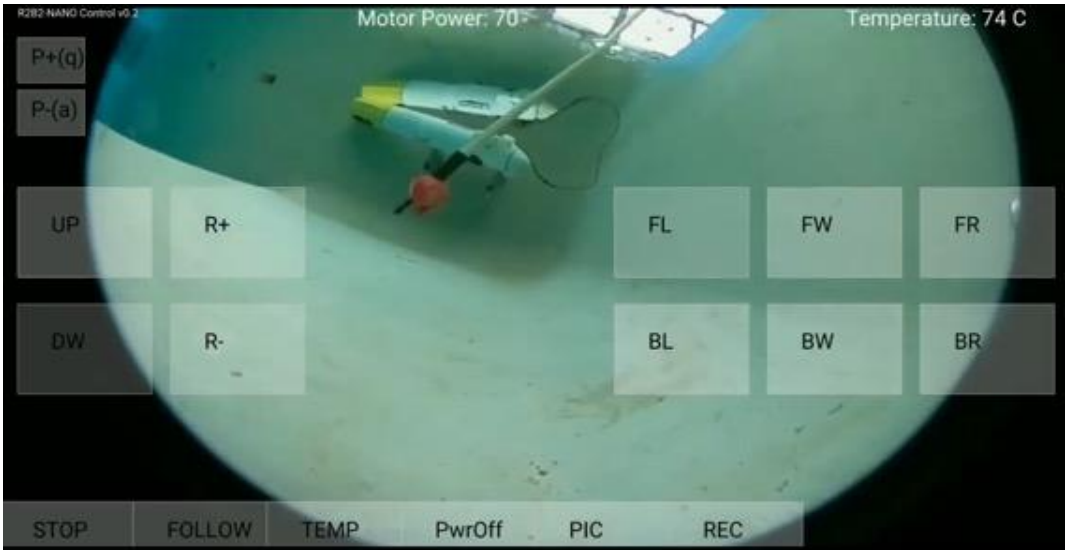


Fig. 7.3 Aplicació mòbil

Capítol 8 Pressupost

En aquest apartat es detalla el cost total del que costaria construir un R2B2. No es té en compte tot el que s'ha necessitat comprar per fer proves, testing i desenvolupament necessari. Per tant, el cost aproximat dels materials per crear una còpia exacte d'aquest projecte és de QUATRE-CENTS VINT EUROS (420 €). Cost en detall a la taula 8.1:

Components	Import en €
GoolRC carregador per balancejar LiPo + adaptador AC	30,00
3S, 50C, 6000mAh bateria de Li-Po	44,00
Raspberry Pi 4 model B – 4GB DDR4 RAM	65,00
Càmera Raspberry	26,00
Powerbank per la Raspberry	28,00
Encapsulament de Blue robòtics	143,00
4 motors	60,00
Controladors de motor TB6612FNG	3,00
Cables pont & USB-C	15,00
Silicona+Cinta aillant+ altres	6,00
TOTAL	**420,00 €**

Taula 8.1 Cost total dels materials del robot

Apart dels materials s'hauria d'afegir les hores de feina necessàries per a realitzar el muntatge. Calculo, que amb l'experiència d'haver desenvolupat i muntat un prototip, tardaria unes 24h (3 dies laborables) en posar en funcionament un nou robot. Per tant, tenint en compte que el salari mitjà d'un enginyer informàtic en acabar la carrera ronda als 22.000 € burts l'any, el cost en euros dels 3 dies treballats en el muntatge seria de DOS-CENTS QUARANTA EUROS (240 €). Si sumem aquest cost als 420 € del cost dels materials suposaria un cost total pel prototip de SIS-CENTS SEIXANTA EUROS (660 €).

