Trabajo Fin de Carrera
Grado en Ingeniería en Tecnologías Industriales

Diseño y construcción de drone autónomo para medición de radiación solar

Autor: Martín Hinojosa Pérez
Tutor: José María Maestre Torreblanca

Departamento de Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería

Sevilla, 2018

Trabajo Fin de Carrera
Grado en Ingeniería de Tecnologías Industriales

Diseño y construcción de drone autónomo para medición de radiación solar

Autor:
Martín Hinojosa Pérez

Tutor:
José María Maestre Torreblanca
Profesor Titular

Dep. de Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2019

Trabajo Fin de Carrera: Diseño y construcción de drone autónomo para medición de radiación solar

Autor: Martín Hinojosa Pérez

Tutor: José María Maestre Torreblanca

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2019

El Secretario del Tribunal

Agradecimientos

Me gustaría agradecer a mis padres la oportunidad que se me ha brindado de poder realizar los estudios de ingeniería además de su apoyo ya que sin ellos todo esto no habría sido posible.

Agradecer al resto de mi familia, mi pareja y mis amigos por todo ánimo que me han brindado todo este tiempo.

También agradecer a los profesores y a la comunidad divulgadora científica que me han despertado el interés científico y el entusiasmo del saber.

Finalmente agradecer a mi tutor Jose María Maestre por brindarme la posibilidad de realizar un proyecto tan interesante y relacionado con mis gustos así como su ayuda durante todo el proceso de elaboración de éste.

Martín Hinojosa Pérez

Grado de Ingeniería de Tecnologías Industriales,

Sevilla, 2019

Resumen

En este proyecto, se estudiará e implementará la autonomía a un dispositivo de manera que pueda realizar una tarea sin la necesidad de la supervisión humana.

La tarea a realizar será la de recorrer una superficie de manera autónoma mientras que realiza mediciones de radiación también de una manera autónoma ya que se requieren ciertos requisitos para tomar las medidas de manera correcta en cada punto de interés.

Para abordar dicho problema real, planteado por el laboratorio de automática, se procederá a realizar un estudio acerca de los vehículos aéreos no tripulados, para elaborar un diseño acorde con dicha aplicación y su posterior construcción.

Dicho dispositivo o drone, deberá ser capaz de realizar vuelos a lo largo de la superficie en concreto, mediante una serie de rutas planificadas con anterioridad con la ayuda de sensores que le proporcionarán la información del medio necesaria para tomar las decisiones adecuadas para realizar dicho vuelo de manera autónoma.

Dado que para tomar la medida correcta de la radiación solar que llega a cada zona el sensor debe de estar alineado con el sol, al drone se le instalará una plataforma con servos que mediante una cámara y una Raspberry instalada en el drone, para que realice un tratamiento de imágenes con el fin de poder alinear de manera autónoma en sensor en la posición correcta. De manera adicional, el drone tiene que permanecer en el punto de medida un tiempo ofreciendo la mayor estabilidad posible. En este primer proyecto no se realizarán medidas de radiación solares, pero se estudiará la estructura que alineará dicho sensor con el sol de manera autónoma.

De esta manera, el prototipo descrito en este documento servirá de base para un futuro proyecto con un alcance mucho mayor enfocado a la misma aplicación. Ya que se realiza un estudio básico acerca de un drone autónomo básico y sus componentes para que se tenga en cuenta de cara al futuro drone y por otro lado se construye una estructura con el seguimiento de objetivo implementado de cara a que el sensor pueda funcionar correctamente una vez instalado en la plataforma.

A continuación se expondrá todo el proceso de estudio, diseño, construcción y control de cada una de las partes que forman el sistema en completo. Así como los previos resultados experimentales que en los que se puede reflejar los objetivos alcanzados.

Abstract

In this project, the autonomy of a device will be studied and implemented so that it can carry out a task without the need for human supervision.

The task to be carried out will be to go over through a surface in an autonomous way while the device is taking radiation measurements also in an autonomous way since certain requirements are required to take the measurements correctly at each point of interest.

To tackle this real problem, proposed by the automatic laboratory, a study will be carried out on unmanned aerial vehicles, to develop a design according to this application and its subsequent construction.

The drone should be able to make entire flights along a defined surface, through a series of previously planned routes with the help of sensors that will provide the information of the necessary means to make the appropriate decisions to make the flight in an autonomous way.

Due to take the correct measurement of the solar radiation that reaches each zone, the sensor must be aligned with the sun. In the drone will be installed a platform formed by servos and a camera and a Raspberry, all to perform an image processing in order to be able to align autonomously the sensor in the correct position. Additionally, the drone will have to remain at the measuring point for a while offering the greatest possible stability. In this first project solar radiation measurements will not be made, but the structure that will align the sensor with the sun in an autonomous way will be studied.

In this way, the prototype described in this document will serve as the basis for a future project with a much greater scope focused on the same application. Since a basic study is made about a basic autonomous drone and its components to be taken into an account for the future drone and on the other hand a structure is built with the objective tracking implemented so that the sensor can work correctly once installed on the platform.

Next, the entire process of study, design, construction and control of each of the parts that make up the entire system will be explained. As well as the previous experimental results that can be reflected in the objectives achieved.

Índice

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

Notación

	Ángulos de Euler
ϕ	Giro alrededor del eje X, denotado como Pitch o Cabeceo.
θ	Giro alrededor del eje Y, denotado como Roll o Alabeo.
ψ	Giro alrededor del eje Z, denotado como Yaw o Guiñada.
	Filtro de Kalman
X(k)	Matriz que modela el estado de la iteración k.
Z(k)	Matriz que modela la medición tomada de la iteración k.
P(k)	Matriz que modela la covarianza del error estimado de la iteración k.
Q(k)	Matriz que modela el acoplamiento del ruido del modelo de la iteración k.
R(k)	Matriz que modela el acoplamiento del ruido de los sensores de la iteración k.
K(k)	Matriz que modela la ganancia de Kalman de la iteración k.
	Magnitudes
KV	Kilo volts asignados de un motor brushless. Relaciona la geometría, número de bobinado e imanes.
A / mA	Amperios / Miliamperios.
mAh	Miliamperios por hora.
V	Voltios.
Kg / g / mg	Kilogramo / Gramo / Miligramo.
Km / m / cm / mm	Kilómetro / Metro / Centímetro / Milímetro.
GHz / MHz	Gigahercios / Megahercios.
GB / MB	Gigabytes / Megabytes.
Mbps	Megabytes por segundos.
	PID
Kp	Ganancia del término proporcional.
Ki	Ganancia del término integral.
Kd	Ganancia del término derivativo.
e(k)	Entrada al PID en la iteración k.
S(k)	Salida del PID en la iteración k.
I(k)	Historial acumulado de las entradas hasta la iteración k.
Td	Tiempo calculado en el método Ziegler Nichols.
Ti	Tiempo calculado en el método Ziegler Nichols.
	Tratamiento de imágenes
RGB	Modelo de digitalización y representación de imágenes correspondiente con los elementos rojo, verde y azul.

