



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA



ESCUELA TÉCNICA
SUPERIOR INGENIEROS
INDUSTRIALES VALENCIA

Curso Académico:

Agradecimientos:

Le doy las gracias a mi tutor el Dr. Pedro García Gil, a los compañeros de laboratorio Alberto y Ricardo, y en especial a Rui y a Javi por todos los consejos y ayuda que me han brindado y por hacer que quisiera volver al laboratorio día tras día, y a la gente que me rodea por aguantarme los días de estrés.

Resumen

Este trabajo de fin de grado se basa en la mejora de una plataforma para la implementación de los algoritmos de control de un quadrotor.

Se comenzará el trabajo a partir de la estructura de un quadrotor ya construido controlado con Raspberry Pi con la finalidad de actualizarlo a Raspberry Pi 2 y añadirle ciertas funciones que aún no se le habían incluido, como el control de los 3 ángulos roll, pitch y yaw a la vez, y mejora de la parte central del rotor para reducir su altura.

El quadrotor se controla mediante la Raspberry Pi 2, que es un miniordenador con grandes prestaciones y de bajo precio. Se implementará con un sistema operativo el cual tiene ya incluido un parche en el kernel para adaptarlo a las funciones de Sistema Operativo de Tiempo Real, el cual es necesario para el correcto funcionamiento del quadrotor.

El miniordenador, que se conectará a un PC-Base que le mandará las respectivas órdenes en cada momento, junto con una IMU, una batería, cuatro motores con sus drivers, un convertidor de tensiones y las respectivas conexiones entre los diferentes elementos, las cuales habrá que adaptar según el protocolo de comunicación que utilicen, formarán el conjunto necesario para la implementación del aparato.

Se trabajará sobre una plataforma basada en una rótula para no limitar ninguno de los grados de libertad de giro del aparato (pitch, roll y yaw).

Palabras clave: Quadrotor, Raspberry, SOTR, Rótula.

Resum

Aquest treball de fi de grau es basa en la millora d'una plataforma per a la implementació dels algoritmes de control d'un quadrotor.

Es començarà el treball a partir de l'estructura d'un quadrotor ja construït controlat amb Raspberry Pi amb la finalitat d'actualitzar-lo a Raspberry Pi 2 i afegir-certes funcions que encara no li havien inclòs, com el control dels 3 angles roll, pitch i yaw a la vegada, i millora de la part central del rotor per reduir la seva altura.

El quadrotor es controla mitjançant la Raspberry Pi 2, que és un miniordinador amb grans prestacions i de baix preu. S'implementarà amb un sistema operatiu el qual té ja inclòs un pegat en el nucli per adaptar-lo a les funcions de Sistema Operatiu de Temps Real, el qual és necessari per al correcte funcionament del quadrotor.

El miniordinador, que es connectés a un PC-Base que li enviarà les respectives ordres en cada moment, juntament amb una IMU, una bateria, quatre motors amb els seus drivers, un convertidor de tensions i les respectives connexions entre els diferents elements, les quals caldrà adaptar segons el protocol de comunicació que utilitzin, formaran el conjunt necessari per a la implementació de l'aparell.

Es treballarà sobre una plataforma basada en una ròtula per no limitar cap dels graus de llibertat de gir de l'aparell (pitch, roll i yaw).

Paraules clau: Quadrotor, Raspberry, SOTR, Ròtula.

Abstract

This work EOG is based on improving a platform for implementing control algorithms of a quadrotor.

Work will start from the structure of a quadrotor already built controlled Raspberry Pi in order to upgrade to Raspberry Pi 2 and add certain features that were not yet included him as control of the 3 angles roll, pitch and yaw at a time, and improving the central part of the rotor to reduce its height.

The quadrotor is controlled by the Raspberry Pi 2, which is a minicomputer with high performance and low price. It will be implemented with an operating system which has already included a patch to the kernel to suit the functions of real-time operating system, which is necessary for the proper functioning of the quadrotor.

The mini-computer, which was connected to a PC-Base that will send the respective orders at all times, along with an IMU, a battery, four engines with their drivers, a converter voltage and the respective connections between the different elements, which should be adapted according to the communication protocol used, they will form the necessary apparatus for implementing the set.

It will work on a platform based on a ball joint to not limit any of the degrees of freedom of rotation of the apparatus (pitch, roll and yaw).

Keywords: Quadrotor, Raspberry, RTOS, Ball joint.

Documentos contenidos en el TFG

- Memoria: 74 páginas.
- Presupuesto: 12 páginas.
- Planos: 8 planos.

Memoria

Contenido

1	Introducción	1
1.1	Motivación.....	1
1.2	Objetivos	1
2	Contexto histórico.	3
2.1	El origen del quadrotor.	3
2.2	Los UAVs.....	4
2.3	Aplicaciones de los Quadrotors.....	9
3	Fundamento teórico del quadrotor	11
3.1	Ángulos de Tait Bryan.....	11
3.2	Movimiento en cruz o en equis.....	11
3.2.1	Configuración en cruz.....	12
3.2.2	Configuración en “equis”	12
3.3	Modelo matemático.....	13
4	Elementos del quadrotor.	17
4.1	Controladores.....	17
4.1.1	Raspberry pi 2 modelo B	17
4.2	Sensores	20
4.2.1	IMU (Unidad de medición inercial)	20
4.3	Actuadores	21
4.3.1	Motores, drivers y hélices	21
4.4	Otros elementos.....	23
5	Comunicación entre elementos	25
5.1	Impulsos PWM	26
5.2	Protocolo de comunicación I2C.....	26
5.3	Protocolo de comunicación UDP (User Datagram Protocol)	27
6	Sistema Operativo de tiempo real	29
6.1	Programación en SOTR.....	29
6.2	El SOTR en el quadrotor.	30
6.2.1	Raspbian con parche Preempt_RT	30
7	Software	33
7.1	Accionamiento de los motores.	33
7.2	Lectura y procesamiento de datos de la IMU	33
7.3	Control.....	34

7.4	Comunicación UDP	34
7.4.1	HMI	34
8	Banco de pruebas	37
8.1	Montaje	37
9	Sistema de control	41
9.1	Control de orientación	41
9.1.1	Ajuste experimental de los parámetros de control.	42
9.1.1.1	Ajuste del Roll y Pitch.	43
9.1.1.2	Ajuste del Yaw	45
10	CONCLUSIONES	47
11	FUTUROS TRABAJOS	49
ANEXO I.	Instalación del sistema operativo Raspbian	51
ANEXO II.	Establecimiento de comunicación SSH entre PC y Raspberry Pi 2.	52
ANEXO III.	Habilitar los puertos I2C para la lectura de la IMU	57
ANEXO IV.	Instalación de un compilador cruzado.	61
ANEXO V.	Programa para establecer rangos de actuación de los motores.	63
ANEXO VI.	Programa para comprobar las medidas de los ángulos de la IMUMPU 6050.	65
ANEXO VII.	Implementación del PID.....	67
12	Ilustraciones, tablas, ecuaciones y gráficas	69
12.1	Ilustraciones	69
12.2	Tablas	70
12.3	Ecuaciones.....	71
12.4	Gráficas.....	71
13	Bibliografía de referencia y de imágenes.	73

1 Introducción

1.1 Motivación

Hace ya varios años que me fascina la automática y siempre me he preguntado si realmente me gusta únicamente por el resultado o porque realmente me gusta saber del tema. Con asignaturas como informática, sistemas automáticos, tecnología automática y laboratorio de automatización mi interés se ha incrementado, sobre todo gracias a esta última asignatura.

Mi idea es hacer el máster de ingeniería industrial con especialidad de automática por tanto decidí buscar un TFG que me interesara y de ayudase a decidir si realmente me gusta la automática.

Ya que quería probarme escogí el tema de los quadrotors porque es un tema que siempre me ha interesado, así como toda la robótica en general, y, además, está en auge hoy en día.

Dentro del tipo de proyecto elegido, la principal motivación fue trabajar con la Raspberry Pi 2, un mini ordenador con características ideales para trabajar con los quadrotors y más barato que la mayoría de los usados en el laboratorio. Además, puesto que en el laboratorio no había plataformas de este tipo, se buscó un soporte sobre el que poder realizar las pruebas a los quadrotors, el cual pudiera sustituir a la plataforma Hover de Quanser, la cual tiene un precio de 22400 euros, un precio desorbitado para muchas de las actividades que se quiere realizar con una plataforma de ese estilo. Por eso se estudiará la viabilidad de una plataforma basada en una simple rotula para el estudio de las funciones del quadrotor, la cual pueda servir para futuros proyectos.

1.2 Objetivos

En este trabajo de fin de grado del grado en ingeniería en tecnologías Industriales “Desarrollo de un software de control basado en Raspberry Pi II para la implementación de algoritmos de control en una plataforma 3D (con 3 grados de libertad) “ se pretende alcanzar los siguientes objetivos partiendo de la estructura montada por Vicente Balaguer en su trabajo de fin de grado:

- Sustitución de la Raspberry Pi inicialmente instalada por la Raspberry Pi 2, la cual ofrece mayores posibilidades debido a sus mejores prestaciones y comprobación de la validez de este ordenador para esta aplicación.
- Comprobación de las conexiones de alimentación anteriormente instaladas y sustitución del material necesario. Puesto que en el anterior proyecto ya se hicieron las conexiones para la alimentación del quadrotor, se ha de revisar el conexionado y comprobar su buen estado, así como cambiar lo necesario, como es el caso de algún motor que no cumplía los requerimientos, hélices y tornillería.
- Realizar todo el conexionado de dicho miniordenador con los sensores y actuadores del quadrotor.
- Construcción de una base de trabajo basada en una rótula, la cual no limite los giros de pitch, roll y yaw, y su validación.

- Implementación de un algoritmo de control que cumpla los requerimientos de un sistema de tiempo real, cumpliendo las exigencias de tiempos de reacción, así como su validación en la plataforma.
- Buscar que el quadrotor sea compacto y cuyo peso esté centrado en su centro geométrico para que su inercia sea lo menor posible. Esto se consigue escogiendo elementos lo más reducidos posible y evitando que haya elementos a mayor altura que las hélices.

Para la consecución de todos los objetivos se lleva a cabo diversas tareas en el siguiente orden:

- Recopilación de información.
- Preparación e instalación del software necesario.
- Revisión de la estructura.
- Colocación y conexionado de los elementos.
- Implementación del programa de control.
- Ajuste del control.

Los objetivos mencionados no son lo único que se pretende alcanzar. Con este trabajo se pretende que la labor de búsqueda de información que se realice no sea un impedimento de tiempo para los proyectos futuros, es decir, hacer de este trabajo un manual que sirva de base para que proyectos de este ámbito puedan llegar más lejos en menos tiempo.

2 Contexto histórico.

La mejora del control, potencia y nuevas plataformas para UAVs está abriendo la puerta a un amplio abanico de posibilidades para realizar trabajos que hasta la fecha eran muy complicados y costosos.

Tanto como para uso privado y lúdico como con fines de cara al mundo de la empresa o de seguridad, los drones están siendo un 'boom'. La aparición de controladores, sensores, así como sistemas operativos cada vez más sencillos de manejar, y baratos, hacen que cada vez haya más gente con acceso a un quadrotor.

2.1 El origen del quadrotor.

El término Quadrotor viene del inglés *Quadrotor Helicopter* o *Quadcopter*, lo que significa que es un aparato propulsado con 4 motores y se levanta como un helicóptero (Vertical Take Off and Landing (VTOL)). Consiste de una estructura simétrica en cruz que consta de las 4 hélices localizadas en los extremos de la estructura.

El primero se construyó en el siglo 20 por Charles Richet aunque sin éxito a la hora del vuelo. Sin embargo, su alumno, Louis Bregét, consiguió en 1907 el primer cuadricóptero pilotado de la historia, aunque solo se elevó unos pocos pies.

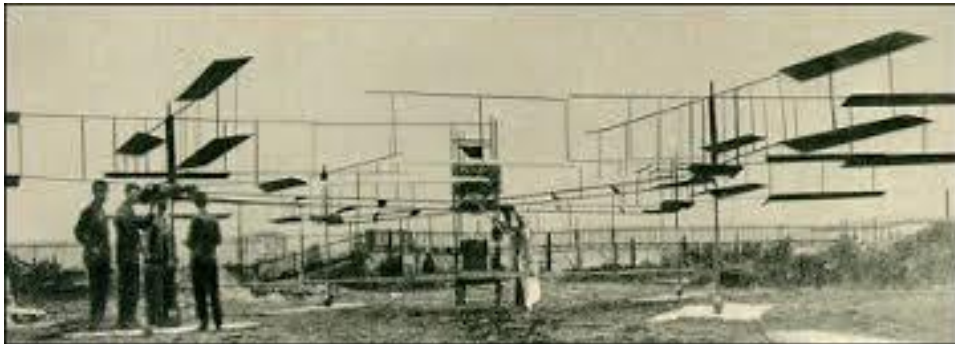


Ilustración 1: Breguet-Richet Autogiro (1907).

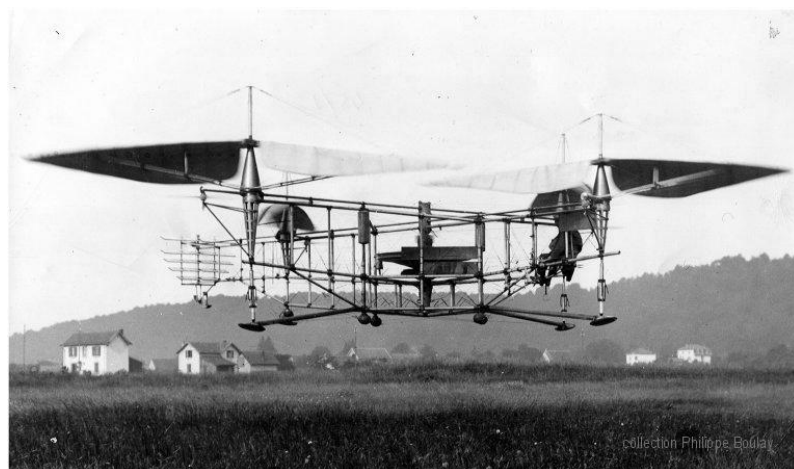


Ilustración 2: Oehmichen No 2.

Con este diseño de 4 rotores y 8 hélices (5 para el plano horizontal, que lo estabilizaban lateralmente, 1 para la dirección y 2 para la propulsión hacia delante), Oehmichen, consiguió el record de distancia, 360 metros recorridos y, posteriormente, se superó con 1 km recorrido.

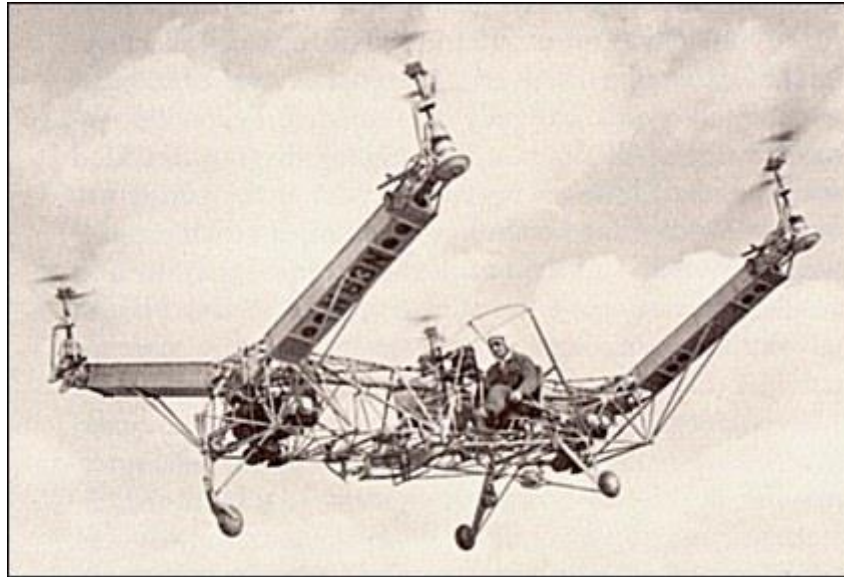


Ilustración 3: Modelo A quadrotor de Convertawings(1956).

Este modelo se diseñó para ser utilizado de prototipo para el diseño de futuros quadrotors que no tuvo mucho éxito en la demanda de pedidos. Fue un acontecimiento importante puesto que fue el primer quadrotor que consiguió un movimiento hacia delante y necesidad de otra hélice, demostró que se podía conseguir este movimiento mediante la diferencia de empuje de los rotores.

2.2 Los UAVs

Aunque es triste, como muchos grandes avances de la historia, los drones surgieron para ganar guerras.

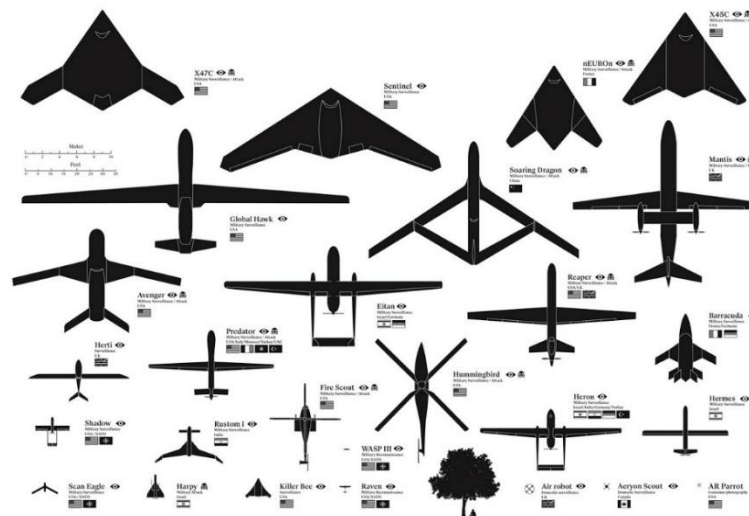


Ilustración 4: UAVs actuales (1).

Las primeras aeronaves no tripuladas fueron los 'torpedos aéreos', antecesores de los actuales misiles crucero, que se desarrollaron para bombas guiadas, blancos aéreos ('drones' en términos anglosajones), aeronaves de reconocimiento, de combate, en fin, para una infinidad de desarrollos.

Sus ventajas son, entre otras, el menor coste, no se arriesgan vidas humanas y poder acceder a sitios peligrosos o de difícil acceso.

A lo largo de los años el nombre que se le ha ido otorgando a este tipo de aparatos ha ido cambiando. Los términos referentes a vehículos aéreos no tripulados (Unmanned Aerial Vehicle,(UAV), Unmanned Aircraft System (UAS), Unmanned Aircraft Vehicle System (UAVS)) se hicieron comunes en los años 90 para describir a las aeronaves robóticas y reemplazó el término vehículo aéreo pilotado remotamente (Remotely Piloted Vehicle, RPV), el cual fue utilizado durante la guerra de Vietnam y con posterioridad. Resumido en la Ilustración 5: Cronología de los nombres aplicados a las aeronaves robóticas..

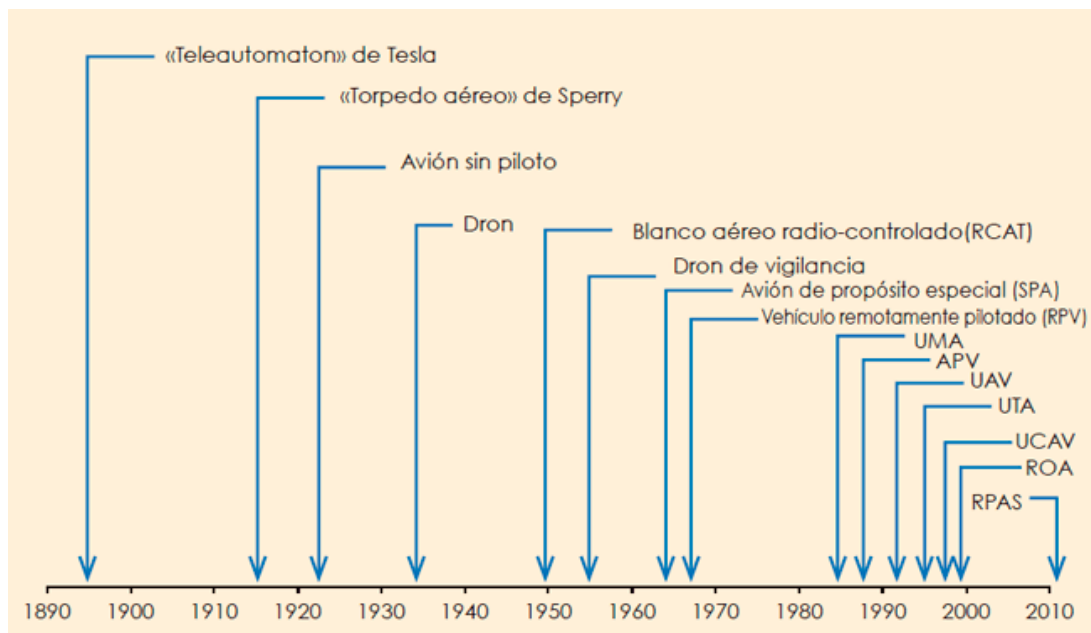


Ilustración 5: Cronología de los nombres aplicados a las aeronaves robóticas.

Existe una clasificación de los UAVs bastante amplia según su autonomía, finalidad de uso, alcance, peso, etc, en parte gracias a no necesitar habitáculo para un piloto, lo cual limita el tamaño y peso mínimos del aparato. [4]

CATEGORÍAS	Acrónimo	Alcance (km)	Altitud de vuelo (m)	Autonomía (h)	Carga máxima (kg)
Micro < 250 g	micro	< 10	250	1	< 5
Mini < 25 Kg	mini	< 10	de 150 a 300	< 2	< 30
Alcance cercano	CR	de 10 a 30	3000	de 2 a 4	150
Alcance corto	SR	de 30 a 70	3000	de 3 a 6	200
Alcance medio	MR	de 70 a 200	5000	de 6 a 10	1250
Altitud baja	LADP	> 250	de 50 a 9000	de 0,5 a 1	350
Autonomía media	MRE	> 500	8000	de 10 a 18	1250
Autonomía alta y altitud baja	LALE	> 500	3000	> 24	< 30
Autonomía alta y altitud media	MALE	> 500	14000	de 24 a 48	1500
Autonomía alta y altitud alta	HALE	> 2000	20000	de 24 a 48	12000
Combate	UCAV	1500	10000	2	10000
Ofensivo	LETH	300	4000	de 3 a 4	250
Señuelo	DEC	500	5000	< 4	250
Extratosférico	STRATO	> 2000	de 20000 a 30000	< 48	ND
EXO-estratosférico	EXO	ND	> 30000	ND	ND

ND: No definido

Tabla 1: Clasificación de los UAVs.

La mayoría de los UAVs utilizados actualmente se encuentran dentro de la categoría de micro y alcance cercano, como son: los dirigibles, los aeroplanos, helicópteros y los Quadrotos. En la siguiente tabla se puede ver la comparativa entre estos últimos mencionados.

Característica	Helicóptero	Aeroplano	Dirigibles	Quadrotor
Capacidad de vuelo estacionario	***		****	***
Velocidad de desplazamiento	***	****	****	***
Maniobrabilidad	***	*	*	****
Autonomía de vuelo	**	***	****	*
Resistencia a perturbaciones externas	**	****	*	**
Auto estabilidad	*	***	***	**
Capacidad de vuelos verticales	****	*	**	****
Capacidad de carga	***	****	*	**
Capacidad de vuelos en interiores	**	*	***	****
Techo de vuelo	**	****	***	*

(****) muy bueno, (***) bueno, (**) regular, (*) malo, () no puede realizarse esa función

Tabla 2: Clasificación de los UAVs de corto alcance.



Ilustración 6: Helicóptero UAV.

Helicópteros: poseen una hélice cuyo eje está en vertical para el sustento en el aire del aparato y otra menor, cuyo eje está en horizontal para compensar el par rotor producido por la hélice superior y llevar la dirección.



Ilustración 7: Aeroplano UAV.

Aeroplanos: su sustentación se basa en la amplia superficie de sus alas y a la forma de las mismas. Posee una hélice que propulsa el aparato. La dirección se maneja mediante un timón y alerones que forman parte de las alas.



Ilustración 8: Zeppelin UAV.

Zeppelin: su sustentación se logra gracias a depósitos de gas que contiene de menor densidad que el aire exterior. Posee un propulsor para su avance y timón para su dirección.



Ilustración 9: Quadrotor UAV.

Quadrotor: su sustento se consigue por las hélices de sus 4 motores y la dirección mediante la variación de velocidad de giro de sus motores. Este aparato es el que será explicado con mayor profundidad a lo largo del documento.

Como se observa, la principal utilidad de los quadrotors es de alcance cercano y en entornos que requieren mucha maniobrabilidad lo que lo hacen perfecto para interiores.

Las principales ventajas de los quadrotors, entre otras, son:

- Aumento de capacidad de carga, en relación a su tamaño, debido a la suma de todos los empujes provocados por las hélices de los cuatro motores.
- Pequeño tamaño y mayor maniobrabilidad, puesto que el aumento del número de motores supone una mayor precisión y rapidez a la hora de realizar cualquier movimiento.
- Motores eléctricos en vez de motores de combustión, los cuales no dejan residuos, por tanto, son ideales para interior de edificios o espacios cerrados.
- Mayor estabilidad. Sus 4 motores aportan un área de sustentación mayor.

Con las nuevas mejoras en autonomía, potencia de carga de los motores, estabilidad y precisión de movimientos, y todo esto sin perjudicar al peso del quadrotor, se le están encontrando múltiples aplicaciones.

2.3 Aplicaciones de los Quadrotors.

En un principio los quadrotors comerciales de menor coste se presentaban como un 'juguete', es decir, únicamente se centraba en su uso para el tiempo libre. Sin embargo, como he comentado anteriormente, su origen es militar y, cada vez más, se busca su utilidad profesional de forma que se ha cubierto gran parte de ámbitos profesionales en los que los quadrotors tienen una gran aplicación ya que suponen una alternativa más barata para realizar algo que antes era impensable o suponía una gran inversión de dinero. Algunos ámbitos son:

Militares

Tiene una gran utilidad en el espionaje gracias a su pequeño tamaño que lo hace más difícil de detectar por los radares. Otro factor a favor es que no se arriesgan vidas humanas en las misiones de detección de bombas, búsqueda de supervivientes, etc.

Fotografía y vídeo

Está teniendo gran aceptación por las personas con gran afición a la fotografía, debido a que es capaz de sacar fotos desde lugares donde es muy difícil el acceso para una persona. Además, gracias a los nuevos avances, está teniendo mucho éxito para los practicantes de deportes, sobre todo los amantes de deportes extremos. Existen quadrotors que te siguen automáticamente por sensor GPS y te graba desde cierta distancia.



Ilustración 10: Quadrotor con seguimiento automático (2).

También para tomar planos en películas, que hasta ahora suponía costes muy altos por la necesidad de contratar un helicóptero.

Retransmisión de eventos

Estos pequeños aparatos están facilitando mucho la retransmisión de eventos de todo tipo. Son utilizados en partidos de fútbol, para ofrecer una imagen mejor de un suceso en una noticia, etc.

Seguridad y vigilancia

Cada vez salen más noticias de utilización de drones para vigilancia. Pueden servir para localizar un incendio o cualquier desastre natural, pudiendo llegar antes que cualquier persona pudiendo transmitir información crucial para ayuda de las posibles víctimas.

Servicio de vigilancia para policías para eventos y situaciones de riesgo es, también, una gran aplicación.

3 Fundamento teórico del quadrotor

Un quadrotor es un aparato que se sostiene en el aire gracias a sus hélices y precisamente por estar sostenido en el aire tiene gran capacidad de movimientos tanto sobre los ejes X, Y y Z, como sobre los ángulos de Tait-Bryan.

3.1 Ángulos de Tait Bryan

Ángulos de Tait-Bryan: se utilizan para describir la orientación del quadrotor en el espacio respecto al marco de referencia fijo, y son: alabeo, cabeceo y guiñada, aunque son más conocidos por sus nombres en inglés (roll, pitch y yaw respectivamente) los cuales se utilizan de ahora en adelante (3).

- **Alabeo (roll)**, es la rotación respecto al eje OX y está denotada por la letra griega ϕ .
- **Cabeceo (pitch)**, es la rotación respecto al eje OY y está denotada por la letra griega θ .
- **Guiñada (yaw)**, es la rotación respecto al eje OZ y está denotada por la letra griega ψ .