Capítol 9 Conclusions i treballs futurs

S'han assolit exitosament tots els objectius proposats del projecte. En aquest sentit, per tant, el meu projecte està completat, però això no vol dir que el robot sigui perfecte. Hi ha molta feina per fer i moltes coses a perfeccionar. El fet de que el robot disposi d'una càmera obre la porta a un munt d'aplicacions de visió per computador i/o comportaments autònoms. A més es podrien fer canvis a l'electrònica i com per exemple posar un set de càmeres duals amb dues Pi zero per calcular geometries i fer reconstruccions 3D, afegir un giroscopi o un baròmetre per obtenir dades de pressió o desenvolupar un sistema d'estabilització automàtica activa.

L'estructura exterior es podria seguir perfeccionant. El meu prototip tenia en compte que la llargada del robot seria de 15-20 cm i ha acabat sent d'uns 30. A més, es podrien posar un suports per aguantar i col·locar bé els motors o unes aletes per ajudar a equilibrar el robot dins l'aigua.

En quant a l'aplicació Android, processing permet fer coses bàsiques de forma relativament fàcil de programar, però el que es pot fer al final resulta ser limitat i la majoria de llibreries per Android s'estan abandonant i perdent tot el suport. Com a treball futur, si es volgués fer l'aplicació més complexa (menús, pop-ups, etc.) s'hauria de traslladar a Android Studio.

Les connexions a través de l'antena WiFi són una mica inconsistents. El fet que els USBs no siguin fiables després de tallar-los i soldar-ne els quatre cables va enrederir molt el projecte. S'hauria de buscar una altre manera per treure una antena fora l'aigua. Com que la Pi 4 no permet soldar-ne una, s'hauria de fer servir la versió de la Raspberry Pi 3 o anteriors, que si que ho permeten, tot i que no és una tasca fàcil i posa en risc la placa.

De cares a poder utilitzar aquest projecte per a alumnes d'institut, també hi hauria la possibilitat d'habilitar un mode de programació a través d'Scratch. Actualment a Vicorob ja es fan tallers amb nens i nenes de totes les edats on es programen robots amb aquest llenguatge de programació per blocs, cada vegada més estès. A més després es proposen una sèrie d'activitats que han de dur a terme els robots i així els alumnes aprenen a programar i a desenvolupar la seva creativitat d'una manera totalment interactiva. Si es pogués programar el ROV d'aquest projecte amb Scratch, es podrien fer tallers amb ell.

També hi ha un seguit de petites millores que es podrien fer a curt termini: per exemple, una petita millora podria ser calibrar la segmentació de manera automàtica utilitzant algorismes de clustering com el region growing o ISODATA (Iterative Self-Organising Data Analysis Technique Algorithm), i d'aquesta manera, podríem preparar el robot per seguir un objecte d'una manera més ràpida i senzilla, sense haver de fer una cal·libració manual cada vegada. Una altra petitat millora podria ser acabar de perfeccionar la flotabilitat. Per exemple si haguéssim utilitzat un software més potent (segurament més car) per fer el model i disseny del robot segurament hauríem pogut simular els pesos i el centre de masses del robot i tenir en compte aquesta informació per calcular la

flotabilitat del robot. També un altre aspecte que ens interessava de principi és que el robot fós “jugable” i en aquest sentit hem procurat que pogués anar depressa. Una possible petita millora podria ser fer testos i documentar-los sobre la velocitat del robot, la bateria, l’acceleració, etc. Finalment, en el mode de seguiment d’un objecte, podríem afegir un botó al mòbil per canviar el que es veu de càmera normal a visió aplicant la segmentació.

La Raspberry és molt potent i es poden fer moltes coses, però gràcies al protocol MQTT i el disseny del funcionament del robot, podem executar moltes coses fora el robot i el resultat segueix sent el mateix. Això permet fer moltíssimes coses i per tant obre la porta a altres projectes que es podran fer partir d’aquest. El límit és la imaginació!

