UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

TRABAJO FIN DE GRADO EN INGENIERÍA AEROESPACIAL

Desarrollo de un vehículo aéreo remotamente tripulado (RPAS) para su uso en aplicaciones civiles

Autora: Celeste Vega Ródenas Antón

Escuela Técnica Superior de Ingeniería del Diseño
Universidad Politécnica de Valencia

cvroan@upv.es

Tutor: Pedro José García Gil

Departamento de Ingeniería de Sistemas y Automática
Universidad Politécnica de Valencia

pggil@isa.upv.es

Valencia, septiembre de 2015

RESUMEN

Este documento representa la memoria de un Trabajo de Final de Grado de la titulación Grado en Ingeniería Aeroespacial.

A lo largo de los distintos apartados que conforman la memoria se relata el trabajo realizado durante estos últimos meses, así como los problemas encontrados durante el desarrollo del proyecto y las soluciones adoptadas a fin de superarlos.

El objetivo de esta memoria es describir la plataforma RPAS (Remotely Piloted Aircraft System) desarrollada para su posterior aplicación en usos civiles, explicando su funcionamiento del modo más sencillo posible, tanto a nivel de hardware como de software, haciendo hincapié en los aspectos más cruciales del proyecto.

Dicha plataforma consiste en un quadrotor diseñado en dos capas de hardware y software: un mini PC que actúa como procesador principal, ejecutando el bucle de control y comunicándose con el resto de dispositivos, y un microcontrolador que obtiene las medidas de los sensores inerciales y envía las señales pertinentes a los motores.

Esto se ve complementado por una interfaz gráfica de usuario para ordenadores con sistema operativo Windows, que actúa como estación de tierra y permite dirigir la trayectoria del quadrotor mediante el uso de periféricos de entrada.

El objetivo principal del proyecto era el desarrollo de un quadrotor funcional, a fin de poner en práctica toda una serie de conocimientos relacionados con el ámbito de la aeronáutica, y adquiridos durante el transcurso de la carrera.

La plataforma se ha desarrollado partiendo de cero, por lo que se ha formado un equipo multidisciplinar para poder llevar a cabo todo el trabajo que ello supone. Así pues, durante el desarrollo del proyecto se ha trabajado tanto individualmente como de forma cooperativa, a fin de lograr los mejores resultados posibles.

Para poder validar el trabajo realizado y los algoritmos de control implementados, se han realizado diversos vuelos de prueba que se analizan en la presente memoria.

Agradecimientos

Sin duda alguna, este proyecto es un punto de inflexión en mi vida como estudiante. Hace cuatro años, no habría podido imaginar que en algún momento sería capaz de lograr algo así.

Es difícil encontrar palabras que expresen la emoción que supone empezar algo desde cero y ver cómo, poco a poco, tus ideas van tomando forma y lo que en un principio parecía imposible, empieza a dejar de serlo. Aún más difícil es expresar la emoción que se siente al ver que todo ese trabajo no ha sido en vano, y que estás cumpliendo tu objetivo, estás creando algo propio.

Sin embargo, este camino no lo he recorrido en solitario. Son muchas las personas que han estado ahí, de un modo u otro, apoyándome.

En primer lugar he de agradecer a mis padres, María Monserrate y José Faustino, haber hecho esto posible. No sólo me han apoyado incondicionalmente y escuchado en los momentos que más lo necesitaba, sino que además han trabajado todo lo que podían y más por darme la oportunidad de estudiar aquello que más deseaba.

También quiero dar las gracias a mis amigos de Orihuela, que hace cuatro años me vieron partir hacia Valencia por primera vez, pero han seguido ahí en todo momento, apoyándome, dando ánimos, haciendo que cada fin de semana de vuelta en el pueblo fuese una pequeña aventura y escuchando historias sobre aviones que no podía evitar contarles. Álvaro, Fabio, Gonzalo, José Miguel, María, Marta, Pablo… Esto no habría sido posible sin vosotros.

No me olvidaré tampoco de la gente que he conocido durante estos cuatro años y que han hecho del día a día algo más ameno. En especial, he de darle las gracias a aquellos que seguiré llamando amigos, incluso cuando nuestros caminos se separen definitivamente.

De todos ellos, quiero mencionar especialmente a mi compañero en este trabajo, Norberto. Ha sido un camino duro, hemos invertido muchas horas, y hemos dudado mucho, pero ver los frutos de todo ello, al final, ha sido la mejor recompensa posible. Gracias, Pedro, por habernos dado la oportunidad de llevar a cabo este proyecto y por haber hecho posible que, a día de hoy, nos sintamos algo más ingenieros.

Por último, quiero dar las gracias a aquellos que han hecho el camino algo más difícil, pues su ayuda ha sido indispensable para no perder la motivación y seguir intentando sobrepasar los límites, incluso en aquellos momentos en que parecía que no podía más.

En resumen, gracias.

Celeste Ródenas Antón

Índice de contenidos

Índice de figuras

1. Introducción

Este documento constituye la memoria de un Trabajo de Final de Grado de la titulación Grado en Ingeniería Aeroespacial, cursado en la Escuela Técnica Superior de Ingeniería del Diseño, en la Universidad Politécnica de Valencia desde el año 2011 hasta el año 2015.

En los últimos años, uno de los campos de mayor auge dentro del ámbito aeronáutico ha sido, sin duda alguna, el desarrollo de aeronaves del tipo UAV (siglas en inglés que se traducen como *Aeronaves no tripuladas*, también conocidas como drones). Si bien el interés militar en este tipo de aeronaves no es algo reciente, el uso civil de éstas se ha visto muy incrementado en la última década, siendo algunas de sus aplicaciones más comunes la fotografía aérea, la vigilancia o la inspección de infraestructuras, por citar algunas.

En dichas aplicaciones civiles, lo más común es que las aeronaves sean controladas a distancia, por un operador que actúa sobre una estación de tierra. En este caso, el término concreto es RPAS (*Aeronaves dirigidas por control remoto*). Este tipo de aeronaves incorporan algoritmos de control que le permiten tomar decisiones en vuelo a fin de conservar en todo momento la estabilidad, sin necesidad de un piloto a bordo que la controle. No obstante, hay una persona encargada en la estación de tierra en todo momento, que vigila el vuelo, dirige la trayectoria y monitoriza los datos que se envían a través del enlace de comunicaciones pertinente.

El objetivo de este proyecto ha sido el desarrollo de un sistema RPAS al completo para su posterior uso en aplicaciones civiles, en concreto un multirotor de cuatro hélices o quadrotor (también conocido como cuadricóptero), incluyendo la estación de tierra, junto a las comunicaciones necesarias.

Así pues, el trabajo realizado consiste en el desarrollo de dicho vehículo aéreo, incluyendo el diseño preliminar, la elección de componentes, el montaje de la estructura y conexiones y el desarrollo del software necesario, así como la creación de una interfaz de usuario que actúa como estación de tierra y permite el control de la trayectoria de la aeronave desde un ordenador.

Este proyecto se ha llevado a cabo durante el curso académico 2014/2015. Durante el primer semestre, de septiembre a enero, se llevó a cabo una labor de documentación previa, así como una primera parte del diseño preliminar. Durante el segundo semestre se concluyó dicha fase preliminar, procediendo a la elección y montaje de las piezas, y se iniciaron las labores de programación. Finalmente, durante los meses de junio y julio se procedió al desarrollo del software de la plataforma, el cual una vez operativo permitió comenzar las pruebas de vuelo así como el ajuste de los algoritmos de control.

Para el desarrollo de este proyecto se han dedicado aproximadamente unas 500 horas, cantidad que supera las 300 horas que corresponden, aproximadamente, a los 12 créditos ECTS. Una parte importante de este tiempo se ha invertido en la documentación previa, así como en la formación y aprendizaje en algunos temas nuevos, que se requerían para la consecución del proyecto. Además, los objetivos del proyecto han variado a lo largo de su desarrollo, lo que ha supuesto un mayor tiempo invertido.

Para la realización del proyecto ha sido necesario aplicar constantemente conocimientos y habilidades adquiridas a lo largo de la titulación, en ámbitos tan diversos como la informática, la electrónica, el control automático, las matemáticas o la mecánica. Además, ha sido necesario expandir dichos conocimientos aprendiendo, por ejemplo, nuevos lenguajes de programación y protocolos de comunicación, así como otros aspectos que no se daban en profundidad en las asignaturas correspondientes.

1.1. Conceptos previos

Para ayudar a la comprensión de la presente memoria, se definirá (por orden alfabético) una serie de conceptos con los que se ha trabajado durante el proyecto.

Acelerómetro: Instrumento que permite medir la aceleración a la que está sometido un sistema en una dirección concreta.

ESC: Regulador electrónico de velocidad (del inglés *electronic speed controller*). Dispositivo que transforma una señal PWM en una señal trifásica que se envía a un motor.

Giróscopo: Instrumento que permite medir la velocidad de giro de un sistema alrededor de un eje concreto.

HMI: Interfaz de usuario, interfaz hombre-máquina (del inglés *human-machine interface*). Software que permite la interacción entre el usuario y el dispositivo o máquina objetivo.

IDE: Entorno de desarrollo integrado (del inglés *integrated development enviroment*). Aplicación informática que proporciona servicios integrales para facilitar el desarrollo de software: editor de código, herramientas de construcción y depurador, por ejemplo.

IMU: Unidad de medida inercial (del inglés *inertial measurement unit*). Dispositivo que permite conocer la orientación de un sistema en el espacio mediante una combinación de acelerómetros y giróscopos.

Magnetómetro: Instrumento que permite medir la intensidad de campo magnético en una dirección concreta.

Microcontrolador: Circuito integrado programable, capaz de ejecutar las órdenes grabadas en su memoria. Consta de unidad central de procesamiento, memoria y periféricos de entrada/salida.

Motor brushless: Motor eléctrico sin escobillas. Motor eléctrico que no emplea escobillas para realizar el cambio de polaridad en el rotor, lo que mejora su comportamiento.

PID: Algoritmo de control proporcional integral derivativo. Mecanismo de control por realimentación ampliamente usado que calcula la desviación entre un valor medido y un valor deseado o referencia.

PWM: Modulación por ancho de pulsos (del inglés *pulse width modulation*). Técnica en la que se modifica el ciclo de trabajo de una señal periódica, ya sea para transmitir información a través de un canal de comunicaciones o controlar la cantidad de energía que se envía a una carga.

RPAS: Vehículo aéreo remotamente pilotado (del inglés *remotely piloted aircraft system*).

Quadrotor: Vehículo aéreo de cuatro rotores, destinados tanto a sustentación como propulsión. Por lo general, se sitúan en los extremos de una estructura en cruz. El movimiento del vehículo se logra variando la velocidad de giro de los motores en función de la trayectoria deseada.

UAV: Vehículo aéreo no tripulado (del inglés *unmanned aircraft vehicle*).

1.2. Motivaciones

El campo de los vehículos aéreos no tripulados ha experimentado un crecimiento muy importante en los últimos años. Se trata de un campo en auge, en el que se está invirtiendo dinero en investigación, gracias a las posibilidades y aplicaciones que ofrecen este tipo de sistemas.

Los sistemas multirotor, y en concreto los quadrotors, son, posiblemente, los sistemas RPAS más demandados para aplicaciones civiles. Esto se debe a su versatilidad, maniobrabilidad y estabilidad. Además, suelen ser sistemas relativamente económicos, lo que los convierte en un producto muy competitivo. Cada vez más empresas deciden incorporarse a este sector, lo que implica un gran aumento de la oferta de trabajo en este ámbito, por lo que la experiencia previa con este tipo de sistemas puede ser decisiva a la hora de conseguir un puesto de trabajo.

Por otra parte, en un terreno más subjetivo, la naturaleza práctica del proyecto también implica una gran motivación. Como estudiante de ingeniería, el hecho de diseñar un sistema partiendo de cero, desarrollarlo y lograr que funcione supone una gran satisfacción personal. El hecho de que el sistema, una vez completado, pueda llegar a volar no hace sino aumentar dicha satisfacción. Además, la naturaleza multidisciplinar del proyecto también constituye un atractivo importante.

Durante el desarrollo del proyecto, se ha profundizado en el lenguaje de programación C, y se ha iniciado el aprendizaje de otros, como C++ y C#, aprendiendo conceptos de programación concurrente y orientada a objetos, aspectos de la informática que no se habían contemplado a lo largo de la carrera, pero muy útiles a la hora de desarrollar sistemas aeronáuticos de este tipo.

También se ha profundizado en diversos aspectos de la electrónica y el control automático, imprescindibles para el desarrollo del proyecto, y se han llevado a la práctica conceptos aprendidos en materias como matemáticas o mecánica por ejemplo. Incluso se han puesto en

práctica algunas técnicas de fabricación necesarias para llevar a cabo el montaje de la plataforma.

1.3. Objetivos del proyecto

Durante una primera fase del desarrollo del proyecto, el objetivo fue el desarrollo de una nueva plataforma RPAS a partir de uno de los prototipos disponibles en el laboratorio, resultado de trabajos de años anteriores. Para ello, se iba a proceder a sustituir algunos de los componentes de la antigua plataforma por otros nuevos con los que el equipo del profesor tutor, Pedro García, no hubiese trabajado, a fin de explorar nuevas posibilidades y ampliar conocimientos.

Así pues, los objetivos iniciales eran:

- Sustituir el mini PC IGEP y el microcontrolador Arduino Micro del prototipo antiguo por un único mini PC BeagleBone Black.
- Modificar el código utilizado en las placas anteriores, para lograr adaptarlo a las características y necesidades de la BeagleBone Black.
- Probar nuevos sensores no utilizados anteriormente: nueva IMU y sensor de ultrasonidos.
- Implementar un control de orientación.
- Implementar un control de posición.
- Lograr el vuelo estable en actitud en interior.
- Lograr el vuelo robusto y fiable en interior.

Como objetivo adicional, en caso de que el tiempo lo permitiese, se contemplaba la posibilidad de adaptar la plataforma para el vuelo en exteriores, añadiendo un sistema GPS y un sensor de altura barométrico.

Para poder lograr los objetivos se partiría de una estructura ya montada, un código funcional en la IGEP que habría que adaptar a la Beagle y una HMI ya desarrollada y disponible en el laboratorio.

Sin embargo, durante la realización del proyecto se dieron una serie de circunstancias que llevaron a replantear los objetivos iniciales.

Por una parte, el otro estudiante involucrado en el desarrollo de la plataforma, Norberto (Vera Vélez, 2015), trabajó durante el primer semestre del curso con el microcontrolador STM32F3 *Discovery* en la asignatura de *Sistemas embarcados para navegación y control*. Esta placa de bajo coste incluye una IMU integrada, la cual se aprendió a utilizar durante el desarrollo de dicha asignatura. Por ello, se decidió incorporar dicho microcontrolador a la plataforma.

Por otra parte, se dio la circunstancia de que, debido a la cantidad de estudiantes que estaban trabajando en el mismo laboratorio, finalmente no sería posible utilizar ninguna de las estructuras ya montadas.

A la vista de los hechos se decidió replantear el proyecto y proceder al desarrollo de una nueva plataforma RPAS partiendo de cero, tanto a nivel de hardware como de software. Esto suponía una carga extra de trabajo que implicaba tener que abandonar parte de los objetivos iniciales.

Así pues, los objetivos del proyecto serían:

- Diseño preliminar de la plataforma en dos capas, haciendo uso de las placas STM y Beagle ya mencionadas, selección de piezas, montaje de la estructura y conexión de los componentes.
- Desarrollo del software necesario para la lectura de la IMU e implementación del PWM en la STM.
- Desarrollo del software necesario para el control, el filtrado de medidas y cálculo de posición y la comunicación con los dispositivos en la Beagle.
- Desarrollo de una HMI que permitiese el control de la trayectoria del quadrotor mediante módulos de radiofrecuencia.
- Implementación de un control de orientación.
- Implementación de un control de posición.
- Vuelo estable en actitud en interior.
- Vuelo robusto y fiable en interior.

A lo largo de posteriores apartados de la presente memoria se relatan los procedimientos llevados a cabo a fin de lograr dichos objetivos, así como los problemas encontrados durante su desarrollo y las soluciones adoptadas, en caso de ser posible.

1.4. Estructuración de la memoria

A continuación se enumeran los apartados que contiene esta memoria, describiéndolos brevemente.

1. Introducción

Se da una visión general del proyecto. Se explican unos conceptos necesarios para la comprensión de éste, las motivaciones para embarcarse en él y los principales objetivos que se querían lograr.

2. Estado del arte de los quadrotor

Se hace una breve reseña de la historia de la aviación no tripulada, se define el concepto de quadrotor y se muestran las tendencias actuales en el mercado, así como las técnicas de control más utilizadas.

3. Fundamentos matemáticos de la plataforma quadrotor

Se explican las bases matemáticas de las que parte este proyecto. Por una parte, se explican las ecuaciones que rigen el comportamiento del quadrotor. Por otra, se explican las ecuaciones

que permiten la obtención de la actitud a partir de las medidas de la IMU y los algoritmos de filtrado utilizados.

4. Diseño preliminar de la plataforma

Se relata el proceso de diseño y concepción del borrador de la plataforma previo a la fase de montaje y desarrollo de software, dando una visión global del funcionamiento de ésta.

5. Hardware de la plataforma

Se describen uno a uno de los elementos que integran la plataforma, así como sus principales características y detalles de interés.

6. Desarrollo de la plataforma

Se describe el proceso de montaje de la plataforma, así como la conexión entre los distintos componentes que la forman, la alimentación y los protocolos de comunicación utilizados.

7. Software de la plataforma

Se describe el software desarrollado durante el proyecto, tanto en el quadrotor como en la estación de tierra.

8. Control de la plataforma

Se describen los algoritmos de control utilizados, así como las medidas necesarias para ejecutarlos, y el proceso de ajuste experimental de éstos.

9. Resultados experimentales

Se exponen y comentan los resultados obtenidos durante la realización de diversas pruebas de vuelo.

10. Conclusiones finales y trabajos futuros

Se comentan las conclusiones que se extraen tras el desarrollo del proyecto y se plantean posibles trabajos futuros para mejorar y completar la plataforma.

11. Presupuesto

Se expone el presupuesto aproximado de la plataforma desarrollada.

12. Bibliografía y referencias

Se citan las referencias utilizadas durante el desarrollo del proyecto.

2. Estado del arte de los quadrotor

2.1. Reseña histórica

En sus orígenes, el desarrollo de aeronaves no tripuladas estuvo ligado, esencialmente, al ámbito militar. El primer uso documentado de este tipo de aeronaves data de 1849, cuando Austria atacó la ciudad italiana de Venecia con globos no tripulados, cargados de explosivos. Este sistema, desarrollado durante meses, contaba con serias desventajas. Por ejemplo, los cambios de viento podían alterar la trayectoria del globo, haciendo que algunos explotasen en las líneas austríacas, en lugar del objetivo. Si bien este diseño difiere bastante de la idea de UAV que se puede tener hoy en día, conceptos como este sembraron un interés que permitirían nuevos desarrollos durante el siglo XX.

Durante la Primera Guerra Mundial (1914-1918), los avances tecnológicos, tales como la invención del giróscopo estabilizador, permitieron el desarrollo de los primeros aviones no tripulados. Diseños como el Hewitt-Sperry *Automatic Airplane* (ver Figura 1), de 1916, sirvieron para demostrar el concepto. Concebidos como "torpedos aéreos", constituyen un antecesor de lo que actualmente se conoce como misil de crucero. No obstante, la tecnología de la época era limitada, y la autonomía de vuelo de estos diseños muy baja, por lo que no fueron utilizados durante el conflicto bélico. Posteriormente, en el periodo de entreguerras, los primero drones controlados por radio entrarían en servicio militar.

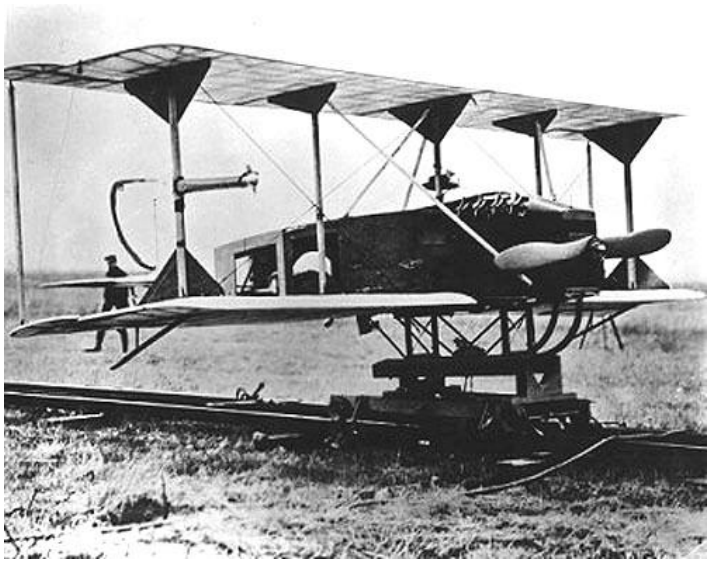

Figura 1: Hewitt-Sperry Automatic Airplane

La Segunda Guerra Mundial (1939-1945) supuso el inicio del uso de estas aeronaves en conflictos bélicos. El desarrollo tecnológico destinado a la guerra fue muy importante en ambos bandos, lo que llevó a la concepción de importantes diseños. La Alemania nazi utilizó en varias ocasiones aeronaves V1 *Buzz Bomb* (ver Figura 2), que contaban con un motor de tipo pulsorreactor y una autonomía de unas 150 millas. El bando aliado, por otra parte, contaba con diseños como el *Radioplane* OQ-2 (ver Figura 3), utilizado comúnmente como dron

objetivo para el entrenamiento de pilotos. Además, también se intentó desarrollar versiones no tripuladas de aeronaves como el B-17 *Flying Fortress* o el B-24 *Liberator*, dando lugar a diseños como el PB4Y-1, si bien su éxito fue muy limitado.

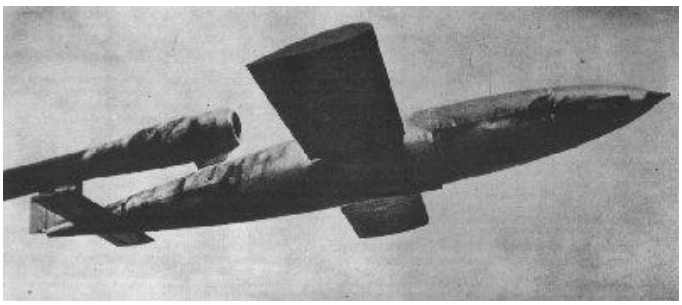

Figura 2: V1 Buzz Bomb

Figura 3: Radioplane OQ-2

El avance en campos como el almacenamiento de energía, el procesamiento de datos o el desarrollo de nuevos sensores fue imprescindible para la evolución de este campo de la aeronáutica. Además, la escalada de tensión que supuso la Guerra Fría en la segunda mitad del siglo XX, hizo que ambos bandos tuvieran un gran interés en interceptar comunicaciones enemigas y fotografiar los enclaves militares del rival.

Así pues, durante la guerra de Vietnam (1955-1975) cobra mucha importancia el concepto de dron de reconocimiento, primándose el desarrollo de aeronaves para labores de espionaje tras las líneas enemigas. Entre los diseños más destacados de la época se encuentran los Ryan *Firebees* (ver Figura 4) de la serie AQM-34, que solían lanzarse desde aeronaves como el Lockheed C-130 *Hercules*.

Figura 4: Ryan Firebee

Algunos otros diseños importantes desarrollados durante la Guerra Fría son el Ryan Model 147 *Lightning Bug* o el Lockheed D-21 (ver Figura 5). Este último era un dron de reconocimiento supersónico, que contaba con pintura antirradar y tenía una gran autonomía de vuelo. Inicialmente, fue diseñado para lanzarse desde la aeronave Lockheed M-21, aunque posteriormente se modificó para poder lanzarse desde aviones Boeing B-52 *Stratofortress*.

Figura 5: Lockheed D-21 montado sobre M-21

Entre la década de los 80 y la de los 90, la tecnología aplicable a los UAV no sólo fue madurando y refinándose, sino que además redujo su tamaño. Esto aumentó enormemente el interés en los drones, ya que cada vez se podía hacer aeronaves de menor tamaño y mayor autonomía, lo que favorecía en gran medida sus usos militares. Uno de los diseños más destacados de esta época es, sin duda, el General Atomics MQ-1 *Predator* (ver Figura 6). Esta aeronave, todavía en servicio, fue concebida a principios de los 90 para labores de reconocimiento aéreo, por lo que cuenta con toda una serie de cámaras y sensores para dicha tarea. No obstante, en versiones posteriores ha sido modificada para incorporar armas y convertirse en un dron de combate.

Figura 6: General Atomics MQ-1 Predator

En la actualidad, el interés por las aeronaves no tripuladas no se limita sólo al ámbito militar. Desde los años 90, el uso de estas aeronaves con fines comerciales o civiles ha ido creciendo paulatinamente. Así pues, hoy en día es común ver drones que se dedican a tareas tan variadas como la fotografía aérea, la vigilancia aérea, la inspección de infraestructuras o incluso labores de rescate.

2.2. Concepto de quadrotor

Las aeronaves pilotadas por control remoto, o RPAS, se enmarcan en el ámbito de los vehículos aéreos no tripulados. Este término, como ya se ha expuesto en la introducción, hace referencia a aeronaves semiautónomas que, si bien no requieren de un piloto que las dirija a bordo, cuentan en todo momento con un operador en una estación en tierra que monitoriza el vuelo y dirige la trayectoria de la aeronave. La nave, no obstante, cuenta con sistemas de control automático que se encargan de mantener un vuelo estable y realizar las correcciones necesarias para recorrer la trayectoria deseada de manera óptima.

El concepto de RPAS engloba aeronaves de todo tipo, desde algunas muy parecidas a lo que se podría concebir como un vehículo aéreo convencional, a otras tipologías exclusivas a estos sistemas no tripulados, que exprimen al máximo la ventaja de no necesitar un piloto a bordo tanto en su forma como en sus actuaciones. Así pues, al igual que con las aeronaves tradicionales es posible encontrar sistemas RPAS tanto de despegue convencional como de despegue vertical.

Dentro de los sistemas RPAS de despegue vertical, y en concreto en un subgrupo de aeronaves de ala rotativa, se enmarcan las aeronaves de tipo multirotor. Un multirotor es un RPAS que cuenta con más de dos hélices o rotores, por lo general, en un mismo plano. El número de hélices de este tipo de sistemas suele oscilar entre 3 y 16, dependiendo tanto el número de éstas como su configuración del propósito y la aplicación de la aeronave.

La Figura 7 muestra un multirotor de 8 hélices asimétrico de la empresa alemana Ascending Technologies, muy bien posicionada en el sector de la grabación de imágenes con sistemas multirotor así como en el desarrollo de éstos.

En la Figura 8 se puede ver otro multirotor de 8 hélices, esta vez simétrico, desarrollado a partir de una plataforma Spreading Wings S1000, de la empresa china DJI, muy popular en el sector de los multirotor de gama media dedicados a la fotografía y filmación.

Figura 7: Multirotor asimétrico Ascending Technologies Falcon 8

Figura 8: Multirotor simétrico DJI Spreading Wings S1000

La aeronave que se ha desarrollado durante este proyecto es un quadrotor, que se engloba dentro del grupo de RPAS ya mencionado. Un quadrotor (o cuadricóptero en castellano) es un tipo concreto de multirotor que cuenta con cuatro motores dispuestos, por lo general, en

forma de cruz, y que controla su orientación y, a través de ella su posición, variando la velocidad de giro de éstos según sea la actuación deseada. Así pues, al contrario que en otras aeronaves de ala rotativa como los helicópteros, el ángulo del rotor así como el de las hélices permanece constante, y las traslaciones y giros se logran variando el empuje de los motores de forma diferencial, lo que implica una menor complejidad de fabricación.