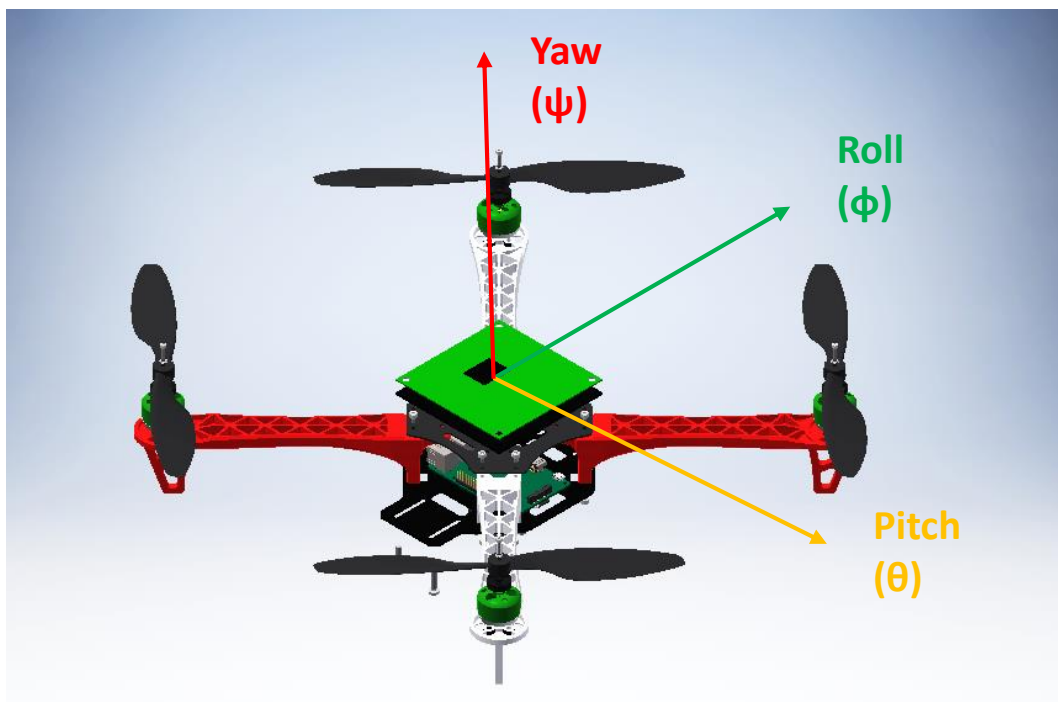


Ilustración 11: Ángulos de Tait-Bryan (Roll, Pitch, Yaw) en configuración X.

3.2 Movimiento en cruz o en equis

La elección de la configuración se basa principalmente en el número de motores y la potencia de éstos. Esta elección solo afecta a los ángulos pitch y roll (4).

3.2.1 Configuración en cruz

Los ejes de los ángulos de rotación coinciden con los brazos del aparato, por tanto, tan solo utiliza un motor para cada rotación, lo cual hace necesitar una mayor potencia de los motores.

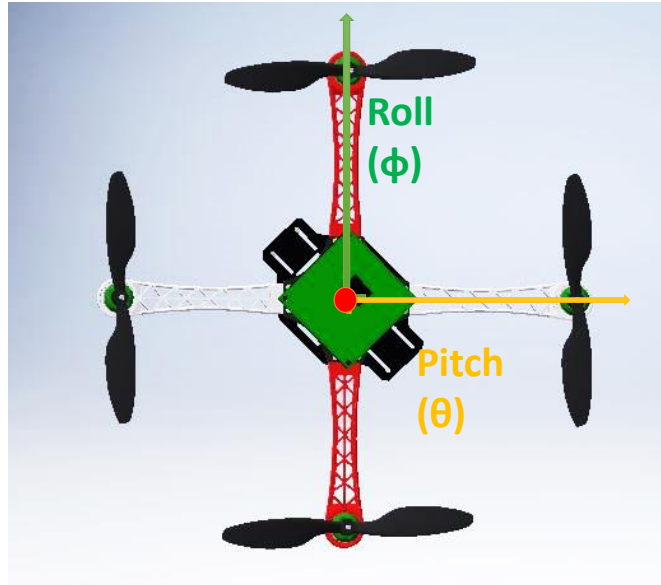


Ilustración 12: Configuración en cruz.

3.2.2 Configuración en "equis"

Los ejes están en la bisectriz de los brazos del quadrotor, es decir, a 45° de cada uno de los brazos. Esta configuración no requiere tanta potencia para la rotación en cada uno de los ejes, puesto que hay 2 motores para realizar el giro. Un punto en contra es que el algoritmo de control de esta configuración es algo más complejo que el anterior.

Esta configuración, además, permite que los aparatos tengan cámaras de foto o vídeo, ya que ningún brazo del rotor se sitúa en el campo de visión de ésta.

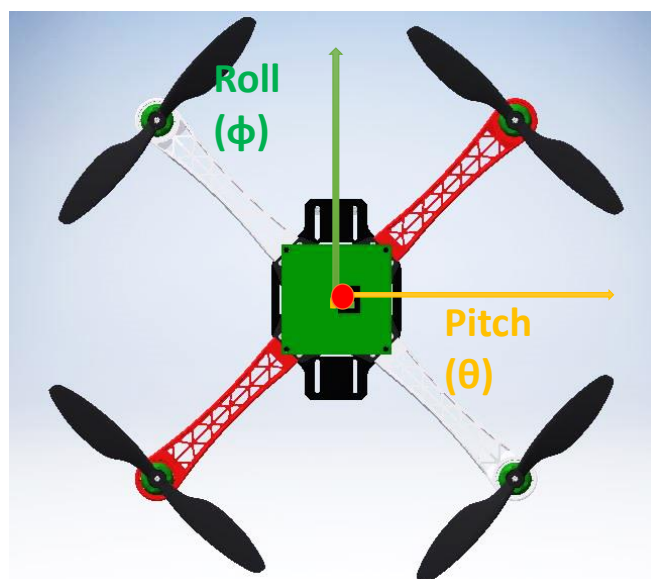


Ilustración 13: Configuración en X.

3.3 Modelo matemático

Para analizar los movimientos del quadrotor se parte de una base teórica ideal, la cual posteriormente a la hora de llevarla a la práctica real en el quadrotor habrá de ser adaptadas teniendo en cuenta parámetros que no se pueden predecir hasta ese momento (como son las pérdidas por rozamiento de motores, que la estructura del dron no sea perfecta, aerodinámica del aparato, etc).

Suponiendo un quadrotor con los orientado como en la Ilustración 14: Orientación del quadrotor en relación con sus motores y ejes.

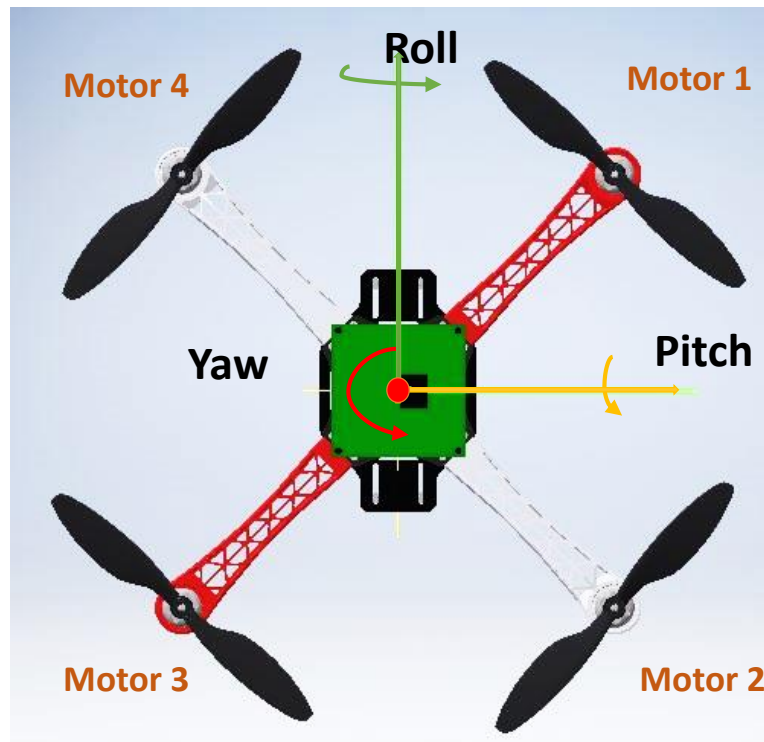


Ilustración 14: Orientación del quadrotor en relación con sus motores y ejes.

Las ecuaciones de comportamiento del quadrotor se pueden obtener mediante aproximación de Euler-Lagrange (5):

$$m\ddot{x} = -u \sin \theta \quad \text{E.1}$$

$$m\ddot{y} = u \cos \theta \sin \varphi \quad \text{E.2}$$

$$m\ddot{z} = u \cos \theta \cos \varphi - mg \quad \text{E.3}$$

Donde u es el empuje total de los 4 rotores, u_x , donde x son los ángulos ψ , θ , φ , son los empujes correspondientes a cada ángulo, m la masa del quadrotor, \ddot{x} , \ddot{y} , \ddot{z} son las aceleraciones lineales en cada eje.

El movimiento del uav se origina a partir de los cambios de velocidad de los rotores. Para que el aparato se desplace en una dirección, sobre el plano XY, los motores de detrás han de aumentar su velocidad de giro y los de delante, simultáneamente, han de disminuir para conseguir un giro en el ángulo correspondiente al eje de desplazamiento que produzca que el empuje total tenga componente en ese eje. Para el ángulo *Yaw* es algo distinto. Éste se produce por diferencia de pares de torsión de cada par de rotores (diferencia de los que giran en mismo sentido con los que giran en sentido contrario) (6) (7).

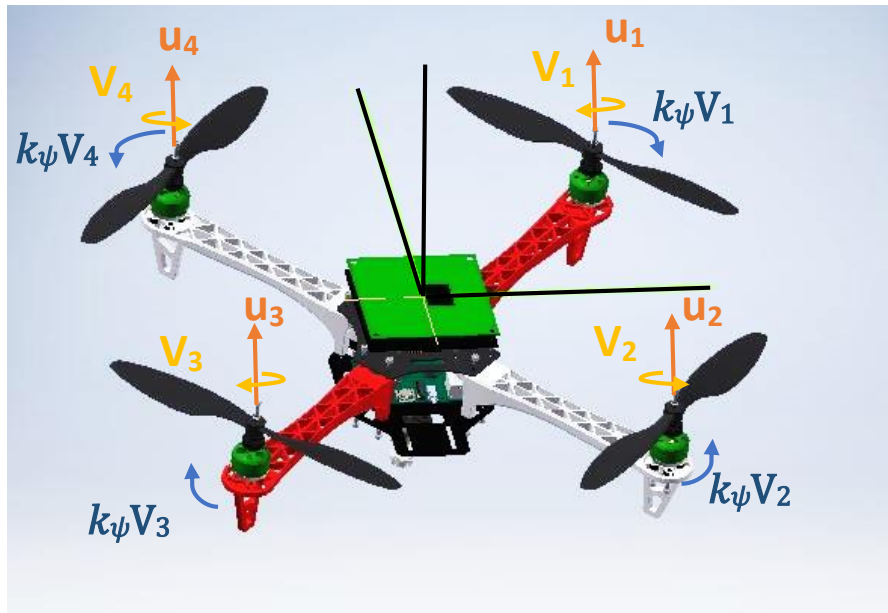


Ilustración 15: Fuerzas y sentidos de giro de las hélices.

Por tanto, las aceleraciones referidas a cada ángulo en función de la velocidad de giro de cada rotor y sus inercias se modelan de la siguiente manera:

$$\dot{\psi} = u_{\psi}$$

$$u_{\psi} = k_{\psi}(-V_1 + V_2 - V_3 + V_4) \quad \text{E.4}$$

$$\dot{\theta} = u_{\theta}$$

$$u_{\theta} = k_{\theta}(V_1 - V_2 - V_3 + V_4) \quad \text{E.5}$$

$$\ddot{\varphi} = u_{\varphi}$$

$$u_{\varphi} = k_{\varphi}(-V_1 - V_2 + V_3 + V_4) \quad \text{E.6}$$

Donde cada K se refiere a los momentos de inercia respecto a cada ángulo y las V_x la velocidad de giro de cada uno de los motores en valor absoluto y ψ , θ , φ son las aceleraciones angulares de los ángulos ψ , θ , φ .

Las ilustraciones Ilustración 16 Ilustración 17 Ilustración 18 explican de manera clara los movimientos en función de la velocidad y el sentido de cada uno de los motores en configuración en cruz, pues es más fácil la explicación y comprensión. El grosor de las flechas indica la velocidad mayor de cada motor.

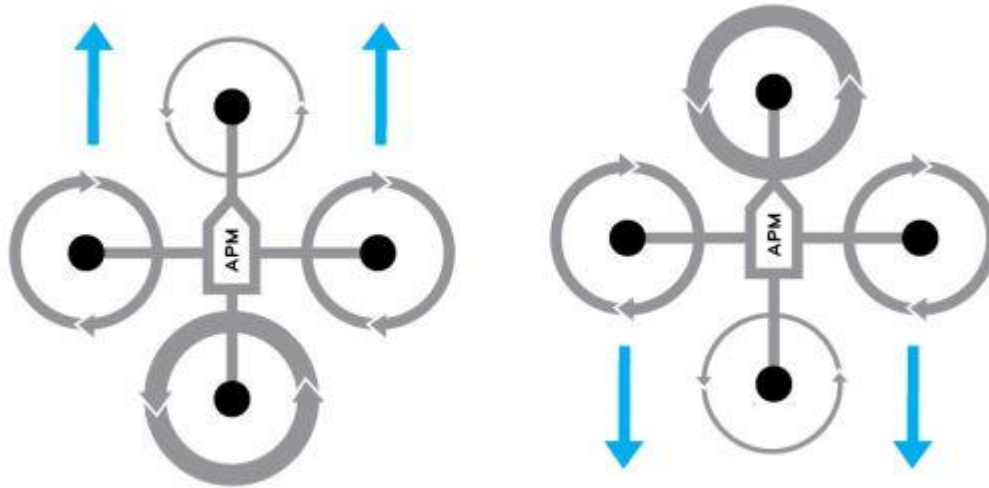


Ilustración 16: Movimientos longitudinales quadrotor (8)

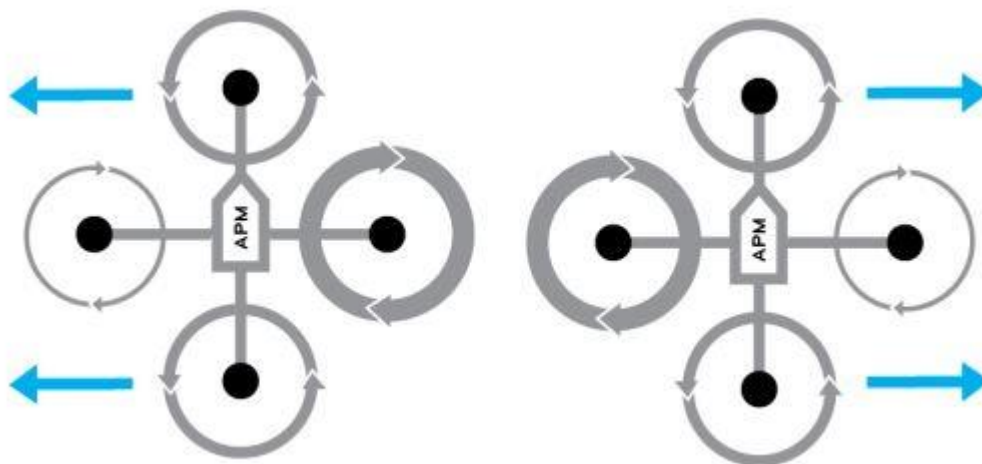


Ilustración 17: Movimientos laterales (8).

Como se observa en las imágenes, dependiendo del motor que gire a mayor velocidad se provoca una desestabilización del quadrotor que se traduce en un empuje con proyección horizontal, que provoca el desplazamiento del aparato.

En la configuración en equis estos giros se producen basados en la misma técnica. Los motores traseros con respecto a la dirección y sentido que de desee desplazar el quadrotor son los que han de provocar la desestabilización.

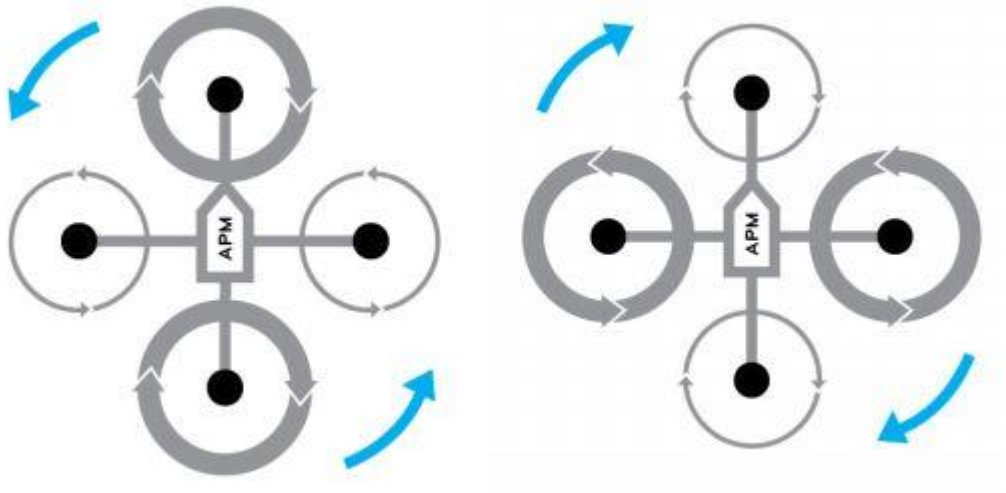


Ilustración 18: Movimientos de rotación (8).

Esta imagen muestra los giros en Yaw. Como se ha explicado anteriormente, cada motor, por su sentido de giro, provoca una fuerza que tiende a girar el quadrotor en Yaw. Como los motores están dispuestos 2 a 2, es decir que los que giran en el mismo sentido están enfrentados, permite un giro en este ángulo sin provocar una desestabilización en el resto de ángulos.

En la configuración en cruz del quadrotor este giro se produce de la misma manera que en la configuración en X.

4 Elementos del quadrotor.

El quadrotor es un aparato que funciona gracias a la interacción de varios elementos que han de estar en perfecto estado porque todos son de suma importancia.

Los elementos principales son: los controladores, los actuadores y los sensores, pero además se necesita de otros elementos que sirven de enlace para la comunicación entre los distintos componentes.

4.1 Controladores

Se trata de elementos que contienen un software que es capaz de comunicarse con los distintos dispositivos del aparato.

Estos elementos han de estar conectados con los sensores, de quienes reciben información del exterior, y con los actuadores, a quienes les procesan la información recibida para determinar qué acción han de realizar para conseguir así un mejor funcionamiento.

En este trabajo se trabaja con Raspberry Pi 2, un ordenador placa reducida y de pequeño tamaño perfecto para las dimensiones del quadrotor. Sobre este ordenador se profundiza más adelante.

4.1.1 Raspberry pi 2 modelo B

Es el ordenador que actúa como controlador del quadrotor. Toda la información pasa por él. Recibe las acciones del PC que ejecuta teniendo en cuenta las medidas que recibe de los sensores en cada momento.



Ilustración 19: Raspberry Pi 2 (9)

Este microcontrolador se conecta con el PC mediante wifi y con los drivers de los motores y la IMU a través de los pins *GPIO* (*General Purpose Input/Output*, Entrada/Salida de Propósito General). La lectura de la IMU se recibe por los GPIO con el protocolo I2C y para los motores se programa los GPIO conectados a estos para que se les envíe señales PWM simuladas.

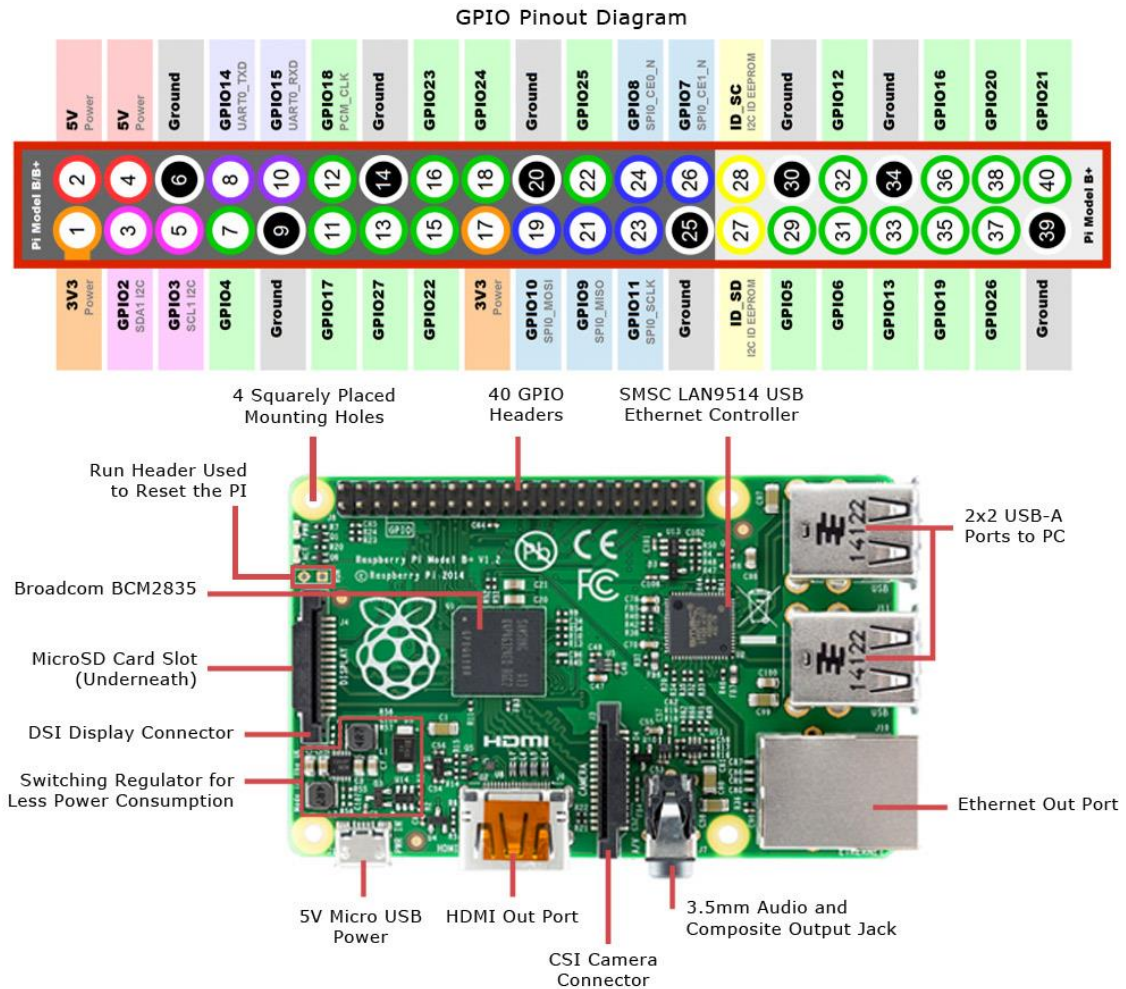


Ilustración 20: Elementos que componen la Raspberry Pi 2 modelo B (10).

Este controlador es una actualización del modelo usado anteriormente Raspberry Pi modelo B. Esta actualización tiene mucha mayor potencia y mayores posibilidades a la hora de desarrollar distintos proyectos (mejora bastante a su predecesora en cuanto a CPU, velocidad y cantidad de memoria).

Las mejoras se observan, sobretodo, a la hora de la estabilidad y de respuesta cuando se ajuste el control, ya que la velocidad de procesamiento de datos es muy necesaria para proyectos que requieren sistemas de tiempo real.

En la Tabla 3 se comparan las especificaciones de la Raspberry Pi 2 y su modelo anterior, el usado para el anterior proyecto.

	RASPBERRY PI MODELO B+	RASPBERRY PI 2 MODELO B
SoC	Broadcom BCM2835	Broadcom BCM2836
CPU	ARM11 ARMv6 700 MHz	ARM11 ARMv7 ARM Cortex-A7 4 núcleos @ 900 MHz
Overclocking	Sí, hasta velocidad Turbo; 1000 MHz ARM, 500 MHz core, 600 MHz SDRAM, 6 overvolt. de forma segura	Sí, hasta arm_freq=1000 sdram_freq=500 core_freq=500 over_voltage=2 de forma segura
GPU	Broadcom VideoCore IV 250 MHz. OpenGL ES 2.0	Broadcom VideoCore IV 250 MHz. OpenGL ES 2.0
RAM	512 MB LPDDR SDRAM 400 MHz	1 GB LPDDR2 SDRAM 450 MHz
USB 2.0	4	4
Salidas de vídeo	HDMI 1.4 @ 1920x1200 píxeles	HDMI 1.4 @ 1920x1200 píxeles
Almacenamiento	microSD	microSD
Ethernet	Sí, 10/100 Mbps	Sí, 10/100 Mbps
Tamaño	85,60x56,5 mm	85,60x56,5 mm
Peso	45 g	45 g
Consumo	5v, 600mA	5v, 900mA, aunque depende de la carga de trabajo de los 4 cores
Precio	35 dólares	35 dólares

Tabla 3: Especificaciones de Raspberry Pi y Raspberry Pi 2 (11).

El sistema operativo utilizado en la Raspberry Pi 2 ha sido Raspbian ya que es el sistema más utilizado y el mejor valorado. Además, por ser un sistema tan utilizado supone la existencia de una mayor cantidad de información para la resolución de posibles problemas.

Un requisito indispensable para la consecución de este proyecto es que la Raspberry tenga un sistema operativo de tiempo real, y lamentablemente Raspbian no lo es, por tanto, se ha instalado un parche para que el sistema operativo trabaje como un sistema de tiempo real, y es el PREEMPT_RT (explicado con mayor profundidad en el apartado 6. *Sistema operativo de tiempo real*)

* La instalación del sistema operativo se detalla en el ANEXO I.

** La habilitación de los puertos i2c para la lectura de la IMU se detallan en el ANEXO III.

4.2 Sensores

Son dispositivos capaces de traducir información del exterior de forma que pueda ser procesada y utilizada por el controlador.

En el Quadrotor el sensor más importante es la IMU (Unidad de Medición Inercial), la cual es capaz de transmitir información de aceleraciones y velocidades angulares y, a partir de estas, se puede obtener la orientación del aparato (ángulos Pitch, Roll y Yaw). Aunque se pueden conectar todos los sensores que se quiera dependiendo de la información que se necesite.

4.2.1 IMU (Unidad de medición inercial)

IMU MPU6050 10dof v2

Dispositivo electrónico que utiliza una combinación de acelerómetros y giróscopos en los 3 ejes que permite obtener la orientación a partir de las medidas de la velocidad y aceleración angular. Además, permite la conexión de un magnetómetro para controlar la altura (12).

Se conecta a la raspberry mediante los GPIO y se comunica mediante el protocolo I2C (ver Protocolo de comunicación I2C)

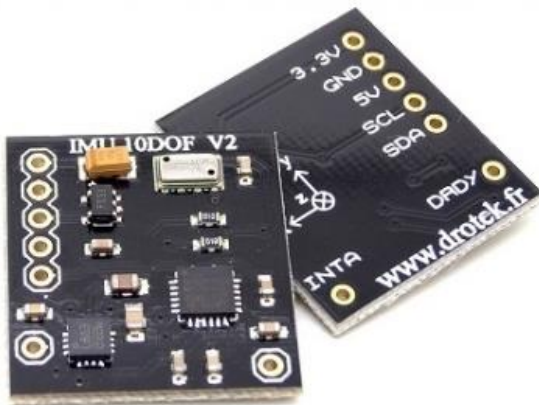


Ilustración 21: Orientación de los ejes de la IMU (12).

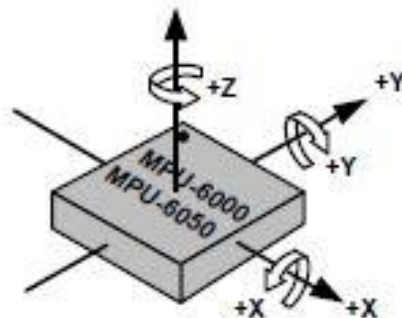


Ilustración 22: IMU MPU6050 10Dof v2

(12).

La IMU se coloca para en una placa de circuitos para una mayor facilidad a la hora del conexasión y, además, al tener mayor superficie se puede controlar mejor su correcta colocación.

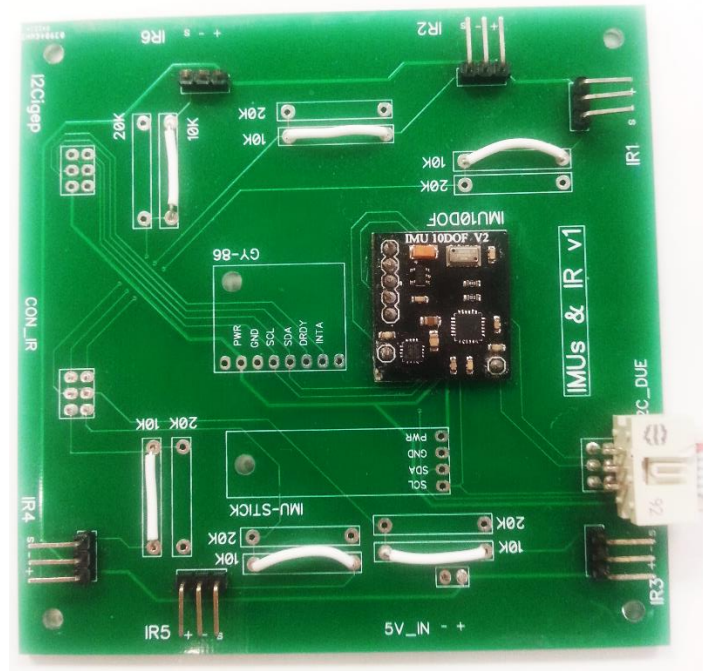


Ilustración 23: IMU MPU6050 10DoF v2 en la placa de circuitos.

