UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

DISEÑO E IMPLEMENTACIÓN DE UN ALGORITMO DE CONTROL AVANZADO PARA LA ESTABILIZACIÓN DE UN QUADROTOR BASADO EN EL AUTOPILOTO PIXHAWK Y UN SISTEMA DE POSICIONAMIENTO RTK - GPS

AUTOR: Igual Bañó, Joan

TUTOR: García Gil, Pedro José

Curso Académico: 2017-18

“A mis padres y hermano, por todo el apoyo
que me han ofrecido durante estos años.

A mis compañeros de laboratorio, Alberto,
Vicente, Ricardo, Joaquín y Conrado;
y a mi tutor, Pedro, por toda la atención
y ayuda que me han prestado.

A Chewie y a mis amigos, por permitirme
ser la persona que soy hoy.”

Resumen

Los reguladores tipo PID son, sin lugar a duda, los de mayor implantación dentro de un entorno industrial. Estos, se componen de 3 factores básicos de control, proporcional, integral y derivativo (P + I + D).

Sin embargo, la cada vez mayor automatización de sistemas y procesos más complejos hace necesaria la aparición de nuevos algoritmos de control que, conservando las principales ventajas de los reguladores tipo PID, se adapten mejor al control de dichos procesos, como pueden ser aquellos multivariables, con una dinámica rápida o inestables.

Este es el caso de los vehículos aéreos no tripulados, y en concreto de los Quadrotors. La mayoría de los autopilotos comerciales todavía usan reguladores tipo PID (con diferentes configuraciones). Sin embargo, la necesidad cada vez mayor, de dotar a estos dispositivos, no solo de la capacidad de vuelo *hover*, sino también de poder ejecutar maniobras agresivas, hace necesario el desarrollo de nuevos algoritmos que, sin perder las propiedades de los reguladores PID, se ajusten mejor al control de estos dispositivos (multivariables e inestables).

Entre estos nuevos algoritmos, se ha popularizado el *Uncertainity and Disturbance Estimator* (UDE) basado en el control a partir de estados conocidos; y el *Extended State Observer* (ESO), el cual es el principal elemento del *Active Disturbance Rejection Control* (ADRC), un nuevo "paradigma" de control que nació con la vocación de sustituir al popular PID.

Por otro lado, si se desea realizar un control de posición de un vehículo aéreo no tripulado, es necesario disponer de algún sistema de posicionamiento de dicho vehículo, a partir del cual poder diseñar el sistema de control. En particular, y para exteriores se han popularizado los sistemas de posicionamiento global basados en sensores RTK-GPS.

Dentro de este contexto, la realización de este TFG se plantea para que el alumno diseñe estrategias de control avanzadas, poniendo en práctica y profundizando en los conocimientos adquiridos en varias disciplinas del Grado en Ingeniería en Tecnologías Industriales, tales como los aprendidos en las asignaturas de Sistemas Automáticos y Tecnología de la Automatización; así como su integración y coincidencia con otras disciplinas como conocimientos de electrónica y programación, necesarias para poder realizar la implementación computacional y ajuste experimental del algoritmo de control diseñado.

Resum

Els reguladors tipus PID són, sense cap dubte, els de major implantació dins d'un entorn industrial. Aquests, es componen de 3 factors bàsics de control, proporcional, integral i derivatiu (P + R + D).

No obstant això, la cada vegada major automatització de sistemes i processos més complexos, fa necessària l'aparició de nous algoritmes de control que, conservant les principals avantatges dels reguladors tipus PID, s'adaptin millor al control de processos més complexos, com poden ser aquells multivariables, amb una dinàmica ràpida o inestables.

Aquest és el cas dels vehicles aeris no tripulats, i en concret dels Quadrotors. La majoria de autopilots comercials encara fan servir reguladors tipus PID (amb diferents configuracions). Però, la necessitat cada vegada major, de dotar aquests dispositius, no només de la capacitat de vol *hover*, sinó també de poder executar maniobres agressives, fa necessari el desenvolupament de nous algoritmes que, sense perdre les propietats dels reguladors PID, puguen adaptar-les al control d'aquests dispositius (multivariables i inestables).

Entre aquests nous algoritmes, s'ha popularitzat el *Uncertainity and Disturbance Estimator* (UDE) basat en el control a partir d'estats coneguts; y el *Extended State Observer* (ESO), el qual és el principal element de l'Active Disturbance Rejection Control (ADRC), un nou "paradigma" de control que va naixer amb la vocació de substituir al popular PID.

Per altre costat, si es vol fer un control de posició d'un vehicle aeri no tripulat, cal disposar d'algun sistema de posicionament d'aquest vehicle, a partir del qual poder dissenyar el sistema de control. En particular, i per a exteriors s'han popularitzat els sistemes de posicionament global passats en sensors RTK-GPS.

Dins d'aquest context, la realització d'aquest TFG es planteja perquè l'alumne dissenyi estratègies de control avançades, posant en pràctica i aprofundint en els coneixements adquirits en diverses disciplines del Grau en Enginyeria en Tecnologies Industrials, com ara els apresos en les assignatures de Sistemes automàtica i Tecnologia de la Automatització, així com la seua integració i coincidència amb altres disciplines, com coneixements d'electrònica i programació, necessàries per poder realitzar la implementació computacional i ajust experimental de l'algoritme de control dissenyat.

Abstract

The PID type regulators are, without a doubt, the most established in an industrial environment. These are composed of 3 basic factors of control, proportional, integral and derivative (P + I + D).

However, the increasing automation of more complex systems and processes, makes necessary the emergence of new control algorithms that, while retaining the main advantages of PID-type regulators, adapt better to the control of more complex processes, such as those multivariables, with fast or unstable dynamics.

This is the case of unmanned aerial vehicles, and specifically the Quadrotors. Majority of commercial autopilots still use PID-type regulators (with different configurations). However, the increasing need to provide these devices, not only hover flight capacity, but also to be able to execute aggressive maneuvers, makes necessary the development of new algorithms that, without losing the properties of the PID regulators, to adjust better to the control of these devices (multivariable and unstable).

Among these new algorithms, the Uncertainity and Disturbance Estimator (UDE) base don the control from known states; and the Extended State Observer (ESO) have become popular. ESO is the main element of the Active Disturbance Rejection Control (ADRC), a new control paradigm that was born with the vocation of replacing the popular PID.

On the other hand, if it is desired to perform a position control of an unmanned aerial vehicle, it is necessary to have some positioning system of said vehicle, from which to design the control system. In particular, and for outdoor applications, global positioning systems bassed on RTK-GPS sensors have become popular.

In this way, the realization of this TFG, is proposed for the student to desing advanced control strategies, putting into practice and deepening the knowledge acquired in various disciplines of the Degree in Engineering in Industrial Technologies, such as those learned in the subjects of Automatic Systems and Automation Technology, as well as its integration and coincidence with other disciplines, such as knowledge of electronics and programming, necessary to be able to carry out the computational implementation and experimental adjustment of the designed control algorithm.

Índice general

Parte I

Memoria

Índice de la Memoria

Índice de Figuras

Capítulo 1

Objetivos y Alcance del trabajo

En este capítulo se describen los diferentes objetivos que se plantearon superar con la realización de este Trabajo Fin de Grado, así como las distintas tareas necesarias para llevarlo a cabo y los resultados obtenidos a partir de las mismas. Finalmente, se especifica cómo está estructurado el presente documento.

1.1. Objetivos

Los objetivos generales del proyecto que se han establecido son dos. Por un lado, realizar un estudio sobre diferentes técnicas de control de orientación sobre un Quadrotor analizando las ventajas y desventajas que presenta cada una; y por otro, plantear un algoritmo de control de posición mediante un sistema RTK-GPS. Así pues, el trabajo se ha dividido en diferentes fases: una primera fase inicial de construcción de un Quadrotor nuevo desde cero; una segunda fase de familiarización con el entorno de programación y simulación que ofrece el Pixhawk; una tercera fase de desarrollo e implementación de diferentes algoritmos de control frente a perturbaciones; y finalmente una cuarta fase de integración del sistema GPS.

Como se podrá observar a continuación, el ámbito de este trabajo ha sido en global de carácter multidisciplinar. Se ha trabajo tanto con hardware y elementos físicos como con software, todo ello a la hora de programar, construir el dron, comunicar diferentes sistemas electrónicos, lectura de señales, soldaduras, etc.

1.1.1. Construcción de un Quadrotor con el dispositivo Pixhawk

La primera fase del trabajo ha consistido en la elaboración y construcción de un Quadrotor perfectamente operativo y que emplee el dispositivo Pixhawk.

1.1.2. Familiarización con el entorno de programación y simulación del Quadrotor

El segundo objetivo ha sido analizar y comprender toda la parte de software que entraña la CPU del dron para posteriormente, poder trabajar sobre ella. Siguiendo una guía facilitada por el desarrollador, se ha instalado en un entorno Linux todos los programas y archivos necesarios para hacer posible una simulación de vuelo sobre la que poder observar, posteriormente, el comportamiento del Quadrotor ante diferentes algoritmos de control sin tener que recurrir al mismo físicamente.

1.1.3. Desarrollo e implementación de algoritmos de control para la orientación

Una vez comprendido todo el entorno del Quadrotor, se ha empezado a trabajar y modificar el mismo. Se han planteado todo tipo de estrategias de control, algunas de ellas de carácter más avanzado, y se han realizado tanto simulaciones como vuelos reales para observar el comportamiento y desempeño de la máquina con los distintos controladores.

1.1.4. Integración del sistema RTK-GPS

Finalmente, la última parte del trabajo ha sido dotar a la plataforma de un sensor RTK-GPS que sea capaz de obtener medidas de posición en coordenadas x, y, z en exteriores y realizar un control de posición sobre la misma.

1.2. Tareas Específicas a Realizar

a) A partir del autopiloto comercial Pixhawk, entender el funcionamiento de los elementos HW y SW que componen un sistema Quadrotor
b) Ensamblado del sistema Pixhawk junto a los demás elementos que componen un sistema Quadrotor (fuselaje, driver-motores, etc.)
c) Modelado del sistema Quadrotor, tanto del modelo no lineal, como de su simplificación linealizada
d) Diseño e implementación de varias configuraciones PID para familiarizar al alumno con el entorno de programación, simulación y ejecución del Pixhawk
e) Diseño e implementación de un sistema de control en cascada, basado en un UDE y en un ESO, para el control integral del Quadrotor
f) Ajuste experimental y validación del algoritmo implementado
g) Integración HW/SW del sistema RTK-GPS en el entorno Pixhawk

1.3. Resultados obtenidos

Como se describirá más adelante en este escrito, han sido muchos los resultados que se han obtenido con este proyecto. Por un lado, se ha conseguido implementar de forma satisfactoria una gran variedad de estrategias de control, desde el PD y PID, hasta el UDE y el ESO. Todos ellos se han podido validar tanto en un entorno de simulación como durante vuelos reales a partir del Quadrotor construido. Por otra parte, se ha logrado disponer al dron de un control de posición basado en un sistema RTK-GPS, consiguiendo, por tanto, un control completo sobre el multi-rotor tanto en orientación como posición.

1.4. Estructura del documento

Se presenta aquí de manera abreviada una descripción de los distintos apartados y aspectos que contiene esta memoria:

1. ***Objetivos y Alcance del trabajo***

   Se describen los diferentes objetivos que se plantearon superar con la realización de este Trabajo Fin de Grado, así como las distintas tareas necesarias para llevarlo a cabo y los resultados obtenidos a partir de las mismas.

2. ***Introducción***

   Se realiza una exposición clara y detallada de los problemas que se pretenden resolver con el proyecto. Así mismo, se especifican cuáles son los antecedentes de este trabajo, es decir, la información de la que se parte, y su justificación**.**

3. ***Estado del arte***

   Se describen diferentes conceptos que serán empleados a lo largo de todo el escrito, así como su contexto, su estado actual y algunos ejemplos de los mismos.

4. ***Modelo teórico del Quadrotor***

   Se pretende deducir y analizar paso a paso todo el modelo teórico y ecuaciones características que representan el comportamiento del Quadrotor durante el vuelo**.**

5. **Descripción de la plataforma**

   Se indica detalladamente los pasos que se han seguido para construir la plataforma de vuelo y las características de sus componentes principales.

6. ***Pixhawk***

   Se describe detalladamente el hardware y software de piloto automático Pixhawk empleado para la realización de este trabajo. A su vez, se describe la estrategia de control que ejecuta y los posibles modos de vuelo.

7. ***Desarrollo y resultados de las leyes de control de orientación implementadas***

   Se presentan y argumentan los diferentes algoritmos de control de orientación desarrollados para el trabajo, así como los resultados obtenidos a partir de los mismos tanto en simulación como en vuelos reales.

8. ***Desarrollo y resultados de la ley de control de posición implementada - Sistema RTK GPS***

   Se presenta y argumenta el algoritmo de control de posición desarrollado para el trabajo a partir de un sistema de posicionamiento RTK – GPS. Así mismo, se introduce de manera abreviada al lector en el estado del arte de los sistemas GPS y su funcionamiento.

9. ***Conclusiones y trabajos futuros***

   Se extraen todas las conclusiones que se han podido obtener a raíz del trabajo realizado, a la vez que se proponen diferentes modos de continuar y mejorar el proyecto que aquí se describe.

Bibliografía

Capítulo 2

Introducción

Durante este capítulo se realiza una exposición clara y detallada de los problemas que se pretenden resolver con el proyecto. Así mismo, se especifica cuáles son los antecedentes de este trabajo, es decir, la información de la que se parte, y su justificación.

El proyecto que nos ocupa consiste en el desarrollo e implementación de un algoritmo para el control tanto de orientación como de posición de un sistema aéreo remotamente pilotado (*Remotely Piloted Aircraft Systems*, RPAS) de tipo Quadrotor, ante incertidumbre de modelado y perturbaciones externas. Este trabajo se ve orientado concretamente hacia el autopiloto comercial Pixhawk.

2.1. Motivación

Como se indica en su página oficial [1], "Pixhawk® es un proyecto de carácter independiente que tiene como objetivo proporcionar hardware de piloto automático industrial de gama alta a comunidades académicas e industriales a bajo costo y alta disponibilidad". De este modo, el módulo de piloto automático Pixhawk ejecuta un sistema operativo en tiempo real (*Real Time Operative System*, RTOS), que proporciona un entorno de estilo POSIX, cuyo software se puede actualizar con un gestor de arranque USB. Estos dispositivos disponen de un código abierto, el cual ha sido en gran parte objeto de este estudio, de modo que ha sido posible editar, programar y ejecutar algoritmos de control propios mediante una herramienta de programación (en nuestro caso *Qt Creator*). Posteriormente, se ha podido comprobar la eficacia de los algoritmos empleados sin necesidad de poner en vuelo al Quadrotor gracias a un entorno de simulación. Esto ha sido de gran ayuda, especialmente en las primeras fases del proyecto, donde los algoritmos no estaban del todo ajustados y podían poner en peligro tanto la estructura del Quadrotor como a las personas a su alrededor.

Por todo ello, y gracias a las numerosas ventajas que presentan los dispositivos Pixhawk, como su accesibilidad o el "plug and play" del que disponen (siendo esta una de sus características más atractivas), se encuentra la motivación necesaria para ahondar en los mismos.

2.2. Antecedentes y Justificación del Trabajo

El proyecto ha sido realizado a partir de la experiencia obtenida durante la estancia en el departamento de Ingeniería de Sistemas y Automática, gracias a una beca colaboración asignada por la UPV. El tiempo total de trabajo de 8 meses ha sido empleado tanto en actividades de carácter práctico, como por ejemplo el montaje de un dron de 4 rotores, soldaduras, montaje de componentes de un PC, etc.; como de carácter teóricas y académicas, como pueden serlo el estudio y análisis de leyes y algoritmos de control, nuevos entornos de SO como Ubuntu, etc. Sin embargo, también ha sido necesario ahondar más en conceptos ya vistos durante el grado para llevar a cabo susodichas tareas, teniendo que prestar especial atención a filtros, señales, comunicación entre diferentes dispositivos, etc.

El grupo de investigación de Pedro García Gil lleva años estudiando la implementación de algoritmos de control para Quadrotors, pero no ha sido sino hasta este año, cuando han empezado a trabajar con los dispositivos Pixhawk. De este modo, mi tarea ha consistido en implementar estrategias con las que ya se había trabajado a lo largo de estos años en esta nueva plataforma, ya que como se cree, supone un papel importante en el futuro para los drones de cuatro rotores de código abierto. Algunos estudios previos sobre Quadrotors realizados por este equipo los podemos encontrar en [2].

Capítulo 3

Estado del Arte

A lo largo de este capítulo se describen diferentes conceptos que serán empleados a lo largo de todo el escrito, así como su contexto, su estado actual y algunos ejemplos de los mismos.

3.1. Concepto de UAS y RPAS

Entendemos como UAS (*Unmanned Aircraft System)* un sistema aéreo no tripulado, donde en vez de ser dirigido por un piloto dentro del mismo, el UAS es controlado desde un operador en el suelo. Sin embargo, esta definición no es lo suficientemente específica para abarcar conceptualmente a un vehículo aéreo de tipo cuadricóptero.

Concretamente, estos pertenecen a un subconjunto dentro de los UAS denominado RPAS (*Remotely Piloted Aircraft System).* Los RPAS, como indican sus siglas, son sistemas aéreos tripulados de manera remota, los cuales presentan dos características bien distinguidas:

- Por un lado, se encuentra el RPA (Remotely Piloted Aircraft), parte que representa la aeronave tripulada por control remoto con un sistema de vuelo semiautomático, la cual es capaz de reaccionar a las indicaciones del piloto que controla el sistema a distancia.
- Por otro lado, el término RPS (Remotely Pilot Station), que es el que hace alusión a la estación remota en tierra desde donde tiene lugar la monitorización de la aeronave en todo instante.

Es bien sabido que a lo largo de la última década los drones han sufrido una notoria evolución y que presentan un papel en la sociedad cada vez más relevante. Los primeros RPAS que surgieron fueron con fines militares, pero gracias a los avances en tecnología y concretamente en electrónica, los componentes necesarios para su realización y uso han experimentado un enorme abaratamiento, permitiéndoles así llegar a formar parte del ámbito urbano y comercial en la actualidad.

Así pues, el hecho de que los RPAS sean controlados remotamente les abre un gran abanico de posibilidades al poder presentar un tamaño mucho menor que si se tuviese que incluir una estancia dentro del mismo para el piloto. Esto disminuye considerablemente su coste y mantenimiento, así como incrementa la agilidad y maniobrabilidad del vehículo, permitiéndole alcanzar lugares que para una persona serian inaccesibles, y, además, sin someterla a ningún tipo de riesgo.

Tal ha sido el auge de este tipo de vehículos durante los últimos años que la Comisión Europea se ha visto obligada a legislar su uso [3]. En lo referente a España, como se indica en [4], los RPAS pueden integrarse junto al resto de tráficos tripulados en espacios aéreos no segregados y en aeródromos. Para ello, según el peso de la aeronave existen diferentes autoridades competentes que regulan su actividad. Buscando la seguridad del ciudadano, su uso queda limitado a la realización de trabajos de carácter técnico o científico, así como vuelos de prueba de producción y mantenimiento, de demostración, investigación y desarrollo de nuevos productos, o para demostrar la seguridad de las operaciones específicas de trabajos de diferente índole.

3.2. Concepto de Quadrotor

Se entiende por Quadrotor, cuadricóptero o cuadri-rotor todo aquel vehículo aéreo no tripulado compuesto por un conjunto de 4 rotores capaces de propulsarlo y mantenerlo en vuelo. A diferencia de otras aeronaves de tipo helicóptero, el Quadrotor se caracteriza por emplear dos pares de hélices fijas, generalmente dispuestas en cruz o equis, las cuales giran en sentidos opuestos. Esto es así ya que de otra forma se produciría un par de giro resultante distinto de cero actuando sobre el dron que lo desestabilizaría. La maniobrabilidad durante el vuelo se consigue variando la velocidad relativa de cada motor. En la figura 1 se muestra el prototipo del Quadrotor Phantom 3 de la empresa DJI:

Figura 1: Quadrotor Phantom 3 de DJI

Así pues, los Quadrotors están emergiendo como una popular plataforma para la investigación en UAVs debido a la simplicidad de su construcción y mantenimiento, su habilidad para mantenerse suspendidos en el aire, y su capacidad de aterrizar o despegar verticalmente. Hoy en día ya son muchas las aplicaciones en las que se ven involucrados los Quadrotors, pues resultan muy competentes a la hora de realizar tareas tan dispares como tratamientos aéreos fitosanitarios, vigilancia de incendios forestales, filmación dentro de la industria cinematográfica, asistencia sanitaria, agricultura de precisión, y un largo etcétera.

No obstante, el Quadrotor no es sino uno más dentro de todos los tipos de multi-rotor. Si bien es cierto que el Quadrotor es el modelo más extendido y sobre el que más se ha enfocado la investigación y desarrollo durante los últimos años por ser el más sencillo, existen otros modelos de multi-rotores donde varía tanto el número de motores como su distribución. De este modo, existen por ejemplo modelos con 3 motores dispuestos en forma de Y, también conocidos como Trirotors; o variantes con 6 y 8 rotores con diferentes configuraciones. En la figura 2 se muestra en la parte derecha un Hexarotor con configuración circular, mientras que en la parte izquierda presenta una configuración en Y.

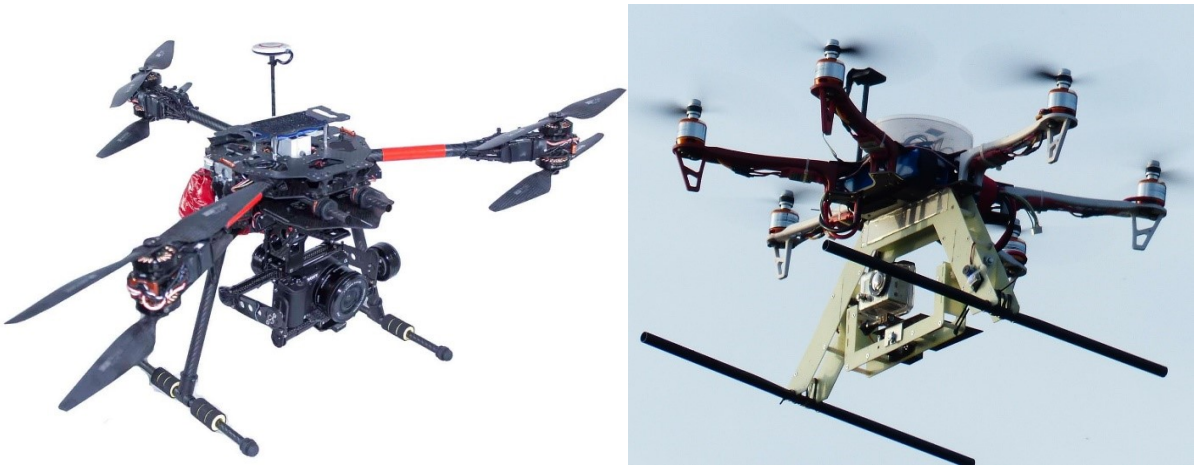


Figura 2: Hexarotor en Y (izquierda), Hexarotor circular (derecha)

Se observa por tanto la gran variedad de diseños que se pueden obtener ya sea cambiando la distribución y/o el número de los rotores. Estas son especificaciones que se elegirán teniendo en cuenta las actividades que el dron vaya a realizar.

3.3. Control del dron

Una vez asimilada la arquitectura del dron, hay que entender como éste es capaz de volar y enfrentarse a todo tipo de perturbaciones sin perder la estabilidad. Es aquí donde entra en juego el controlador. La mayoría de multi-rotores cuentan con control de tipo asistido donde la acción de control se divide en dos partes. Por un lado, existe un algoritmo de control encargado de estabilizar el helicóptero que funciona de manera autónoma, y por otro, encontramos el control de posición, llevado a cabo por el piloto desde una emisora de radiofrecuencia o por ordenador.

Sin embargo, para poder realizar dicho control, es necesario disponer de los sensores adecuados para obtener medidas de las variables a controlar. Para ello, los controladores presentan unidades de medida inercial, también conocidas por sus siglas como IMUs, las cuales se encargan de obtener la orientación del dron, y enviarle dicha información al controlador para que este pueda actuar. Más información al respecto la podemos encontrar en [5] y [6]. A su vez, encontramos otros tipos de sensores como los sensores de ultrasonidos, los barométricos, magnetómetros y GPS. Todos ellos con

sus ventajas y desventajas, los cuales suelen integrarse de manera conjunta para que unos solventen las carencias de otros y viceversa.

No obstante, el dron necesita de un ordenador de abordo que sea capaz de procesar la información que le suministran dichos sensores, así como ejecutar el algoritmo de control adecuadamente. En la figura 3 se muestra como ejemplo de ordenador de abordo el dispositivo Pixhawk, objeto de estudio de este TFG.

Figura 3: Pixhawk

Actualmente, la mayoría de multi-rotores implementan un controlador de tipo PID [7]. Esto es debido a la sencillez y eficacia que estos han demostrado durante los últimos años. Sin embargo, los PID conllevan determinados problemas asociados a la parte integral que otras estrategias de control más avanzadas como el UDE o el ESO son capaces de resolver, como se demostrará más adelante. Finalmente, existen muchas otras estrategias de control como lo son las de tipo predictivo, algoritmos de aprendizaje, etc.; las cuales sin embargo escapan al alcance de este trabajo debido a su excesiva complejidad.

Capítulo 4

Modelo teórico del Quadrotor

Durante este capítulo se pretende deducir y analizar paso a paso todo el modelo teórico y ecuaciones características que representan el comportamiento del Quadrotor durante el vuelo.

El Quadrotor es un prototipo muy útil a la hora de entender el fenómeno aerodinámico que se produce en las máquinas voladoras que pueden desplazarse y mantenerse suspendidas en el aire.

Para abordar paso a paso los diferentes conceptos que se presentan, este capítulo se ha estructurado del siguiente modo: en el apartado 4.1 se establecen cuáles son las variables características de un Quadrotor; en el 4.2 se indica el sistema de ejes en el que situar la aeronave y el tipo de notación que se empleará a lo largo de todo el escrito; en 4.3 y 4.4 se abarca respectivamente el modelo cinemático y dinámico del dron; en el punto 4.5 se agruparán las ecuaciones del modelo obtenidas en los apartados anteriores; y finalmente en el 4.6 se mostrará una simplificación del modelo.

Es pertinente mencionar que para el desarrollo del modelo se ha optado por seguir el enfoque Newtoniano frente al de Euler-Lagrange, puesto que resulta más ilustrativo. Gran parte de este capítulo se ha inspirado en los diferentes razonamientos elaborados en [8], [9], y [10].

4.1. Variables características de un Quadrotor

Entendemos el cuadricóptero como un sólido rígido libre en el espacio. Por ello, este presenta 6 grados de libertad, los cuales denotaremos como: $(x, y, z, \phi, \theta, \psi) \in \mathbb{R}^6$.

- (x, y, z) representan la posición de su centro de masas (c.d.m.).
- (ϕ, θ, ψ) son los ángulos de Euler (alabeo, cabeceo y guiñada respectivamente), que denotan la orientación del objeto. De ahora en adelante nos referiremos a ellos por sus nombres en inglés: roll, pitch y yaw.

4.2. Sistema de ejes

4.2.1. Sistema de referencia inercial o "Earth Frame"

Se entiende como marco de referencia inercial aquel que describe el movimiento del Quadrotor visto desde un observador fijo, solidario a la tierra. Para ello, se emplea un sistema de referencia muy utilizado en la ingeniería aeroespacial conocido como NED (North, East, Down), donde el vector $\vec{\imath}$ apunta al norte geográfico, el $\vec{\jmath}$ al este y el $\vec{k}$ al centro de la tierra. Cabe destacar que la altura del dron z generalmente adoptará un valor negativo, ya que su vector $\vec{k}$ apunta al centro de la tierra.

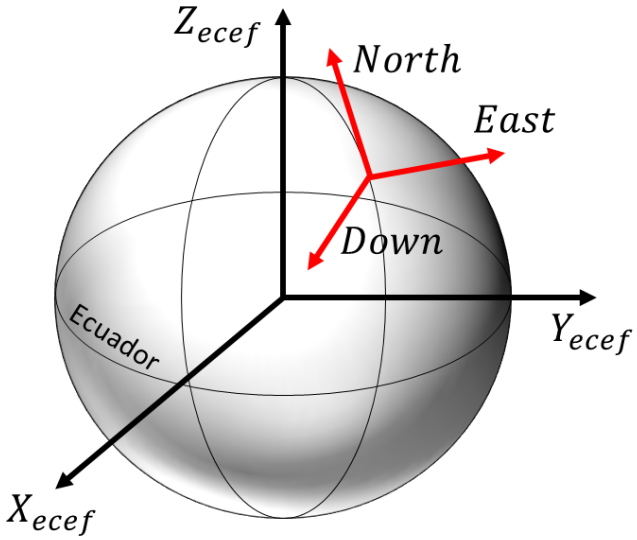


Figura 4: Sistema de referencia inercial

Para indicar que cierta variable está expresada en el entorno del "Earth Frame" usaremos el superíndice ${}^{e}\mathcal{O}$

4.2.2. Sistema de referencia del Quadrotor o "Body Frame"

Para especificar la orientación de la aeronave en el espacio respecto a un sistema de referencia fijo se emplean los ángulos de Euler. Este marco de referencia se encuentra fijo en el c.d.m. de manera que se desplaza juntamente con el Quadrotor. Su vector $\vec{\jmath}$ apuntará a la cabeza del dron, $\vec{\imath}$ hacia uno de sus laterales, y su vector $\vec{k}$ se mantendrá perpendicular al plano de las hélices, de modo que los ejes x, y, z quedan como muestra la figura 5.

Para denotar que la variable pertinente está expresada en el marco del "Body Frame" emplearemos el superíndice ${}^{b}\mathcal{O}$.

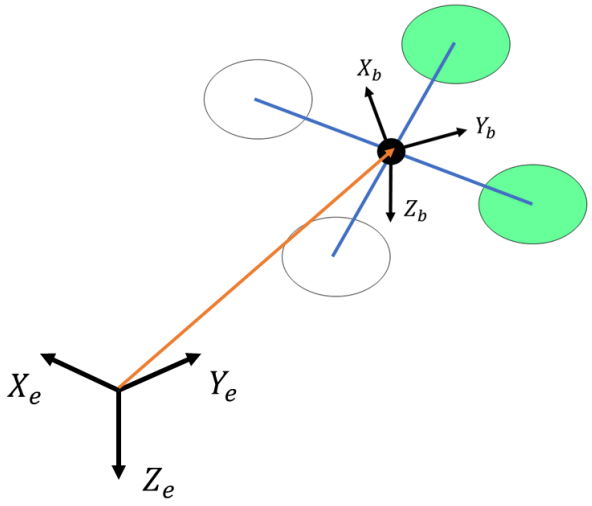


Figura 5: Sistema de referencia del Quadrotor

4.2.3. Cambios de base a partir de la matriz de rotación

Para pasar del sistema de referencia ${}^{e}\mathcal{O}$ al sistema de referencia ${}^{b}\mathcal{O}$, lo primero será trasladar el origen de ${}^{e}\mathcal{O}$ al c.d.m. de la nave. A continuación, se irá rotando uno a uno los ángulos yaw, roll y pitch a lo largo de los ejes z, y, x respectivamente. De este modo, cualquier posible orientación del Quadrotor queda definida por estas tres rotaciones, por lo que $\{x, y, z, \phi, \theta, \psi\}$ determina unívocamente la posición del dron en el espacio.