En general, la estructura de los quadrotors suele ser rígida. Los diseños más convencionales de este tipo de sistemas son cruciformes, aunque no es raro encontrar algunos con estructura en forma de H, por ejemplo, sobre todo en aquellos en que la disposición de los motores no presenta simetría radial. La estructura cuenta con unos brazos a los que van anclados los motores y un tren de aterrizaje, que en ocasiones se sustituyen por soportes bajo los motores que permiten reducir el volumen de la estructura. Los componentes electrónicos, por lo general, se alojan en la zona central del sistema. El modelo DJI Phantom 2 (ver Figura 9) representa una disposición tradicional de quadrotor.

Figura 9: Quadrotor DJI Phantom 2

No obstante, se pueden encontrar modelos en los que la estructura presenta articulaciones, por ejemplo, para retraer el tren de aterrizaje o incluso mover los brazos de los motores. Este es el caso del modelo DJI Inspire 1 (ver Figura 10). Este quadrotor cuenta con unas articulaciones que permiten variar la forma de la estructura. Así cuando se encuentra en tierra los brazos están inclinados hacia abajo, y actúan a modo de tren de aterrizaje. Por otra parte, una vez en vuelo, las articulaciones giran y los motores se desplazan hacia arriba, lo que permite un mayor campo de visión para la cámara que lleva equipada, sin interferencias de ningún elemento estructural de la plataforma.

Así pues, cabe esperar la innovación en gran cantidad de aspectos de los sistemas quadrotor, con el desarrollo de nuevas plataformas pensadas para aplicaciones concretas que se alejen de la idea más convencional a fin de adaptarse de la mejor forma posible al fin para el que se han concebido.

Figura 10: Quadrotor DJI Inspire 1

Las principales ventajas de los quadrotors, que hacen que sean sistemas tan populares en el ámbito de los RPAS, son:

- Gran maniobrabilidad. Su tamaño y peso reducidos, así como sus cuatro motores, hacen que los quadrotors puedan dar respuestas rápidas y agresivas a fin de variar su trayectoria. Esto los hace ideales para llegar a lugares de difícil acceso.
- Gran estabilidad. Son capaces de lograr un vuelo a punto fijo muy estable lo que es imprescindibles para la toma de fotografías o filmación de calidad
- Son una de las opciones más económicas dentro del ámbito de los RPAS lo que los hace atractivos tanto a nivel profesional, como para usuarios con fines de ocio.

No obstante, cuentan con una limitación importante, que es su autonomía de vuelo. El uso de motores eléctricos, que es lo más común en estos sistemas, los hace depender de baterías cuya duración es escasa si se quiere mantener un compromiso con el peso de la plataforma. Incluso los quadrotors de gama alta no suelen ser capaces de volar durante más de media hora, lo que hace que para ciertas aplicaciones no puedan competir con otras tipologías de RPAS.

Aun así, las ventajas ya mencionadas los hacen ideales para una infinidad de aplicaciones, gracias a la gran versatilidad que presentan. Entre otras:

- Aplicaciones civiles: Labores de búsqueda y rescate tanto marítimas como en montaña; monitorización de carreteras, infraestructuras, oleoductos, zonas protegidas o contaminación atmosférica, por ejemplo; reconocimiento y gestión de desastres; aplicaciones topográficas…
- Aplicaciones comerciales: Fotografía y filmación tanto en eventos como para fines artísticos; labores agrícolas; vigilancia aérea…
- Aplicaciones militares: Reconocimiento y espionaje; interceptación de comunicaciones; transporte de carga bélica…

El abanico de posibilidades es amplio y también lo es el de plataformas disponibles en el mercado, que van desde pequeños drones para fines de ocio que cuestan algo menos de 100€, hasta modelos profesionales de gama alta como los que ofrecen empresas como Draganfly o Service-Drone, como por ejemplo el Draganflyer X4 (ver Figura 11), el cual incorpora una estructura de fibra de carbono, así como una cámara de alta calidad y soporte gimbal, estando su precio en torno a los 8000€.

Figura 11: Quadrotor Draganflyer X4

2.3. Control de un quadrotor

Los quadrotors son los multirotors más usados en labores de investigación y desarrollo, así como plataforma de pruebas para diversos algoritmos de control. Su menor precio, tamaño y complejidad los hacen más atractivos respecto a las alternativas de más hélices.

Los primeros quadrotors funcionales se controlaban de manera manual, sin la asistencia de sistemas automáticos, lo que hacía imprescindible la habilidad del piloto para lograr un vuelo medianamente estable y seguro. Sin embargo, en la actualidad esto ya no es un aspecto crucial, puesto que prácticamente todos los quadrotors vuelan de forma asistida, mediante sistemas de control automático. Es decir, si bien el piloto dirige la trayectoria del sistema, ya no ha de preocuparse de mantener un vuelo estable, pues la plataforma de control se encarga de ello.

Estos sistemas de control requieren una serie de datos obtenidos de distintos sensores. Estos sensores pueden ir embarcados en la plataforma (IMUs, sensores barométricos, ultrasonidos, sistema GPS…) o bien ser externos a ella (sistema de cámaras externas que establezcan la posición del quadrotor, por ejemplo). Los algoritmos implementados en el sistema procesan dichos datos y generan las acciones de control pertinentes sobre los motores, permitiendo el vuelo estable.

Dichos algoritmos de control se suelen procesar a bordo del quadrotor, por lo que es necesario contar con dispositivos electrónicos con una elevada capacidad de procesamiento (una opción popular en la actualidad son los mini PC basados en arquitectura ARM). También es posible ejecutarlos de forma remota, en ordenadores de mayor potencia, aligerando el peso de la plataforma, a la que se ha de enviar las acciones de control de forma inalámbrica. Estos algoritmos se ejecutan de forma periódica, generalmente con un período del orden de milisegundos, lo que permite obtener velocidades de respuesta mucho mayores que con un piloto humano, lo que se traduce en un vuelo mucho más estable.

A pesar de esto, puede resultar difícil dirigir la trayectoria del quadrotor mientras se cumple también la misión a la que se destina. Por ejemplo, en el caso de la fotografía aérea es común la actuación de dos operadores, uno que pilote la aeronave y otro que se encargue de la cámara. Así, el piloto podrá mantener el contacto visual con la aeronave y evitar cualquier situación anómala en vuelo, como la presencia de un obstáculo.

Por ello, la tendencia actual es, generalmente, la de realizar de forma autónoma la mayor cantidad de procesos posibles, a fin de facilitar la tarea del piloto. Esto incluye la automatización de maniobras como el despegue y el aterrizaje, la detección y evasión de obstáculos o el seguimiento de trayectorias mediante waypoints o también objetivos móviles, por ejemplo.

Todo esto no hace sino mostrar que los algoritmos de control son una parte esencial de los quadrotors en la actualidad, y parte importante del origen del éxito e interés con el que cuentan estos sistemas.

Algunas de las técnicas de control propuestas para quadrotors autónomos en los últimos años son:

- Conmutación de distintos reguladores lineales según el punto de funcionamiento (por ejemplo, mediante algoritmos PID).
- Reguladores basados en un índice cuadrático LQR, con una posible extensión considerando ruidos gaussianos LQG.
- Control por modos deslizantes, especialmente enfocado al cálculo de trayectorias.
- Control robusto combinado esquemas PID para actitud y control H-infinito para posición.
- Estrategias de control avanzadas: algoritmos de aprendizaje, leyes adaptativas, redes neuronales…

No obstante, muchas de estas estrategias avanzadas se encuentran aún en fase de desarrollo y, aunque pueden llegar a mejorar el comportamiento del quadrotor en algunas situaciones, la complejidad y el aumento de la necesidad de capacidad de procesamiento pueden llegar a contrarrestar las ventajas que suponen.

Así pues, todavía en la actualidad la estrategia de control más utilizada es la de los algoritmos PID, o en todo caso modificaciones de éstos, ya que ofrecen una sencillez, bajo coste computacional y facilidad de ajuste difíciles de igualar. Sin embargo, es posible que con el

tiempo cambien las tendencias, y el desarrollo de nuevas tecnologías permita implementar algoritmos más avanzados sin aumentar los tiempos de ejecución en los sistemas de control.

3. Fundamentos matemáticos de la plataforma quadrotor

A continuación, como introducción teórica al proyecto, se presentan los modelos matemáticos más relevantes que se han tenido en cuenta previamente al desarrollo de la plataforma quadrotor. Estos modelos no sólo facilitan la comprensión del funcionamiento de este tipo de sistemas sino que, en algunos casos, son imprescindibles para el desarrollo posterior del software necesario para que la plataforma esté operativa.

Así pues, se explicará tanto un modelo matemático de la dinámica de un quadrotor como distintos procedimientos para el cálculo de la orientación del sistema a partir de una IMU. El interés de estos modelos matemáticos radica en su importancia a la hora de diseñar de manera efectiva el bucle de control.

El procesado de las medidas de los sensores es necesario para el cálculo de la actitud del quadrotor y la comparación de ésta con el valor de referencia deseado en el bucle de control. Por otra parte, la comprensión del modelo teórico permite establecer sobre qué motores han de actuar las acciones de control que proporcionen los algoritmos.

3.1. Modelo teórico de un quadrotor

En este apartado, se desarrolla el modelo de comportamiento teórico de un quadrotor genérico. Dicho modelo es conocido, y es posible encontrar una amplia bibliografía al respecto, con explicaciones muy detalladas y desarrollos matemáticos en profundidad. No obstante, a continuación se mostrará un desarrollo sencillo, que permite conocer las ecuaciones que rigen el movimiento de este tipo de sistemas RPAS.

El conocimiento del modelo dinámico teórico del sistema es importante puesto que:

- Permite conocer de antemano cuál va a ser el comportamiento aproximado del quadrotor, a fin de poder decidir las estrategias de control apropiadas.
- Puede servir como plataforma de simulación para la prueba y ajuste de los sistemas de control, sin necesidad de recurrir al prototipo físico.

La obtención de dicho modelo puede llevarse a cabo mediante dos métodos distintos que conducen, obviamente, al mismo resultado. Por una parte, es posible aplicar un enfoque lagrangiano de la física, calculando las leyes de movimiento a partir de las ecuaciones de Euler-Lagrange[1]. Por otra parte, se puede aplicar un enfoque newtoniano al problema en cuestión. Puesto que para el propósito de esta demostración se puede considerar el quadrotor como un

[1] Las ecuaciones de Euler-Lagrange se plantean para un sistema descrito por sus coordenadas generalizadas y, en su forma más general, presentan la siguiente forma:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = 0$$

donde L es el lagrangiano del sistema y q_i las coordenadas generalizadas del sistema.

sólido rígido, se describirá tanto el movimiento de traslación como el de rotación mediante las ecuaciones de Newton-Euler[2]. Se ha optado por este enfoque ya que:

- Permite visualizar de manera más clara cuáles son las fuerzas y los momentos a los que está sometido el quadrotor, y cómo influyen en el movimiento.
- Implica el trabajo con matrices de rotación que se pueden usar posteriormente para el procesamiento de las medidas de los sensores, en caso de ser necesario.

La plataforma quadrotor se puede plantear como un sólido rígido libre en el espacio. Esto se traduce en que tendrá 6 grados de libertad: $(x, y, z, \phi, \theta, \psi) \in \mathbb{R}^6$, donde (x, y, z) representa la posición del centro de masas del sistema y (ϕ, θ, ψ) su orientación en el espacio. El ángulo ϕ, conocido como *roll* o ángulo de alabeo, representa el giro sobre el eje x. El ángulo θ, conocido como *pitch* o ángulo de cabeceo, representa el giro sobre el eje y. Por último, el ángulo ψ, conocido como *yaw* o ángulo de guiñada, representa el giro sobre el eje z.

La Figura 12 presenta una estructura quadrotor genérica, incluyendo las correspondientes velocidades angulares, así como los pares y las fuerzas inducidos por los cuatro rotores (numerados de 1 a 4).

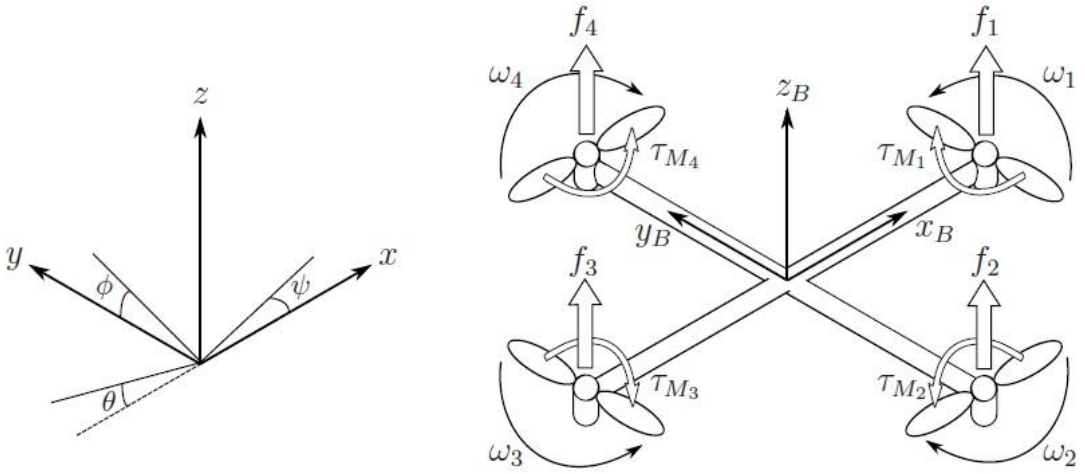

Figura 12: Sistemas de referencia inercial y ligado al cuerpo en un quadrotor

[2] Las ecuaciones de Newton-Euler, para un sistema de referencia cuyo origen coincide con el centro de masas (c.d.m. en adelante) de cuerpo en cuestión, se pueden expresar en forma matricial como:

$$\begin{pmatrix} F \\ \tau \end{pmatrix} = \begin{bmatrix} m\, I_3 & 0 \\ 0 & I_{cm} \end{bmatrix} \begin{pmatrix} a_{cm} \\ \alpha \end{pmatrix} + \begin{pmatrix} 0 \\ \omega \times I_{cm}\, \omega \end{pmatrix}$$

donde F es la fuerza total que actúa sobre el c.d.m., m la masa del cuerpo, I_3 la matriz identidad, a_{cm} la aceleración del c.d.m., τ el par total que actúa respecto el c.d.m., I_{cm} el momento de inercia respecto el c.d.m., ω la velocidad angular del cuerpo y α la aceleración angular del cuerpo.

La posición absoluta del quadrotor se define en el sistema de referencia inercial de ejes x, y, z con el vector ξ. En cuanto a la actitud del quadrotor, se define, de nuevo en los mismos ejes, con el vector η, que incluye los ángulos mencionados con anterioridad. El vector q incluye tanto la posición como la orientación del sistema.

$$\xi = \begin{pmatrix} x \\ y \\ z \end{pmatrix}, \qquad \eta = \begin{pmatrix} \phi \\ \theta \\ \psi \end{pmatrix}, \qquad q = \begin{pmatrix} \xi \\ \eta \end{pmatrix} \tag{1}$$

El origen del sistema de referencia ligado al cuerpo se encuentra en el centro de masas del quadrotor. En los ejes cuerpo, las velocidades lineales están determinadas por V_B y las velocidades angulares por ν.

$$V_B = \begin{pmatrix} v_{x,B} \\ v_{y,B} \\ v_{z,B} \end{pmatrix}, \qquad \nu = \begin{pmatrix} p \\ q \\ r \end{pmatrix} \tag{2}$$

La matriz de rotación de ejes cuerpos al sistema inercial es:

$$R = \begin{bmatrix} C_\theta C_\psi & S_\phi S_\theta C_\psi - C_\phi S_\psi & C_\phi S_\theta C_\psi + S_\phi S_\psi \\ C_\theta S_\psi & S_\phi S_\theta S_\psi + C_\phi C_\psi & C_\phi S_\theta S_\psi - S_\phi C_\psi \\ -S_\theta & S_\phi C_\theta & C_\phi C_\theta \end{bmatrix} \tag{3}$$

donde $S_\alpha = \sin\alpha$ y $C_\alpha = \cos\alpha$. La matriz de rotación R es ortogonal por lo que $R^{-1} = R^T$, siendo esta la matriz de rotación de sistema inercial a ejes cuerpo.

La matriz de transformación para las velocidades angulares desde el sistema inercial al ligado al cuerpo es W_η, siendo su inversa la que permite la transformación contraria.

$$\begin{aligned} \dot{\eta} = W_\eta^{-1}\,\nu, \qquad & \begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} = \begin{bmatrix} 1 & S_\phi T_\theta & C_\phi T_\theta \\ 0 & C_\phi & -S_\phi \\ 0 & \frac{S_\phi}{C_\theta} & \frac{C_\phi}{C_\theta} \end{bmatrix} \begin{pmatrix} p \\ q \\ r \end{pmatrix} \\ \nu = W_\eta\,\dot{\eta}, \qquad & \begin{pmatrix} p \\ q \\ r \end{pmatrix} = \begin{bmatrix} 1 & 0 & -S_\theta \\ 0 & C_\phi & S_\phi C_\theta \\ 0 & -S_\phi & C_\phi C_\theta \end{bmatrix} \begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} \end{aligned} \tag{4}$$

donde $T_x = \tan x$. La matriz W_η es invertible si $\theta \neq \frac{(2k-1)\phi}{2}, (k \in \mathbb{Z})$.

Se asume que el quadrotor posee una estructura simétrica con los cuatro brazos alineados con los ejes x e y del sistema de referencia ligado al cuerpo. Así, la matriz de inercia I es una matriz diagonal donde $I_{xx} = I_{yy}$.

$$I = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{5}$$

La velocidad angular del rotor i, denotada por ω_i , crea una fuerza f_i en la dirección del eje del rotor. Además, la velocidad angular y aceleración del rotor crean un par τ_{M_i} alrededor, también, de dicho eje.

$$f_i = k\omega_i^2 \,, \qquad \tau_{M_i} = b\omega_i^2 + I_M\dot{\omega}_i \tag{6}$$

donde k es una constante que representa la relación entre la velocidad angular del rotor y la sustentación que genera, b es una constante que representa la relación entre la misma velocidad angular y la resistencia que se genera, e I_M es el momento de inercia del rotor. Por lo general, el efecto de $\dot{\omega}_i$ es pequeño, por lo que suele omitirse.

La combinación de las fuerzas de los distintos rotores crea un empuje T en la dirección del eje z cuerpo. El par τ_B, por otra parte, consiste en los pares τ_ϕ, τ_θ y τ_ψ en la dirección de los ángulos correspondientes en el sistema de referencia ligado al cuerpo. La expresión de dichos pares puede deducirse a partir del esquema mostrado en la Figura 1.

$$T = \sum_{i=1}^{4} f_i = k\sum_{i=1}^{4} \omega_i^2 \,, \qquad T_B = \begin{pmatrix} 0 \\ 0 \\ T \end{pmatrix} \tag{7}$$

$$\tau_B = \begin{pmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{pmatrix} = \begin{pmatrix} l\,k\,(-\omega_2^2 + \omega_4^2) \\ l\,k\,(-\omega_1^2 + \omega_3^2) \\ \sum_{i=1}^{4} \tau_{M_i} \end{pmatrix} \tag{8}$$

donde l es la distancia entre el rotor y el centro de masa del quadrotor.

Se puede ver que, para el modelo planteado, el movimiento de *roll* se logra disminuyendo la velocidad de giro del motor 2 y aumentando la del número 4, o viceversa. De manera similar, el movimiento de *pitch* se logra disminuyendo la velocidad de giro del motor 1 y aumentando la del número 3, o viceversa. El movimiento de *yaw*, por su parte, se logra aumentando la velocidad angular de dos motores no contiguos, y disminuyendo la de los dos restantes.

En cuanto a la fuerza de la gravedad, se puede definir de forma sencilla en el sistema de referencia inercial, siendo su expresión:

$$G = -mg \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{9}$$

donde m representa la masa del quadrotor y g la aceleración de la gravedad.

Una vez establecida la cinemática del modelo, así como las expresiones de las fuerzas externas, se puede obtener el modelo dinámico, mediante las ecuaciones de Newton-Euler, considerando el quadrotor como un sólido rígido.

En primer lugar, se plantea el equilibrio de fuerzas en el sistema de referencia ligado al cuerpo. La resultante de las fuerzas será la suma de la acción debida a la gravedad (que se debe rotar debidamente) y el empuje que inducen los rotores. Además, como se trata de un sistema de

referencia no inercial, se debe considerar un término ficticio adicional de fuerzas de inercia (generalmente conocido como fuerza centrífuga), a fin de poder aplicar las leyes de Newton. Así pues, la ecuación de equilibrio en cuestión es:

$$m\,\dot{V}_B + \nu \times (m\,V_B) = R^T G + T_B$$

$$\begin{pmatrix} \dot{v}_{x,B} \\ \dot{v}_{y,B} \\ \dot{v}_{z,B} \end{pmatrix} = \begin{pmatrix} r\,v_{y,B} - q\,v_{z,B} \\ p\,v_{z,B} - r\,v_{x,B} \\ q\,v_{x,B} - p\,v_{y,B} \end{pmatrix} + g \cdot \begin{pmatrix} S_\theta \\ -S_\phi C_\theta \\ -C_\phi C_\theta \end{pmatrix} + \frac{T}{m} \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{10}$$

En el caso del sistema de referencia inercial, no es necesario considerar ningún término ficticio, por lo que sólo intervienen tanto la fuerza gravitacional como el empuje, quedando la expresión tal como sigue:

$$m\,\ddot{\xi} = G + R\,T_B$$

$$\begin{pmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{pmatrix} = -g \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} + \frac{T}{m} \cdot \begin{pmatrix} C_\phi S_\theta C_\psi + S_\phi S_\psi \\ C_\phi S_\theta S_\psi - S_\phi C_\psi \\ C_\phi C_\theta \end{pmatrix} \tag{11}$$

Para plantear el equilibrio de momentos en el sistema de referencia ligado al cuerpo, de nuevo se debe considerar un término inercial adicional. Además, también se tiene en cuenta los efectos giroscópicos inducidos por el giro de los rotores, que quedan recogidos en el término Γ. Así pues, la ecuación de equilibrio es:

$$I\,\dot{\nu} + \nu \times (I\,\nu) + \Gamma = \tau_B$$

$$\dot{\nu} = I^{-1}\left(-\begin{pmatrix} p \\ q \\ r \end{pmatrix} \times \begin{pmatrix} I_{xx}\,p \\ I_{yy}\,q \\ I_{zz}\,r \end{pmatrix} - I_r \cdot \begin{pmatrix} p \\ q \\ r \end{pmatrix} \times \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \cdot \omega_\Gamma + \tau_B \right)$$

$$\begin{pmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{pmatrix} = \begin{pmatrix} \frac{I_{yy} - I_{zz}}{I_{xx}}\,q\,r \\ \frac{I_{zz} - I_{xx}}{I_{yy}}\,p\,r \\ \frac{I_{xx} - I_{yy}}{I_{zz}}\,p\,q \end{pmatrix} - I_r \cdot \begin{pmatrix} \frac{q}{I_{xx}} \\ -\frac{p}{I_{yy}} \\ 0 \end{pmatrix} \cdot \omega_\Gamma + \begin{pmatrix} \frac{\tau_\phi}{I_{xx}} \\ \frac{\tau_\theta}{I_{yy}} \\ \frac{\tau_\psi}{I_{zz}} \end{pmatrix} \tag{12}$$

donde $\omega_\Gamma = \omega_1 - \omega_2 + \omega_3 - \omega_4$ e I_r el momento de inercia de los rotores.

Las aceleraciones angulares en el sistema de referencia inercial se pueden calcular a partir de las de ejes cuerpo mediante la inversa de la matriz W_η de transformación y su derivada respecto del tiempo:

$$\ddot{\eta} = \frac{d}{dt}\left(W_\eta^{-1}\,\nu\right) = \frac{d}{dt}\left(W_\eta^{-1}\right)\nu + W_\eta^{-1}\,\dot{\nu} \tag{13}$$

si bien las expresiones que se derivan de este cálculo son extensas y, además, no son imprescindibles para la descripción del modelo.

Así pues, las Ecuaciones 10 y 12 sumadas a las relaciones cinemáticas descritas con anterioridad describen, por completo, el movimiento de la plataforma quadrotor genérica estudiada. La figura 13 agrupa las ecuaciones que rigen el modelo dinámico del quadrotor en el sistema de referencia ligado al cuerpo. Se aprecia que es un sistema de ecuaciones diferenciales no lineales que, en un principio, puede ser difícil de tratar a la hora de aplicar leyes de control.

$$\dot{v}_{x,B} = r\, v_{y,B} - q\, v_{z,B} + g \sin\theta$$
$$\dot{v}_{y,B} = p\, v_{z,B} - r\, v_{x,B} - g \sin\phi \cos\theta$$
$$\dot{v}_{z.B} = q\, v_{x,B} - p\, v_{y,B} - g \cos\phi\ cos\theta + \frac{T}{m}$$
$$\dot{p} = \frac{I_{yy} - I_{zz}}{I_{xx}}\, q\, r - \frac{I_r}{I_{xx}}\, q\, \omega_\Gamma + \frac{\tau_\phi}{I_{xx}}$$
$$\dot{q} = \frac{I_{zz} - I_{xx}}{I_{yy}}\, p\, r + \frac{I_r}{I_{yy}}\, p\, \omega_\Gamma + \frac{\tau_\theta}{I_{yy}}$$
$$\dot{r} = \frac{I_{xx} - I_{yy}}{I_{zz}}\, p\, q + \frac{\tau_\psi}{I_{zz}}$$

Figura 13: Modelo dinámico del quadrotor en sistema de referencia ligado al cuerpo

El modelo mostrado describe el comportamiento íntegro del quadrotor. No obstante, en la práctica se van a dar una serie de situaciones que van a permitir simplificar el modelo. Así pues se pueden admitir las siguientes hipótesis a modo de simplificación:

1. Ángulos pequeños: Los ángulos de inclinación no deberían sobrepasar los 20 grados durante un funcionamiento normal de la plataforma.
2. Velocidades angulares pequeñas: Los términos derivados de las fuerzas de inercia pueden asumirse nulos.
3. Términos giroscópicos despreciables.

La primera hipótesis permite establecer la siguiente relación entre las velocidades angulares en el sistema de referencia inercial y las correspondientes en ejes cuerpo:

$$\begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} = \begin{bmatrix} 1 & S_\phi T_\theta & C_\phi T_\theta \\ 0 & C_\phi & -S_\phi \\ 0 & \frac{S_\phi}{C_\theta} & \frac{C_\phi}{C_\theta} \end{bmatrix} \begin{pmatrix} p \\ q \\ r \end{pmatrix} \simeq \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} p \\ q \\ r \end{pmatrix} = \begin{pmatrix} p \\ q \\ r \end{pmatrix} \quad (14)$$

Las otras dos hipótesis, por su parte, permiten definir las aceleraciones en el sistema de referencia ligado al cuerpo del siguiente modo:

$$\begin{pmatrix} \dot{v}_{x,B} \\ \dot{v}_{y,B} \\ \dot{v}_{z,B} \end{pmatrix} = g \cdot \begin{pmatrix} S_\theta \\ -S_\phi C_\theta \\ -C_\phi C_\theta \end{pmatrix} + \frac{T}{m} \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \quad (15)$$

$$\begin{pmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{pmatrix} = \begin{pmatrix} \frac{\tau_\phi}{I_{xx}} \\ \frac{\tau_\theta}{I_{yy}} \\ \frac{\tau_\psi}{I_{zz}} \end{pmatrix} \tag{16}$$

Así pues, a partir de las Ecuaciones 14 y 16 se puede establecer que:

$$\begin{pmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{pmatrix} = \begin{pmatrix} \frac{\tau_\phi}{I_{xx}} \\ \frac{\tau_\theta}{I_{yy}} \\ \frac{\tau_\psi}{I_{zz}} \end{pmatrix} \tag{17}$$

Por otra parte, la aplicación de la Ecuación 15 y su posterior rotación nos llevaría al resultado ya descrito en la Ecuación 11.

$$\ddot{x} = (\cos\phi \sin\theta \cos\psi + \sin\phi \sin\psi)\frac{T}{m}$$
$$\ddot{y} = (\cos\phi \sin\theta \sin\psi - \sin\phi \cos\psi)\frac{T}{m}$$
$$\ddot{z} = (\cos\phi \cos\theta)\frac{T}{m} - g$$
$$\ddot{\phi} = \frac{\tau_\phi}{I_{xx}}$$
$$\ddot{\theta} = \frac{\tau_\theta}{I_{yy}}$$
$$\ddot{\psi} = \frac{\tau_\psi}{I_{zz}}$$

Figura 14: Modelo dinámico simplificado del quadrotor en sistema de referencia inercial

La Figura 14 muestra el conjunto de las ecuaciones simplificadas del quadrotor. Estas ecuaciones son más intuitivas y permiten abordar el diseño de las estrategias de control de manera sencilla.