4.3 Actuadores

Dispositivos capaces de transformar una señal (eléctrica en nuestro caso) emitida por el controlador para generar una reacción en un aparato que ha de hacer un proceso.

Los más importantes, y únicos, en este proyecto son los cuatro motores, sin los cuales no sería posible el vuelo del Quadrotor.

4.3.1 Motores, drivers y hélices

Se utilizan 4 motores *Robbe Roxxy*, modelo *Brushless BL Outrunner 2824-34* [5], unos motores trifásicos con el bobinado en el estator y los imanes permanentes en el rotor que funcionan por impulsos PWM. Generan una potencia máxima de 90W, 1100 KV (rpm/V). Esto significa unas revoluciones por minuto de 1100 por voltio aplicado y su rango de trabajo es de 7 a 12V (13).



Ilustración 24: Motores Brushless BL Outrunner 2824-34.

Las hélices utilizadas son unas hélices genéricas de 10x4.5". Son las hélices que se recomienda usar para estos motores, grandes en comparación con otros modelos de quadrotors, lo que creará una mayor sustentación del mismo.



Ilustración 25: Hélices de 10x4.5".

Los drivers utilizados para la comunicación entre el controlador y los motores son: dos *HW30A 30A Brushless ESC YELLOW* de la marca *Hobbywing* [6], uno de la marca *YGE* [7] modelo 30i y otro de la misma marca, pero modelo 25i.

El hecho de distintas marcas y modelos no supone un problema puesto que conociendo su rango de actuación se adaptan para que puedan trabajar conjuntamente sin notar diferencia. El único problema es la configuración de inicio. Todos están configurados para un inicio rápido, lo cual no es la mejor configuración porque puede haber problemas con la inicialización, en especial con los de la marca *YGE*, aunque no es un problema importante porque una vez se consigue que arranquen todos trabajan perfectamente.

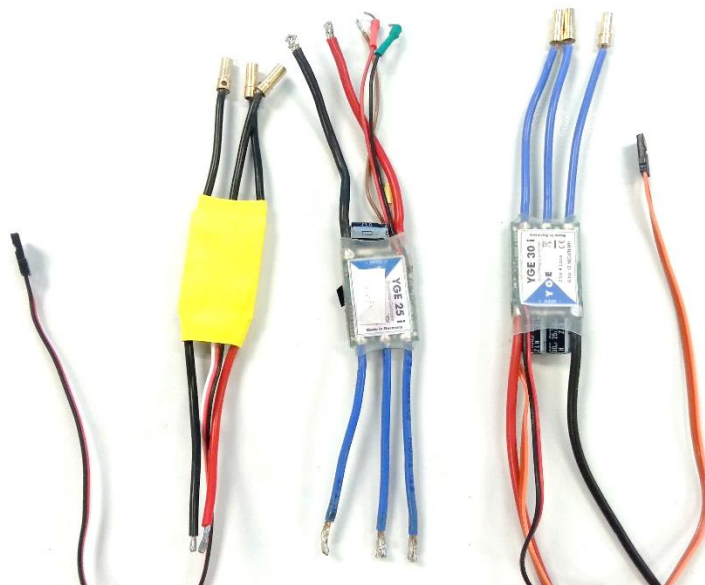


Ilustración 26: Drivers: HW30A 30A Brushless ESC YELLOW, YGE 30i y YGE 25i.

4.4 Otros elementos

Además de los elementos básicos para el Quadrotor mencionados, se necesitan elementos intermedios que se utilizan para la comunicación y alimentación:

- **Raspberry Pi to Arduino Shields Connection Bridge:** de la empresa *Cooking Hacks*. Es un adaptador de puertos (14), el cual es útil en fases de pruebas para proteger los GPIO de la Raspberry de constantes modificaciones de las conexiones.

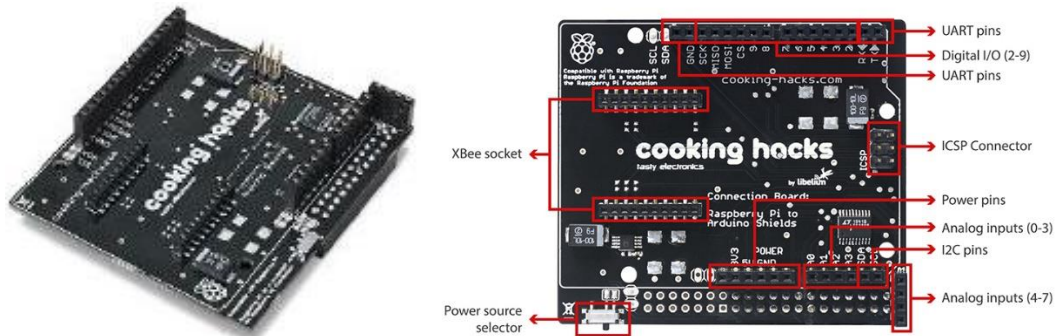


Ilustración 27: Raspberry Pi to Arduino Shields Connection Bridge (14).

Ilustración 28: Especificación de puertos de Raspberry Pi to Arduino Shields Connection Bridge (14).

- **PC:** necesario ya que es desde donde se va a ajustar el control del quadrotor, enviarle las referencias, etc. Ha de trabajar con Ubuntu o Macintosh, ya que el HMI ha sido programado para ser ejecutado en estos SO, y tener conexión inalámbrica.
- **USB wifi o cable Ethernet:** se utiliza para la comunicación entre el PC y la Raspberry. Esta comunicación se puede realizar con ambos elementos, pero en fases de ajuste de control para evitar la existencia de cables colgando que puedan enrollarse con las hélices y crear error en los resultados de control, se realiza con el usb wifi. La comunicación entre ambos aparatos se realiza por SSH (ver ANEXO II).



Ilustración 29: USB wifi Ralink (15).



Ilustración 30: Cable Ethernet.

- **Batería y fuente de alimentación:** para etapas de pruebas se utiliza la fuente de alimentación variable, ya que permite realizar todas las pruebas que se deseen, al contrario que la batería que se descarga a los pocos minutos de uso, pero que es utilizada en fases de ajuste de control.



Ilustración 31: Fuente de alimentación MAAS SPS-9600 (16).

Ilustración 32: Batería Gens ace 5300 mAh (17).

- **Convertidor de tensión:** debido a que se utiliza una misma fuente de alimentación tanto para la Raspberry, que funciona a 5 V, como para los motores y drivers, que funcionan a 12 V, se ha de colocar un convertidor de tensiones para la Raspberry.



Ilustración 33: Conversor de tensión.

5 Comunicación entre elementos

Una cosa imprescindible a la hora de conectar distintos elementos electrónicos es conocer antes el protocolo de comunicación de cada uno.

Cada elemento de el quadrotor tiene un protocolo de comunicación diferente por el que recibe o envía la información:

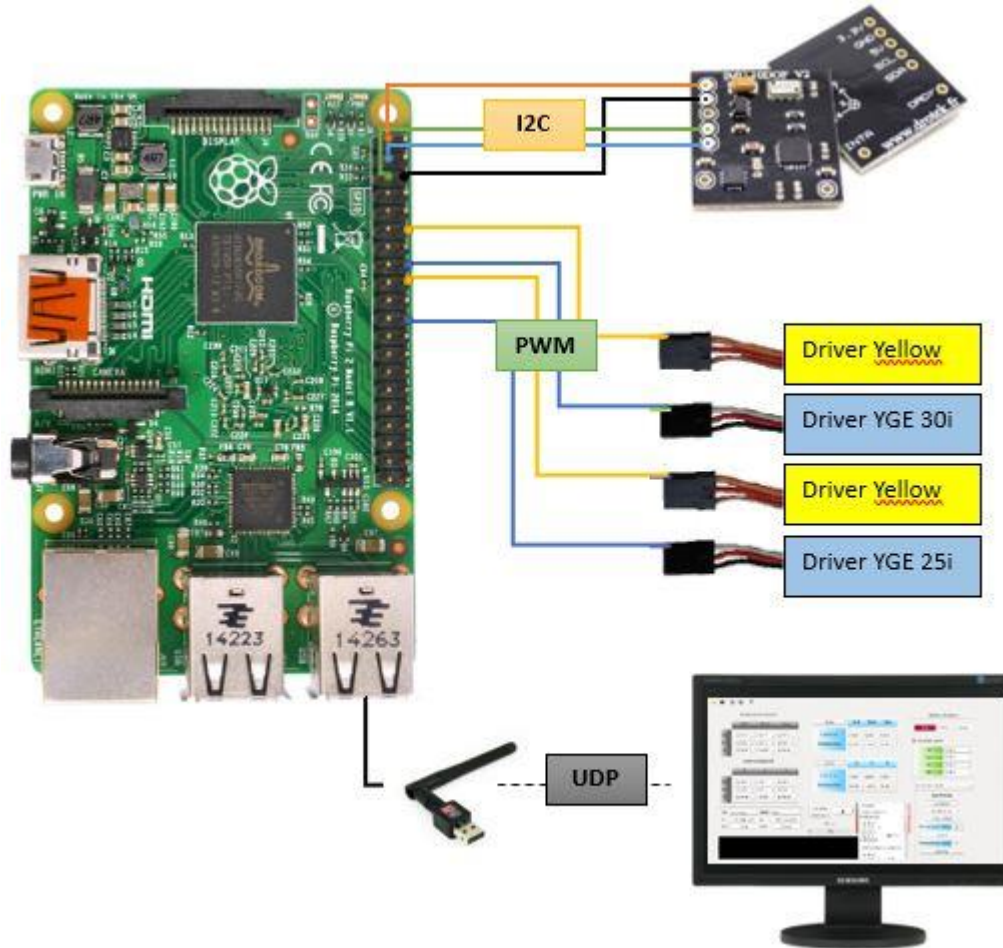


Ilustración 34: Esquema de conexiones.

Raspberry Pi 2	1	3V3	-	3.3V	IMU (I2C)
	3	GPIO2	-	SDA	
	5	GPIO3	-	SCL	
	6	Ground	-	GND	
	12	GPIO 18	-	Yellow	Drivers (PWM)
	16	GPIO 23	-	YGE 30i	
	18	GPIO 24	-	Yellow	
	22	GPIO25	-	YGE 25i	
USB wifi		-	HMI	Ordenador (UDP)	

Tabla 4: Origen y destino de las conexiones.

5.1 Impulsos PWM

Modulación por ancho de pulsos de una señal periódica. Consiste en enviar bits a 1 lógico durante un porcentaje del periodo de ciclo para que el porcentaje medio del tiempo que la señal da el 1 lógico se traduzca en los drivers en el mismo porcentaje en voltaje que se produce en los motores, en nuestro caso el rango de actuación de los GPIO es de 0 a 3,3 Voltios y los drivers le cambian la escala para adecuarlo a la del motor que es de 7 a 12V.

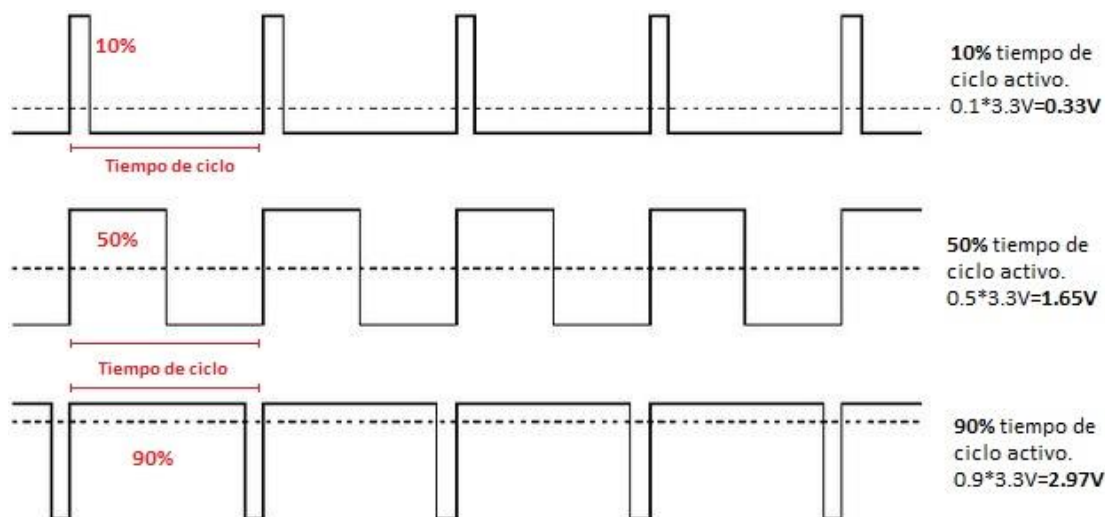


Ilustración 35: Señales PWM.

5.2 Protocolo de comunicación I2C

I²C es el nombre de un protocolo de comunicación serie diseñado por Philips que se utiliza esencialmente entre dispositivos que pertenecen al mismo circuito, por ejemplo, sensores con un microcontrolador. Es un tipo de bus de bajo coste muy utilizado, pero es muy susceptible de interferencias.

Como es nuestro caso. Este es el protocolo que se utiliza la IMU para transmitir la información que obtiene al microcontrolador.

Utiliza el modo maestro-esclavo, donde el maestro inicia la transferencia y el esclavo reacciona. Utiliza 2 líneas: la SCL y SDA. La primera la emite el maestro y contiene el reloj para la transmisión sincrónica, y la segunda es la de transmisión de datos. El maestro establece e inicia la conexión y dependiendo del bit menos significativo le comunica al esclavo si debe recibir o enviar datos. Además de los bits de dirección y el de lectura o escritura está el bit ACK/NACK (acknowledge/ not acknowledge) que es enviado por el esclavo al escribir y desde el maestro al leer (18). En las siguientes imágenes se resume el funcionamiento.

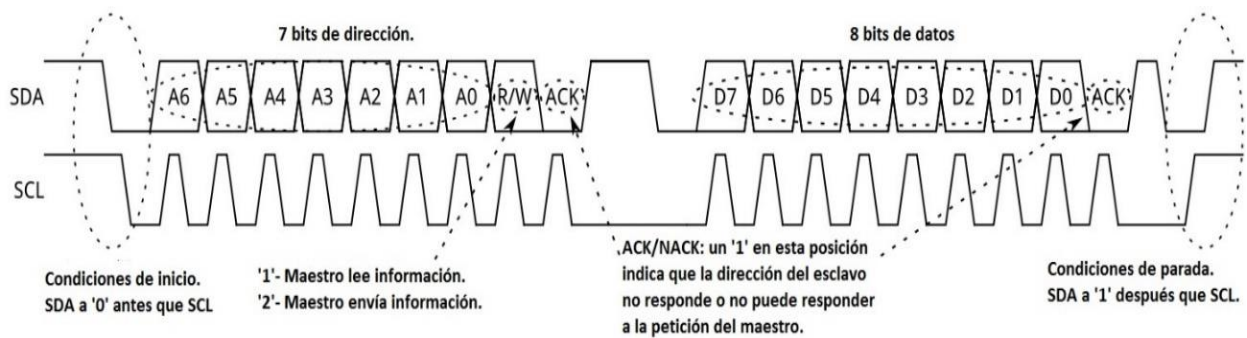


Ilustración 36: Esquema de los paquetes del protocolo de comunicación I2C (18).

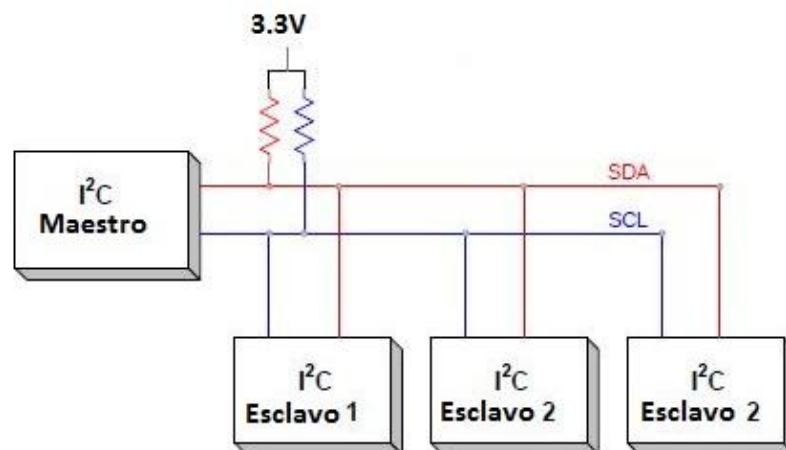


Ilustración 37: Elementos de la conexión I2C (19)

5.3 Protocolo de comunicación UDP (User Datagram Protocol)

Es el protocolo utilizado para la transmisión de información entre el PC y la Raspberry Pi.

Se trata de un protocolo de comunicación basado en el intercambio de datagramas, los cuales contienen ya suficiente información para realizar envío de estos datagramas sin previa conexión con el dispositivo receptor, aunque no tiene confirmación de recibo de los paquetes de datos. Es un protocolo muy utilizado en tareas de control y envío de audio y vídeo por una red.

Es una conexión más "ligera" que la TCP (*Transmission Control Protocol*) porque la TCP es una conexión fiable (asegura la recepción en destino de la información) pero eso tiene un coste en el peso de los datos que se envían (los datos de la conexión UDP son de 8 bytes y los de TCP son de 20), esto hace que el tamaño máximo de datos que se puede enviar en cada paquete sea mayor en el del protocolo UDP (20).

UDP

+	Bits 0 - 15	16 - 31
0	Puerto origen	Puerto destino
2	Longitud del Mensaje	Suma de verificación
64	Datos	

Tabla 5: Esquema de los paquetes del protocolo de comunicación UDP.

TCP

+	Bits 0 - 3	4 - 7	8 - 15	16 - 31
0	Puerto Origen			Puerto Destino
32	Número de Secuencia			
64	Número de Acuse de Recibo (ACK)			
96	longitud cabecera TCP	Reservado	Flags	Ventana
128	Suma de Verificación (Checksum)			Puntero Urgente
160	Opciones + Relleno (opcional)			
224	Datos			

Tabla 6: Esquema de los paquetes del protocolo de comunicación TCP.

6 Sistema Operativo de tiempo real

Es un sistema informático que interacciona continuamente con el entorno físico y reacciona en un tiempo determinado, esto significa que está constantemente leyendo y pidiendo paso a nuevos procesos que requieren ser realizados en un momento determinado, donde no son permisibles los tiempos de espera. Estos sistemas son muy utilizados para procesos de control.

Existen dos tipos principales de SOTR: los *blandos* (SRT, *Soft Real Time*) y los *duros* (HRT, *Hard Real Time*). Los primeros aseguran que la mayoría de veces se cumplen las restricciones de tiempo de latencia y los segundos lo aseguran siempre.

Para controlar las acciones que puede o no realizar en cada momento, los SOTR trabajan con semáforos y prioridades y funciona de la siguiente manera: si una acción con prioridad baja se está ejecutando y otra con prioridad alta necesita ejecutarse en ese mismo momento, en SOTR paraliza la acción con baja prioridad para dejar paso a la de alta prioridad (lo que se conoce como interrupción) en un tiempo determinado (latencia, que en estos sistemas es de microsegundos), cosa que no sucede en los Sistemas Operativos de Propósito General (SOPG), como Windows, los cuales no funcionan con prioridades y dejarían pasar al segundo proceso una vez hubiese acabado el proceso en ejecución. Pero por el contrario, para llegar a estas restricciones de latencia los SOTR requieren mayor memoria RAM que los SOPG y una mayor complejidad a la hora de programar (21).

6.1 Programación en SOTR

Puesto que son sistemas operativos distintos, la programación también cambia en ciertos aspectos (21):

- **Lagunas:** Algo necesario a tener en cuenta al programar en tiempo real es la necesidad de liberar la memoria porque de lo contrario crea *lagunas* en la memoria, que son reservas de memoria que han sido utilizadas pero que siguen activas, aunque no se utilicen.
- **Prioridades:** en los SOPG los procesos se ejecutan por orden de llegada, sin embargo, en SOTR cada proceso ha de tener una prioridad en función de lo necesario que es ese proceso con respecto a los demás, siempre que el semáforo esté desactivado.

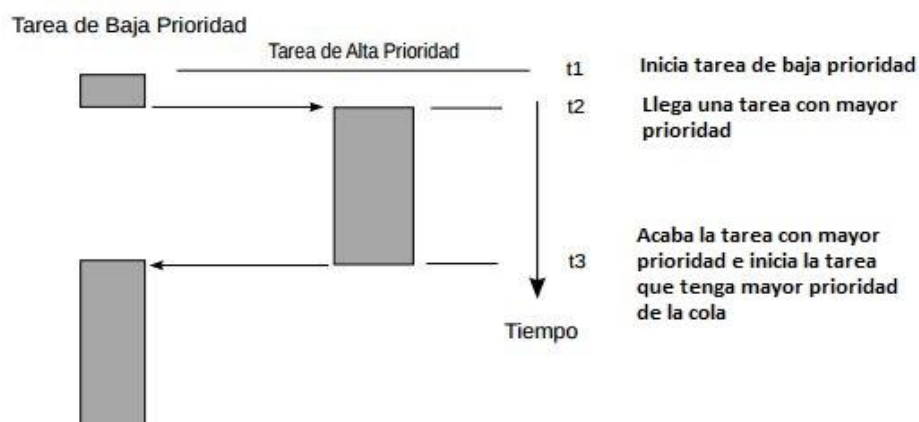


Ilustración 38: Diagrama de comportamiento de tareas con prioridades.

- **El semáforo:** es el encargado de paralizar los procesos que lleguen después del que lo active. Esto es útil, por ejemplo, para evitar que procesos que muestran datos por pantalla se interrumpan entre sí y no se pueda recibir correctamente el mensaje de cada uno. Una vez el proceso que ha activado el semáforo lo desactiva, los demás procesos pueden iniciarse por orden de prioridad.

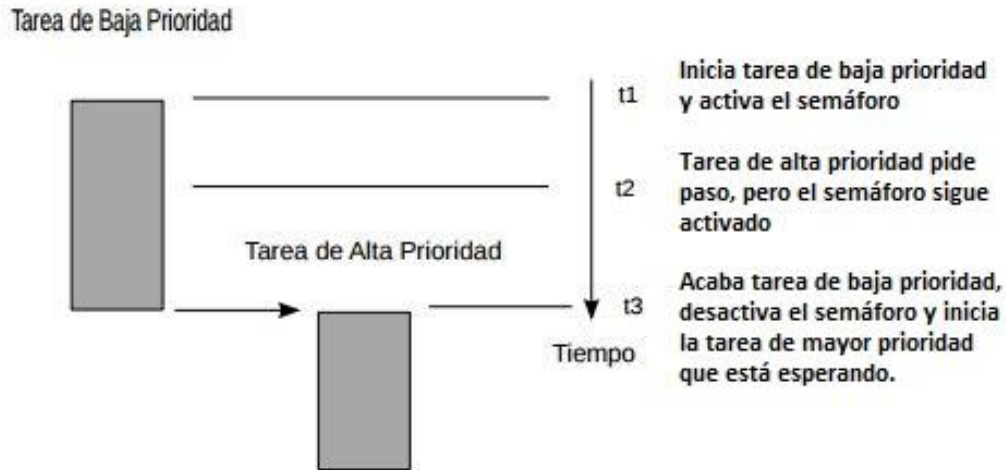


Ilustración 39: Diagrama de comportamiento de tareas con prioridades y semáforo.

6.2 El SOTR en el quadrotor.

El quadrotor es un aparato que necesita estar constantemente recibiendo información del exterior y generando reacciones instantáneas. El control para la estabilización del quadrotor necesita modificar la acción de los motores continuamente y un retraso supondría la posible pérdida del control del mismo.

Existen SOTR como VxWorks, QNX, MaRTE OS, entre otros, pero se ha preferido continuar con Raspbian por la inmensa comunidad de usuarios que tiene e instalarle un parche de tiempo real basado en Linux: *Preempt_RT*.

6.2.1 Raspbian con parche Preempt_RT

Como se ha comentado anteriormente, el sistema operativo instalado en la Raspberry es Raspbian, que es un Sistema Operativo de Propósito General, por ello se necesita un parche que actúe en el kernel que cambie la forma de trabajar del planificador de procesos. Este parche es el *Preempt_RT*. Este parche ofrece un rendimiento equivalente o superior a otros SOTR dedicados, gracias a un uso menor de la CPU velocidades más altas del bucle.

Se trata de un SOTR basado en Linux, lo cual permite tanto un programador de procesos en tiempo real para tareas críticas como uno más eficiente y completo para tareas que no necesitan ser realizadas en tiempo real (22).

Se puede observar la latencia en microsegundos que tiene este sistema una vez instalado escribiendo en la línea de comandos lo siguiente: `sudo cyclicttest -l1000000 -m -n -a0 -t1 -p99 -i400 -h400 -q` (23).


```
# Total: 000007244  
# Min Latencias: 00011  
# Avg Latencias: 00013  
# Max Latencias: 00034  
# Histogram Overflows: 00000  
# Histogram Overflow at cycle number:  
# Thread 0:
```

Ilustración 40: Resultados de latencia del sistema operativo de tiempo real.

7 Software

El programa se ha programado en C++ orientado a objetos con el IDE Code::Blocks (24) en Ubuntu. Este programa se encarga de la recogida y procesamiento de datos de los sensores y del accionamiento y control del quadrotor. Todos estos procesos están comunicados para buscar que el quadrotor sea autónomo a partir de unas referencias que se le envíen.

Se ha descompuesto en múltiples hilos, para su mayor facilidad a la hora de detectar errores y facilidad para añadir nuevos sensores o actuadores, y un programa principal que los ejecuta. Si bien, estos hilos han de estar conectados entre ellos puesto que cada uno depende de los demás. Estos hilos son: control, accionamiento de los motores, lectura y procesamiento de datos de la IMU y comunicación UDP. Cada hilo tiene su función y se ha realizado por separado comprobando su funcionamiento para posteriormente juntarlos todos en un mismo programa.

Puesto que Raspberry Pi 2 no utiliza Ubuntu como sistema operativo, se ha necesitado la instalación de un compilador cruzado (compilador gnuabi arm para Linux (25), ver ANEXO IV). Gracias a este compilador cruzado se podrán crear ejecutables legibles por Raspberry y que se le enviarán para ser ejecutados desde allí.

7.1 Accionamiento de los motores.

Para realizar este hilo se ha utilizado la librería PIGPIO. Gracias a esta librería se pueden generar los pulsos PWM (ver Impulsos PWM) a través de los GPIO necesarios para el movimiento de los motores.

Para comprender mejor el funcionamiento de los motores se ha implementado un programa que inicie los motores y les vaya aumentando la velocidad para conocer los rangos de actuación, porque, como se ha explicado, se utilizan distintos drivers, los cuales se activan con distinto valor de impulsos PWM y no tienen el mismo rango. En el programa, el accionamiento de los motores necesita de las librerías PIGPIO (ver ANEXO V).

En el programa principal este hilo se ha ejecutado con una frecuencia de 500 y una prioridad menor que el hilo de control.

7.2 Lectura y procesamiento de datos de la IMU

Este hilo se encarga de inicializar la IMU, recoger sus lecturas y procesarlas mediante el Filtro de Kalman, que se encarga de traducir los valores de las medidas de giróscopos y acelerómetros de modo que descarte medidas erróneas y las correctas poder ser procesadas para conseguir la medida de los 3 ángulos.

Las lecturas de la IMU se realizan a una frecuencia de 333Hz y una prioridad mayor que todas. El filtro de Kalman se ejecuta a 300Hz (menor que la de lectura de la IMU para evitar que el filtro traduzca valores constantes) pero a una prioridad mayor que las demás, pero menor que la de la IMU porque traduce la información necesaria para el resto de hilos.

Al igual que en el hilo de accionamiento se lleva a cabo un programa para comprobar los valores de medida de la IMU mediante las librerías: *ComponentSensors.cc*, *ComponentSensors.h*, *i2cDevice.cc*, *i2cDevice.h*, *i2c-dev.h*, *ImuMPU.cc*, *ImuMPU.h*, *KalmanDesacoplado.cc*, *KalmanDesacoplado.h*, *MatrixBasicLibrary.cc*, *MatrixBasicLibrary.h*, *MatrixMath.cc*, *MatrixMath.h*, *Sensors.h* (ver ANEXO VI).

7.3 Control

Este hilo se ejecuta desde el PC mediante el HMI. Este hilo permite la elección del modo de trabajo del quadrotor: modo automático, manual o Stop.

- **Modo automático:** cuando se activa este modo, el quadrotor se estabiliza en el ángulo que se le especifique en el HMI gracias al PID aplicado. Este modo es el que se utiliza para realizar los ajustes del PID y se escribe el fichero necesario para graficar los resultados del ajuste (ver ANEXO VII).
- **Modo Manual:** este modo únicamente es capaz de modificar la velocidad de giro de cada motor. Este modo se utiliza para iniciar el movimiento de los motores y evitar imprevistos causados por la diferencia de posición inicial del quadrotor y las referencias implementadas en el HMI.
- **Stop control:** este modo para los motores, guarda el archivo de resultados y espera parado hasta nueva orden.

La prioridad de este hilo ha de ser mayor que la del hilo de accionamiento y se ha ejecutado a una frecuencia de 333Hz.

7.4 Comunicación UDP

Este hilo se encarga de establecer correctamente la conexión entre la Raspberry y el PC para su posible transmisión de lecturas y medidas de control a través del HMI.