La matriz de cosenos directores R que resulta de aplicar sucesivas rotaciones es la siguiente:

$$R_e^b = \begin{bmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \sin\phi\cos\theta \\ \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi & \cos\phi\cos\theta \end{bmatrix}$$

$$ {}^{b}\mathcal{O} = R_e^b \cdot {}^{e}\mathcal{O} \tag{4.1}$$

Por otro lado, si queremos pasar del sistema de referencia ${}^{b}\mathcal{O}$ al sistema de referencia ${}^{e}\mathcal{O}$ lo único que tendremos que hacer será obtener la inversa de la matriz de cosenos directores anterior, para lo cual, al tratarse de una matriz ortonormal, bastará con transponerla:

$$R_b^e = R_e^{b\,T} \tag{4.2}$$

$$ {}^{e}\mathcal{O} = R_b^e \cdot {}^{b}\mathcal{O} \tag{4.3}$$

$$R_b^e = \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix}$$

4.3. Modelo cinemático del Quadrotor

Los helicópteros convencionales usan un dispositivo mecánico conocido como plato cíclico para alterar el ángulo de inclinación de las palas del rotor de forma cíclica, para así obtener los pares de control sobre el roll y el pitch del vehículo. Por el contrario, el Quadrotor no tiene un plato cíclico para modificar el ángulo del rotor, sino que consta de 4 rotores con una hélice cada uno, los cuales variando su velocidad relativa pueden conseguir inclinar el dron en el ángulo deseado.

¿Pero cómo se explica la relación que existe entre la inclinación que sufre el dron y la velocidad angular con la que la experimenta? Pues bien, esto se encarga de explicarlo el modelo cinemático del Quadrotor. Los resultados siguientes que aquí se muestran se han obtenido de [7].

Primero, se expresa la velocidad de traslación del c.d.m. de la nave en el sistema de referencia inercial, a partir de la velocidad del sistema de referencia móvil rotándolo con la matriz calculada anteriormente R_b^e:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{pmatrix}^e = R_b^e \cdot \begin{pmatrix} v_x^b \\ v_y^b \\ v_z^b \end{pmatrix} \tag{4.4}$$

Donde (v_x^b, v_y^b, v_z^b) representan respectivamente, las velocidades en (x, y, z) en el marco de referencia del Quadrotor ${}^b\mathcal{O}$.

Segundo, tras varias rotaciones se pueden relacionar los ángulos de Euler con las velocidades angulares de giro expresadas en el marco del Quadrotor ${}^b\mathcal{O}$ a partir de la siguiente matriz de rotación:

$$\begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \cdot \begin{pmatrix} w_x^b \\ w_y^b \\ w_z^b \end{pmatrix} \tag{4.5}$$

Donde (w_x^b, w_y^b, w_z^b) representan las velocidades angulares expresadas en el sistema de referencia ${}^b\mathcal{O}$.

4.4. Modelo dinámico del Quadrotor

4.4.1. Fuerzas

Peso. Por el hecho de encontrarse el Quadrotor dentro del campo gravitatorio terrestre, este se ve sometido a una fuerza peso aplicada en su c.d.m., tal que:

$${}^e\vec{F}_g = \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix}$$

La cual, empleando la matriz de rotación R_e^b para obtenerla respecto al sistema de referencia solidario al Quadrotor ${}^b\mathcal{O}$, queda:

$$^{b}\vec{F}_g = R_e^b \cdot {}^{e}\vec{F}_g \tag{4.6}$$

$$^{b}\vec{F}_g = mg \cdot \begin{bmatrix} -\sin\theta \\ \sin\phi\cos\theta \\ \cos\theta\cos\phi \end{bmatrix} \tag{4.7}$$

Empuje. La fuerza F_i producida por el motor i es proporcional al cuadrado de su velocidad angular:

$$F_i = k \cdot w_i^2$$

Cabe destacar que cada motor puede girar solo en un sentido, de modo que la fuerza F_i adopta siempre un signo positivo, como se muestra en la figura 6.

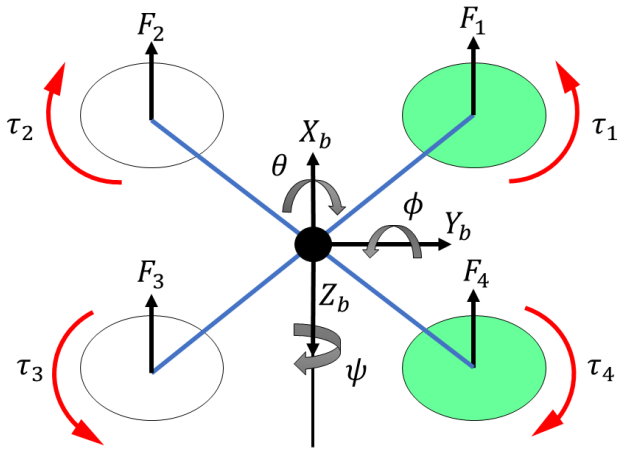


Figura 6: Acciones sobre el Quadrotor

La velocidad de giro del motor i está controlada por variadores de velocidad que responden a pulsos eléctricos para fijar la velocidad angular deseada:

$$w_i = k_{var} \cdot \delta$$

El empuje total de los motores será:

$$F_{Mot} = F_1 + F_2 + F_3 + F_4$$

La fuerza total resultante que actúa sobre el c.d.m. del Quadrotor teniendo en cuenta tanto el empuje como la fuerza gravitatoria es, por tanto:

$$F_{Res} = -F_{Mot} + F_g$$

La cual, expresada de forma vectorial en el marco de referencia del Quadrotor adopta la forma:

$$ {}^{b}\vec{F}_{Res} = \begin{bmatrix} 0 \\ 0 \\ -(F_1 + F_2 + F_3 + F_4 - F_g) \end{bmatrix} \tag{4.8} $$

4.4.2. Momentos

Roll y Pitch. **Momento debido al empuje aerodinámico.** Como se puede deducir de la figura 6, si se quiere, por ejemplo, inclinar el Quadrotor hacia adelante en el ángulo del pitch se tendrá que aumentar la velocidad de los motores traseros mientras se reduce la velocidad de los motores delantero. Del mismo modo, el movimiento de roll se obtiene de la diferencia de los motores laterales.

Los pares de acción sobre el roll y el pitch serán respectivamente:

$$ \tau_\phi = (-F_1 - F_2 + F_3 + F_4) \cdot l/2 \tag{4.9} $$

$$ \tau_\theta = (F_1 + F_4 - F_1 - F_3) \cdot l/2 \tag{4.10} $$

Donde l representa la distancia entre el rotor i y el eje de giro.

Yaw. Momento debido al arrastre aerodinámico. El movimiento de yaw se produce al aumentar el par de giro de los motores delantero y trasero, mientras se disminuye el de los motores laterales, o viceversa, según el sentido en el que se desee rotar.

La ecuación que lo representa atendiendo al sentido de τ_i visto en la figura 6 es:

$$ \tau_\psi = -\tau_1 + \tau_2 - \tau_3 + \tau_4 \tag{4.11} $$

Las aspas de los rotores del dron, al encontrarse en el seno de un fluido como lo es el aire, generan fricción con el mismo. Sin embargo, es pertinente hacer notar que los rotores 1 y 3 giran en sentido antihorario, mientras que los 2 y 4 giran en sentido horario. Con esta configuración, los pares aerodinámicos producidos por esa reacción al par de giro de cada motor tienden a cancelarse, por lo que es posible conseguir anular el par que haría rotar al dron en yaw escogiendo las velocidades adecuadas.

Es importante mencionar que todas estas variaciones expresadas a lo largo de 4.4 se pueden lograr manteniendo el empuje total que sustenta al dron en el aire constante.

4.4.3. Aceleraciones lineales

Como es bien sabido, la segunda ley de Newton establece que:

$$ m \cdot \frac{dv}{dt} = \vec{F}_{Res} $$

Siendo m la masa del Quadrotor, $v = (v_x^b, v_y^b, v_z^b)^T$ la velocidad de traslación del c.d.m. expresada en el entorno móvil ${}^{b}\mathcal{O}$, y $\vec{F}_{Res}$ la fuerza de empuje resultante expresada en el mismo sistema de referencia.

Sin embargo, las leyes de Newton solo son aplicables en marcos de referencia inerciales. Por tanto, debido a que el sistema de referencia ${}^{b}\mathcal{O}$ gira con una velocidad angular $w_b = (w_x^b, w_y^b, w_z^b)^T$, será necesario aplicar la ecuación de Coriolis para realizar el cambio al sistema de referencia inercial ${}^{e}\mathcal{O}$, con tal de poder expresar la derivada temporal correctamente y así, poder aplicar las leyes de Newton:

$$\frac{dv}{dt_e} = \left(\frac{dv}{dt_b} + w_b \times v\right)$$

Sustituyendo nos queda:

$$m \cdot \left(\frac{dv}{dt_b} + w_b \times v\right) = \vec{F}_{Res}$$

Y despejando el término de la derivada, y resolviendo el producto vectorial de las velocidades concluimos que:

$$\begin{pmatrix} \dot{v_x^b} \\ \dot{v_y^b} \\ \dot{v_z^b} \end{pmatrix} = \begin{bmatrix} -w_z^b v_y^b + w_y^b v_z^b \\ -w_x^b v_z^b + w_z^b v_x^b \\ -w_y^b v_x^b + w_x^b v_y^b \end{bmatrix} + \frac{1}{m} \cdot \vec{F}_{Res} \tag{4.12}$$

Finalmente, sustituyendo $\vec{F}_{Res}$ por la expresión obtenida en (4.8), y distinguiendo entre el empuje de los motores y el peso obtenemos que:

$$\begin{pmatrix} \dot{v_x^b} \\ \dot{v_y^b} \\ \dot{v_z^b} \end{pmatrix} = \begin{bmatrix} -w_z^b v_y^b + w_y^b v_z^b \\ -w_x^b v_z^b + w_z^b v_x^b \\ -w_y^b v_x^b + w_x^b v_y^b \end{bmatrix} + \frac{1}{m} \cdot \begin{bmatrix} 0 \\ 0 \\ -(F_1 + F_2 + F_3 + F_4) \end{bmatrix} + g \cdot \begin{bmatrix} -\sin\theta \\ \sin\phi\cos\theta \\ \cos\theta\cos\phi \end{bmatrix} \tag{4.13}$$

4.4.4. Aceleraciones angulares

En lo que respecta a la rotación, la segunda ley de Newton establece que la suma de los momentos angulares aplicados sobre el c.d.m. del cuerpo debidos a fuerzas externas, es igual a la derivada del momento angular en el c.d.m. respecto al tiempo:

$$\tau_{Res} = \sum \vec{M} = \frac{dL}{dt}$$

Donde $\tau_{Res} = (\tau_\phi, \tau_\theta, \tau_\psi)$ es el par resultante aplicado y L es el momento angular.

Al igual que sucedía anteriormente, la derivada temporal debe estar expresada en un sistema de referencia inercial, por lo que se debe volver a recurrir a la ecuación de Coriolis, la cual toma la siguiente forma esta vez:

$$\frac{dL}{dt_e} = \left(\frac{dL}{dt_b} + w_b \times L\right)$$

Definiendo L como el producto entre la velocidad angular expresada en ${}^{b}\mathcal{O}$ y la matriz de inercia I, que por simplicidad se considera diagonal:

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

$$L = I \cdot w_b$$

Y sustituyendo en la ecuación de Coriolis, se obtiene que:

$$\frac{dL}{dt_e} = \left(\frac{dw_b}{dt_b} + w_b \times (I \cdot w_b)\right) = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \cdot \begin{pmatrix} \dot{w_x^b} \\ \dot{w_y^b} \\ \dot{w_z^b} \end{pmatrix} + \begin{bmatrix} (I_{zz} - I_{yy}) \cdot w_y^b \cdot w_z^b \\ (I_{xx} - I_{zz}) \cdot w_x^b \cdot w_z^b \\ (I_{yy} - I_{xx}) \cdot w_y^b \cdot w_x^b \end{bmatrix} \quad (4.14)$$

Así pues, se llega a la expresión final sustituyendo la parte izquierda de la ecuación anterior por el par resultante τ_{Res}, y despejando la aceleración angular:

$$\begin{pmatrix} \dot{w_x^b} \\ \dot{w_y^b} \\ \dot{w_z^b} \end{pmatrix} = \begin{bmatrix} \frac{(-I_{zz}+I_{yy})}{I_{xx}} \cdot w_y^b \cdot w_z^b \\ \frac{(-I_{xx}+I_{zz})}{I_{yy}} \cdot w_x^b \cdot w_z^b \\ \frac{(-I_{yy}+I_{xx})}{I_{zz}} \cdot w_y^b \cdot w_x^b \end{bmatrix} + \begin{pmatrix} \frac{\tau_\phi}{I_{xx}} \\ \frac{\tau_\theta}{I_{yy}} \\ \frac{\tau_\psi}{I_{zz}} \end{pmatrix} \quad (4.15)$$

4.5. Modelo dinámico resultante

Finalmente, cogiendo las ecuaciones (4.4), (4.5), (4.13) y (4.15), se obtiene el modelo completo del Quadrotor:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{pmatrix} = \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix} \cdot \begin{pmatrix} v_x^b \\ v_y^b \\ v_z^b \end{pmatrix}$$

$$\begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \cdot \begin{pmatrix} w_x^b \\ w_y^b \\ w_z^b \end{pmatrix}$$

$$\begin{pmatrix} \dot{v_x^b} \\ \dot{v_y^b} \\ \dot{v_z^b} \end{pmatrix} = \begin{bmatrix} -w_z^b v_y^b + w_y^b v_z^b \\ -w_x^b v_z^b + w_z^b v_x^b \\ -w_y^b v_x^b + w_x^b v_y^b \end{bmatrix} + \frac{1}{m} \cdot \begin{bmatrix} 0 \\ 0 \\ -(F_1 + F_2 + F_3 + F_4) \end{bmatrix} + g \cdot \begin{bmatrix} -\sin\theta \\ \sin\phi\cos\theta \\ \cos\theta\cos\phi \end{bmatrix}$$

$$\begin{pmatrix} \dot{w_x^b} \\ \dot{w_y^b} \\ \dot{w_z^b} \end{pmatrix} = \begin{bmatrix} \frac{(-I_{zz} + I_{yy})}{I_{xx}} \cdot w_y^b \cdot w_z^b \\ \frac{(-I_{xx} + I_{zz})}{I_{yy}} \cdot w_x^b \cdot w_z^b \\ \frac{(-I_{yy} + I_{xx})}{I_{zz}} \cdot w_y^b \cdot w_x^b \end{bmatrix} + \begin{pmatrix} \frac{\tau_\phi}{I_{xx}} \\ \frac{\tau_\theta}{I_{yy}} \\ \frac{\tau_\psi}{I_{zz}} \end{pmatrix}$$

4.6. Simplificación del modelo

Para llevar a cabo la simplificación del modelo del Quadrotor mostrado en 4.5 se han tenido en cuenta las siguientes hipótesis:

- **Hipótesis de los ángulos pequeños**. Durante el vuelo, se asume que la nave nunca se inclinará un ángulo mayor de 20 grados, con tal de no dar giros demasiado bruscos que puedan propiciar la inestabilidad. Por ello, la ecuación (4.5) se puede simplificar del siguiente modo:

$$\begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} = \begin{bmatrix} 1 & \sin\phi \tan\theta & \cos\phi \tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi \sec\theta & \cos\phi \sec\theta \end{bmatrix} \cdot \begin{pmatrix} w_x^b \\ w_y^b \\ w_z^b \end{pmatrix} \cong \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{pmatrix} w_x^b \\ w_y^b \\ w_z^b \end{pmatrix} = \begin{pmatrix} w_x^b \\ w_y^b \\ w_z^b \end{pmatrix} \quad (4.16)$$

- **Hipótesis de las velocidades angulares pequeñas.** Del mismo modo que los ángulos, también consideraremos que las velocidades angulares a las que se someterá el dron nunca serán demasiado grandes, por lo que las ecuaciones (4.13) y (4.15) quedarán como:

$$\begin{pmatrix} \dot{v_x^b} \\ \dot{v_y^b} \\ \dot{v_z^b} \end{pmatrix} = \frac{1}{m} \cdot \begin{bmatrix} 0 \\ 0 \\ -(F_1 + F_2 + F_3 + F_4) \end{bmatrix} + g \cdot \begin{bmatrix} -\sin\theta \\ \sin\phi \cos\theta \\ \cos\theta \cos\phi \end{bmatrix} \quad (4.17)$$

$$\begin{pmatrix} \dot{w_x^b} \\ \dot{w_y^b} \\ \dot{w_z^b} \end{pmatrix} = \begin{pmatrix} \frac{\tau_\phi}{I_{xx}} \\ \frac{\tau_\theta}{I_{yy}} \\ \frac{\tau_\psi}{I_{zz}} \end{pmatrix} \quad (4.18)$$

Atendiendo a (4.16) y (4.18) obtenemos que:

$$\begin{pmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{pmatrix} = \begin{pmatrix} \frac{\tau_\phi}{I_{xx}} \\ \frac{\tau_\theta}{I_{yy}} \\ \frac{\tau_\psi}{I_{zz}} \end{pmatrix} \quad (4.19)$$

Por otra parte, teniendo en cuenta (4.17) y (4.4), operando deducimos que:

$$\begin{pmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{pmatrix} = \begin{bmatrix} -\cos\phi \sin\theta \cos\psi - \sin\phi \sin\psi \\ -\cos\phi \sin\theta \sin\psi + \sin\phi \cos\psi \\ -\cos\phi \cos\theta \end{bmatrix} \cdot \frac{F_{Mot}}{m} + \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix} \tag{4.20}$$

Por lo que desglosando (4.20) y (4.19) se llega finalmente al modelo simplificado del Quadrotor, el cual, como se puede apreciar, se muestra más intuitivo que el completo:

$$\ddot{x} = (-\cos\phi \sin\theta \cos\psi - \sin\phi \sin\psi) \cdot \frac{F_{Mot}}{m}$$

$$\ddot{y} = (-\cos\phi \sin\theta \sin\psi + \sin\phi \cos\psi) \cdot \frac{F_{Mot}}{m}$$

$$\ddot{z} = (-\cos\phi \cos\theta) \cdot \frac{F_{Mot}}{m} + g$$

$$\ddot{\phi} = \frac{\tau_\phi}{I_{xx}}$$

$$\ddot{\theta} = \frac{\tau_{\ddot{\theta}}}{I_{yy}}$$

$$\ddot{\psi} = \frac{\tau_\psi}{I_{zz}}$$

Sin embargo, a estas ecuaciones falta por añadirles un factor desconocido que incluya todo tipo de perturbaciones (d) y que a la vez es función de las dinámicas entrelazadas de los distintos ángulos. De esta forma, tomando por ejemplo la ecuación del pitch, queda del siguiente modo:

$$\ddot{\theta} = \frac{\tau_{\ddot{\theta}}}{I_{yy}} + f(\dot{\phi}, \dot{\theta}, \dot{\psi}, d)$$

y resolviéndola por Laplace con condiciones iniciales nulas se obtiene:

$$s^2 \cdot \theta = \frac{1}{I_{yy}} \cdot \tau_\theta + f$$

De donde, considerando parte del término $\frac{1}{I_{yy}} \cdot \tau_\theta$ como una constante K_θ, y otra parte como una entrada del sistema, se obtiene un modelo de doble integrador como el siguiente, el cual se empleará más adelante:

$$\theta = \frac{K_\theta}{s^2} \cdot u + \frac{1}{s^2} \cdot d \tag{4.21}$$

Capítulo 5

Descripción de la plataforma

En esta sección se indica detalladamente los pasos que se han seguido para construir la plataforma de vuelo y las características de sus componentes principales.

Para llevar a cabo todo el proyecto y poder cumplir los objetivos propuestos, el primer paso ha sido construir una plataforma firme y resistente que sea capaz de hacer volar al dron. Así mismo, también ha sido necesario que esta tenga la estructura y tamaño adecuado para poder portar todos los dispositivos imprescindibles para que las funciones del Quadrotor que sean necesarias según la fase del proyecto estén operativas.

5.1. Estructura de soporte

El soporte se trata del chasis F450 de la empresa DJI [11], el cual cuesta alrededor de 40 €. Este se muestra en la figura 7. Es muy resistente, por lo que es capaz de soportar cualquier tipo de golpe o choque sin comprometer la parte electrónica de la nave. Incluye una placa para soldar las conexiones a los variadores.

Figura 7: Chasis F450 DJI

5.2. Motores eléctricos y Variadores

Los motores incorporados al dron son el modelo E310 2312 de la marca DJI. Se trata de un sistema de propulsión diseñado específicamente para helicópteros multi-rotor que pesan entre 1 y 2.5 kg. Gracias a sus rodamientos de 4mm x 12mm y su eje de acero de aleación de alta resistencia, los motores ofrecen una mayor resistencia y durabilidad que otros motores del mercado.

Así mismo, los variadores o ESCs (del inglés, *"Electronic Speed Control"*) son dispositivos capaces de controlar la velocidad de rotación del motor atendiendo a pulsos eléctricos. Con sensores integrados, son capaces de obtener diagnósticos en tiempo real del estado del sistema de propulsión. Cabe destacar que estos emplean arquitectura de onda sinusoidal para reducir el consumo de energía y aumentar con ello la eficiencia en todas las maniobras, aunque también permiten la clásica comunicación PWM.

Los motores se muestran en la parte izquierda de la figura 8, y los ESCs en la parte derecha.

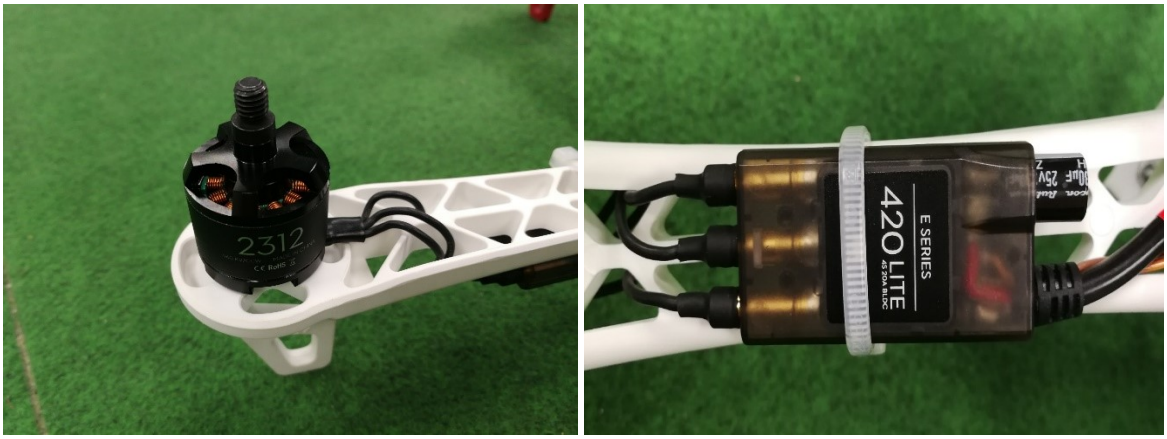


Figura 8: Motor E310 2312 (izquierda), variador ESC 420 (derecha)

5.3. Hélices

Las hélices empleadas en la plataforma se tratan de las Z-Blade de DJI. Éstas están optimizadas aerodinámicamente en base a su geometría, pasando por un estricto control de calidad. Las mismas se muestran en la figura 9.

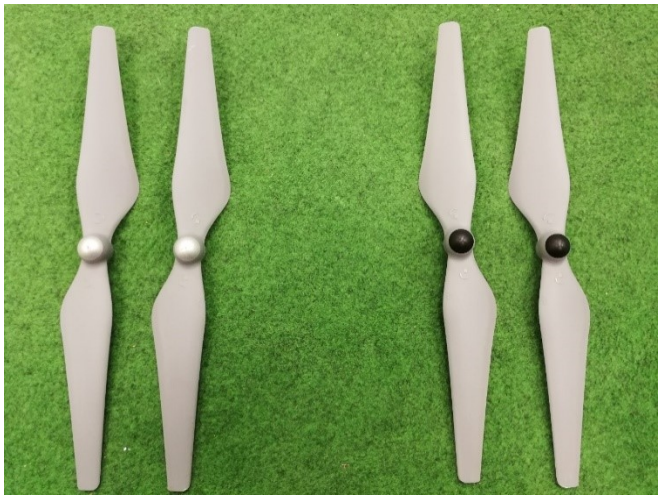


Figura 9: Hélices Z-Blade DJI

5.4. Batería

Han sido dos las baterías empleadas a lo largo del proyecto. Ambas presentan un diseño compacto, pensado especialmente para aplicaciones donde se requiere una forma determinada y un tamaño ajustado, como es el caso. A continuación, se muestran las especificaciones de cada una:

- *Batería Zippy Compact 5800 (figura 10 (izquierda)):*
  Capacidad: 5800mAh
  Voltaje: 3S1P / 3 Cell / 11.1V
  Descarga: 25C Constante / 35C Burst
  Peso: 411g (incluyendo cable, enchufe y envoltorio)
  Dimensiones: 154x45x29mm
  Balance de enchufe: JST-XH
  Enchufe de descarga: Bullet 5.5mm

- *Batería Tattu 1300 (figura 10 (derecha)):*
  Capacidad: 1300 mAh
  Voltaje: 3S1P / 3 Cell / 11.1V
  Descarga: 45C Constante / 90C Burst
  Peso: 120g (incluyendo cable, enchufe y envoltorio)
  Dimensiones: 73x32x22mm
  Balance de enchufe: JST-XH R
  Enchufe de descarga: Bullet 5.5mm

Figura 10: Batería Zippy (izquierda), Batería Tattu (derecha)

5.5. Pixhawk

El dispositivo Pixhawk es un autopiloto de alto rendimiento destinado a helicópteros, multi-rotores, automóviles, embarcaciones y, en definitiva, cualquier plataforma robótica capaz de moverse. Dada la importancia de este dispositivo durante todas las etapas del trabajo y al ser objeto de estudio, se ha optado por asignarle un capítulo entero al mismo dentro de este documento donde se recopila toda su información. Para ello, dirigirse al Capítulo 6.

5.6. GPS

Para el control de posición que se ha realizado sobre la máquina se ha decidido optar por un set de 2 placas C94-M8P y dos antenas GPS de la marca ublox© [12], disponiendo una placa y una antena en la estación de referencia fija, y la otra placa y antena en el Quadrotor, con el fin de obtener un sistema diferencial de GPS. Su implementación ha resultado relativamente sencilla ya que el programa *QGroundControl* del que se hablará a continuación ha identificado el sistema RTK-GPS nada más conectarlo. Más información respecto al GPS se encuentra en el Capítulo 8.

5.7. Software y Entorno de simulación

La primera tarea a realizar ha sido elegir el sistema operativo con el que se iba a trabajar. En este caso, se ha optado por Linux, concretamente por Ubuntu 16.04 ya que era el único SO que permitía aprovechar todas las funciones de los programas que a continuación se describen, así como diferentes entornos de simulación.

5.7.1. Qt Creator

El firmware del PX4 se ha obtenido a partir de su página web [13]. Sin embargo, para trabajar con él se necesita una herramienta de programación capaz no solo de compilar el código, sino también de trabajar con el proyecto entero y que permita cambiar de un archivo a otro fácilmente sin perder de vista la ruta donde nos encontramos. Para ello, se ha optado por el programa *Qt Creator* debido a su intuitiva interfaz, predictor de lenguaje de programación y sistema de organización del texto en forma de colores. Todo el código del Pixhawk se ha trabajado en lenguaje C++. En la figura 11 se muestra la interfaz del programa.

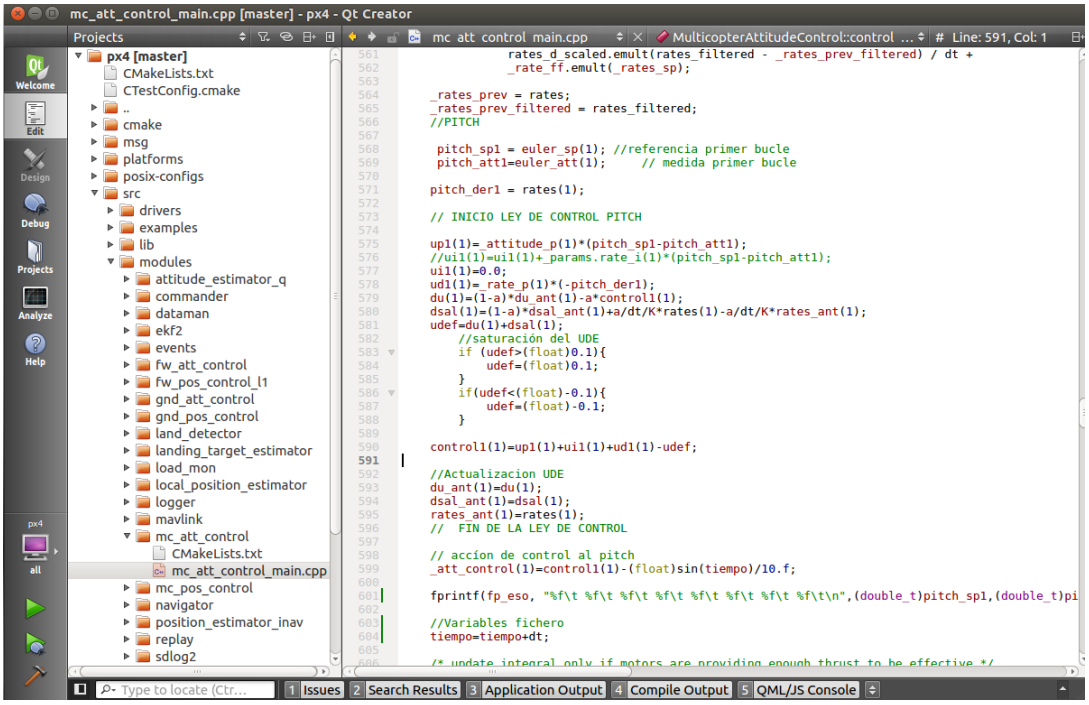


Figura 11: Ventana de Qt Creator

5.7.2. QGroundControl

La siguiente aplicación a tener en cuenta se trata del *QGroundControl* [14]. Esta es una herramienta básica en lo que se refiere a comunicación entre el dron y el PC. *QGroundControl* ofrece control total durante el vuelo y planificación de misiones para cualquier dron habilitado para MAVlink. Proporciona configuración para vehículos propulsados de diferentes tipos, entre ellos los que funcionan con PX4, de modo que a través de él ha sido posible alterar todos aquellos parámetros del dron que se desease. A su vez, permite calibrar todos los sensores y comprobar su correcto funcionamiento. De nuevo, la intuitiva interfaz que presenta ha sido un factor clave a la hora de trabajar con esta herramienta, así como la detección automática del modelo del Quadrotor con el que se está trabajando con tan solo conectarlo al ordenador. En la imagen 12 se muestra un ejemplo del mismo, y en la figura 13 una de las pestañas de configuración de parámetros.