3.2. Obtención de la orientación a partir de una IMU

En una plataforma quadrotor, el conocimiento de la orientación del sistema en todo momento es un factor crucial. Una mayor precisión a la hora de calcular su actitud se traduce en un mejor funcionamiento del sistema de control, el cual podrá calcular de manera correcta la desviación entre los ángulos de referencia establecidos y la actitud del quadrotor en cada ejecución, generando acciones de control efectivas, que permitan lograr la estabilidad deseada. Así pues, es imprescindible contar con sensores que nos permitan conocer el estado del sistema.

En este apartado, se desarrollan procedimientos para el cálculo de la orientación a partir de tres tipos de sensores embarcados: acelerómetros, giróscopos y magnetómetros. Estos sensores nos proporcionan información acerca de aceleraciones, velocidades angulares y campo magnético, respectivamente.

Para indicar la actitud del quadrotor, se recurre a los ángulos ya descritos en la sección anterior, que son: el ángulo de alabeo o *roll* ϕ (giro en x), el ángulo de cabeceo o *pitch* θ (giro en y) y el ángulo de guiñada o *yaw* ψ (giro en z).

Los ángulos de alabeo y cabeceo se pueden calcular fácilmente a partir de las lecturas que proporciona el acelerómetro, utilizando las proyecciones del vector gravedad en el plano x-y. Si se denomina $F = (F_x, F_y. F_z)$ al vector de fuerzas de gravedad, se tiene que:

$$\phi = arc\tan\left(\frac{F_y}{F_z}\right) \tag{18}$$

$$\theta = arc\tan\left(\frac{-F_x}{\sqrt{F_y^2 + F_z^2}}\right) \tag{19}$$

Si se tienen en cuenta, además, las lecturas aportadas por el magnetómetro, se puede calcular el ángulo de guiñada con respecto al norte magnético. Se define el vector de fuerzas magnéticas $M = (M_x, M_y, M_z)$ y se rota con los ángulos calculados anteriormente para obtener las componentes m_n y m_E (norte y este, respectivamente):

$$\begin{aligned} m_n &= M_x \cos\theta + M_y \sin\phi + M_z \cos\phi \sin\theta \\ m_E &= M_y \cos\phi - M_z \sin\phi \end{aligned} \tag{20}$$

Utilizando estas dos componentes del campo magnético se puede calcular el ángulo de guiñada:

$$\psi = arc\tan\left(\frac{m_E}{m_N}\right) \tag{21}$$

Este método de cálculo de la orientación es bastante preciso en caso de que el sistema se encuentre en reposo. No obstante, la naturaleza de los dos tipos de sensores mencionados lo hace poco fiable durante la operación del quadrotor.

Por una parte, puesto que el acelerómetro mide el efecto de todas las fuerzas que actúan sobre el sistema, sus lecturas no se van a limitar a reflejar la posición del vector gravedad. Cualquier pequeña fuerza que actúe sobre la plataforma puede falsear las medidas. Esto es crucial en vuelo, puesto que las actuaciones sobre el quadrotor se van a ver reflejadas en las lecturas del sensor, lo que además se ve agravado por las vibraciones presentes en el sistema debidas, esencialmente, al funcionamiento de los motores.

Por otra parte, el magnetómetro es muy sensible al ruido electromagnético. Teniendo en cuenta que la plataforma requiere de dispositivos electrónicos para su funcionamiento, se puede deducir que un diseño mal planteado puede provocar lecturas poco fiables.

Así pues, se hace patente la necesidad de considerar el último tipo de sensor mencionado: el giróscopo. Los giróscopos proporcionan medidas de la velocidad angular alrededor de un eje o, lo que es lo mismo, la derivada del ángulo correspondiente. Por lo tanto, para obtener la orientación, se puede integrar la velocidad angular. Puesto que en los sistemas de control las lecturas se realizan de modo discreto, se puede llegar a esta expresión:

$$\theta(t) = \theta(0) + \int_0^t \dot{\theta}(t)dt \approx \theta(0) + \sum_0^t \dot{\theta}(t)\, T_s \tag{22}$$

donde T_s es el período de muestreo de las lecturas y $\theta(0)$ la lectura inicial de ángulo correspondiente obtenida, por ejemplo, con el método descrito anteriormente.

Las lecturas que proporcionan los giróscopos suelen ser muy precisas, por lo que constituyen una medida fiable de la derivada del ángulo. Sin embargo, la aproximación discreta de la integral implica que no se de la misma situación para el cálculo de la orientación. Si la información del giróscopo cambia más rápido que la frecuencia de muestreo, el algoritmo no lo detectará, por lo que la aproximación integral será incorrecta. La acumulación de estos errores a lo largo del tiempo provoca una deriva en el ángulo calculado que implica, conforme su magnitud vaya creciendo, un mal control debido a la determinación errónea del estado del sistema.

Por lo tanto, queda patente la necesidad de fusionar de algún modo la medida de los distintos sensores, de modo que se puedan aprovechar las ventajas que presenta cada uno según la situación, minimizando el efecto de las desventajas. Esto se puede lograr mediante algoritmos de filtrado.

A continuación se describen dos ejemplos de algoritmos de filtrado: el filtro complementario (sin estimación de la deriva provocada por la integración de los giróscopos) y el filtro de Kalman (con estimación de la deriva). Las medidas de los ángulos utilizadas por estos algoritmos se pueden obtener por los métodos ya descritos. En caso de aplicarlos sobre los ángulos de alabeo o de cabeceo, se fusionan los datos obtenidos por el acelerómetro y el giróscopo. En el caso del ángulo de guiñada, se hace lo propio con los datos de magnetómetro y giróscopo.

En los siguientes desarrollos matemáticos, θ denota el ángulo estimado por el filtro, a el ángulo calculado a partir de las lecturas del acelerómetro (o del magnetómetro en el caso del *yaw*) y ω las lecturas de velocidad angular proporcionadas por el giróscopo. Además, se denomina Δt al intervalo de tiempo entre muestras, y n o k al índice del tiempo discreto. Las cantidades estimadas de una variable se indican con un acento circunflejo (por ejemplo $\hat{x}$) en caso de que haya posibilidad de confusión.

3.2.1. Filtro complementario

El filtro complementario fusiona los datos del acelerómetro y la integración del giróscopo implementando un filtro paso bajo[3] sobre los primeros, un filtro paso alto[4] sobre los segundos y sumando la salida de ambos. Esto se traduce en que, a corto plazo, se utilizan las medidas derivadas del giróscopo, muy preciso y no susceptible al ruido de las fuerzas externas, mientras que a largo plazo se utiliza la medida del acelerómetro, atenuando el efecto de la deriva del giróscopo. Su expresión en el dominio de la frecuencia es:

$$\theta = \frac{1}{1+\tau s} a + \frac{\tau s}{1+\tau s}\frac{1}{s}\omega = \frac{a+\tau\omega}{1+\tau s} \tag{23}$$

donde τ permite determinar las frecuencias de corte del filtro.

Dicha expresión se debe pasar a un dominio de tiempo discreto mediante la siguiente transformación:

$$1+\tau s = \left(1+\frac{\tau}{\Delta t}\right) - \frac{\tau}{\Delta t} z^{-1} \tag{24}$$

Insertando en la Ecuación 23 y operando se llega al resultado final:

$$\theta_k = \alpha\,(\theta_{k-1} + \omega_k \Delta t) + (1-\alpha) a_k \tag{25}$$

donde $\alpha = \frac{\tau}{\Delta t}/\left(1+\frac{\tau}{\Delta t}\right)$.

Esta expresión se puede implementar fácilmente a la hora de desarrollar el software necesario para la plataforma, requiriendo escasa capacidad de cómputo. La variación del parámetro α permite cambiar el peso en el ángulo estimado de las medidas derivadas del acelerómetro y del giróscopo. Así, un mayor valor de α proporcionará estimaciones menos susceptibles al ruido, pero con una mayor deriva en el tiempo, mientras que un menor valor de α tendrá el efecto contrario.

---

[3] Un filtro paso bajo es un filtro que permite el paso de señales cuya frecuencia es menor a una determinada frecuencia de corte, mientras que atenúa aquellas cuya frecuencia es mayor. En la aplicación considerada implica disminuir el efecto del ruido de alta frecuencia (por ejemplo, aquel provocado por las vibraciones inducidas por los motores). Su expresión en el dominio de la frecuencia es:

$$H(s) = \frac{1}{\tau s + 1}$$

donde τ es la constante de tiempo del filtro, que permite determinar la frecuencia de corte.

[4] Un filtro paso alto hace el efecto contrario, lo que atenúa el efecto del ruido de baja frecuencia. Su expresión en el dominio de la frecuencia es:

$$H(s) = \frac{\tau s}{\tau s + 1}$$

3.2.2. Filtro de Kalman

El filtro del Kalman toma en cuenta el ruido mediante matrices de covarianza que se actualizan regularmente, en cada ejecución del algoritmo, usando ecuaciones que suelen ser extensas y complicadas. Sin embargo, si las matrices son constantes, las ecuaciones del filtro de Kalman se simplifican enormemente. Por ello, para el siguiente desarrollo, se asumirá que dichas matrices son constantes, puesto que no cabe esperar grandes variaciones en el espectro de ruido del acelerómetro y el giróscopo, a fin de mostrar un modelo del filtro de Kalman sencillo, pero que al menos tenga en cuenta la deriva de los giróscopos.

El filtro de Kalman en tiempo discreto se aplica sobre sistemas modelados en espacio de estados como:

$$\begin{aligned} x_k &= Ax_{k-1} + Bu_k + w \\ z_k &= Hx_k + v \end{aligned} \tag{26}$$

donde x_k es el vector de estados, u_k el vector de control y z_k el vector de medidas.

Por otra parte, las ecuaciones del filtro de Kalman son:

$$\begin{aligned} \hat{x}_k^- &= A\hat{x}_{k-1} + Bu_k \\ \hat{x}_k &= \hat{x}_k^- + K(z_k - H\hat{x}_k^-) \end{aligned} \tag{27}$$

donde $\hat{x}_k^-$ es la estimación del estado predicha y $\hat{x}_k$ la actualizada.

Por lo general, un mismo sistema se puede describir mediante distintos modelos en espacio de estados. Esto se traduce en que se pueden desarrollar distintos filtros de Kalman según cuál sea el modelo planteado.

Como punto de partida de un flitro de Kalman sencillo, se puede plantear el siguiente modelo en espacio de estados:

$$\begin{aligned} \theta_k &= \theta_{k-1} + (\omega_k \mp b_{k-1})\Delta t \\ b_k &= b_{k-1} \\ z_k &= a_k \end{aligned} \tag{28}$$

donde b representa sesgo (*bias*) del giróscopo.

Los vectores de estado y matrices, por su parte, son:

$$\begin{gathered} x = \begin{pmatrix} \theta \\ b \end{pmatrix}, \qquad u = (\omega), \qquad A = \begin{pmatrix} 1 & \mp\Delta t \\ 0 & 1 \end{pmatrix}, \qquad B = \begin{pmatrix} \Delta t \\ 0 \end{pmatrix}, \\ z = (a), \qquad H = (1 \quad 0), \qquad K = \begin{pmatrix} K_0 \\ K_1 \end{pmatrix} \end{gathered} \tag{29}$$

Las ecuaciones del filtro de Kalman que se derivan de esto son:

$$\begin{aligned}\hat{\theta}_k^- &= \hat{\theta}_{k-1} + \left(\omega_k \mp \hat{b}_{k-1}\right)\Delta t \\ \hat{\theta}_k &= (1 - K_0)\,\hat{\theta}_k^- + K_0\, a_k \\ \hat{b}_k &= \hat{b}_{k-1} + K_1\left(a_k - \hat{\theta}_k^-\right)\end{aligned} \tag{30}$$

Estas ecuaciones se pueden expresar del siguiente modo:

$$\begin{aligned}\hat{\theta}_k &= \alpha\,\hat{\theta}_{k-1} + (1 - \alpha)\,a_k + \alpha\left(\omega_k \mp \hat{b}_{k-1}\right)\Delta t \\ \hat{b}_k &= \hat{b}_{k-1} + K_1\left(a_k - \hat{\theta}_k^-\right)\end{aligned} \tag{31}$$

donde $\alpha = 1 - K_0$.

En este caso, puesto que se trata de una implementación sencilla del filtro de Kalman, se puede observar el parecido con la expresión del filtro complementario. No obstante, en este caso se corrige la velocidad angular con el *bias* estimado, lo que debería mejorar la estimación del ángulo proporcionada por el filtro a cambio de algo más de coste computacional.

De todos modos, es posible plantear otros modelos de filtro de Kalman más completos, así como variaciones, tales como el filtro de Kalman extendido, que permitan conocer con mayor exactitud el estado del sistema, a cambio de un mayor coste de cálculo. Sin embargo, no se expondrá aquí su desarrollo, ya que es posible encontrar una amplia bibliografía al respecto y su desarrollo se extendería en exceso.

4. Diseño preliminar de la plataforma

Como ya se ha mencionado en la introducción a esta memoria, el desarrollo del proyecto pasó por dos fases distintas, que implicaban diferentes arquitecturas para la plataforma, tanto a nivel de hardware como de software. A continuación, se describe el primer borrador de la plataforma en ambas fases, planteando las partes y funciones que el sistema ha de tener para lograr su cometido, así como la relación necesaria entre éstas, sin entrar en detalle acerca de los componentes utilizados o los programas posteriormente desarrollados.

No obstante, en apartados posteriores del documento, se ignorará la versión inicial no construida, y se procederá a explicar detalladamente tanto la arquitectura hardware como software de la versión final de la plataforma, desarrollada durante la mayor parte de la duración del proyecto.

4.1. Planteamiento inicial de la plataforma

En un principio, el proyecto se plantea como una continuación del trabajo ya realizado por el equipo del profesor tutor Pedro García. En el laboratorio se contaba con varios prototipos funcionales cuya arquitectura se basaba en dos capas (un mini PC IGEP v2 y un microcontrolador Arduino Due) ejecutando el control a dos niveles, a cambio de un mayor peso de la plataforma así como un mayor consumo energético.

Por una parte, el microcontrolador se encargaba de la conexión con dispositivos como los controladores de los motores o diversos sensores inerciales así como de ultrasonidos. Por otra parte, el mini PC, que cuenta con sistema operativo, se encargaba de la conexión inalámbrica con la estación de tierra así como de otros dispositivos que requerían más potencia de cálculo, como una cámara destinada al control de posición por flujo óptico. El control se ejecutaba a dos niveles, haciendo uso de ambas placas.

La idea inicial del proyecto es reducir dicha arquitectura a un solo nivel, haciendo uso de un nuevo mini PC, el BeagleBone Black. Dicha placa cuenta con una capacidad de procesamiento similar a la IGEP v2 (CPU ARM Cortex-A8 a 1 GHz y 512 MB de memoria RAM en ambos casos), ambas funcionan con sistemas operativos en base Linux y a la hora de programar en ambas placas, las diferencias son mínimas.

Además, el uso de este nuevo mini PC presentaba unas ventajas importantes:

- El precio de la BeagleBone Black es de unos 65€, frente a los 179€ que cuesta la IGEP v2 en la actualidad.
- La BeagleBone cuenta con una mayor cantidad de puertos que la IGEP, lo que, en un principio, permite conectar todos los dispositivos a ésta, incluidos los controladores de los motores, ya que incorpora salidas PWM.

Figura 15: Mini PCs IGEP v2 y BeagleBone Black

Así pues, es evidente la reducción del coste de la plataforma, tan sólo sustituyendo la IGEP por la BeagleBone Black. No obstante, las posibilidades en lo que a conectividad se refiere también permiten prescindir del microcontrolador presente en las plataformas anteriores reduciendo aún más el coste, y también el peso del sistema.

Por lo tanto, se plantea una plataforma a un solo nivel, con un mini PC que se encargue de todo lo relacionado con el control: lectura de sensores, filtrado y cálculo de orientación, comunicación con la estación de tierra, algoritmos de control y actuación sobre los motores. Al mini PC habría que conectar al menos una IMU y un sensor de ultrasonidos, para poder obtener las medidas necesarias para el control deseado. La estación de tierra, por su parte, ha de permitir la comunicación del usuario con el sistema, a fin de enviarle las referencias y parámetro pertinentes haciendo uso de una serie de periféricos. El esquema de funcionamiento global se muestra en la Figura 16.

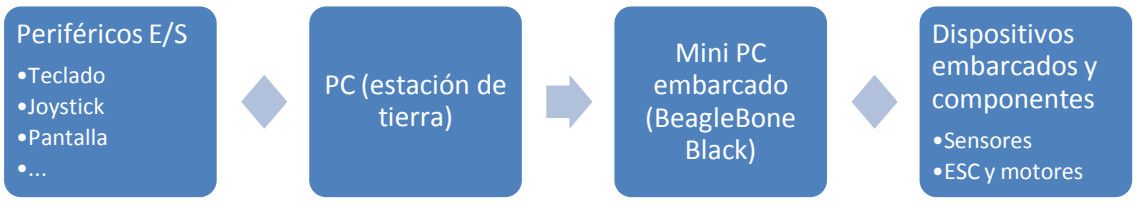

Figura 16: Esquema de funcionamiento global de la primera versión de la plataforma

Para el desarrollo de dicha idea de la plataforma, se utilizaría una de las estructuras ya disponibles en el laboratorio, se partiría del software desarrollado para la IGEP v2 y el Arduino Due, haciendo las modificaciones necesarias para unificarlo y compatibilizarlo con el nuevo mini PC, y se haría uso también de la interfaz de usuario ya desarrollada en el laboratorio para comunicarse con el sistema y enviarle las referencias y parámetros de control necesarios para el vuelo.

No obstante, se dan unas circunstancias que suponen el abandono de esta idea y el planteamiento de un nuevo borrador de la plataforma.

En primer lugar, el otro estudiante compañero de desarrollo de la plataforma, Norberto Vera Vélez, había trabajado durante el primer semestre del curso con el microcontrolador STM32F3

Discovery en una de las asignaturas que cursaba: *Sistemas embarcados para navegación y control*. Esta placa lleva una IMU incorporada, la cual se había aprendido a utilizar durante el transcurso de la asignatura, siendo además su coste bastante reducido, lo que no suponía un aumento considerable sobre el precio de la plataforma ya planteada. Así pues, se decidió aprovechar la experiencia en dicho microcontrolador, por lo que sería necesario replantear la plataforma.

Además, también se dio la circunstancia de que, debido a la cantidad de estudiantes que estaban trabajando en el mismo laboratorio, finalmente no sería posible utilizar ninguna de las estructuras ya montadas, por lo que iba se iba a tener que comenzar la construcción del quadrotor desde cero, siendo necesaria la selección de nuevas piezas con las que en un principio ya se contaba.

4.2. Versión desarrollada de la plataforma

Teniendo en cuenta las circunstancias mencionadas, se decide iniciar el diseño de una nueva plataforma, partiendo de cero, en la que toda la arquitectura, tanto a nivel de hardware como de software esté planteada por los estudiantes involucrados en el desarrollo de este proyecto. Esto implica rediseñar el sistema quadrotor al completo, así como la estación de tierra. El nuevo planteamiento, que pretende hacer uso tanto de la BeagleBone Black como de la STM32F3 *Discovery,* supone un diseño del software a dos capas.

Por una parte, la placa Beagle se encargará de ejecutar el bucle de control, lo que requiere la comunicación con el resto de dispositivos, a fin de obtener los parámetros necesarios para la ejecución de éste. El mini PC se conectará de forma inalámbrica a la estación de tierra, mientras que la conexión con la STM se realizará por USB, aprovechando el puerto presente en la Beagle, y logrando una conexión estable y rápida. Además, se conectará un sensor de ultrasonidos que nos permita obtener medidas de la altura.

Mediante los protocolos pertinentes se enviará la información necesaria entre los dispositivos. Una vez recibidos los mensajes, se procesarán, obteniendo los valores correspondientes a cada parámetro. En el caso de los procedentes de los sensores, se han de filtrar, a fin de lograr estimaciones utilizables de la orientación o la posición. Así, se puede proceder a la ejecución de los algoritmos de control que generarán las acciones de control que han de recogerse en un mensaje, enviado a la STM.

Por otra parte, la placa STM se encargará de la lectura de la IMU que tiene incorporada, así como de la generación de las señales PWM a enviar a los motores, a partir de las acciones de control generadas y enviadas por la Beagle. En este caso, se ha de programar la comunicación con la Beagle de modo que la placa responda con la información correcta a las peticiones del mini PC.

Se plantea como objetivo lograr ejecutar el bucle de control a unos 5 ms, en la línea del resto de sistemas disponibles en el laboratorio, a fin de lograr una respuesta rápida del sistema y, con ello, un comportamiento suave y estable. Esto, como se verá en el apartado posterior de la

memoria dedicado a ello, condicionará el desarrollo del software de la plataforma en los dos niveles de hardware.

Para la estructura del quadrotor, a fin de ahorrar tiempo, se decide buscar un kit de montaje comercial, que incluya todas las piezas necesarias y permita dedicar más tiempo a la electrónica de la plataforma, así como al software. Se opta por el kit DJI F450 E300 que, además de la estructura, incluye cuatro motores con sus respectivos controladores de velocidad, así como hélices. En la Figura 17 se pueden ver las piezas que incluye dicho kit en el quadrotor ya montado, junto a algunos de los componente electrónicos mencionados.

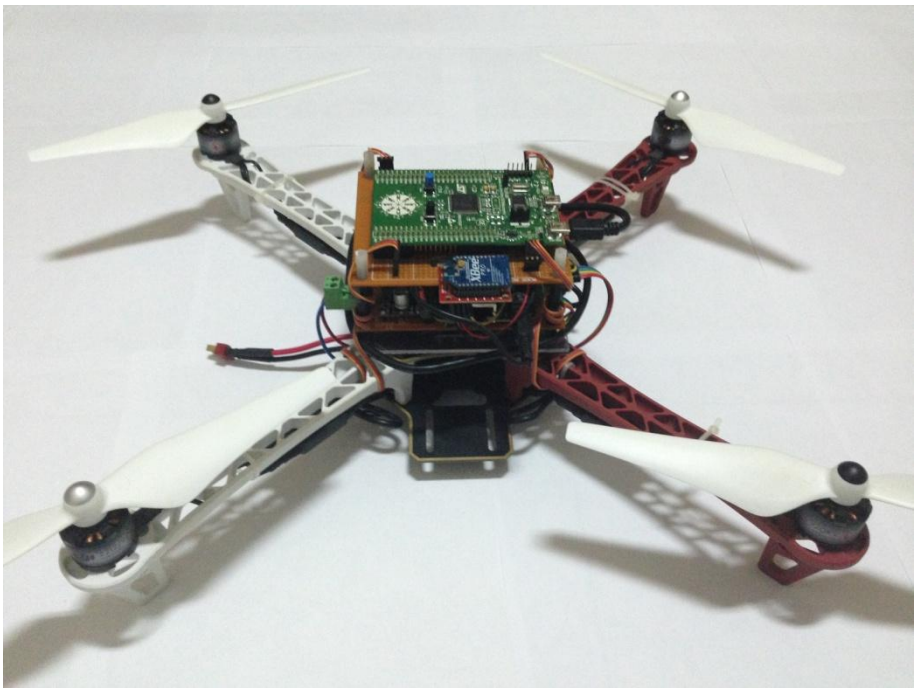

Figura 17: Vista de la versión desarrollada del quadrotor

En cuanto a la estación de tierra, se decide crear una nueva interfaz de usuario compatible con el sistema operativo Windows. El software deberá permitir la entrada por parte del usuario de una serie de parámetros de los algoritmos de control, a fin de poder ajustarlos en vuelo, así como las referencias de ángulos y posición pertinentes.

Para la conexión del PC que ejecute el programa con el quadrotor, se decide apostar por el uso de módulos de radiofrecuencia que ofrezcan un mayor alcance que la conexión wifi utilizada por algunos de los prototipos antiguos del laboratorio.

Además se decide programar la entrada de los valores de referencia mediante la interfaz de programación de aplicaciones, o API, XInput. Esta API, disponible junto con el sistema operativo Windows, permite la interacción con dispositivos de entrada como el mando de la consola Xbox 360, de forma nativa, o incluso los de las consolas Playstation 3 y 4, instalando controladores adicionales, lo que ofrece un amplio abanico de posibilidades. Estos dispositivos,

aparte de resultar bastante ergonómicos, ponen a disposición del usuario una gran cantidad de botones y *sticks* bastante precisos, lo que los hace ideales para la aplicación deseada.

Finalmente, la Figura 18 muestra un esquema simplificado del funcionamiento de la plataforma a desarrollar y la conexión entre los distintos componentes.

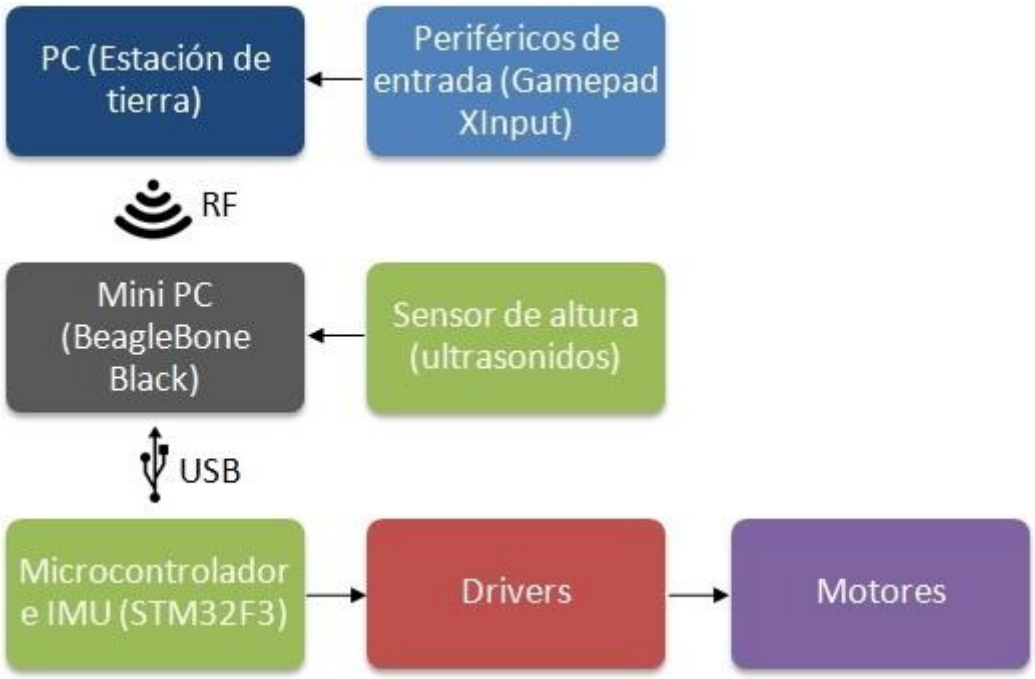

Figura 18: Esquema de funcionamiento global de la versión final de la plataforma

5. Hardware de la plataforma

En este apartado de la memoria se presentan los principales componentes, sensores y procesadores utilizados en la versión final del quadrotor, describiendo sus características. Además, también se presenta el hardware necesario para el funcionamiento de la estación de tierra.