R	Elemento rojo (Red) del modelo RGB.
G	Elemento verde (Verde) del modelo RGB.
B	Elemento azul (Blue) del modelo RGB.
HSV	Modelo de digitalización y representación de imágenes correspondiente con los elementos matiz, saturación y valor.
H	Elemento matiz (Hue) del modelo HSV.
S	Elemento saturación (Saturation) del modelo HSV.
V	Elemento valor (Value) del modelo HSV.
M	Número de filas de las que se compone una matriz cualquiera.
N	Número de columnas de las que se compone una matriz cualquiera.
x	Número de fila en la matriz correspondiente a la imagen.
y	Número de columna en la matriz correspondiente a la imagen.
r	Radio de circunferencia en píxeles.
A.C.	Antes de Cristo.
D.C.	Referido a un motor, que trabaja en tensión continua.
PWM	Modulación por ancho de pulsos (Pulse width modulation).
CW	Referido a la hélice, régimen de revoluciones en sentido horario.
CCW	Referido a la hélice, régimen de revoluciones en sentido antihorario.
C	Referido a una batería, tasa de descarga.
S	Referido a una batería, número de celdas que compone la batería.
GPIO	Referido a la Raspberry, pin designado con propósito general de entrada o salida (General Purpose Input/Output).
sen	Seno.
cos	Coseno.

1 INTRODUCCIÓN A SISTEMAS AUTÓNOMOS

Inteligencia es la habilidad para adaptarse al cambio

Stephen Hawking, 1942-2018

1.1 Sistema completo

Para desarrollar el sistema completo, podemos dividir el proyecto en una primera etapa de diseño con dos subsistemas que se pueden desarrollar independientemente uno del otro para poder comprobar su eficacia de una manera más rigurosa y así asegurarnos su correcto funcionamiento para un posterior ensamblaje de ambos sistemas en un solo.

Dichos subsistemas suponen los grandes bloques a desarrollar en este documento, y consisten en:

- Construcción y diseño de un drone. En este bloque se definirán los componentes básicos de un drone para estudio en el diseño. Se explicará de una manera general su construcción y finalmente, una vez esté operativo, se sintonizará para que pueda ser capaz de realizar vuelos estáticos.

  Un vuelo estático quiere decir que el drone deberá permanecer en un punto en el espacio definido por una altura, y una posición en el plano XY. Esto quiere decir que no solo deberá controlar los 4 motores que lo forman de manera que sea estable, si no que deberá permanecer en dicho punto espacial a pesar de ráfagas de viento u otros elementos que puedan perturbarlo.

- Elaboración de un sistema que sea capaz de poder analizar el entorno del drone para poder buscar y alinearse con el sol. Dicho sistema estará constituido por una Raspberry y una plataforma donde se aloja la cámara y se mueva acorde a unos servos. De manera experimental se planteará el problema de seguimiento de objeto para una cámara montada en una plataforma de servos y así poder realizar un programa de tratamiento de imágenes junto a un control de servo por PID.

- Se unificarán ambos proyectos en un solo sistema, y se realizarán cambios tanto en la estructura del drone para alojar el sistema de control, como del programa de la Raspberry y de la estructura de la cámara para poder realizar un programa de búsqueda del sol. Como se ha comentado en el resumen, no se realizarán medidas de radicación solar, al alcance del proyecto se limita a diseñar la estructura de control que se encargará de alinear la plataforma alojadora del sensor con el sol.

De manera conclusiva, se comentarán los resultados obtenidos a lo largo del proyecto, así como consideraciones acerca de aspectos mejorables del proyecto o futuras aplicaciones a seguir investigando.

1.2 Motivaciones

La automatización siempre ha estado fuertemente ligada a procesos industriales, donde ha podido suponer una mejora en el tiempo y calidad de ejecución. Sin embargo el avance tecnológico que se está produciendo en nuestra era está provocando que el área de aplicación de automatización se amplíe de manera vertiginosa.

El ámbito de aplicación puede comprender aspectos como:

- La domótica: es una aplicación de la automatización dentro de viviendas. Hoy en día es normal hablar de una vivienda en la que se pueda controlar los intervalos de riego del jardín, o el estado de las luces o las persianas dependiendo de la hora del día. Una de las últimas revoluciones son los asistentes virtuales, sistemas que nos permiten controlar todos estos aspectos mediante comandos de voz, e incluso organizar nuestra agenda. Se trata de un uso de la automatización que se encuentra en auge.
- En el ámbito automovilístico también se están haciendo grandes avances. Se están desarrollando automóviles que son capaces de conducir de manera autónoma bajo una serie de requisitos en las carreteras. Esto es posible gracias a la gran cantidad de sensores que le permiten al sistema conocer el entorno de una manera muy eficiente, tanto de la situación de asfalto u otros elementos de la calzada, como los demás automóviles y civiles a su alrededor.
- Como hobbies o proyecto educativo, la existencias de microcontroladores como "Arduino" o "Raspberry" han permitido al ciudadano de a pie o a sociedades educativas como colegios y universidades, aplicar conceptos para automatizar todo tipo de aplicaciones o tareas en un ámbito muy amplio.

Todas estas nuevas aplicaciones, dejan en evidencia la clara posibilidad de poder automatizar cada vez más aspectos tanto complejos como simples, con una importancia industrial o simplemente educativa.

Sumado a todo esto, se encuentra el gran desarrollo de los vehículos no tripulados o Unmanned Aerial Vehicle (UAV), que han supuesto un uso de éstos en otros ámbitos que no sean únicamente el militar.