Este hilo se ejecuta a 50Hz y tiene la prioridad más baja de todas.

7.4.1 HMI

Esta interfaz ha sido creada por Natalia Díaz-Otero Zafrilla y se ha utilizado para este proyecto (Ilustración 41: HMI..

Se trata de una interfaz que permite la visualización de datos enviados por el quadrotor, así como el envío de referencias o parámetros de control al mismo.

Ha resultado muy útil a la hora de ajustar los PID para el control ya que permite la modificación de las constantes de los PID para el control en pleno vuelo, así como recibir los desplazamientos del quadrotor en todos sus ejes (aunque esta última parte no corresponde al alcance de este trabajo) y sus giros. También permite la elección del modo de funcionamiento y la variación de la potencia de los motores (26).

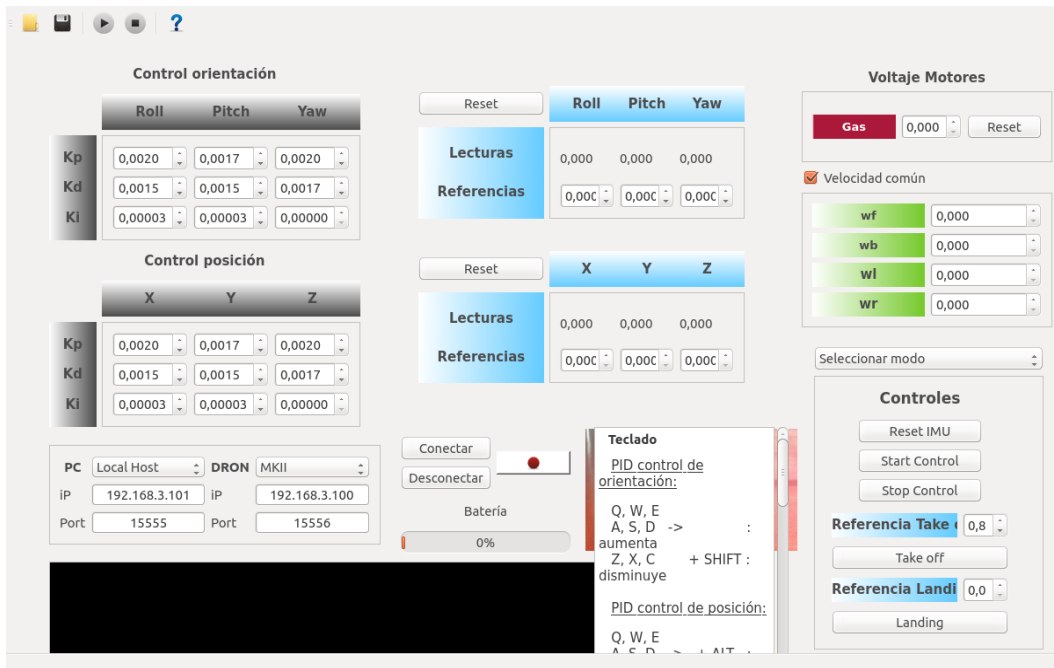


Ilustración 41: HMI.

8 Banco de pruebas

Se ha construido una base con el fin de poder comprobar resultados de los algoritmos de control. Además, esta base puede ser utilizada para otros posibles trabajos futuros puesto que no depende de las dimensiones del quadrotor.

La estructura está basada en una rótula como el elemento que permite los giros de los ángulos Pitch, Roll y Yaw. Este elemento imita de forma aproximada los movimientos que permite la plataforma *Hover de Quanser* para el ajuste de algoritmos de control.



Ilustración 42: Plataforma Hover de Quanser (27)

Esta plataforma sirve para investigación y enseñanza. Se utiliza para implementar sistemas de control, ya que con su infinidad de sensores puedes obtener gran cantidad de información útil para su posterior aplicación en quadrotors (27).

8.1 Montaje

Elementos básicos:

- Base de madera de 237 x 400 x 15 mm
- Mástil de madera de 55 x 55 x 375 mm
- 4 escuadras de metal con 4 agujeros
- 16 tornillos para madera

Elementos específicos del montaje 1:

- Rótula 1 (Sistema completo de cojinete de fricción con soporte más el perno esférico con rosca macho de 8 mm)
- Tuerca Tee Nut de 8 mm
- 4 tornillos de 4 mm
- 4 tuercas de 4 mm

Elementos específicos del montaje 2:

- Rótula 2 (Mini ball head hot shoe adapter)
- Placa fina de metal de 50 x 50 mm
- 4 tornillos para madera.

El hecho de que haya 2 montajes tiene una explicación. En un principio se probó con el montaje 1 pero a la hora de su uso prolongado no funcionó bien puesto que el elemento principal, que es la rótula (Ilustración 43), resultó no ser apta para este tipo de uso. La rótula funcionaba bien los primeros ensayos, pero conforme se hacían ensayos iba perdiendo la facilidad de movimiento, lo cual obligaba a ajustar el control cada vez que se ensayaba. A pesar de ello se ha decidido mencionarlo ya que el montaje 2 no obliga a desmontar el 1, por tanto, sigue montado para posibles montajes futuros que precisen de esa técnica. Por tanto se buscó otra solución para conseguir una mayor fiabilidad en los ensayos y esta fue sustituida por la de la Ilustración 44

Los requisitos principales a la hora de buscar un rótula eran el peso que era capaz de soportar y el ángulo de giro que permite. La rótula 1 es capaz de soportar pesos de mas de 10 veces el peso del quadrotor y permite un giro entre 34° y -34° desde la vertical (28), lo que es suficiente para la comprobación de resultados, pero resultó que con el uso perdía la facilidad de rotación y se endurecía. La rótula 2 es de cámara de fotos pero ha resultado ser apta para el uso que se le da en este trabajo. Esta es capaz de soportar varias veces el peso del quadrotor y ángulos de 28 grados desde la vertical (29).



Ilustración 43: Rótula 1 (Sistema completo de cojinete de fricción con soporte más el perno esférico con rosca macho) (28).

Ilustración 44: Rótula 2 (Mini ball head hot shoe adapter) (29).

Una mejora de la rótula 2 con respecto a la anterior es que, además de que su uso prolongado es mejor, permite el bloqueo del movimiento, lo cual es cómodo en fases de pruebas y ayuda a la durabilidad de la rótula evitando que descansa el peso del aparato en el ángulo máximo.

Las dimensiones de la tabla y del mástil son suficientes para que los movimientos que realiza el quadrotor no vuelquen la plataforma, ya que el testeo se realiza a partir de que el quadrotor se mantenga estable en ángulos que no superan los 20 grados desde la vertical, y que las hélices del mismo no toquen el suelo en puntos de máxima inclinación de la rótula.

Los elementos básicos se montan de manera sencilla: el mástil se sitúa en el centro de la base de madera sujetado por las escuadras y tornillos.

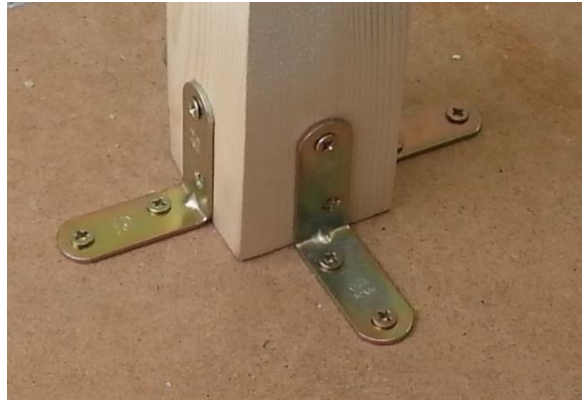


Ilustración 45: Montaje de la base y el mástil

Para el montaje 1 al mástil se le han limado las aristas de la cara superior, para evitar que el quadrotor roce en puntos de máximo ángulo de la rótula 1, y se le ha realizado un agujero para introducir la tuerca Tee Nut junto con un poco de pegamento, ya que este tipo de tuercas sirven para bloquear los esfuerzos a torsión pero el quadrotor provoca un esfuerzo de tracción que se ha de tener en cuenta.



Ilustración 46: Montaje 1.



Ilustración 47: Montaje 1 con quadrotor.

El montaje 2 permite mantener la tuerca Tee Nut del montaje 1. En este montaje se coloca la base enroscable de la rótula 2 sobre la tuerca y encima se le coloca la placa de metal atornillada al mástil.



Ilustración 48: Placa metálica para el montaje 2.

Ilustración 49: Montaje 2 con quadrotor

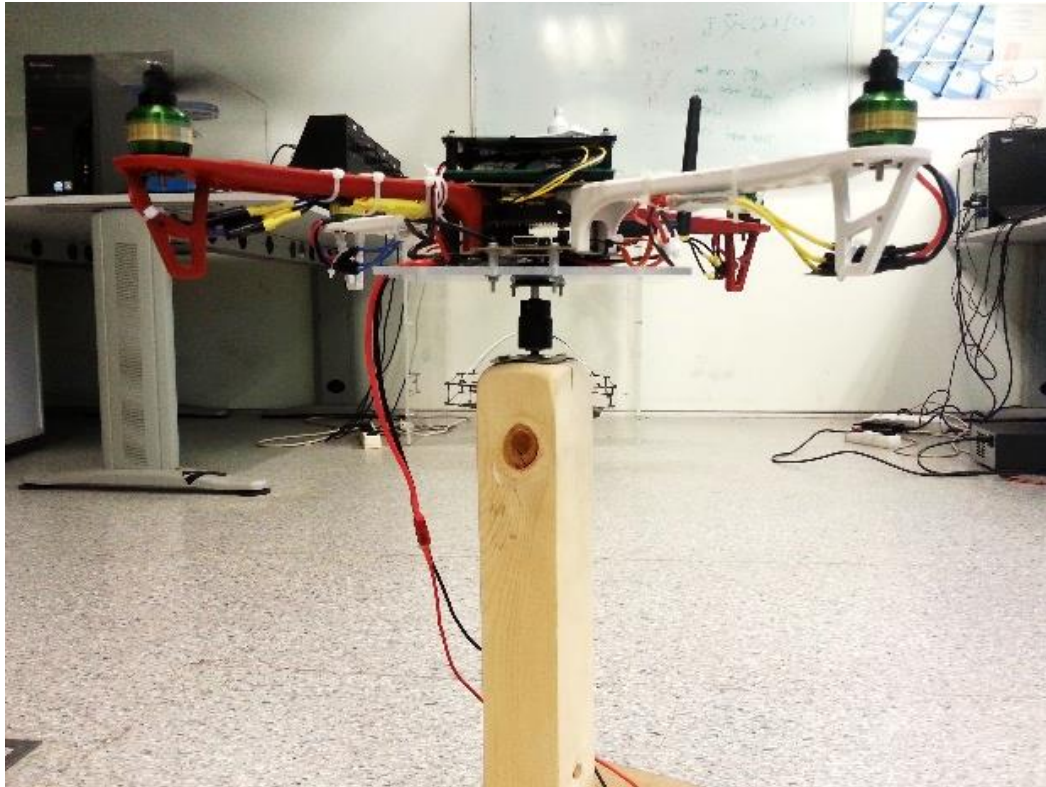


Ilustración 50: Rótula 2 enganchada a la base.

Ilustración 51: Montaje 2 con quadrotor.

9 Sistema de control

El control es una de los principales objetivos de este trabajo, y de los más experimentales. Se requiere un control bien ajustado si se quiere tener un quadrotor dispuesto a hacer lo que se le ordene.

Según los objetivos que se quieran obtener con el modelado se hará más incapié en un sistema de control distinto (30)

- Sistema de aumento de estabilidad (SAS, Stability Augmentation Systems): Su función es amortiguar las sacudidas del quadrotor con independencia de su trayectoria o actitud de vuelo.
- Sistema de aumento del control (CAS, Control Augmentation Systems): La función de este tipo de sistema de control es asegurar una respuesta en cualquier momento además de asegurar el SAS.
- Sistemas de piloto automático (Autopilots): Sistemas de control superior que son capaces de realizar por si solos ciertas acciones como despegue, vuelo estacionario...

El principal objetivo en esta parte es conseguir la estabilidad para posteriormente ajustar el control y reducir el error, por tanto, se busca obtener un CAS.

9.1 Control de orientación

Como se ha comentado anteriormente, el objetivo de este trabajo es llegar a controlar la orientación del quadrotor en los 3 ángulos Roll, Pitch y Yaw (31).

Para el control de la orientación se implementa un PID (P, PD, PI o PID) con el esquema que se muestra.

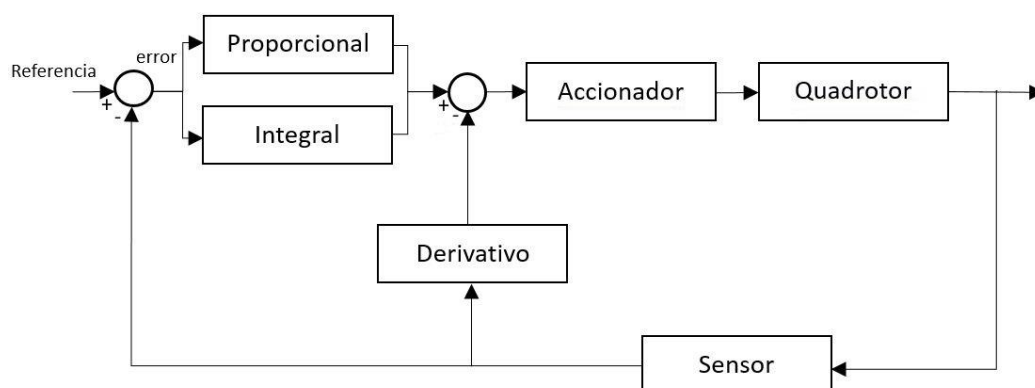


Ilustración 52: Esquema de control.

Se trata de un PID modificado, que se puede modificar en pleno vuelo gracias al HMI hasta conseguir los resultados más parecidos a los deseados (error cero, respuesta rápida ante

escalones y perturbaciones y sin oscilaciones). Las partes proporcional e integral actúan directamente sobre el error, pero la parte derivativa actúa a partir de las medidas que obtiene la IMU.

$$e = R - m \quad \text{E.7}$$

$$P = K_p \cdot e \quad \text{E.8}$$

$$D = K_d \cdot \dot{m} \quad \text{E.9}$$

$$I = I_{ant} + K_i \cdot e \quad \text{E.10}$$

$$PID = P + I - D \quad \text{E.11}$$

Donde e es el error, R es la referencia aplicada, m es la medida en grados que obtiene la IMU, \dot{m} es la velocidad angular en grados / segundo, P es la acción proporcional, D es la acción derivativa, I es la acción integral.

El PID en el programa se implementa en el archivo `x3d.cc` como se explica en el ANEXO VII.

9.1.1 Ajuste experimental de los parámetros de control.

Debido a las numerosas variables que intervienen en el control su realización teórica es demasiado elaborada, por tanto, el ajuste del control de los diferentes ángulos se ha llevado de manera experimental.

Las pruebas del control se realizan con el quadrotor colocado en la base y bien sujeto porque a la hora de ajustar el control puede haber oscilaciones o movimientos no esperados que pueden dañar a quien lo esté testeando.

Debido a las particularidades de la rótula finalmente instalada el quadrotor queda más “libre”, lo cual a la hora de ajustar el control puede complicar el proceso.

Las pruebas de cada uno de los tipos de controladores (P, PI, PD, PID) se ha realizado permitiendo al quadrotor pequeños movimientos para poder controlar las posibles oscilaciones, creando variaciones en las referencias de menos de 5 grados.

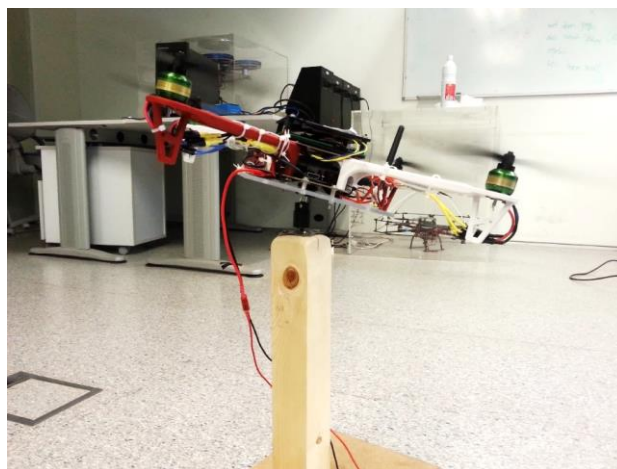


Ilustración 53: Quadrotor durante el ajuste del control.

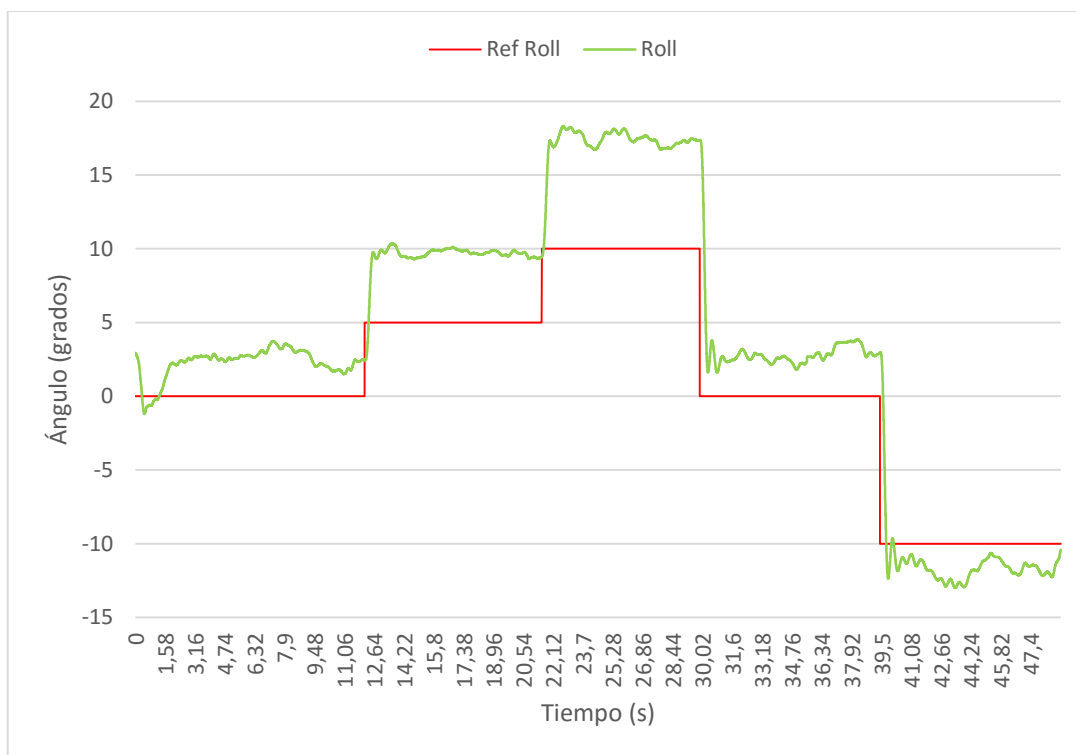
9.1.1.1 Ajuste del Roll y Pitch.

Gracias a que la estructura del quadrotor es simétrica el ajuste de estos ángulos se puede realizar simultáneamente ya que ambos controladores van a ser iguales o prácticamente iguales.

Se ha escogido ajustar primero estos ángulos porque el Yaw depende más de estos ángulos, ya que si el quadrotor no está estabilizado no se puede realizar un correcto ajuste de este ángulo.

Para comenzar se ha probado con un proporcional (P) de distintos valores, pero en ningún momento se ha conseguido la estabilidad del quadrotor.

Con el fin de eliminar la inestabilidad se prueba con acción derivada para amortiguar la acción de control. Con el PD se consigue la estabilidad del quadrotor, aunque con un gran error respecto a las referencias.



Gráfica 1: Ajuste del Roll con un PD.

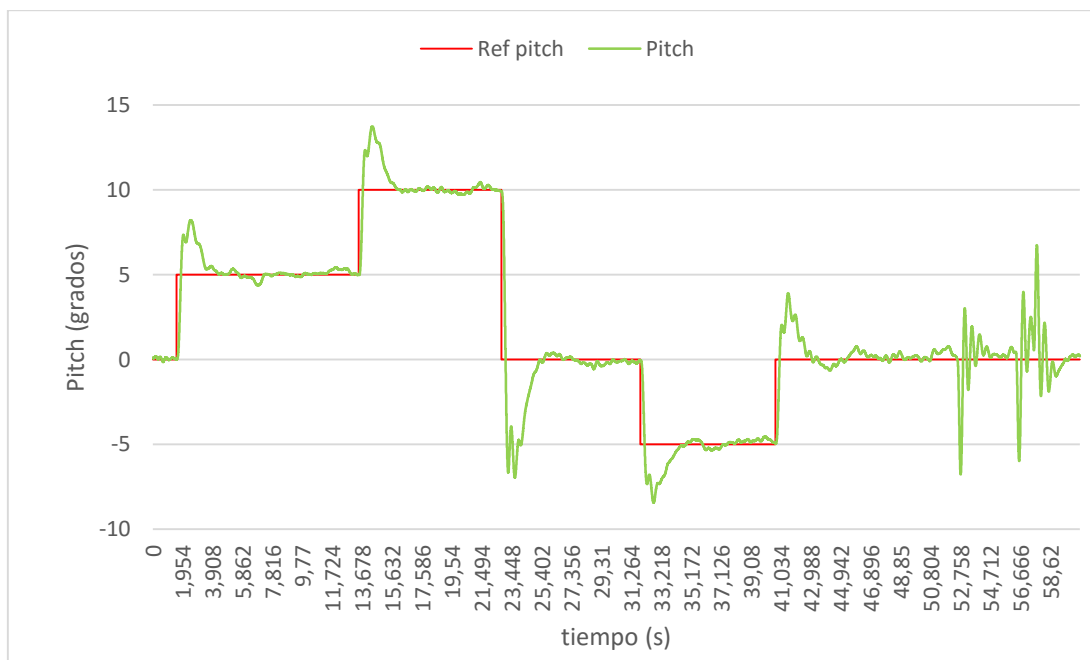
Este controlador responde bastante bien a las referencias, aunque tiene bastante error, el cual se reducirá colocando acción integral. A pesar de ello, este controlador no se vuelve inestable, como sucedía con el proporcional.

El proporcional con acción integral (PI) no se prueba porque la acción integral desestabiliza los sistemas y el proporcional de por sí ya es inestable.

Por último, se prueba con un PID y tras varias modificaciones de los parámetros se obtienen estos resultados.



Gráfica 2: Ajuste del Roll con un PID.



Gráfica 3: Ajuste del Pitch con un PID.

Como se observa se ha reducido o eliminado el error existente en el PD gracias a la acción integral.

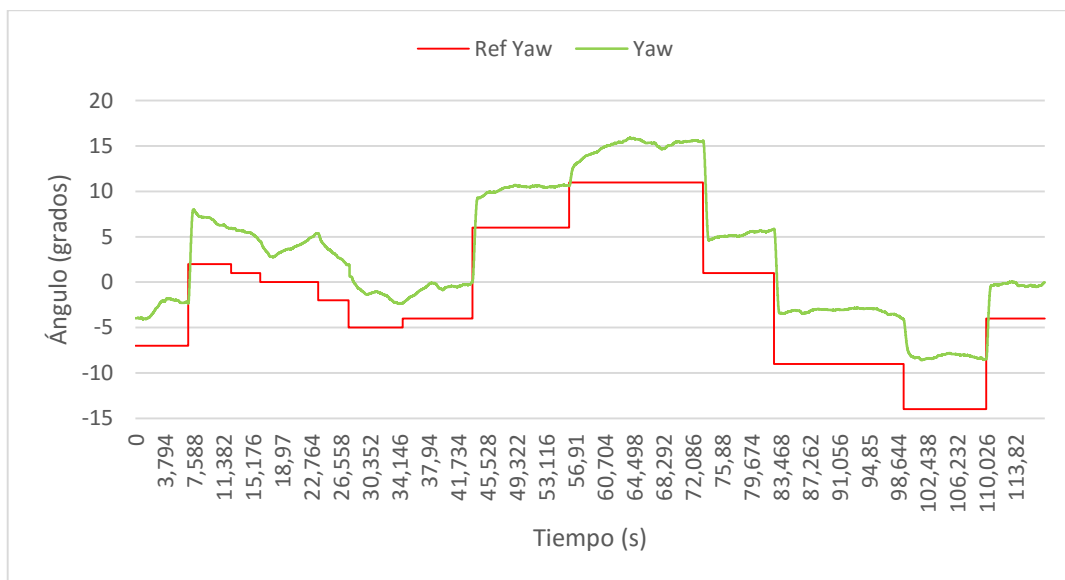
Al final de las gráficas se observan unas lecturas de los ángulos que corresponden a perturbaciones que se le han aplicado para comprobar la respuesta del sistema ante acciones exteriores.

9.1.1.2 Ajuste del Yaw

Una vez se ha conseguido la estabilidad del quadrotor con respecto a la horizontal se ha de ajustar el control de la posición de giro con respecto al eje Z.

La medida de este ángulo contiene más error porque solo se utiliza el giróscopo de la IMU para obtenerla, sin embargo, para los ángulos Pitch y Roll se utilizan, además, los acelerómetros. Este error hace que el quadrotor gire a velocidad muy pequeña sobre sí mismo en sentido horario, lo cual es grave si se quiere mantener la posición de forma prolongada. A pesar de todo este error no afecta al testeo de resultados porque la velocidad de giro que provoca el error es muy pequeña comparada con la velocidad de giro del quadrotor por cambios de la referencia.

Como anteriormente se comienza con un controlador proporcional obteniendo resultados bastante buenos, es decir, responde bien a las referencias, aunque con algo de error, el cual se elimina con el integrador.



Gráfica 4: Ajuste del Yaw con un P.

Para suavizar la acción de control se prueba, como con el Pitch y el roll, con un PD pero con resultados contrarios. Este controlador vuelve inestable el sistema ante referencias o perturbaciones. Esto se debe a que la lectura del sensor percibe mucho ruido y hace que la acción derivada obtenga valores muy distintos en cada instante.

Con el fin de eliminar el ruido y posibles perturbaciones que perjudiquen el control se implementa un filtro de paso bajo obteniendo una señal filtrada buena, pero con un retraso mínimo que en este contexto de sistemas de tiempo real resulta ser más perjudicial que recibir una señal con ruido. Por tanto, el siguiente controlador para probar es el PI.

Este controlador consigue eliminar el error a cambio de aumentar ligeramente las oscilaciones hasta alcanzar el régimen estacionario, lo cual no es un problema importante ya que es un giro que no afecta a la estabilidad, lo cual es el objetivo principal.



Gráfica 5: Ajuste del Yaw con un PI.

Como en el ajuste del Pitch y Roll, al final de las gráficas se observan unas lecturas de los ángulos que corresponden a perturbaciones que se le han aplicado para comprobar la respuesta del sistema ante acciones exteriores.

Los valores finales que se han implementado para el control de los 3 ángulos simultáneamente son:

$U=0.3$	Roll	Pitch	Yaw
k_p	0.0070	0.0070	0.0170
k_d	0.0019	0.0019	0.0000
k_i	0.0001	0.0001	0.0001

Tabla 7: Valores de los parámetros del PID.

Donde u es la carga de los motores (30% del máximo), k_p , k_d y k_i son las ganancias de la parte proporcional, derivativa e integral respectivamente para cada uno de los ángulos.

A pesar de que los resultados son buenos, las vibraciones que transmite la base, la proximidad del suelo (creando el “ejecto suelo”, el cual hace que el propio aire impulsado por las hélices rebote en el suelo creando perturbaciones), las medidas falsas que se reciben por el ruido creado por los motores, etc, hacen que sea complicado obtener un ajuste más preciso.

10 CONCLUSIONES

Durante el recorrido de este trabajo se ha validado la Raspberry Pi 2 para su uso en este ámbito, es más, ha resultado ser una plataforma autosuficiente para comunicarse con los diversos elementos del quadrotor cumpliendo, además, los requisitos de sistema de tiempo real.

Gracias a la inmensa cantidad de información que hay sobre la Raspberry Pi 2 los problemas que se presentan se solucionan con gran rapidez, como son el proceso para obtener lecturas de la IMU, el cual requiere de conocimientos más profundos sobre Raspberry, o la utilización de los impulsos PWM.

Para conseguir el objetivo de la altura del quadrotor se colocó la raspberry junto con la placa Cooking Hacks en el hueco que hay dentro de la estructura del quadrotor de forma que los puertos necesarios como son los usb, el de alimentación, HDMI y los GPIO, queden accesibles, aunque de esta forma se dificulta algo el acceso a ciertos pins, dejando fuera de la estructura únicamente la IMU a la altura de las hélices para obtener mejores lecturas. Con esta configuración del montaje se utiliza el hueco que se utilizaba para poner la batería, por lo que esta configuración es válida para el testeado sobre plataforma, por no para quadrotors que vayan a tener vuelo libre.