Si bien no se profundiza en la interconexión de los dispositivos y la alimentación de éstos, más adelante en este documento se puede encontrar información al respecto.

5.1. Componentes

En esta sección se procede a describir las partes más importantes que componen el quadrotor, a excepción de los procesadores y el módulo de radiofrecuencia.

5.1.1. Chasis

El chasis utilizado en la plataforma es un Flame Wheel F450, de la empresa DJI. Se optó por esta estructura teniendo en cuenta la robustez que ofrece, unida a su ligereza. Esto la hace capaz de soportar fuertes impactos sin romperse, mientras que también ayuda a reducir las vibraciones transmitidas por los motores, lo que permite obtener medidas más fiables en los sensores. La Figura 19 muestra el aspecto de dicho chasis, si bien la versión adquirida cuenta con dos brazos blancos y dos rojos, en lugar de los negros. Esto permite conocer la orientación del sistema en vuelo de forma visual.

Figura 19: Chasis DJI F450 Flame Wheel

Otra ventaja de este chasis es la presencia de circuitería PCB integrada en su base, lo que permite soldar a ella los cables de alimentación y los de los controladores ESC de los motores, reduciendo la longitud necesaria de éstos y aumentando la seguridad de plataforma.

El peso de la estructura es de unos 282 g, siendo su diagonal de unos 450 mm y la distancia entre motores contiguos de unos 320 mm. La altura es de 55 mm. Según las especificaciones del fabricante, cumpliendo las recomendaciones, la plataforma es capaz de soportar pesos al despegue entre 800 y 1200 g. Puesto que el chasis se ha adquirido como parte del kit de montaje DJI F450 E300 que, además de la estructura, incluye los cuatro motores con sus respectivos controladores de velocidad y hélices, no debería haber inconveniente a la hora de lograr dichas cifras.

5.1.2. Motores

Los motores utilizados son del modelo 2212/920 (ver Figura 20), fabricados por la empresa DJI, e incluidos en el kit de montaje ya mencionado. Estos motores cuentan con un estator de dimensiones 22x12 mm y una constante de velocidad de 920 rpm/V, es decir, pueden girar a 920 revoluciones por minutos en caso de ser alimentados por 1 V.

Cada unidad pesa unos 50 g. Se recomienda su alimentación con baterías LiPo de 3 celdas, a tensiones de 11.1 V. Su punto óptimo de funcionamiento es bajo una alimentación de 7.2 A, consumiendo hasta 80 W. Esto permite levantar cargas de 600 g por cada eje (usando hélices del tamaño recomendado, 9.4x4.3”), si bien la carga recomendada es de 300 g.

Figura 20: Motores DJI 2212/920

5.1.3. Controladores de los motores

Los controladores electrónicos de velocidad (ESC) de los motores utilizados en la plataforma son del modelo E300 Mark II (ver Figura 21), también de la empresa DJI, e incluídos en el kit de montaje.

Están pensados para ser alimentados por baterías LiPo de 3 o 4 celdas, con tensiones entre 11.1 y 14.8 V, siendo la intensidad máxima admitida de 15 A.

Figura 21: Controlador DJI E300 Mark II

Los ESC reciben la alimentación de la batería, así como una señal PWM enviada, en el caso de esta plataforma, por el microcontrolador STM. Su función es generar una señal trifásica, a partir del PWM, que se envía a los motores, controlando así su velocidad de giro. Este modelo de ESC, de tipo Opto, cuenta con circuitos de señal y potencia aislados.

Estos controladores pueden operar en un rango de señales PWM entre 30 y 450 Hz de frecuencia. Una mayor frecuencia en la señal PWM permite lograr un ajuste más fino de la velocidad de giro del motor, logando un comportamiento suave, que puede beneficiar el control de la plataforma.

Por lo general, los ESC presentan un valor mínimo y máximo de ciclo de trabajo de la señal PWM, correspondientes al 0% y 100% de potencia. Estos valores suelen ser calibrables, aunque en este modelo en concreto no es posible, por lo que se han tenido que establecer mediante pruebas en el laboratorio.

En cuanto al cableado, estos controladores cuentan con cables coaxiales, que permiten una alta compatibilidad electromagnética, lo que beneficia el funcionamiento del sistema y, en concreto, el de los sensores.

5.1.4. Hélices

Las hélices utilizadas en la plataforma, último de los componentes incluidos en kit de montaje, son rotores de tamaño estandarizado 9.4x4.3” (ver Figura 22), también de la empresa DJI. Se corresponden con la medida recomendada por el fabricante por la plataforma, por lo que permiten lograr la sustentación deseada, incluso con un margen para futuras ampliaciones del sistema. Además, están diseñadas a fin de reducir su inercia, lo que permite usar algoritmos de control agresivos, con una mayor ganancia, sin comprometer la estabilidad.

El kit incluye dos tipos hélices, según el sentido de giro del motor al que vayan unidas. Una gran ventaja de estos rotores es la facilidad de montaje, ya que no requieren de piezas extra como tornillos o tuercas. Las hélices se roscan sobre el eje del motor, siendo distinto el sentido según el giro de éste. Así, un funcionamiento normal de la plataforma tenderá a mantener fijadas las hélices gracias al sentido de la fuerza de resistencia aerodinámica generada.

Figura 22: Hélices DJI 9.4x4.3"

5.1.5. Batería

La batería utilizada es una batería LiPo Zippy Compact 5800 3S 25C (ver Figura 23). Se trata de una batería con una capacidad de 5800 mAh, 3 celdas en serie, voltaje nominal de 11.1 V y una descarga máxima de 25 C continuos. Su peso es de 433 g aproximadamente.

Figura 23: Batería Zippy Compact 5800 3S 25C LiPo

Se escogió dicho modelo por su reducido tamaño en comparación al resto de baterías disponibles en el laboratorio, siendo esta la única que encajaba en la góndola de la estructura destinada a su transporte.

Las baterías LiPo son muy utilizadas en quadrotors debido a su alta relación carga/peso y su precio económico. Sin embargo, es necesario llevar mucho cuidado al cargarlas, no superando

las especificaciones en corriente de carga, e intentando igualar la carga en las tres celdas, a fin de lograr el mejor rendimiento posible y prolongar la vida útil de la batería. En caso de no cumplir los requisitos, se corre el riesgo de deteriorar la batería, pudiendo derramar esta el componente químico interno o incluso llegar a explotar. No obstante, las baterías deterioradas suelen hincharse, por lo que visualmente se puede apreciar está condición y sustituirla por otra en mejores condiciones antes de que implique un riesgo para la plataforma.

Por ello, las baterías LiPo se suelen cargar con cargadores comerciales especializados, que se encargan de regular la corriente, igualar las celdas, etc.

La capacidad de descarga de la batería utilizada es más que suficiente para la plataforma, proporcionando más de los 60 A máximos que podría llegar a requerir la plataforma, alrededor de 30 A en caso de un uso normal.

Las baterías de este tipo de uso general suelen incluir circuitos de de seguridad de corte de suministro, que cortan el suministro de corriente cuando el voltaje es demasiado bajo, a fin de no dañar la batería. No obstante, en este caso es preferible dañar la batería antes que el resto del sistema, por lo que el modelo elegido no cuenta con dicho circuito.

Así pues, es necesario contar con un dispositivo como el de la Figura 24, que monitorice el voltaje de la batería, emitiendo un pitido cuando la batería, o alguna de sus celdas, alcanza el voltaje mínimo establecido. En caso de oír dicho aviso acústico, se deberá proceder al aterrizaje del quadrotor.

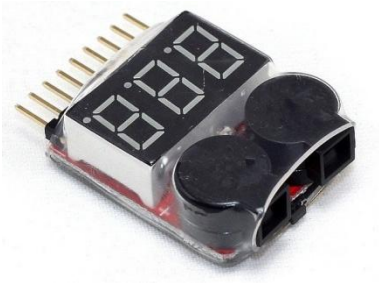

Figura 24: Indicador de voltaje genérico

El funcionamiento normal de este tipo de baterías suele ser el siguiente:

- Se cargan controladamente hasta el voltaje indicado por el fabricante, generalmente en torno a los 12.5 V.
- Se conectan al dispositivo (el quadrotor en este caso) y comienzan a descargarse. Al principio la descarga es rápida, hasta alcanzar los 11.1 V de voltaje nominal, que se suele mantener durante el 80% de tiempo de uso de la batería aproximadamente.
- Finalmente, el voltaje comienza a disminuir bruscamente. A pesar de ello se pude usar sin peligro hasta alcanzar el voltaje mínimo indicado, que suele ser de 9 V, momento en el que se debería desconectar para evitar daños.

En el caso de la batería concreta utilizada no se ha logrado encontrar un diagrama de descarga, si bien cabe esperar un comportamiento similar al resto de baterías LiPo. No obstante, hay que considerar también aspectos externos que afectan a la duración de la batería, como la temperatura.

Teniendo en cuenta las especificaciones conocidas, en un entorno estándar y bajo un consumo normal de la plataforma, cabe esperar duraciones entre 10 y 15 minutos por parte de la batería.

5.1.6. Regulador de tensión

La plataforma utiliza un regulador de tensión genérico HCJ-IPM-V2 (ver Figura 25), a fin de reducir los 11.1 V nominales de la batería a los 5 V necesarios para alimentar sensores y procesadores. Dicho regulador admite tensiones de entrada en 3 y 40 V, y su salida se puede regular entre 1.5 y 35 V, bajo la condición de que el voltaje de entrada sea, al menos, 1.5 V superior al de salida. Además, suministra una corriente máxima de 3 A, suficiente para cubrir las necesidades de los dispositivos que alimenta.

Cuenta con un potenciómetro que permite el ajuste del voltaje de salida. Sin embargo, el regulador utilizado, uno de los disponibles en el laboratorio, lo tiene encolado, a fin de evitar la variación de dicha tensión, estando fija en 5.3 V, diferencia que se ha considerado aceptable, y que no afecta al funcionamiento del sistema.

Figura 25: Regulador de tensión HCJ-IPM-V2

5.2. Procesadores

En esta sección se procede a describir los dos procesadores utilizados en la plataforma: la placa BeagleBone Black, mini PC con una gran capacidad de procesamiento, y el microcontrolador STM32F3 *Discovery*, que cuenta con una IMU integrada.

5.2.1. STM32F3 Discovery

El microcontrolador STM32F3 *Discovery* (ver Figura 26), fabricado por la empresa STMicroelectronics, constituye una plataforma de desarrollo muy atractiva con un amplio abanico de funcionalidades que permiten el desarrollo de proyectos muy variados, lo que se une a un precio bastante bajo.

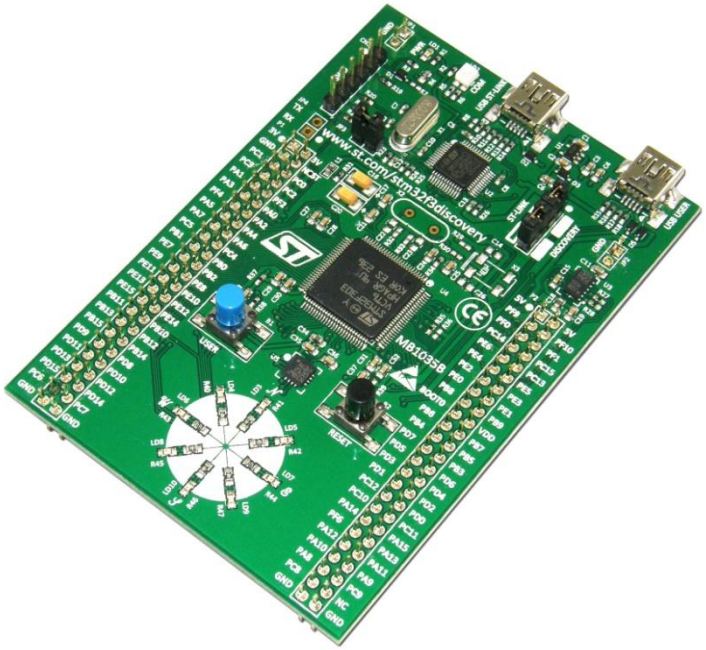

Figura 26: Microcontrolador STM32F3 Discovery

Las principales características de esta placa son:

- Microprocesador ARM Cortex-M4 de 32 bits, modelo STM32F303VCT6, a 72 MHz.
- 48 KB de RAM y 256 KB de memoria flash.
- Alimentación mediante USB o una fuente externa de 3 o 5 V.
- Suministro a 3 o 5 V para aplicaciones externas.
- Sensor L3GD20, giróscopo de 3 ejes.
- Sensor LSM303DLHC, acelerómetro de 3 ejes y magnetómetro de 3 ejes.
- Diez LEDs programables.
- Dos botones pulsadores.
- 100 pines digitales de E/S, con diversas funciones, incluyendo soporte PWM en algunos.
- Dos puertos USB, uno para programación y otro para conexión con dispositivos.

Estas características la convierten en una plataforma de desarrollo muy versátil lo que, unido a la experiencia anterior del compañero de desarrollo de la plataforma, llevó a tomar la decisión de usarla, en lugar de utilizar alguna de las IMU disponibles en el laboratorio. Más adelante se describirá las características de los sensores que incorpora.

5.2.2. BeagleBone Black

El mini PC BeagleBone Black (ver Figura 27), producido por la empresa Texas Instruments en asociación con Digi-Key y Newark Element14, es una plataforma orientada al desarrollo de software libre embarcado.

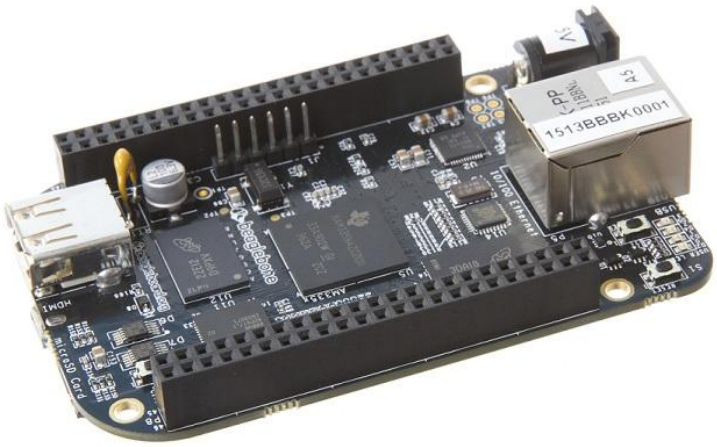

Figura 27: Mini PC BeagleBone Black

Su potencia de cálculo es equiparable a la de ordenadores portátiles antiguos, en un tamaño mucho más reducido, al igual que su consumo. Además, del mismo modo que dichos dispositivos, puede ejecutar un sistema operativo que sea compatible con su arquitectura ARM.

Esta plataforma cuenta además con una comunidad de usuarios activa, en gran parte gracias a su precio, muy competitivo, lo que facilita la iniciación en el desarrollo de software compatible y la solución de los problemas que vayan surgiendo durante su uso.

Las principales características de la BeagleBone Black son:

- Microprocesador ARM Cortex-A8 de 32 bits, modelo AM3358/9, a 1 GHz.
- 512 MB de RAM DDR3.
- 4 GB de memoria eMMC flash.
- Acelerador de gráficos 3D y unidad de coma flotante NEON.
- Dos microcontroladores de 32 bits PRU.
- Alimentación mediante USB o una fuente externa de 5 V.
- Suministro a 3.3 o 5 V para aplicaciones externas.
- Cuatro LEDs programables.
- 46 pines digitales de E/S, con diversas funciones, incluyendo soporte PWM, interfaz UART o comunicación I2C, por ejemplo.
- Dos puertos USB, uno de *host* y otro de cliente, puertos Ethernet y HDMI y lector de tarjetas microSD.

El modelo montado en la plataforma cuenta con un sistema operativo Debian, en base Linux, de bajo peso, con un *kernel* 3.8 estable, lo que permite un funcionamiento bastante robusto del quadrotor. No obstante, se podría sacar más partido a la placa en caso de instalar un sistema operativo en tiempo real, si bien se intentó, sin éxito, la instalación de Xenomai, una versión del *kernel* de Linux modificada para tiempo real.

Las características mencionadas le dan una gran capacidad de procesamiento a este mini PC, que hacen que sea el centro de esta plataforma quadrotor, en torno al cual gira el funcionamiento del resto de dispositivos. Además, algunos de ellos incluso se alimentan a partir de la Beagle, aparte de estar comunicados con ella mediante diversos protocolos que se describirán más adelante.

5.3. Sensores

En esta sección se describen los sensores utilizados en la plataforma: aquellos incorporados en la placa STM y un sensor de ultrasonidos montado en la base del quadrotor.

5.3.1. Acelerómetros y magnetómetros

El sensor utilizado para las medidas de aceleraciones y campo magnético es el LSM303DLHC (ver Figura 28) incorporado en la placa STM, y que cuenta con 3 acelerómetros y 3 magnetómetros, uno para cada eje.

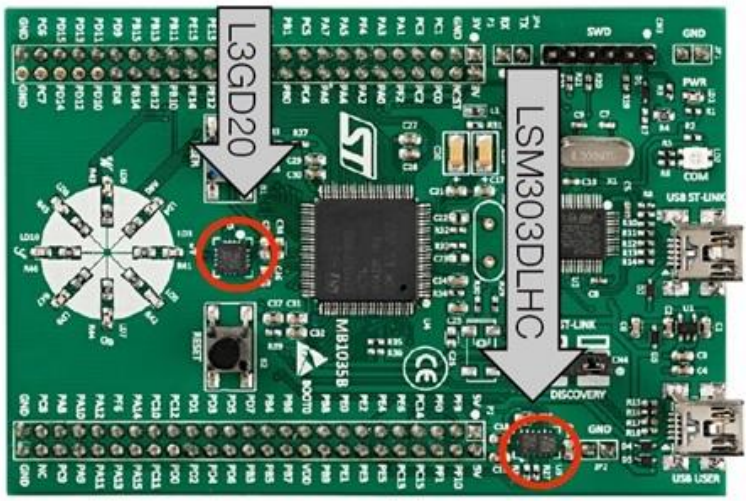

Figura 28: Sensores LSM303DLHC y L3GD20 en la placa STM32F3 Discovery

Este sensor cuenta con un rango de medida ajustable de aceleración linear que escala entre ±2 / ±4 / ±8 / ±16 G y un rango de medida de campo magnético de ±1.3 / ±1.9 / ±2.5 / ±4.0 / ±4.7 /±5.6 / ±8.1 gauss.

Teniendo en cuenta que la plataforma no debería sufrir aceleraciones muy elevadas ni campos magnéticos de alta intensidad, se ha decidido escoger un rango de ±2 G para el acelerómetro y ±1.9 gauss para el magnetómetro, a fin de mejorar la resolución. La medida se proporciona en 16 bits.

El sensor se comunica mediante protocolo I2C, siendo posible acceder a él mediante código desde la misma placa STM.

Tiene un rango de operación en temperatura de -40 a 85° y cuenta con dos modos de trabajo: normal y bajo consumo. Se ha optado por el primero, ya que da una mayor resolución en la medida.

5.3.2. Giróscopos

El sensor utilizado para la medida de velocidades angulares es el L3GD20 (ver Figura 28) incorporado en la placa STM, que cuenta con 3 giróscopos, uno para cada eje.

Este sensor cuenta con un rango de medida ajustable que escala entre ±250 / ±500 / ±2000 grados por segundo. Se ha optado por la escala de ±250, de nuevo por la mayor resolución que ofrece. La medida también se proporciona en 16 bits.

El L3GD20 posee interfaces de comunicación SPI e I2C. No obstante, se ha optado por I2C, a fin de que la comunicación sea equivalente a la del resto de sensores.

El rango térmico de operación coincide con el del LSM303DLHC, siendo de -40 a 85°.

5.3.3. Sensor de ultrasonidos

El sensor utilizado para la medida de altura es un sensor de ultrasonidos SFR10 (ver Figura 29).

Figura 29: Sensor de ultrasonidos SFR10

Este sensor emite un pulso de ultrasonidos a 40kHz en dirección al suelo, donde rebota, regresando al sensor. Midiendo el tiempo que tarda en hacer dicho recorrido se puede conocer la altura, que será la mitad de la distancia recorrida por el pulso.

Este sensor se alimenta a 5 V, y usa el protocolo I2C para la comunicación, que en este caso se maneja desde la BeagleBone Black. Además, se puede configurar para que proporcione las medidas en pulgadas, centímetros o microsegundos.

La medida tiene un rango de dos bytes. El valor máximo se corresponde con el tiempo máximo (65 ms aproximadamente) que el sensor espera a la llegada del eco, tras emitir un pulso. En caso de no llegar, la medida proporcionada será dicho valor máximo. Ese valor se corresponde con 11.29 m, que constituye el alcance teórico máximo, aunque por lo general, el alcance efectivo no será de más de 6 metros. La precisión indicada por el fabricante es de 3-4 cm.

Es posible medir con una mayor frecuencia, a cambio de limitar el alcance máximo. No obstante, se ha decido que la toma de medidas a 65 ms es aceptable, manteniendo el rango de 6 metros y posibilitando un vuelo en exteriores limitado.

El diagrama de radiación del sensor el relativamente directivo, con un ancho de haz de 60° a -6 dB, lo que permite obtener una buena respuesta a ángulos bajos (el control limita los ángulos de *pitch* y *roll* del quadrotor a 10°), evitando la recepción de ecos no deseados.

5.4. Estación de tierra

En esta sección se describen los elementos que conforman la estación de tierra: un ordenador con sus periféricos habituales (pantalla, teclado y ratón), periféricos de entrada para asignar las referencias y un módulo de radiofrecuencia.

5.4.1. Ordenador

Para poder ejecutar la interfaz hombre-máquina desarrollada se requiere de un ordenador que cuente con pantalla, teclado y ratón. El sistema operativo ha de ser Windows y debe contar, al menos, con dos puertos USB libres para un periférico de entrada y el módulo de radiofrecuencia.

Durante las pruebas se han utilizado ordenadores con sistema operativo Windows 7, siendo el funcionamiento estable y sin ningún problema aparente. En un principio, el software debería ser también compatible con versiones posteriores como Windows 8, 8.1 y 10.

Además, se deben instalar los drivers tanto del periférico a utilizar para las referencias como del módulo de radiofrecuencia. Las librerías de sistema utilizadas en el software deben venir incluidas en el sistema operativo de serie.

5.4.2. Periféricos de entrada

Para la entrada de los valores de referencia para el control de la plataforma quadrotor se ha optado por el uso de periféricos de entrada compatibles con la interfaz de programación de aplicaciones, o API, XInput, desarrollada por Microsoft.

Esto ofrece un amplio abanico de posibilidades, ya que la cantidad de periféricos compatibles es muy grande, con alternativas de diversas empresas como Logitech, Sony o Microsoft, y diversas opciones en lo que a precio se refiere. No obstante, dicha API se desarrolló con el mando de la consola Xbox 360 (ver Figura 30) en mente, de la empresa Microsoft.

Figura 30: Mando de Xbox 360

Este ha sido el periférico utilizado durante las pruebas del sistema. Se trata de un mando ergonómico, con una gran cantidad de botones digitales, dos gatillos analógicos y un par de sticks también analógicos.

Así, las posibilidades para la entrada de las referencias de la plataforma son muy amplias, dotando al usuario de una gran precisión a la hora de controlar los ángulos del quadrotor, o incluso otras funciones, como la potencia global de los motores.

5.4.3. Módulos de radiofrecuencia

Para la comunicación de la estación de tierra con el sistema quadrotor se utilizan dos módulos de radiofrecuencia Xbee Pro Serie 1 (ver Figura 31), de la empresa Digi International, uno conectado al ordenador y otro embarcado.

Los módulos Xbee se comunican mediante el estándar IEEE 802.15.4, destinado a definir y controlar el acceso a redes inalámbrica de área personal con bajas tasas de transmisión de datos.

En concreto, los módulos Serie 1 utilizados funcionan a 2.4 GHz y vienen configurados y enlazados el uno con el otro de fábrica. La conexión con los dispositivos a comunicar se realiza mediante protocolos serie TTL, permitiendo establecer un enlace remoto sencillo entre ordenadores.

Figura 31: Módulo de radiofrecuencia Xbee Pro Serie 1

Estos módulos se alimentan a 3.3 V, con una intensidad de 215 mA. La potencia de salida es de 60mW (+180dBm) y su alcance máximo en condiciones ideales (espacios abiertos y sin interferencias electromagnéticas) es de 1500 m.

Además, es posible configurarlos mediante un software proporcionado por Digi: XCTU. Este programa permite configurar aspectos como la tasa de transmisión, siendo necesario configurar ambos módulos por igual para que el sistema funcione, ya que la comunicación es asíncrona. Es posible también modificar parámetros avanzados que permitan la encriptación de la información, por ejemplo, aunque no se ha hecho uso de ellos.

Para la conexión de la Xbee con la BeagleBone Black se ha utilizado una placa Xbee Explorer, fabricada por la empresa Sparkfun, que facilita la comunicación entre los dispositivos así como la alimentación, ya que también son compatibles con 5 V, gracias a un regulador interno. En el caso del ordenador, la placa utilizada es la Xbee Explorer USB, que permite la conexión y alimentación por USB. La Figura 32 muestra el aspecto de ambas placas, que cuentan con LEDs que permiten conocer el estado de la comunicación, avisando de la recepción y el envío de datos.

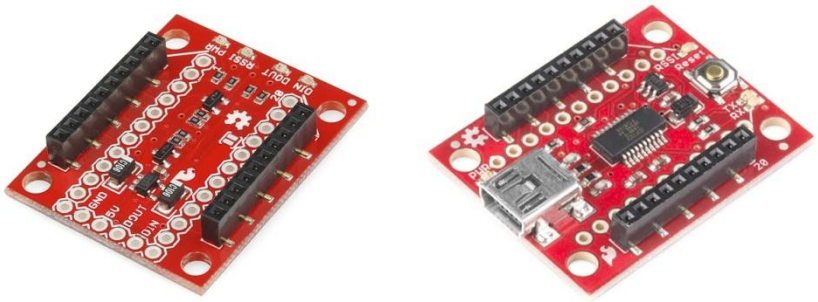

Figura 32: Placas Sparkfun Xbee Explorer y Explorer USB

En el caso de la comunicación entre el mini PC Beagle y la Xbee Explorer se ha usado uno de los puertos UART de la Beagle, lo que ha limitado bastante la velocidad de transmisión ya que no era posible lograr una comunicación fiable a altas tasas de transmisión. No obstante, puesto que el envío de referencias desde la estación de tierra no debe ser crucial para que el quadrotor mantenga un vuelo estable, los tiempos de transmisión para mensajes completos con todos los parámetros, que son del orden de 100 ms, se pueden considerar aceptables.

6. Desarrollo de la plataforma

A continuación, se describe el proceso de montaje de la plataforma quadrotor, posterior a la selección y adquisición de las piezas que conforman su arquitectura hardware, ya mencionadas con anterioridad en la memoria.