La investigación referente a los vehículos no tripulados las encabezó el carácter militar. El primer UAV se construyó en la primera guerra mundial, y consistía en un avión monoplano capaz de transportar 114 kg de cargamento y volar a una distancia de 480 km con motores de 200 CV. Los sistemas aparatosos, complejos y muy pesados propició que solo tuvieran un uso militar, pero poco a poco estas aeronaves han ido cogiendo componentes cada vez más eficientes y pequeños tanto en dimensiones como en peso. Esto ha propiciado que se les pueda encomendar muy diversas tareas con un precio enormemente menor, y así alcanzar el uso civil con una serie de leyes que legislan su uso y que se encuentran en plena actualización acorde con este desarrollo tecnológico [1].

En esta tesitura, aparecieron empresas con varios tipos de propósitos que investigaron y desarrollaron nuevos conceptos a los que aplicar a estos nuevos tipos de drones mucho más asequibles y baratos.

Por numerar algunos [2]:

- En el ámbito cinematográfico y la fotografía: Anteriormente la mejor manera de obtener tomas aéreas era mediante grandes cámaras a bordo de un helicóptero. Sin embargo ahora se pueden filmar las mismas escenas con mucha más eficiencia y calidad, con un coste muchísimo mejor desde drones. Derivado de este ámbito, también se han desarrollado drones que son capaces de ayudar realizar un estudio del terreno, a ayudar en tareas de búsqueda, de acceder a sitios donde un humano se encontraría comprometido (zonas de radiación por ejemplo), o a encontrar incendios en bosques y ayudar a extinguirlos en condiciones adversas como la noche.
- En el transporte: Con las condiciones adecuadas tanto de regulación de espacio aéreo y del material transportado, se ha abierto un mercado que aporta a sus clientes una entrega rápida a través de drones. Éste es el uso que por ejemplo la empresa Amazon ha desarrollado para sus drones en ciertas partes del mundo. También ha sido utilizado por socorristas para poder llevar de manera inmediata un elemento flotante a gente que se encuentra en problemas.
- En el ocio: La posibilidad de pilotar un UAV mediante unas gafas de realidad aumentada, y poder pilotarlo desde la perspectiva del aparato mismo, ha abierto un nuevo tipo de carreras en las que los pilotos tienen que realizar lo más rápido posible un circuito que consiste tanto en giros en el eje horizontal, como el vertical. También se han desarrollado drones con control por gestos para poder lanzarlos al aire y mediante órdenes simples tomar fotos de nosotros mismos desde el aire, o con la posibilidad de que te filmen mientras escalas.
- En la educación: Un drone consiste en un complejo sistema que congrega muchos tipos de tecnologías y estudios, que lo hace un elemento ideal para su estudio y desarrollo continuo.

> Muchos de los avances que esta tecnología ofrece en sus actualizaciones cada día, viene de organizaciones universitarias donde estudiantes publican estudios o participan activamente en ellos.

Todo esto, me ha motivado a querer hacer un proyecto en el que unifique estas dos grandes tecnologías que se están desarrollando tan rápidamente y así recalcar la importancia que hoy en día tienen estas tecnologías en aplicaciones tan lejanas a las que se diseñaron en una primera instancia que era la militar.

Particularmente en el departamento de automatización se han embarcado en proyecto en el cual quieren diseñar un sistema que de manera autónoma pueda comprobar la radiación solar que llega a cada una de las zonas de espejos de una planta solar y así poder tomar medida en contra de los gradientes de temperatura creados por las sombras ocasionales de las nubes. El sensor necesario para la medición resulta muy caro, y la superficie total es muy extensa por lo que ha optado por diseñar un vehículo no tripulado que capaz de recorrer toda las instalaciones mientras que realiza medidas.

En este proyecto se realizará un primer prototipo en el que se unificarán las tecnologías referentes a los drones, con sistemas autónomos y de control, para abordar este tema en cuestión.

No puedo acabar este capítulo sin mencionar la tremenda similitud que le veo al concepto de la web, concepto inicialmente militar y que se ha vuelto imprescindible en todo el mundo.

1.3 Definición de sistemas autónomos

Según la definición dada por la Real Academia Española, un sistema es un conjunto de cosas que relacionadas entre sí ordenadamente contribuyen a determinado objeto. Por otro lado, la definición del adjetivo autónomo es el que trabaja por cuenta propia. Uniendo ideas, un sistema autónomo consistirá en un conjunto de cosas que se relacionan ordenadamente contribuyendo a un objeto de manera propia.

Esta definición aplicada al campo de la ingeniería nos deja muchos tipos de interpretaciones, ya que el sistema puede consistir en una máquina, en un controlador, en un proceso… Aunque la idea de trabajar por cuenta propia queda bastante clara, consiste en que el sistema constituya el objeto de manera autosuficiente, sin que el usuario supervise ni aporte información al sistema.

El libro de Benjamin C. Kuo, Sistemas de control automático [3], nos introduce la definición de sistemas de control automático de una manera que podemos entender de mejor manera el gran ámbito que abarca esta definición. Para ello partimos de unas necesidades en nuestra vida diaria, con unos objetivos a cumplir:

- Se requiere el poder controlar la temperatura y humedad de casa y edificios para poder asegurar una comodidad.
- Se requiere que un automóvil o aeronave ofrezca la posibilidad de usarlo de manera segura y exacta para poder ir de un lugar a otro.
- En una industria, un proceso de manufactura tiene una gran cantidad de objetivos para que los productos y así poder elaborarlos lo más rápido, de la manera más eficiente.

Todos estos objetivos pueden ser llevados a cabo por un ser humano, que dispone de la capacidad de razonar y llegar a una decisión en función de unos parámetros. Por ejemplo, un corredor de una carrera de 100 metros razona y lleva a cabo sus acciones acorde al objetivo de hacer los 100 metros en el menor tiempo posible, pero por otro lado el corredor de una maratón no solo tiene que llevar a cabo el objetivo de llegar el primero a la línea de meta, sino que tiene también el objetivo de controlar el consumo de energía y desarrollar la mejor estrategia para cumplir el requisito anterior.

Entonces si podemos analizar el medio físico para poder adquirir los datos de las variables que afectan a la toma de decisión de una tarea, que suele ser rutinaria, podemos diseñar un sistema de control que esté implementada en el sistema o externa al sistema con posibilidad de influir en él.