Capítol 10 Apèndix

Avantatges i desavantatges eix motor

Un dels problemes d'un del model de referència portat a terme en el TFM titulat "Plataforma col·laborativa R2B2" (Fig. 2.2) va ser que el R2B2 no anava gaire ràpid. Tot i els motors girar a gran velocitat no agafava gaire velocitat molt probablement degut a les hèlixs que tenia. És per això que en la versió 1 del robot es volia utilitzar unes hèlixs comprades, especials per barques o vehicles de radio control aquàtics.

El principal problema que suposa fer aquest canvi és que les hèlixs són massa grans. Per tant havia de trobar una manera per adaptar l'eix del motor de 1mm a la hèlice i a més, haver de encapsular el motor per fer-lo resistent a l'aigua.

Fig. 10.1 Comparació de l'hèlix amb el motor de la versió 1.

A continuació es relata les diferents opcions que vaig estudiar, idear i implementar.

Opció 1:

La primera idea va ser intentar subjectar l'helix a l'eix amb una cola forta. Emplenar el forat de l'hèlice amb un material que al assecar-se s'endureixi i fer-li el forat de 1mm. Vaig utilitzar la cola Loctite. Els avantatges eren que no augmentem gaire el pes i és fàcil de fer. En quant a desavantatges, va resultar no ser molt resistent, l'eix del motor entrava i sortia massa fàcilment, i l'hèlice no reutilitzable.

Fig. 10.2 Hèlixs amb cola a l'interior

Opció 2:

Comprar un acoblament d'eix de transmissió. Aquests serveixen per allargar o canviar el diàmetre d'un eix de motor. Són molt utilitzats. Els avantatges eren la robustesa, quedava professional i que existeix una gran varietat de conversions. En quant a desavantatges però, era una compra atípica i per tant difícil d'aconseguir ràpidament, afegia més pes. A més, com que s'enganxen als 2 eixos amb visos, l'eix del nostre motor sobresortiria massa poc i no es podria enganxar bé.

Fig. 10.3 Acoblament de l'eix. Font: Ebay

Opció 3:

Una altre opció era posant l'eix dins la hèlice i xafant-lo amb sobre el del motor, intentant que quedin enganxats. Era fàcil de fer però l'eix del motor era una mica massa petit i no aguantava suficient o el ferro es trencava. A més, era poc elegant.

Fig. 10.4 Eix de l'hèlix aplastat sobre l'eix del motor

Opció 4:

Emplenant el forat de l'hèlice amb cola i aguantant-hi l'eix del motor a dins. El resultat en quan a resistència era dels millors, tot i això l'esforç i temps de fabricació era molt elevat i en cas que es trenqués era difícil de reutilitzar o reenganxar.

Per tal d'allargar l'eix vam provar d'afegir un tub de metall petit, que s'adaptava millor a la mida de l'eix original. No aguantava suficient ni a pressió, ni enganxat amb una cola per a metalls.

Fig. 10.5 Eix del motor recobert amb un altre eix enganxat amb cola

Opció 5:

Ja havia provat posar a pressió l'eix dins el forat de l'hèlix tapat amb cola i no havia resultat però el material era relativament flexible i això feia que no oferís prou resistència. Fent un forat de 0.9mm en el plàstic de la fresadora utilitzat també en el xassís i posant a pressió l'eix del motor de 1mm de diàmetre aconseguíem una resistència increïble amb tant sols uns pocs mil·límetres de penetració.

Vàrem dissenyar una peça cilíndrica del mateix diàmetre que el motor i de 9mm d'altura. En un costat hi vam fer el forat de 0.9 mm i en l'altre un forat de mètric 3.