Así pues, en este apartado se mencionarán diversos aspectos del montaje de la estructura, así como la conexión entre los dispositivos, haciendo hincapié en dos aspectos cruciales: la alimentación de la plataforma y los protocolos de comunicación utilizados.

6.1. Montaje de la plataforma

Como ya se ha expuesto con anterioridad, para la estructura del quadrotor se optó por adquirir un kit de montaje DJI F450 E300. Además de los motores, los reguladores de velocidad y las hélices, dicho kit incluye todas las piezas necesarias para el montaje del chasis del quadrotor:

- Placa superior del chasis
- Placa inferior con conexiones para alimentación integradas
- Cuatro brazos a los que anclar los motores
- 20 tornillos de 3 mm de diámetro y 8 de longitud
- 30 tornillos de 2.5 mm de diámetro y 5 de longitud
- Cables para conectar la batería

El chasis F450 ofrece una gran facilidad de montaje. Además, es una opción bastante popular debido a su relación calidad/precio, por lo que es posible encontrar tutoriales en internet que explican el proceso de montaje.

Los motores se anclaron a sus respectivos brazos mediante cuatro de los tornillos de métrica 3 haciendo uso de una llave Allen adecuada. Como se deduce del modelo teórico, el sentido de rotación de los motores debe alternarse, por lo que los motores contiguos a uno de giro horario han de ser antihorarios, y viceversa. Los brazos se atornillaron a las placas del chasis mediante los tornillos de métrica 2.5, dos para la placa inferior y cuatro para la superior, por cada unidad.

En cuanto a las conexiones de la placa inferior, en primer lugar, se soldó un conector DEAN tipo T macho a los cables de conexión a la batería, recubriendo posteriormente la zona con tubos termorretráctiles (a los que es necesario aplicar calor para que queden fijados) que protegen la soldadura. Tras ello, se soldaron los cables de alimentación de los ESC, así como los de la batería al circuito impreso en la placa inferior del chasis. Estas soldaduras se cubrieron con cinta aislante para evitar daños.

La conexión entre los motores y los ESC se realiza mediante tres cables de alimentación. En el caso de los de giro antihorario, se conectaron respetando el orden de los cables. Los de giro horario, por otra parte, se conectaron cruzando dos de los tres.

Una vez concluido el montaje de las piezas del kit que formaban la estructura básica del quadrotor, era necesario crear un soporte para los componentes electrónicos a usar en el control de la plataforma. Siguiendo el ejemplo de los prototipos que había en el laboratorio, se procedió a construir una estructura en forma de torre que alojase dichos componentes, partiendo de una placa de prototipado.

Dicha placa se cortó por la mitad, a fin de crear una estructura de dos pisos, taladrando agujeros en las esquinas, para poder pasar los soportes y espaciadores necesarios. Como base de la estructura se usó una placa de metacrilato, cortada y taladrada, que fue anclada al chasis mediante tornillos de la misma métrica que los del kit de montaje pero más largos, adquiridos de manera independiente.

Para el anclaje de los componentes electrónicos a la placa de prototipado se usaron tornillos y tuercas en el caso de la BeagleBone Black y el módulo Xbee montado en una Sparkfun Explorer, y conectores soldados en el caso del regulador de tensión y la STM32F3 *Discovery*. La primera planta de la estructura aloja el mini PC Beagle y el regulador de tensión; mientras que la segunda planta, la placa STM y el Xbee.

La Figura 33 muestra el resultado del proceso de elaboración de la estructura descrita, pudiéndose apreciar los componentes electrónicos ya descritos.

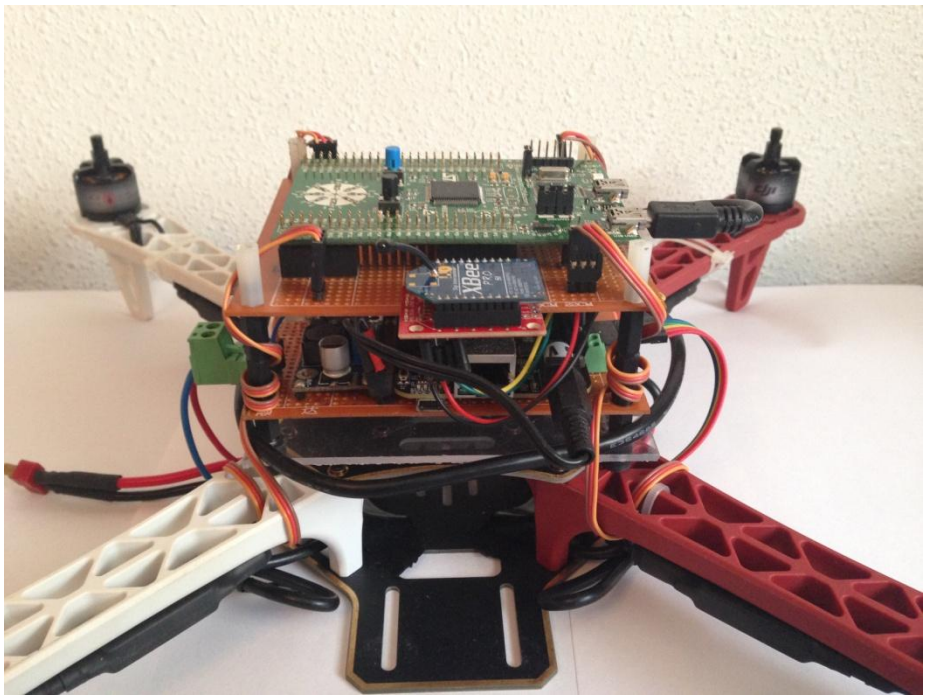

Figura 33: Vista de la estructura de soporte para los componentes electrónicos

Inicialmente se deseaba usar PCB en lugar de placas de prototipado, pero su diseño y pedido implicaba perder un tiempo que se podía dedicar a otras tareas más cruciales. Además las placas utilizadas son bastante más ligeras que las PCB, lo que es una ventaja importante. Por otra parte, al no contar con circuitos impresos, las conexiones entre los dispositivos se han

realizado, o bien con cables de prototipado disponibles en el laboratorio, o bien con soldaduras sobre las placas utilizadas.

6.2. Alimentación de la plataforma

La Figura 24 muestra de manera esquemática los voltajes a los que se alimentan los distintos componentes electrónicos del quadrotor, así como el elemento que se lo suministra.

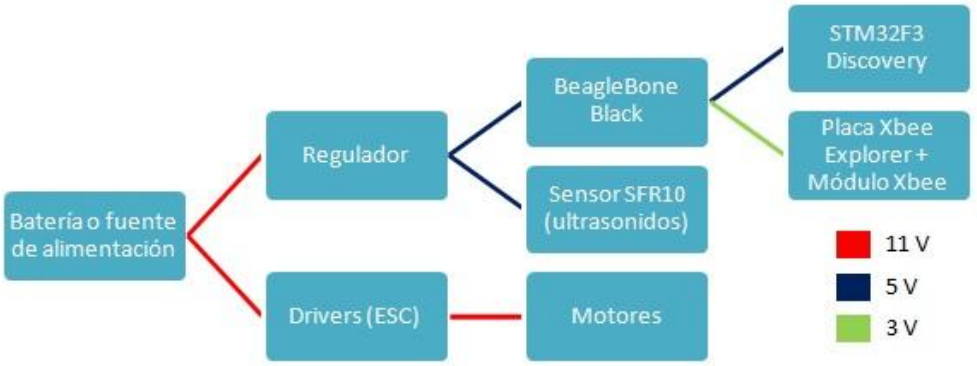

Figura 34: Esquema de alimentación del quadrotor

La batería, como ya se ha mencionado, se encuentra conectada al circuito impreso del chasis, al que están conectados también los ESC así como un cable que parte hacia la torre de eléctronica y se conecta al regulador de tensión, todos ellos alimentados a 11 V.

A dicho regulador se conectan el sensor de ultrasonidos y el mini PC Beagle, este último haciendo uso del conector tipo *jack* con el que cuenta, ambos a 5 V aproximadamente.

Por último, la STM se alimenta de la Beagle mediante USB a 5 V, así como el módulo Xbee, en este caso a partir de uno de los pines de alimentación a 3.3 V.

6.3. Comunicaciones de la plataforma

El funcionamiento de la plataforma planteada requiere de la comunicación entre algunos de los dispositivos electrónicos que la componen con su centro neurálgico, el mini PC BeagleBone Black, a fin de intercambiar la información necesaria para lograr el control del sistema quadrotor.

Esta comunicación se ha llevado a cabo mediante distintos protocolos, haciendo a su vez uso de distintas interfaces de comunicación. La Figura 35 muestra de manera esquemática los protocolos usados en la plataforma quadrotor para la comunicación entre los cuatro dispositivos embarcados que intecambian información.

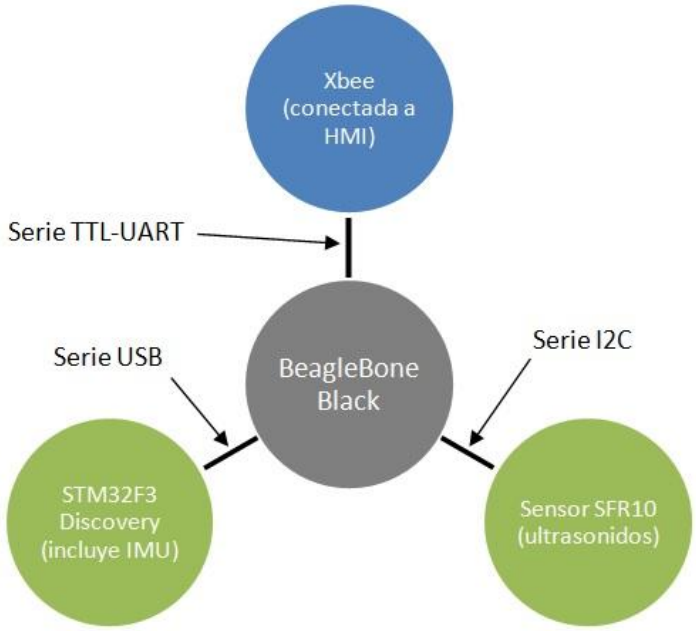

Figura 35: Esquema de comunicaciones del quadrotor

En el caso de la comunicación entre el sensor SFR10 y la Beagle, se hace uso del protocolo de comunicación en serie síncrona I2C, es decir, la información se transmite bit a bit por un mismo canal, con el apoyo de una señal del reloj. Este protocolo es también el que usa la placa STM de manera interna para acceder a su IMU.

En el caso de la comunicación con la STM y el módulo Xbee, la comunicación es serie asíncrona, es decir, no se apoya en una señal de reloj externa, por lo que es necesario establecer de manera idéntica una serie de parámetros en los dispositivos comunicados a fin de que los datos se envíen y reciban correctamente.

A continuación se explica en mayor profundidad los fundamentos de ambos protocolos de comunicación.

6.3.1. Serie TTL, UART y USB

La comunicación serie asíncrona es una transmisión de datos sin apoyo de una señal de reloj externa. Este tipo de protocolos es perfecto para minimizar la cantidad requerida de cables, así como de pines de entrada y salida. No obstante, requiere un esfuerzo extra a fin de lograr enviar y recibir datos de forma fiable.

Así pues, en las comunicaciones en serie asíncrona se definen una serie de mecanismos o parámetros cuya finalidad es lograr una transmisión de datos robusta, minimizando los errores. Los elementos que definen la comunicación en este protocolo son:

- Tasa de baudios
- Bits de datos

- Bits de sincronización
- Bits de paridad

La tasa de baudios especifica la velocidad de transmisión de los datos en una línea en serie. Se trata de un parámetro crucial, que se ha de configurar por igual en los dos dispositivos comunicados, a fin de que los mensajes sean correctamente recibidos, ya que determina el tiempo que se dedica a la transmisión de cada dato. Esto implica tanto el tiempo que el transmisor mantiene el estado alto o bajo correspondiente a lo que se quiere transmitir, como el tiempo que el receptor invierte en la lectura de cada unidad de datos.

Las tasas de baudios más comunes son 1200, 2400, 4800, 9600, 19200, 38400, 57600 y 115200. A mayor tasa, más rápido se envía y se recibe la transmisión, aunque también aumenta el riesgo de errores en la comunicación.

Cada bloque de datos transmitido se envía como un paquete de bits, que incluye, además de la información a transmitir, bits de sincronización y, a veces, de paridad.

Los bits de datos, que suelen ser de 5 a 9, siendo 8 lo más común, se transmiten de forma que el bit que se envía primero sea el menos significativo o LSB. Este paquete de bits, por lo general, suele representar un carácter del mensaje a enviar, que puede estar codificado en estándares como ASCII.

Los bits de sincronización son el bit de inicio y el bit (o bits) de parada, y marcan el comienzo y el final de un paquete de datos. El bit de inicio está indicado por una línea en espera que pasa de estado alto a bajo, mientras que los de parada marcan la transición de vuelta a la espera manteniendo un estado alto.

Por último, los bits de paridad constituyen una forma simple de comprobación de errores. Para producir dicho bit, se suman los bits de datos, y según el resultado sea par o impar, el estado será alto o bajo. No obstante, es común prescindir de su uso, optando por otros métodos de comprobación de errores más avanzados.

Una vez definida la estructura de los datos a enviar, se debe definir el medio. La comunicación en serie asíncrona se puede realizar con tan sólo dos cables, entre dos dispositivos que cuenten con dos pines para dicho fin: el RX, receptor, y el TX, transmisor.

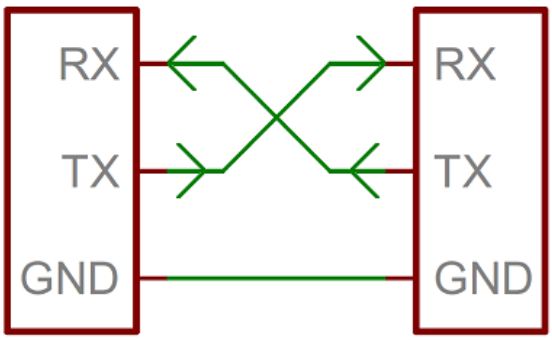

Figura 36: Ejemplo de una interfaz serie asíncrona

La Figura 36 muestra cómo se conectan los dispositivos. El RX de uno va al TX del otro, y viceversa. Este tipo de interfaces puede ser *full* dúplex, lo que significa que ambos dispositivos pueden enviar o recibir simultáneamente, o semi-dúplex, que significa que los dispositivos se tienen que turnar enviando y recibiendo. En el caso de que un dispositivo tan solo escuche, sólo habrá una línea TX a RX, y la interfaz será de tipo símplex.

Este tipo de comunicación se puede implementar de distintos modos, según sea la señal que se envíe, que depende del estándar. Una de las implementaciones más populares es la comunicación serie TTL.

La comunicación TTL se realiza con señales que se mueven dentro del rango de alimentación de los microcontroladores de 0 V a 3.3 o 5V. Una señal a nivel VCC (3.3 o 5 V) es un estado alto, es decir, o un bit 1, o un bit de parada o un indicador de que la línea está en espera. Una señal a nivel GND (0 V) representa o bien el bit de inicio, o un bit 0.

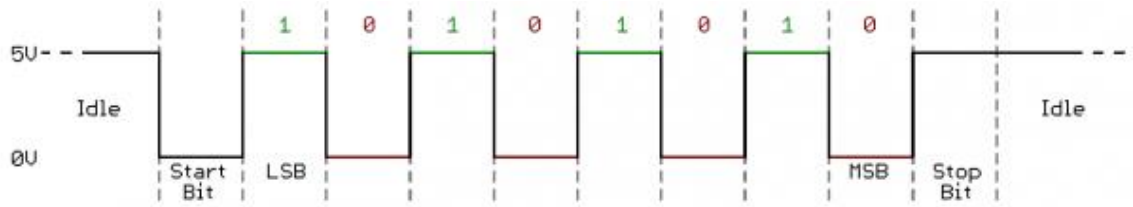

Figura 37: Ejemplo de una trama de comunicación serie TTL

Otro estándar popular es RS-232. En este caso, las señales suelen variar en un rango entre -13 V y 13 V, aunque las especificaciones permiten cualquier valor desde ±3 V a ±25 V. En este caso el voltaje bajo indica un bit 1, un bit de parada o que la línea está en espera, y un voltaje alto o un bit de inicio o un bit de datos 0.

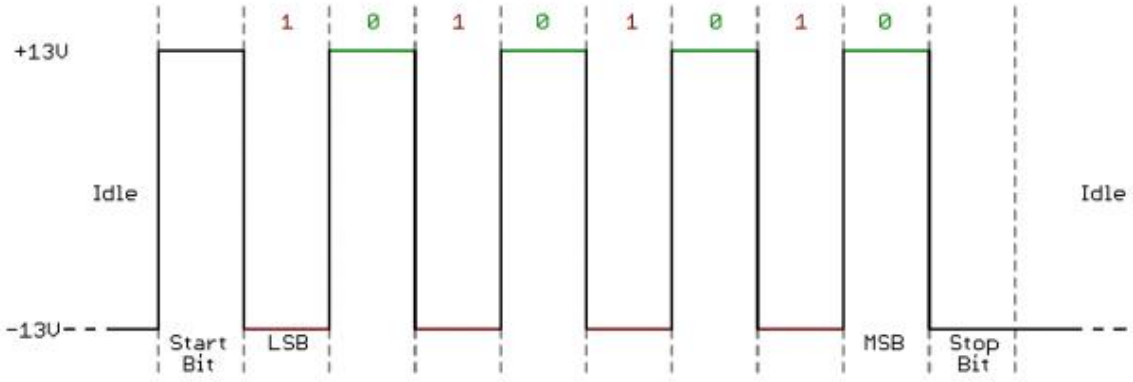

Figura 38: Ejemplo de una trama de comunicacíon serie RS-232

Por lo general, TTL es más fácil de implementar en sistemas embarcados, aunque los voltajes más bajos lo hacen más susceptible a las pérdidas en la transmisión frente a estándares como RS-232 u otros más complejos como RS-485.

A la hora de implementar este tipo de comunicación serie asíncrona, los microcontroladores y mini PCs cuentan con interfaces o puertos que envían y reciben los paquetes de datos. Un ejemplo muy común de esto es UART.

Un receptor/transmisor asíncrono universal (cuyas siglas en inglés corresponden a UART) es un bloque de circuitería presente en la mayoría de mini PCs y microcontroladores responsable de implementar la comunicación en serie, actuando como intermediario entre interfaces en serie y en paralelo. Así pues, en un extremo el UART es un bus paralelo que cuenta con una serie de líneas de datos y algunos pines de control, mientras que en el otro extremo es un bus en serie, con las conexiones RX y TX ya mencionadas.

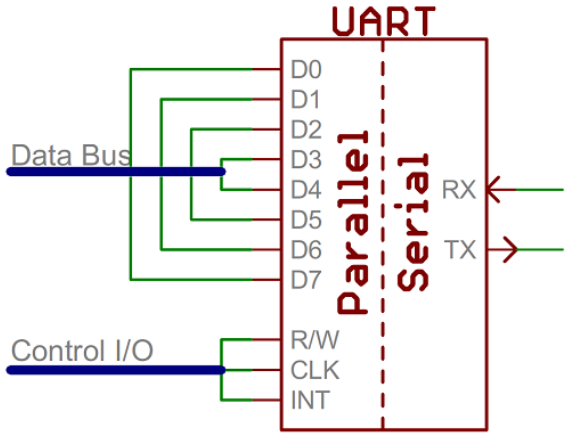

Figura 39: Ejemplo de una interfaz UART simplificada

La línea de transmisión del UART se encarga de crear los paquetes de datos, añadiendo los bits de sincronización y paridad indicados, y enviándolos por el TX en el tiempo establecido por la tasa de baudios. La línea de recepción, por otra parte, lee el RX teniendo en cuenta la tasa de baudios, extrae los bits de datos y pasa la información al bus paralelo.

Los UART más avanzados no sólo se encargan de traducir la información, sino que además cuentan con un búfer en el que almacenar aquello que se recibe, a la espera de que el procesador acceda a los datos.

En el caso de los mini PCs actuales, es muy común también que cuenten con puertos USB. Este tipo de puertos también permite la comunicación en serie asíncrona. Sin embargo, se trata de un estándar mucho más definido, que no sólo incluye el protocolo de comunicación, sino también cables, conectores e incluso la posibilidad de alimentar a los dispositivos.

A efectos prácticos, en la plataforma se ha dado un uso similar al USB al que se podría dar a una interfaz TTL-UART, por lo que no se considera necesario explicar de manera detallada las implicaciones de dicho estándar.

En el sistema quadrotor desarrollado, se ha utilizado USB para la conexión entre la placa STM y la Beagle, haciendo uso del único puerto disponible en ambas, mientras que para la conexión del módulo Xbee con el mini PC se ha utilizado el canal UART 2 de éste. El uso de USB para la

conexión con la STM ha permitido alcanzar tasas de baudios más altas (115200) de manera estable, lo que ha sido importante para lograr los objetivos establecidos en cuanto al tiempo de muestreo de las medidas para el bucle de control. En el caso de la Xbee, la comunicación se efectúa a 9600 baudios, aunque dado que no se trata de un elemento crucial, es aceptable. En ambos casos se ha prescindido del bit de paridad, y se ha utilizado un solo bit de parada.

6.3.2. Serie I2C

El protocolo de comunicación en serie I2C es un protocolo síncrono pensado para la comunicación a corta distancia de varios dispositivos o nodos esclavos con uno o más nodos maestros, mediante el uso, tan sólo, de dos cables.

Un nodo maestro es aquel que genera la señal de reloj e inicia la comunicación con los esclavos. Un nodo esclavo recibe el reloj y responde cuando un maestro lo requiere. I2C soporta la presencia de más de dos maestros en el bus. Sin embargo, no pueden comunicarse entre ellos y se han de turnar para hacerlo con los esclavos. La mayoría de dispositivos I2C pueden comunicarse a frecuencias entre 100 y 400 kHz.

Un bus I2C consta de dos señales: SCL y SDA. SCL es la señal de reloj y SDA la de datos. Si bien la señal de reloj es generada por el maestro que se esté comunicando en cada momento, los esclavos pueden forzar un estado bajo del reloj en ocasiones, para retrasar el envío de datos por parte del maestro o requerir más tiempo para preparar datos. Los dispositivos I2C sólo pueden forzar un estado bajo en la señal. Por ello, cada línea del bus cuenta con una resistencia de pull-up, a fin de devolver el estado alto cuando ningún dispositivo está asignando un estado bajo. La selección de dichas resistencia depende de los dispositivos conectados al bus.

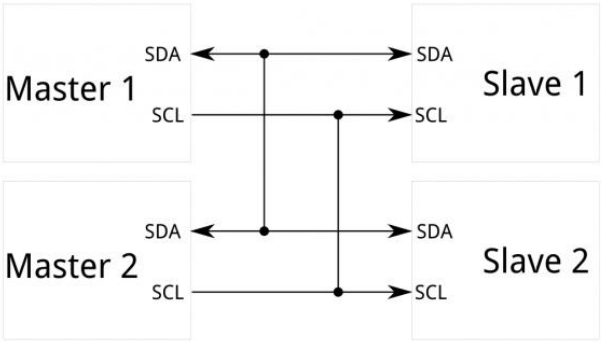

Figura 40: Ejemplo de un bus I2C

El protocolo de comunicación I2C es relativamente complejo en relación a otros como la comunicación en serie asíncrona. Los mensajes pueden ser de dos tipos: direcciones o datos. En los mensajes de dirección el maestro indica el esclavo al que se está refiriendo. Los mensajes de datos son paquetes de 8 bits que el maestro envía a un esclavo, o viceversa. Los

datos se ponen en la línea SDA cuando SCL pasa a estado bajo, y se leen cuando SCL sube. El tiempo entre el cambio en el reloj, y la lectura/escritura depende de los dispositivos conectados.

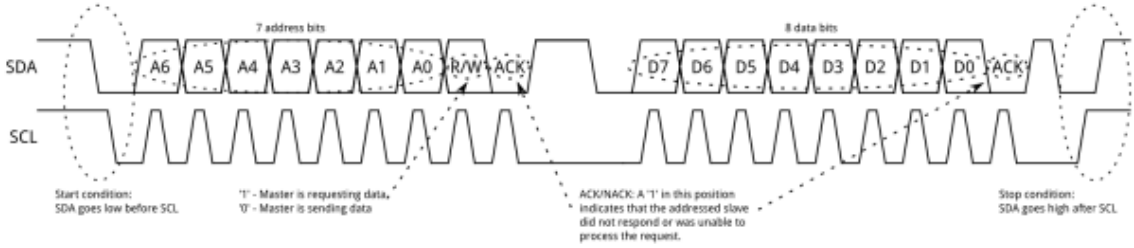

Figura 41: Ejemplo de una trama de comunicación I2C

Para iniciar un mensaje de dirección, el maestro deja SCL en alto y baja SDA. Esto avisa a los esclavos de que se va a iniciar una transmisión. Si dos maestros intentan iniciarla, el primero en bajar SDA obtiene el control del bus.

El mensaje de dirección es lo primero en cualquier secuencia de comunicación I2C. Para direcciones de 7 bits, la dirección se envía con el bit más significativo (MSB) primero y va seguida de un bit que indica si es una operación de lectura (1) o escritura (0).

El noveno bit del paquete es el bit NACK/ACK, que aparece tanto en las direcciones como en los datos. Una vez que los 8 bits anteriores se han enviado, el esclavo obtiene el control de SDA. Si éste no baja dicha línea, se entiende que no ha recibido el mensaje, o no lo ha podido interpretar, por lo que se corta la comunicación y el maestro ha de decidir cómo actuar.

Una vez que se ha enviado y recibido correctamente la dirección el esclavo, se pueden enviar los paquetes de datos. El maestro continuará enviando pulsos de reloj a un intervalo regular y los datos se pondrán en el SDA, por el esclavo o el maestro según sean operaciones de lectura o escritura, respectivamente.

Una vez que se han enviado todos los paquetes de datos, el maestro generará una condición de parada, que se define por un transición de cero a uno en el SDA tras una transición de cero a uno en el SCL, manteniendo el estado alto. Durante una operación de escritura normal, el estado de SDA no debería variar mientras SCL está en alto, para evitar falsas condiciones de parada.

El uso de este protocolo de comunicación en sistemas embarcados es muy común. En el caso de la plataforma quadrotor desarrollada, el protocolo de comunicación I2C se ha utilizado para la lectura de los sensores que incorpora el sistema. Por una parte, la placa STM cuenta con un bus I2C al que están conectado acelerómetro, giróscopo y magnetómetro. Por otra parte, en el caso del sensor de ultrasonidos SFR10 se ha utilizado el segundo bus I2C del mini PC BeagleBone Black para comunicarse con él.

7. Software de la plataforma

En este apartado se describe el software desarrollado para lograr el funcionamiento de la plataforma quadrotor planteada. Así pues, a continuación se explican de manera sencilla los programas que incluye el proyecto, tanto los correspondientes al mini PC Beagle y la placa STM, que constituyen el software embarcado, como el correspondiente a la interfaz de usuario de la estación de tierra.

Además, se comentan algunos otros aspectos, como el lenguaje de programación utilizado en cada caso, así como los entornos de desarrollo elegidos para la escritura de código y posterior compilación.

7.1. Software de la BeagleBone Black

Como ya se ha mencionado con anterioridad, el BeagleBone Black es un mini PC de arquitectura ARM compatible con sistemas operativos en base Linux. Concretamente, la unidad montada en la plataforma cuenta con la versión Debian 7 instalada en su memoria eMMC. Teniendo en cuenta esto, se optó por desarrollar el software desde un PC que contase también con Linux, a fin de trabajar con las mismas librerías que en el dispositivo embarcado, y tan solo tener en cuenta el cambio de arquitectura a la hora de compilar, mediante el uso de herramientas o *toolchains* de compilación cruzada.