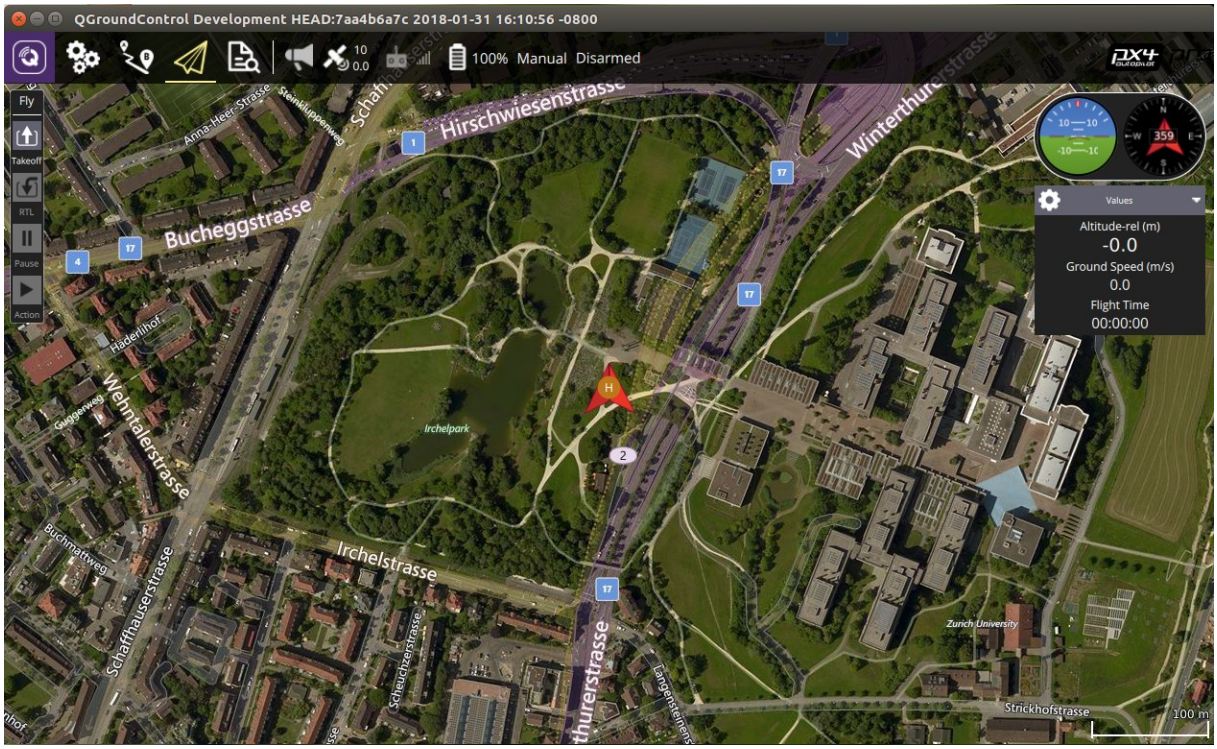


Figura 12: Ventana principal de QGroundControl

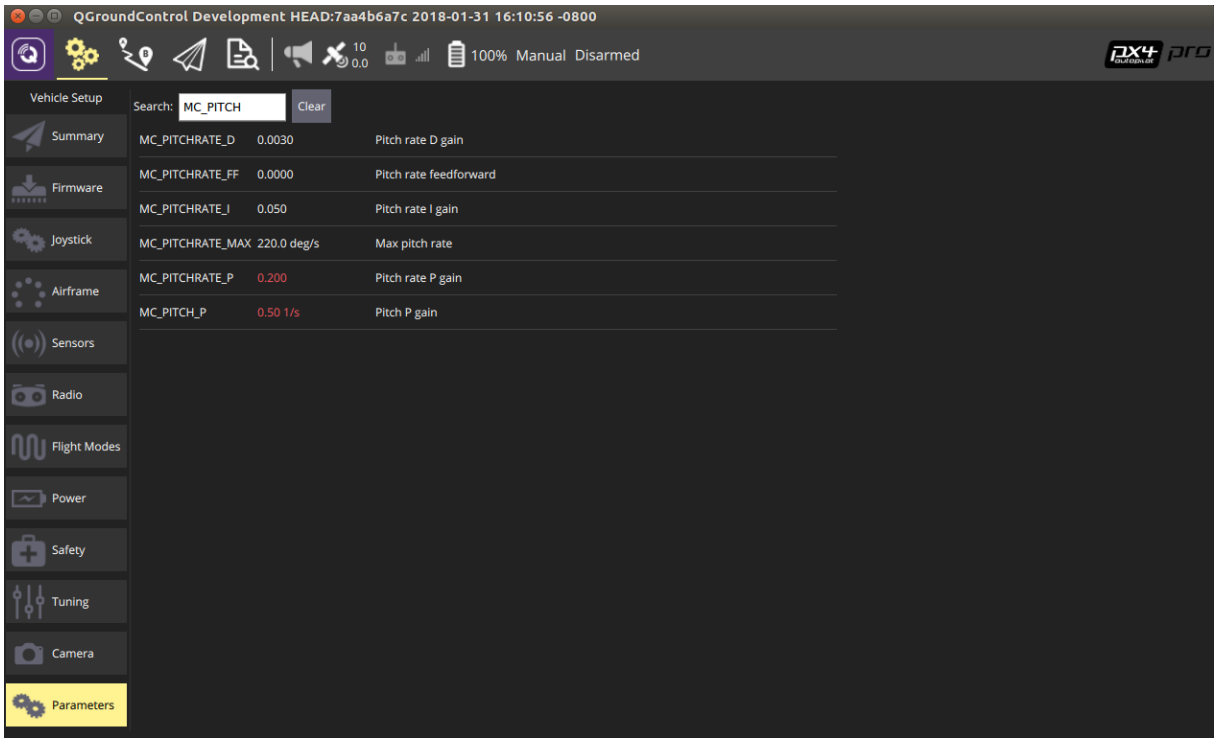


Figura 13: Ventana de parámetros de QGroundControl

5.7.3. Simulador jMAVSim

jMAVSim ha sido la herramienta principal empleada para la simulación de vuelos del Quadrotor. *jMAVSim* es un simulador de multi-rotores que permite volar drones que utilizan el código PX4. Se ha empleado, entre otras cosas, para comprobar que el vehículo podía despegar, volar de manera estable, y ajustar diversos parámetros. Gracias a él, se ha podido observar el comportamiento del dron en vuelo con el algoritmo de control propio de manera simulada, de modo que no había ningún peligro que comprometiese la estructura del helicóptero ni la salud de las personas de alrededor. En la figura 14 se muestra el entorno de simulación de *jMAVSim*.

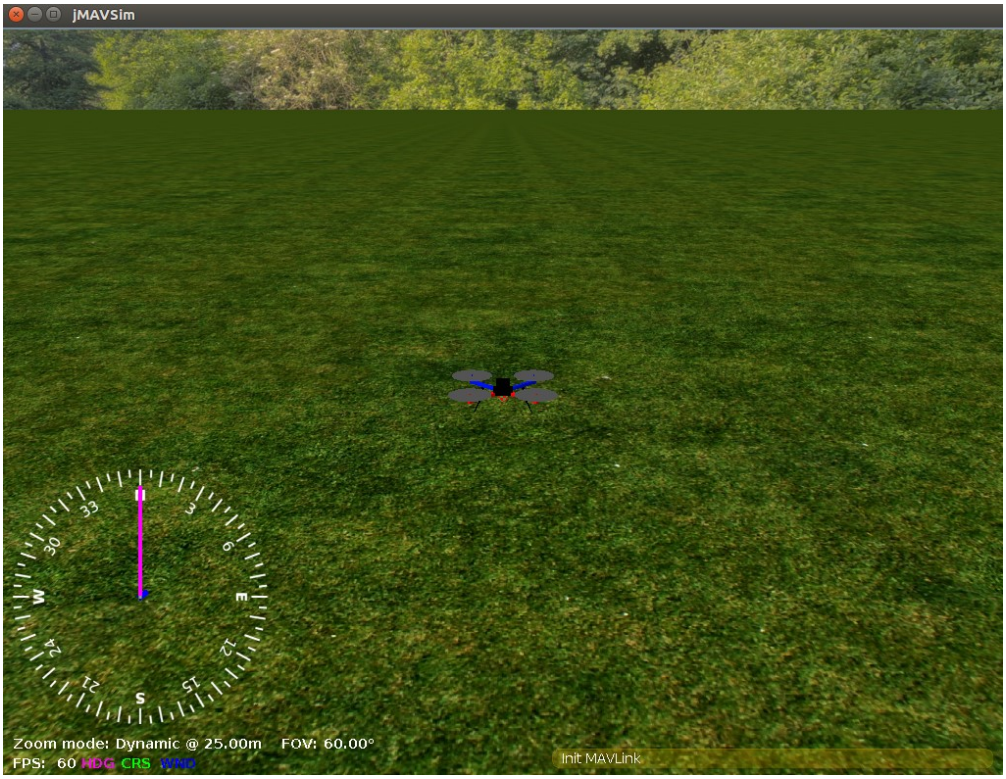


Figura 14: Simulador jMAVSim

5.8. Etapas de construcción

La arquitectura del Quadrotor ha ido evolucionando a lo largo del proyecto a la par que surgían nuevas necesidades que satisfacer. Para ello, las fases de construcción del dron han sido las siguientes:

- **Fase 1**. Una vez montada y atornillada cada parte (menos la placa superior) del chasis F450 de DJI, mostrado en la figura 7, se ha procedido a incorporar los 4 motores eléctricos E310 y la batería Zippy mencionados en los apartados anteriores. Así mismo, se han soldado los cables de la alimentación a la placa inferior de la estructura, y realizado el cableado oportuno. Los variadores se han sujetado a cada una de las patas mediante bridas, y la parte sobrante de los cables se ha enroscado alrededor de la parte interior de las patas. También se ha incorporado el dispositivo necesario para la telemetría junto a la batería. El resultado se muestra en la figura 15 (izquierda)

- **Fase 2**. Se ha atornillado la placa superior del chasis, y sobre la misma se ha unido el dispositivo Pixhawk con cinta de doble cara asegurándose de que este quede perfectamente alineado con la parte delantera del dron y quede simétricamente dispuesto. Finalmente, se han conectado los pines de la alimentación, radio, y motores en sus respectivas casillas y enroscado las hélices *z-blade* mencionadas en 5.3. En la figura 15 (derecha) se muestra el estado del Quadrotor tras esta fase.

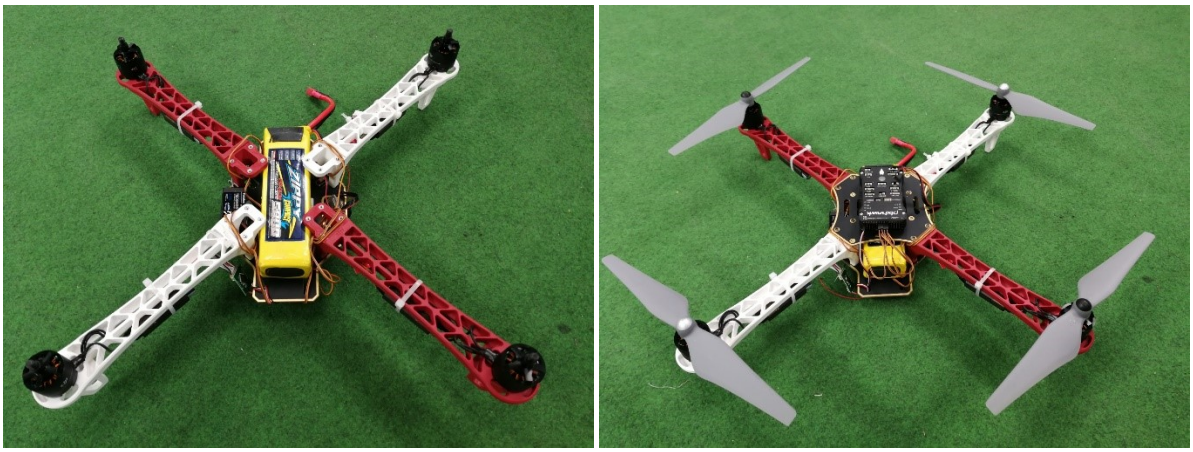


Figura 15: Fase 1 (izquierda), Fase 2 (derecha)

Tras la segunda etapa de construcción, el Quadrotor ya se encontraba perfectamente operativo y listo para volar. Sin embargo, no había espacio suficiente para ubicar toda la aparamenta electrónica necesaria para el GPS. Por ello, fue necesario incluir un tercer piso al dron.

- **Fase 3**. La idea original fue la de quitar ciertos tornillos de la placa superior y sustituirlos por tornillos de la misma métrica de M2,5 pero mucho más largos. No obstante, se necesitaba que los tornillos fuesen de unos 6 o 7 cm de largo, pero solo se contaba con tornillos de 3 cm de longitud de dicha métrica. Por ello, la solución adoptada consistió en recortar una tabla de plástico y realizarle una operación de taladrado con agujeros de la métrica adecuada para atornillarla sobre la placa superior. Sobre esta placa se dispuso el Pixhawk, como muestra la figura 16 (izquierda).

- **Fase 4**. Ya que sí que se disponía de tornillos de métrica M3 de la longitud adecuada, en las esquinas de la tabla recortada se realizó un taladrado de dicha métrica. A continuación, se fijaron con tuercas los tornillos y se incluyó el tercer piso como se muestra en la figura 16 (derecha).

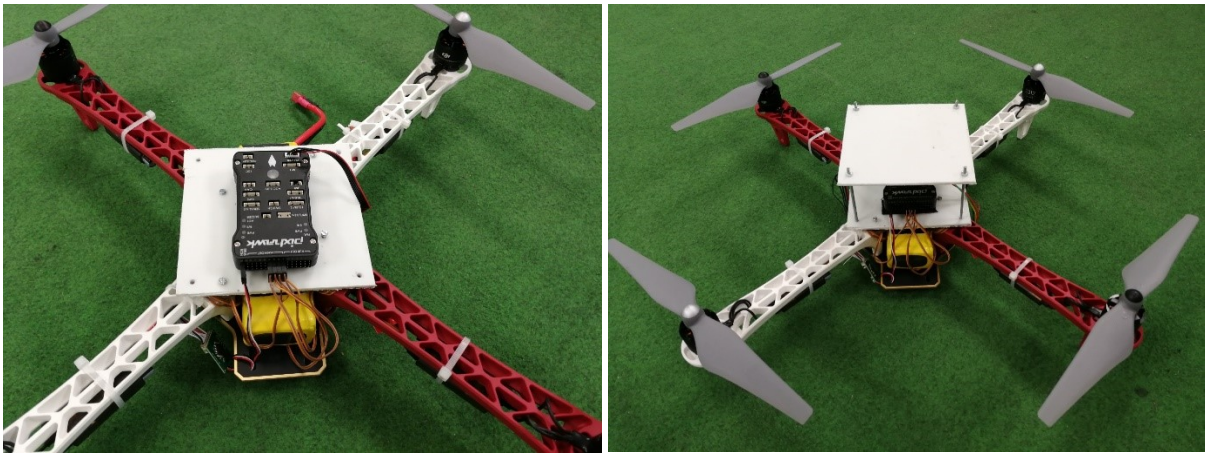


Figura 16: Fase 3 (izquierda), Fase 4 (derecha)

- **Fase 5**. Finalmente, se dispusieron en el tercer piso la placa del GPS, su antena y la antena de la radio. El resultado final se encuentra en la figura 17

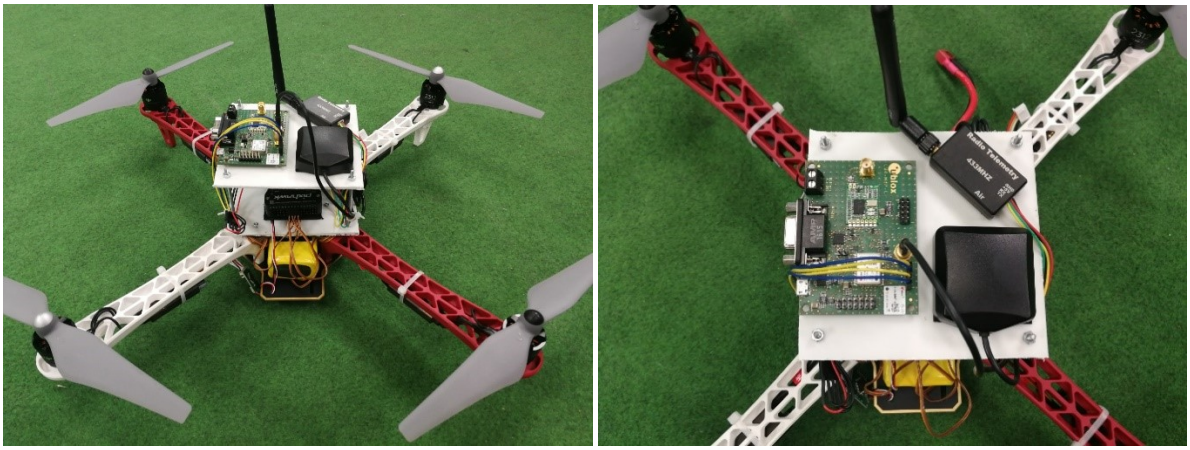


Figura 17: Fase 5

Capítulo 6

Pixhawk

En este capítulo se describe detalladamente el hardware y software de piloto automático Pixhawk empleado para la realización de este trabajo. A su vez, se describe la estrategia de control que ejecuta y los posibles modos de vuelo.

Como ya se advierte en el Capítulo 2 de este escrito, Pixhawk® es un proyecto de carácter independiente cuyo objetivo es ofrecer hardware de piloto automático industrial de gama alta. Su software de piloto automático permite ejecutar un sistema operativo en tiempo real (*Real Time Operative System*, RTOS), que proporciona un entorno de estilo POSIX. Dicho software se puede actualizar fácilmente vía USB. Es una plataforma autosuficiente por sí misma, que cuenta tanto con hardware como con software, y que es capaz de llevar a cabo diferentes aplicaciones sobre el piloto automático, facilitando siempre la interacción entre el humano y la máquina.

Una de las grandes ventajas de estos dispositivos frente a otros de carácter más comercial, y motivo por el cual ha resultado de gran interés su estudio, es que disponen de un código abierto. Esto quiere decir que ha sido posible editar el código de la propia máquina, y por tanto programar y ejecutar algoritmos de control propios desarrollados en el laboratorio, e incluirlos en el código usando una herramienta de programación. En el caso que nos ocupa, se ha decidido emplear el programa *Qt Creator* para realizar dicha tarea.

6.1. Hardware PX4

El proyecto de PX4 ha diseñado a lo largo del tiempo diferentes módulos de hardware capaces de dirigir a los vehículos en diferentes operaciones, ya sean de tipo asistido o plenamente autónomas por parte de la máquina. Cabe destacar que la mayoría de estos módulos son diseños de hardware abierto.

Entre los módulos que se encuentran disponibles hoy en día encontramos:

- Autopiloto PX4FMU (del inglés "*Flight Managemente Unit*")
- PX4IO: módulos dedicados específicamente a la plataforma capaces de emitir señales e indicaciones a los motores (llamados IO, del inglés "*Input/Output*")
- Módulo de cámara de flujo óptico

Dentro de la empresa existe una gran cantidad de modelos, cada uno con sus ventajas y carencias, según la aplicación a la que esté enfocado. En nuestro caso, se ha empleado el Pixhawk 1 para el desarrollo del proyecto por diferentes razones. La primera es que, al llevar más tiempo en el mercado, se trata de un prototipo con mayor historial de estudio sobre el mismo por lo que trabajar con él resultaría una tarea más asequible. Así mismo, el menor coste económico que ofrecía frente al resto de modelos ha sido un factor a tener en cuenta.

6.2. Pixhawk 1

Pixhawk 1 es un autopiloto de alto rendimiento adecuado para una gran variedad de aeronaves, como multi-rotores, helicópteros, automóviles, embarcaciones o, en definitiva, cualquier tipo de sistema robótico que sea capaz de moverse. Está enfocado principalmente a aplicaciones de carácter industrial y de investigación. Su principal ventaja frente a otros modelos similares es que combina los módulos PX4FMU y PX4IO mencionados en el apartado anterior. En la figura 18 se muestra una imagen del mismo.

Figura 18: Pixhawk 1

Sus características más relevantes son:

- 168 MHz / 252 MIPS Cortex-M4F
- 14 salidas PWM/servo, de las cuales 8 se pueden anular manualmente o automáticamente por fallo, y 6 auxiliares compatibles con alta potencia
- Sistema de copia de seguridad de datos para recuperación en vuelo y anulación manual. Este cuenta con procesador dedicado y alimentación independiente
- Entradas de fuente de alimentación redundantes y conmutación por error automática
- Interruptor de seguridad externo
- Indicador de tipo visual principal de tipo LED multicolor
- Tarjeta microSD para el registro de datos durante el vuelo

Dentro de estas características es necesario destacar la gran importancia que han tenido dos de ellas. Por un lado, tenemos el indicador LED principal situado justo debajo de la flecha que señala la parte delantera del dispositivo. Este LED hace uso no solo de diferentes colores, sino también de variaciones en la frecuencia de parpadeo para indicar el estado del dispositivo. Esto ha sido de gran ayuda cuando surgía algún problema, ya que permite identificar con mayor facilidad cuál es el fallo y, por tanto, permite al usuario encontrar la solución al problema más rápidamente. Por ejemplo, cuando parpadeaba en rojo intermitentemente quería decir que el dron no se podía armar, lo cual nos conducía a revisar el interruptor de seguridad; o cuando parpadeaba en amarillo, se comprobaba el estado de la batería y si era necesario se ponía a cargar. El código de colores que sigue el Pixhawk se muestra en la figura 19.

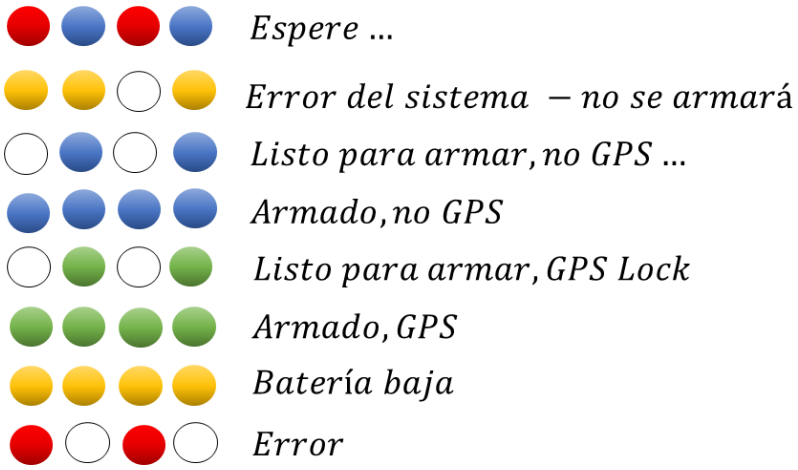


Figura 19: Código de colores del LED principal

Por otro lado, la otra propiedad que presenta el dispositivo y que ha resultado de vital importancia ha sido la ranura para la tarjeta microSD. Esto es debido a que mientras que durante las simulaciones el código diseñado sí que permitía guardar las variables de interés en ficheros, durante vuelos reales esto no era posible. Por ello, el poder disponer de una tarjeta donde escribir dichas variables fue crucial a la hora de poder avanzar con el proyecto y analizar correctamente los resultados de los diferentes algoritmos de control.

6.3. Especificaciones

A continuación, se listan las especificaciones del Pixhawk 1:

Procesador

- 32bit STM32F427 Cortex M4 core con FPU
- 168 MHz
- 256 KB RAM
- 2 MB Flash
- 32 bit STM32F103 coprocesador de seguridad frente al fallo

Sensores

- ST Micro L3GD20H 16 bit giroscopio
- ST Micro LSM303D 14 bit acelerómetro / magnetómetro
- Invensense MPU 6000 3-axis acelerómetro/giroscopio
- MEAS MS5611 barómetro

Interfaces

- 5x UART (serial ports), uno capaz de soportar alta potencia, 2x con HW control de flujo
- 2x CAN (uno con un transceptor interno de 3.3V, uno en el conector de expansión)
- Entrada Spektrum DSM / DSM2 / DSM-X® compatible con satélite
- Entrada y salida compatible con Futaba S.BUS®
- Entrada de señal de suma PPM
- Entrada RSSI (PWM o voltaje)
- I2C
- SPI
- Entradas ADC de 3.3 y 6.6V
- Puerto micro USB interno y extensión de puerto micro USB externa

Sistema de alimentación y Protección

- Controlador de diodo con anulación automática
- Servo rail de alta potencia (máximo 10V) y alta corriente (10A +)
- Todas las salidas periféricas protegidas contra sobre-corriente, todas las entradas protegidas contra ESD

Tipos de alimentación

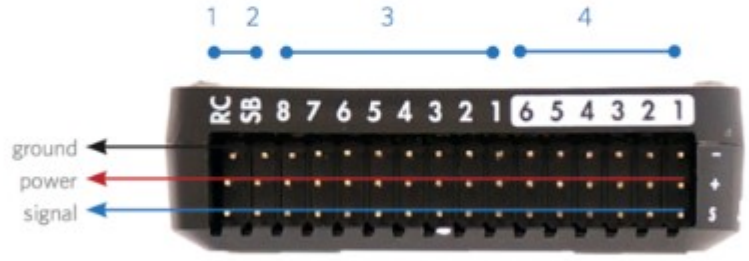
El Pixhawk puede ser alimentado de tres maneras distintas

- Entrada de alimentación (4.8 - 5.4 V)
- Entrada de servo rail (4.8 – 5.4 V) con un máximo de hasta 10 V con anulación manual
- Entrada micro USB

Conectores

1 Spektrum DSM receiver
2 Telemetry (radio telemetry)
3 Telemetry (on-screen display)
4 USB
5 SPI (serial peripheral interface) bus
6 Power module
7 Safety switch button
8 Buzzer
9 Serial
10 GPS module
11 CAN (controller area network) bus
12 I²C splitter or compass module
13 Analog to digital converter 6.6 V
14 Analog to digital converter 3.3 V
15 LED indicator

1 Input/output reset button
2 SD card
3 Flight management reset button
4 Micro-USB port



1 Radio control receiver input
2 S.Bus output
3 Main outputs
4 Auxiliary outputs

Figura 20: Conectores del Pixhawk

Periféricos

Los periféricos compatibles con el sistema Pixhawk son:

- Airspeed digital PX4AIRSPEED
- Módulo GPS ublox
- Puerto USB externo
- LED multicolor externo
- RC Futaba

6.4. Estrategia de control por defecto

6.4.1. Control de orientación del Pixhawk

El controlador de orientación funciona empleando el método de control en cascada. Existen, de este modo, dos bucles. El bucle externo calcula el error entre la referencia o ángulo deseado (a partir de ahora nos referiremos a él por su denominación en inglés "*setpoin*t") y la medida o ángulo actual. Dicho error se multiplica por una ganancia P, y el algoritmo de control genera a partir del mismo un setpoint de velocidad, es decir, la velocidad deseada en el marco del Quadrotor ${}^{b}\mathcal{O}$. El bucle externo se trata, por tanto, de un controlador de tipo proporcional. Este setpoint de velocidad pasa al bucle interno, donde se calcula el error de velocidad a partir de la medida de los motores y se actúa sobre la velocidad de dichos motores para que la máquina rote con la velocidad angular deseada. Aquí se usa un controlador proporcional-integral-derivativo, PID, para convertir el error de velocidad en comandos de alto nivel para los motores, obteniéndose así, la acción de control. El diagrama que representa el control de orientación se muestra en la figura 21.

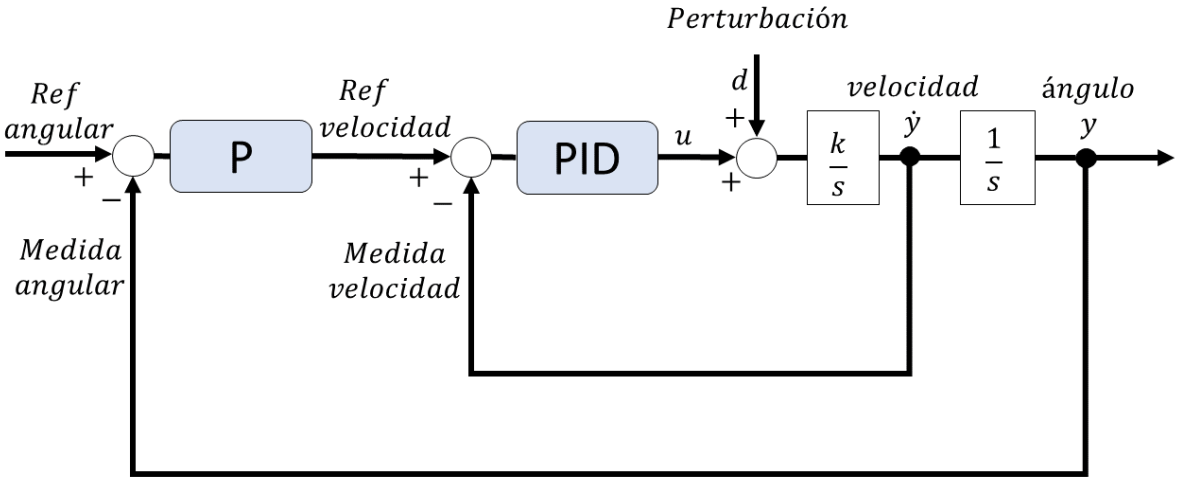


Figura 21: Control de orientación por defecto

De este modo, las ganancias del control tienen un significado intuitivo. Por ejemplo, supongamos que el dron se encuentra inclinado hacia delante 0.5 radianes (aproximadamente 28.7 grados). Si la ganancia proporcional del bucle externo del roll es de 6, el helicóptero tratará de compensar esos 0.5 radianes de desviación en altitud con una velocidad de 6 veces dicho valor, es decir, 3 radianes por segundo (o 171 grados por segundo). A continuación, si la ganancia del bucle interno del roll es de 0.1, la salida del control para el roll será de $3 \cdot 0.1 = 0.3$. Esto significa que reducirá la velocidad de los motores traseros un 30% e incrementará la de los delanteros otro 30% con el fin de inducir un momento angular y estabilizar al Quadrotor.

6.4.2. Control de posición del Pixhawk

El control de posición del Pixhawk funciona a partir de las estimaciones realizadas por un filtro de Kalman de estado extendido, también conocido como EKF2 ("*State Extended Kalman Filter*"). Este es capaz de obtener las estimaciones de diferentes estados como orientación (en quaterniones), velocidad (NED), posición (NED), velocidad del viento (NED), etc. El algoritmo de control consiste en un control en cascada con dos bucles, donde dependiendo del modo, se realiza un baipás del bucle externo. Este baipás se representa en la figura 22 como un multiplexor. Por tanto, el control de posición (cuando están activos los dos bucles) solo se utiliza en determinados casos:

- Cuando se mantiene la posición
- Cuando la velocidad solicitada en un eje es nula

En caso contrario, se activará solo el bucle interno, obteniéndose por tanto únicamente control en la velocidad.