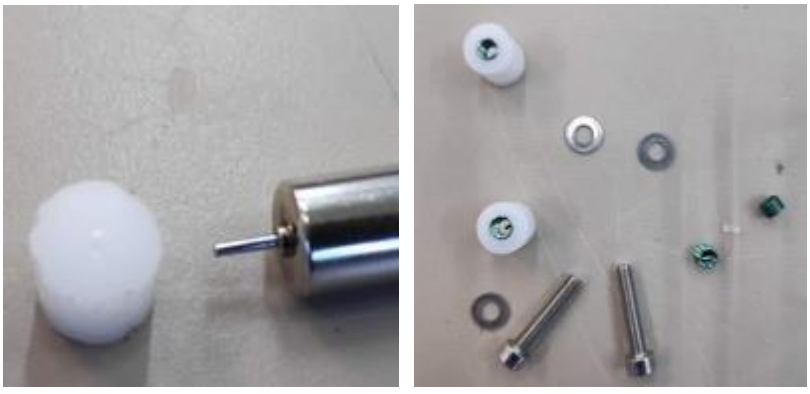


Fig. 10.6 Motor amb l'acoblament per l'eix (esq.) - Acoblament amb l'helicoil instalat (dreta)

Hi posem l'helicoil[19] (rosca de color verd per fixar els cargols), el cargol i unes femelles per distribuir la força més equitativament per tot el cilindre. D'aquesta manera obtenim unes hèlixs resistents, intercanviables, i relativament fàcils de fabricar. L'únic

desavantatge és que si poses i treus moltes vegades aquestes hèlixs, al final acabaran perdent resistència i s'hauran de fer de nou.

Fig. 10.7 Resultat final de les hèlixs

Tot i aguantar-se fort a l'eix, no era ni molt menys perfecte. Al muntar els motors amb les hèlixs al robot em vaig adonar que en la meitat de les hèlixs que havia fet, l'eix o l'hèlix en si no estaven alineats. Això es nota sobretot en el moment en que dones potència als motors:

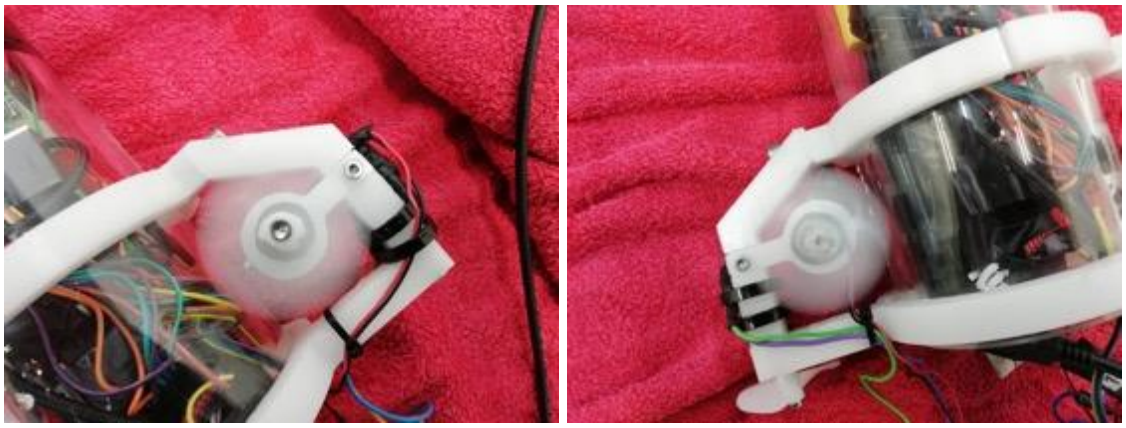


Fig. 10.8 Demostració visual de quan l'eix queda centrat (esq.) i quan no (dreta)

En la imatge de l'esquerra es pot veure ben marcat el vis de metall al centre de l'hèlix. Això ens indica que l'eix no es mou, només rota. En canvi a la imatge de la dreta, aquest vis es veu completament difuminat.