Así pues, el entorno de desarrollo o IDE utilizado ha sido Code::Blocks, software libre orientado, esencialmente, al desarrollo en los lenguajes C y C++. Entre las ventajas que presentaba este entorno se encuentran la facilidad a la hora de configurar el compilador, o la ayuda en el desarrollo del código (notificación de errores con posibles soluciones, sugerencias en la escritura de código, orientación a la hora de trabajar con punteros…), además de su uso muy extendido (mucha información disponible en internet, así como solución de problemas) unido a la experiencia previa de otros miembros del laboratorio que también lo habían utilizado. La Figura 42 muestra el aspecto de este IDE en el ordenador utilizado durante el proyecto, cuyo sistema operativo era Ubuntu 14.04 LTS.

Para compilar los códigos se han utilizados las *toolchains* de Linaro, disponibles también de forma libre en internet, y que permiten la compilación cruzada a arquitecturas ARM, desde sistemas como el utilizado, de arquitectura Intel, por ejemplo. Los resultados obtenidos con dichas herramientas han sido satisfactorios, sin problemas a la hora de ejecutar los programas en la Beagle.

Si bien el desarrollo de los programas de la Beagle se inició en C, al final se tuvo que adaptar el código al lenguaje C++, extensión del primero que permite la programación orientada a objetos. Esto da la posibilidad programar de forma concurrente, es decir, mediante tareas que se ejecutan de forma paralela sin tener que esperar las unas a las otras e interactúan entre sí, siendo los objetos el soporte de las variables con las que trabaja cada uno de los procesos.

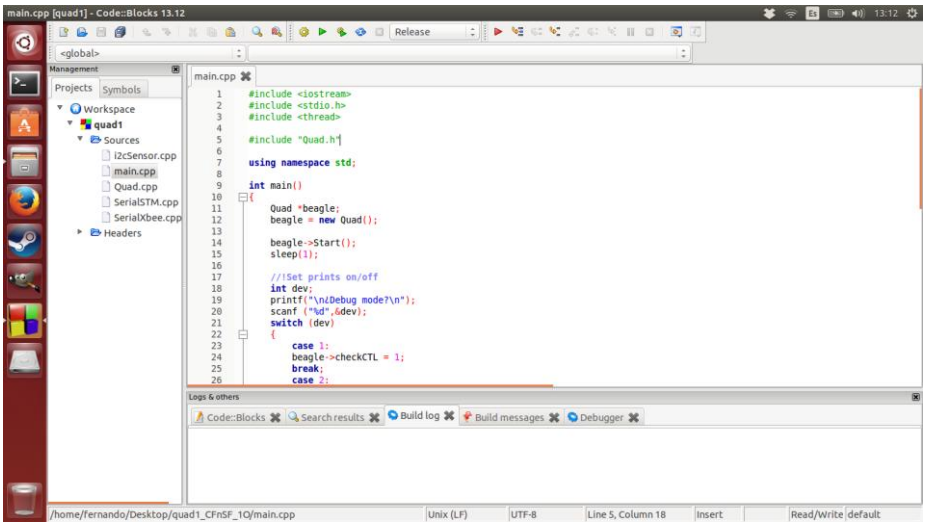

Figura 42: Aspecto de Code::Blocks IDE

Para lograr dicha estructura concurrente se han utilizado algunas clases disponibles en las librerías de C++11, segunda iteración más reciente del lenguaje. En concreto, las más importantes han sido std::thread, clase que sirve para representar un hilo de ejecución, y std::mutex, clase que permite la protección del acceso a partes del código con las que trabajan distintas tareas o threads. Por las características de su funcionamiento, estos últimos suelen denominarse semáforos. El uso de semáforos permite bloquear partes del código cuando uno de los threads accede a una de las zonas protegidas por éstos. Así se evita, por ejemplo, la lectura o escritura simultánea de varibles a las que pueden acceder dos o más threads.

Una vez compilados los códigos, Code::Blocks genera un archivo binario que se puede copiar a la Beagle mediante el protocolo de comunicación SSH, el cual permite el acceso a máquinas remotas a través de una red. Este archivo se puede ejecutar en el mini PC gracias a la *toolchain* utilizada, y comprobar su funcionamiento.

El BeagleBone Black constituye el centro en torno al cual gira el software de la plataforma. Se encarga de ejecutar los algoritmos de control y comunicarse con el resto de dispositivos, que obtienen las lecturas y parámetros necesarios para su función principal. Cada una de las tareas que realiza la Beagle se ejecutan de forma concurrente mediante el uso de los threads ya mencionados.

El programa ejecutado por la Beagle se organiza mediante una serie de objetos, cada uno relacionado con cada una de las tareas a efectuar, que incluyen tanto las variables relacionadas con ésta, como distintas funciones que definen tanto de acciones para inicializar o cerrar dicha tarea, así como el bucle a ejecutar periódicamente por el thread. La Figura 43 muestra de manera esquemática los objetos utilizados en la versión actual del código, así como su función principal y las variables que maneja cada uno. Se puede observar que hay un objeto principal, encargado del bucle de control, y una serie de objetos, que interactúan con el primero, y se encargan de manejar las comunicaciones con los dispositivos.

El objeto central encargado del bucle de control es el que crea, al comienzo del programa, el resto de objetos, esto permite el acceso de manera sencilla a las variables que manejan éstos, que incluyen parámetros cruciales como las medidas de la IMU, o las referencias, por ejemplo.

Figura 43: Organización en objetos del programa del mini PC Beagle

Al inicio de la ejecución del programa, el thread principal crea dicho objeto, denominado *beagle*, y llama a una función interna a éste que se encarga de la creación del resto de objetos, *stm*, *xbee* y *sfr10*, los cuales manejan las comunicaciones con el dispositivo correspondiente, para finalmente inicializarlas. Tras ello, se lanzan los threads que se encargan de los bucles de comunicación y, posteriormente, el que se encarga de los algoritmos de control.

Una vez se han lanzado todos los threads, el hilo principal queda a la espera, mientras que las tareas que constituyen el programa se van ejecutando de manera concurrente. A continuación se explica la función de cada una de las tareas, así como la secuencia de acciones que constituye cada una.

7.1.1. Comunicación con la STM32F3 Discovery

El hilo de comunicación con el microcontrolador STM se encarga tanto del envío del ciclo de trabajo de la señales PWM de cada uno de los motores como de la recepción de las lecturas de la IMU incorporada en dicha placa. Así pues, el objeto correspondiente incluye variables para cada una de las tres lecturas del acelerómetro, del giróscopo y del magnetómetro, así como el *duty* de los cuatro motores.

Anteriormente al lanzamiento del thread correspondiente, se llama a una función de inicialización. Esta función maneja la apertura del puerto serie correspondiente y su configuración. En este caso el puerto correspondiente a la placa STM, al conectarla por USB a la Beagle es */dev/ttyACM0*, según la nomenclatura del sistema de archivos virtual de Linux, que permite al acceso al hardware. Dicho puerto se configura a una tasa de 115200 baudios, con 8 bits de datos, sin paridad y un solo bit de parada (parámetros ya descritos en el apartado anterior de la memoria). Tras ello, el puerto queda preparado para comunicarse de manera correcta con la STM.

La secuencia de acciones de la función correspondiente al bucle de comunicación se puede observar en la Figura 44.

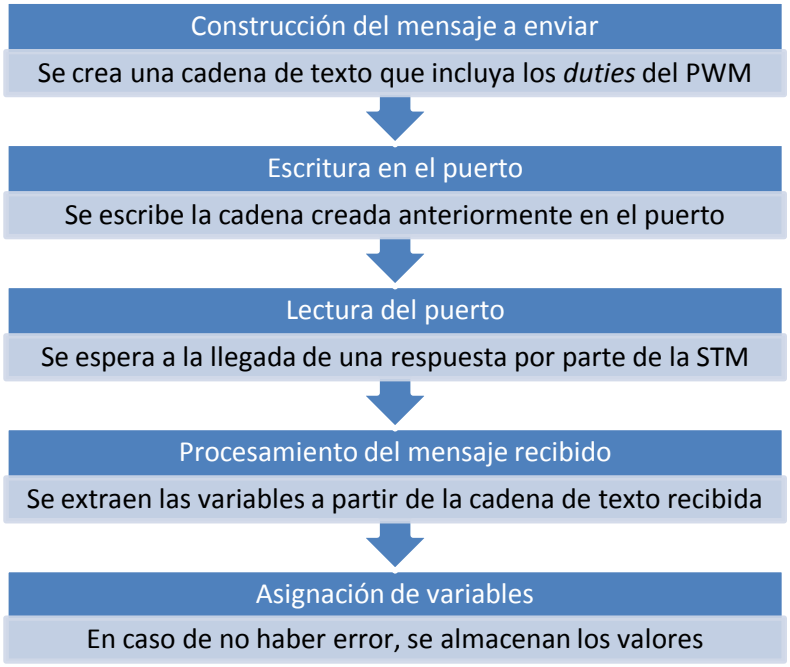

Figura 44: Secuencia de la comunicación con la STM en el programa de la Beagle

Esta función comienza con el lanzamiento del thread correspondiente y se ejecuta de manera periódica hasta el final de la operación del quadrotor.

En primer lugar se construye el mensaje que se le ha de enviar a la STM, una cadena de texto que incluye los cuatro ciclos de trabajo del PWM, a partir de los resultados asignados por el bucle de control (o unos valores iniciales, en caso de que éste no se haya iniciado aún, que

mantienen los motores parados). Esta cadena se escribe en el puerto, enviándose a la STM. Una vez enviado el mensaje, se inicia la lectura del puerto, a la espera de la respuesta del microcontrolador, que será otra cadena de texto que incluye las nueve medidas de la IMU, separadas por comas.

Este mensaje se ha de procesar a fin de obtener valores útiles que almacenar en las variables. Para ello se utiliza una función de C que permite cortar una cadena de texto en distintas cadenas o *tokens* más pequeñas, especificando el separador, denominada *strtok*. Hay que tener especial cuidado con esta función, puesto que si el mensaje, debido a algún error de comunicación, tiene menos *tokens* de los esperados, devolverá un puntero nulo como respuesta, lo que si no se tiene en cuenta, puede llevar a una violación de acceso o *segmentation fault*, que detenga la ejecución del programa, parando en seco los motores con el quadrotor en vuelo.

Así pues, la asignación de los nuevos valores sólo se efectúa en caso de que el mensaje incluya los nueve *tokens* necesarios, a fin de evitar un intento de acceso a una dirección de memoria nula. Esta asignación se realiza tras transformar en valores numéricos las nueves cadenas de texto obtenidas. Una vez asignados los valores (o sin asignarlos, en caso de que haya habido algún error) el bucle comienza de nuevo.

En la versión actual del código, se ha logrado que este bucle se ejecute en torno a los 2.5 ms, lo que da margen para lograr el objetivo de 5 ms en el bucle de control. En cuanto a los errores de comunicación, son poco comunes, fallando como mucho en torno a unas diez veces no consecutivas en períodos de prueba del programa de hasta media hora, por lo que no afectan en absoluto al vuelo seguro de la plataforma.

Por último, el objeto también incluye una función para finalizar las comunicaciones y cerrar el puerto.

7.1.2. Comunicación con el sensor de ultrasonidos

El hilo de comunicación con el sensor de ultrasonidos SFR10 se encarga de la recepción de las lecturas correspondientes a la altura de la plataforma quadrotor.

Anteriormente al lanzamiento del thread correspondiente, se llama a una función de inicialización. Esta función maneja la apertura del puerto I2C correspondiente y su configuración. En este caso el puerto correspondiente al sensor, conectado al segundo bus I2C de la Beagle, es */dev/i2c-1*, según la nomenclatura del sistema de archivos virtual de Linux. Tras abrir el puerto, se especifica la dirección del dispositivo I2C al que dirigirse, en este caso 0x70. Puesto que el resto del tiempo se va a continuar la comunicación con el mismo dispositivo, ya que no hay ningún otro conectado al mismo bus, no es necesario especificar la dirección de manera periódica en el bucle.

La secuencia de acciones de la función correspondiente al bucle de comunicación se puede observar en la Figura 45.

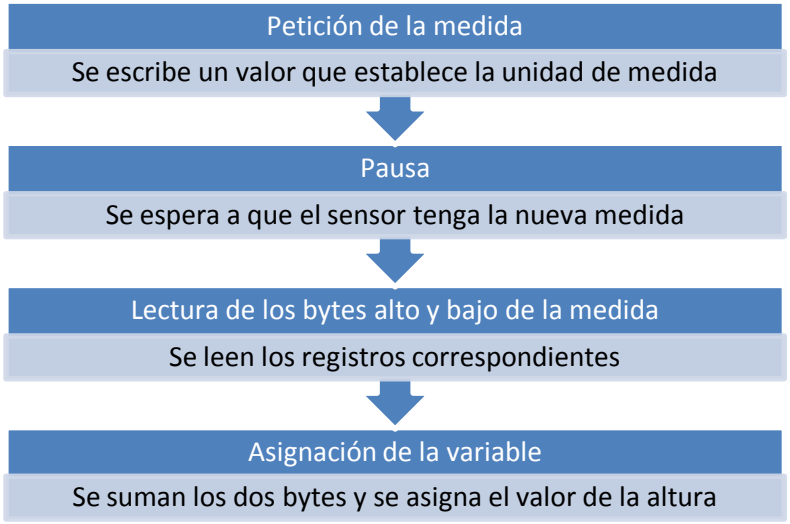

Figura 45: Secuencia de la comunicación con el SFR10 en el programa de la Beagle

Esta función comienza con el lanzamiento del thread correspondiente y se ejecuta de manera periódica hasta el final de la operación del quadrotor.

En primer lugar se escribe en el puerto la dirección del registro 0x00 y, después, se escribe el valor a asignar a dicho registro, 0x51, lo que especifica que se desea una nueva medida en centímetros. Tras ello, se espera 65 ms, que es el tiempo máximo que tardará el sensor en tener la nueva medida.

Así, se puede proceder a la lectura de los dos registros (la lectura tiene un randgo de dos bytes) que almacenan el valor de la medida. Primero se escribe 0x03 en el puerto, y se lee el byte alto, después se escribe 0x02, y se lee el byte bajo. Tras ello, se suman los bytes de manera correcta, teniendo en cuenta lo que representa cada uno y se asigna a la variable que representa la altura del quadrotor.

Finalmente, el objeto también incluye una función para finalizar las comunicaciones y cerrar el puerto.

7.1.3. Comunicación con la estación de tierra

El hilo de comunicación con el módulo Xbee (y mediante éste con la HMI de la estación de tierra) se encarga de la recepción de los parámetros y referencias del control. Así pues, el objeto correspondiente incluye variables para los parámetros de los algoritmos de control de cada uno de los grados de libertad a controlar, así como las referencias deseadas.

Anteriormente al lanzamiento del thread correspondiente, se llama a una función de inicialización. Esta función activa el segundo canal UART de la placa, desactivado al inicio, y maneja la apertura del puerto serie correspondiente, así como su configuración. En este caso el puerto correspondiente al módulo Xbee, al conectarlo al UART2 a la Beagle es */dev/ttyO2*, según la nomenclatura del sistema de archivos virtual de Linux. Dicho puerto se configura a

una tasa de 9600 baudios, con 8 bits de datos, sin paridad y un solo bit de parada. Tras ello, el puerto queda preparado para comunicarse de manera correcta con la HMI.

La secuencia de acciones de la función correspondiente al bucle de comunicación se puede observar en la Figura 46.

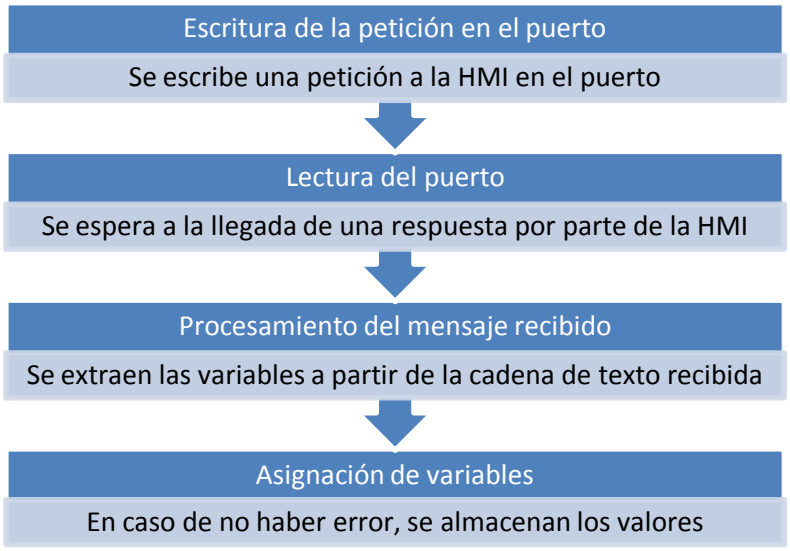

Figura 46: Secuencia de la comunicación con la HMI en el programa de la Beagle

Esta función comienza con el lanzamiento del thread correspondiente y se ejecuta de manera periódica hasta el final de la operación del quadrotor. La operación de este puerto es análoga a la del puerto correspondiente a la STM, salvo que en este caso el mensaje que se escribe es fijo, ya que no se envía información a la HMI y tan solo se ha de realizar la petición. El procesamiento del mensaje recibido es igual al de la otra tarea.

De nuevo, el objeto también incluye una función para finalizar las comunicaciones y cerrar el puerto.

7.1.4. Bucle de control

El último hilo a considerar en el programa de la Beagle es el que ejecuta el control de la plataforma. Como ya se ha comentado, el objeto correspondiente a esta tarea crea al resto de objetos y, por lo tanto, puede acceder a sus variables. Esto permite actualizar el valor de las lecturas de los sensores, las referencias y los parámetros de control en cada iteración de forma sencilla.

En el desarrollo de este bucle, se calcula la orientación de la plataforma mediante el filtrado correspondiente, se calculan las acciones de control mediante las leyes utilizadas y se obtienen los *duties* del PWM. Además, también se van almacenando los distintos valores de cada iteración en un fichero de texto, a fin de poder trabajar con ellos después, para graficarlos, por ejemplo.

Previamente al inicio del bucle propiamente dicho, se inicializan las variables, se abre el fichero de texto mencionado y se calcula la orientación inicial del quadrotor. Tras ello, se inicia el bucle de control, cuya secuencia de acciones se puede observar en la Figura 47.

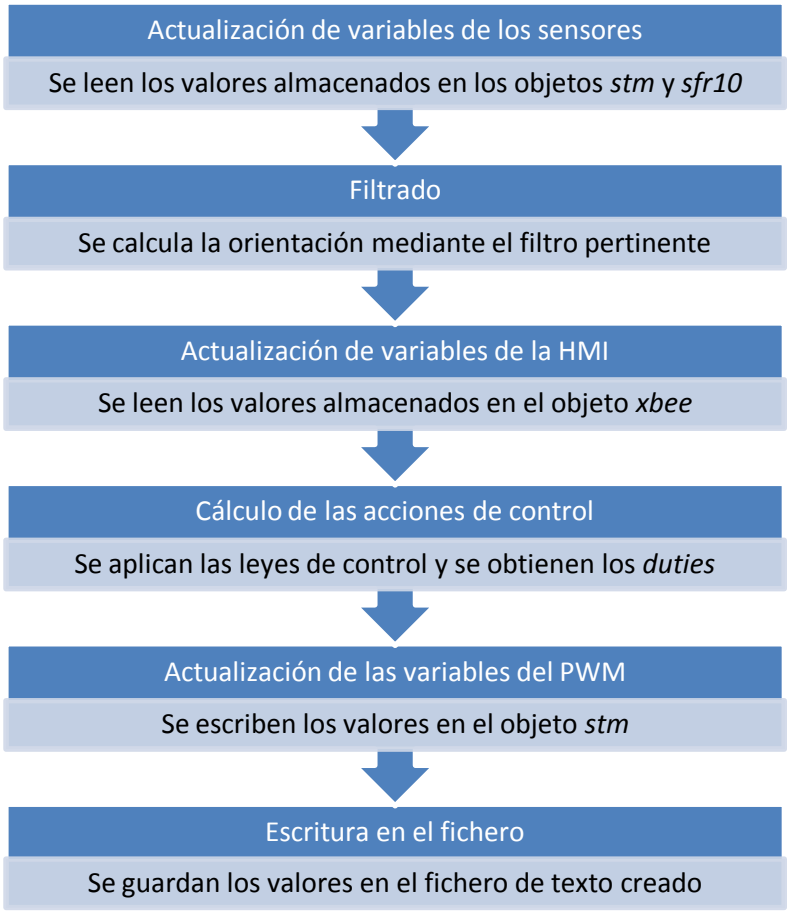

Figura 47: Secuencia del bucle de control en el programa de la Beagle

En primer lugar, se actualizan las variables internas de la tarea con los valores de las lecturas de los sensores presentes en el mismo instante en los objetos correspondientes. El acceso a estos parámetros está protegido mediante los semáforos ya mencionados en ambos threads, a fin de evitar lecturas y escrituras simultáneas.

Con los valores correspondientes a la nueva iteración, se procede a calcular los ángulos del quadrotor, por una parte, mediante las lecturas de acelerómetro y magnetómetro y, por otra, mediante las del giróscopo. En la versión actual del programa se utiliza un filtro complementario para la fusión de dichas medidas. En cuanto a la derivada de los ángulos, se asigna directamente la lectura del giróscopo, casi carente de ruido, obteniendo unos resultados satisfactorios. Los valores de orientación obtenidos se almacenan en variables correspondientes al instante anterior, para la próxima iteración del filtro.

Tras ello, se actualizan las variables internas correspondientes de la tarea con los valores de referencias y constantes de los algoritmos de control presentes en el objeto correspondiente, también haciendo uso de semáforos.

Así, se puede proceder al cálculo de las acciones de control correspondientes a cada grado de libertad, según las leyes establecidas (que se comentan en el próximo apartado de la memoria). Los ciclos de trabajo de los PWM se calculan añadiendo las acciones de control a un *duty* global, que también se especifica desde la HMI. Después, se actualizan los valores del PWM presentes en el objeto de la STM, para que sean enviados.

Finalmente, se almacena el resultado de las variables de interés en el fichero de texto mencionado, para su posterior procesamiento.

Actualmente, el bucle de control se ejecuta aproximadamente a los 5 ms planteados como objetivo. Esto permite lograr un buen control de la plataforma, con una respuesta muy suave y muy poca oscilación ante las perturbaciones en el estado del sistema.

7.2. Software de la STM32F3 Discovery

En la plataforma desarrollada, el microcontrolador STM32F3 *Discovery* se ha utilizado para la lectura de la IMU que incorpora, así como el envío de las señales PWM a los motores. Para la programación de esta placa se ha utilizado el IDE Keil uVision 5, cuyo aspecto se puede observar en la Figura 48. Este IDE cuenta con una gran cantidad de librerías que permiten acceder de forma sencilla a muchas de las funcionalidades de la STM. El programa se ha escrito en lenguaje C, de manera secuencial.

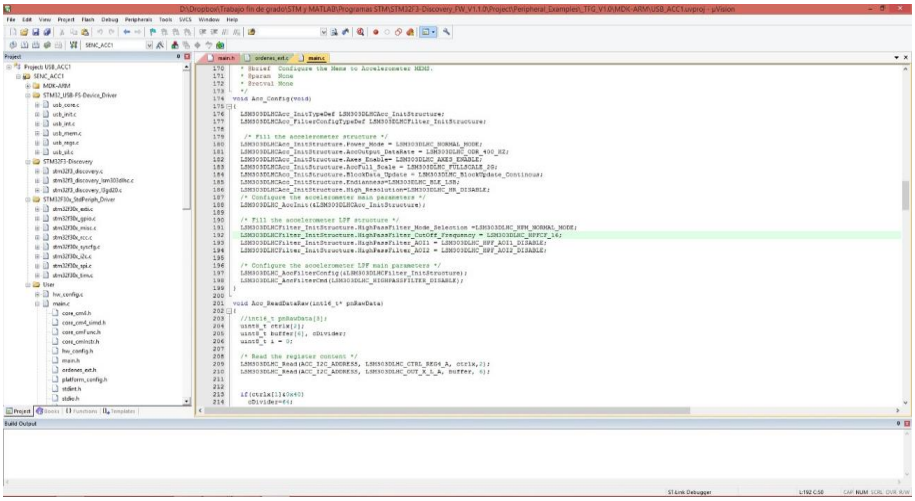

Figura 48: Aspecto de Keil uVsion 5 IDE

A continuación se comentan brevemente las dos funciones principales del programa de la STM. Puesto que el desarrollo de dicho software fue tarea, principalmente, del compañero de

desarrollo del proyecto, Norberto Vera Vélez, se puede consultar su memoria para obtener más detalles al respecto.

7.2.1. Lectura de la IMU

La comunicación con los sensores que incluye la placa STM se realiza mediante I2C, protocolo descrito en el apartado anterior de la memoria y utilizado también en la lectura del sensor de ultrasonidos SFR10. El funcionamiento es similar al ya descrito, sólo que con valores distintos para la dirección de los dispositivos, así como para los registros a los que se ha de acceder.

La dirección I2C de serie para el acelerómetro es 0b0011001x, donde x es un bit que indica si la operación es de lectura (1) o escritura (0). Los registros correspondientes a las medidas de aceleración son: byte bajo x 0h28, byte alto x 0h29, byte bajo y 0h2A, byte alto y 0h2B, byte bajo z 0h2C, byte alto z 0h2D. Estas direcciones están escritas en formato hexadecimal y se corresponden con una dirección binaria de 8 bits.

Por otro lado, la dirección de serie para el magnetómetro es 0b0011110x. Los registros correspondientes a las lecturas de campo magnético son: byte alto x 0h03, byte bajo x 0h04, byte alto z 0h05, byte bajo z 0h06, byte alto y 0h07, byte bajo y 0h08. En este caso, los registros no están ordenados, aspecto a tener en cuenta en el código.

La dirección I2C del giróscopo es 0b110101xy, donde el valor de x es 1 si el pin SDO del sensor L3GD20 está conectado a alimentación y 0 en caso contrario. La y, al igual que en los casos anteriores, se corresponde con el bit de lectura/escritura. Las direcciones de los registros correspondientes a las velocidades angulares leídas por los giróscopos son: byte bajo x 0h28, byte alto x 0h29, byte bajo y 0h2A, byte alto y 0h2B, byte bajo z 0h2C, byte alto z 0h2D.

En las primeras versiones del programa la lectura de los registros estaba planteada de forma ineficiente, siendo los tiempos de lectura de más de 6 ms en ocasiones. Actualmente, las modificaciones en el código permiten leer todos los sensores en torno a los 2 ms.

7.2.2. Generación de señales PWM

La placa STM se encarga también de generar las señales PWM que se envían a los ESC de los motores, haciendo uso de uno de los relojes o *timers* disponibles en el microcontrolador, así como de funciones de comparación que establecen el ciclo de trabajo de la señal, a partir de los valores enviados por la Beagle.

Una señal PWM es una onda cuadrada cuya anchura de pulso se modula, variando así el valor medio de la onda. Esto permite, por ejemplo, cambiar la velocidad de giro de un motor de corriente continua, o incluso también de alterna, mediante el uso de ESC.

La Figura 49 muestra un ejemplo de la salida por uno de los puertos dedicados al PWM en la STM, haciendo uso de un osciloscopio para visualizar la forma de la onda.

Figura 49: Vista en el osciloscopio de una señal PWM generada por la placa STM

Actualmente, el programa genera señales PWM de una frecuencia de 400 Hz, con una resolución de 2000 unidades, lo que permite lograr un comportamiento suave de los motores, con bastante sensibilidad.