Dicho sistema de control consta de un sistema completo en sí y tiene un objetivo o meta a alcanzar. Tendrá una manera de afectar al mundo físico a través de una salida, y desde la entrada se le administrará todos los datos que influyen en la toma de decisiones para que el sistema de control desarrolle la mejor estrategia para llevar a cabo el objetivo. Por ello es de vital importancia estudiar el proceso para saber de la manera más

aproximada como se puede modelar el sistema a control, además de englobar todos los elementos que lo involucran.

Como la mayoría de las veces no es posible tener un modelo del sistema totalmente exacto al real, el sistema de control deberá ser capaz de sobrellevar dichas diferencias, incluso interferencias externas que puedan perturbar el proceso.

Para reflejar todo esto, analizaremos el caso de 2 tipos de sistemas autónomos. Uno de ellos el sistema de control será externo al sistema a controlar, y en otro formará parte del sistema.

1. Supongamos el sistema formado por un cuarto de baño de un establecimiento. Disponemos dentro del cuarto una bombilla que ilumina el cuarto y un extractor que se encarga de recircular el aire del habitáculo. Como hay muchos tipos de clientes (los que son educados y los que no), no tenemos la certeza de que se esté dando un buen uso de los interruptores de la luz y del extractor. Por ello instalamos un sistema de control que se dedica a gestionar los momentos en los que éstos están encendidos.

   Para ello el sistema de control se conecta con los interruptores por relés para poder influir sobre ellos, y se le conecta un sensor de movimiento que se instala en la esquina del cuarto de manera que puede notar movimiento en todo el baño.

   La estrategia que sigue dicho sistema de control en este caso es muy simple, una vez que detecte movimiento en el baño, enciende tanto la luz como el extractor y los deja encendido mientras detecte movimiento. Sin embargo pasado un determinado tiempo desde la última vez que detectó movimiento, apagará las luces y el extracto.

   De esta manera, consigue de manera autónoma que se aproveche en gran medida la energía que se consume en el baño.

   En este caso, el control es muy simple. Se considera un control de lazo abierto ya que la iluminación del cuarto por ejemplo, que es una salida del sistema de control, no es objeto de estudio para el control. Si por casualidad la bombilla estuviera defectuosa y no iluminara correctamente a cierto voltaje y el sistema de control tuviera la posibilidad de suministrarle más voltaje, en este tipo de control no tendría la manera de realizar la estrategia adecuadamente para mantener el baño bien iluminado y se perdería un poco el objetivo a conseguir.

2. Ahora supongamos el sistema formado por una cámara frigorífica. El sistema está creado para que en el interior se la cámara se alcance cierta temperatura, por lo que en este caso el sistema de control está implementado dentro de la cámara, no es algo externo a ella que se tenga que instalar ya que forma parte de la propia cámara frigorífica.

   La cámara tiene la capacidad a través de un ciclo de Carnot de intercambiar energía con el ambiente en forma de calor mediante unos radiadores que se encuentran a mayor temperatura que el ambiente. Esa salida de energía del sistema se transforma en una salida de calor de la cámara por lo que se enfría. Sin embargo es un proceso que debe de ser controlado ya que el ciclo puede estar en funcionamiento o no.

   En esta tesitura entra el sistema de control. Su objetivo es el de conseguir que dentro de la cámara se alcance una temperatura, pero a su vez es la información que necesita para poder controlarlo. Este tipo de control implemente la realimentación de la información acerca de la temperatura de la cámara para que pueda ser usada como referencia a la hora de calcular cuánto tiempo debe de estar encendido el ciclo para alcanzar la temperatura que quiero.

   Este tipo de control se considera como control de lazo cerrado, y se consigue proporcionando al sistema de control información directa acerca de las consecuencias de su influencia en el objetivo a controlar.

1.4 Historia de los sistemas autónomos

La automatización ha estado presente en toda la historia, incluso desde la antigüedad. Por ejemplo, lejos de considerarla como fantasía o magia, ciertas civilizaciones llegaron a automatizar algunos procesos con la tecnología que disponían.

Fue en Grecia donde nació una gran investigación referente a la automatización, de mano como por ejemplo Ctesibio de Alejandría (300 A.C) que consiguió desarrollar un reloj de cuco de agua, o el gran maestro de la automática Herón de Alejandría (100 A.C).

Herón llegó a diseñar y construir innumerables autómatas basados en la neumática. Llegó hasta crear el primer motor impulsado por vapor aunque sin finalidad práctica: el eolípila. También llegó a diseñar un teatro automatizado, descrito en documentos, donde se describen cómo un sistema de poleas con bolsas de granos que iban intercambiando los granos y así ir moviendo las plataformas con ruedas.

Aunque la invención más destacada de Herón y por la que más se le conoce, fueron las puertas automatizadas de un templo. Cuando el sacerdote quería abrirlas encendía el fuego de la pira, que calentaba un depósito con agua y aire. Por efecto sifón cuando el aire se calentaba y se expandía empujaba el agua hacia otro depósito con unas cadenas. Cuando el peso del agua fuera suficiente esas cadenas junto a unas poleas serían capaces de abrir las puertas. De igual manera, cuando el fuego se extinga y el depósito se enfríe, el agua volvería a su depósito inicial y el contrapeso de la polea cerraría las puertas [4].

Figura 1-1 Motor a vapor de Herón [4]

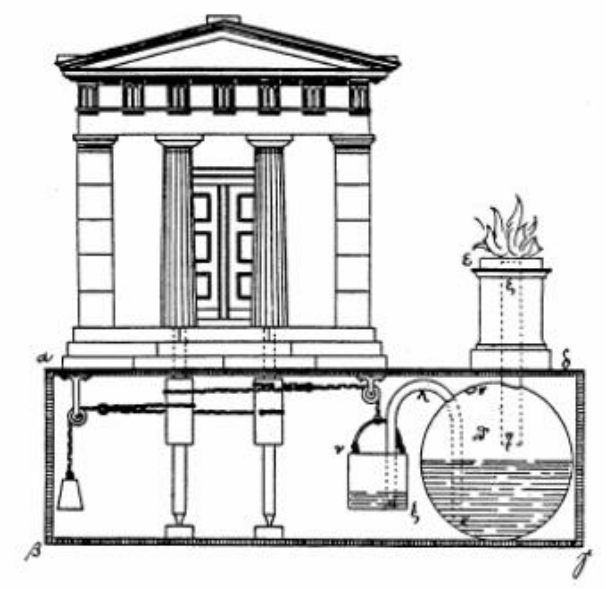


Figura 1-2 Puertas automatizadas del templo de Herón [4]

En España también poseemos un poco de tecnología autónoma de la Edad Antigua, en el faro de la torre de Hércules en A Coruña. Se trata de un faro considerado hoy día patrimonio de la humanidad y que según una investigación de los restos arqueológicos y de planos antes de la rehabilitación, presenta una propuesta acerca del mecanismo utilizado para hacer rotar la luminaria y así no hacer confundir a los marineros la luz del faro con una estrella del cielo.