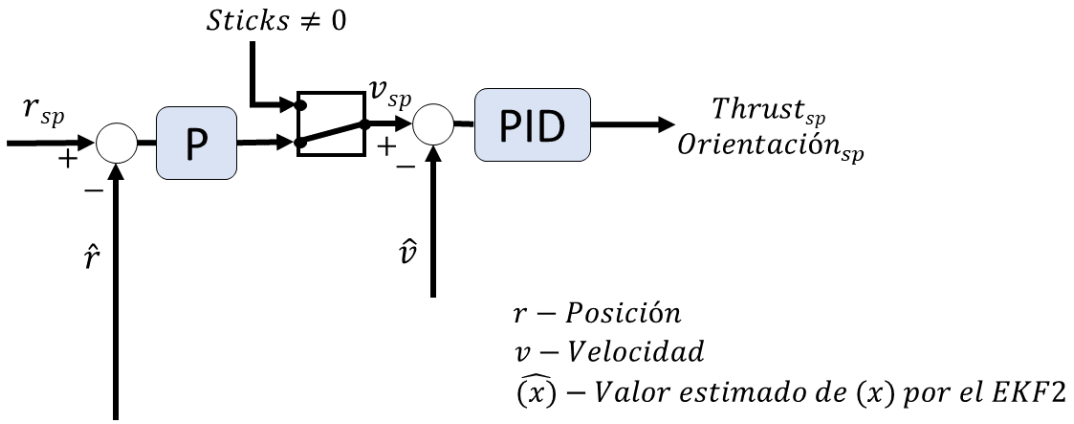


Figura 22: Control de posición por defecto

6.5. Rango de actuación de motores. Saturación

Con el fin de inclinar el dron un determinado ángulo, bajo ciertas condiciones sería posible que un motor obtuviese una entrada más alta que su velocidad máxima posible y que otro obtenga una entrada menor que cero. Si esto sucediese, las fuerzas creadas por los motores violarían el modelo de control y sería probable que el multi-rotor acabase inestabilizándose y volcándose. Para evitar esto, los mezcladores de los multi-rotores en PX4 incluyen un límite de banda o rango. Si uno de los rotores abandona esta banda de seguridad y se sitúa por encima de la misma, se opta por reducir el empuje total del sistema, de modo que se pueda satisfacer el porcentaje relativo que el controlador está emitiendo. Como resultado, el Quadrotor podría no subir o perder cierta altura, pero al menos no se inestabilizará. Sucede lo mismo cuando el rotor se sale del rango de seguridad por el límite inferior, en cuyo caso se escalará para que, aunque el comando de giro sea muy brusco, el helicóptero no vuelque al despegar cuando el empuje es muy pequeño.

6.6. Modos de vuelo

A continuación, se exponen en dos tablas los diferentes modos de vuelo, especificándose en las mismas las características que estos presentan, los sensores que necesitan para funcionar correctamente, y el color del LED principal del Pixhawk con el que se asocian.

Nota: ya que todas las plataformas y software utilizan los nombres de los modos de vuelo en inglés, aquí también se ha optado por mostrarlos de dicha manera.

6.6.1. Modos de vuelo manual

Los modos de vuelo manual son aquellos donde el usuario tiene un control directo sobre el vehículo a través del RC o joystick.

Modos	Sensores	Características
Manual/Stabilized 	-	Las entradas se transmiten como comandos de posición angular en roll y pitch, y comandos de velocidad angular en yaw. El autopiloto controla la orientación, de modo que regula los ángulos del roll y el pitch a cero cuando los sticks de la RC están centrados. Sin embargo, en este modo la posición del vehículo no está controlada, por lo que la posición puede variar debido a perturbaciones tales como el viento, etc.
Acro 	-	Las instrucciones del piloto se pasan como comandos de roll, pitch y yaw al autopiloto. El autopiloto controla la velocidad angular, pero no la orientación. Por tanto, a diferencia del control manual, si los sticks del RC están centrados, el Quadrotor no se nivelará.

		Esto permite que el vehículo se invierta por completo, permitiéndole hacer piruetas, backflips y todo tipo de acrobacias.
Rattitude 	-	Las indicaciones del piloto se transmiten al autopiloto como comandos de velocidad angular de roll, pitch y yaw si superan el umbral del modo, es decir, si los sticks de la RC están a cierta distancia de la posición central. Si no, las entradas se pasan como comandos de posición angular en roll y pitch y comando de velocidad angular en el yaw. De esta forma, el autopiloto actúa como controlador de velocidad angular cuando los sticks se encuentran en posiciones alejadas del centro, mientras que cuando están centrados, el autopiloto ejerce como controlador de orientación.

Tabla 1: Modos de vuelo manual

6.6.2. Modos de vuelo asistido

Los modos de vuelo asistido son, como su nombre indica, modos donde el Quadrotor es controlado por el usuario, pero al mismo tiempo ofrecen cierto nivel de asistencia a la hora de estabilizar el dron frente a posibles perturbaciones como el viento, etc.

Modos	Sensores	Características
Position 	GPS + barómetro/Sensor de flujo óptico	El roll controla la velocidad en dirección derecha-izquierda; el pitch controla la velocidad en dirección delante-atrás; y el yaw controla la velocidad angular de guiñada como en el modo manual. El throttle (acelerador) controla la subida-bajada del dron como en el modo altitude. Esto significa que la posición del Quadrotor permanece estable por el autopiloto frente a cualquier tipo de perturbación cuando los sticks están centrados.
Altitude 	Sensores capaces de medir la altura o altitud, como por ejemplo un barómetro	Las entradas de roll, pitch y yaw se tratan como en el modo Manual/Stabilized. El stick del throttle tiene una gran zona muerta, pues el dron no empieza a despegar hasta que se supera la mitad del recorrido total del stick. Cuando está centrado, se mantiene la altitud. El autopiloto solo controla la altitud, por lo que la posición del Qaudrotor puede variar ante la presencia de perturbaciones.

Tabla 2: Modos de vuelo asistido

Capítulo 7

Desarrollo y resultados de las leyes de control de orientación implementadas

En esta sección se presentan y argumentan los diferentes algoritmos de control de orientación desarrollados para el trabajo, así como los resultados obtenidos a partir de los mismos tanto en simulación como en vuelos reales. Este capítulo hace referencia, por tanto, a todas las estrategias de control implementadas para el control de orientación del Pixhawk.

Los reguladores tipo PID han sido durante estos últimos años los que mayor impacto han tenido en la industria. Estos, se componen de 3 factores básicos de control: proporcional, integral y derivativo (P + I + D). Sin embargo, es bien sabido que no siempre terminan de adaptarse correctamente a algunos procesos, especialmente a los más complejos.

Si se desea error nulo en régimen permanente, la acción integral "I" es necesaria por dos razones: la inherente presencia de incertidumbres de modelado; y la posible existencia de perturbaciones externas. Pero, es conocido que una excesiva acción integral, puede introducir oscilaciones e incluso inestabilizar el sistema, máxime en un sistema inestable. Notar que, la acción integral se consigue añadiendo un polo en el origen, es decir, en el límite de la zona estable, por lo que es lógico pensar que ese nuevo elemento puede ser una nueva fuente de inestabilidad. Es, por tanto, por este mismo motivo por el que uno de los objetivos de este TFG se ha centrado en desarrollar nuevas estrategias de control más complejas capaces de solventar las desventajas que supone incluir un integrador a la acción de control, pero manteniendo su principal ventaja: error nulo en régimen permanente. De este modo, se procede a explicar todas las estrategias de control implementadas, así como las fortalezas y carencias que cada una de ellas presenta.

Cabe mencionar que el presente capítulo se estructura del siguiente modo: cada algoritmo de control trabajado se describirá siguiendo la misma estructura de subapartados: descripción teórica de la estrategia de control, implementación de esta en lenguaje de programación, resultados obtenidos en simulación y, finalmente, resultados obtenidos en vuelo real.

7.1. Controlador PD

Como se indica en el Capítulo 6, apartado "6.4.1. Control de orientación del Pixhawk", el control de orientación por defecto consiste en dos controladores dispuestos en cascada: un controlador P en el bucle externo y un controlador PID en el bucle interno.

De esta forma, únicamente a modo de toma de contacto con el algoritmo PX4, inicialmente se optó por desarrollar un controlador PD en el bucle interno que sustituyese al PID original, dejando intacto el bucle externo. Esto se hizo así para asegurarse de que se estaba controlando las variables del bucle interno de referencia y medida de velocidad del dron, y que no se había malinterpretado el funcionamiento del código original. Atendiendo al modelo teórico de doble integrador deducido en el Capítulo 4, el diagrama que representa esta estrategia de control se muestra en la figura 23.

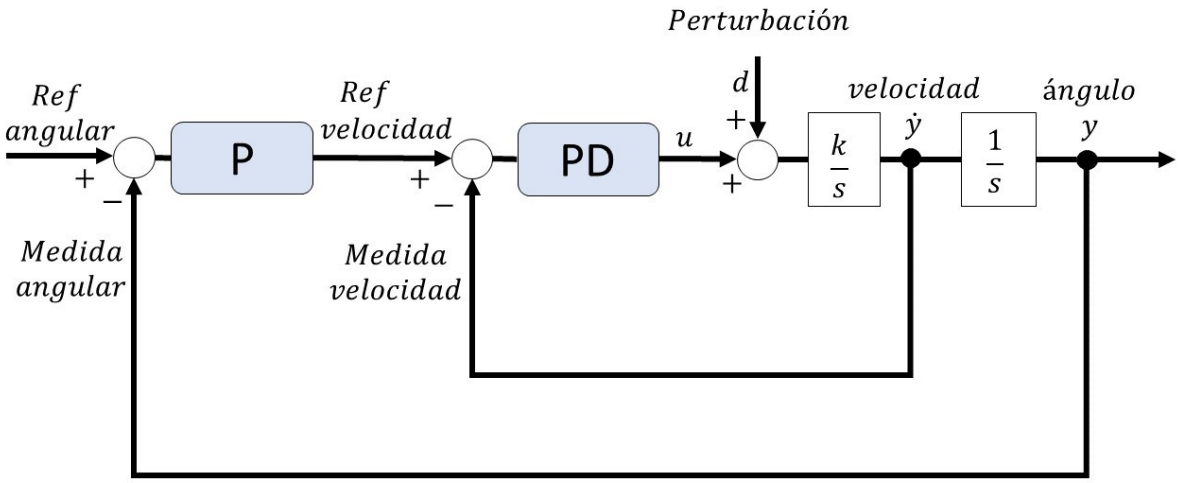


Figura 23: Controlador PD implementado en el bucle interno

Una vez dominado por completo el bucle interno del controlador, el siguiente paso fue el de ejercer un control total sobre el dron, y no solo sobre la velocidad. De este modo, el segundo PD que se desarrolló suplantaba al algoritmo en cascada por defecto, permitiéndonos realizar finalmente un control sobre la posición angular en el marco del Quadrotor ${}^{b}\mathcal{O}$. Así pues, en este apartado se describe el proceso que se siguió para desarrollar esta estrategia de control y las conclusiones que se obtuvieron de la misma.

7.1.1. Concepto de controlador PD

Un controlador PD es, como su nombre indica, aquel que posee tanto acción proporcional como acción derivada al error. La acción proporcional es instantánea, depende del presente. Esta consiste en una constante que denominaremos Kp, la cual puede, con el valor adecuado, mejorar el comportamiento en el régimen transitorio y permanente, aunque con la posibilidad de introducir oscilaciones. Se trata,

en definitiva, de una amplificación del error. Por otro lado, tenemos la acción derivada, que trata de predecir el error futuro. Es muy sensible al ruido y en ciertas ocasiones, como con los sistemas de primer orden, se recomienda no usarla. Sin embargo, con su parámetro característico Td se puede mejorar el comportamiento del sistema durante el transitorio, haciendo posible eliminar las oscilaciones introducidas por la acción proporcional.

7.1.2. Implementación del PD como sustituto del control en cascada original

Sabiendo que como se estudia en la asignatura de TAU, las ecuaciones que describen a un controlador PD son:

$$u(t) = Kp \cdot err(t) + KpTd\frac{derr(t)}{dt} \tag{7.1}$$

$$u(t) = P(t) + D(t) \tag{7.2}$$

Siendo $\mathrm{u(t)}$ la acción de control, $\mathrm{P(t)}$ y $\mathrm{D(t)}$ las acciones proporcional y derivada respectivamente, y Kp y Td los parámetros característicos del PD.

Podemos discretizar (7.1) y (7.2) del siguiente modo:

$$P(k) = Kp \cdot err(k) \tag{7.3}$$

$$D(k) = -Kd \cdot \dot{y} \tag{7.4}$$

$$u(k) = P(k) + D(k) \tag{7.5}$$

Donde Kp y Kd son las constantes proporcional y derivada, e $\dot{y}$ la velocidad en la dirección del ángulo a controlar.

La representación que describe el comportamiento de dichas ecuaciones se muestra en la figura 24.

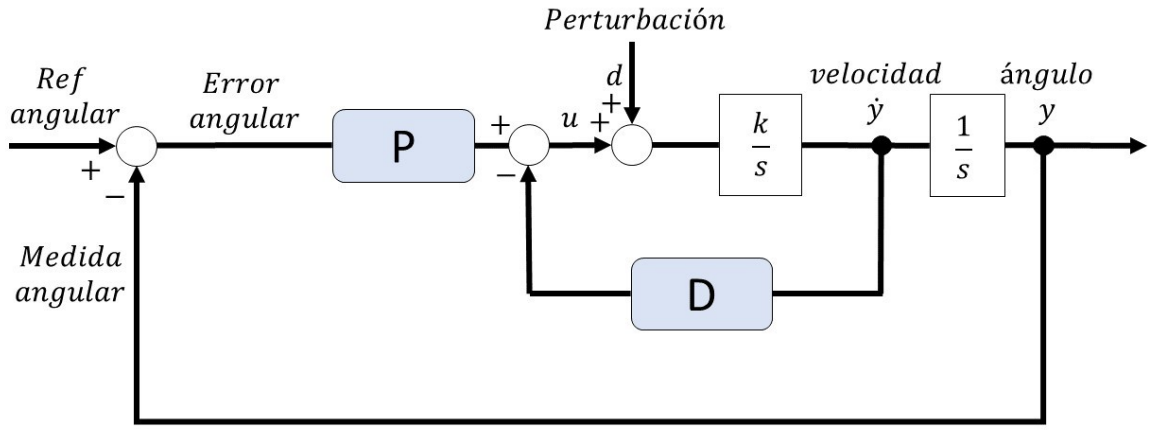


Figura 24: Control de orientación con PD

Con el fin de obtener una estrategia de control como la que se muestra en las ecuaciones (7.3), (7.4) y (7.5), el algoritmo a desarrollar ha sido el siguiente (se recuerda que, para observar mejor los resultados aislados de nuestro control, únicamente se ha realizado control sobre un ángulo, en este caso el pitch):

1.Inicializar variables, abrir fichero de escritura

2.Comenzar bucle de control hasta desarme del Quadrotor:

3.Obtención de las variables necesarias para el control del pitch:

3.1. Lectura de los sensores de la IMU y la referencia por radio // de aquí se obtiene en forma de quaterniones el setpoint y la medida

3.2. Obtener en forma de vector de ángulos de Euler el setpoint y la medida a partir de los quaterniones

3.3. Obtener la derivada del ángulo del pitch a partir de la medida de velocidad de los sensores

4.Ley de control sobre el pitch: //cálculo de la acción de control a aplicar en el instante de muestreo

4.1. Cálculo de la acción proporcional

4.2. Cálculo de la acción derivada

4.3. Cálculo de la acción de control como la suma de la proporcional más la derivada

5.Saturación de la acción de control:

5.1. Si la acción de control es mayor que 0.5, esta se iguala a 0.5

5.2. Si la acción de control es menor que -0.5, esta se iguala a -0.5

6.Se envía la acción de control a los motores en forma de tanto por 1//aplicación de la acción de control

7.Se guarda en fichero el valor de las variables de interés//esto se utiliza para la simulación

8.Se escribe en la tarjeta microSD el valor de las variables de interés//esto se utiliza para vuelos reales

9.Desarme del Quadrotor

7.1.3. Resultados obtenidos en simulación

A partir del simulador *jMAVSim* descrito en 5.7.3. se han realizado dos simulaciones:

La primera de ellas consiste en un vuelo sin perturbaciones más allá de las que el propio simulador incorpora por ligeros efectos del viento, los cuales consideramos despreciables. El resultado se muestra en la figura 25. Habiendo ajustando adecuadamente las ganancias a partir del panel de parámetros de *QGroundControl*, se observa claramente cómo, en un entorno simulado y ante ausencia de perturbaciones, el PD cumple con su función.

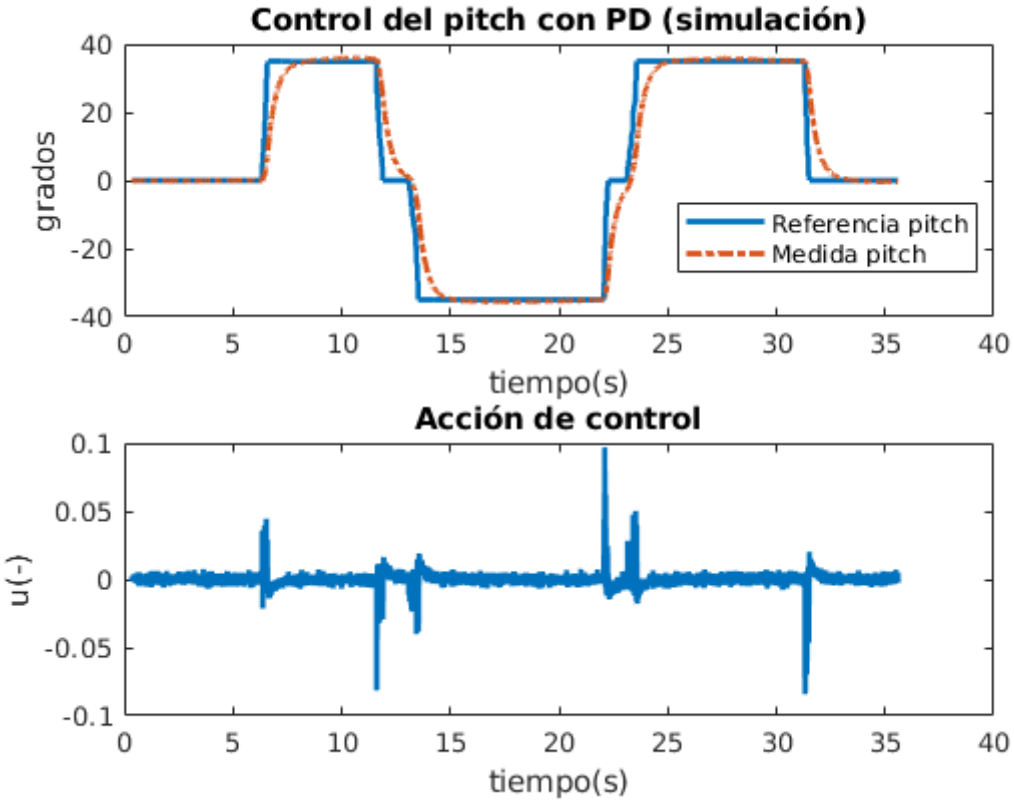


Figura 25: Control del pitch con PD (simulación)

Sin embargo, en la segunda simulación el PD empieza a fallar a la hora de seguir a la referencia. En este segundo caso, se le ha introducido una perturbación constante de 0.05 directamente a la acción de control. El resultado se muestra en la figura 26, la cual demuestra cómo, en presencia de perturbaciones existe siempre un error entre la referencia y la medida que el controlador no es capaz de solventar.

Cabe mencionar que el Quadrotor no despega hasta el segundo 5, por lo que el control no empieza a actuar hasta ese instante. Por ello se debe que en los primeros 5 segundos la acción de control se encuentre en -0.05, que es precisamente, el valor de la perturbación introducida por código.

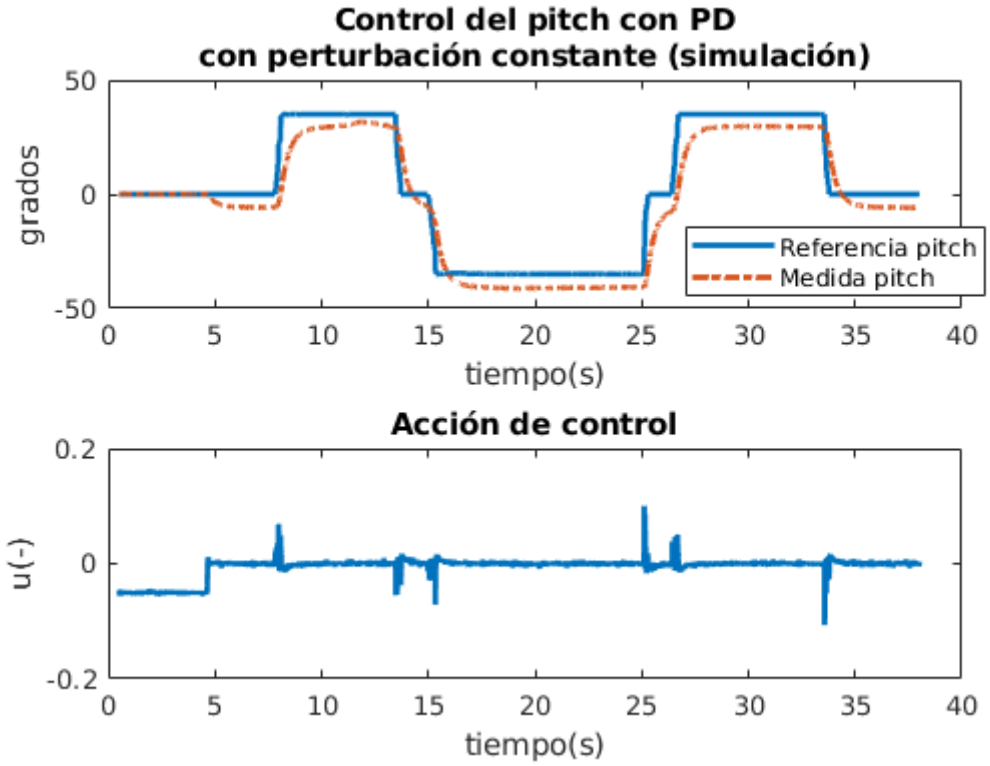


Figura 26: Control del pitch con PD con perturbación constante (simulación)

7.1.4. Resultados obtenidos en vuelo real

Al igual que en simulación, sendos experimentos se han vuelto a realizar en vuelos reales. Los resultados obtenidos son, como cabría esperar, similares a los obtenidos en simulación, aunque con un desempeño claramente peor por parte del controlador, debido a todas las perturbaciones a las que se ve sometido el Quadrotor. Sin considerar la perturbación de 0.05 introducida voluntariamente por código, el resto de las perturbaciones pueden ser debidas a desequilibrios en la distribución del peso por la estructura, o al conocido efecto suelo, donde el propio viento que generan las hélices cuando están cerca del suelo rebota y ocasiona turbulencias. Los resultados de ambos experimentos, sin y con perturbación introducida voluntariamente, se muestran en las figuras 27 y 28 respectivamente.

En el primer vuelo sin perturbación, vemos como el PD sigue con mayor o menor error la referencia. Sin embargo, en presencia de una perturbación constante de valor 0.05 se vuelve a demostrar cómo, el controlador PD, es incapaz de rechazar dicha perturbación, quedándose siempre la medida sin alcanzar la referencia deseada.

Queda pues alcanzado el objetivo del desarrollo del controlador PD: demostrar la imposibilidad del PD a la hora de rechazar una perturbación, y surgiendo por tanto de aquí, la necesidad de incluir una acción integral, lo cual nos lleva directamente al siguiente punto.

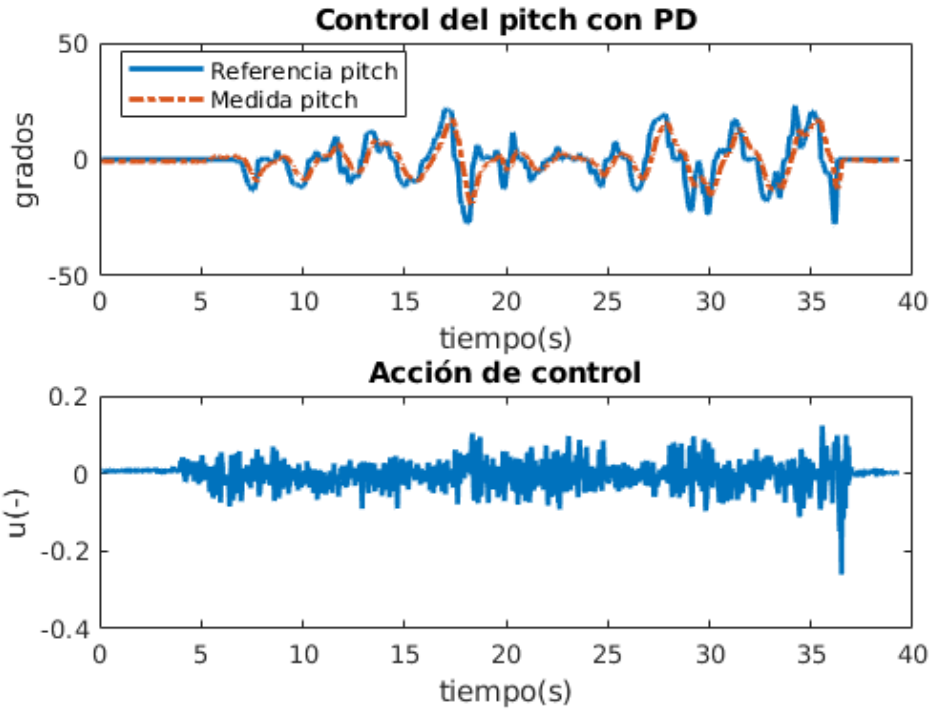


Figura 27: Control del pitch con PD (vuelo real)

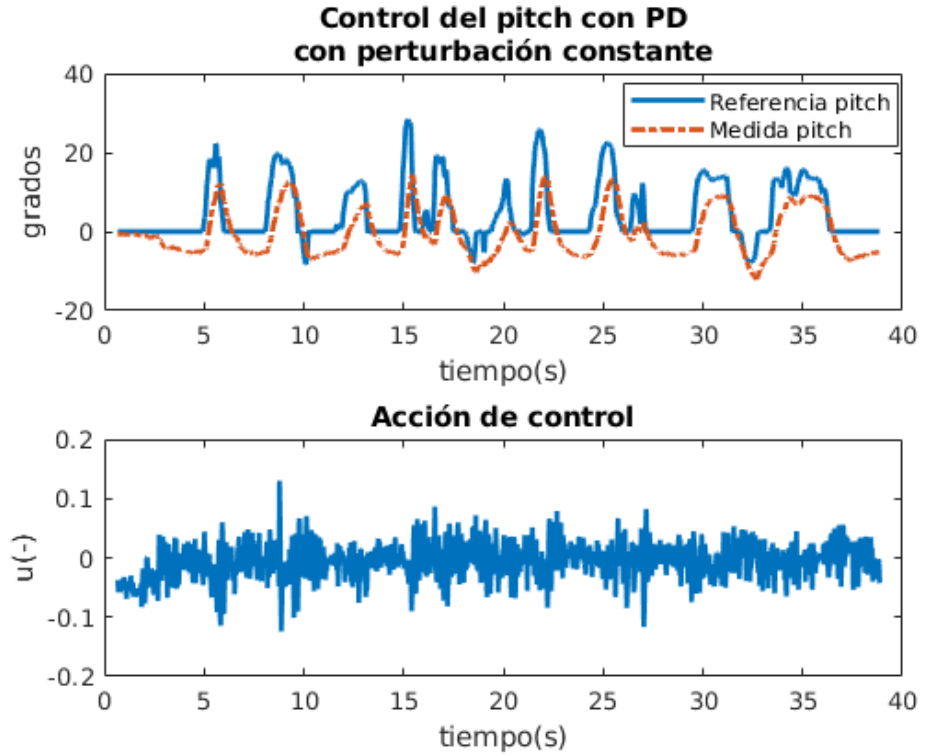


Figura 28: Control del pitch con PD con perturbación constante (vuelo real)

7.2. Controlador PID

Como se acaba de advertir, el controlador PD es incapaz de funcionar correctamente en presencia de perturbaciones. Es por ello por lo que, si se desea error nulo en régimen permanente, resulta inevitable tener que incluir acción integral (al menos por el momento).

7.2.1. Concepto de PID

El PID es el mecanismo de control por realimentación más utilizado en todo el sector industrial. Al igual que el PD, cuenta con acción proporcional y derivada, ambas desempeñando las funciones anteriormente descritas. Por tanto, la diferencia radica en la acción integral, la cual depende del estado anterior, y permite conseguir un error nulo en régimen permanente. Ésta dependerá de un parámetro Ti, el cual ajustaremos con libertad con el objetivo de conseguir un tiempo de establecimiento razonable sin introducir demasiada sobre-oscilación.

7.2.2. Implementación del PID

Del mismo modo que se procedía con el PD, se sabe que las ecuaciones que describen al PID son:

$$u(t) = Kp \cdot err(t) + \frac{Kp}{Ti}\int_0^t err(\theta)d\theta + KpTd\frac{derr(t)}{dt} \tag{7.6}$$

$$u(t) = P(t) + I(t) + D(t) \tag{7.7}$$

Cuya forma discretizada se muestra en las ecuaciones siguientes:

$$P(k) = Kp \cdot err(k) \tag{7.8}$$

$$D(k) = -Kd \cdot \dot{y} \tag{7.9}$$

$$I(k) = I(k-1) + Ki \cdot err(k-1) \tag{7.10}$$

$$u(k) = P(k) + I(k) + D(k) \tag{7.11}$$

A continuación, se muestra la representación de la estrategia de control a implementar:

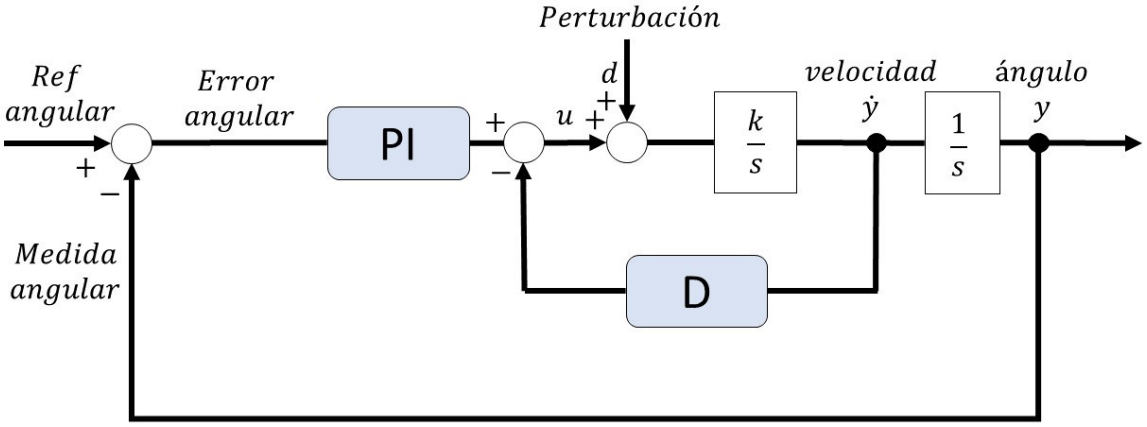


Figura 29: Control de orientación con PID

Su implementación en código procede del siguiente modo:

1.Inicializar variables, abrir fichero de escritura

2.Comenzar bucle de control hasta desarme del Quadrotor:

3.Obtención de las variables necesarias para el control del pitch:

3.1. Lectura de los sensores de la IMU y la referencia por radio // de aquí se obtiene en forma de quaterniones el setpoint y la medida

3.2. Obtener en forma de vector de ángulos de Euler el setpoint y la medida a partir de los quaterniones

3.3. Obtener la derivada del ángulo del pitch a partir de la medida de velocidad de los sensores

4.Ley de control sobre el pitch: //cálculo de la acción de control a aplicar en el instante de muestreo

4.1. Cálculo de la acción proporcional

4.2. Cálculo de la acción derivada

4.3. Cálculo de la acción integral

4.4. Cálculo de la acción de control como la suma de la proporcional, derivada e integral

5.Saturación de la acción de control:

5.1. Si la acción de control es mayor que 0.5, esta se iguala a 0.5

5.2. Si la acción de control es menor que -0.5, esta se iguala a -0.5

6.Se envía la acción de control a los motores en forma de tanto por 1//aplicación de la acción de control

7.Actualización de variables para la siguiente iteración

7.1. Acción integral anterior se iguala a la actual

7.2. Error anterior se iguala al actual

8.Se guarda en fichero el valor de las variables de interés//esto se utiliza para la simulación

9.Se escribe en la tarjeta microSD el valor de las variables de interés//esto se utiliza para vuelos reales

10.Desarme del Quadrotor

7.2.3. Resultados obtenidos en simulación

En la figura 30 se muestra el resultado que se obtiene del PID al aplicarle la misma simulación con perturbación que se le había aplicado anteriormente al PD. En este caso, se observa perfectamente como el PID es capaz de rechazar la perturbación por completo en un tiempo de establecimiento de aproximadamente 2 segundos. Sin embargo, también podemos advertir cómo, al incorporar la acción integral, aparece una sobre-oscilación que en el PD no existía. Si se pretende disminuir el tiempo de establecimiento aumentando el valor de Ki, la sobre-oscilación aumenta, llegando incluso a inestabilizar el sistema. Por tanto, hay que hallar un valor de compromiso para Ki, lo cual se ha conseguido en nuestro caso para un valor de Ki igual a 0.003. He aquí el problema que se planteaba en la introducción de este capítulo: al añadir acción integral estamos añadiendo un polo en el origen, lo cual está al límite de la estabilidad, dando lugar a sobre-oscilaciones no deseadas que pueden causar la inestabilización del Quadrotor.