La validación del banco de pruebas requirió la prueba de dos montajes, ya que con el primero durante el ajuste de control se obtenían valores que no concordaban para los mismos valores de control en distintos días, debido al progresivo deterioro de la rótula. Con el cambio de rótula se hubo de ajustar de nuevo el control, que al tener mejor movimiento resultó ser más costoso. Pero, a pesar del buen funcionamiento de la rótula, el banco tiene dos inconvenientes: el mástil es muy bajo y para alta carga de motores el “efecto suelo” (perturbaciones que se producen por la proximidad del suelo, debido a que el aire desplazado por las hélices al chocar contra el suelo rebota provocando perturbaciones) llega a perturbar de forma importante al quadrotor, y el segundo inconveniente es que debido a que la base es rígida, esta transmite todas las vibraciones, las cuales perturban al sensor y producen resultados que en vuelo libre no se observarían. Por tanto, el banco de pruebas resulta útil para comprobar algoritmos de control de manera aproximada y sin grandes exigencias de precisión.

11 FUTUROS TRABAJOS

Existen infinidad de posibles trabajos futuros gracias a las prestaciones de la Raspberry Pi 2, pero como siempre, la versión mejorada de algo bueno es mejor todavía, así que una posible opción es la adaptación del programa utilizado a la Raspberry Pi 3, la cual tiene mejores prestaciones que mejorarían el tiempo de procesamiento de las funciones.

En el sistema de control se podría mejorar el ajuste, buscando una buena relación entre una respuesta rápida sin sacudidas y el error mínimo modificando el PID implementado o buscar otras estrategias de control que sustituyan al PID.

El programa se podría mejorar con la implementación del algoritmo para el control de altura y de posición. Además, se puede implementar desde sistemas de piloto automático de manera que tenga cierta secuencia de acciones a realizar como el despegue, hasta sistemas de seguimiento.

El banco de pruebas se puede mejorar con un sistema de rodamientos más parecido al de Hover de Quanser con su eje de rotación de todos los ángulos más cerca al centro geométrico del quadrotor. Con dicho sistema, además, se podría ajustar el control de manera individual bloqueando el resto de giros obteniendo un mejor ajuste. Otra mejora sería idear algún sistema de guías que permitiera al quadrotor cierto desplazamiento en los ejes X-Y.

El quadrotor como tal se podría mejorar cambiando los drivers para que sean todos iguales y cambiando su modo de inicio, ya que su actual modo tiene problemas a la hora de la inicialización, aunque una vez iniciados no tienen problemas.

Creación de una estructura de seguridad para evitar accidentes y tener la posibilidad de hacerlo volar en proyectos que lo requieran.

ANEXO I. Instalación del sistema operativo Raspbian

Requisitos:

- Tarjeta SD mini de 8G.
- Archivo: emlid-raspberrypi2-raspbian-rt-20150401.img
- Monitor con salida HDMI.
- Cable HDMI.
- Teclado usb.
- Cable usb mini.
- Raspberry Pi 2

1.- Crear la imagen del Sistema en la tarjeta SD mini.

Para ello se ha utilizado el programa Win 32 Disk imager.

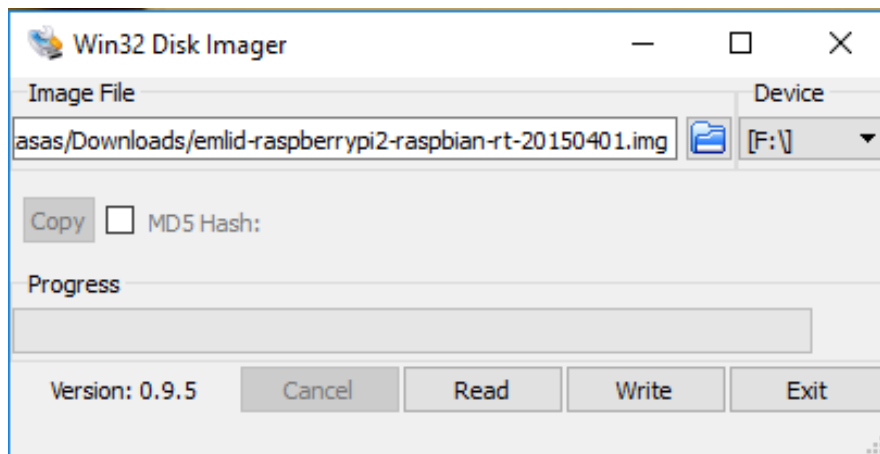


Ilustración 54: Grabar imagen del sistema con Win32DiskImager.

Se especifica la ubicación del archivo .img en el ordenador, en *Device* se escoge la tarjeta SD mini y se presiona *write*.

2.-Iniciar el SO en Raspberry.

Para iniciarlo, primero hemos de conectar el teclado (puerto USB) y el monitor a la Raspberry (puerto HDMI)

A continuación, introducimos la tarjeta SD mini en la Raspberry Pi 2 y la conectamos a la corriente mediante el cable usb mini.

Por el monitor vemos como el sistema se va instalando y al acabar nos pedirá usuario (*pi*) y contraseña (*raspberrypi*). Una vez lo introducimos ya se tiene el sistema instalado e iniciado.

***Consejo:** continuar seguidamente con el anexo 2 para no requerir más del monitor ni del teclado usb y trabajar a través del PC.

ANEXO II. Establecimiento de comunicación SSH entre PC y Raspberry Pi 2.

En este anexo se explica cómo habilitar la conexión ssh entre el PC y la Raspberry. Esta comunicación es necesaria para el traspaso de los archivos entre ambos dispositivos y no depender de un monitor y teclado externos.

Se van a establecer dos métodos de conexión ssh: por wifi y por Ethernet. Para ambos casos se ha de configurar tanto la Raspberry como el ordenador.

Requisitos:

- Monitor con salida HDMI.
- Teclado usb.
- Cable de alimentación usb mini (como el de los móviles Android).
- Usb wifi y/o cable ethernet.
- Raspberry Pi 2.
- Tarjeta SD mini con el S.O. instalado.

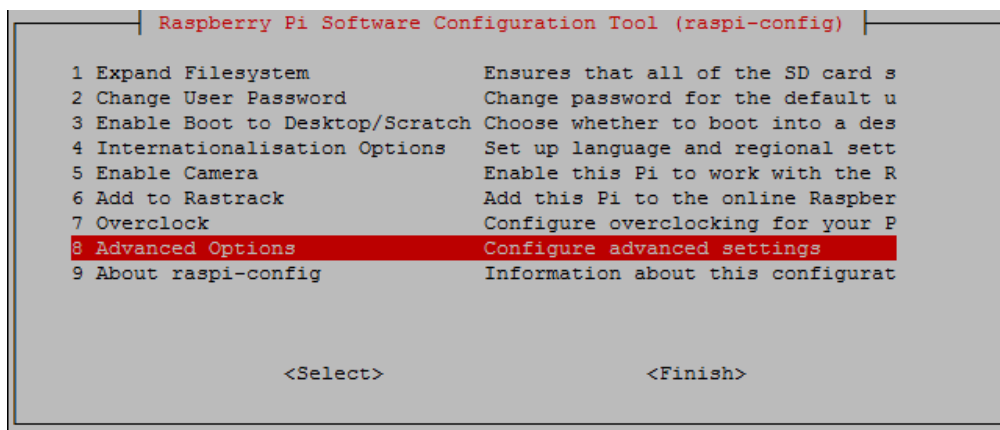
1.- Habilitar la conexión SSH de la Raspberry

Conectar el teclado a un puerto usb de la Raspberry, el monitor con la raspberry mediante el HDMI y la raspberry a la corriente con el cable usb mini. Una vez cargado el sistema se requiere de usuario y contraseña:

User: pi

Password: raspberry

Para habilitar la conexión se ha de acceder al menú de configuración de Raspberry escribiendo: `sudo raspi-config`, y mediante el teclado se selecciona *Advanced Options* -> *ssh* -> *Enable*



```
Raspberry Pi Software Configuration Tool (raspi-config)

1 Expand Filesystem           Ensures that all of the SD card s
2 Change User Password        Change password for the default u
3 Enable Boot to Desktop/Scratch Choose whether to boot into a des
4 Internationalisation Options Set up language and regional sett
5 Enable Camera                Enable this Pi to work with the R
6 Add to Rastrack              Add this Pi to the online Raspber
7 Overclock                    Configure overclocking for your P
8 Advanced Options             Configure advanced settings
9 About raspi-config           Information about this configurat

                                <Select>                                <Finish>
```

Ilustración 55: Raspi-config.

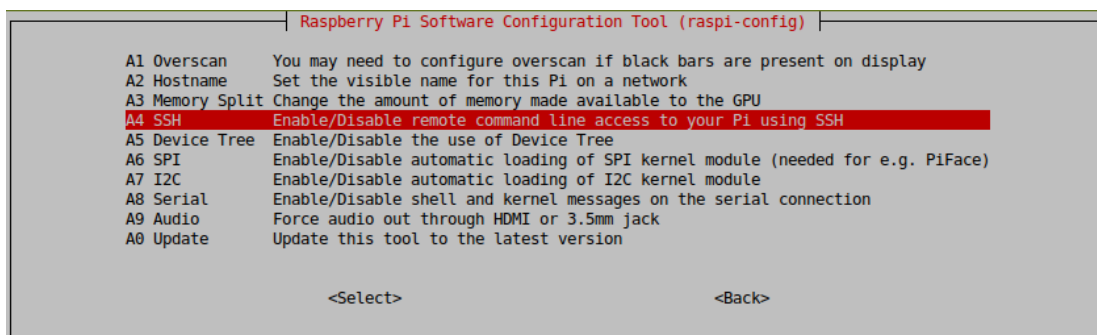


Ilustración 56: Raspi-config habilitar ssh.

2.- establecer IP estática.

El archivo *interfaces* establece las direcciones IP que adquiere la Raspberry según se conecte de una u otra manera a otro dispositivo. Para acceder a dicho archivo se escribe en la línea de comandos de Raspberry: `sudo nano /etc/network/interfaces`.

Y se ha de sustituir el contenido por:

```
GNU nano 2.2.6 File: /etc/network/interfaces
# The loopback interface
  auto lo
  iface lo inet loopback
# Wired or wireless interfaces
#auto eth0
#iface eth0 inet dhcp
  auto eth0
  iface eth0 inet static
    address 10.50.0.10
    netmask 255.255.255.0
    network 10.50.0.0
    broadcast 10.50.0.255
    gateway 10.50.0.1
# Wireless interfaces
  auto wlan0
  iface wlan0 inet static
    wireless_mode ad-hoc
    wireless_essid RPI2
    wireless_channel 1
    address 192.168.3.100
    netmask 255.255.255.0
    network 192.168.3.0
    gateway 192.168.3.1
# Ethernet/RNDIS gadget (g ether)
# ... or on host side, usbnet and random hwaddr
  auto usb0
  iface usb0 inet static
    address 192.168.7.2
    netmask 255.255.255.0
    network 192.168.7.0
    gateway 192.168.7.1
```

Ilustración 57: Contenido del fichero interfaces.

Tras modificar el archivo se ha de guardar y reiniciar la Raspberry para activar los cambios realizados.

3.- Configuración de la red en el PC.

Red inalámbrica:

El primer paso es enchufar el usb wifi a la Raspberry e iniciarla.

Haciendo click en el icono de red del ordenador se abre un desplegable con los nombres de redes tanto inalámbricas como cableadas que detecta el ordenador. Una vez que nos detecte la red *RPI2* (es normal que tarde un rato) la seleccionamos.

Para acceder a la configuración de la red se selecciona *editar conexiones* en el menú de redes.

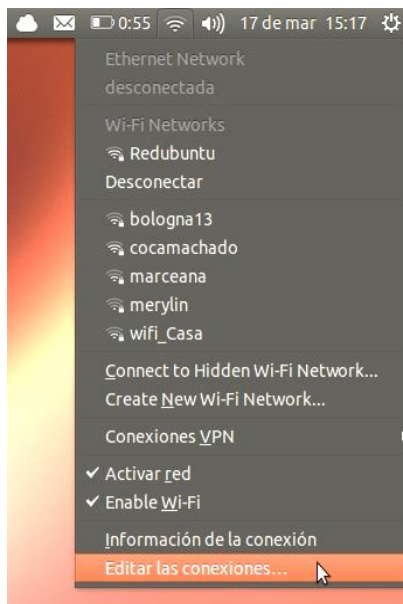


Ilustración 58: Editar conexiones Ubuntu.



Ilustración 59: Conexiones de red de Ubuntu

A continuación, se escoge *RPI2* y el botón *Editar...* y se cumplimentan las pestañas como se muestra en las siguientes imágenes.

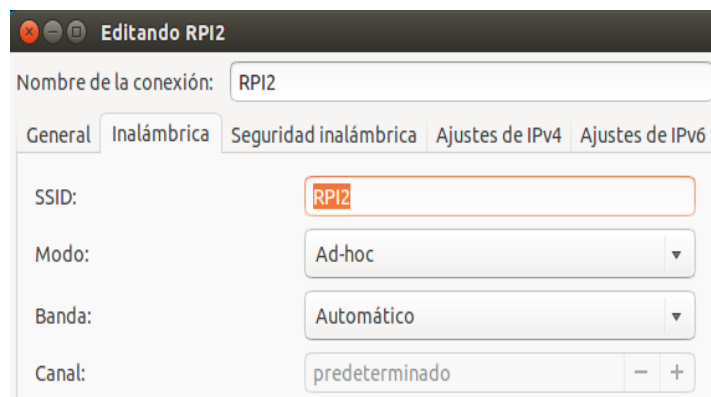


Ilustración 60: Configuración de red wifi en el pc.

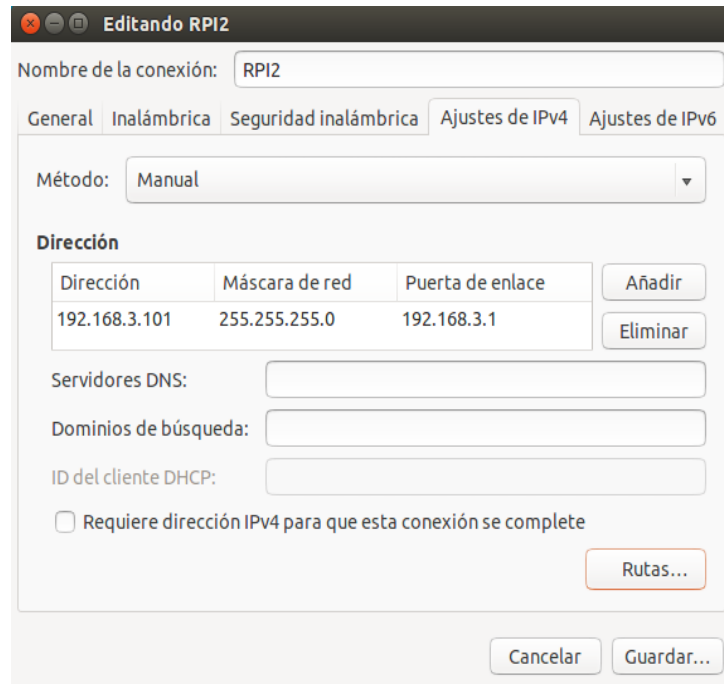


Ilustración 61: Configuración de red wifi en el pc.

Con esto ya se tiene configurada la conexión ssh por wifi entre ambos dispositivos.

Red cableada:

De forma equivalente al en el apartado anterior se ha de conectar el cable Ethernet entre la Raspberry y el PC.

Se selecciona *editar conexiones* y editamos la *red cableada* como se muestra en la siguiente imagen.



Ilustración 62: Configuración de red cableada en el pc.

4.- Conectarse a la Raspberry.

Desde la terminal del PC se escribe: `ssh pi@192.168.3.100` para la conexión ssh por wifi y `ssh pi@10.50.0.10` para la conexión por Ethernet, nos pregunta si es una conexión segura y hay que escribir *yes* y cuando pida la contraseña se escribe: *raspberrry*. Esto inicia el usuario PI del aparato que está en conectado con IP: 10.50.0.10 o 192.168.3.100. Tras la contraseña ha de aparecer algo similar a la imagen siguiente.

```
Linux navio-rpi 3.18.9-rt5-v7+ #1 SMP PREEMPT RT Thu Mar 26 10:31:34 UTC 2015 ar
mv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Wed Apr 27 17:17:51 2016 from 10.50.0.1
pi@navio-rpi ~ $
```

Ilustración 63: Ejemplo de línea de comandos al iniciar conexión ssh.

Con estos pasos ya se tiene la conexión ssh con la Raspberry tanto por wifi como por Ethernet.

*En fases de prueba de programas es más cómodo trabajar con el cable de Ethernet ya que permite estar conectado a internet a la vez que a la Raspberry.

**Para transferir archivos desde el PC a la Raspberry se utiliza el siguiente comando: “`scp [ubicación del archivo en tu PC] pi@192.168.3.100 (para conexión wifi)/ pi@10.50.0.10(para conexión cableada) :~[ubicación de destino en la Raspberry]`” (por ejemplo el traspaso de un ejecutable llamado *motorIMU* a la carpeta ejecutables de la Raspberry: `scp /home/fjavicasas/Escritorio/motorIMU/bin/Debug/motorIMU pi@10.50.0.10:~/ejecutables`).

ANEXO III. Habilitar los puertos I2C para la lectura de la IMU

En este anexo se realiza paso a paso el proceso de activación de los puertos I2C de la Raspberry para leer la información de la IMU.

1.- Conexión de la IMU con la Raspberry.

En primer lugar, para poder recibir información, hemos de tener conectada la IMU a la Raspberry en los GPIO SCL, SDA, 3,3V y a Ground mediante la placa COOKING HACKS como se muestra en la imagen.

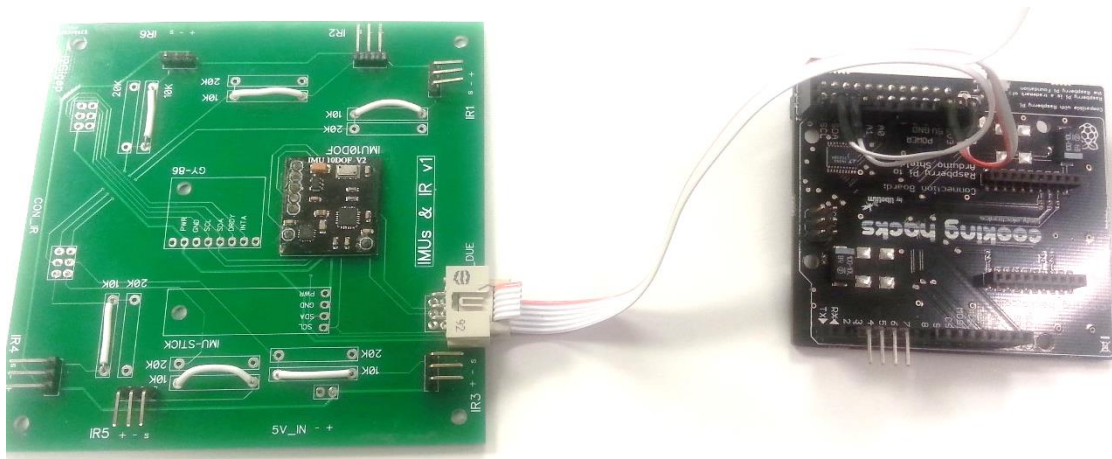


Ilustración 64: Conexión entre la IMU y la Cooking Hacks.

*Con la ayuda de un polímetro se averiguan las correspondientes salidas del cable.

**Esta conexión se puede realizar sin necesidad de dicha placa conectando directamente la IMU a los GPIO correspondientes.

***La explicación del proceso de comunicación mediante protocolo I2C se realiza en el apartado **¡Error! No se encuentra el origen de la referencia.**

2.- Actualización y activación de los protocolos de comunicación de la Raspberry.

En primer lugar se ha de comprobar que los archivos para la comunicación I2C están actualizados. Para ello se inicia sesión en Raspberry por ssh y se escribe lo siguiente:

```
sudo apt-get install Python-smbus
```

```
sudo apt-get install i2c-tools
```

En acabarse la instalación, se ha de acceder al menú de configuración de Raspberry escribiendo: `sudo raspi-config` y entrando en la casilla *Advanced options* → I2C

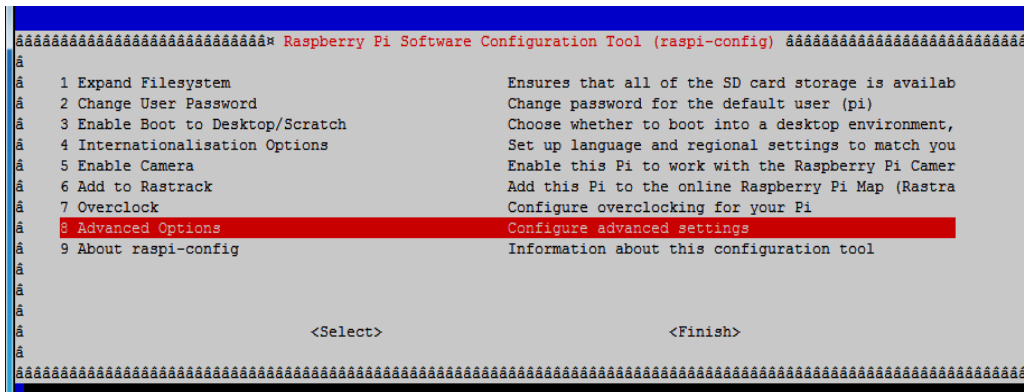


Ilustración 65: Raspi-config advanced options

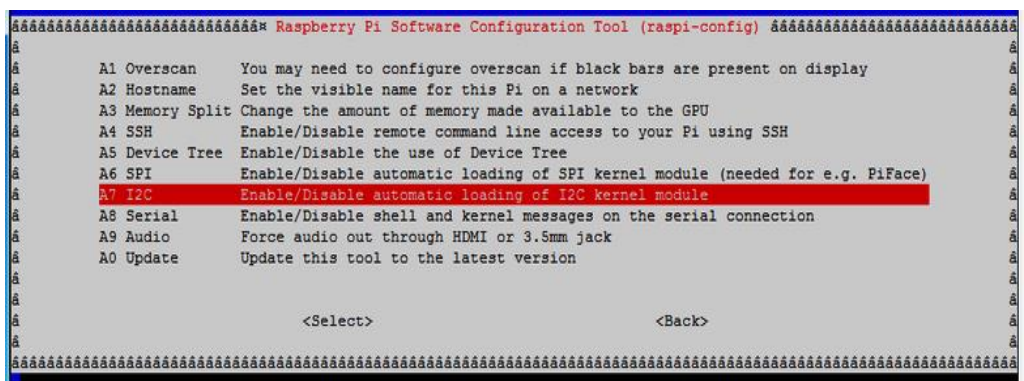


Ilustración 66: Raspi-config habilitar I2C

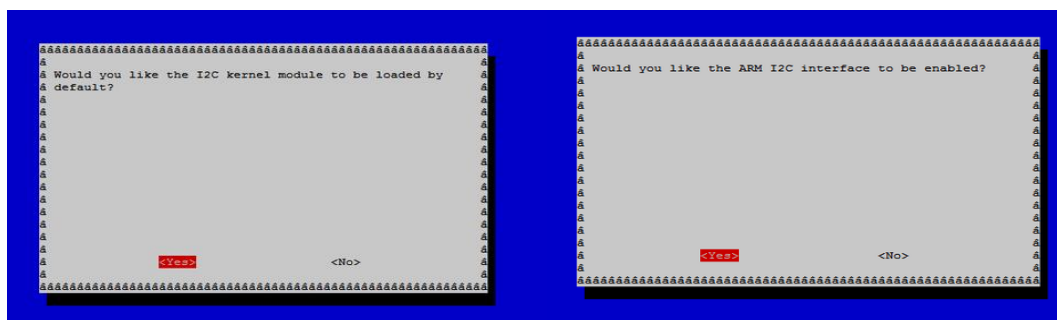
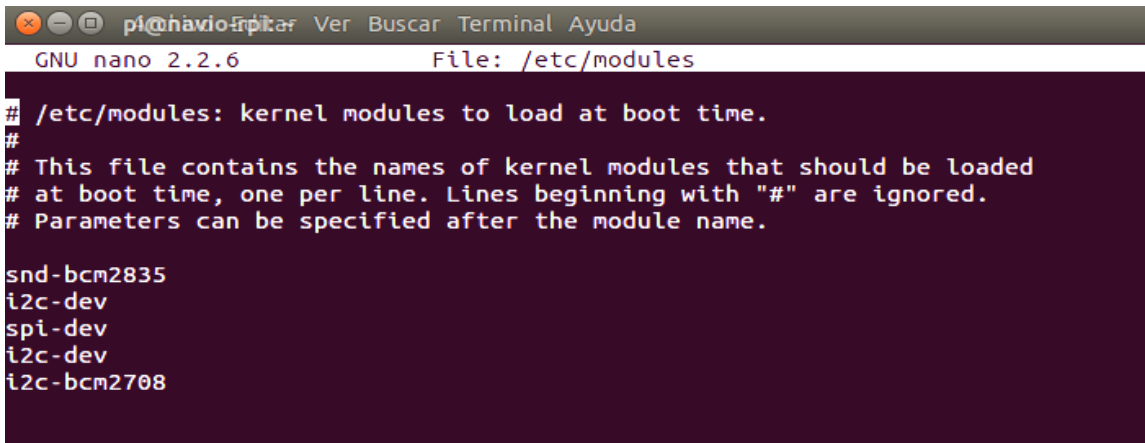


Ilustración 67: Raspi-config habilitar I2C.

Ilustración 68: Raspi-config habilitar I2C.

Con este paso se ha habilitado la conexión I2C de la Raspberry para la comunicación por I2C.

Se ha de reiniciar Raspberry para que los cambios sean efectivos: `sudo reboot` y se ha de volver a iniciar sesión en Raspberry. A continuación, será necesario modificar lo el archivo `modules` mediante el comando: `sudo nano /etc/modules` y, una vez dentro del fichero, añadir `i2c-bcm2708` `i2c-dev` al final del archivo, quedando el archivo como se muestra en la imagen.

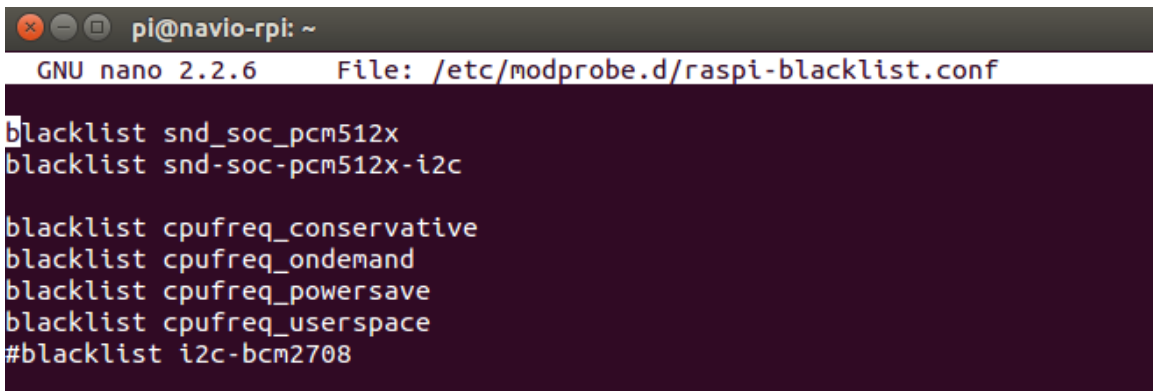


```
pi@navio-rpi: ~
GNU nano 2.2.6 File: /etc/modules

# /etc/modules: kernel modules to load at boot time.
#
# This file contains the names of kernel modules that should be loaded
# at boot time, one per line. Lines beginning with "#" are ignored.
# Parameters can be specified after the module name.

snd-bcm2835
i2c-dev
spi-dev
i2c-dev
i2c-bcm2708
```

Se ha de guardar el archivo y a continuación modificar el archivo *blacklist.conf* para elegir los programas que Raspberry ha, o no, de utilizar para el proceso de comunicación. Para ello se escribe: *sudo nano /etc/modprobe.d/raspi-blacklist.conf* y los archivos que coincidan con los de la Ilustración 69: Archivo *raspi-blacklist.conf*. se modifican como se muestra en dicha imagen.



```
pi@navio-rpi: ~
GNU nano 2.2.6 File: /etc/modprobe.d/raspi-blacklist.conf

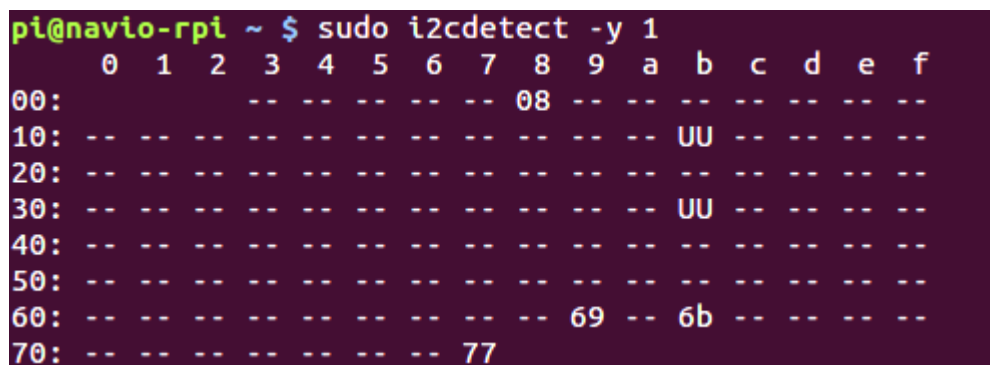
blacklist snd_soc_pcm512x
blacklist snd-soc-pcm512x-i2c

blacklist cpufreq_conservative
blacklist cpufreq_ondemand
blacklist cpufreq_powersave
blacklist cpufreq_userspace
#blacklist i2c-bcm2708
```

Ilustración 69: Archivo *raspi-blacklist.conf*.