El mecanismo constaba de una cuerda enrollada alrededor del asiento de la luminaria, en el que por un lado tenía un contrapeso y en el otro un cilindro de madera alojado en un depósito. También había otro depósito conectado al anterior pero con un sifón invertido con la rama ascendente, aún quedan los restos en el faro de los huecos correspondientes a la localización de dichos depósitos. Cuando el agua inundaba el depósito, el cilindro de madera comenzaba a sentir una fuerza de Arquímedes y en el momento en el que superara dicha fuerza a el peso del cilindro y de la fricción del eje rotatorio de la luminaria, procedería a subir y así hacer girar

el eje. Una vez que el agua del depósito secundario conectado a el del cilindro de madera llegara a una altura, se vaciaría automáticamente por el sifón y así conseguiría hacer retornar el cilindro de madera por su propio peso [5].

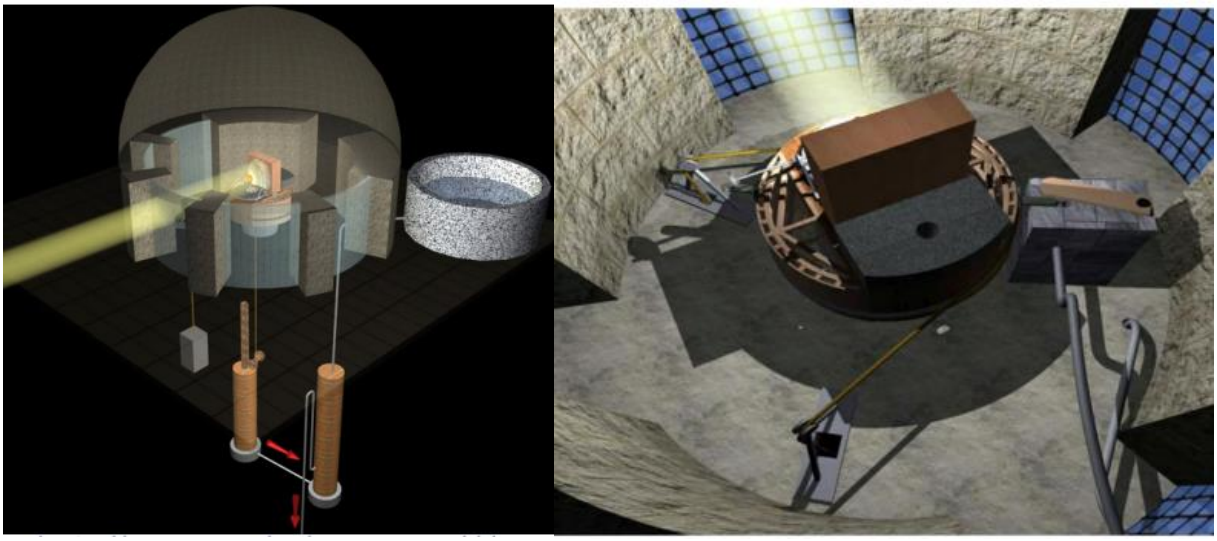


Figura 1-3 Modelo 3D del sistema de iluminaria del faro de Hércules [5]

También se conservan procesos automatizados en España más conocidos que el faro de Hércules, como los molinos de viento. Los molinos de viento eran construcciones que aprovechaban la fuerza del viento mediante unas aspas para mover un eje. Mediante engranajes, ese movimiento conseguían transmitirlo a otro eje donde unas piedras cónicas se movían alrededor de un eje y así poder moler granos. También existieron molinos hidráulicos con un funcionamiento similar [6].

Pero no fue hasta la revolución industrial, cuando empezó de verdad la innovación en el ámbito de la automatización. Un icono de dicha revolución es el ingeniero mecánico James Watt de Escocia. A James no se le atribuye la invención de la máquina de vapor ni del regulador centrífugo, pero llevó a cabo las innovaciones en estas tecnologías que las impulsaron como principal fuente de energía de la época.

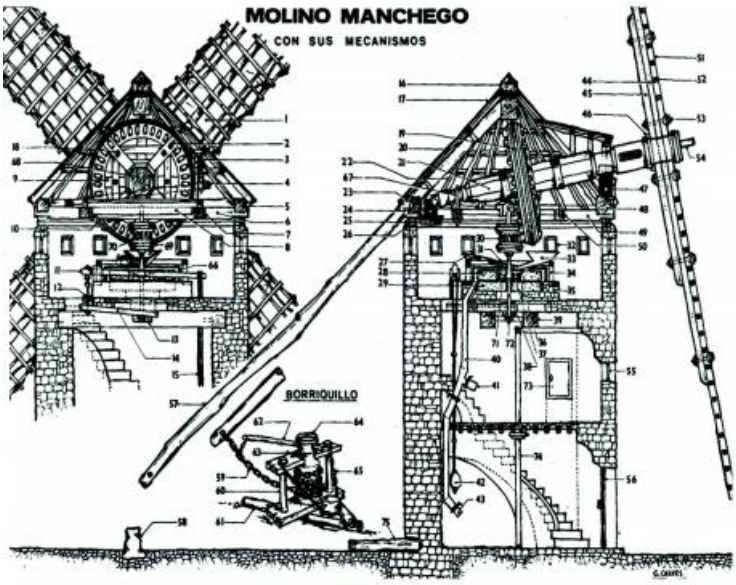


Figura 1-4 Estructura interna de un molino de viento

Respecto a la máquina de vapor, partió de la idea de la máquina de Newcomen (Otra actualización de la máquina de vapor) y solventó la causa de la ineficiencia energética en la que se perdía tres cuartos de la energía producida en calentar el pistón y el émbolo. Para ello desarrolló una cámara de condensación separada que supuso una actualización de la máquina de vapor que la llevó a su uso en muchos ámbitos.