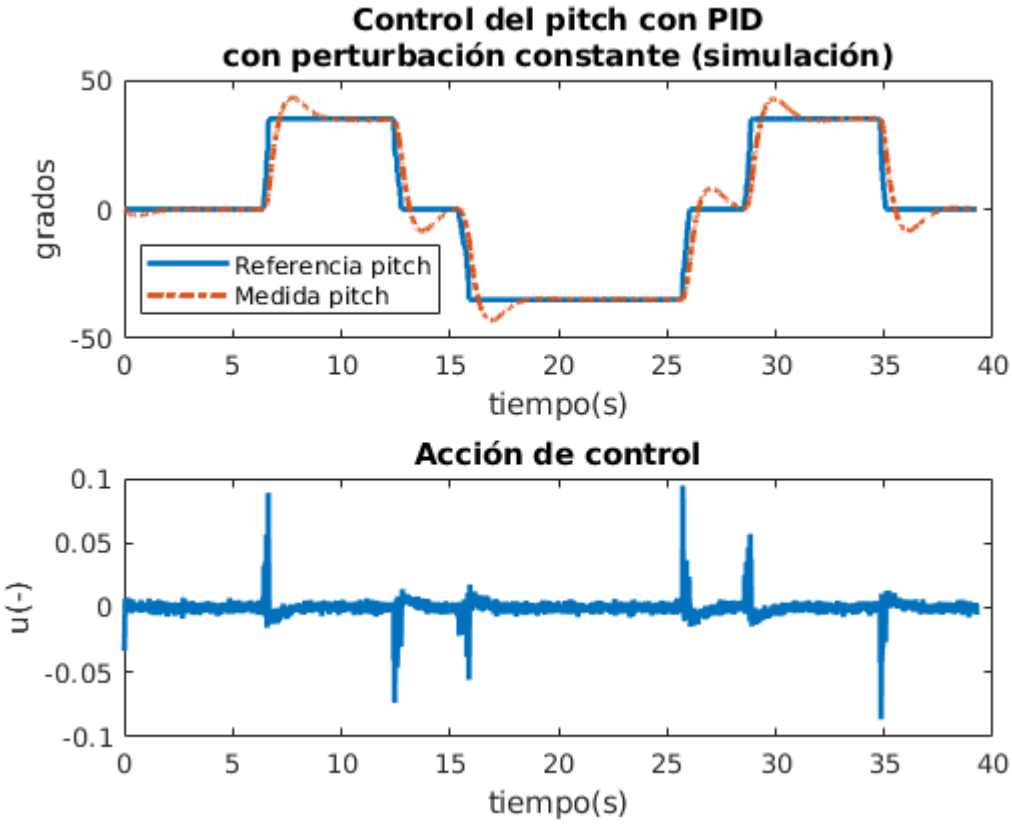


Figura 30: Control del pitch con PID con perturbación constante (simulación)

7.2.4. Resultados obtenidos en vuelo real

A continuación, se presentan las gráficas obtenidas de los vuelos reales, siendo la figura 31 el control del PID, y la 32 la acción integral. Como se puede apreciar, el PID trata de anular el efecto de la perturbación, llegando a conseguir un error nulo entre la referencia y la medida, aunque con cierto retraso. En la gráfica de la acción integral se aprecia perfectamente el caso de las sobre-oscilaciones que anteriormente se mencionaba: para una perturbación constante establecida en -0.05, el integrador trata de anularla sumándole a la acción de control 0.05. Sin embargo, vemos como este valor oscila demasiado, llegando incluso a doblar en ciertos instantes el valor de la perturbación que debería anular y añadiéndole, por tanto, sobre-oscilaciones al sistema.

He aquí el problema que se planteaba en la introducción de este capítulo: al añadir acción integral estamos añadiendo un polo en el origen, lo cual está al límite de la estabilidad, dando lugar a sobre-oscilaciones no deseadas que pueden causar la inestabilización del Quadrotor. Se presenta aquí, por tanto, la oportunidad de tratar de hallar una estrategia de control capaz de anular las diferentes perturbaciones a las que se vea sometido el sistema al igual que lo hace la acción integral, pero sin introducir las problemáticas sobre-oscilaciones ya mencionadas. Esto se verá a continuación con el UDE y el ESO.

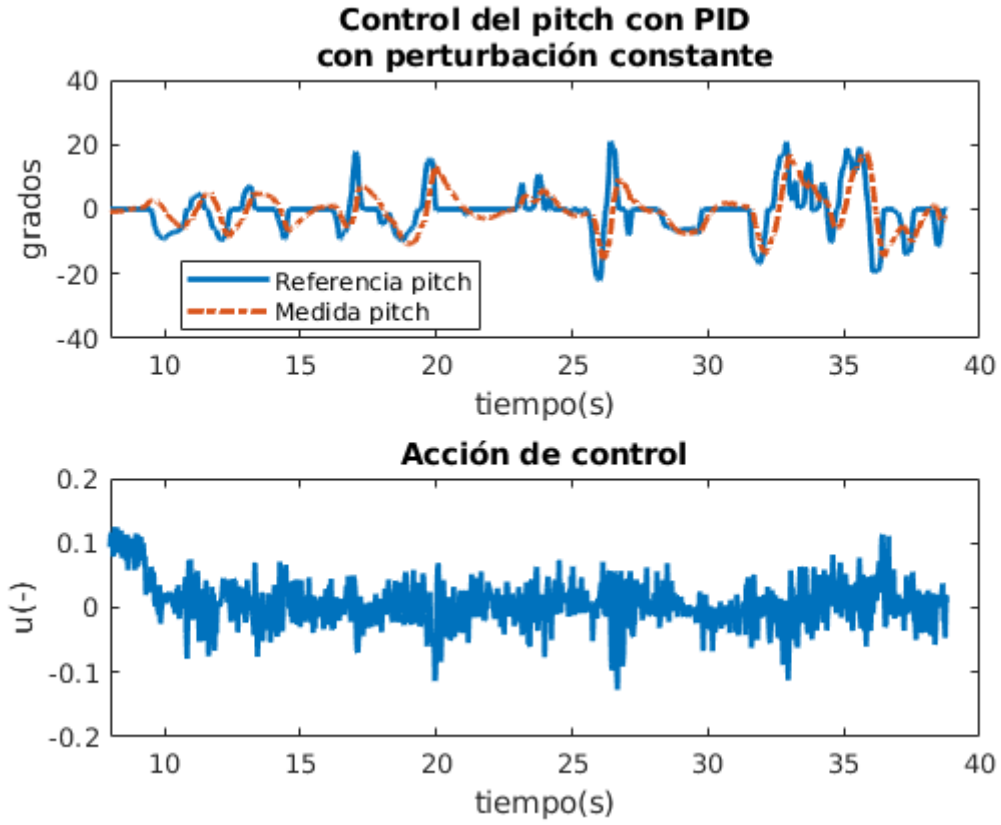


Figura 31: Control del pitch con PID con perturbación constante (vuelo real)

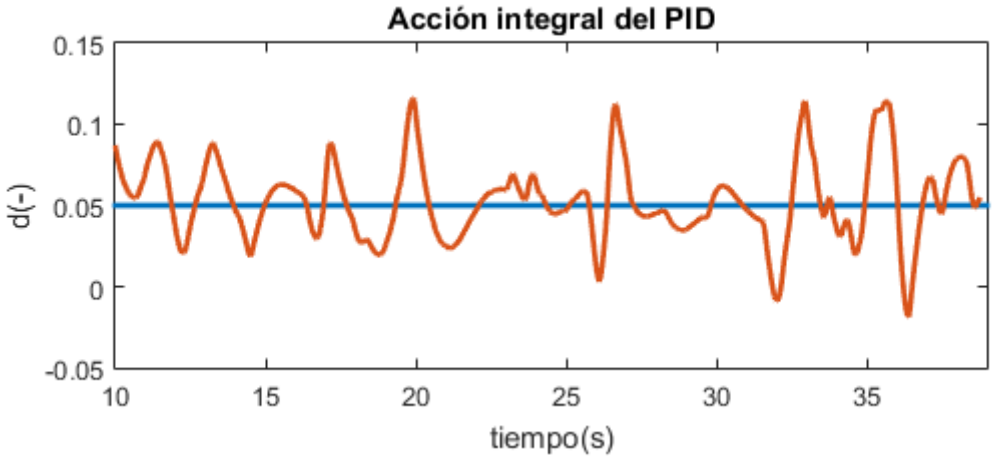


Figura 32: Estimación de la perturbación con PID

7.3. Controlador PD + UDE

7.3.1. Conceptos previos. Diseño en espacio de estados y realimentación del estado

El objetivo de técnicas de control tradicionales como el diseño en el lugar de las raíces es ubicar el polo dominante con tal de controlar el proceso de manera estable y con un tiempo de establecimiento determinado. Sin embargo, es bien sabido que en el lugar de las raíces existen infinidad de criterios de diseño, todos ellos más o menos aproximados, con relativa dificultad a la hora de ajustar los parámetros del controlador y situar todos los polos en el lugar deseado. Es por estas limitaciones y otros aspectos que aquí no se consideran, el motivo por el cual se desarrollaron otras técnicas de control, derivadas de lo que se define como modelos en espacio de estados [15]. La representación de espacio de estados es, en definitiva, un modelo matemático que representa el comportamiento de un sistema físico a partir de un conjunto de entradas, salidas y variables de estado. Estas se encuentran relacionadas entre sí por ecuaciones diferenciales, las cuales se combinan finalmente en una ecuación diferencial matricial de primer orden. Una representación general de un sistema lineal en espacio de estados adopta la siguiente forma:

$$\dot{x}(t) = A \cdot x(t) + B \cdot u(t) \tag{7.12}$$
$$y(t) = C \cdot x(t) + D \cdot u(t)$$

Donde A, B, C y D son matrices cuyas dimensiones dependen del número de entradas, salidas y variables de estado. El diagrama que representa el comportamiento de dichas ecuaciones se muestra en la figura 33:

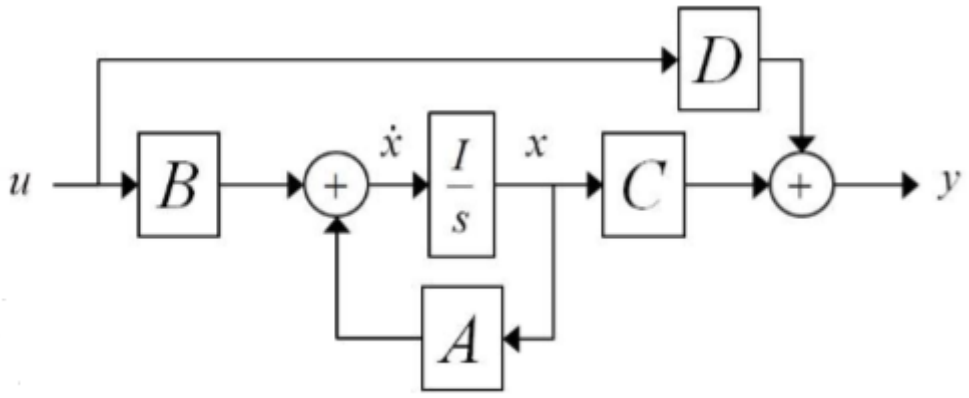


Figura 33: Modelo en espacio de estados

Una de las herramientas básicas que deriva del diseño en espacio de estados es el control por realimentación del estado. Este consiste en el diseño de una matriz de realimentación tal que, dado un modelo en espacio de estados como el anterior, con un control por realimentación del estado de la forma:

$$u(t) = -K \cdot x(t)$$

Se obtiene:

$$\dot{x}(t) = (A - BK) \cdot x(t) \tag{7.13}$$

El diseño de la matriz K se puede realizar de varias formas, la más intuitiva a priori es por asignación de polos. Una manera más ilustrativa de entender el concepto se muestra en la figura 34.

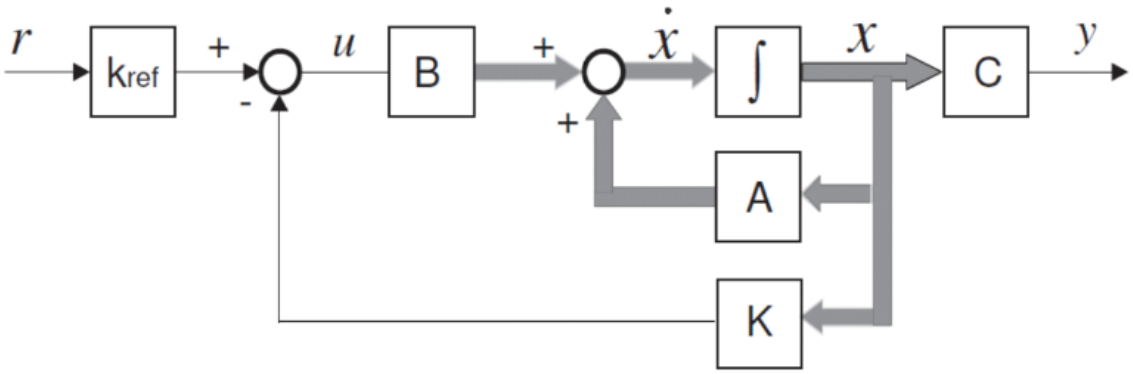


Figura 34: Realimentación del estado

Cabe destacar que, a diferencia de un diseño en el lugar de las raíces, con una realimentación del estado es posible asignar todos los polos del bucle cerrado. Sin embargo, esto implica que solo se está modificando la dinámica del sistema en bucle cerrado, pero no se está teniendo en cuanta el seguimiento de una referencia ni el rechazo de perturbaciones. Es aquí donde entrará en juego el UDE, el cual se explica a continuación.

7.3.2. Concepto de UDE

El UDE (del inglés, "*Uncertainity and Disturbance Estimator*") es una técnica de control avanzado capaz de estimar las perturbaciones a partir de la predicción del estado siguiente. Es decir, el UDE se encargará de rechazar las perturbaciones, considerando como tales todo aquello que difiera del modelo real. Por tanto, para que el UDE desempeñe su función correctamente será necesario conocer el modelo del sistema y, en definitiva, todos los estados. Esta es precisamente la característica que lo diferencia del controlador ESO explicado más adelante, pues como se verá, para la implementación del ESO no será necesario conocer todos los estados del sistema. Las ecuaciones que aquí se destacan se encuentran desarrolladas con mayor profundidad en [16].

Partiendo del modelo teórico simplificado del Quadrotor para el pitch obtenido en la ecuación (4.21) del Capítulo 4:

$$\theta = \frac{K_\theta}{s^2} \cdot u + \frac{1}{s^2} \cdot d$$

Sabemos que el modelo aproximado del Quadrotor se corresponde con el de un doble integrador. Tomando como ejemplo únicamente las ecuaciones del pitch, si queremos expresarlo de forma matricial en espacio de estados, queda del siguiente modo:

$$\dot{x}(t) = A \cdot x(t) + B \cdot u(t) + B \cdot d(t) \tag{7.14}$$
$$y(t) = C \cdot x(t)$$

Siendo:

$$x(t) = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$
$$y(t) = \begin{pmatrix} \theta \\ \dot{\theta} \end{pmatrix}$$
$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$$
$$B = \begin{bmatrix} 0 \\ K_\theta \end{bmatrix}$$
$$C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Donde K_θ es la constante del modelo, asociada al pitch. Puesto que no se ha seguido un proceso de identificación del modelo, el valor de K_θ se ha obtenido a partir de vuelos experimentales. Para ello, se ha observado el comportamiento del Quadrotor para diferentes valores de K_θ, y se ha concluido finalmente que su valor debía estar comprendido entre 80 y 120. Por tanto, se ha decidido darle un valor de 100, el cual a la vista de los resultados se considera bastante próximo al valor real.

De ahora en adelante se omitirá el término *(t)* con el objetivo de simplificar la escritura de las ecuaciones. Cabe destacar que de la ecuación (7.14) todo es conocido salvo la perturbación. Por tanto, para estimarla primero la despejaremos. Para ello, se hace uso de la matriz pseudoinversa de B, donde:

$$B^+ = (B^T \cdot B)^{-1} \cdot B^T = \begin{bmatrix} 0 & \frac{1}{K_\theta} \end{bmatrix}$$

Obteniendo así:

$$\hat{d}_\theta = B^+ \cdot \dot{x} - (B^+ \cdot A)x - (B^+ \cdot B)u \tag{7.15}$$

De la ecuación (7.15) anterior podemos realizar algunas simplificaciones:

$$B^+ \cdot A = [0 \quad K_\theta] \cdot \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$
$$B^+ \cdot B = 1$$
$$B^+ \cdot \dot{x} = \begin{bmatrix} 0 & \frac{1}{K_\theta} \end{bmatrix} \cdot \begin{pmatrix} \dot{x_1} \\ \dot{x_2} \end{pmatrix} = \frac{1}{K_\theta} \cdot \dot{x_2}$$

Por lo que el término que acompaña a x se descarta, y la ecuación queda del siguiente modo:

$$\hat{d}_\theta = \frac{1}{K_\theta} \cdot \dot{x_2} - u \tag{7.16}$$

Si resolvemos la ecuación (7.16) aplicando Laplace y condiciones iniciales nulas, se llega a un modelo que no se puede implementar, dado que el grado del numerador es superior al del denominador. Por ello, surge la necesidad de incluir un filtro paso bajo con ganancia unitaria en régimen permanente, al cual se le ha asignado una constante de tiempo de 0.1.

Finalmente, la ecuación que se obtiene al incluir el filtro y desarrollarla es la siguiente:

$$\hat{d}_\theta = \frac{1}{K_\theta} \cdot \frac{s}{0.1s+1} x_2 - \frac{1}{0.1s+1} u \tag{7.17}$$

7.3.3. Implementación del UDE

El diagrama que muestra el funcionamiento del PD+UDE es el siguiente:

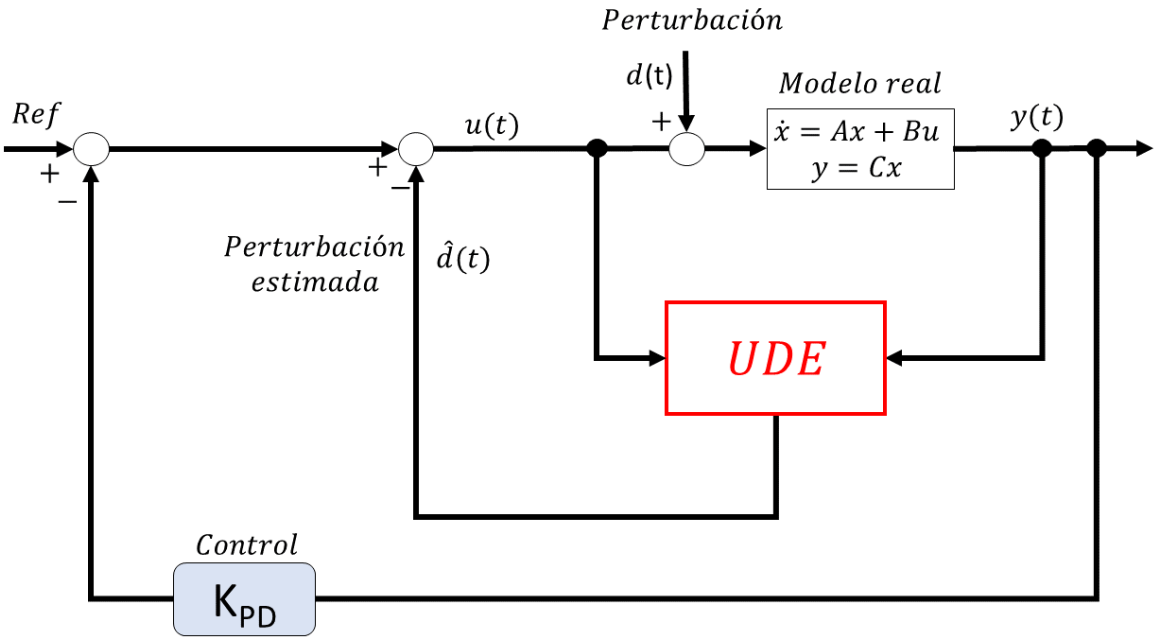


Figura 35: Control de orientación con PD + UDE

El primer paso para poder implementarlo era discretizar la ecuación (7.17). Para ello, se ha hecho uso de la función de Matlab "c2d", la cual, aplicando la transformada en z, es capaz de obtener la función discretizada. Por tanto, se ha procedido a realizar esto con cada término de la ecuación (7.17) por separado, y el resultado ha sido el siguiente:

Por un lado, tenemos la estimación de la perturbación asociada al término de x_2:

$$\widehat{d_{x_2}} = (1-\alpha)\cdot \widehat{d_{x_2}}_{ant} + \frac{\alpha}{ts\cdot K_\theta}\cdot x_2 - \frac{\alpha}{ts\cdot K_\theta}\cdot x_{2_{ant}} \tag{7.18}$$

Y por otro, la estimación de la perturbación relacionada con la entrada u:

$$\widehat{d_u} = (1-\alpha)\cdot \widehat{d_u}_{ant} - \alpha\cdot u_{ant} \tag{7.19}$$

Siendo ts el tiempo de muestreo, x_2 la medida de velocidad en la dirección del pitch, y α el único parámetro a ajustar para conseguir el funcionamiento deseado del UDE. Se advierte aquí claramente una de las principales ventajas del UDE, y es que se puede ajustar con un único parámetro. Esto resulta, por ende, una tarea muy sencilla en comparación con el ajuste algo engorroso de otros controladores, como el PID.

El valor de la perturbación total estimada por el UDE que se le restará a la acción de control será, por tanto:

$$\hat{d} = \widehat{d_{x_2}} + \widehat{d_u} \tag{7.20}$$

El código a implementar ha sido el siguiente:

1.Inicializar variables, abrir fichero de escritura

2.Comenzar bucle de control hasta desarme del Quadrotor:

3.Obtención de las variables necesarias para el control del pitch:

3.1. Lectura de los sensores de la IMU y la referencia por radio // de aquí se obtiene en forma de quaterniones el setpoint y la medida

3.2. Obtener en forma de vector de ángulos de Euler el setpoint y la medida a partir de los quaterniones

3.3. Obtener la derivada del ángulo del pitch a partir de la medida de velocidad de los sensores

4.Ejecución del UDE:

4.1. Cálculo de $\widehat{d_u}$

4.2. Cálculo de $\widehat{d_{x_2}}$

4.3. Cálculo de la perturbación total estimada

5.Saturación del UDE:

5.1. Si la perturbación estimada es mayor que 0.1, esta se iguala a 0.1

5.2. Si la perturbación estimada es menor que -0.1, esta se iguala a -0.1

6.Actualización de las variables del UDE

6.1. $\widehat{d_{u_{ant}}}$ se iguala a $\widehat{d_u}$

6.2. $\widehat{d_{x_{2_{ant}}}}$ se iguala a $\widehat{d_{x_2}}$

6.3. $x_{2_{ant}}$ se iguala a x_2

7.Ley de control sobre el pitch://cálculo de la acción de control a aplicar en el instante de muestreo

7.1. Cálculo de la acción proporcional

7.2. Cálculo de la acción derivada

7.3. Cálculo de la acción de control como la suma de la proporcional y la derivada, restándole a todo ello la perturbación total estimada

8.Se envía la acción de control a los motores en forma de tanto por 1//aplicación de la acción de control

9.Actualización de la acción de control

9.1. u_{ant} se iguala a u

10.Se guarda en fichero el valor de las variables de interés//esto se utiliza para la simulación

11.Se escribe en la tarjeta microSD el valor de las variables de interés//esto se utiliza para vuelos reales

12.Desarme del Quadrotor

7.3.4. Resultados obtenidos en simulación

Las figuras siguientes muestran el desempeño del UDE durante una simulación en la cual se le introduce una perturbación constante de 0.05, al igual que en los otros controladores.

Se puede apreciar que el UDE cumple perfectamente con su función, siendo capaz de anular por completo la perturbación y conseguir un error nulo frente a una entrada en escalón en un tiempo de establecimiento menor de 2 segundos. Demostramos aquí, por tanto, la principal ventaja del UDE frente al controlador PID anteriormente descrito: mientras que el PID introducía inevitablemente sobre-oscilaciones para anular la perturbación, el UDE es capaz de hacerlo sin sobre-oscilación alguna y en un tiempo de establecimiento menor en nuestro caso. Para conseguirlo únicamente ha hecho falta ajustar el parámetro α, que en nuestro caso adopta un valor de 0.01.

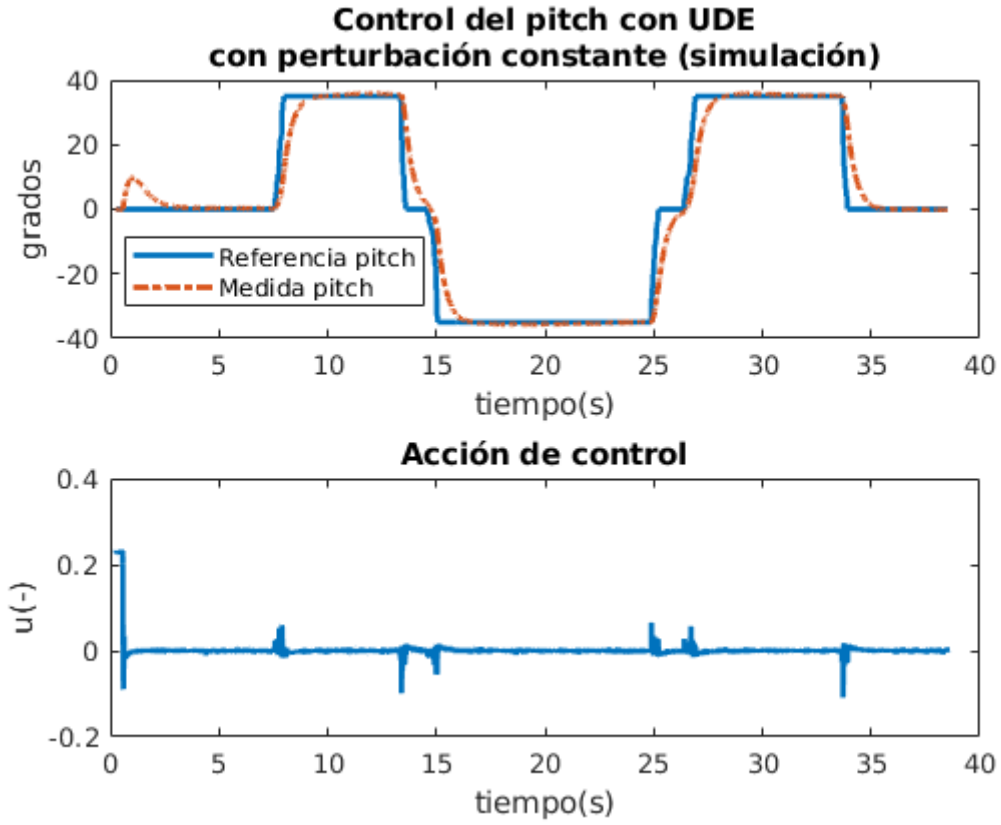


Figura 36: Control del pitch con UDE con perturbación constante (simulación)

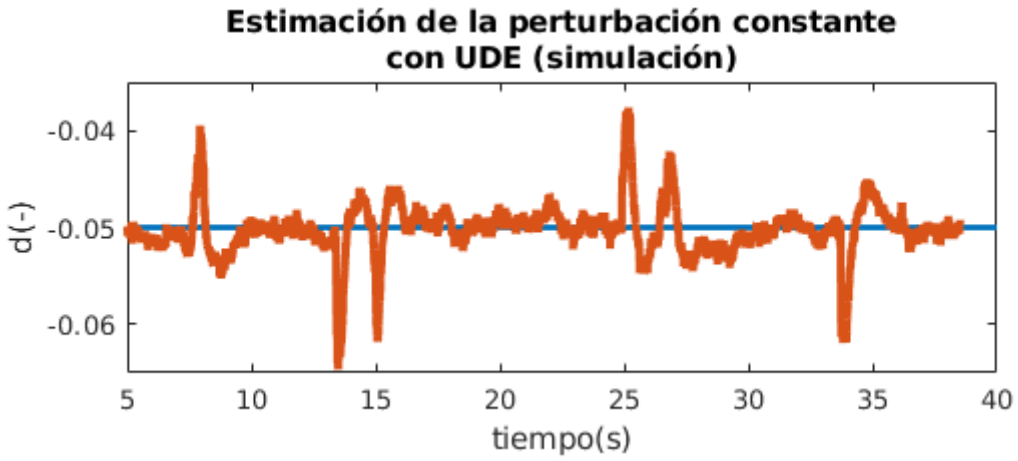


Figura 37: Estimación de la perturbación con UDE

También es necesario mencionar que, a diferencia de la acción integral del PID, que llegaba a doblar el valor de la perturbación a estimar, en el caso del UDE este es mucho más preciso, teniendo solo un error de ±0.01.

Tras observar el buen rendimiento del UDE ante perturbaciones constantes, se decidió someterlo a un experimento con una perturbación de carácter senoidal. Los resultados se muestran en las figuras 38 y 39.