Como en el paso anterior, hay que guardar los cambios y reiniciar para hacer efectivos los cambios.

Con estos pasos queda activada y configurada la conexión I2C de la Raspberry. Ya están los protocolos I2C de la Raspberry actualizados y configurados. Únicamente queda testear el puerto por el que se recibe la información. Los puertos activos se comprueban escribiendo: *sudo i2cdetect -y 1*.



```
pi@navio-rpi ~ $ sudo i2cdetect -y 1
   0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:  -- -- -- -- -- -- -- -- 08 -- -- -- -- -- --
10:  -- -- -- -- -- -- -- -- -- -- UU -- -- -- --
20:  -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30:  -- -- -- -- -- -- -- -- -- -- UU -- -- -- --
40:  -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50:  -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60:  -- -- -- -- -- -- -- -- 69 -- 6b -- -- -- --
70:  -- -- -- -- -- -- -- -- 77
```

Ilustración 70: Puertos que reciben información en I2C.

Para testear qué puertos reciben información se escribe: `i2cget -y 1 0x(puerto)` como se observa en la imagen.

```
pi@navio-rpi ~ $ sudo i2cget -y 1 0x69
0xf7
pi@navio-rpi ~ $ sudo i2cget -y 1 0x77
Error: Read failed
pi@navio-rpi ~ $ sudo i2cget -y 1 0x6b
Error: Read failed
pi@navio-rpi ~ $ sudo i2cget -y 1 0x08
0x00
```

Ilustración 71: Muestra de como averiguar el puerto que recibe datos de la IMU.

En nuestro caso el puerto que recibe la información de la IMU es el 0x69. Este dato es necesario saberlo para conocer la dirección desde donde se reciben los datos a la hora de la implementación del programa en C++.

ANEXO IV. Instalación de un compilador cruzado.

Requisitos:

- IDE Code::Blocks
- Compilador cruzado (25)

El compilador cruzado sirve para crear ejecutables que van a ser ejecutados en otra plataforma distinta de la que compila el programa. Es necesario para este trabajo ya que el compilador trabaja con Ubuntu con arquitectura X86 y donde se ejecuta trabaja con Raspbian con arquitectura Arm.

Se inicia el Code::Blocks y en la barra de herramientas: *Settings* -> *Compiler...*

En barra de *Selected Compiler* de la ventana de herramientas del compilador se hace click en *Copy* y después en *Rename* y se escribe el nombre que se le quiera dar al compilador cruzado, en este caso *RPI2*.

A continuación, se selecciona la pestaña de *Toolchain executables*. En la barra *Compiler's installation directory* ponemos la dirección de la carpeta que hemos descargado

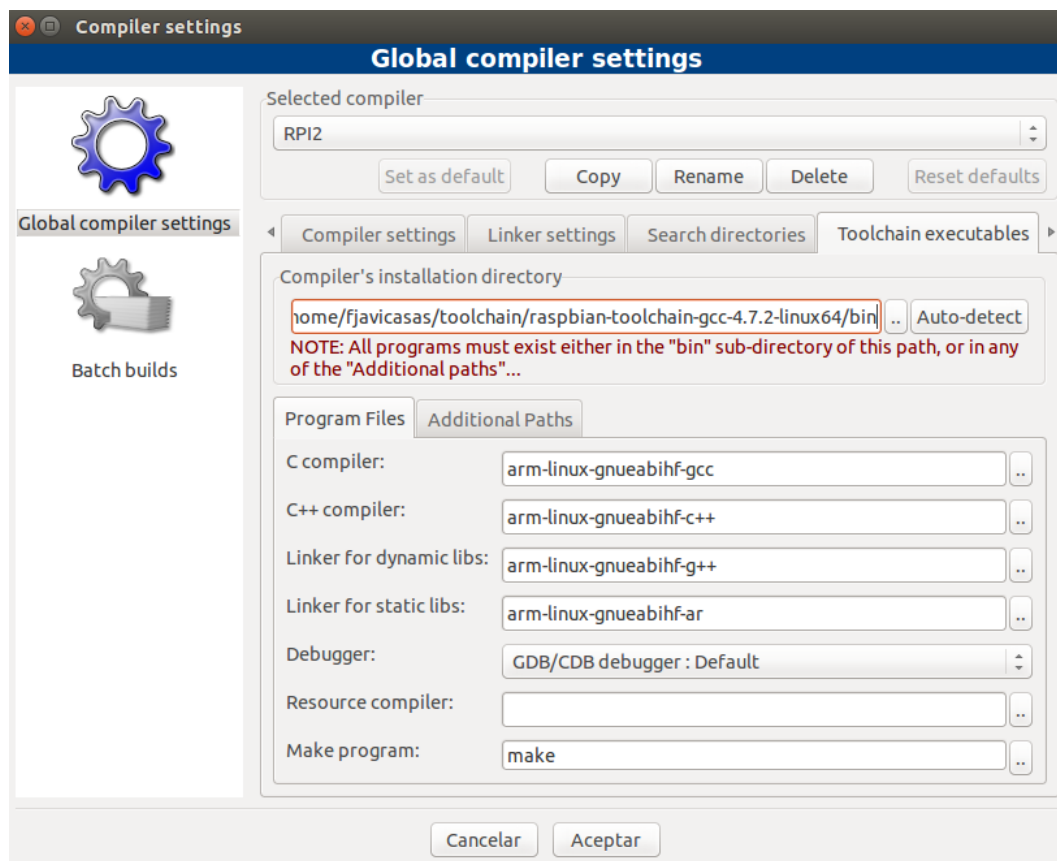


Ilustración 72: Configuración del compilador cruzado.

ANEXO V. Programa para establecer rangos de actuación de los motores.

Requisitos:

- IDE Code:Blocks (24) para Ubuntu con compilador cruzado (25) (ver ANEXO IV).
- Librería PIGPIO (32).

Como se ha comentado, a pesar de que todos los motores son iguales cada uno está conectado a drivers distintos. Esto supone que hay que conocer el mínimo impulso PWM que hace que el motor comience a girar y el máximo impulso que hace que el motor acelere, es decir, su rango de actuación.

Para conocerlo se va a realizar un programa en C++ que inicialice cada driver, con su motor, conectado a un GPIO disponible en la Raspberry Pi 2 (y si se va a usar la placa *Cooking Hacks* hay que asegurarse que estén disponibles también en dicha placa).

1.- Crear un proyecto y añadir las librerías.

Se requiere de la librería *PIGPIO*, concretamente de los archivos: *pigpio.h*, *pigpio.c*, *command.c* y *command.h*.

En primer lugar, se crea un proyecto de *Console application* estableciendo como compilador al compilador cruzado.

A continuación, se ha de añadir los archivos a la carpeta donde se halla el proyecto y añadirlos al proyecto haciendo click con el botón derecho sobre el proyecto en Code::Blocks -> *add files...* y seleccionando las librerías mencionadas.

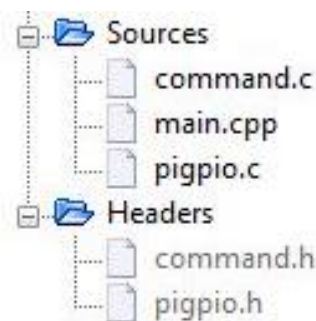


Ilustración 73: Librerías incluidas en el programa para probar los motores.

2.- Escribir el código.

Para la implementación de este programa se utilizan las siguientes funciones:

- **gpioInitialise():** establece la conexión con los motores. Sin esta función no se pueden iniciar los impulsos PWM.
- **gpioSetPWMfrequency(pin, [frecuencia]):** establece la frecuencia a la que trabajan los motores, es decir, la frecuencia a la que puede cambiar de valor de la velocidad del motor.
- **gpioSetPWMrange(pin, [rango]):** establece la variación mínima de pulso PWM, que se traduce a la mínima variación de la velocidad de los motores. Esta precisión aumentará cuanto mayor sea el valor del rango. En este programa se ha establecido que el rango va

de 0 a 2500, lo que significa que la variación mínima de porcentaje tiempo de señal activa es de 0.04%.

- **gpioGetMode(pin):** obtiene si el pin está a 1 (activo, a 3,3V) o a 0 (apagado).
- **gpioSetMode(pin, [0/1]):** establece el valor 1 o 0 al pin.
- **gpioPWM(pin, [valor dentro del rango]):** establece un valor dentro del rango anteriormente establecido, el cual se traducirá en un impulso PWM, que será de mayor amplitud cuanto más se acerque al valor del extremo superior del rango.
- **gpioTerminate():** finaliza la comunicación PWM de todos los GPIO de la Raspberry.
- Conocidas estas funciones ya se puede implementar el programa.

En el *main* se escribe lo siguiente:

```
#include <stdio.h>
#include <unistd.h>
#include <pthread.h>
#include "pigpio.h"

int main()
{
    int i, pin;
    printf("introduce el pin conectado al motor que desea testear:\n");
    scanf("%d",&pin);

    if (gpioInitialise() < 0) {
        printf("motor no inicializado\n"); }
    else {
        printf("motor inicializado, YESSS\n"); }

    gpioSetPWMPfrequency(pin, 400);
    gpioSetPWMrange(pin, 2500);
    gpioSetMode(pin, 0);
    gpioPWM(pin, 1000);
    printf("pin al mínimo\n");
    sleep (4);
    for(i=1000 ;i<2000; i=i+10) {
        gpioPWM(pin, i);
        printf("pin: %d a %d\n",pin,i);
        sleep(2);
    }
    return 0;
}
```

Después de compilar únicamente queda pasar a la Raspberry el ejecutable que se encuentra dentro de la carpeta *bin/Debug* de la carpeta del proyecto, ejecutarlo y apuntar los valores a los que se inicia el giro de las hélices y los de máximo giro.

ANEXO VI. Programa para comprobar las medidas de los ángulos de la IMUMPU 6050.

Requisitos:

- IDE Code::Blocks (24).
- *ComponentSensors.cc*, *ComponentSensors.h*, *i2cDevice.cc*, *i2cDevice.h*, *i2c-dev.h*, *ImuMPU.cc*, *ImuMPU.h*, *KalmanDesacoplado.cc*, *KalmanDesacoplado.h*, *MatrixBasicLibrary.cc*, *MatrixBasicLibrary.h*, *MatrixMath.cc*, *MatrixMath.h*, *Sensors.h*. (32)

Este programa se utiliza para comprobar las medidas de la IMU (los giros positivos y negativos de la IMU, los errores de medida en ángulo 0, etc.).

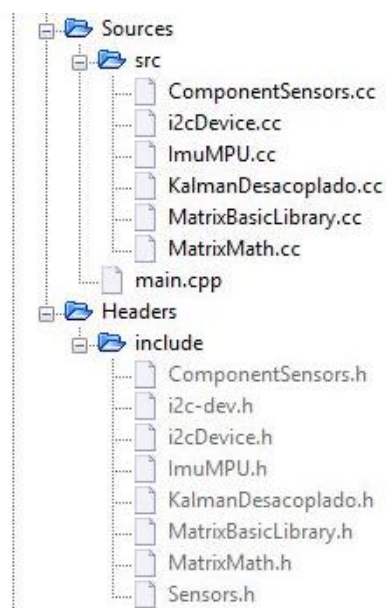


Ilustración 74: Librerías incluidas en el programa para comprobar medidas de la IMU.

Se necesita saber el funcionamiento de las siguientes funciones:

- **ImuMPU(frecuencia)**: establece la frecuencia a la que la IMU envía lecturas.
- **Setup()**: esta función está en *ComponentSensors.h* e inicializa la IMU.
- **Start()**: esta función se encuentra en la misma librería que la anterior e inicia las lecturas de la IMU.
- **ReturnPitch()**: esta función devuelve el valor en bruto del Pitch (sin pasar por el filtro de Kalman)
- **ReturnRoll()**: esta función devuelve el valor en bruto del Roll (sin pasar por el filtro de Kalman)
- **ReturnRollDEG()**: devuelve la medida en grados del Roll.
- **ReturnPitchDEG()**: devuelve la medida en grados del Pitch.

- **ReturnYawDEG():** devuelve la medida en grados del Yaw.
- **KalmanDesacoplado(frecuencia, ImuMPU *objeto):** inicializa el filtro y se especifica de donde recibe los datos.

En el *main* se escribe lo siguiente:

```
#include <iostream>
#include <iostream>
#include "../include/ImuMPU.h"
#include "../include/i2c-dev.h"
#include "../include/i2cDevice.h"
#include "../include/Sensors.h"
#include "ImuMPU.h"
#include "KalmanDesacoplado.h"

using namespace std;

int main()
{
    int i;
    i=0;
    ImuMPU *IMU_MPU;
    KalmanDesacoplado *b;

    IMU_MPU = new ImuMPU(333);
    IMU_MPU->Setup();
    IMU_MPU->Start();

    b = new KalmanDesacoplado(333,IMU_MPU);
    b->Setup();
    b->Start();
    IMU_MPU->Setup();

    While(i<0){
    IMU_MPU->Start();
    cout << " pitch:" << b->ReturnPitchDEG();
    cout << " roll:" << b->ReturnRollDEG();
    cout << " roll:" << b->ReturnYawDEG();
    i++;
    }
    return 0;
}
```

ANEXO VII. Implementación del PID

La implementación de la acción de control explicada en el punto 9.1 Control de orientación se implementa en el archivo `x3d.cc` de la siguiente manera:

Para su comprensión se necesita conocer el funcionamiento de la función:

- **saturación(Variable, Límite_superior, Límite_inferior):** esta función limita el valor que puede alcanzar la variable introducida tanto superior como inferiormente. En este caso se utiliza: en las acciones de control para evitar que hagan inestable el sistema ante perturbaciones muy altas, o en los motores para evitar que se mande una referencia de carga superior a la que pueden dar.

(6)El código es el siguiente:

```
                //Control del pitch

wpp=saturacion(Board->kppitch*(x3drefpitch-Board->KalmanPredictor-
                >ReturnPitchDEG()),0.2,-0.2);

wpd=saturacion(Board->kdpitch*(-Board->KalmanPredictor-
                >ReturnVpitchDEG()),0.1,-0.1);

wpi=saturacion(wpi+(x3drefpitch-Board->KalmanPredictor-
                >ReturnPitchDEG())*Board->kipitch,Board->kix,-Board->kix);

wp=wpd+wpp+wpi;

                //Control del roll

wrp=saturacion(Board->kproll*(x3drefroll-Board->KalmanPredictor-
                >ReturnRollDEG()),0.2,-0.2);

wrd=saturacion(Board->kdroll*(-Board->KalmanPredictor-
                >ReturnVrollDEG()),0.1,-0.1);

wri=saturacion(wri+(x3drefroll-Board->KalmanPredictor-
                >ReturnRollDEG())*Board->kiroll,Board->kix,-Board->kix);

wr=wrd+wrp+wri;

                //Control del yaw

wyp=saturacion(Board->kpyaw*(x3drefyaw-Board->KalmanPredictor-
                >ReturnYawDEG()),0.4,-0.4);

wyd=saturacion(Board->kdyaw*(-Board->KalmanPredictor-
                >ReturnVyawDEG()),0.2,-0.2);

wyi=saturacion(wyi+(x3drefyaw-Board->KalmanPredictor-
                >ReturnYawDEG())*Board->kiyaw,Board->kix,-Board->kix);

wy=wyd+wyp+wyi;
```

Donde w_r , w_p y w_y son las acciones de control en el roll, pitch y yaw respectivamente. K_p __, k_d __ y k_i __ son las ganancias controlados desde el HMI para el ajuste de la parte proporcional, derivada e integral de cada ángulo. Se utilizao la misma sintaxis para w_{pp} , w_{rp} , w_{yp} , etc. *ReturnVrollDEG()*, *ReturnVpitchDEG()*, *ReturnVyawDEG()* son funciones que devuelven la velocidad de cada ángulo en grados/segundo; *ReturnRollDEG()*, *ReturnPitchDEG()* y *ReturnYawDEG()* son funciones que devuelven la medida en grados obtenida por el sensor y filtrada en el filtro de Kalman para cada ángulo.

Estas acciones de control se aplican a los motores en el mismo archivo mediante el siguiente código.

```
refmotor1=saturacion(Board->umotor+(wr-wp+wy),1,0); //pin 18  
refmotor2=saturacion(Board->umotor+wr+wp-wy,1,0); //pin 23  
refmotor3=saturacion(Board->umotor+(-wr+wp+wy),1,0); //pin 24  
refmotor4=saturacion(Board->umotor-wr-wp-wy,1,0); //pin 25
```

Donde *umotor* es la referencia de carga aplicada a los motores y *refmotorX* son las referencias aplicadas a cada motor teniendo en cuenta las acciones de control.

12 Ilustraciones, tablas, ecuaciones y gráficas

12.1 Ilustraciones

Ilustración 1: Breguet-Richet Autogiro (1907).	3
Ilustración 2: Oehmichen No 2.....	3
Ilustración 3: Modelo A quadrotor de Convertawings(1956).	4
Ilustración 4: UAVs actuales (1).	4
Ilustración 5: Cronología de los nombres aplicados a las aeronaves robóticas.	5
Ilustración 6: Helicóptero UAV.....	7
Ilustración 7: Aeroplano UAV.....	8
Ilustración 8: Zeppelin UAV.....	8
Ilustración 9: Quadrotor UAV.	8
Ilustración 10: Quadrotor con seguimiento automático.....	9
Ilustración 11: Ángulos de Tait-Bryan (Roll, Pitch, Yaw) en configuración X.	11
Ilustración 12: Configuración en cruz.....	12
Ilustración 13: Configuración en X.	12
Ilustración 14: Orientación del quadrotor en relación con sus motores y ejes.	13
Ilustración 15: Fuerzas y sentidos de giro de las hélices.....	14
Ilustración 16: Movimientos longitudinales quadrotor (3)	15
Ilustración 17: Movimientos laterales (3).	15
Ilustración 18: Movimientos de rotación (3).....	16
Ilustración 19: Raspberry Pi 2 (4)	17
Ilustración 20: Elementos que componen la Raspberry Pi 2 modelo B (5).	18
Ilustración 21: Orientación de los ejes de la IMU (7).....	21
Ilustración 22: IMU MPU6050 10Dof v2 (7).....	20
Ilustración 23: IMU MPU6050 10Dof v2 en la placa de circuitos.....	21
Ilustración 24: Motores Brushless BL Outrunner 2824-34.....	21
Ilustración 25: Hélices de 10x4.5"	22
Ilustración 26: Drivers: HW30A 30A Brushless ESC YELLOW, YGE 30i y YGE 25i.	22
Ilustración 27: Raspberry Pi to Arduino Shields Connection Bridge (8).....	23
Ilustración 28: Especificación de puertos de Raspberry Pi to Arduino Shields Connection Bridge (8).....	23
Ilustración 29: USB wifi Ralink.....	23
Ilustración 30: Cable Ethernet.....	23
Ilustración 31: Fuente de alimentación MAAS SPS-9600 (9).....	24
Ilustración 32: Batería Gens ace 5300 mAh (10).....	24
Ilustración 33: Conversor de tensión.	24
Ilustración 34: Esquema de conexiones.....	25
Ilustración 35: Señales PWM.....	26
Ilustración 36: Esquema de los paquetes del protocolo de comunicación I2C (11).	27
Ilustración 37: Elementos de la conexión I2C (12)	27
Ilustración 38: Diagrama de comportamiento de tareas con prioridades.	29
Ilustración 39: Diagrama de comportamiento de tareas con prioridades y semáforo.	30
Ilustración 40: Resultados de latencia del sistema operativo de tiempo real.....	31
Ilustración 41: HMI.....	35
Ilustración 42: Plataforma Hover de Quanser (15)	37

Ilustración 43: Rótula 1 (Sistema completo de cojinete de fricción con soporte más el perno esférico con rosca macho) (16).	38
Ilustración 44: Rótula 2 (Mini ball head hot shoe adapter) (17).	38
Ilustración 45: Montaje de la base y el mástil.....	39
Ilustración 46: Montaje 1.....	39
Ilustración 47: Montaje 1 con quadrotor.	39
Ilustración 48: Placa metálica para el montaje 2.	39
Ilustración 49: Montaje 2 con quadrotor	40
Ilustración 50: Rótula 2 enganchada a la base.	40
Ilustración 51: Montaje 2 con quadrotor.	40
Ilustración 52: Esquema de control.....	41
Ilustración 53: Quadrotor durante el ajuste del control.	42
Ilustración 54: Grabar imagen del sistema con Win32DiskImager.....	51
Ilustración 55: Raspi-config.	52
Ilustración 56: Raspi-config habilitar ssh.....	53
Ilustración 57: Contenido del fichero interfaces.	53
Ilustración 58: Editar conexiones Ubuntu.....	49
Ilustración 59: Conexiones de red de Ubuntu	54
Ilustración 60: Configuración de red wifi en el pc.....	54
Ilustración 61: Configuración de red wifi en el pc.....	55
Ilustración 62: Configuración de red cableada en el pc.	55
Ilustración 63: Ejemplo de línea de comandos al iniciar conexión ssh.	56
Ilustración 64: Conexión entre la IMU y la Cooking Hacks.....	57
Ilustración 65: Raspi-config advanced options.....	58
Ilustración 66: Raspi-config habilitar I2C.....	58
Ilustración 67: Raspi-config habilitar I2C.....	58
Ilustración 68: Raspi-config habilitar I2C.....	58
Ilustración 69: Archivo raspi-blacklist.conf.....	59
Ilustración 70: Puertos que reciben informacion en I2C.....	59
Ilustración 71: Muestra de como averiguar el puerto que recibe datos de la IMU.	60
Ilustración 72: Configuración del compilador cruzado.....	61
Ilustración 73: Librerías incluidas en el programa para probar los motores.....	63
Ilustración 74: Librerías incluidas en el programa para comprobar medidas de la IMU.	65

12.2 Tablas

Tabla 1: Clasificación de los UAVs.	6
Tabla 2: Clasificación de los UAVs de corto alcance.....	7
Tabla 3: Especificaciones de Raspberry Pi y Raspberry Pi 2.....	19
Tabla 4: Origen y destino de las conexiones.....	25
Tabla 5: Esquema de los paquetes del protocolo de comunicación UDP.	28
Tabla 6: Esquema de los paquetes del protocolo de comunicación TCP.	28

12.3 Ecuaciones

Ecuación 1: Proyección de la aceleración del conjunto en el eje X.	13
Ecuación 2: Proyección de la aceleración del conjunto en el eje Y.....	13
Ecuación 3: Proyección de la aceleración del conjunto en el eje Z.	13
Ecuación 4: Aceleración angular en Yaw.....	14
Ecuación 5: Aceleración angular en Pitch.....	14
Ecuación 6: Aceleración angular en Roll.....	14

12.4 Gráficas

Gráfica 1: Ajuste del Roll con un PD.....	43
Gráfica 2: Ajuste del Roll con un PID.....	44
Gráfica 3: Ajuste del Pitch con un PID.....	44
Gráfica 4: Ajuste del Yaw con un P.	45
Gráfica 5: Ajuste del Yaw con un PI.	46

13 Bibliografía de referencia y de imágenes.

1. Universidad de Valencia. [En línea] <http://drones.uv.es/>.
2. Think big, drones autónomos. [En línea] <http://blogthinkbig.com/drones-autonomos-camara-video/>.
3. Diseño de algoritmos de navegación y. [aut. libro] Claudio Pose. *Tesis de grado de Ingeniería Electrónica*. Universidad de Buenos Aires : s.n., 2015.
4. *Proyecto de Trabajo de Grado: Plataforma para el control cooperativo y estabilidad en cuadrotros*. Levkovsky, Juan Pablo Jiménez Michael. Bogotá : Pontificia Universidad Javeriana, Facultad de electrónica, 2013.
5. P. Castillo, R. Lozano y A. Dzul. *Modelling and Control of Mini-Flying Machines, Springer-Verlag in Advances in Industrial Control, 2005*. 2005.
6. Frasset, Alberto Castillo. *Desarrollo integral de un quadrotor: diseño de un algoritmo de control para la posición X-Y basado en señales GPS*. Valencia : Universidad politécnica de Valencia, 2014.
7. Lorda, Luis Ródenas. *Plataforma de desarrollo para el control de estabilidad en tiempo real de un vehículo aéreo tipo quadrotor*. Valencia : Universidad Politécnica de Valencia, 2013.
8. Rodríguez, Omar Luque. *Estudio y desarrollo de un sistema de control de un cuadricóptero*. Universidad del País Vasco : s.n., 2014.
9. Raspipc. [En línea] <http://raspipc.es/blog/>.
10. Community Forum - Emlid. [En línea] <https://community.emlid.com/t/navio-on-board-connectors-and-poweroff>.
11. Raspberry Pi - Teach, Learn, and Make with Raspberry Pi. [En línea] <https://www.raspberrypi.org/products/raspberry-pi-2-model-b/>.
12. Drotek. [En línea] <http://www.drotek.com/>.
13. Garín, Vicente Balaguer. *Sistema de tiempo real basado en Raspberry Pi para el control de un quadrotor*. Valencia : Universidad Politécnica de Valencia, 2015.
14. Cooking Hacks. [En línea] <https://www.cooking-hacks.com>.
15. Ralink. [En línea] <http://ralink.software.informer.com/>.
16. Maas elektronik. [En línea] <http://www.maas-elektronik.com/>.
17. Gensace. [En línea] <http://www.gensace.de/>.
18. Sparkfun. [En línea] <https://learn.sparkfun.com>.
19. Cypress. [En línea] <http://www.cypress.com/>.
20. Hansson, Niklas and Rudner, Mikael. *Control of a Quadrotor*. Lund, Suecia : LUND UNIVERSITY LIBRARIES, 2011.

21. Universidad de Oviedo, isa. Sistemas operativos de tiempo real (SOTR). [En línea] Universidad de Oviedo, departamento de sistemas y automática.
<http://isa.uniovi.es/docencia/TiempoReal/Recursos/temas/sotr.pdf>.
22. Redeweb. [En línea] <http://www.redeweb.com/>.
23. Emlid. [En línea] <https://docs.emlid.com>.
24. Code::Blocks. [En línea] <http://www.codeblocks.org/>.
25. Compilador cruzado arm gnuabi. [En línea]
<https://s3.amazonaws.com/RTI/Community/ports/toolchains/raspbian-toolchain-gcc-4.7.2-linux64.tar.gz>.
26. Díaz-Otero, Natalia. *Desarrollo de un sistema de telemetría e interfaz gráfica para el control de un vehículo aéreo pilotado remotamente (RPAS)*. Valencia : s.n., 2015.
27. Quanser. [En línea] <http://www.quanser.com/>.
28. Icus. [En línea] <http://www.igus.es/>.
29. Cineroid. [En línea] <http://www.cineroid.com/>.
30. Raffo, Guilherme Viana. *Modelado y control de un helicóptero quadrotor*. Sevilla : s.n., 2007.
31. Karl J. Aström, Björn Wittenmark. *Computer-controlled systems, Theory and Design*.
32. Libreria Pigiopio. [En línea] <http://abyz.co.uk/rpi/pigpio/>.

Desarrollo de un software de control basado en Raspberry Pi II para la implementación de algoritmos de control en una plataforma 3D (con 3 grados de libertad)

Presupuesto

Contenido

1	Introducción	1
1.1	Información previa	1
1.1.1	Mano de obra	1
1.1.2	Tareas a realizar	1
1.1.3	Componentes del quadrotor	2
1.1.4	Componentes del banco de pruebas.....	3
1.1.5	Material alquilado utilizado.....	3
2	Detalles del presupuesto	5
2.1	Cuadro de precios unitarios	5
2.2	Cuadro de precios descompuestos	5
2.3	Presupuesto de ejecución material, presupuesto de ejecución por contrata y presupuesto base de licitación.....	12

1 Introducción

En el presente presupuesto se proporciona toda la información del precio total del trabajo desarrollado. El presupuesto total se ha dividido en capítulos para desarrollar la información presupuestaria de cada proceso desarrollado para llegar a los objetivos marcados.

1.1 Información previa

En un primer apartado se muestra la información de los componentes del quadrotor y su precio, en el segundo apartado se muestra la información de los materiales utilizados para construir el banco de pruebas y hora del material utilizado y por último el salario por hora del ingeniero graduado en tecnologías industriales.