A partir de esta máquina, diseñó un regulador centrífugo para poder controlar la velocidad de giro de manera que ésta permanezca constante. Para ello el regulador debía de conocer la velocidad del eje, pero a su vez su influencia sobre la máquina o salida del regulador, que constaba en una válvula que regulaba el caudal de vapor a presión a la máquina de vapor, modificaba la velocidad. Este tipo de sistemas se denominan control por lazo cerrado ya que introducen la realimentación de la salida sobre el control. Aplicado al regulador, éste actúa sobre el sistema para que se alcance una velocidad constante, pero dicha velocidad también sirve como información que usa el regulador para controlar el sistema.

Mecánicamente, el regulador consistía en un par de masas conectadas a un eje de rotación que se movía a la misma velocidad angular que el eje principal de la máquina de vapor, que por fuerza centrífuga del propio eje pueden desplazarse en el plano vertical mientras rotan junto al eje. Dicho movimiento en el plano vertical puede estar asociado a una válvula, de manera que a menos altura de las masas más abierta se encuentra la válvula, y mientras más altura alcancen las masas más cerrada se encuentra la válvula [7].

Aún existen máquinas que usan este tipo de reguladores, incluso lo podemos ver reflejado en el escudo de la titulación del grado de las tecnologías industriales.

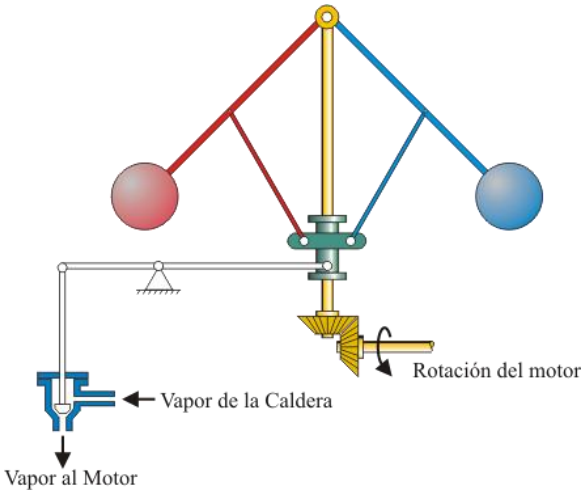


Figura 1-5 Esquema del regulador de Watt

Figura 1-6 Escudo del grado de ingeniería en tecnologías industriales

En la segunda revolución industrial, junto al desarrollo de las tecnologías eléctricas empezaron a mostrar un nuevo abanico de posibilidades para la automatización. Inventos como la electricidad como tecnología energética, o las turbinas de gas para producirla fueron claves para empezar una innovación sin precedentes referentes a autómatas para controlar cada vez más maquinas que se iban desarrollando.

Más adelante se fueron dando grandes descubrimientos que consolidaron la automación tal y como la conocemos en la actualidad:

- El desarrollo de la electrónica a principios del siglo XX.
- El desarrollo de los semiconductores a partir de los años 50.
- Primeros autómatas programables a partir de los años 60.
- Desarrollo de microprocesadores a partir de los años 70.

Actualmente, disponemos de todo tipos de autómatas. Estos son compactos y sencillos para ser usados en todo tipo de aplicaciones, con una gran posibilidad para la portabilidad o modularidad y de ampliación [8].

2 DESCRIPCIÓN DEL SISTEMA: DRONE Y RASPBERRY

2.1 Drone

Para adquirir el conocimiento necesario para poder entender cómo funciona un drone y qué papel juega cada uno de sus elementos primeramente se hace un estudio previo y así entender dichas cuestiones. Para ello se ha consultado [9], [10], [11] y [12]. Dicho conocimiento se verá reflejado más adelante donde se expondrá la elección de cada componente del drone así como su funcionamiento.

Un primero estudio de análisis puede consistir en elegir el tipo de dron. Para ello se explicará las 2 categorías de drones que existen.

Por una parte tenemos los drones de ala fija. Tienen un diseño similar a la mayoría de las aeronaves donde la aerodinámica toma un papel muy importante, ya que sirve para que se genere una fuerza sustentadora y así permanecer en el aire. Por ello están diseñados para realizar vuelos de altura a grandes velocidades pero sin una gran capacidad de carga ni de maniobrabilidad. En estos drones, los motores se colocan de modo que el eje del motor coincida con la dirección en la que va el drone. Para maniobrar realizan cambios en la inclinación de aletas y así conseguir un empuje lateral para ir cambiando de dirección. En estos tipos de drones incluso es factible usar motores de combustión para conseguir el empuje y así alcanzar mayores velocidades.

El otro gran tipo de drones son los de ala rotativa, y de los muchos tipos que existen se hablará de lo multirotores. Estos drones tienen la gran diferencia respecto a los de ala fija en la manera en la que se sustentan en el aire. Generan una sustentación a través de las hélices de sus rotores colocados en vertical. Existen subtipos dependiendo de la cantidad de rotores que lo formen:

- 3 motores: tricópteros.
- 4 motores: cuadricópteros.
- 6 motores: hexacópteros.
- 8 motores: octacópteros.

Esta manera de sustentarse en el aire les brinda a estos drones les brinda la capacidad de poder realizar despegues verticales, y obtienen una inmensa mayor maniobrabilidad que los de ala fija. Son capaces de volar a gran altura, pero dada su poca autonomía debida a la gran demanda de energía por parte de los motores, estos drones suelen usarse para volar a poca altura o para cargar con pesos ligeros.

Otra característica importante y muy determinante para su elección es que el drone de ala rotativa tiene la capacidad de poder realizar vuelos estáticos. Vuelo imposible de realizar para el de tipo ala fija ya que necesita cierta velocidad para generar sostenibilidad en el aire.

Debido a la gran importancia de la estabilidad de este proyecto, se ha elegido el modelo de drone formado por 4 motores ya que es la configuración que muestra una gran estabilidad. Otra opción hubiera sido un drone formado por 6 hélices o incluso 8 motores ya que podría dar incluso más estabilidad. Pero para el alcance de este proyecto un cuadricóptero es suficiente.

2.1.1 Definición y componentes básicos

Una vez elegido el modelo del drone que se va a usar, procederemos a explicar los componentes básicos imprescindibles en este tipo de drones. También se comentarán sus condiciones de uso para justificar su elección en este proyecto.