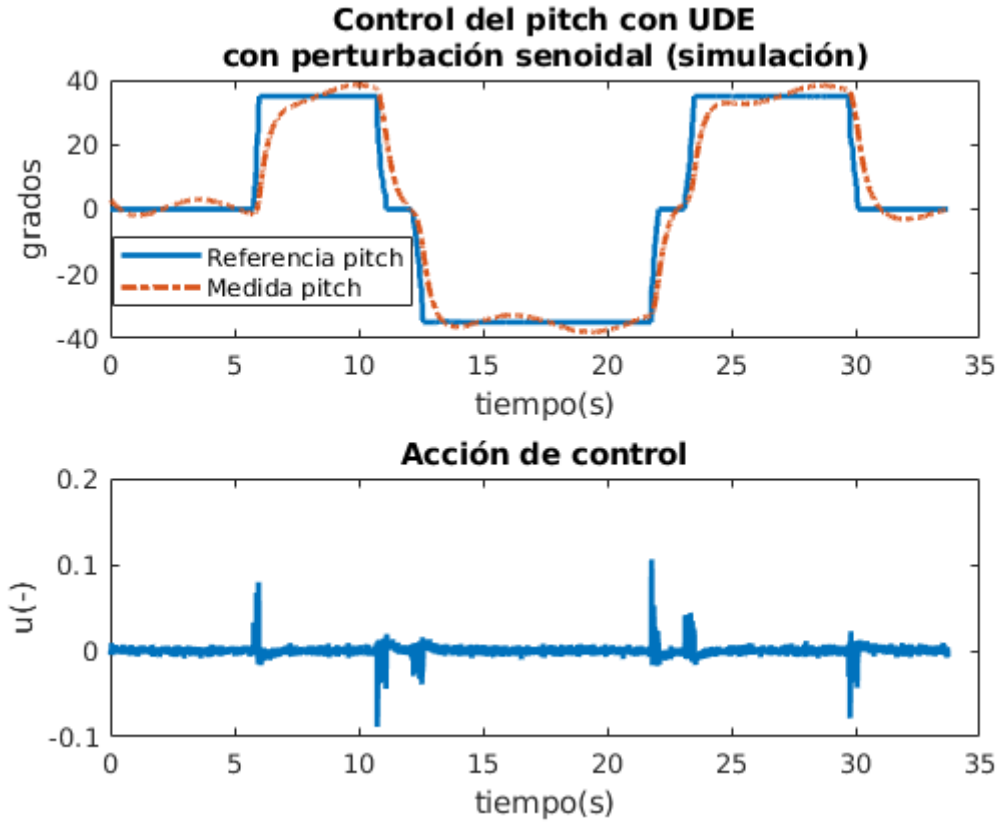


Figura 38: Control del pitch con UDE con perturbación senoidal (simulación)

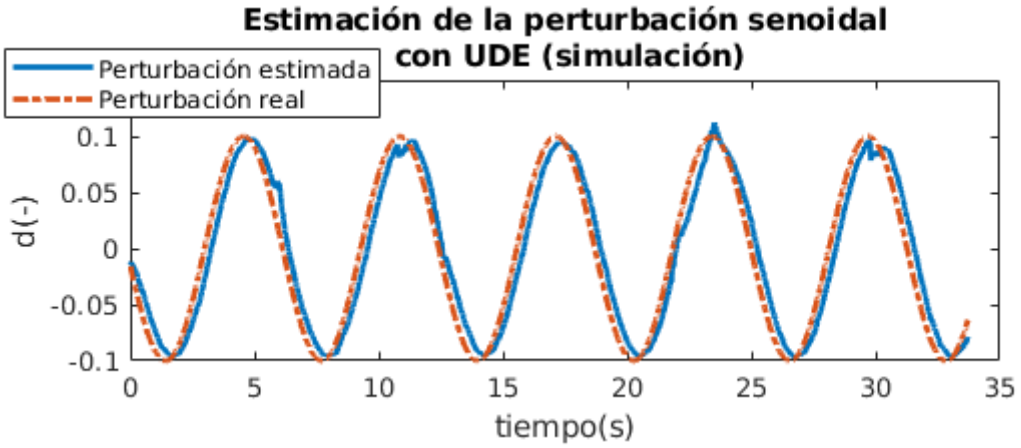


Figura 39: Estimación de la perturbación senoidal con UDE (simulación)

En este caso, como se ejemplifica en la gráfica 38, el UDE ya no es capaz de contrarrestar por completo la perturbación senoidal, quedando siempre la medida oscilando entre la referencia. Así mismo, en la estimación de la perturbación senoidal se observa un pequeño retraso entre la perturbación introducida y la estimada por el UDE. Queda entonces validada la capacidad de rechazo de perturbaciones del UDE únicamente a aquellas que son de naturaleza constante.

7.3.5. Resultados obtenidos en vuelo real

Los resultados conseguidos tras realizar un vuelo real introduciéndole por código una perturbación constante de 0.05 se muestran en las figuras 40 y 41.

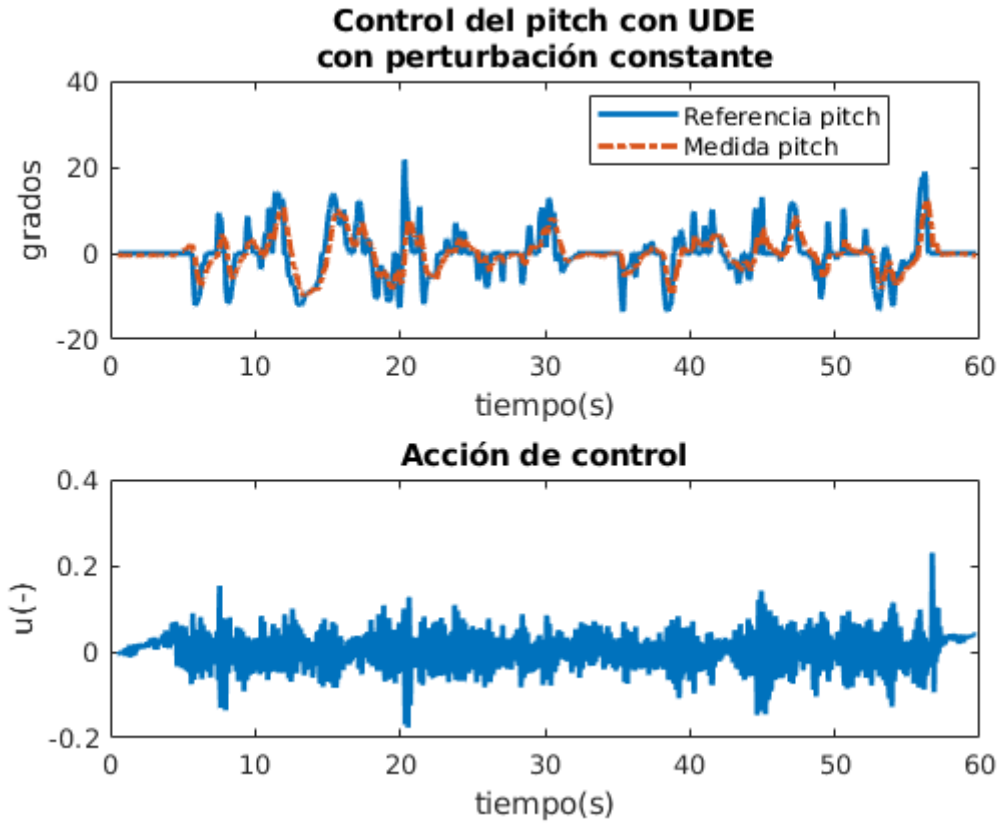


Figura 40: Control del pitch con UDE con perturbación constante (vuelo real)

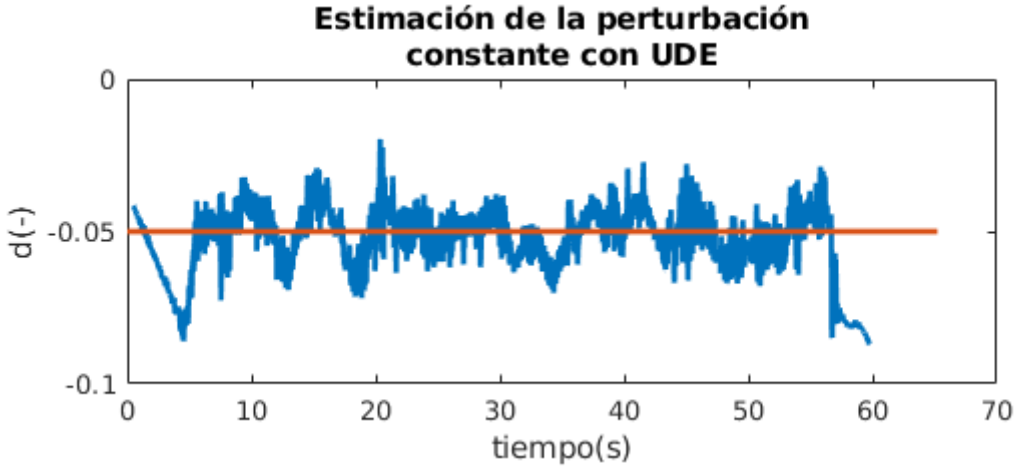


Figura 41: Estimación de la perturbación constante con UDE (vuelo real)

En la figura 40 se aprecia con claridad cómo el UDE es capaz de seguir la referencia sin problemas a lo largo de todo el vuelo. Así mismo, teniendo en cuenta que los picos de principio y fin que se observan en la estimación de la perturbación son debidos al despegue y aterrizaje, el UDE también demuestra de nuevo un desempeño muy superior al del PID a la hora de contrarrestarla.

Queda por tanto evidenciada la superioridad del UDE frente al PID. Sin embargo, para implementar correctamente el UDE es necesario un conocimiento del modelo bastante próximo al real, debiendo tener acceso a variables como la velocidad en la dirección del pitch. Sin embargo, ¿y si partes de ese sistema fuesen totalmente desconocidas para nosotros? La respuesta nos la dará el ESO, algoritmo que se detalla a continuación.

7.4. Controlador PD + ESO

7.4.1. Conceptos previos. Observadores de estado

Partiendo de los conceptos previos explicados en el apartado del UDE, es necesario advertir que los métodos de asignación de polos por realimentación del estado consideran una realimentación de todos los estados. Ahora bien, imaginémonos que algunos de estos estados no pudiesen medirse. Estrategias de control como el UDE quedarían inservibles en estos casos. Esto da lugar a los observadores de estados, los cuales consisten en sistemas dinámicos cuyos estados convergen al sistema real. Por tanto, básicamente lo que hacen es "observar" el sistema y dar como resultado una estimación del mismo. Esto se ilustra en la figura 42.

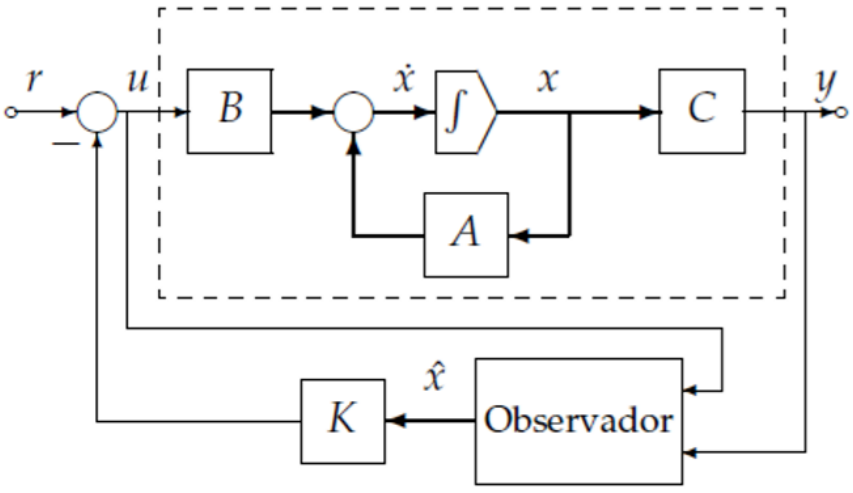


Figura 42: Observador

7.4.2. Concepto de ESO

Como ya se venía anunciando, en teoría de control el ESO (del inglés "*Extended State Observer*") es un sistema que proporciona una estimación del estado de un sistema real, a partir de las medidas de la entrada y salida de dicho sistema real. He aquí por tanto la ventaja que presenta el ESO frente al UDE:

mientras que para implementar el UDE se necesita conocer los diferentes estados del sistema, en el ESO no es necesario conocer más allá de la entrada y la salida. El ESO, por tanto, es una estrategia de carácter mucho más general que puede ser aplicada a cualquier sistema independientemente de los estados que se conozcan. La diferencia entre un observador de estado y un observador de estado extendido es que mientras que el observador nos estima las variables del sistema x_1 y x_2, el ESO es capaz de estimar no solo x_1 y x_2, sino también la perturbación que afecta al modelo. Para entender cómo el ESO lleva a cabo dichas estimaciones, se describe a continuación su modo de proceder de manera similar a como se hace en [17].

El diagrama que ilustra el funcionamiento del observador de estado extendido es el siguiente:

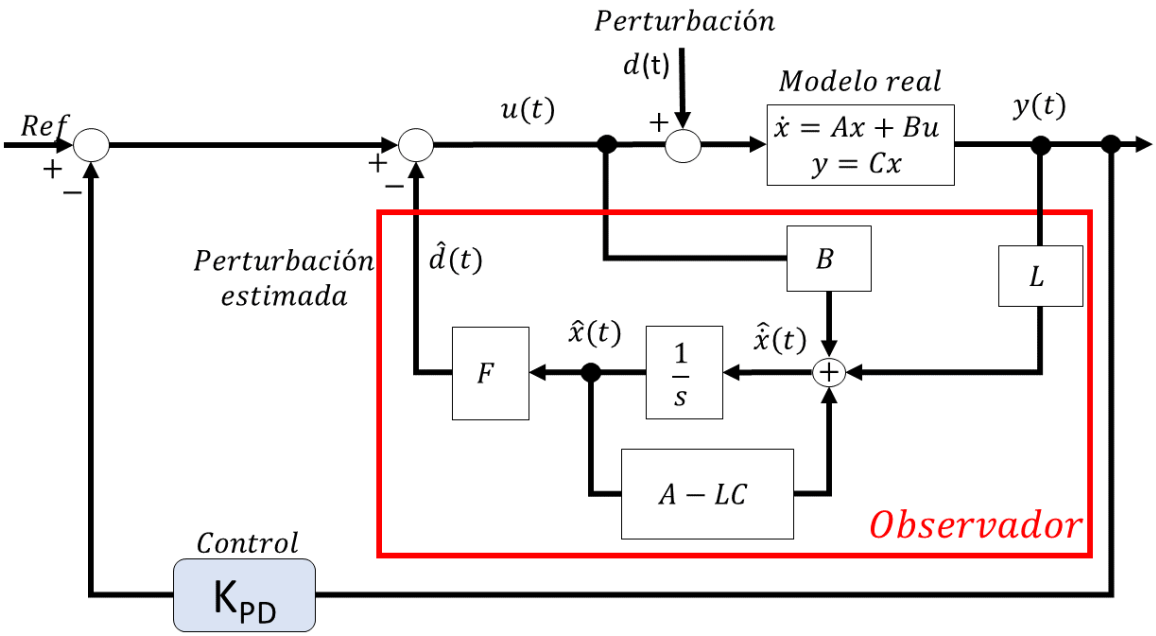


Figura 43: Control de orientación con PD + ESO

Donde:

$$B = \begin{bmatrix} 0 \\ K_\theta \\ 0 \end{bmatrix}; L = \begin{bmatrix} L1 \\ L2 \\ L3 \end{bmatrix}; F = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}; C = [1 \quad 0 \quad 0]$$

$$A - LC = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & K_\theta \\ 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} L1 \\ L2 \\ L3 \end{bmatrix} \cdot [1 \quad 0 \quad 0] = \begin{bmatrix} -L1 & 1 & 0 \\ -L2 & 0 & K_\theta \\ -L3 & 0 & 0 \end{bmatrix}$$

$\hat{x}(t)$ es un vector de 3 variables que contiene la información del estado estimado. A partir del diagrama podemos extraer la ecuación que describe el comportamiento del ESO:

$$\dot{\hat{x}}(t) = [A - LC] \cdot \hat{x}(t) + B \cdot u(t) + L \cdot y(t) \tag{7.21}$$

Por lo que, sustituyendo en (7.21) el valor de las matrices resulta:

$$\begin{pmatrix} \dot{\widehat{x_1}} \\ \dot{\widehat{x_2}} \\ \dot{\hat{d}} \end{pmatrix} = \begin{bmatrix} -L1 & 1 & 0 \\ -L2 & 0 & K_\theta \\ -L3 & 0 & 0 \end{bmatrix} \cdot \begin{pmatrix} \widehat{x_1} \\ \widehat{x_2} \\ \hat{d} \end{pmatrix} + \begin{bmatrix} 0 \\ K_\theta \\ 0 \end{bmatrix} \cdot u + \begin{bmatrix} L1 \\ L2 \\ L3 \end{bmatrix} \cdot y$$

De este modo, $\widehat{x_1}$ nos proporcionará la estimación del modelo, $\widehat{x_2}$ la derivada del mismo, y $\hat{d}$ la estimación de la perturbación, la cual restaremos a la acción de control. De la ecuación que aquí se expone, en lo referente a parámetros será el vector columna L el que podremos ajustar libremente para conseguir el desempeño del ESO deseado. En el siguiente apartado se darán más detalles del mismo.

7.4.3. Implementación el ESO

Para llevar a cabo la implementación del ESO el primer paso ha sido, al igual que con el UDE, discretizar la ecuación característica (7.21) que lo define. Para ello se ha vuelto a recurrir a Matlab, donde introduciendo el valor de las matrices se ha obtenido el resultado siguiente para cada variable del vector $\hat{x}(t)$:

$$\widehat{x_1} = (1 - L1) \cdot \widehat{x_1}_{ant} + ts \cdot \widehat{x_2}_{ant} + L1 \cdot y \tag{7.22}$$

$$\widehat{x_2} = -L2 \cdot \widehat{x_1}_{ant} + \widehat{x_2}_{ant} + K_\theta \cdot ts \cdot \hat{d}_{ant} + L2 \cdot y \tag{7.23}$$

$$\hat{d} = -L3 \cdot \widehat{x_1}_{ant} + \hat{d}_{ant} + L3 \cdot y \tag{7.24}$$

Es importante destacar que el vector de parámetros L se ha obtenido a partir de un script desarrollado en Matlab donde introduciéndole el valor de 3 polos, nos devuelve los parámetros del vector L ya discretizados. Por tanto, para ajustar el ESO únicamente había que observar su comportamiento, y si por ejemplo se quería disminuir el tiempo de establecimiento había que volver a situar los polos para que fuesen más rápidos. Tras varios experimentos de ajuste, los parámetros obtenidos para un desempeño óptimo fueron:

$$L = \begin{bmatrix} 0.45 \\ 2.1167 \\ 0.03333 \end{bmatrix}$$

Mientras que el valor de K_θ se mantuvo en 100, para poder comparar posteriormente el ESO con el UDE.

De este modo, el código que se ha programado ha sido el siguiente:

```
1.Inicializar variables, abrir fichero de escritura
```

2.Comenzar bucle de control hasta desarme del Quadrotor:

3.Obtención de las variables necesarias para el control del pitch:

3.1. Lectura de los sensores de la IMU y la referencia por radio // de aquí se obtiene en forma de quaterniones el setpoint y la medida

3.2. Obtener en forma de vector de ángulos de Euler el setpoint y la medida a partir de los quaterniones

3.3. Obtener la derivada del ángulo del pitch a partir de la medida de velocidad de los sensores

4.Ejecución del ESO:

4.1. Cálculo de $\widehat{x_1}$

4.2. Cálculo de $\widehat{x_2}$

4.3. Cálculo de $\hat{d}$

5.Saturación de la perturbación estimada del ESO:

5.1. Si la perturbación estimada es mayor que 0.1, esta se iguala a 0.1

5.2. Si la perturbación estimada es menor que -0.1, esta se iguala a -0.1

6.Actualización de las variables del ESO

6.1. $\widehat{x_{1_{ant}}}$ se iguala a $\widehat{x_1}$

6.2. $\widehat{x_{2_{ant}}}$ se iguala a $\widehat{x_2}$

6.3. $\hat{d}_{ant}$ se iguala a $\hat{d}$

7.Ley de control sobre el pitch://cálculo de la acción de control a aplicar en el instante de muestreo

7.1. Cálculo de la acción proporcional

7.2. Cálculo de la acción derivada

7.3. Cálculo de la acción de control restándole la perturbación estimada por el ESO

8.Se envía la acción de control a los motores en forma de tanto por 1//aplicación de la acción de control

9.Se guarda en fichero el valor de las variables de interés//esto se utiliza para la simulación

10.Se escribe en la tarjeta microSD el valor de las variables de interés//esto se utiliza para vuelos reales

11.Desarme del Quadrotor

7.4.4. Resultados obtenidos en simulación

Al igual que se ha procedido con el UDE, se ha sometido al ESO a una simuñación en *jMAVSim* con una perturbación constante de -0.05. El resultado se muestra en las figuras 44, 45 y 46.

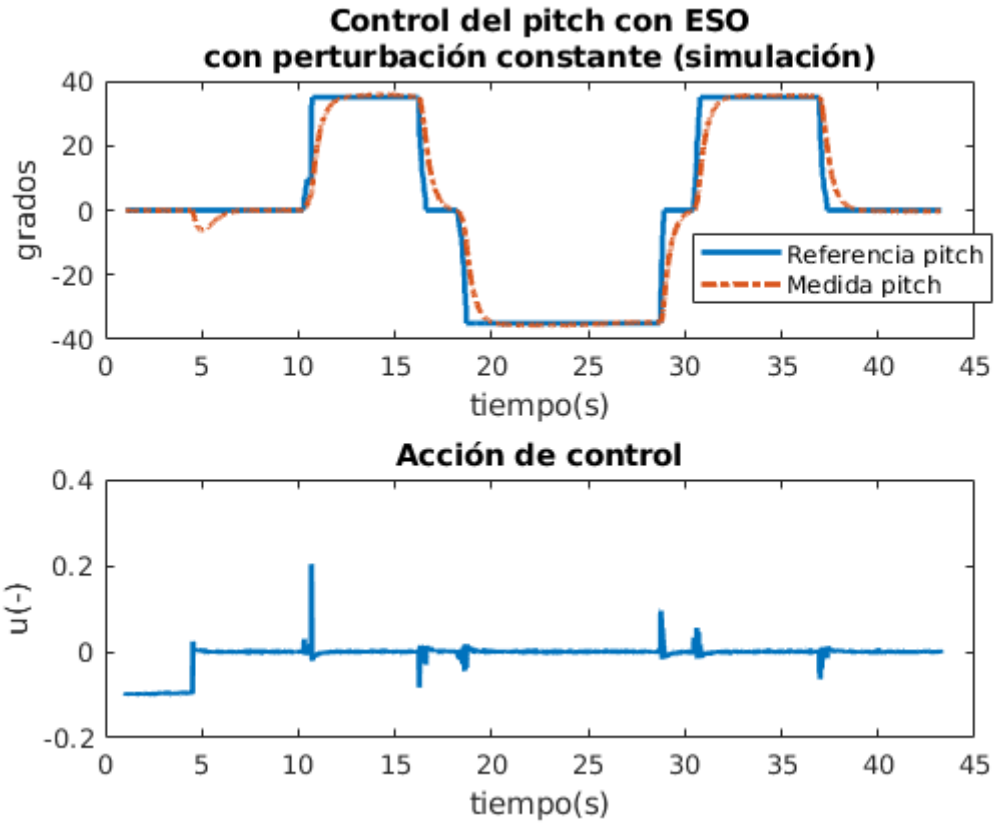


Figura 44: Control del pitch con ESO con perturbación constante (simulación)

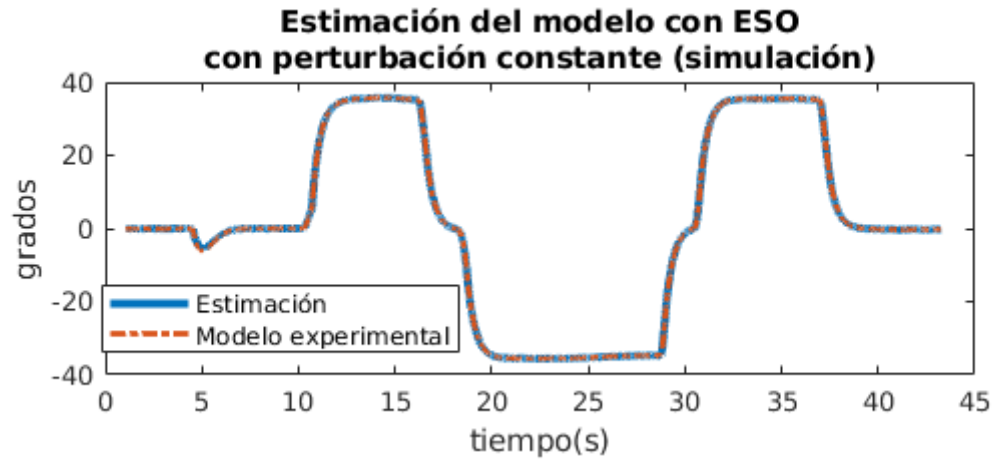


Figura 45: Estimación del modelo con ESO (simulación)

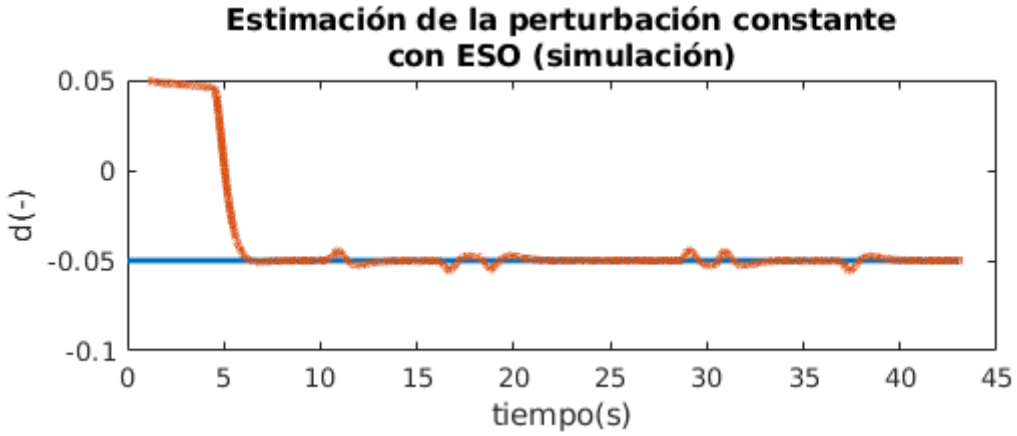


Figura 46: Estimación de la perturbación constante con ESO (simulación)

Analizando los resultados obtenidos, podemos concluir que el desempeño del ESO frente a una perturbación constante en simulación es prácticamente idéntico al del UDE, con un tiempo de establecimiento menor de 2 segundos, y con un error en la perturbación estimada algo inferior al del UDE. Sin embargo, el resultado que más puede sorprender a primera vista es el de la estimación del modelo. En esta gráfica, se ha representado la $\widehat{x_1}$ estimada por el ESO frente a la x_1 del modelo experimental, y se puede apreciar perfectamente cómo la estimación del modelo es eminentemente perfecta.

7.4.5. Resultados obtenidos en vuelo real

Una vez más, se somete al Quadrotor a un experimento con perturbación constante introducida por código, en este caso, un vuelo real. Los resultados se muestran en las figuras 47, 48 y 49.

De nuevo y al igual que con el UDE, se obtienen unos resultados muy parecidos. La medida sigue a la referencia con bastante acierto, y la perturbación estimada no dista mucho del valor impuesto de -0.05. La única diferencia remarcable sería la rapidez de actuación de la estimación de la perturbación, donde en el caso del ESO se aprecia que es más lenta que en la del UDE. Esto se debe a la selección de polos a la hora de obtener el vector de parámetros L anteriormente descrito. Sin embargo, dado que de este modo ya se conseguía un rendimiento satisfactorio, se optó por no poner polos más rápidos dada la tendencia que tenía el sistema a desestabilizarse.

De nuevo, al igual que sucedía con la simulación, el ESO es capaz de estimar el modelo experimental de manera muy precisa. De esto podemos concluir que independientemente de que se conozca el modelo o no, se pueden adoptar estrategias de control como la del ESO o el UDE, obteniendo unos rendimientos bastante elevados en ambos casos. Queda demostrado pues, que no existe gran diferencia entre los resultados del UDE y del ESO en lo que respecta a sistemas de segundo orden como el nuestro. En ciertos casos donde el sistema tiene mucho ruido de medida tal vez se pueda ajustar mejor un observador como el ESO, pero a efectos prácticos, ambos tienen prestaciones muy similares. El uso del UDE, por tanto, quedará relegado a aquellos sistemas donde se tenga información completa del estado, mientras que el ESO se empleará en aquellos donde no la haya.

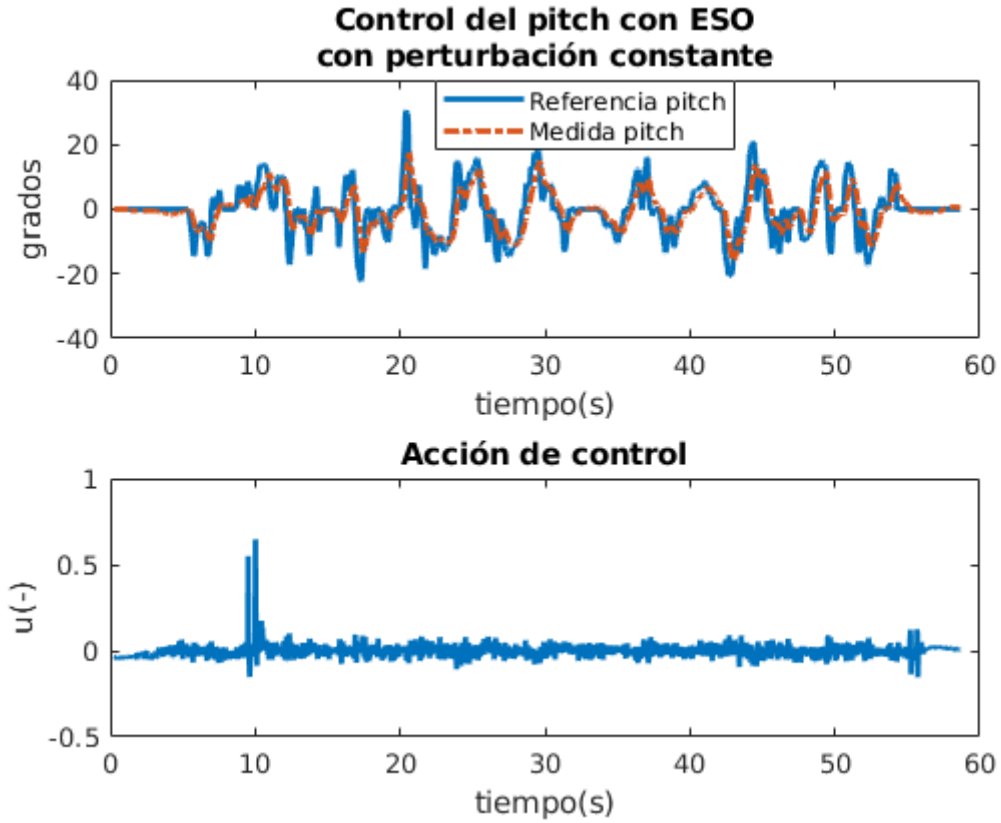


Figura 47: Control del pitch con ESO con perturbación constante (vuelo real)

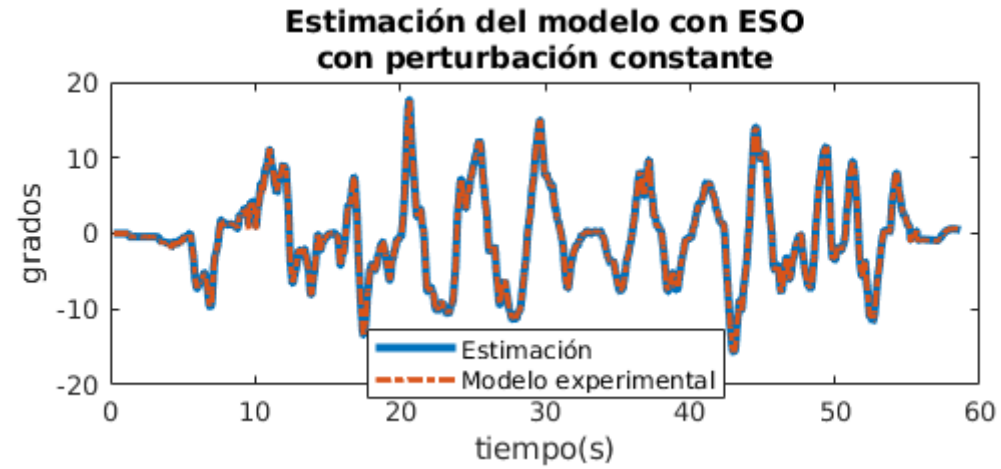


Figura 48: Estimación del modelo con ESO (vuelo real)

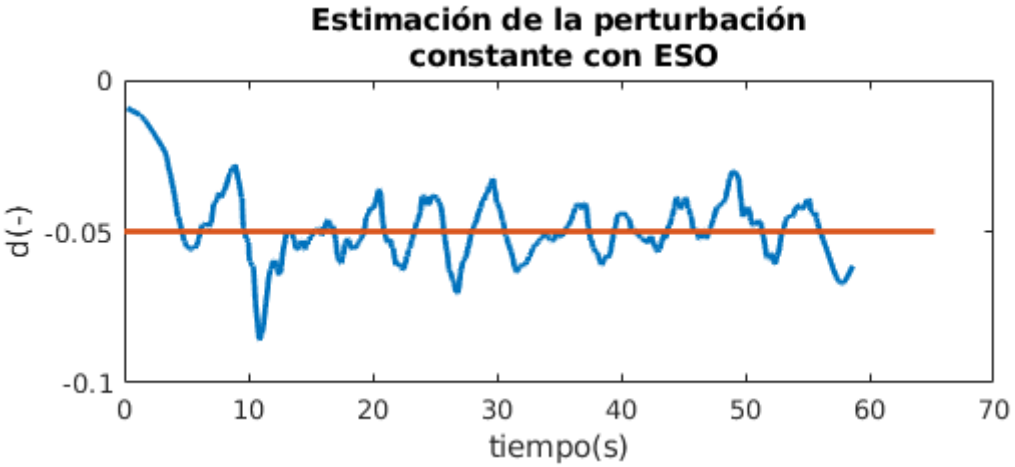


Figura 49: Estimación de la perturbación constante con ESO (vuelo real)

Capítulo 8

Desarrollo y resultados de la ley de control de posición implementada – Sistema RTK GPS

En esta sección se presenta y argumenta el algoritmo de control de posición desarrollado para el trabajo a partir de un sistema de posicionamiento RTK – GPS. Así mismo, se introduce de manera abreviada al lector en el estado del arte de los sistemas GPS y su funcionamiento.