7.3. Software de la estación de tierra

Para el desarrollo del software de la estación de tierra, que se quería que fuese compatible con ordenadores con sistema operativo Windows, se ha utilizado el IDE Microsoft Visual Studio Community 2013, cuyo aspecto se puede apreciar en la Figura 50.

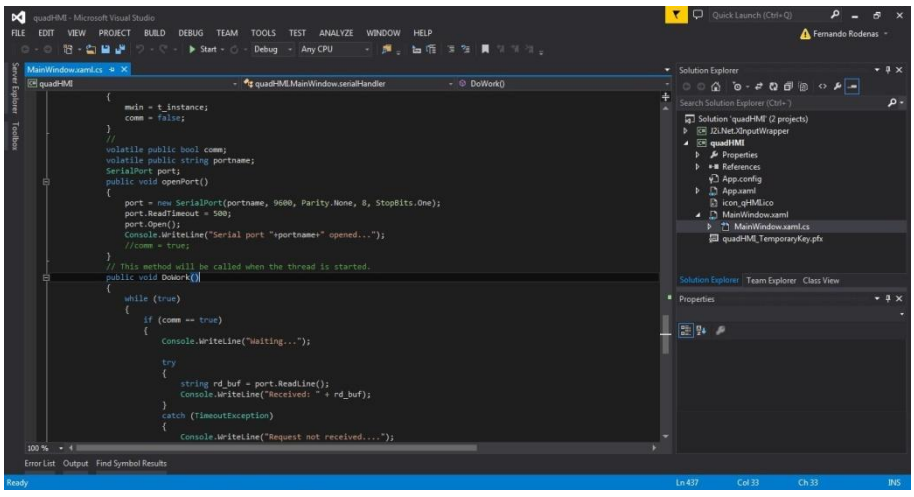

Figura 50: Aspecto de Microsoft Visual Studio Community 2013 IDE

Entre las razones por las que se optó por este entorno se encuentran la gran variedad de lenguajes compatibles, las sugerencias de código, las ayudas visuales que permiten conocer de forma rápida el tipo de cada variable o el depurador con el que cuenta, por ejemplo.

En concreto, el lenguaje utilizado para el desarrollo del programa ha sido C#. Este lenguaje es bastante similar a Java (con el que ya se tenía algo de experiencia previa), ambos de más alto nivel que C++. La ventaja de su uso sobre este último en el caso de la HMI es que es compatible, de serie, con *Windows Presentation Foundation* (WPF), un entorno de herramientas creadas por Microsoft para el desarrollo de aplicaciones de escritorio de Windows, que permite crear interfaces gráficas de usuario de manera sencilla.

Además, para facilitar la interacción mediante código con los dispositivos de entrada compatibles con la API XInput, se ha hecho uso de una librería de clases en C#, J2i.Net.XInputWrapper, disponible de forma libre en internet, que permite un acceso sencillo al estado de los diversos botones y *sticks* que incorporan estos periféricos.

El software de la HMI se divide en dos partes. Por un lado, la parte visible, la interfaz gráfica, que permite la interacción del usuario con el sistema, mediante el uso de botones y campos de texto, así como los dispositivos XInput mencionados. Por otro lado, el código que se ejecuta en segundo plano y que se encarga de actualizar las variables en función de lo que se haga sobre la interfaz y de mantener la comunicación con el módulo Xbee conectado al PC y, con éste, al quadrotor.

En lo que al código se refiere, gran parte de éste está destinado al procesado de la entrada de los controladores XInput, que se ha de convertir, en última instancia, en los valores de referencia para el control de la plataforma. Otra parte se encarga de definir las funciones que son llamadas al hacer click en los distintos botones de la interfaz.

Una parte vital es la comunicación con el Xbee. Se ha definido de manera similar a la Beagle, creando un objeto que la maneje y lanzando un thread que ejecute la tarea periódicamente, con los cambios pertinentes por el distinto lenguaje utilizado. La configuración del puerto, tal y como requiere el protocolo, es la misma en ambos dispositivos.

El bucle de comunicación espera a la petición por parte de la Beagle, al recibirla genera una cadena de texto a partir de los valores que se han de enviar (referencias y parámetros de control), y la escribe en el puerto. En las primeras versiones, a veces la comunicación se perdía porque ambos dispositivos quedaban al mismo tiempo a la espera de un mensaje del otro. Para evitar esto, se estableció un tiempo máximo de espera en la HMI, pasado el cual se reinicia la comunicación, solucionando así el problema.

Otro aspecto incorporado en el código es la posibilidad de almacenar en un fichero de texto los valores de las constantes de los algoritmos de control, a fin de poder recuperarlas rápidamente en usos posteriores de la HMI.

A continuación, se explica brevemente el funcionamiento de la interfaz gráfica de usuario desarrollada. Como apoyo a la interfaz, en la versión actual se cuenta además con una ventana adicional que muestra las salidas que se escriben en consola, lo que permite monitorizar la comunicación con el quadrotor.

7.3.1. Funcionamiento de la interfaz de usuario

La Figura 51 muestra el aspecto de la interfaz gráfica de la estación de tierra.

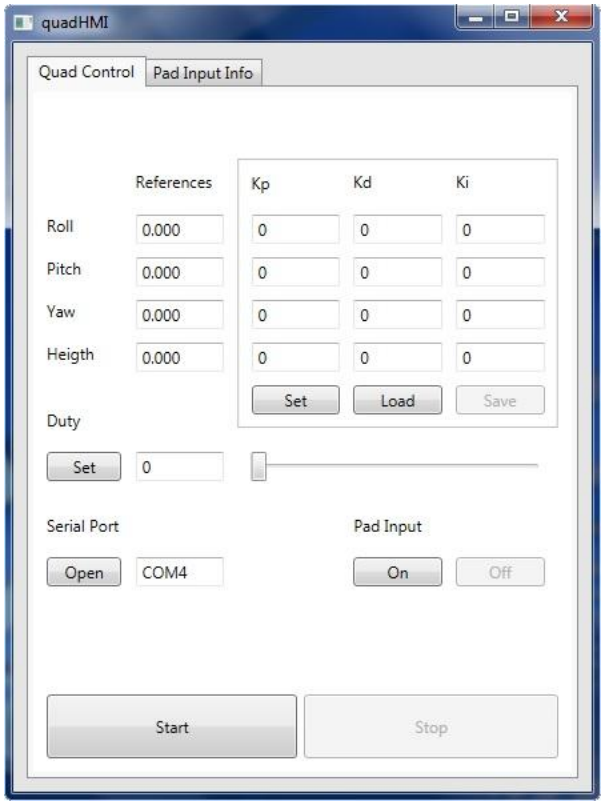

Figura 51: Aspecto de la interfaz de usuario desarrollada

La interfaz de usuario permite la escritura de los parámetros de las leyes de control mediante campos de texto dinámicos. Tras variar los valores, estos se actualizan pulsando el botón *Set* correspondiente. Los botones *Save* y *Load* permiten guardar los últimos valores aplicados o cargar los últimos guardados respectivamente.

El ciclo de trabajo global de los motores se puede modificar con un *slider*, pulsando posteriormente el botón *Set* que corresponde.

Además, una vez se activa la entrada XInput, con el botón *On*, bajo *Pad Input*, las referencias, en el caso de usar un mando de Xbox 360, se modifican de la siguiente manera:

- Roll: Stick izquierdo, eje X.
- Pitch: Stick izquierdo, eje Y.
- Yaw: Gatillos izquierdo y derecho.
- Altura: Stick derecho, eje Y.

También se puede modificar el *duty* global haciendo uso de los botones A y B del mando mencionado.

En el campo de texto bajo *Serial Port* se debe especificar el puerto COM al que se encuentra conectado el módulo Xbee, y posteriormente pulsar *Open*, para abrir dicho puerto. Tras ello se puede iniciar la comunicación con el quadrotor pulsando *Start*. En caso de querer detenerla, tan solo hay que pulsar *Stop*.

8. Control de la plataforma

8.1. Configuración de control

En el tercer apartado de la memoria se ha desarrollado el modelo teórico de un quadrotor genérico cuyo sistema de referencia ligado al cuerpo tiene los ejes x e y alineados con los brazos de la estructura. Esto condiciona las expresiones a las que se llegan finalmente, ya que varía la influencia de las acciones de control sobre cada uno de los motores.

Esta configuración de control se denomina control en cruz (+). Como alternativa a este planteamiento, y cada vez con un uso más extendido en el control de estabilidad de quadrotors, se puede plantear una configuración de control en equis (x), en la que los ejes cuerpo están girados 45° respecto al eje z.

La Figura 52 muestra ambas configuraciones de manera esquemática, pudiéndose apreciar la diferencia en cuanto a la orientación del sistema de referencia ligado al cuerpo en ambas, si se supone que la dirección de vuelo coincide con los ejes x respectivos.

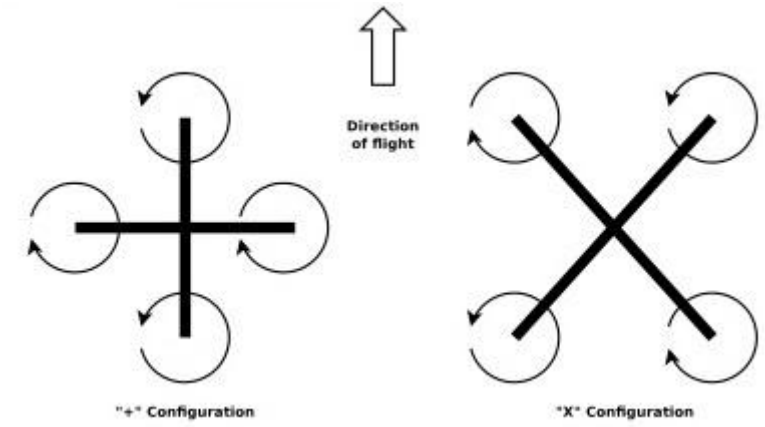

Figura 52: Esquemas de control en cruz (+) y control en equis (x)

En el caso del control en equis, puesto que los ejes cuerpo no están alineados con los brazos de la estructura, cada motor genera tanto momento de cabeceo como de alabeo, lo que implica que todas las acciones de control actuarán sobre todos los motores, pues.

No obstante, el modelo teórico es el mismo en ambos casos, ya que independientemente de la configuración de control, lo único que cambia es la definición de los ángulos que determinan la actitud del quadrotor y las entradas de control de cada uno de ellos, en las que ahora intervienen los cuatros, para todos los casos.

Las consecuencias de aplicar esta configuración en una plataforma quadrotor (que se supone simétrica radialmente en el plano $x - y$), en contraposición a una configuración en cruz, son:

- Inercias respecto a los ejes x e y similares. Si bien ahora hay dos motores a cada lado el eje, la distancia de éstos al eje se reduce lo que compensa el aumento de inercia, manteniendo valores similares a la configuración en cruz.
- Mayor momento de alabeo y cabeceo generado por los motores. Para la misma fuerza generada por cada motor, los momentos aumentan más de un 40%. Se obtiene una respuesta más rápida y con mayor rango de acción ya que es más fácil generar los momentos necesarios.
- Mayor complejidad matemática. Para esta configuración, todos los motores intervienen en todas las acciones de control, lo que lleva a expresiones matemáticas más extensas.
- Control más estable en *yaw*. Al realizar un movimiento con esta configuración, se actúa sobre los cuatro motores. Esto permite contrarrestar o, al menos, suavizar comportamientos no lineales que aparecen con lo configuración en cruz al acelerar un motor, y frenar el opuesto.
- Comportamiento más suave. Esta configuración permite lograr un vuelo más estilizado, con una respuesta dinámica más suave.

Teniendo en cuenta estos aspectos, se decidió apostar por la configuración de control en equis en la plataforma desarrollada. Además, en caso de montar una cámara más adelante, se podría aprovechar parte de la góndola de la batería para anclarla, y solo la configuración en equis permitiría grabar en la misma dirección de vuelo.

8.2. Ley de control teórica

Una vez conocido el modelo teórico de los sistemas quadrotor y elegida la configuración de control, se han implementado los algoritmos necesarios para el control de la plataforma en actitud y altura. La Figura 53 muestra un esquema simplificado del control de la plataforma.

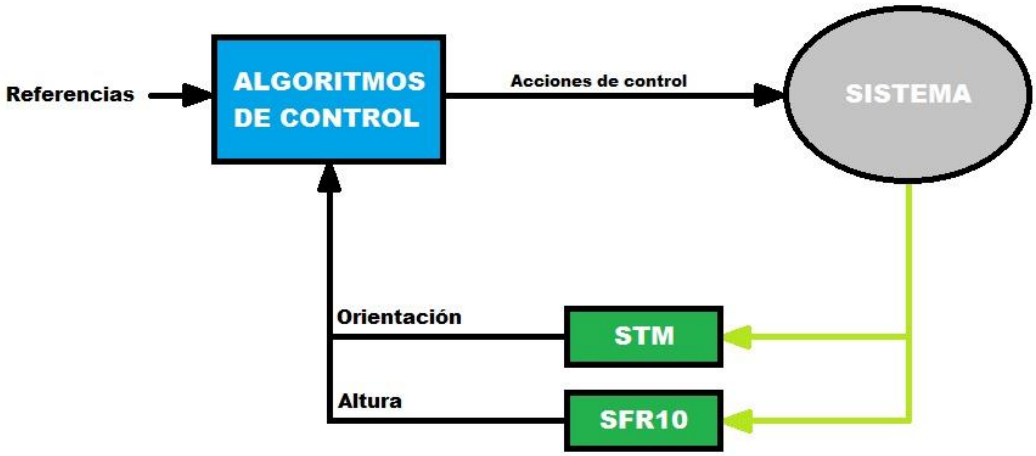

Figura 53: Esquema de control de la plataforma

Para dicho control se ha optado por el uso de algoritmos PD. Los algoritmos PD, que se enmarcan en el grupo de los PID, constituyen un mecanismo de control de implementación sencilla, que consta de dos parámetros:

- Un componente proporcional que tiene en cuenta la desviación entre el valor medido en cada instante y el deseado, o referencia.
- Un componente derivativo que se anticipa a la futura desviación de la variable, a partir de su derivada.

En un principio, no se ha considerado necesario incluir la componente integral con la que cuentan los controles PID completos.

Los algoritmos utilizados presentan la siguiente forma:

$$u_\alpha = k_{p_\alpha}\left(\alpha_{ref} - \alpha\right) + k_{d_\alpha}\,\dot{\alpha} \tag{32}$$

donde α es la variable a controlar, que en el caso de la plataforma desarrolla será o bien la altura, o bien uno de los tres ángulos que definen la orientación del quadrotor: alabeo, cabeceo y guiñada.

En dicho control, la constante k_p ha de ser positiva ya que debe favorecer el movimiento, a fin de reducir la desviación respecto a la referencia. La constante k_d, por su parte, será negativa, ya que se debe oponer al movimiento.

Puesto que se ha optado por una configuración de control en equis, todas las acciones de control intervendrán en el cálculo del ciclo de trabajo de cada motor. Teniendo en cuenta el modelo teórico desarrollado, y la posterior rotación a la configuración elegida, las expresiones de la salida a los motores, para un control de orientación, quedan:

$$\begin{aligned} V_1 &= u + u_\phi + u_\theta + u_\psi \\ V_2 &= u - u_\phi + u_\theta - u_\psi \\ V_3 &= u - u_\phi - u_\theta + u_\psi \\ V_4 &= u + u_\phi - u_\theta - u_\psi \end{aligned} \tag{33}$$

donde V_n representa el ciclo de trabajo de cada motor.

La acción de control u representa una acción global, que permite controlar de forma general la sustentación de la plataforma quadrotor, a partir de un ciclo de trabajo base, igual para todos los motores, y que se puede especificar desde la estación de tierra, como se ha explicado anteriormente en la memoria.

Las acciones de control u_α son las correspondientes a sus respectivos ángulos: alabeo o ϕ, cabeceo o θ y guiñada o ψ.

En caso de incorporar una acción de control para la altura, partiendo de la ecuación en z del modelo simplificado, se llega a las siguientes expresiones:

$$
\begin{aligned}
V_1 &= \frac{u}{\cos\phi\cos\theta} + u_z + u_\phi + u_\theta + u_\psi \\
V_2 &= \frac{u}{\cos\phi\cos\theta} + u_z - u_\phi + u_\theta - u_\psi \\
V_3 &= \frac{u}{\cos\phi\cos\theta} + u_z - u_\phi - u_\theta + u_\psi \\
V_4 &= \frac{u}{\cos\phi\cos\theta} + u_z + u_\phi - u_\theta - u_\psi
\end{aligned}
\tag{34}
$$

donde u_z representa la acción de control correspondiente a la altura.

Para establecer el control, es necesario dar un valor a las constantes k_p y k_d de cada uno de los algoritmos de control correspondientes a cada grado de libertad intervenido. Este ajuste se puede realizar o bien de manera teórica, mediante el modelo desarrollado, o bien de manera experimental.

A continuación se explica brevemente el ajuste experimental llevado a cabo en la plataforma quadrotor desarrollada.

8.3. Ajuste experimental de los algoritmos de control

Los reguladores PD utilizados para el control de actitud y altura requieren el ajuste de unas constantes numéricas para cada una de las acciones de control que definen el comportamiento del sistema.

Normalmente, el valor óptimo de estas constantes se puede averiguar de forma teórica (tal como se ha hecho en asignaturas como *Control automático* durante el Grado), a partir del modelo desarrollado. No obstante, esto requiere del cálculo de una serie de parámetros de dicho modelo difíciles de obtener, como son las inercias del quadrotor, o la posición real de su centro de gravedad.

Además, este modelo es ideal, lo que supone que no hay garantías de que las constantes calculadas vayan a ser las óptimas en la práctica. Cualquier imperfección en el quadrotor o cualquier efecto aerodinámico no tenido en cuenta, por ejemplo, puede hacer que el comportamiento del quadrotor difiera en gran medida del teórico.

Por ello, y a fin de adquirir nuevos conocimientos, se decidió realizar un ajuste experimental de las constantes de los PD. Dicho ajuste se ha realizado en más de una ocasión, tras distintos cambios en la plataforma. No obstante, el orden de magnitud de las constantes se ha mantenido, a pesar de las modificaciones.

En primer lugar, se tiene que determinar mediante pruebas experimentales el valor del ciclo de trabajo al que cada motor comienza a girar, y el valor máximo que pueden aceptar. En este caso, la variación es ínfima, habiendo tan sólo un motor que comienza a girar un poco antes que el resto. Tras realizar los ajustes necesarios, ya se puede acotar la acción de control referente a la sustentación del quadrotor, común a los cuatro motores. Después, se

comprueba motor a motor que responden correctamente ante las variaciones de la acción de control.

Conocidos los signos con los que se suman a cada motor las acciones de control referentes a actitud y altura, se puede proceder al ajuste de los reguladores PD, cuatro distintos en este caso. Para ajustar el PD se debe partir de $k_p = 0$ y $k_d = 0$.

Primero, se va subiendo el k_d hasta que el quadrotor responde a las perturbaciones en el grado de libertad a ajustar, oponiéndose al movimiento. Cuando la acción de control sea suficiente como para evitar hasta cierto punto que el quadrotor abandone la posición inicial, sin llegar a impedir el movimiento totalmente, se puede proceder al ajuste del k_p.

Así pues, se sube poco a poco el valor de k_p hasta que el quadrotor, fuera de la posición correspondiente a la referencia, empiece a reaccionar. Debe llegar a un punto en que la acción de control tenga fuerza suficiente para llevar el sistema de vuelta a la referencia.

Tras ello, se debe afinar el ajuste, intentando en todo momento mantener la estabilidad del sistema. Por lo general, en los valores obtenidos k_p suele ser aproximadamente el doble de k_d. Una vez obtenida una relación entre ambos que resulte funcional, se puede probar a aumentarlos. Esto permite una respuesta más rápida y agresiva del sistema. Sin embargo, hay un límite, puesto que a partir de cierto punto el sistema puede llegar a oscilar en exceso, inestabilizándose.

Se trata de un proceso iterativo, hasta encontrar unos valores que den el comportamiento deseado. Además, este proceso se debe repetir para cada grado de libertad, puesto que las constantes, por lo general, serán distintas para cada uno.

Además, puesto que el procedimiento experimental requiere una interacción física con el sistema, a fin de perturbarlo y comprobar su respuesta, se ha de realizar tomando las medidas de seguridad necesarias, como el uso de guantes y gafas de protección.

9. Resultados experimentales

Una vez montada la plataforma y desarrollado un software que permitiese el funcionamiento de ésta, se procedió a comenzar las pruebas experimentales del sistema quadrotor. Dichas pruebas se realizaron con unos medios que se describen a continuación y que condicionan en gran medida el alcance de los resultados obtenidos.

El desarrollo de esta fase de pruebas no solo permitió conocer el comportamiento del sistema y ajustar los algoritmos de control, sino que también sirvió para ayudar a depurar parte del código programado, poniendo al descubierto errores o limitaciones que se solventaron sobre la marcha.

No obstante, dichas pruebas también pusieron de manifiesto limitaciones importantes de la plataforma, principalmente debidas al hardware utilizado, que se comentarán más adelante, en base a los resultados mostrados.

9.1. Características de las pruebas experimentales

El uso de un chasis comercial como el DJI F450 en la plataforma supuso un ahorro de tiempo importante en el desarrollo de ésta. Sin embargo, esta estructura no cuenta con mecanismos de protección, tales como una cubierta de poliestireno expandido, o barras exteriores que eviten el contacto con las hélices, así como algún punto de agarre, para sujetar el sistema en las primeras fases del ajuste del control. Además, la forma del chasis y la disposición de los posibles puntos de anclaje dificultan el diseño de un posible mecanismo de seguridad.

Debido a esto, se decidió comenzar las pruebas en un entorno controlado, que restringiese la libertad del quadrotor, a fin de evitar daños tanto al sistema como a las personas situadas alrededor de éste. Para ello, se hizo uso de un soporte para pruebas disponible en el laboratorio, cuyo aspecto se puede aprecia en la Figura 54.

Este soporte cuenta con una barra que puede girar, a la que se ha de anclar el quadrotor mediante una serie de tornillos y tuercas, gracias a una placa de metacrilato taladrada unida a dicha barra. Así, se restringen todos los grados de libertad de la plataforma excepto uno: o el ángulo de alabeo, o el de cabeceo.

Si bien esto no permite comprobar el comportamiento real del quadrotor en vuelo, si que da la posibilidad de ver si el sistema se ha planteado correctamente y si se es capaz de lograr un comportamiento estable en, al menos, un grado de libertad.

Así pues, los objetivos principales de las pruebas a realizar en dicho soporte eran:

- Testear la estabilidad del software desarrollado.
- Comprobar la fiabilidad del algoritmo de filtrado utilizado para la orientación.
- Comprobar que las acciones de control se habían tenido en cuenta correctamente.
- Intentar lograr un comportamiento estable en el ángulo de cabeceo.

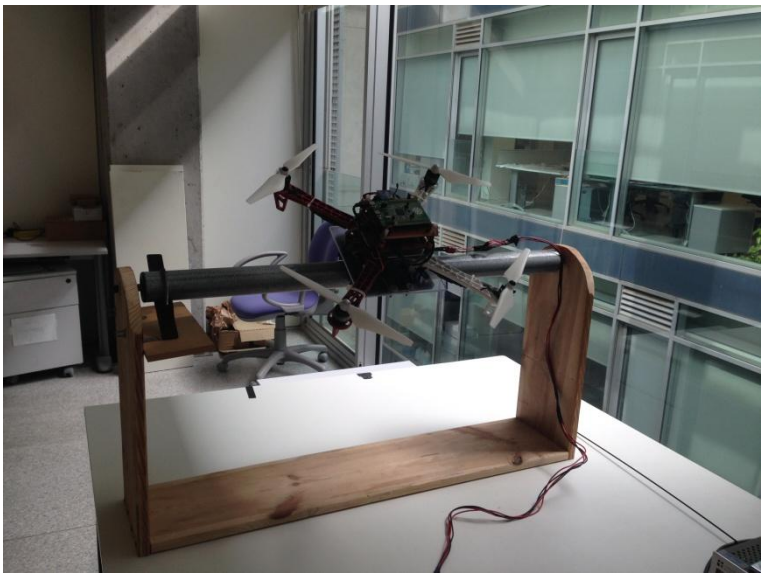

Figura 54: Vista del quadrotor montado en el soporte para las pruebas

A pesar del uso del soporte mencionado, durante todas las pruebas realizadas se tomaron medidas de seguridad extra. En todo momento había dos personas a cargo de la prueba: una operando la estación de tierra, y otra junto al soporte, o bien sujetándolo o bien perturbando el ángulo de la barra, a fin de comprobar la respuesta del quadrotor. Como medida de prevención, el encargado del soporte siempre lleva guantes y gafas de protección (ver Figura 55).

Figura 55: Guantes y gafas de protección

A continuación, se muestran algunos de los resultados obtenidos durante el desarrollo de varias de estas pruebas, y se comentan las conclusiones que se extraen de ellos.

9.2. Resultados obtenidos

Durante la mayor parte del desarrollo de las pruebas en el soporte, el algoritmo de filtrado utilizado en la plataforma ha sido un filtro complementario, tal como el descrito en el punto 3.2.1 de la memoria. El valor del parámetro α, que permite ajustar el peso relativo de las medidas del acelerómetro y las integraciones del giróscopo, se ha ido variando, a fin de

intentar encontrar un valor óptimo, que minimizase el ruido del primero, y la deriva del segundo.

La Figura 56 muestra el valor del ángulo de cabeceo o *pitch* a lo largo del tiempo para dos pruebas, ambas con un filtro complementario $\alpha = 0.99$, posición horizontal del quadrotor (bloqueando el giro de la barra del soporte) y constantes de control nulas.

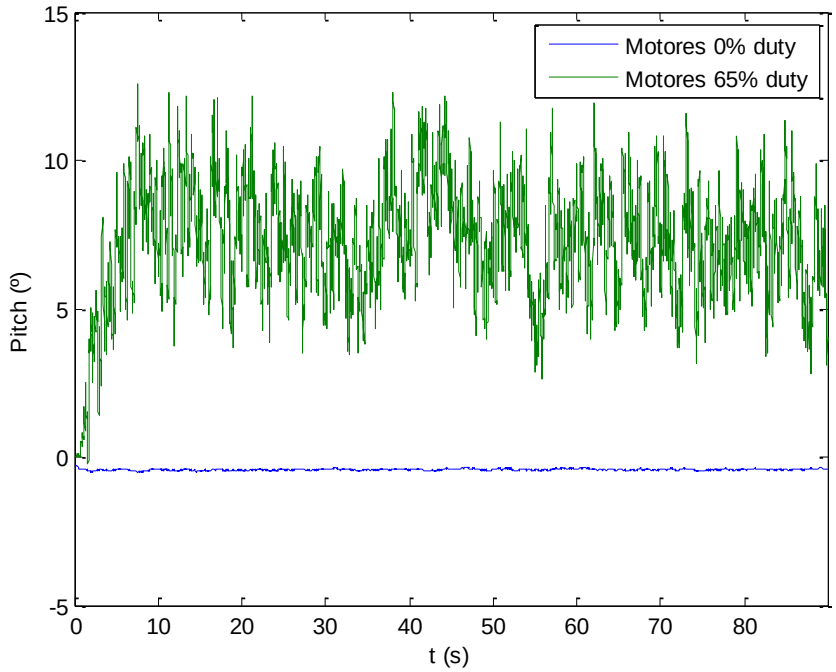

Figura 56: Comparativa del ángulo de cabeceo en posición horizontal, con y sin motores

La Figura 57, por su parte, muestra la actitud del quadrotor durante ambas pruebas.