- **Controladora de Vuelo.**

  La controladora de vuelo se puede considerar como el cerebro de un drone. Es el encargado de gestionar la potencia asignada a cada rotor para conseguir ciertos movimientos del drone o incluso estabilidad. Todo esto mediante modelos matemáticos ya implantados que modelan todo tipo de drones.

  Dado que el estudio del modelo matemático en el que se basa un drone no es objeto de estudio de este proyecto se ha optado por adquirir una controladora de vuelo y así adquirir la posibilidad de un vuelo estático a partir de un sintonizado de variables y PID.

  Para nuestro proyecto se ha elegido la controladora de vuelo Ardupilot APM 2.6. Más adelante se justificará dicha elección y contrastará con otras opciones.

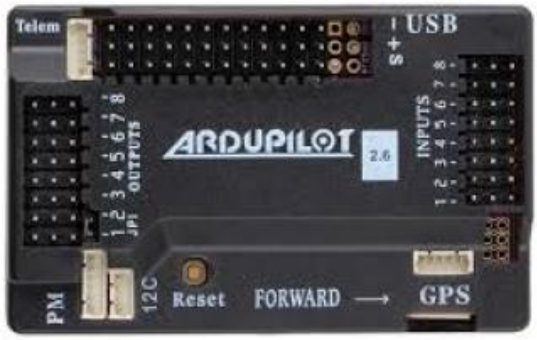


Figura 2-1 Controladora de vuelo Ardupilot

- **Armazón o cuerpo.**

  Consiste en la estructura rígida que le dará forma y consistencia al drone. En ella irán instalados todos los elementos por lo que deberá ser rígida pero ligera. El tipo de cuerpo determina la envergadura del drone así como una idea del peso que lo constituirá.

  Dado que el drone tendrá no solo que levantar el mismo drone, si no que tendrá que llevar consigo el sistema formado por la Raspberry, cámara y batería para la Raspberry se ha optado por un cuerpo estándar del distribuidor de componentes de drones DJI, el cuerpo del drone F450.

  El cuerpo está formado por dos placas de un material de resina en las que están instaladas circuitos eléctricos ideales para el reparto de tensión de la batería. También lo forman cuatro brazos rígidos de plástico en los que irán instalados los motores. Los brazos son rígidos y un poco elásticos para soportar fuerzas o golpes.

Figura 2-2 Componentes que forman el cuerpo del drone F450 de DJI

- **Motores.**

  Los motores usados en este tipos de drones dependen mucho de la carga que tienen que levantar, así como la potencia instantánea que se le demanda. Hay dos tipos de motores para los drones de una envergadura usada en nuestro drone: motores brushless y motores brushed.

Los motores brushed o con escobilla suelen usarse cada vez menos y sólo para levantar cargas muy pequeñas. Esto es así ya que el constante régimen de gro del motor provoca grandes pérdidas en las escobillas en forma de calor, y así reduciendo su eficiencia y su vida útil. La única ventaja notable de los motores brushed consiste en que no necesita de un control electrónico para poder mover el rotor adecuadamente, pero con la tecnología electrónica actual es fácilmente solucionable. Los que usaremos pues son los motores brushless o sin escobilla que ofrecen unas prestaciones más eficientes y adecuadas al peso del drone.

Dentro de los motores brushless, tenemos dos subtipos atendiendo a qué parte del motor realiza el giro: los inrunner y los outrunner. Los outrunner consiguen el movimiento girando una carcasa exterior en la que está alojada el eje rotatorio, mientras que los inrunner poseen un núcleo rotatorio que se encuentra dentro del motor. La principal diferencia en cuanto a prestaciones es la relación velocidad de giro con el torque de éste. Los inrunner pueden alcanzar regímenes de giro bastante mayores pero a costa de un menor torque. Suelen usarse para drones tipo ala fija ya que la propia aeronave se ayuda de la aerodinámica para sustentarse. En nuestro caso elegiremos los outrunners por la capacidad de poder levantar más peso a costa de una mejor velocidad ya que es justo lo que buscamos.

Los motores brushless poseen diferentes características según una nomenclatura, dependiendo de la necesidad. Se nombrará un motor cualquiera que usaremos como modelo para ir describiendo sus características: el motor EMAX 2204-2300KV.

- Primero tenemos la marca del fabricante del motor.
- Las cuatro primeras cifras nos aportan datos acerca de las dimensiones del motor. Las dos primeras es la longitud del motor y las dos siguientes la altura, ambas en centímetros. En el caso del ejemplo tendríamos un motor de Diámetro de 22 centímetros y de altura 4 centímetros.
- Las cuatro últimas cifras junto a la simbología "KV" corresponde a los Kilo Volts del motor, que consiste en las revoluciones por minuto a la que girará el motor cuando se le aplique 1 Voltio de tensión. Es una característica que depende de varios factores como: número de espiras, diámetro del cobre usado en el bobinado, potencia de imanes o la geometría del motor.

Figura 2-3 Motor brushless de la marca DARKSALMON

Los KV generan potencia, que no empuje. El empuje depende de las hélices usadas. Sin embargo los motores de bajo KV son motores con mayores números de espiras y menor consumo, que sacrifican menos revoluciones por minuto por un par mayor. Esto es necesario cuando buscamos motores que puedan junto a las hélices suministrar un gran empuje para un drone de considerables dimensiones y peso.

Al existir tanta variedad de motores, se ha elegido unos estándar y económicos de bajos KV ya que el sistema completo poseerá un gran peso: el DARKSALMON A2212-1000KV.

- **Esc o variadores.**

Al estar tratando con motores brushless outrunners, necesitamos de un dispositivo electrónico capaz de controlar la velocidad del motor. Dichos dispositivo es el Esc (Electronic Speed Controller) o variador, aunque también pueden ser usados para aplicar un freno dinámico.

Se encuentran conectados tanto a la fuente como a la controladora, y a partir de los datos que reciba de la controladora, suministrará un voltaje u otro al motor. Cambiando el orden de la disposición de cables que poseen se puede conseguir que el motor gire en sentido contrario al estipulado.

Los Esc manejan grandes cantidades de corriente, por lo que los fabricantes aconsejan elegir Esc que puedan manejar aproximadamente un 10% o 25% más de la corriente que el motor pueda consumir. Dicho esto, los Esc pueden tener la capacidad de manejar 10 amperios en modelos más pequeños o hasta 200 amperios en modelos grandes.