El Sistema de Posicionamiento Global, comúnmente conocido como GPS (del inglés, *"Global Position System"*), consiste en un sistema capaz de determinar la posición de un cuerpo en tiempo real en cualquier lugar de la Tierra. Para ello, el GPS emplea un método basado en la trilateración, el cual se explicará más adelante En este capítulo únicamente se hace alusión al GPS a modo introductorio, aunque podemos hallar más información sobre los mismos en [18].

Así pues, el hecho de poder obtener la posición de un objeto en cualquier zona del planeta a partir de un único receptor convierte al sistema GPS en una herramienta muy útil hoy en día. Con una precisión del orden de magnitud de metros o centímetros, este instrumento ha demostrado ser idóneo en muchos casos para todo tipo de vehículos destinados a recorrer grandes distancias. Sin embargo, a la hora de extrapolarlo al entorno del Quadrotor, encontramos diferentes dificultades en cuanto a precisión se refiere. Por una parte, dado que el Quadrotor está diseñado para operar en espacios pequeños, una precisión de por ejemplo 20 metros, que podría resultar perfectamente válida para vehículos de mayor tamaño como aviones o navíos, resulta inadmisible para un control de posición de un multi-rotor. Por otro lado, el hecho de que el Quadrotor vuele a no demasiada altura, dificulta la captación de satélites de la señal GPS, lo cual no supone ningún problema para aquellos vehículos que circulan en espacios abiertos.

De este modo, para comprender el problema de la precisión que acabamos de advertir, es necesario comprender el funcionamiento del GPS. Como se mencionaba anteriormente, el GPS funciona empleando un método de posicionamiento basado en la trilateración. Este consiste en determinar la posición relativa del cuerpo respecto de tres o más puntos de referencia conocidos y las distancias a

los mismos. De este modo, supongamos que las distancias L_1 y L_2 desde el objeto a los puntos $\mathcal{O}_1$ y $\mathcal{O}_2$ son conocidas. De las dos circunferencias con centro en $\mathcal{O}_1$ y $\mathcal{O}_2$ y radios L_1 y L_2 respectivamente, las únicas dos posibles soluciones se encuentran en la intersección de ambas. Seguidamente, se introduce un tercer punto de referencia $\mathcal{O}_3$ procediendo del mismo modo, por lo que una de las dos anteriores soluciones queda fuera, dejando por tanto una única solución. Un ejemplo de este método se muestra en la figura 50.

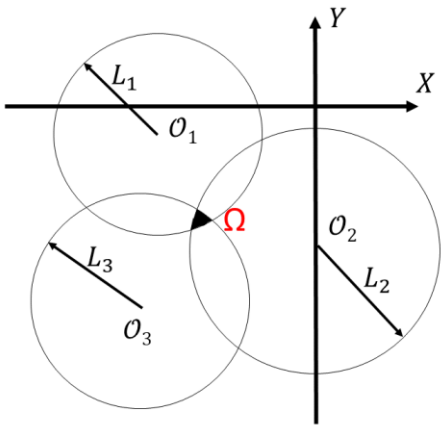


Figura 50: Trilateración

Sin embargo, estas distancias que a priori se han considerado datos del problema, generalmente no suelen ser conocidas de manera exacta, por lo que la solución, en vez de ser única, se corresponde con una región Ω, la cual abarca el conjunto de posibles soluciones. De esta forma, conforme se vayan introduciendo más puntos de referencia $\mathcal{O}_i$ el rango de soluciones se irá acotando y por tanto el error irá disminuyendo.

Una vez comprendido el método de la trilateración, es menester prestar atención a cómo obtenemos esas distancias a los puntos de referencia, pues como se ha visto, el error en la posición dependerá de la precisión de las mismas. En el caso que nos ocupa, dichas distancias serán las distancias del cuerpo a los satélites.

Los satélites se encuentran orbitando alrededor de la Tierra. Durante su trayecto, van emitiendo señales que llevan codificadas la hora atómica en la que dicha señal fue emitida, y diversos parámetros mediante los cuales el receptor es capaz de obtener la posición del satélite emisor. Por otra parte, los receptores también tienen conocimiento de la hora atómica en la cual dicha señal fue recibida en el planeta. Por tanto, con una simple resta son capaces de conocer el tiempo que ha tardado la señal en recorrer la distancia desde el satélite hasta el receptor, y sabiendo que dichas señales viajan a la velocidad de la luz, resulta trivial obtener esta distancia.

Sin embargo, tanto el dato de la hora atómica del satélite como la del receptor llevan errores de medida, los cuales, por pequeños que sean, teniendo en cuenta la velocidad a la que viaja la señal, podría resultar en un error de posición inaceptable. Así mismo, encontramos otras fuentes de error a parte del de los relojes, como lo pueden ser el hecho de que la velocidad de la luz no es constante a lo

largo de toda la atmosfera; el efecto multi-trayectoria, ocasionado por las posible múltiples reflexiones de las señales de los satélites antes de llegar al receptor; o un error inducido por un ruido aleatorio.

Es por todo ello por lo que resulta imperativo emplear técnicas que reduzcan dicho error a su mínima expresión. Una de ellas, y la que se ha empleado en este proyecto, ha sido la de los sistemas RTK, o como se los conoce cuando se refieren al uso particular de la red GPS, sistemas DGPS.

8.1. Sistemas RTK-GPS o DGPS

Los sistemas RTK (del inglés, "*Real Time Kinematic*") o los sistemas DGPS (del inglés, *"Differential GPS"*) consisten en sistemas basados en señales GPS y una estación de referencia en tierra fija. De este modo, bastará con conocer la posición exacta de esta estación de referencia, para poder deducir el error cometido y transmitirlo por radio en tiempo real al resto de receptores en un radio de entre 50 y 100 km. Para ello, la estación compara su posición conocida con la obtenida a partir de las señales GPS, obteniendo la diferencia entre ambas y las correcciones que se deben aplicar para solventarlas. Estas correcciones serán transmitidas al resto de usuarios, cuyos sistemas DGPS las emplearán para determinar su posición de manera mucho más precisa. En nuestro caso, nos ha permitido obtener una precisión de posición del orden de 20 cm, lo cual ha sido realmente satisfactorio para los experimentos realizados con el Quadrotor. Para que esto fuese posible, se ha empleado un set de 2 placas C94-M8P y dos antenas GPS de la marca ublox©, disponiendo una placa y una antena en la estación de referencia fija, y la otra placa y antena en el Quadrotor. Estas se muestran en la figura 51.

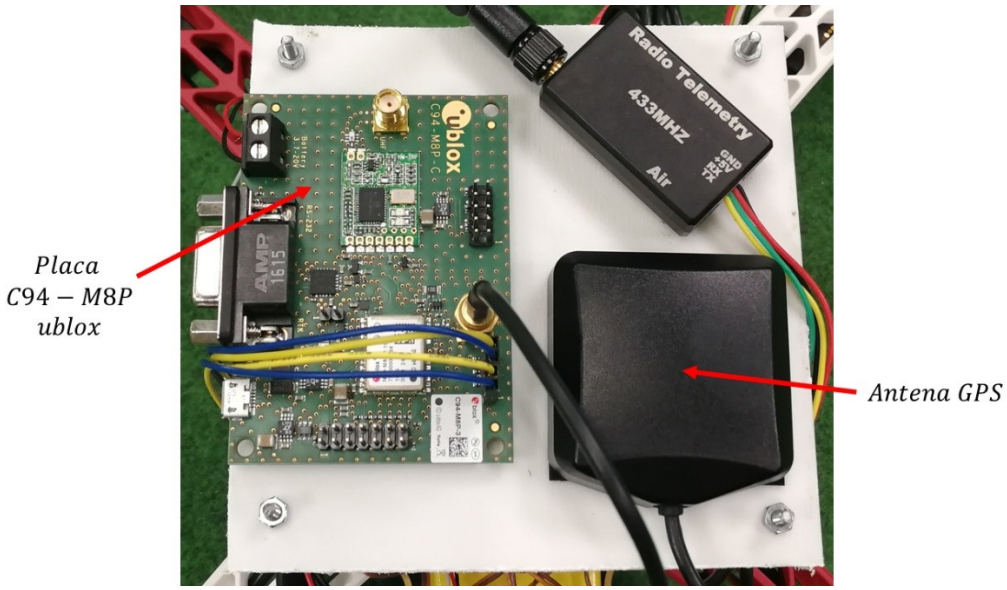


Figura 51: Placa y antena GPS de ublox

8.2. Control de posición mediante sistema RTK-GPS implementado

Como ya se advertía en el apartado "*6.4.2. Control de posición del Pixhawk*" de este mismo escrito, el control de posición del Pixhawk funciona a partir de las estimaciones realizadas por un filtro de Kalman de estado extendido, también conocido como EKF2 ("*State Extended Kalman Filter*"). Este no es más que un observador que mediante diversas iteraciones, es capaz de seleccionar la matriz de realimentación del estado para la cual se obtiene una covarianza en el error de la estimación mínima. Más información al respecto la podemos encontrar en [19].

De este modo, el filtro de Kalman nos permite obtener las estimaciones de diferentes estados como lo son la orientación (en quaterniones), la velocidad (NED) y la posición (NED). Todos ellos serán necesarios para realizar un correcto control de posición, el cual se explica a continuación.

El control de posición desarrollado para el Quadrotor se muestra en la figura 52.

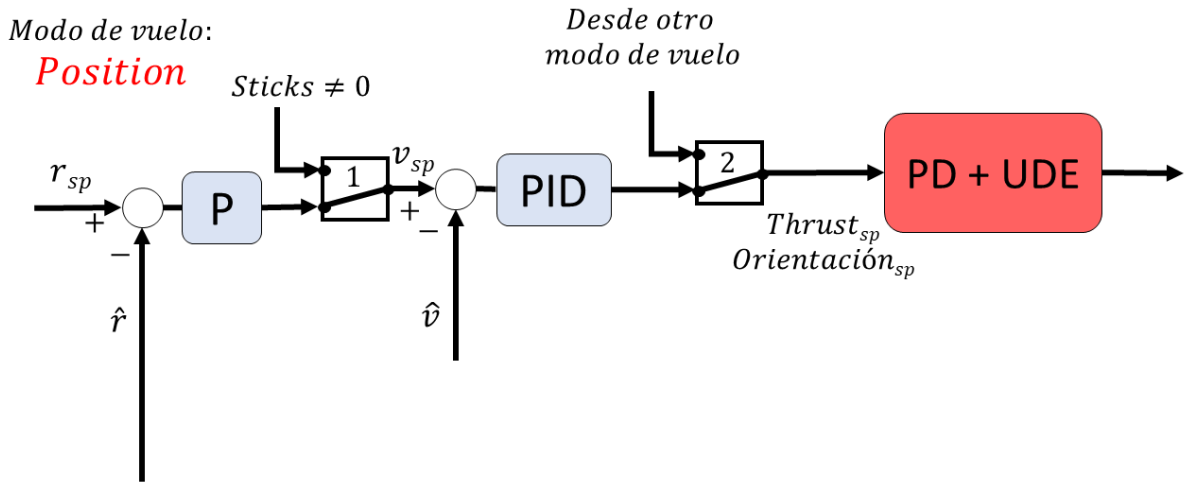


Figura 52: Control de posición con 3 niveles

Como se puede apreciar, a diferencia de los controles de orientación donde se prescindía por completo del código por defecto, en el caso del control de posición se ha decidido mantener parte de la estrategia de control original. Concretamente, lo que se ha hecho ha sido mantener los bucles superiores P y PID, y se le ha añadido un tercer nivel inferior con el PD y el UDE propios.

De este modo, el funcionamiento del control de posición es el siguiente:

Mientras el modo de vuelo actual no sea el de *Position,* el control realizará un baipás de los dos primeros bucles y accederá directamente al PD + UDE. Esto se ilustra en la figura 52 a modo de multiplexor 2. En estas circunstancias, las referencias serán dadas directamente por radio y el dron no contará con control de posición alguno, por lo que en definitiva se estará aplicando únicamente control de orientación, como sucedía en el capítulo anterior de este trabajo.

Una vez activado el modo *Position*, mientras los sticks de la radio estén en una posición distinta de cero, es decir, mientras se desee mover al dron, únicamente se activará el control de velocidad ofrecido por el PID y el de orientación por el PD + UDE. Esto se ilustra en el multiplexor 1. El control de velocidad se encargará de generar los setpoints tanto de la orientación como del thrust, y se los transmitirá al control de orientación propio, el cual se encargará de estabilizar el vehículo.

Por tanto, cuando realmente se posee un control de posición es cuando se activan todos los bucles, y esto se da en dos casos:

- Cuando se mantiene la posición
- Cuando la velocidad solicitada en un eje es nula

De este modo, imaginemos que estamos volando el dron en modo *Position* y queremos llevarlo a un punto r. Para ello, moveremos al dron por radio hasta la posición deseada, y una vez allí dejaremos los sticks a 0 con tal de mantener la posición. Es en este instante cuando se activará el control de posición. El dron, por inercia, tenderá a seguir moviéndose en la dirección que llevaba, por lo que se desplazará del punto r deseado. En ese preciso momento, el control comparará la posición actual del dron $\hat{r}$ obtenida a partir de las estimaciones del filtro de Kalman y la señal GPS, con el punto r deseado y generará internamente una referencia de velocidad a partir de la diferencia entre ambos puntos y una ganancia Kp del controlador P. Esta referencia será transmitida al PID, que generará las referencias de orientación y thrust, las cuales a su vez serán enviadas al controlador PD + UDE, encargándose de estabilizar el dron, a la vez de corregir la posición.

Por tanto, podremos ir moviendo el dron libremente y dejarlo quieto en el lugar deseado gracias a las medidas del GPS y el uso que nuestro control hace de ellas. Los resultados de los experimentos realizados se describen en el siguiente punto.

8.3. Resultados obtenidos en simulación

Al igual que se ha hecho con el control de orientación, el control de posición se ha sometido a múltiples simulaciones con tal de asegurar su correcto funcionamiento antes de pasar a vuelos reales. En las siguientes figuras se muestran los resultados de una simulación con una duración de casi unos 100 segundos, la cual ha consistido en mover y dejar quieto al dron un total de 3 veces manteniendo la altura; y aplicándole una perturbación constante de -0.05 durante todo el vuelo. El objetivo era ver si se conseguía un funcionamiento adecuado de cada uno de los bucles de la estrategia de control descrita en el apartado anterior.

Por una parte, era necesario comprobar el correcto seguimiento de las medidas del GPS y la IMU tanto de posición como de velocidad en el modo de vuelo *Position*; y por otra observar cómo se comportaba el UDE bajo la entrada de referencias generadas automáticamente por código cuando se mantenía la posición del dron, a la vez que rechazaba la perturbación introducida por código.

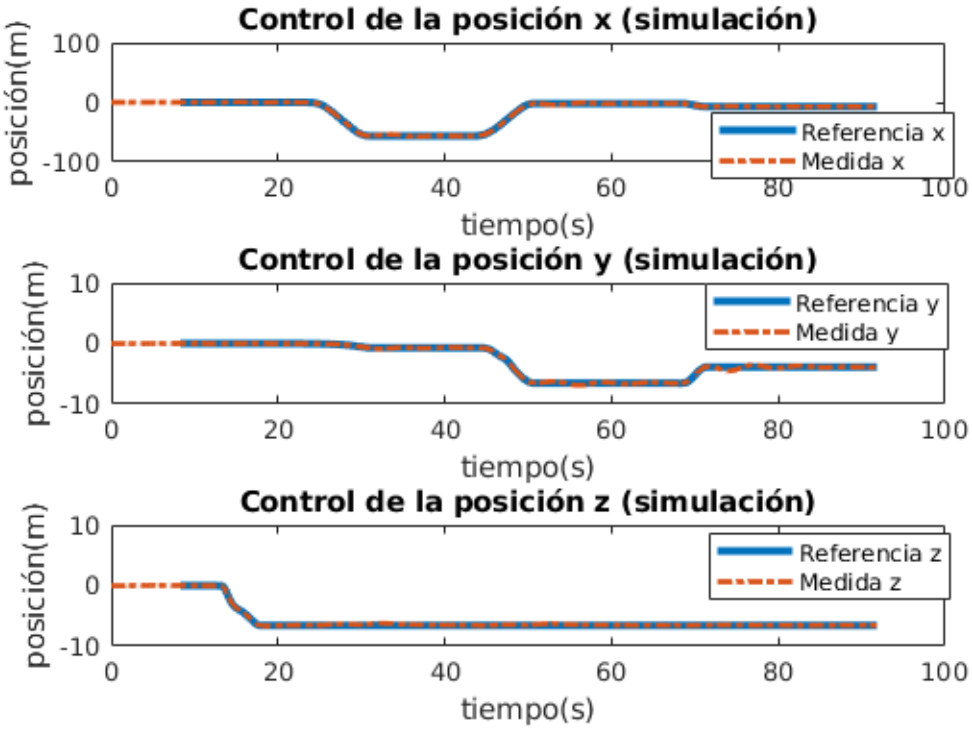


Figura 53: Control de posición x, y, z (simulación)

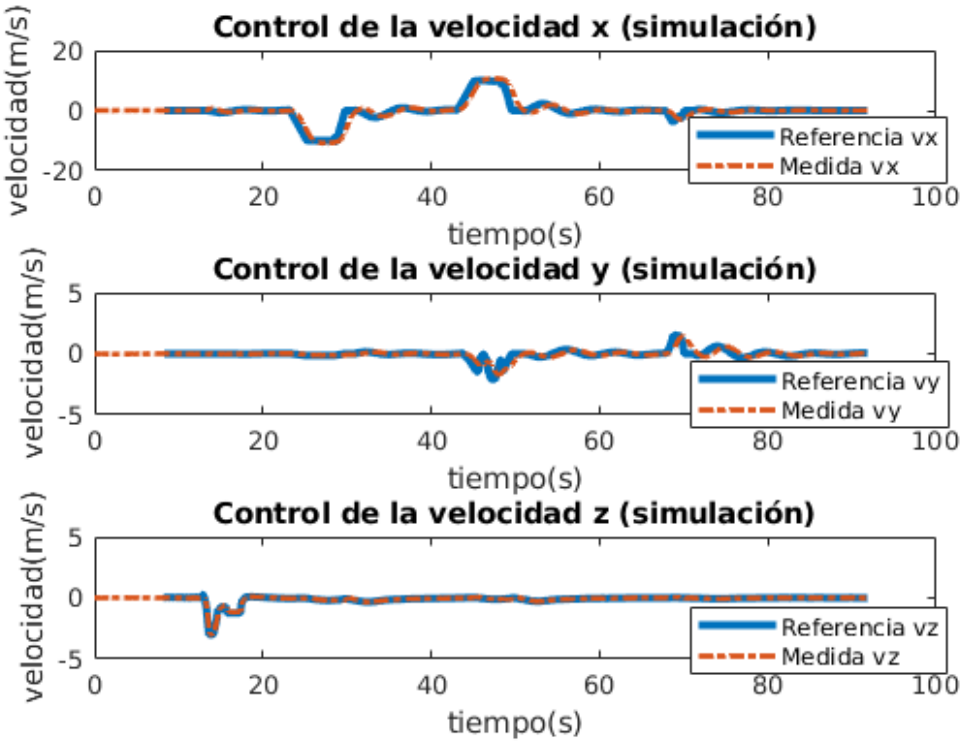


Figura 54: Control de velocidad x, y, z (simulación)

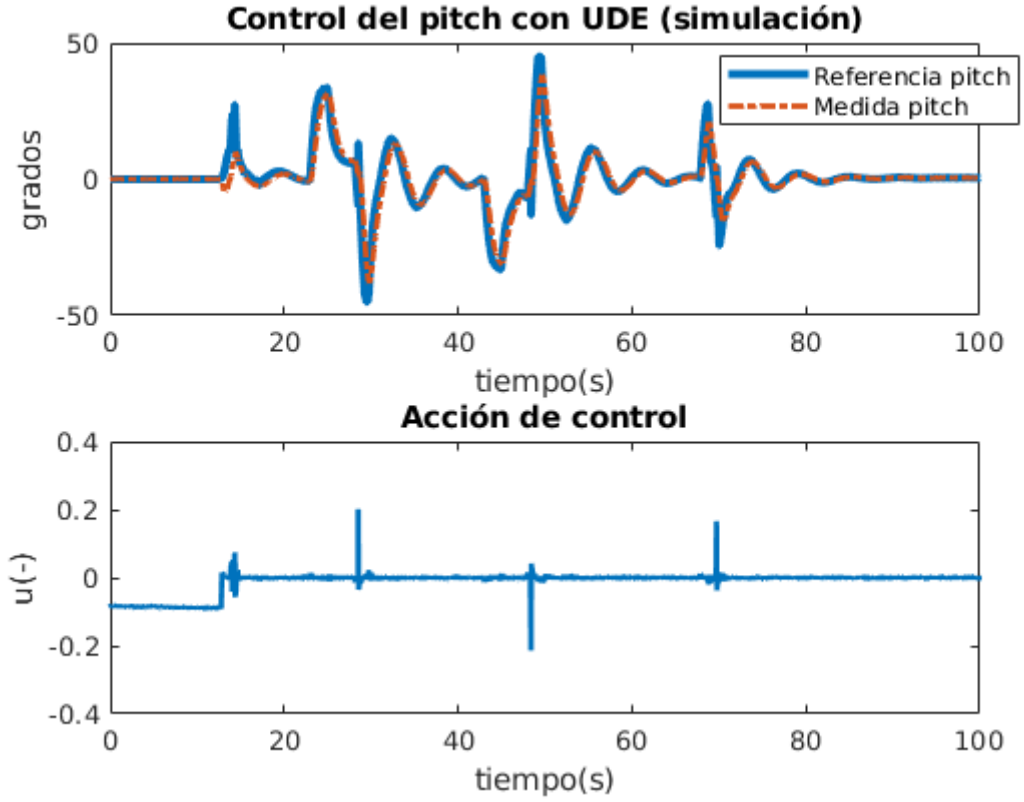


Figura 55: Control del pitch con UDE (simulación)

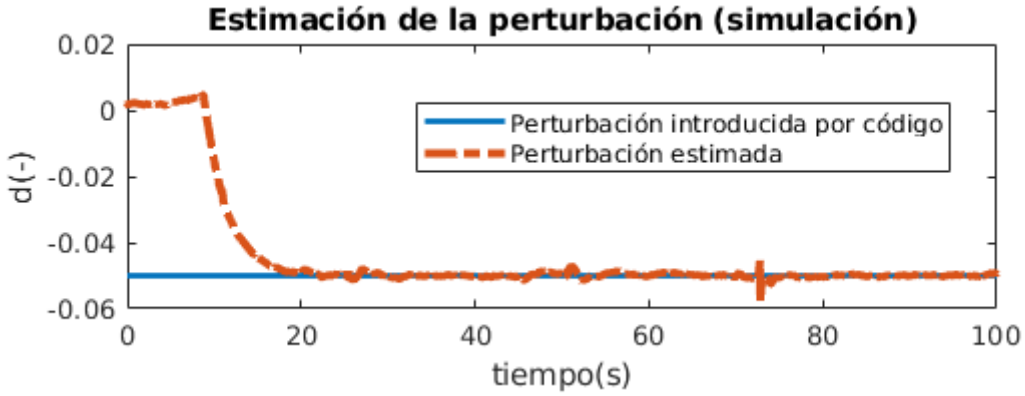


Figura 56: Estimación de la perturbación con UDE (simulación)

Como se puede observar en las figuras 53, 54, 55 y 56, a priori los resultados obtenidos en simulación son más que satisfactorios. El control de la posición en x, y, z es prácticamente exacto, y el de la velocidad bastante acertado, donde como cabría esperar, los picos se corresponden con las variaciones de la posición en la gráfica superior. Notar que la posición en z es negativa, ya que como se advertía en el modelo teórico del Quadrotor, el sentido del eje solidario a la altura en NED apunta al centro de la Tierra.

Sin embargo, lo realmente interesante del experimento son los resultados obtenidos por parte del UDE, que a grosso modo es donde reside el control. En la figura 55 se puede apreciar cómo, aunque la posición en x, y, z no varíe en diferentes tramos de la gráfica, el control no deja de actuar. Esto se debe a que, como se explicaba anteriormente, cuando se le indica al dron que mantenga la posición, se activa el bucle superior que genera de manera automática las referencias necesarias para corregir el error de posición. En nuestro caso, el UDE cumple notablemente con su cometido a la hora de responder ante dichas referencias y controlar al Quadrotor de manera adecuada, permaneciendo prácticamente solapadas las líneas de medida y referencia. Por otro lado, al mismo tiempo que se encarga de controlar la posición, el UDE es capaz de contrarrestar la perturbación introducida de -0.05 de manera muy precisa, con un error que no llega al ±0.005

Es necesario mencionar que, aunque se ha implementado el algoritmo en todos los ángulos, únicamente se ha representado aquí el control del pitch, ya que los resultados eran muy similares tanto en roll como yaw y las conclusiones a las que se llega son las mismas.

A la luz de los buenos resultados obtenidos en simulación, el siguiente paso ha sido someter al Quadrotor a vuelos reales.

8.4. Resultados obtenidos en vuelos reales

En el caso de vuelos reales, se ha sometido al dron al igual que en simulación, a varios experimentos donde se movía y se dejaba quieto el Quadrotor sucesivas veces. Una vez comprendida cómo funciona la estrategia de control, es fácil deducir que lo realmente importante del control de posición es ver si el dron es capaz de mantener su posición, pues durante su movimiento de un punto a otro, no existe gran diferencia con los vuelos ya mostrados en el capítulo anterior. Por ello, en las siguientes figuras se muestra un fragmento de los resultados obtenidos durante uno de los vuelos, donde se le indicaba al vehículo que debía quedarse quieto. Lo que se pretendía por tanto era observar si era capaz de realizar las correcciones pertinentes para mantener la posición a la vez que rechazaba diversas perturbaciones debidas al viento, etc.