1.1.1 Mano de obra.

<i>Código</i>	<i>Mano de obra</i>	<i>Precio (€) /hora</i>	<i>Horas</i>	<i>Importe (€)</i>
MO.GITI	Graduado en Tecnologías Industriales	20	288,5	5770

1.1.2 Tareas a realizar

<i>Tareas</i>	<i>Tiempo (h)</i>
<i>Revisión de conexiones y elementos de la base del quadrotor</i>	2
<i>Información previa y preparación de software necesario</i>	50
<i>Construcción del convertidor de tensiones</i>	1.5
<i>Montaje y conexionado de todos los elementos</i>	5
<i>Implementación de los algoritmos de control</i>	150
<i>Montaje del banco de pruebas</i>	10
<i>Ajuste del control</i>	50
<i>Realización del modelo en CAD y planos</i>	20

1.1.3 Componentes del quadrotor

<i>Código</i>	<i>Componentes del quadrotor</i>	<i>Precio (€)</i>	<i>Nº de unidades</i>	<i>Importe (€)</i>
<i>M.RPI2</i>	Raspberry Pi2	39,9	1	39,9
<i>M.TSD</i>	Tarjeta micro-SD de 8GB de clase 10	7,16	1	7,16
<i>M.EF450</i>	Estructura dji Flame wheel arf kit f450	49,99	1	49,99
<i>M.WIF</i>	Wifi-USB de alcance extendido	9,8	1	9,8
<i>M.RBRIDGE</i>	Raspberry Pi to Arduino Shields Connection Bridge	40	1	40
<i>M.MOT</i>	Motores Brushless BL Outrunner 2824-34	28,95	4	115,8
<i>M.DY</i>	Driver HW30A 30A Brushless ESC YELLOW	16,52	2	33,04
<i>M.D30I</i>	Driver YGE 30i	25,95	1	25,95
<i>M.D25I</i>	Driver YGE 25i	32,95	1	32,95
<i>M.IMU</i>	IMU MPU6050	18,9	1	18,9
<i>M.HEL</i>	Hélice genérica de 10x4,5	2,5	4	10
<i>M.BAT</i>	Gens Ace 5300mAh 11.1V 30/60C 3S1P Bateria Lipo	69	1	69
<i>M.CT</i>	Módulo LM2596 Convertidor de Voltaje DC-DC Buck 1.25V-35V	9,9	1	9,9
<i>M.PC</i>	Placa de circuitos SODIAL(R) 4pcs Doble-Lado Prototipo FR-4 PCB Stripboard	2,42	1	2,42
<i>M.RM</i>	Regleta Clema enchufable macho recto 5,00mm - 2 contactos.	0,25	2	0,5
<i>M.RH</i>	Regleta Clema enchufable hembra acodada 5,00mm - contactos	0,59	2	1,18
<i>M.HE</i>	Rollo de hilo de Estaño para electrónica 100gr	8,49	1	8,49
Total				474,98

1.1.4 Componentes del banco de pruebas

<i>Código</i>	<i>Componentes del banco de pruebas</i>	<i>Precio (€)</i>	<i>Nº de unidades</i>	<i>Importe (€)</i>
<i>M.MM</i>	Bloque de madera de 55 x 55 x 375 mm	4	1	4
<i>M.BM</i>	Tabla de madera de 237 x 400 x 15 mm	3	1	3
<i>M.EM</i>	Escuadra metálica de 40 mm de alto con 4 agujeros	0,5	4	2
<i>M.TM</i>	Tornillo para madera de 4 mm de diámetro	0,1	20	2
<i>M.R1</i>	Sistema completo de cojinete de fricción con soporte más el perno esférico con rosca macho GFSM-08-AG	5,67	1	5,67
<i>M.R2</i>	Mini ball head MBH-M	39	1	39
<i>M.PM</i>	Placa metálica de 50 x 50 x 1	1	1	1
<i>M.TN</i>	Tuerca Tee Nut	0,7	1	0,7
<i>M.PI</i>	Pegamento instantáneo	7	1	7
			<i>Total</i>	64,37

1.1.5 Material alquilado utilizado

<i>Código</i>	<i>Material utilizado</i>	<i>Precio (€)/ hora</i>	<i>Horas</i>	<i>Importe (€)</i>
<i>MA.FA</i>	Fuente de alimentación regulable 30A	7,5	58,5	438,75
<i>MA.TL</i>	Taladradora	5	1	5
<i>MA.LM</i>	Lima para madera	0,5	1	0,5
<i>MA.SE</i>	Soldador de estaño	4	1,5	6
<i>MA.POL</i>	Polímetro	1,3	3,5	4,55
<i>MA.OMG</i>	Ordenador de media gama	8,7	275	2392,5
			<i>Total</i>	2847,3

2 Detalles del presupuesto

2.1 Cuadro de precios unitarios

El presupuesto se ha dividido en unidades de obra que corresponden a las tareas a realizar mencionadas anteriormente. La siguiente tabla muestra el precio total de la cada una de las unidades de obra.

<i>Código</i>	<i>Unidad de obra</i>	<i>Precio (€)</i>
UO-1	Revisión del montaje de la estructura	4808,61
UO-2	Información previa y preparación del software necesario	1623,23
UO-3	Implementación de los algoritmos de control	4434,15
UO-4	Construcción del convertidor de tensión.	69,97
UO-5	Montaje y conexionado de los elementos	239,37
UO-6	Montaje del banco de pruebas	267,66
UO-7	Ajuste del control	1735,55
UO-8	Realización del modelo en CAD y planos	591,22

2.2 Cuadro de precios descompuestos

En este apartado se da toda la información detallada de los costes de cada unidad de obra. Las unidades de obra introducidas son: Revisión del montaje de la estructura, preparación del software, construcción del conversor de tensiones, montaje y conexionado de todos los elementos, implementación de los algoritmos de control, montaje del banco de pruebas, ajuste del control y realización del modelo en CAD y planos.

Cabe destacar que, puesto que la estructura de la que se parte es al que se utilizó para el trabajo de Vicente Balaguer Garín, se ha indicado el presupuesto base de licitación únicamente de la unidad de obra encargada de la construcción de dicha estructura del trabajo de Vicente. Dicho presupuesto se incluye en la unidad de obra 1.

Desarrollo de un software de control basado en Raspberry Pi II para la implementación de algoritmos de control en una plataforma 3D (con 3 grados de libertad)

Construcción del quadrotor por Vicente Balaguer Garín.

Descripción	Unidades	Rendimiento	Precio (€)	Importe (€)
Graduado en Tecnologías Industriales	h	70	35	2450
Técnico	h	3	25	75
Frame dji f450	Ud	1	39	39
Lamina de polietileno de 6mm	cm2	200	0,029	5,8
Motores Brushless BL Outrunner 2824-34	Ud	4	28,95	115,8
Driver HW30A 30A Brushless ESC YELLOW	Ud	2	16,52	33,04
Driver YGE 30i	Ud	1	25,95	25,95
Driver YGE 25i	Ud	1	32,95	32,95
Osciloscopio	h	3	52,6	157,8
Hélice genérica de 10x4,5	Ud	4	2,5	10
Fuente de alimentación regulable 30A	h	10	7,5	75
Cables y tornillería en general	Ud	1	40	40
<i>Costes directos complementarios (1%)</i>				30,60
<i>Costes indirectos (2%)</i>				61,81
TOTAL PRECIO UNIDAD DE OBRA				3152,76
<i>Gastos generales (15%)</i>				472,91
<i>Beneficio Industrial (6%)</i>				189,16
<i>Presupuesto de ejecución por contrata</i>				3814,84
<i>IVA (21%)</i>				801,11
Presupuesto base de licitación				4615,95

UO-1 *Revisión del montaje de la estructura* *Revisión y cambio de elementos instalados en la estructura utilizada para un proyecto anterior. Voltaje, funcionamiento de motores y drivers y comprobación de tornillería.*

Código	Descripción	Unidades	Rendimiento	Precio (€)	Importe (€)
MO.GITI	Graduado en Tecnologías Industriales	Ud	2	20	40
M.EST	Estructura del quadrotor	Ud	1	4615,96	4615,96
MA.FA	Fuente de alimentación MAAS SPS-9600	h	2	5	10
MA.POL	Polímetro	h	2	1,3	2,6
<i>Total</i>					4668,56
<i>Costes directos complementarios (1%)</i>					46,68
<i>Costes indirectos (2%)</i>					93,37
TOTAL PRECIO UNIDAD DE OBRA					4808,61

UO-2 *Información previa y preparación del software necesario* *Recopilación de información e instalación y preparación previa del software como el SOTR, habilitar conexiones, etc.*

Código	Descripción	Unidades	Rendimiento	Precio (€)	Importe (€)
MO.GITI	Graduado en Tecnologías Industriales	h	50	20	1000
M.TSD	Tarjeta micro-SD de 8GB de clase 10	ud	1	7,16	7,16
M.RPI2	Raspberry Pi 2	Ud	1	39,9	39,9
MA.MNT	Monitor	h	10	5,1	51
MA.TUSB	Teclado USB	h	10	2,3	23
MA.CHDMI	Cable HDMI	Ud	1	19,9	19,9
MA.OMG	Ordenador de media gama	h	50	8,7	435
<i>Total</i>					1575,96
<i>Costes directos complementarios (1%)</i>					15,75
<i>Costes indirectos (2%)</i>					31,51
TOTAL PRECIO UNIDAD DE OBRA					1623,24

Desarrollo de un software de control basado en Raspberry Pi II para la implementación de algoritmos de control en una plataforma 3D (con 3 grados de libertad)

UO-3 *Implementación de los algoritmos de control* *Programación del software encargado del movimiento del quadrotor y su respuesta ante perturbaciones y referencias*

Código	Descripción	Unidades	Rendimiento	Precio (€)	Importe (€)
MO.GITI	Graduado en Tecnologías Industriales	Ud	150	20	3000
MA.OMG	Ordenador de media gama	h	150	8,7	1305
<i>Total</i>					4305
<i>Costes directos complementarios (1%)</i>					43,05
<i>Costes indirectos (2%)</i>					86,1
TOTAL PRECIO UNIDAD DE OBRA					4434,15

UO-4 *Construcción del convertidor de tensión.* *Soldadura de cables entre los componentes y regulación del convertidor.*

Código	Descripción	Unidades	Rendimiento	Precio (€)	Importe (€)
MO.GITI	Graduado en Tecnologías Industriales	Ud	1,5	20	30
M.CT	Módulo LM2596 Convertidor de Voltaje DC-DC Buck 1.25V-35V	Ud	1	9,9	9,9
M.PC	Placa de circuitos SODIAL(R) 4pcs Doble-Lado Prototipo FR-4 PCB Stripboard	Ud	1	2,42	2,42
M.RM	Regleta Clema enchufable macho recto 5,00mm - 2 contactos.	Ud	2	0,25	0,5
M.RH	Regleta Clema enchufable hembra acodada 5,00mm - 2 contactos	Ud	2	0,59	1,18
M.HE	Rollo de hilo de Estaño para electrónica 100gr	Ud	1	8,49	8,49
MA.SE	Soldador de estaño	h	1,5	4	6
MA.FA	Fuente de alimentación MAAS SPS-9600	h	1,5	5	7,5
MA.POL	Polímetro	h	1,5	1,3	1,95
<i>Total</i>					67,94
<i>Costes directos complementarios (1%)</i>					0,67
<i>Costes indirectos (2%)</i>					1,35
TOTAL PRECIO UNIDAD DE OBRA					69,97

UO-5 **Montaje y conexionado de los elementos** *Realización del montaje de los elementos en el quadrotor en la estructura y el conexionado entre ellos.*

Código	Descripción	Unidades	Rendimiento	Precio (€)	Importe (€)
MO.GITI	Graduado en Tecnologías Industriales	h	5	20	100
M.EST	Estructura del quadrotor	Ud	1	4615,96	0
M.TSD	Tarjeta micro-SD de 8GB de clase 10	ud	1	7,16	0
M.RPI2	Raspberry Pi 2	Ud	1	39,9	0
M.IMU	IMU MPU6050	Ud	1	18,9	18,9
M.CH	Raspberry Pi to Arduino Shields Connection Bridge	Ud	1	40	40
MA.OMG	Ordenador de media gama	h	5	8,7	43,5
M.MH	Cables macho hembra	ud	1	5	5
MA.FA	Fuente de alimentación MAAS SPS-9600	h	5	5	25
<i>Total</i>					232,4
<i>Costes directos complementarios (1%)</i>					2,32
<i>Costes indirectos (2%)</i>					4,64
TOTAL PRECIO UNIDAD DE OBRA					239,37

Desarrollo de un software de control basado en Raspberry Pi II para la implementación de algoritmos de control en una plataforma 3D (con 3 grados de libertad)

UO-6 Montaje del banco de pruebas *Preparación de elementos y montaje. Incluye el montaje 1 y 2*

Código	Descripción	Unidades	Rendimiento	Precio (€)	Importe (€)
<i>MO.GITI</i>	Graduado en Tecnologías Industriales	Ud	10	20	200
<i>M.MM</i>	Mástil: bloque de madera de 55 x 55 x 375 mm	Ud	1	4	4
<i>M.BM</i>	Base: tabla de madera de 237 x 400 x 15 mm	Ud	1	3	3
<i>M.EM</i>	Escuadras metálicas de 40 mm de alto con 4 agujeros	Ud	4	0,5	2
<i>M.TM</i>	Tornillos para madera de 4 mm de diámetro	Ud	20	0,1	2
<i>M.R1</i>	Sistema completo de cojinete de fricción con soporte más el perno esférico con rosca macho GFSM-08-AG	Ud	1	5,67	5,67
<i>M.R2</i>	Mini ball head MBH-M	Ud	1	29	29
<i>M.PM</i>	placa metálica de 50 x 50 x 1	Ud	1	1	1
<i>M.TN</i>	Tuerca Tee Nut	Ud	1	0,7	0,7
<i>M.PI</i>	Pegamento Instantáneo	Ud	1	7	7
<i>MA.TL</i>	Taladradora	h	1	5	5
<i>MA.LM</i>	Lima para madera	h	1	0,5	0,5
<i>Total</i>					259,87
<i>Costes directos complementarios (1%)</i>					2,59
<i>Costes indirectos (2%)</i>					5,19
TOTAL PRECIO UNIDAD DE OBRA					267,66

UO-7 *Ajuste del control* *Calibrado de la IMU y ajuste experimental del PID.*

<i>Código</i>	<i>Descripción</i>	<i>Unidades</i>	<i>Rendimiento</i>	<i>Precio (€)</i>	<i>Importe (€)</i>
<i>MO.GITI</i>	Graduado en Tecnologías Industriales	h	50	20	1000
<i>MA.OMG</i>	Ordenador de media gama	h	50	8,7	435
<i>MA.FA</i>	Fuente de alimentación MAAS SPS-9600	h	50	5	250
<i>Total</i>					1685
<i>Costes directos complementarios (1%)</i>					16.85
<i>Costes indirectos (2%)</i>					33.7
TOTAL PRECIO UNIDAD DE OBRA					1735.55

UO-8 *Realización del modelo en CAD y planos* *Modelo del quadrotor y del banco de pruebas en cad y planos de los componentes.*

<i>Código</i>	<i>Descripción</i>	<i>Unidades</i>	<i>Rendimiento</i>	<i>Precio (€)</i>	<i>Importe (€)</i>
<i>MO.GITI</i>	Graduado en Tecnologías Industriales	h	20	20	400
<i>MA.OMG</i>	Ordenador de media gama	h	20	8,7	174
<i>Total</i>					574
<i>Costes directos complementarios (1%)</i>					5,74
<i>Costes indirectos (2%)</i>					11,48
TOTAL PRECIO UNIDAD DE OBRA					591,22

2.3 Presupuesto de ejecución material, presupuesto de ejecución por contrata y presupuesto base de licitación.

<i>Código</i>	<i>Unidad de obra</i>	<i>Precio</i>
UO-1	Revisión del montaje de la estructura	4808,61
UO-2	Información previa y preparación del software necesario	1623,23
UO-3	Implementación de los algoritmos de control	4434,15
UO-4	Construcción del convertidor de tensión.	69,97
UO-5	Montaje y conexionado de los elementos	239,37
UO-6	Montaje del banco de pruebas	267,66
UO-7	Ajuste del control	1735,55
UO-8	Planos	591,22
	Presupuesto de ejecución material	13769,79
	<i>Gastos generales (15%)</i>	2065,47
	<i>Beneficio Industrial (6%)</i>	826,187
	Presupuesto de ejecución por contrata	16661,45
	<i>IVA (21%)</i>	3498,90
	Presupuesto base de licitación	20160,35

El presupuesto total de ejecución material asciende a:

TRECE MIL SETECIENTOS SESENTA Y NUEVE EUROS CON SESENTA Y NUEVE CÉNTIMOS.

El presupuesto de ejecución por contrata asciende a:

DIECISEIS MIL SEISCIENTOS SESNTA Y UN EUROS CON CUARENTA Y CINCO CÉNTIMOS.

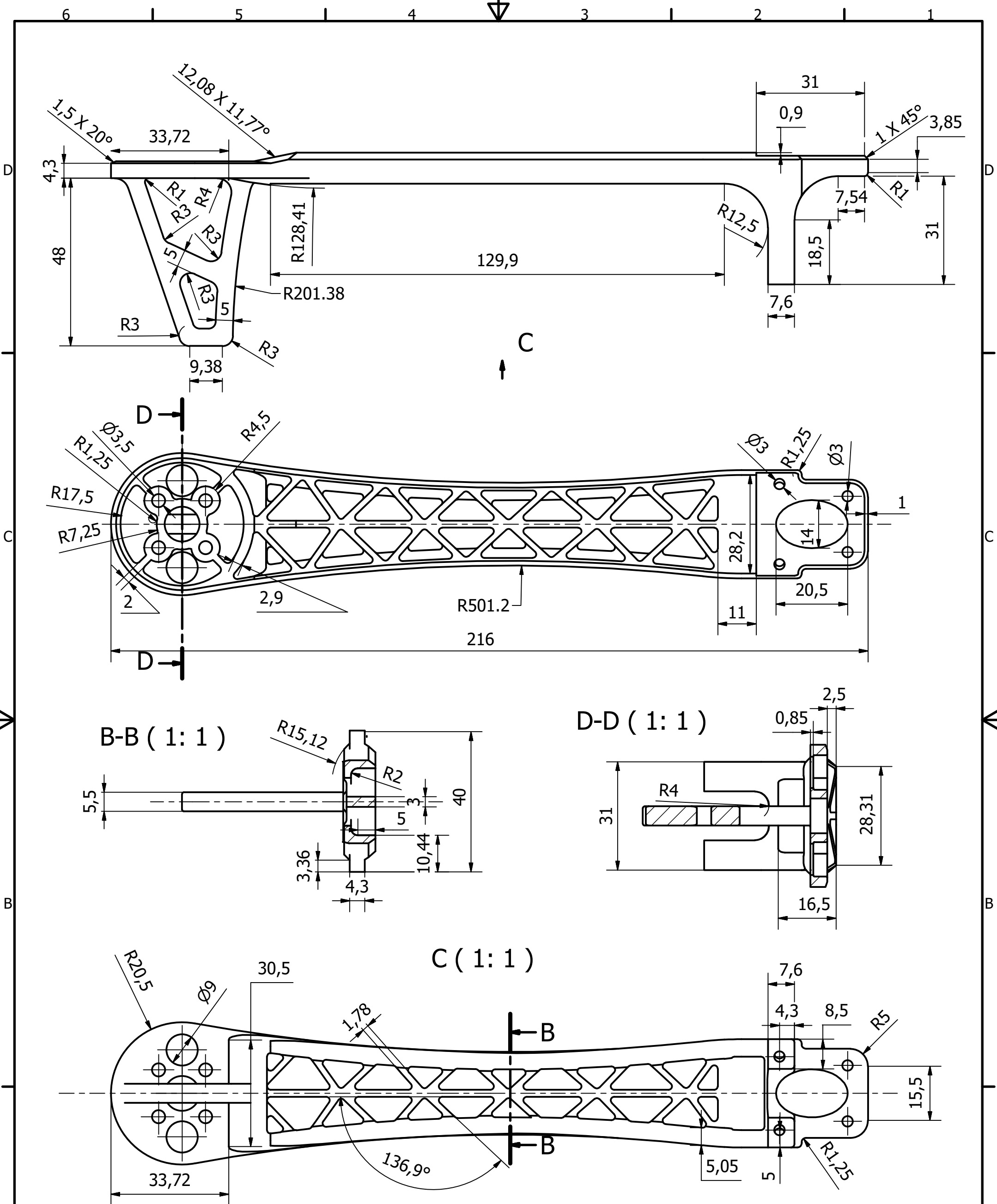
El presupuesto de base de licitación asciende a:

VEINTE MIL CIENTO SESENTA EUROS CON TREINTA Y CINCO CÉNTIMOS.

Planos

Contenido

- Plano 1: Brazo de estructura DJI flame wheel f450.
- Plano 2: Placa de potencia y placa de unión.
- Plano 3: Motor.
- Plano 4: Hélice y pieza de unión al motor.
- Plano 5: Tuerca Tee Nut M8, base, mástil y escuadra.
- Plano 6: Rótula 2: Mini Ball Head MBH-M.
- Plano 7: Banco de pruebas.
- Plano 8: Quadrotor.

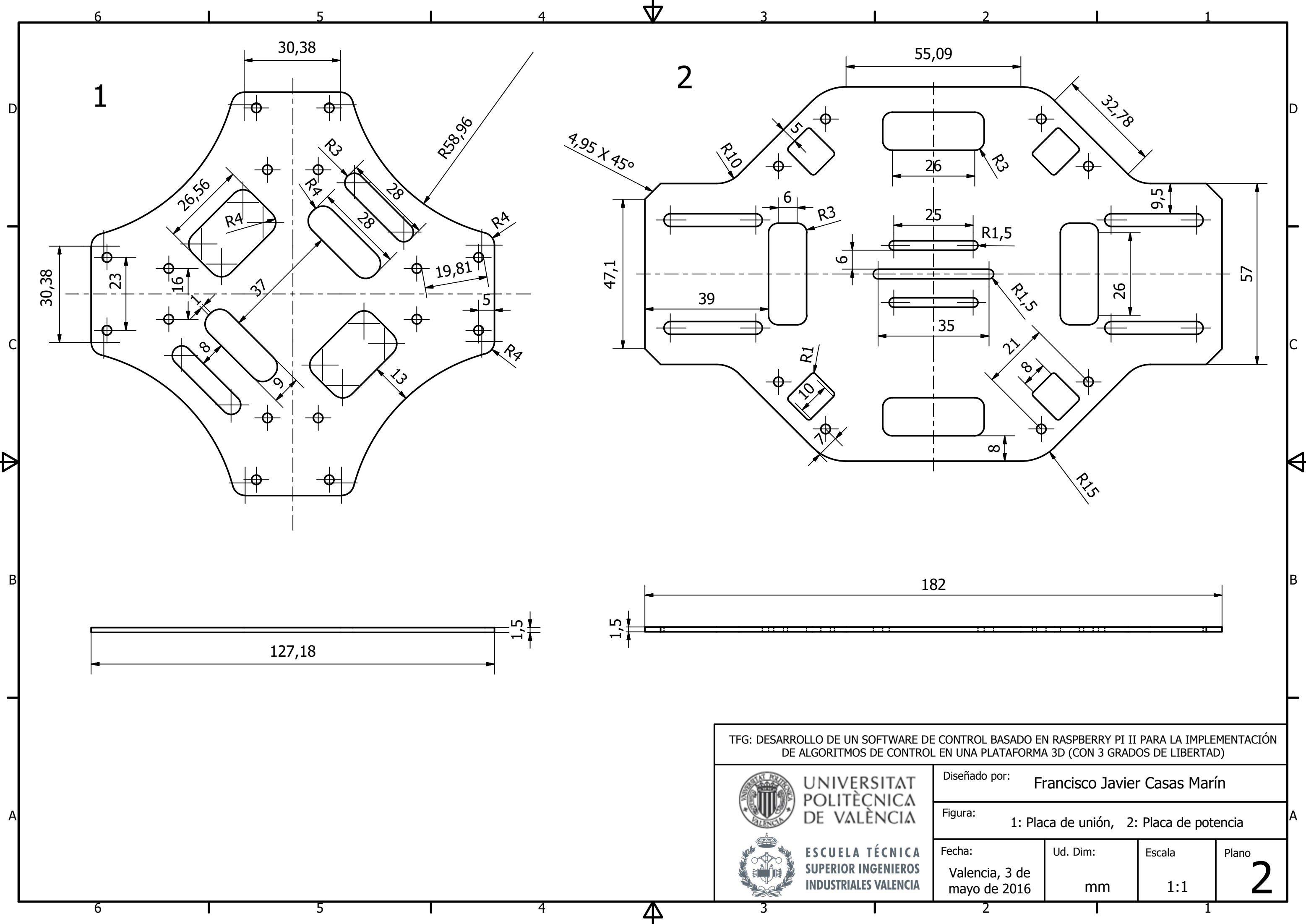


TFG: DESARROLLO DE UN SOFTWARE DE CONTROL BASADO EN RASPBERRY PI II PARA LA IMPLEMENTACIÓN DE ALGORITMOS DE CONTROL EN UNA PLATAFORMA 3D (CON 3 GRADOS DE LIBERTAD)


UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

Diseñado por: Francisco Javier Casas Marín			
Figura: Brazo de DJI Flamewheel Kit			
Fecha: Valencia, 3 de mayo de 2016	Ud. Dim: mm	Escala: 1:1	Plano: 1



TFG: DESARROLLO DE UN SOFTWARE DE CONTROL BASADO EN RASPBERRY PI II PARA LA IMPLEMENTACIÓN DE ALGORITMOS DE CONTROL EN UNA PLATAFORMA 3D (CON 3 GRADOS DE LIBERTAD)



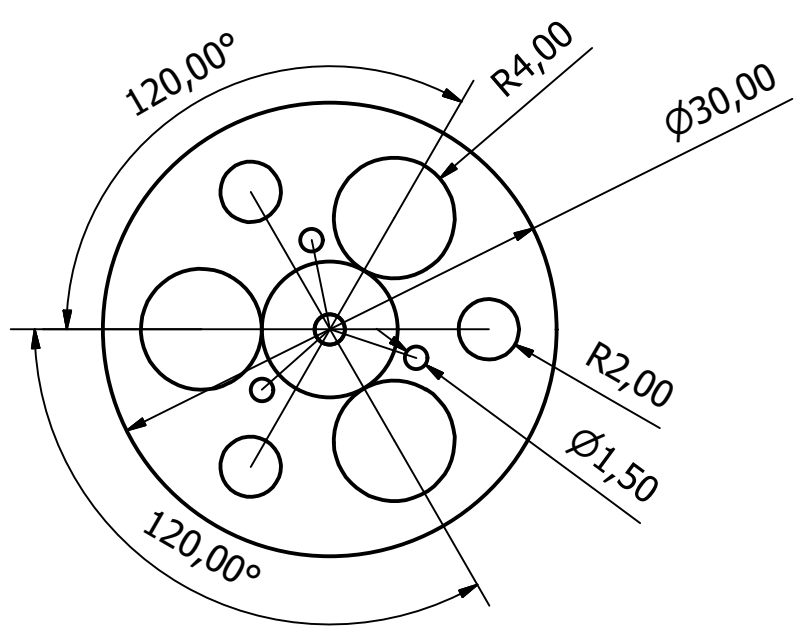
Diseñado por: **Francisco Javier Casas Marín**

Figura: **1: Placa de unión, 2: Placa de potencia**

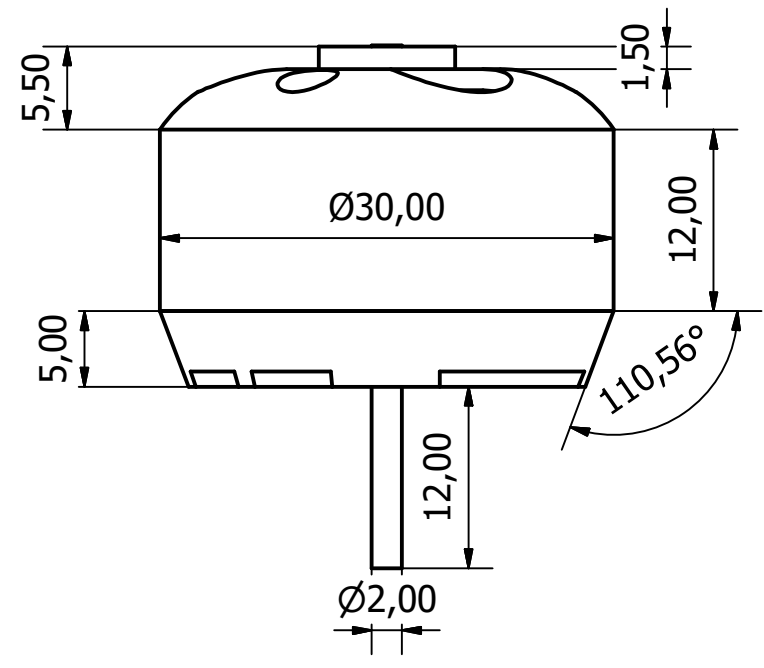
Fecha: Valencia, 3 de mayo de 2016	Ud. Dim: mm	Escala 1:1	Plano 2
---------------------------------------	----------------	---------------	-------------------

6 5 4 3 2 1

A (2 : 1)