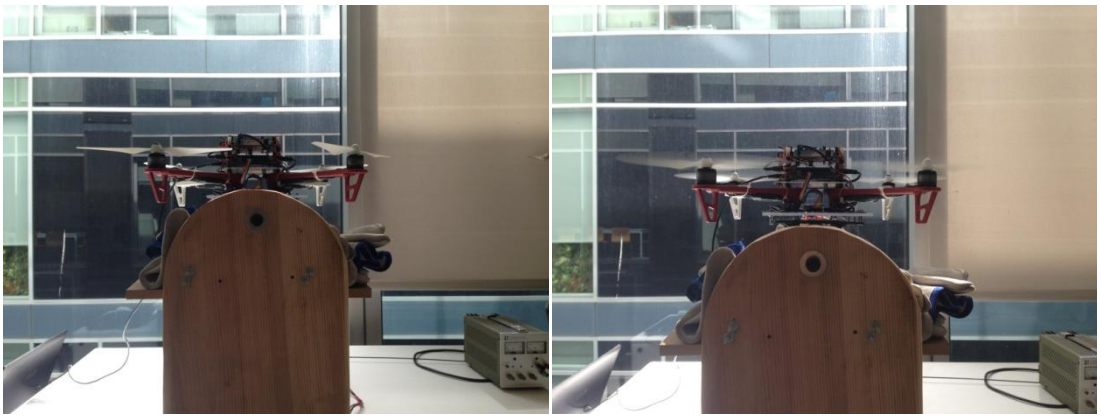

Figura 57: Vista del quadrotor en posición horizontal, con y sin motores

Se puede observar que, si bien la posición del quadrotor es prácticamente idéntica en las dos pruebas, los resultados obtenidos son muy distintos.

En el primer caso, con los motores apagados, el filtro funciona correctamente y las variaciones en el ángulo son de muy pequeña magnitud. Se puede apreciar una pequeña desviación respecto a cero, si bien es posible que la posición del quadrotor no fuese totalmente horizontal durante la prueba.

En caso de estar operativos los motores, los resultados empeoran drásticamente. Para un ciclo de trabajo del 65%, no sólo se observa una desviación importante, siendo la media de la medida de unos 7.5°, sino que además el ruido inducido por el funcionamiento de los motores y las vibraciones es muy importante, llegando a dar saltos bruscos de más de 5°.

Esto plantea la necesidad de descubrir el origen del problema, a fin de intentar solventarlo. En la Figura 58 se puede observar las lecturas de acelerómetro y giróscopo durante dicha prueba con motores.

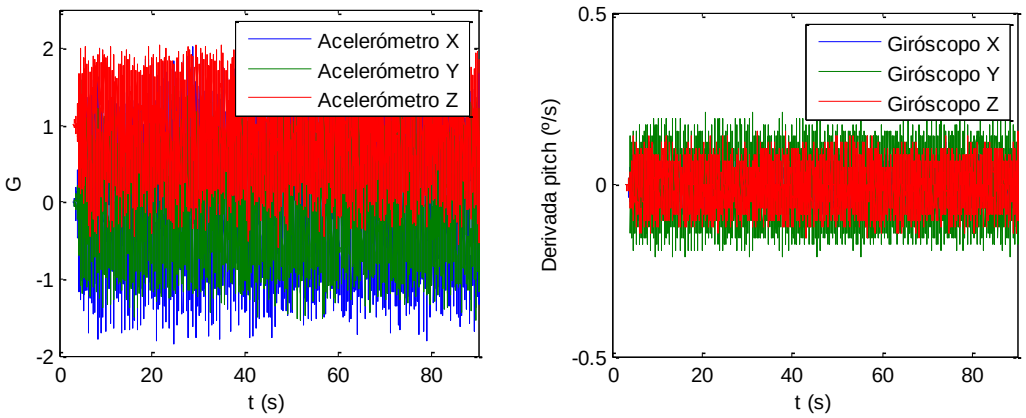

Figura 58: Lecturas de los sensores durante la prueba horizontal con motores

Observando los datos del giróscopo, se deduce que el problema no proviene de este sensor, ya que el ruido no es de gran magnitud y está muy lejos de saturar, teniendo en cuenta el rango de ±250 configurado para el sensor.

Sin embargo, en el caso del acelerómetro los datos son preocupantes. En posición horizontal, las lecturas en x e y deberían ser cero, mientras que en z, la unidad. Si bien la media de las lecturas coincide con esto, el ruido es tal que la mayor parte del tiempo satura (el rango elegido era de ±2 G). Así pues, ésta parece ser la causa de la mala estimación del ángulo.

Debido a esto, se decide comprobar el comportamiento dinámico del quadrotor tanto mediante el filtro complementario como teniendo en cuenta sólo la medida de los giróscopos.

La Figura 59 muestra el valor del ángulo de cabeceo o *pitch* a lo largo del tiempo para dos pruebas, una con un filtro complementario $\alpha = 0.99$, y otra con $\alpha = 1$, es decir, tomando solo los valores correspondientes a la integración del giróscopo. Estas pruebas se realizaron con control, tras establecer unos valores funcionales de las constantes mediante el ajuste experimental. El quadrotor parte de una posición inclinada, y el control lo lleva a una posición horizontal al inicio de los ensayos.

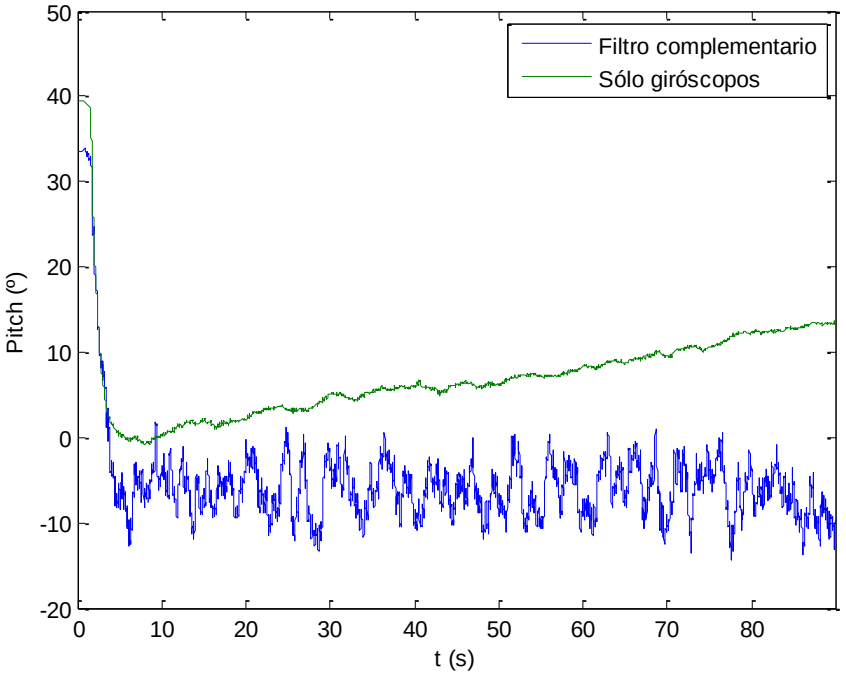

Figura 59: Comparativa del ángulo de cabeceo con control y referencia cero, bajo filtro complementario o solo giróscopos

Como cabía esperar, el uso exclusivo de los giróscopos para la estimación provoca una deriva en el ángulo que hace que el quadrotor tienda a inclinarse cada vez más. En el tiempo de la prueba mostrada, de un minuto y medio, se llega a una desviación importante, de más de diez grados. Bajo un uso normal, más prolongado, el error al que se llega es insostenible, por lo que esta configuración no es viable para el vuelo de la plataforma.

Figura 60: Vista del quadrotor con control, filtro complementario y referencia cero

En cuanto a la estimación del filtro complementario, la gráfica muestra unos resultados con bastante ruido y una media de -7° aproximadamente. La Figura 60 muestra la actitud del

quadrotor durante la mayor parte de la prueba. El ruido introducido por el acelerómetro hace que el sistema sea incapaz de conocer correctamente su orientación, y no se logra llegar a la referencia.

Así pues, se deduce que ninguna de las opciones mostradas es válida para mantener un vuelo estable controlado. No obstante, las diversas pruebas realizadas arrojan ciertos resultados positivos muy importantes:

- El sistema es capaz de mantenerse en torno al ángulo que, debido al ruido, cree erróneamente que se corresponde con la referencia.
- La respuesta ante las perturbaciones es buena (ver Figura 61). Si se altera el ángulo de la plataforma, ésta tiende a volver a la supuesta referencia de manera suave, con pocas oscilaciones.
- El sistema capta los cambios en la referencia y responde ante ellos variando su actitud en consecuencia.

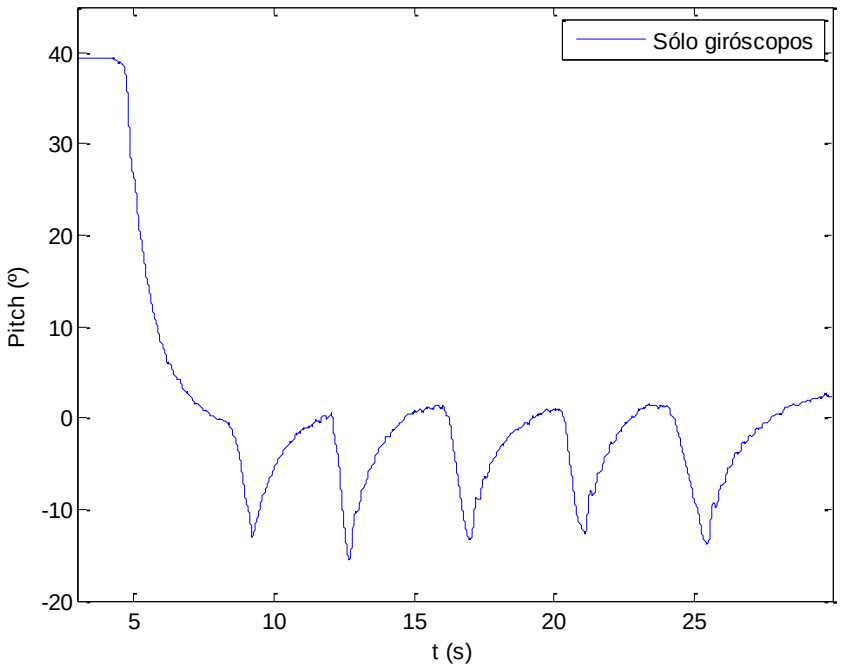

Figura 61: Respuesta ante perturbaciones en el ángulo de cabeceo con referencia cero

Por lo tanto, se puede concluir que el control se ha implementado bien y el problema radica en la medida de la actitud. Lo más seguro es que, debido a su bajo coste, los acelerómetros de la placa STM sean de baja calidad, captando en exceso el ruido inducido por los motores. Por ello, se plantean dos mejoras de la plataforma a corto plazo, que se están llevando a cabo en la actualidad, y que deberían mejorar considerablemente el comportamiento de la plataforma:

- Usar un algoritmo de filtrado más avanzado. El compañero de desarrollo del sistema quadrotor, Norberto Vera, realizó pruebas con un filtro de Kalman implementado en Matlab (en caso de desear más detalles al respecto, se puede consultar su memoria).

Los resultados, en lo que al cálculo de la orientación de la plataforma se refiere, fueron muy positivos. El quadrotor se acercaba con gran precisión al valor de referencia. No obstante, la necesidad de conectar un PC, sobremesa o portátil, a la placa STM lo hacía totalmente inviable. En la actualidad se está adaptando dicho filtro a C++ a fin de implementarlo en la Beagle.

- Usar una IMU de mayor calidad de entre las disponibles en el laboratorio. En la actualidad se está adaptando el sistema para realizar pruebas con la IMU Sparkfun Razor 9DOF, a fin de comprobar si mejoran las lecturas.

Por lo tanto, aunque actualmente la plataforma no se encuentra en condiciones para volar de manera libre, sin restricciones, se espera que en poco tiempo sea posible.

10. Conclusiones finales y trabajos futuros

10.1. Conclusiones

A lo largo de este documento, se ha expuesto el proceso de desarrollo de una plataforma quadrotor partiendo de cero, comentando no sólo características, procedimientos o modelos teóricos, sino también algunas de las conclusiones que se han ido extrayendo durante el tiempo dedicado al proyecto. Si bien los resultados obtenidos pueden mejorarse, y no ha sido posible todavía realizar un vuelo del quadrotor como tal, se puede considerar un gran logro haber llegado hasta este punto, teniendo en cuenta los imprevistos y problemas surgidos durante el desarrollo, muchos de ellos solventados de forma efectiva.

Cuando se tomó la decisión de embarcarse en un proyecto así, se subestimó la magnitud de éste. Conforme se iba avanzando en el desarrollo de la plataforma, eran más los conocimientos requeridos, la información a buscar, la complejidad de las ideas a plasmar… Todo esto unido a los pequeños problemas que surgen día a día en este tipo de proyectos, como una soldadura que falla o una parte del código que no funciona como supuestamente debería, no ha hecho más que aumentar el tiempo requerido por el proyecto.

Por todo esto, no ha sido posible llegar a este punto habiendo cumplido todos los objetivos inicialmente planteados. No obstante, el trabajo no ha sido en vano, y la plataforma dista mucho de aquella idea inicial difusa que no terminaba de tomar forma. El quadrotor está montado, el software funciona de forma estable, sin errores, y los resultados experimentales, si bien no se pueden considerar los óptimos para el vuelo del quadrotor, invitan al optimismo, revelando una situación problemática que, no obstante, se puede llegar a corregir en un futuro. Y de hecho, se sigue trabajando en ello.

Desde el punto de vista personal, lo más satisfactorio ha sido, sin duda, la gran cantidad conocimientos adquiridos, en ámbitos muy diversos, muchos de los cuales no habría sido posible adquirir de otra manera. Haber profundizado en varios lenguajes de programación, algunos previamente desconocidos, haber aprendido protocolos de comunicación entre dispositivos, haber implementado algoritmos de control y de filtrado, o incluso haber puesto en práctica algunas de las cosas que se conocían de un modo teórico gracias a asignaturas previas. Todo esto supone un conocimiento práctico muy útil, un *know-how* que puede ser de gran ayuda en el futuro, a la hora de afrontar el mercado laboral.

Además, el hecho de ver como todos esos conocimientos toman forma, y que cada artículo que se lee, cada libro o página web que se consulta, se traducen en un pequeño paso en el desarrollo, que acerca el objetivo de volar el quadrotor, ha sido una gran motivación, no sólo para llegar a este punto del proyecto, sino también para tener la predisposición a seguir trabajando en la plataforma en el futuro.

Así pues, a continuación se comentan algunas de las posibles mejoras que se pueden efectuar sobre la plataforma en su estado actual, así como trabajos futuros a tener en cuenta una vez que se logre el funcionamiento óptimo de ésta.

10.2. Trabajos futuros

Entre las principales mejoras a realizar en la plataforma a corto plazo, a fin de cumplir con todos los objetivos inicialmente planteados, se pueden destacar las siguientes:

- Instalación de una nueva IMU. Como ya se ha comentado, la plataforma en su estado actual presenta un problema de medida, debido a los sensores que incluye la placa STM. Una opción sencilla para solventar esta situación, en la que ya se trabaja, es sustituir dicha IMU por una de mayor calidad. Esto podría complementarse con la implementación del PWM en el mini PC Beagle, aprovechando la funcionalidad de algunos de sus pines, para prescindir totalmente de la placa STM.
- Algoritmos de filtrado avanzados. Como alternativa a lo anterior, aunque también como posible complemento, se plantea la inclusión de algoritmos de filtrado avanzados, más complejos que el filtro complementario utilizado, como por ejemplo un filtro de Kalman extendido.
- Medidas de seguridad. Para iniciar las pruebas de vuelo sin necesidad de usar un anclaje, se puede plantear el diseño y fabricación de protecciones para el quadrotor, tales como una cubierta de poliestireno expandido.

Adicionalmente, una vez cumplidos los objetivos iniciales de la plataforma, posibilitando un vuelo robusto y fiable en interiores con control de actitud y altura, se pueden plantear las siguientes mejoras a la plataforma, algunas de ellas de gran magnitud:

- Control de posición para interiores. Se puede plantear un control por flujo óptico para la posición en interior, incorporando una cámara a la plataforma y desarrollando el software necesario. No obstante, debido al coste computacional que implica, se ha de estudiar si el hardware actual lo soportaría manteniendo tiempos de ejecución aceptables, o si sería necesaria la inclusión de otro mini PC.
- Vuelo en exteriores. Para adaptar la plataforma al exterior, se plantean dos posibles mejoras. Por una parte, un control de posición por GPS, que requeriría de la inclusión de un sensor GPS, además del software para la comunicación, el procesado y el control. Por otra parte, un control de altura tanto por ultrasonidos como con un sensor barométrico, que se habría de incluir, así como los algoritmos necesarios para la fusión de las medidas de ambos.
- Seguimiento de objetivos. En caso de aplicar alguna de las mejoras anteriores, se puede estudiar la posibilidad de realizar un seguimiento de objetivos, ya sea por flujo óptico, o GPS.
- Sistema operativo en tiempo real. Para aprovechar al máximo la potencia del mini PC, se recomienda la instalación de un sistema operativo en tiempo real. Si bien durante el desarrollo del proyecto se intentó, sin éxito, con Xenomai, se puede probar alguna alternativa.
- Medidas de seguridad avanzadas. Por ejemplo, se puede programar un procedimiento de aterrizaje de emergencia en caso de perder el contacto con la estación de tierra, o de que la batería esté a punto de agotarse. Para esto último, se debería implementar un sistema que monitorice el estado de la batería.

- Algoritmos de control avanzados. Se podrían implementar algoritmos de mayor complejidad que los PID, siempre y cuando no se comprometa en exceso los tiempos de ejecución establecidos. Así pues se podría probar con un control en espacio de estados, algoritmos LQR o LQG, modos deslizantes o esquemas mixtos de PID y H-infinito, por ejemplo.
- Sustitución de componentes electrónicos. Por último, se plantea la posibilidad de sustituir componentes electrónicos del quadrotor por otros que ofrezcan un rendimiento similar a menor precio. Por ejemplo, se podría estudiar la posibilidad de cambiar el BeagleBone Black por una Raspberry Pi 2 Model B, que se puso a la venta durante el desarrollo del proyecto, a un precio más reducido, con rendimiento similar y tres puertos USB más.

Presupuesto

El objetivo de este presupuesto es mostrar la aportación económica necesaria por parte de una empresa para la realización del proyecto descrito en esta memoria. Durante la elaboración de dicho proyecto, se ha hecho uso de una serie de materiales y equipos necesarios a fin de lograr crear una plataforma quadrotor operativa. Si bien gran parte de estos componentes y equipos utilizados estaban disponibles en el laboratorio inicialmente, es cierto que los estudiantes involucrados han realizado algunas aportaciones económicas a fin de acelerar el desarrollo del proyecto, y poder continuar el trabajo en la plataforma posteriormente a la entrega de este trabajo.

A continuación, se procede a describir el presupuesto global de este Trabajo de Fin de Grado. Este presupuesto incluye tanto el trabajo realizado por el alumno que escribe esta memoria como el de su compañero, Norberto Vera Vélez, a fin de reflejar el coste total real de la plataforma en su estado actual. Además, cabe resaltar que sin el trabajo coordinado entre los dos estudiantes no habría sido posible llegar a este punto.

A lo largo de las siguientes secciones se presenta una estimación del coste total del proyecto, clasificado según su origen. No se han tenido en cuenta posibles gastos como la manutención, el abastecimiento de energía o el alquiler de un inmueble, bajo la suposición de que el proyecto se realizaría en una empresa ya establecida en la que estos gastos son imprescindibles para su funcionamiento, y el proyecto representaría una pequeña parte de éste.

Coste del material y equipos utilizados

En esta sección se considera el coste de los materiales utilizados en la plataforma, así como los equipos necesarios para su desarrollo. A fin de facilitar la visualización del coste de cada una de las partes, se mostrarán tablas individuales para la sección aire (quadrotor) y la sección tierra (estación de tierra) del RPAS, así como para los equipos utilizados.

Cabe mencionar que no se considera gasto en licencias software pues en todo momento se ha intentado utilizar opciones de software libre o gratuitas, a disposición de quien desee utilizarlo. En caso de que alguno de estos programas no permita el uso profesional con fines económicos, es posible encontrar alternativas libres de restricciones, totalmente funcionales.

La Figura 62 recoge los costes del sistema quadrotor, mientras que la Figura 63 los de la estación de tierra.

La Figura 64 recoge los costes de los equipos necesarios para el desarrollo de la plataforma RPAS.

La Figura 65 recoge el coste total del material y equipos utilizados.

Componente	Precio en el mercado (incluye impuestos)
Kit de montaje DJI F450 E300	159.00 €
Mini PC BeagleBone Black	60.73 €
Batería LiPo Zippy Compact 5800	51.24 €
Módulo de radiofrecuencia Xbee Pro S1	39.57 €
Sensor de ultrasonidos SFR10	35.15 €
Microcontrolador STM32F3 Discovery	12.81 €
Placa Sparkfun Xbee Explorer	11.50 €
Regulador de tensión HCJ-IPM-V2	4.59 €
Barra de contactos 40 pines x2	3.88 €
Pines hembra Arduino x6	3.60 €
Cable USB mini-B	3.20 €
Indicador de voltaje genérico	3.19 €
Cable con conector tipo Jack	2.95 €
Barra de contactos 40 pines acodada	2.48 €
Placa de metacrilato 11x11 cm	2.32 €
Conector Dean-T	1.15 €
Tornillería variada	0.99 €
Placa de prototipado 22x10 cm	0.98 €
Cables de prototipado varios	0.95 €
Espaciadores para PCB varios	0.75 €
Cinta aislante	0.63 €
Tubos termorretráctiles varios	0.50 €
Hilo de soldar	0.31 €
TOTAL	**402.47 €**

Figura 62: Tabla de coste de los componentes del quadrotor

Componente	Precio en el mercado (incluye impuestos)
Ordenador con sistema operativo Windows	499.99 €
Módulo de radiofrecuencia Xbee Pro S1	39.75 €
Mando de Xbox 360 con cable	29.99 €
Placa Sparkfun Xbee Explorer USB	27.71 €
Conector USB mini-B a USB	2.40 €
TOTAL	**599.84 €**

Figura 63: Tabla de coste de los componentes de la estación de tierra

Equipo	Precio en el mercado (incluye impuestos)
Ordenador para desarrollo de software	799.99 €
Osciloscopio digital	384.95 €
Fuente de alimentación regulable	329.95 €
Soporte para pruebas con un grado de libertad	200.00 €
Multímetro digital	12.95 €
Cables USB x2	6.40 €
Regleta	5.95 €
Soldador de estaño	5.45 €
TOTAL	**1745.64 €**

Figura 64: Tabla de coste de los equipos utilizados durante el proyecto

Concepto	Coste
Coste de los componentes del quadrotor	402.47 €
Coste de los componentes de la estación de tierra	599.84 €
Coste de los equipos utilizados durante el proyecto	1745.64 €
TOTAL	**2747.95 €**

Figura 65: Tabla de coste global de los materiales y equipos utilizados

Coste de la mano de obra

En esta sección se considera el coste de las horas invertidas en el proyecto por las dos personas involucradas. Para ello se ha supuesto el trabajo realizado por un Ingeniero Técnico Aeronáutico, recién graduado en el Grado en Ingeniería Aeroespacial, estimando su salario en unos 1200€/mes. Suponiendo una jornada de trabajo de 8 horas, y 20 días laborables, el coste de la hora de trabajo asciende a 7.5€.

La Figura 66 muestra una tabla con el coste derivado de las horas de trabajo del autor de esta memoria, considerados individualmente.

La Figura 67 muestra, por su parte, una tabla con el coste de la mano de obra, considerando las horas invertidas por los dos estudiantes involucrados, el cual se tendrá en cuenta a la hora de calcular el coste final.

Concepto o tarea	Horas invertidas	Salario por hora	Coste
Documentación	75	7.5 €	562.50 €
Estudio preliminar	100	7.5 €	750.00 €
Diseño y desarrollo	225	7.5 €	1687.50 €
Experimentación y ajuste	100	7.5 €	750.00 €
TOTAL	**500**		**3750.00 €**

Figura 66: Tabla de coste individual de la mano de obra

Concepto o tarea	Horas invertidas	Salario por hora	Coste
Documentación	150	7.5 €	1125.00 €
Estudio preliminar	125	7.5 €	937.50 €
Diseño y desarrollo	525	7.5 €	3937.50 €
Experimentación y ajuste	200	7.5 €	1500.00 €
TOTAL	**1000**		**7500.00 €**

Figura 67: Tabla de coste grupal de la mano de obra

Coste total final

El coste final del proyecto asciende a la cantidad de 10247.95 €. Queda recogido y desglosado en la Figura 68.

Concepto	Coste
Coste de los materiales y equipos	2747.95 €
Coste de la mano de obra	7500.00 €
TOTAL	**10247.95 €**

Figura 68: Tabla resumen del coste total del proyecto

Bibliografía y referencias

Ascending Technologies. (s.f.). Disponible en: http://www.asctec.de/

BeagleBoard.org Foundation. (s.f.). Disponible en: http://beagleboard.org/

Bresciani, T. (2008). *Modelling, identification and control of a quadrotor helicopter*. Lund: Department of Automatic Control, Lund University.

Canonical. (s.f.). Disponible en: http://askubuntu.com/

Castillo Frasquet, A. (2014). *Desarrollo integral de un quadrotor: Diseño del control de posición*. Valencia: Trabajo final de grado, ETSII-UPV.

CodeProject. (s.f.). Disponible en: http://www.codeproject.com/Articles/492473/Using-XInput-to-access-an-Xbox-Controller-in-M

Digi International. (s.f.). Disponible en: http://www.digi.com/

DJI. (s.f.). Disponible en: http://www.dji.com/

Draganfly. (s.f.). Disponible en: http://www.draganfly.com/

Element14. (s.f.). Disponible en: http://www.element14.com/community/

eLinux. (s.f.). Disponible en: http://elinux.org/BeagleBoardDebian

Luukkonen, T. (2011). *Modelling and control of quadcopter*. Espoo: Trabajo independiente de investigación, Aalto University School of Science.

Microsoft Developer Network. (s.f.). Disponible en: https://msdn.microsoft.com/es-es/library/kx37x362.aspx

OlliW's Bastelseiten. (s.f.). Disponible en: http://www.olliw.eu/2013/imu-data-fusing/

Ródenas Lorda, L. (2013). *Plataforma de desarrollo para el control de estabilidad en tiempo real de un vehículo aéreo tipo quadrotor*. Valencia: Proyecto final de carrera, ETSID-UPV.

Sparkfun. (s.f.). Disponible en: https://www.sparkfun.com/

Stack Overflow. (s.f.). Disponible en: http://stackoverflow.com/

STMicroelectronics. (s.f.). Disponible en: http://www.st.com/web/en/catalog/tools/FM116/SC959/SS1532/LN1848/PF254044

Sweet, M. R. (1999). *Serial programming guide for POSIX operating systems*.

The C++ Resources Network. (s.f.). Disponible en: http://www.cplusplus.com/

The Geek Stuff. (s.f.). Disponible en: http://www.thegeekstuff.com/

Unmanned Aerial Vehicle Systems Association. (s.f.). Disponible en: http://www.uavs.org/commercial

Vera Vélez, N. (2015). *Desarrollo de un vehículo aéreo remotamente tripulado (RPAS) para su uso en aplicaciones civiles*. Valencia: Trabajo final de grado, ETSID-UPV.

Verdú Torres, D. (2014). *Desarrollo integral de un quadrotor: Control de orientación basado en una IMU de bajo coste y control de altura mediante un sensor barométrico*. Valencia: Trabajo final de grado, ETSII-UPV.