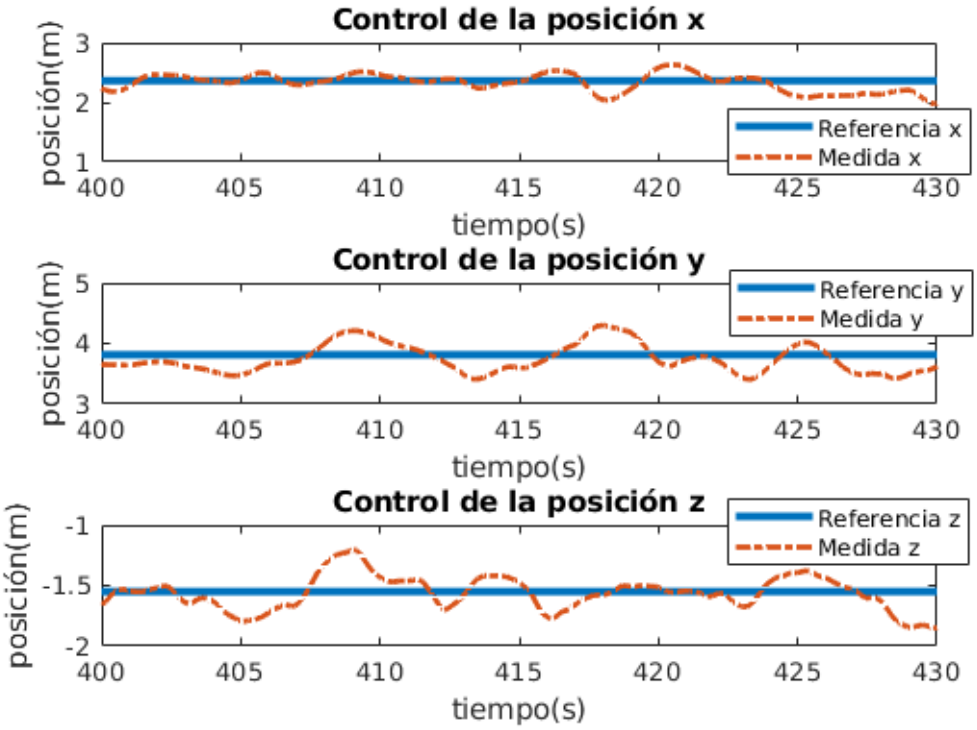


Figura 57: Control de posición x, y, z (vuelo real)

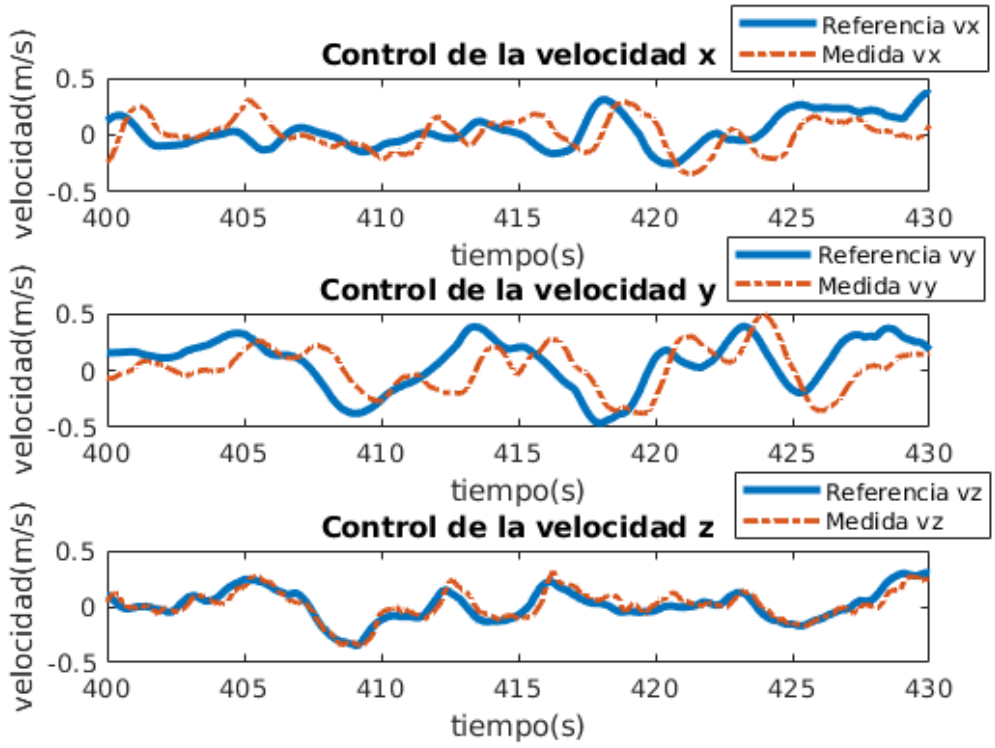


Figura 58: Control de velocidad x, y, z (vuelo real)

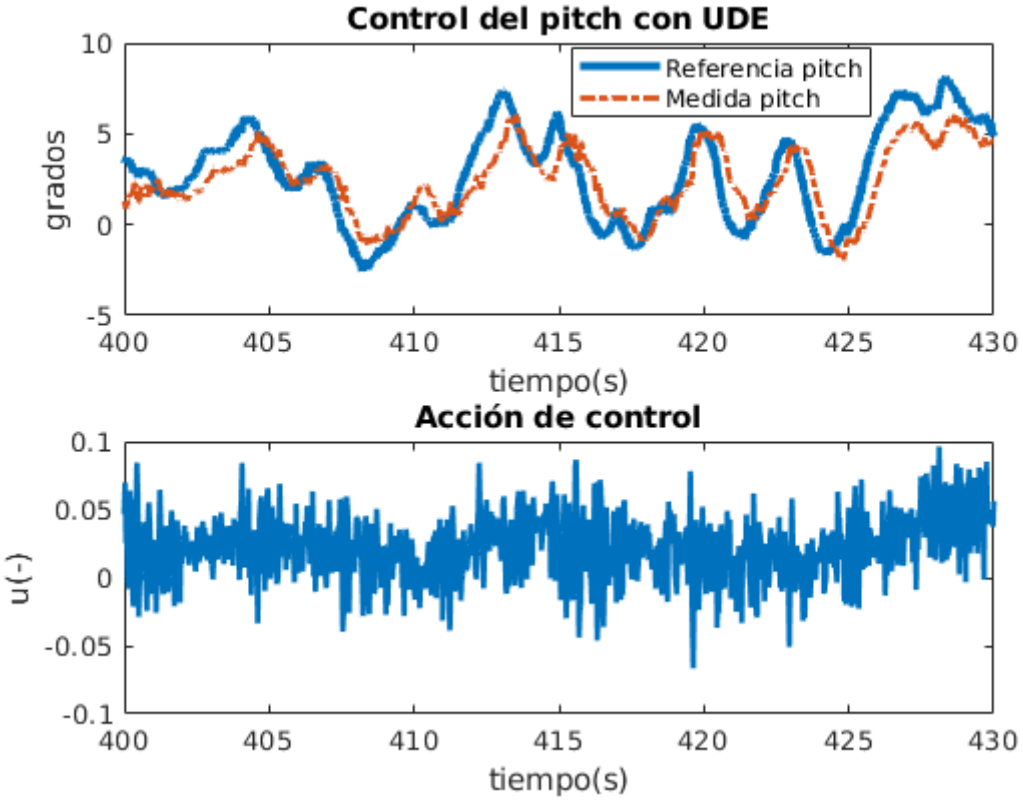


Figura 59: Control del pitch con UDE (vuelo real)

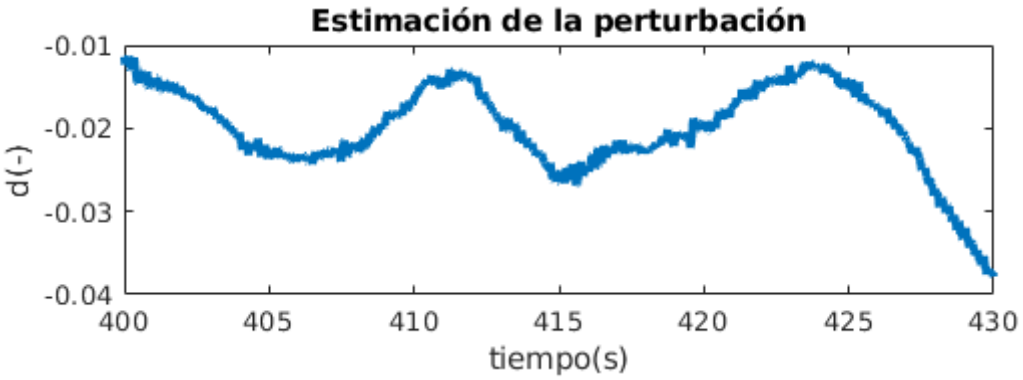


Figura 60: Estimación de la perturbación con UDE (vuelo real)

Como se puede observar a raíz de los resultados obtenidos, el control de la posición en x, y, z, a pesar de no ser exacto como en la simulación, se muestra realmente satisfactorio con una precisión de entre ± 20-30 cm, lo cual no habría sido posible de no contar con un sistema RTK-GPS como se explicaba al comienzo de este capítulo. En el caso de la velocidad, se puede apreciar cómo la velocidad en el eje z es realmente precisa, y en los ejes x e y se obtiene una precisión de ±0.15 m/s aproximadamente.

Si atendemos al control del pitch con el UDE en la figura 59, vemos cómo la medida sigue la estela de la referencia adecuadamente sin ningún pico indeseado. A la vista de esta gráfica, con el objetivo de obtener un control todavía más ajustado y preciso se planteó aumentar ligeramente la velocidad del

PD con tal de que la medida y la referencia estuviesen más solapadas, que sería lo ideal. Sin embargo, esto derivó en un comportamiento propicio a la inestabilidad del Quadrotor, por lo que no fue posible llevarlo a cabo. A pesar de ello, los resultados se encuentran dentro de lo que cabría esperar.

Finalmente, de manera ilustrativa, se muestra en 3D en la figura 61 el resultado de un experimento en el que se pretendía que el dron se quedase quieto en una altura de 5 metros, y que luego ascendiese hasta los 10 metros y de nuevo permaneciese allí. En este caso se observa una clara tendencia del dron a desviarse en el eje x, la cual era la dirección en la que le incidía el viento. Como ya se ha visto en el capítulo anterior, el UDE únicamente es capaz de rechazar a la perfección perturbaciones de naturaleza constante, dando como resultados en el resto de los casos una mejora en el comportamiento del Quadrotor, pero sin llegar a rechazar la perturbación del todo. Dicha desviación se debe, por tanto, a la acción del viento variable sobre el dron. A pesar de ello, la precisión que se ha obtenido es de ± 50 cm, la cual resulta bastante aceptable.

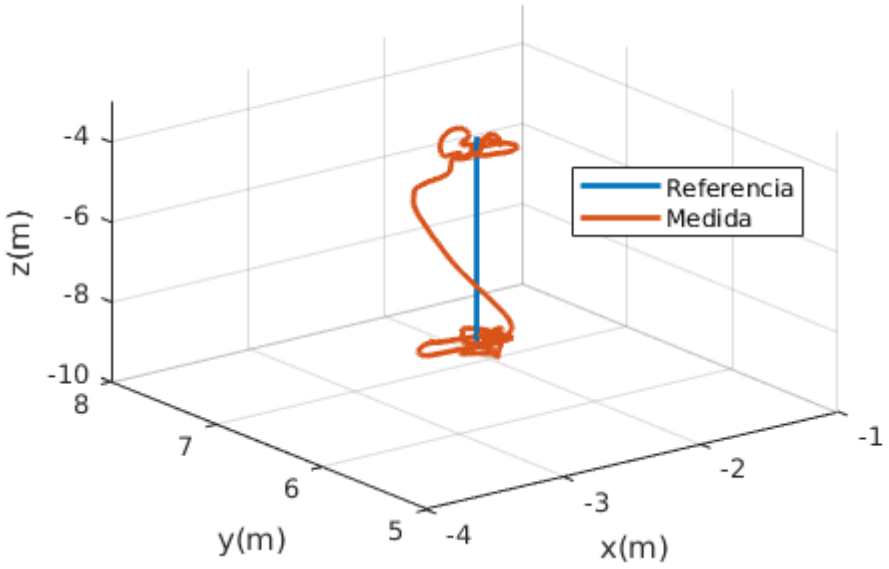


Figura 61: Control de posición en gráfica 3D

Capítulo 9

Conclusiones y trabajos futuros

En este capítulo final se extraen todas las conclusiones que se han podido obtener a raíz del trabajo realizado, a la vez que se proponen diferentes modos de continuar y mejorar el proyecto que aquí se describe.

Como se indica en los primeros capítulos de esta redacción, los drones son aeronaves sin tripulación que están cobrando cada vez mayor relevancia en la vida cotidiana. Para manejarlos, se recurre a algoritmos de control basados en los PID, estrategia que, a pesar de tener muchas ventajas, también acarrea algunos inconvenientes difíciles de obviar.

A lo largo del Capítulo 7, se ha hecho todo un recorrido entre diferentes técnicas de control de orientación. Se ha mostrado la aparente necesidad de incluir acción integral al controlador inicial PD del que se partía, ya que no era capaz de rechazar las perturbaciones. Sin embargo, como indica la figura 62, se ha visto que a la hora de incluir acción integral se introducen también sobre-oscilaciones que podrían llegar a causar la inestabilidad del Quadrotor. De ello, surge la necesidad de ahondar en el estudio de nuevas técnicas capaces de permitir error nulo en el régimen permanente, pero sin llegar a introducir las indeseadas sobre-oscilaciones ya mencionadas. Fruto de este estudio, surgen estrategias de control como el *Uncertainity and Disturbance Estimator* (UDE), o el *Extended State Observer* (ESO). Estas son capaces de predecir en tiempos de establecimiento muy pequeños perturbaciones constantes que se le introducen al sistema y rechazarlas correctamente sin sobre-oscilación alguna, como se muestra en las figuras 63 y 64. Así mismo, también podemos concluir a la vista de estas gráficas que no existe ninguna diferencia en lo que respecta a rendimiento y desempeño entre ambas estrategias. Su diferencia radica a la hora de implementarlas, dónde para el UDE será necesario un conocimiento completo de las diferentes variables que componen el estado del sistema, mientras que el ESO podrá ser implementado sin necesidad de conocer dichas variables, las cuales, además, serán estimadas por el ESO junto con la perturbación de manera extremadamente precisa.

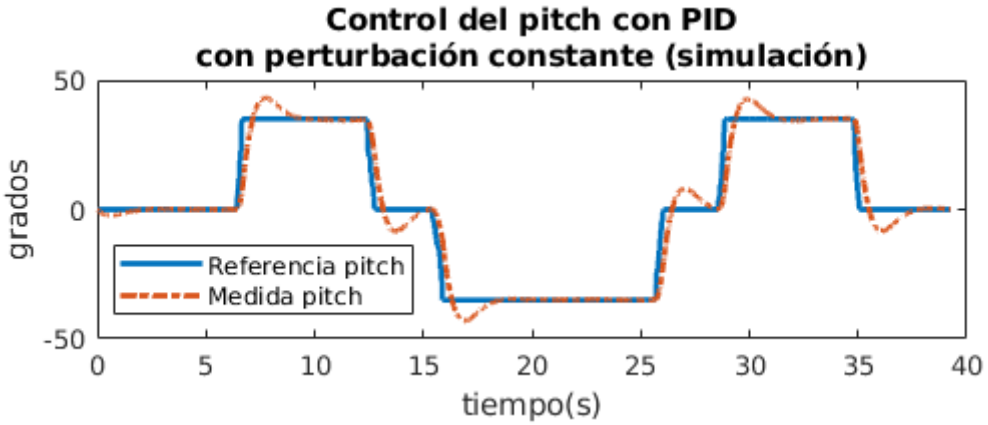


Figura 62: Control con PID

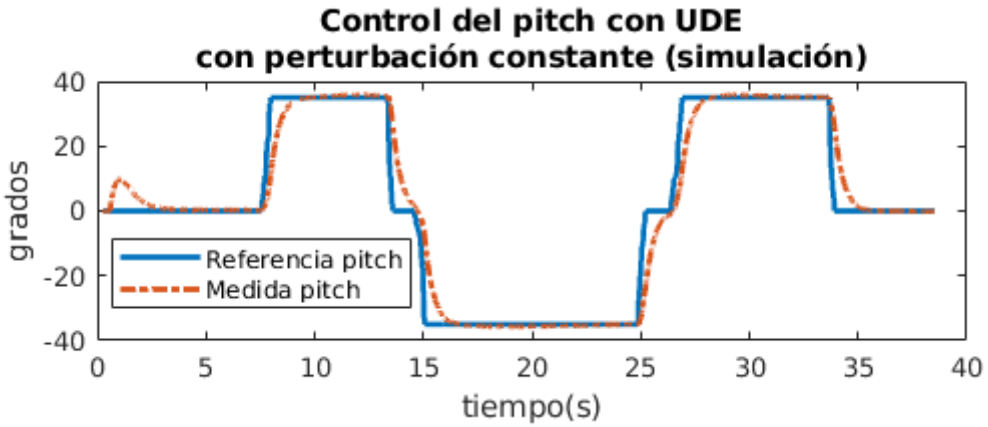


Figura 63: Control con PD + UDE

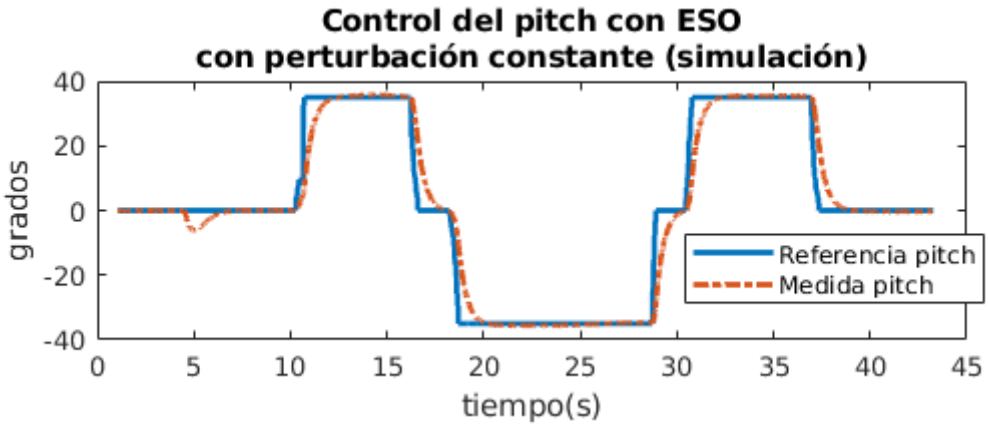


Figura 64: Control con PD + ESO

Sin embargo, durante los vuelos reales se ha observado cómo el dron era incapaz de mantener la posición, aunque no se le diese ninguna indicación de moverse por radio. Es por ello, que una vez solventado el problema de la estabilidad del Quadrotor y su control de orientación, se ha planteado un control de posición mediante un sistema RTK-GPS. La elección del GPS como sistema de posicionamiento ha sido debida a que, aunque conscientes de sus potenciales inconvenientes en cuanto a precisión se refiere, este también presenta numerosas ventajas. Entre ellas cabe destacar su fácil implementación, pudiendo aprovechar algoritmos ya desarrollados para el control de orientación; su bajo coste; y el hecho de que permita volar el Quadrotor en exteriores con un seguimiento de sus coordenadas.

De este modo, se han realizado numerosos vuelos y se ha podido concluir que, a la luz de los resultados obtenidos, el dron es capaz de mantener la posición cuando así se le indica con un error de precisión no mayor de medio metro, a la vez que rechaza diversas perturbaciones de manera bastante aproximada, tratándose estas principalmente de ráfagas de viento en diferentes direcciones. Sin embargo, es menester recordar que en lo que respecta al control de posición, la tarea ha consistido en añadir un tercer nivel de control al código que viene implementado por defecto en el Pixhawk, con el fin de obtener un mejor rechazo de perturbaciones con las nuevas técnicas de control desarrolladas.

Por ello se propone como trabajo futuro un desarrollo completo de este tipo de control, así como la mejora en el ajuste de los parámetros con el fin de obtener un desempeño más rápido a la vez que preciso. Otra posible función a implementar más adelante sería la de realizar un control de posición basado en *waypoints*, el cual está cobrando cada vez mayor importancia.

Así mismo, otra de las tareas que se podría plantear en un futuro sería la de implementar un control de posición basado en sistemas de flujo óptico, los cuales son muy usados en el ámbito del Quadrotor para vuelos en interiores. De hecho, uno de los objetivos que se planteó inicialmente con la realización de este trabajo fue la de implementar dicho control de posición a la vez que el del RTK-GPS. Sin embargo, por falta de tiempo se decidió descartarlo, permitiendo tratar con mayor profundidad cada técnica de control que aquí se desarrolla.

Finalmente, podemos concluir pues, que el Pixhawk es una herramienta muy versátil en lo que a RPAS se refiere. Su adaptabilidad a la hora de comunicarse con diferentes programas y sistemas ha facilitado la implementación de todos los algoritmos de control y la comprobación de su correcto funcionamiento gracias al simulador *jMAVSim*. Del mismo modo, gracias a su código abierto y multitud de funciones, ha hecho posible realizar todo un estudio a través de diversas técnicas de control con diferentes grados de complejidad, lo cual ha permitido, en definitiva, adentrarse en el mundo de la automática y control industrial.

Bibliografía

[1] (Página web) Home – Pixhawk Flight Controller Hardware Project, < https://pixhawk.org/ >.

[2] Castillo, A., Sanz, R., Garcia, P., & Albertos, P. (2016, August). A quaternion-based and active disturbance rejection attitude control for quadrotor. In *Information and Automation (ICIA), 2016 IEEE International Conference on* (pp. 240-245). IEEE.

[3] European RPAS Steering Group, *Roadmap for the integration of civil Remotely–Piloted Aircraft Systems into the European Aviation System*, junio 2013.

[4] España. Real Decreto 1036/2017, de 15 de diciembre, por el que se regula la utilización civil de las aeronaves pilotadas por control remoto. Boletín Oficial del Estado, 15 de diciembre de 2017, núm. 316, pp. 129609 a 129641

[5] Nemec, D., Janota, A., Hruboš, M., & Šimák, V. (2016). Intelligent real-time MEMS sensor fusion and calibration. *IEEE Sensors Journal*, *16*(19), 7150-7160.

[6] McCarron, B. (2013). Low-cost imu implementation via sensor fusion algorithms in the arduino environment.

[7] Randal W. Beard*, Quadrotor Dynamics and Control*. Brigham Young University, 2008.

[8] Cazorla, D. M. Modelado dinámico y diseño de estrategia de control mediante estimadores para el vuelo autónomo de un quadrotor.

[9] Mahony, R., Kumar, V., & Corke, P. (2012). Multirotor aerial vehicles. *IEEE Robotics and Automation magazine*, *20*(32).

[10] Castillo, P., Lozano, R., & Dzul, A. E. (2006). *Modelling and control of mini-flying machines* (pp. 39-59). Physica-Verlag.

[11] (Página web) DJI – The Future Of Possible, < https://www.dji.com/es >.

[12] (Página web) u-blox, < https://www.u-blox.com/en >.

[13] (Página web) GitHub, < https://github.com/PX4/Firmware >.

[14] (Página web) QGC – QgroundControl – Drone Control, < http://qgroundcontrol.com/ >.

[15] Vicente Balaguer, *Diseño e Implementación de una plataforma para la validación segura de algoritmos de control avanzados en vehículos aéreos no tripulados tipo Quadrotors, Universidad Politécnica de Valencia, 2017.*

[16] Albertos, P., Sanz, R., & Garcia, P. (2015, April). Disturbance rejection: A central issue in process control. In *Systems and Control (ICSC), 2015 4th International Conference on* (pp. 1-8). IEEE.

[17] Sanz, R., Garcia, P., & Albertos, P. (2015, July). Active disturbance rejection by state feedback: Experimental validation in a 3-DOF quadrotor platform. In *Society of Instrument and Control Engineers of Japan (SICE), 2015 54th Annual Conference of the* (pp. 794-799). IEEE.

[18] Geoffrey Blewitt, *Basics of the GPS thecnique: Observation Equations*, Department of Geomatics, University of Newcastle.

[19] D. Alazard, *Introduction to Kalman Filtering*, septiembre 2005.

Parte II

Presupuesto

Índice del Presupuesto

Índice de Tablas

1. Introducción

En la segunda parte de este documento se va a obtener el presupuesto del proyecto denominado “Diseño e implementación de un algoritmo de control avanzado para la estabilización de un Quadrotor basado en el autopiloto Pixhawk y un sistema de posicionamiento RTK-GPS”. Para ello, por una parte, se definen los precios de mano de obra, materiales y maquinaria; y por otra, se detallan las diferentes unidades de obra de las que se compone este trabajo.

2. Unidades de obra

Las diferentes unidades de obra en las que se descompone este proyecto siguen la estructura de los capítulos redactados en la memoria de este documento:

- Construcción del Quadrotor
- Familiarización con el software y el entorno de simulación
- Desarrollo e Implementación de las leyes de control de orientación
- Desarrollo e Implementación de la ley de control de posición mediante GPS

3. Cuadro de precios de mano de obra

Para la realización de este proyecto se ha tenido en cuenta la intervención de un graduado en Ingeniería en Tecnologías Industriales, así como la intermitente ayuda de un técnico de laboratorio.

CUADRO DE PRECIOS DE LA MANO DE OBRA					
Código	Empleado	Salario (€/mes)	Horas diarias	Días laborales	Salario (€/h)
MO. GITI	Graduado en Tecnologías Industriales	3200	8	20	20
MO. TEC	Técnico de laboratorio	1400	8	20	8.75

Tabla 1: Cuadro de precios de la mano de obra

4. Cuadro de precios de materiales

Para la realización de este proyecto han sido necesarios diferentes tipos de materiales. El cuadro de precios de materiales muestra un listado de todos ellos junto con la cantidad necesaria y el precio unitario de cada uno.

CUADRO DE MATERIALES					
Código	**Ud.**	**Material**	**Precio (€)**	**Aporte unitario**	**Total (€)**
MT. CHA	u	Chasis F450 DJI	40	1	40
MO. MOT	u	Motores E310 2312 DJI	60	4	240
MT. VAR	u	Variadores 420 Lite	80	4	320
MT. HEL	u	Hélices Z-Blade	5	4	20
MT. TAB	u	Tabla de plástico	6	1	6
MT. TOR	x20u	Tornillos y tuercas M2.5	1.5	1	1.5
MT. BRI	x100u	Bridas	1.9	1	1.9
MT. CAS	u	Cinta aislante	2.3	1	2.3
MT. BAZ	u	Batería Zippy Compact 5800	59	1	59
MT. BAT	u	Batería Tattu 1300	42	1	42
MT. MCB	u	Medidor de carga de batería	5.99	1	5.99
MT. EF	u	Emisora Futaba T6EX	124.02	1	124.02
MT JSTK	u	Joystick Logitech xtreme	46.21	1	46.21
MT. PIX	u	Pixhawk 1	162	1	162

MT.UBX	x2u	Set de placa C94-M8p + Antena GPS	509	1	509
MT. MTLB	u	Licencia MatLab	1900	1	1900
MT. SD	u	Tarjeta microSD 4 GB	8	1	8

Tabla 2: Cuadro de materiales

5. Cuadro de precios de maquinaria

Para la realización de este cuadro se han tenido en cuenta todos los tipos de maquinaria empleados durante el transcurso del trabajo. Se ha considerado que cada máquina se ha obtenido única y exclusivamente para este proyecto, por lo que el tiempo de vida útil se ha reducido al tiempo de uso.

CUADRO DE MAQUINARIA						
Código	**Ud.**	**Maquinaria**	**Precio (€)**	**Aporte unitario**	**Horas de uso**	**Amortización**
M. OSM	u	Ordenador de sobremesa	418	1	500	0.84
M. OP	u	Ordenador portátil	689	1	140	4.92
MT. CDB	u	Cargador Baterías LiPo	14.6	1	50	0.29
MT. FA	u	Fuente de alimentación 40A	146.85	1	50	2.94
MT. TAL	u	Taladradora con juego de brocas y limas	42	1	16	2.63
MT. SES	u	Soldador de estaño	11.9	1	15	0.8
MT. HER	u	Kit de herramientas	53.23	1	120	0.44

Tabla 3: Cuadro de maquinaria

6. Cuadro de precios unitarios

En el siguiente cuadro se muestra el precio unitario de cada unidad de obra vista en el apartado 2 de este presupuesto:

CUADRO DE PRECIOS UNITARIOS		
Código	**Descripción**	**Precio (€)**
UO1	*CONSTRUCCIÓN DEL QUADROTOR*	**2397.27**
UO2	*FAMILIARIZACIÓN CON EL SOFTWARE Y ENTORNO DE SIMULACIÓN*	**2578.45**
UO3	*DESARROLLO E IMPLEMENTACIÓN DE LAS LEYES DE CONTROL DE ORIENTACIÓN*	**5206.39**
U04	*DESARROLLO E IMPLEMENTACIÓN DE LA LEY DE CONTROL DE POSICIÓN MEDIANTE GPS*	**3611.84**

Tabla 4: Cuadro de precios unitarios

7. Cuadro de precios descompuestos

A continuación, se detalla cada unidad de obra vista en el capítulo anterior:

UO1	**CONSTRUCCIÓN DEL QUADROTOR**				
Código	Ud	Descripción	Precio (€)	Rendimiento	Importe (€)
MO. GITI	h	Graduado en Tecnologías Industriales	20	45	900
MO. TEC	h	Técnico de laboratorio	8.75	2	17.5
MT. TAL	h	Taladradora con juego de brocas y limas	2.625	10	26.25
MT. SES	h	Soldador de estaño	0.8	3	2.4
MT. HER	h	Kit de herramientas	0.44	35	15.4

MT. CHA	u	Chasis F450 DJI	40	1	40
MO. MOT	u	Motores E310 2312 DJI	60	4	240
MT. VAR	u	Variadores 420 Lite	80	4	320
MT. HEL	u	Hélices Z-Blade	5	4	20
MT. TAB	u	Tabla de plástico	6	1	6
MT. TOR	x20u	Tornillos y tuercas M2.5	1.5	0.4	0.6
MT. BRI	x100u	Bridas	1.9	0.08	0.15
MT. CAS	u	Cinta aislante	2.3	0.5	1.15
MT. BAZ	u	Batería Zippy Compact 5800	59	1	59
MT. PIX	u	Pixhawk 1	162	1	162
MT.UBX	x2u	Set de placa C94-M8p + Antena GPS	509	1	509
MT. SD	u	Tarjeta microSD 4 GB	8	1	8
				Total	2327.45
				Costes directos complementarios	0.01	23.27
				Costes indirectos	0.02	46.55
				Total de unidad de obra	**2397.27**

Tabla 5: UO1 - Construcción del Quadrotor

UO2	**FAMILIARIZACIÓN CON EL SOFTWARE Y ENTORNO DE SIMULACIÓN**				
Código	Ud	Descripción	Precio (€)	Rendimiento	Importe (€)
MO. GITI	h	Graduado en Tecnologías Industriales	20	20	400
M. OSM	h	Ordenador de sobremesa	0.84	16	13.44
M. OP	h	Ordenador portátil	4.92	4	19.68

MT. EF	u	Emisora Futaba T6EX	124.02	1	124.02
MT JSTK	u	Joystick Logitech xtreme	46.21	1	46.21
MT. MTLB	u	Licencia MatLab	1900	1	1900
				Total	2503.35
				Costes directos complementarios	0.01	25.03
				Costes indirectos	0.02	50.07
				Total de unidad de obra	**2578.45**

Tabla 6: UO2 – Familiarización con el software y entorno de simulación

UO3	**DESARROLLO E IMPLEMENTACIÓN DE LAS LEYES DE CONTROL DE ORIENTACIÓN**				
Código	Ud	Descripción	Precio (€)	Rendimiento	Importe (€)
MO. GITI	h	Graduado en Tecnologías Industriales	20	240	4800
M. OSM	h	Ordenador de sobremesa	0.84	200	168
MT. CDB	h	Cargador Baterías LiPo	0.29	12	3.48
MT. FA	h	Fuente de alimentación 40A	2.94	12	35.28
MT. BAT	u	Batería Tattu 1300	42	1	42
MT. MCB	u	Medidor de carga de batería	5.99	1	5.99
				Total	5054.75
				Costes directos complementarios	0.01	50.55
				Costes indirectos	0.02	101.10
				Total de unidad de obra	**5206.39**

Tabla 7: U03 – Desarrollo e Implementación de las leyes de control de orientación

UO4	DESARROLLO E IMPLEMENTACIÓN DE LA LEY DE CONTROL DE POSICIÓN MEDIANTE GPS				
Código	Ud	Descripción	Precio (€)	Rendimiento	Importe (€)
MO. GITI	h	Graduado en Tecnologías Industriales	20	160	3200
M. OSM	h	Ordenador de sobremesa	0.84	100	84
M. OP	h	Ordenador portátil	4.92	40	196.8
MT. CDB	h	Cargador Baterías LiPo	0.29	8	2.32
MT. FA	h	Fuente de alimentación 40A	2.94	8	23.52
				Total	3506.64
				Costes directos complementarios	0.01	35.07
				Costes indirectos	0.02	70.13
				Total de unidad de obra	**3611.84**

Tabla 8: UO4 – Desarrollo e Implementación de la ley de control de posición mediante GPS

8. Presupuesto de ejecución material, presupuesto de inversión y presupuesto base de licitación

Código	Concepto	Coste (€)
UO1	*CONSTRUCCIÓN DEL QUADROTOR*	2397.27
UO2	*FAMILIARIZACIÓN CON EL SOFTWARE Y ENTORNO DE SIMULACIÓN*	2578.45
UO3	*DESARROLLO E IMPLEMENTACIÓN DE LAS LEYES DE CONTROL DE ORIENTACIÓN*	5206.39
U04	*DESARROLLO E IMPLEMENTACIÓN DE LA LEY DE CONTROL DE POSICIÓN MEDIANTE GPS*	3611.84
PRESUPUESTO TOTAL DE EJECUCIÓN MATERIAL		**13793.95**
15% Gastos generales		2069.10
6% Beneficio industrial		827.64
PRESUPUESTO TOTAL DE INVERSIÓN		**16690.68**
21% IVA		3505.04
PRESUPUESTO BASE DE LICITACIÓN		**20195.72**

Tabla 9: Presupuesto total

Por tanto, el presupuesto total de ejecución material asciende a 13.793,95 €.

El presupuesto total de inversión asciende a 16.690,68 €.

El presupuesto base de licitación asciende a 20.195,72 €.