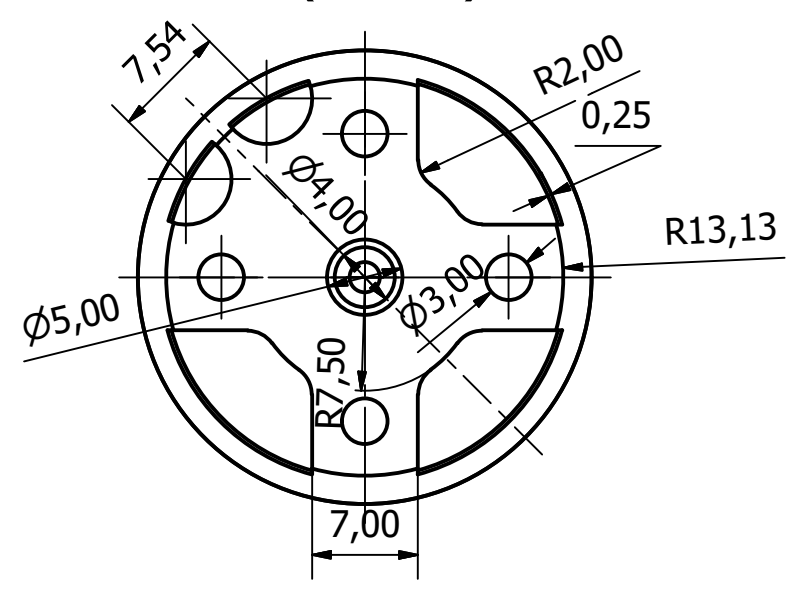


A ↓



↑ B

B (2 : 1)

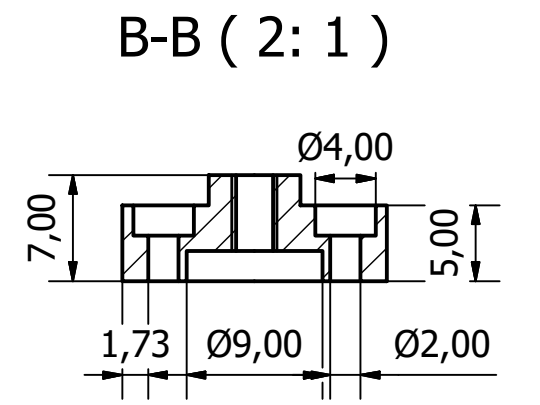
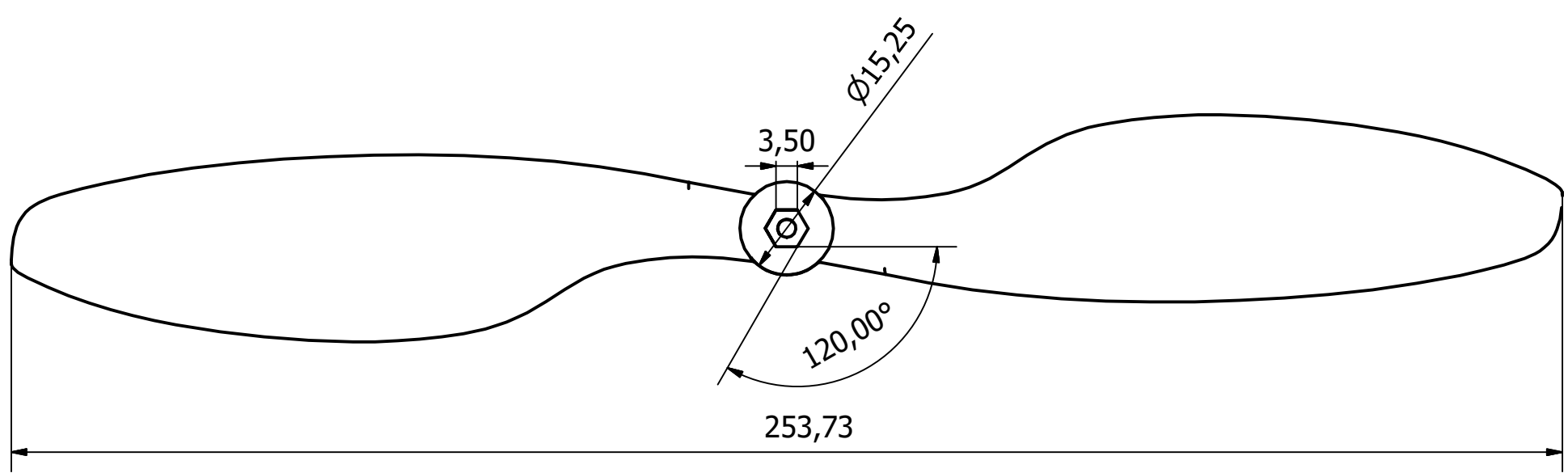
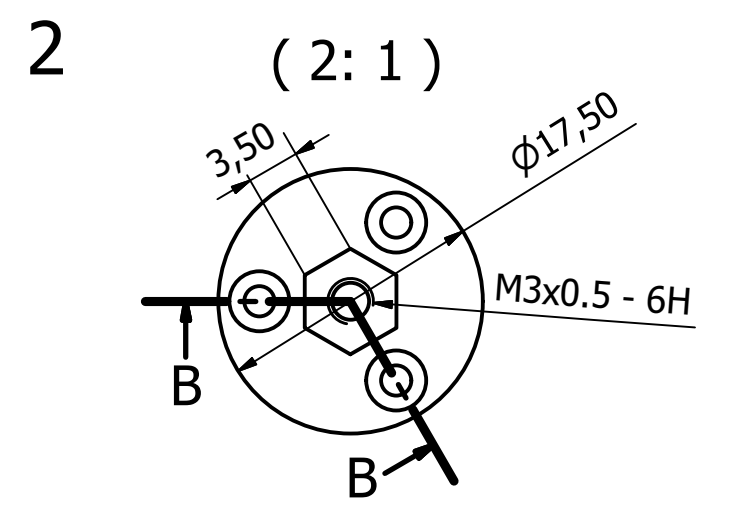
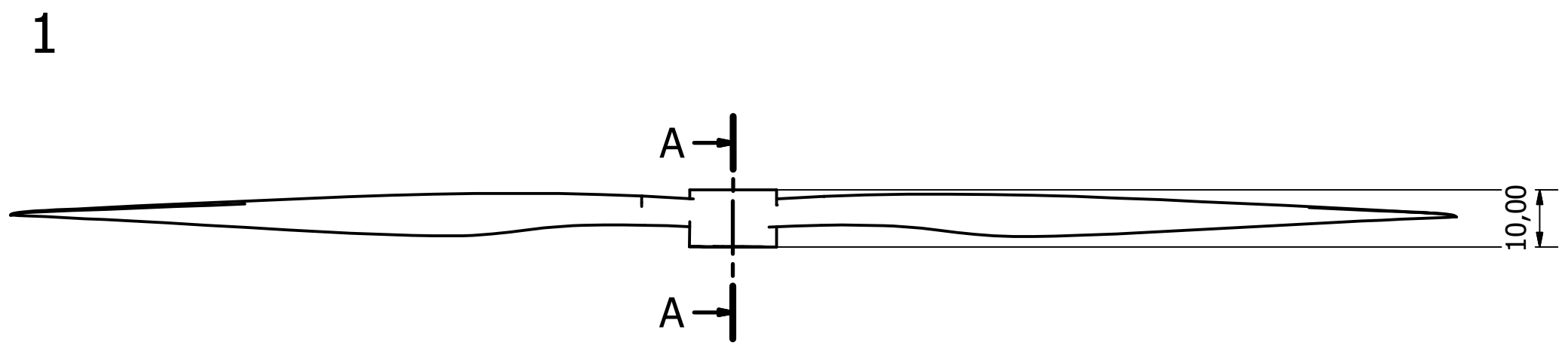


TFG: DESARROLLO DE UN SOFTWARE DE CONTROL BASADO EN RASPBERRY PI II PARA LA IMPLEMENTACIÓN DE ALGORITMOS DE CONTROL EN UNA PLATAFORMA 3D (CON 3 GRADOS DE LIBERTAD)

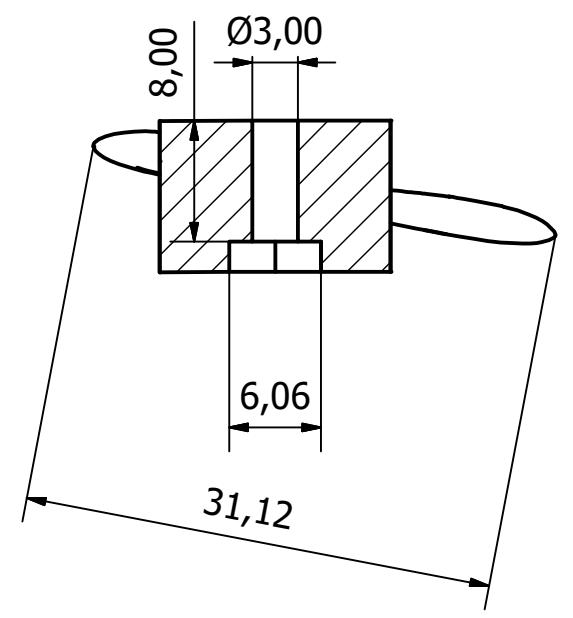
UNIVERSITAT POLITÈCNICA DE VALÈNCIA
ESCUOLA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

Diseñado por: Francisco Javier Casas Marín			
Figura: Motor Robbe Roxxy Brushless BL Outrunner 2824-34			
Fecha: Valencia, 3 de mayo de 2016	Ud. Dim: mm	Escala: 2:1	Plano: 3

6 5 4 3 2 1



A-A (2:1)



TFG: DESARROLLO DE UN SOFTWARE DE CONTROL BASADO EN RASPBERRY PI II PARA LA IMPLEMENTACIÓN DE ALGORITMOS DE CONTROL EN UNA PLATAFORMA 3D (CON 3 GRADOS DE LIBERTAD)

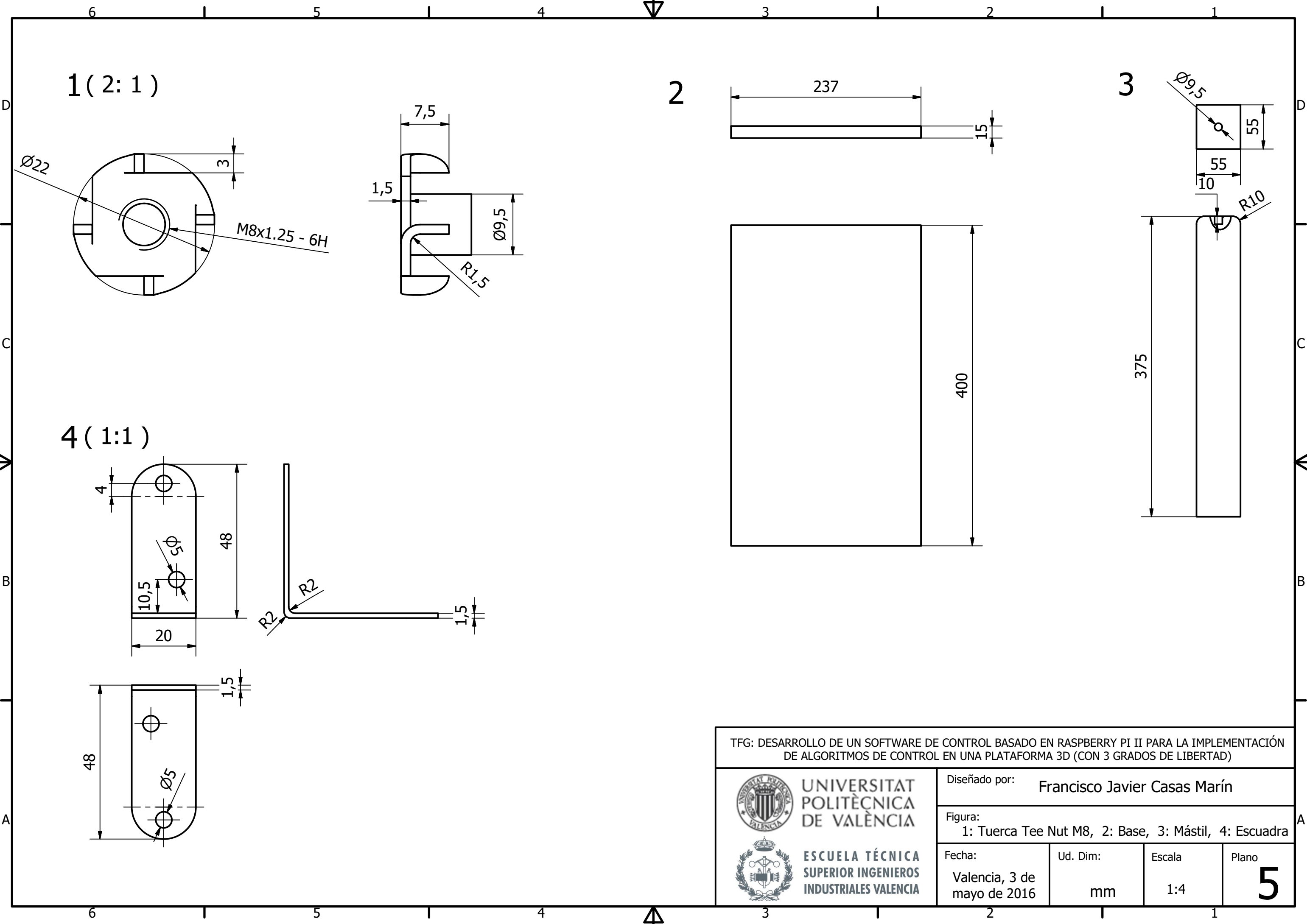

UNIVERSITAT POLITÈCNICA DE VALÈNCIA


ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

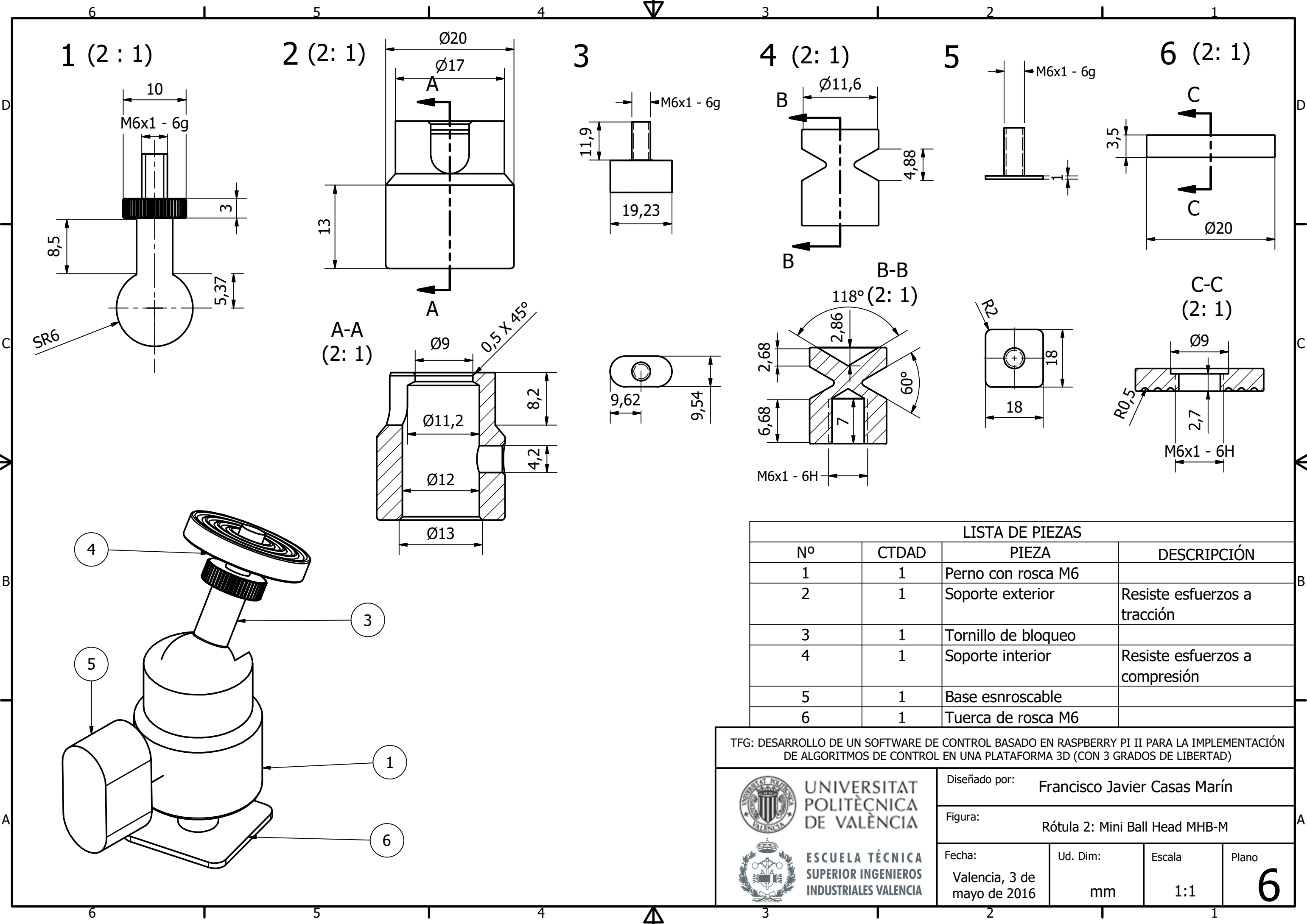
Diseñado por: **Francisco Javier Casas Marín**

Figura: **1: Hélice de 10x4.5", 2: Pieza unión hélice-motor**

Fecha: Valencia, 3 de mayo de 2016	Ud. Dim: mm	Escala 1:1	Plano 4
---------------------------------------	----------------	---------------	-------------------




TFG: DESARROLLO DE UN SOFTWARE DE CONTROL BASADO EN RASPBERRY PI II PARA LA IMPLEMENTACIÓN DE ALGORITMOS DE CONTROL EN UNA PLATAFORMA 3D (CON 3 GRADOS DE LIBERTAD)			
 UNIVERSITAT POLITÈCNICA DE VALÈNCIA		Diseñado por: Francisco Javier Casas Marín	
 ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA		Figura: 1: Tuerca Tee Nut M8, 2: Base, 3: Mástil, 4: Escuadra	
Fecha: Valencia, 3 de mayo de 2016	Ud. Dim: mm	Escala 1:4	Plano 5

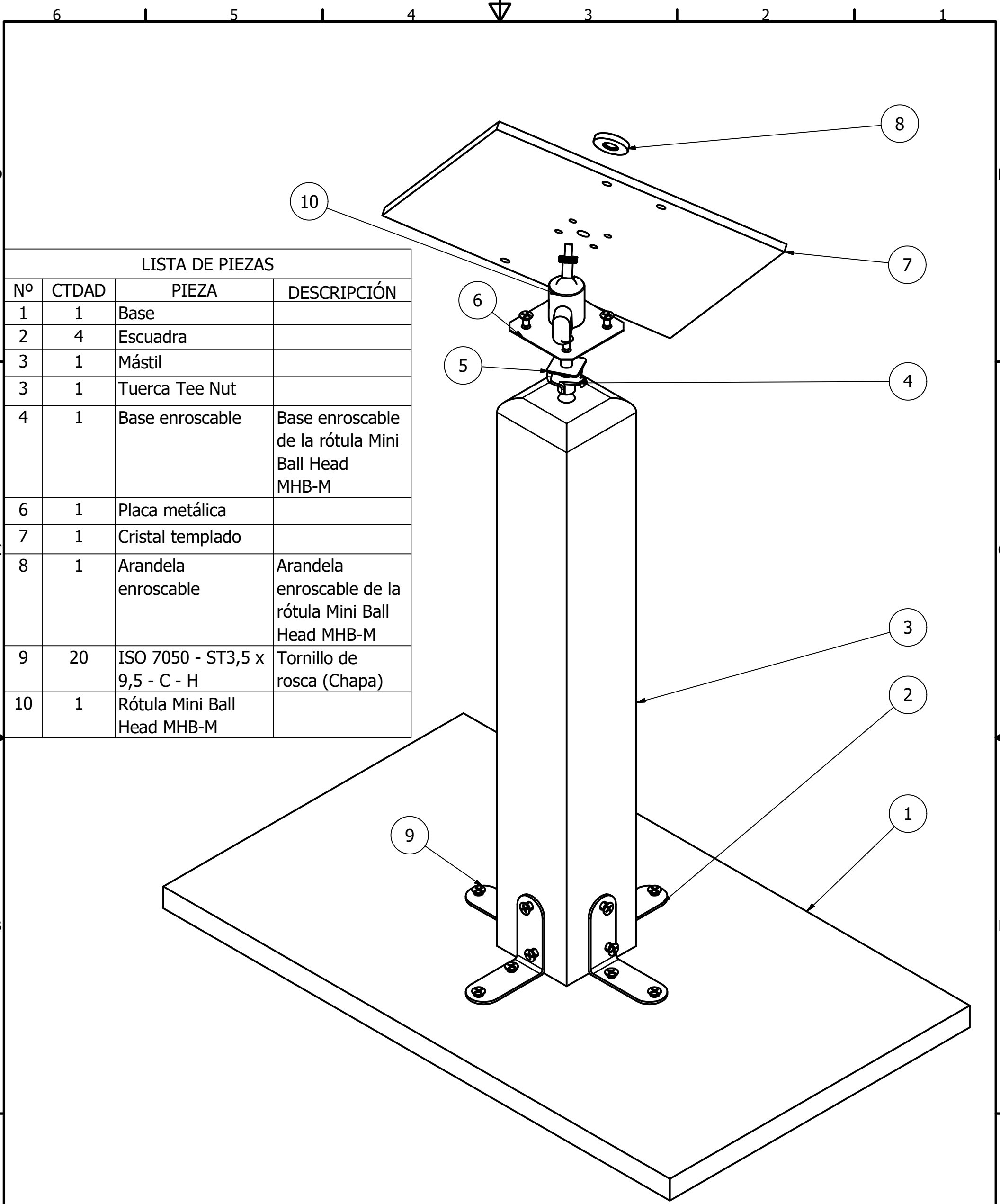


LISTA DE PIEZAS

Nº	CTDAD	PIEZA	DESCRIPCIÓN
1	1	Perno con rosca M6	
2	1	Soporte exterior	Resiste esfuerzos a tracción
3	1	Tornillo de bloqueo	
4	1	Soporte interior	Resiste esfuerzos a compresión
5	1	Base esnroscable	
6	1	Tuerca de rosca M6	

TFG: DESARROLLO DE UN SOFTWARE DE CONTROL BASADO EN RASPBERRY PI II PARA LA IMPLEMENTACIÓN DE ALGORITMOS DE CONTROL EN UNA PLATAFORMA 3D (CON 3 GRADOS DE LIBERTAD)

	Diseñado por: Francisco Javier Casas Marín		
	Figura: Rótula 2: Mini Ball Head MHB-M		
Fecha: Valencia, 3 de mayo de 2016	Ud. Dim: mm	Escala 1:1	Plano 6



LISTA DE PIEZAS

Nº	CTDAD	PIEZA	DESCRIPCIÓN
1	1	Base	
2	4	Escuadra	
3	1	Mástil	
3	1	Tuerca Tee Nut	
4	1	Base enroscable	Base enroscable de la rótula Mini Ball Head MHB-M
6	1	Placa metálica	
7	1	Cristal templado	
8	1	Arandela enroscable	Arandela enroscable de la rótula Mini Ball Head MHB-M
9	20	ISO 7050 - ST3,5 x 9,5 - C - H	Tornillo de rosca (Chapa)
10	1	Rótula Mini Ball Head MHB-M	

TFG: DESARROLLO DE UN SOFTWARE DE CONTROL BASADO EN RASPBERRY PI II PARA LA IMPLEMENTACIÓN DE ALGORITMOS DE CONTROL EN UNA PLATAFORMA 3D (CON 3 GRADOS DE LIBERTAD)



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA



ESCUELA TÉCNICA
SUPERIOR INGENIEROS
INDUSTRIALES VALENCIA

Diseñado por: **Francisco Javier Casas Marín**

Figura: **Banco de pruebas**

Fecha:
Valencia, 3 de
mayo de 2016

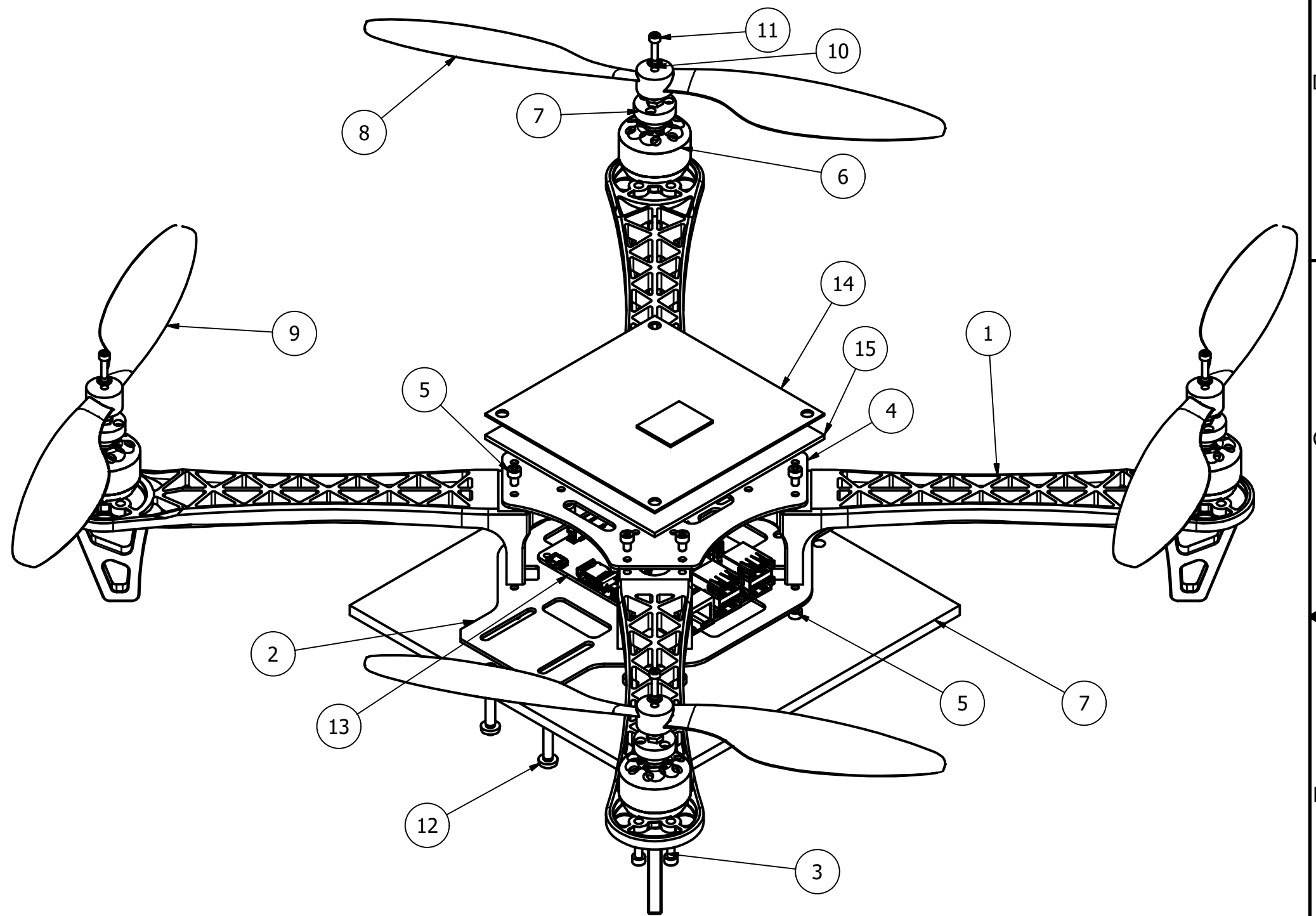
Ud. Dim:
mm

Escala
1:2


Plano

7

LISTA DE PIEZAS			
Nº	CTDAD	PIEZA	DESCRIPCIÓN
1	4	Brazo	Brazo de la estructura DJI Frame wheel f450
7	1	Cristal templado	Placa de union entre el quadrotor y la base
2	1	Placa de potencia	
3	16	DIN 912 - M3 x 8	Tornillo de cabeza cilíndrica para sujeción de los motores
4	1	Placa de unión	
5	24	DIN 912 - M3 x 5	Tornillo de cabeza cilíndrica para union de brazos a la placa de unión y a la placa de potencia
6	4	Motor	
7	4	Pieza soporte helice	
8	2	Hélice 10x4.5"	Hélice genérica de 10x4.5" de sentido antihorario
9	2	Hélice 10x4.5" R	Hélice genérica de 10x4.5" de sentido horario
10	4	ASME B18.21.2M - 3 - Tipo B	Arandelas de bloqueo (Serie métrica) Arandelas de bloqueo con dientes externos avellanadas - Tipo B.
11	4	ISO 4762 - M2,5 x 10	Tornillo de cabeza cilíndrica con hueco hexagonal
12	4	DIN 7985 (Z) - M4x20-Z	Tornillos de cabeza cilíndrica abombada con hueco cruciforme - Tipo Z
13	1	Raspberry Pi 2	
14	1	IMU MPU6050	
15	2	Placa de proteccion	Placa de plastico para proteger la IMU



TFG: DESARROLLO DE UN SOFTWARE DE CONTROL BASADO EN RASPBERRY PI II PARA LA IMPLEMENTACIÓN DE ALGORITMOS DE CONTROL EN UNA PLATAFORMA 3D (CON 3 GRADOS DE LIBERTAD)

	Diseñado por: Francisco Javier Casas Marín		
	Figura: Quadrotor		
Fecha: Valencia, 3 de mayo de 2016	Ud. Dim: mm	Escala: 1:2	Plano: 8