Eines i materials més específics

Fig. 10.9 Broques de mètric 3

Fig. 10.10 Helicoil

Bibliografia

[1] Core Electronics. (2020) MQTT. Recuperat Març 2020 des de https://core-electronics.com.au/tutorials/getting-started-with-home-automation-using-mqtt.html

[2] App Code Labs. (2020) MQTT. Recuperat Març 2020 des de https://appcodelabs.com/introduction-to-iot-build-an-mqtt-server-using-raspberry-pi

[3] OpenSCAD. (2020) Datasheet. Recuperat Març 2020 des de http://www.openscad.org/cheatsheet/index.html

[4] Top Tech Boy. (2020) RaspberryPi IP càmera. Recuperat Març 2020 des de http://www.toptechboy.com/tutorial/low-cost-raspberry-pi-ip-camera/

[5] RaspberryPi. (2020) Raspberry Pi 4. Recuperat Març 2020 des de https://www.raspberrypi.org/

[6] Rogers Hobby Center. (2020) Bateries LiPo. Recuperat Març 2020 des de https://rogershobbycenter.com/lipoguide

[7] Toshiba. (2020) TB6612FNG. Recuperat Març 2020 des de https://toshiba.semicon-storage.com/us/product/linear/motordriver/detail.TB6612FNG.html

[8] R2B2nano. (2018) TFM de referència. Recuperat Març 2020 des de https://r2b2-nano-docs-en.readthedocs.io/ca/latest/r2b2

[9] Wikipedia. (2020) Connected-component labeling. Recuperat Març 2020 des de https://en.wikipedia.org/wiki/Connected-component_labeling

[10] Amazon. (2020) Charmast Mini 10400mAh PowerBank. Recuperat Març 2020 des de https://www.amazon.es/Charmast-PowerBank-10400mAh-Delivery-Portable/dp/B07L921HCB/ref=sr_1_6?__mk_es_ES=%C3%85M%C3%85%C5%BD%C3%95%C3%91&keywords=powerbank%2Busb%2Bc&qid=1574808944&smid=A2SZMR92AT3KBC&sr=8-6&th=1

[11] Amazon. (2020) WiFi USB Adapter. Recuperat Març 2020 des de https://www.amazon.es/gp/product/B074RHW3M6/ref=ppx_yo_dt_b_asin_title_o03_s00?ie=UTF8&psc=1

[12] Amazon. (2020) Cable allargador USB 2.0 (3 m). Recuperat Març 202 des de https://www.amazon.es/gp/product/B00NH11PEY/ref=ppx_yo_dt_b_asin_title_o00_s00?ie=UTF8&psc=1

[13] Manual Shelf. (2020) Motor N2738-125. Recuperat Març 2020 des de https://www.manualshelf.com/manual/igarashi/2738-125-gc-5/datasheet-german.html

[14] Ipoweradd. (2020) POWERADD Energy Cell Power Bank 5000mAh http://es.ipoweradd.com/prod_view.aspx?TypeId=10&Id=458&FId=t3:10:3

[15] Amazon. (2020) HRB RC Batería de Li-Po de XT60 Plug (3S, 50C, 11.1V,6000mAh). Recuperat Març 2020 des de https://www.amazon.es/dp/B07MPXD4MV/ref=emc_b_5_t?th=1

[16] RaspberryPi. (2020) Camera module V2. Recuperat Març 2020 des de https://www.raspberrypi.org/products/camera-module-v2/

[17] Amazon. (2020). Fisheye raspberryPi càmera. Recuperat Març 2020 des de https://www.amazon.es/Waveshare-Raspberry-Camera-Fisheye-Raspberry-pi/dp/B00RMV53Z2/ref=sr_1_27?__mk_es_ES=%C3%85M%C3%85%C5%BD%C3%95%C3%91&dchild=1&keywords=raspberry+pi+camera+module&qid=1590399683&s=electronics&sr=1-27

[18] Google Code. (2013) Processing library for MJPEG stream acquisition from IP cameras. Recuperat Març 2020 des de https://code.google.com/archive/p/ipcapture/

[19] Boellhoff. (2020). Helicoil. Recuperat Març 2020 des de https://www.boellhoff.com/es-es/elementos-de-fijacion/sistemas-de-fijacion/filetes-insertos-helicoil.php