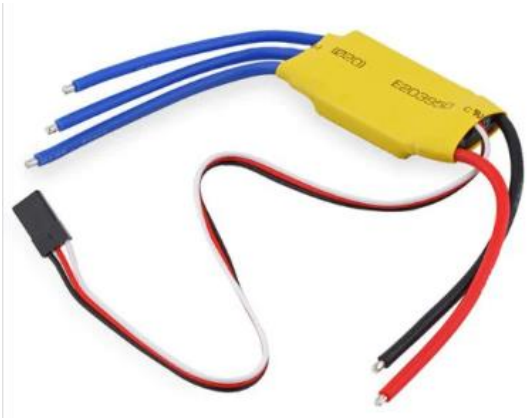


Figura 2-4 Variadores de 30 A

Según el fabricante del motor elegido, el motor consume alrededor de 12 A, por lo que por lo menos debemos elegir unos Esc de hasta 15 A. Al final se ha decantado por elegir unos de 30 A por la compatibilidad económica que presentaban al comprarse junto a los motores. Esc o variadores.

- **Hélices.**

Para que los motores puedan aportar un empuje al drone, deben de ser equipados con un elemento aerodinámico. Dicho elemento es la hélice, y dependiendo de la cantidad del empuje, de la fuerza del empuje, de las características del motor… influyen en la determinación de qué tipo de hélice necesitamos para cada caso.

Para empezar a entender un poco mejor las hélices explicaré la notación usada que la define con la usada en el tipo de hélices que hemos elegido: las Gemfan 8045x2 de Carbono Nylon.

En el número de 4 dígitos podemos conocer dos datos de la Hélice. Los dos primeros números definen la longitud de la hélice en pulgadas y los dos siguientes el grado de inclinación de la hélice. Normalmente los fabricantes de los motores aconsejan una serie de hélices con las que el motor está preparado para funcionar y de hecho según ese criterio he elegido estas hélices.

El tamaño y el grado de inclinación tienen relación con la cantidad de aire que son capaces de mover. A más tamaño más aire mueven, y a más grado de inclinación también mueven más aire. La cantidad de aire que las hélices pueden mover está relacionada con los KV de los motores ya que a mayor cantidad de aire, más fuerza hay que ejercer para poder mover las hélices… Por lo que necesitaremos un motor que en vez de proporcionarnos una gran velocidad de giro nos proporcione un buen par. Es decir un KV bajo.

La última cifra nos indica cuántas palas forman la hélice. En nuestro caso está formado por 2 hélices es normal ver hélices de 3 o 4 palas. Al tener más palas, la hélice pesa más y tiene más empuje ya que el área eficaz aumenta y por tanto empuja más pala. Si a esto le sumamos una carga al dron puede ocasionar que el motor sufra más. Por lo que suelen usarse en motores no hechos para para levantar grandes cargas ya que por el contrario suelen ser mucho más livianos y el motor no sufre tanto en este aspecto, es decir los tengan altos KV.

Por último el material usado, en este caso una mezcla de carbono y nylon. Se trata de una gama media, ya que las de plástico pueden romperse muy fácilmente en contraposición con las de carbono puro que son muy resistentes.

Este tipo de hélices suelen venir en pares ya que puede haber 2 tipos de hélices de un mismo modelo. Las hélices CW y las CCW. La designación hace referencia al sentido de giro en el que la hélice estará moviéndose, siendo las CW para sentido horario y las CWW para sentido antihorario.

Las hélices presentan un problema de fábrica y es que éstas no tienen la seguridad de estar bien balanceadas. Una pala puede pesar algunos miligramos más que la otra pala y aunque parezca una diferencia diminuta, a altas revoluciones pueden suponer grandes vibraciones en el drone o incluso

daños al motor.

Por ello siempre hay que equilibrarlas antes de usarlas. Para ello las hélices se sujetan mediante imanes en una estructura que minimiza cualquier posible rozamiento del eje y así hace visible por medio de la gravedad qué pala pesa más. Para arreglar el defecto se puede limar la pala más pesada o colocar trozos de cinta adhesiva en la pala menos pesada.

Figura 2-5 Par de hélices Gemfan 8045x2 de Carbono Nylon

Figura 2-6 Equilibrador magnético de hélices

- **Batería.**

  Es el elemento que aporta la energía al sistema para poder funcionar. En el mercado hay infinidad de tipos de batería atendiendo a especificaciones de voltaje y corriente, qué tipo de material ha sido utilizado para su elaboración, cantidad de energía que se demanda en un instante, capacidad…

  Primeramente se ha decantado por una batería tipo Lipo, constituidas por litio y polímero. Esta elección ha sido fundamentada por ciertas características:

  - Tienen gran variedad de tamaños y forma, para poder adecuarlas a nuestro sistema. Pero sobre todo son ligeras, característica muy fundamental para la aplicación a la vamos a asignar.
  - A pesar de su reducido tamaño respecto a otras, no sacrifican capacidad de almacenar grandes cantidades de energía. Mientras más autonomía podamos tener mejor.
  - Poseen una tasa de descarga alta. Esta tasa de descarga o "C" consiste en la velocidad con la que la batería se puede descargar de forma segura. Este parámetro es importante pues la demanda de energía por parte de los motores no siempre es constante y en ciertas ráfagas pueden exigir bastante a la batería. Además también hay una tasa de "BURST" que consiste define la capacidad de descarga en
  - Varias baterías pueden agruparse en celdas para formar un de mayor capacidad. Cada celda tiene un voltaje entre 3.7 y 4.2 voltios, y atendiendo a cuántas celdas o "S" componen la batería podemos disponer de un voltaje u otro. Por ejemplo una batería 1S tendría una salida de hasta 4.2 V, una de 2S de 8,4 V…

  La batería elegida es una Turnigy Nano-Tech 3S (12.6 voltios), tiene la capacidad de 3000 mAh, su tasa de descarga es de 25C, y la de BURST 50C.

  Son unos valores normales, dentro de la aplicación que le vamos a dar al drone dentro de un rango económico. Otras baterías de incluso 4S, 50C o muchos más miliamperios hora podría estar mejor ya que aunque fuera más grande, la potencia que suministre puede compensarlo y un así disponer de mejor prestaciones pero son más caras.

  Este tipo de baterías que se encuentran divididas en celdas necesitan de cargadores especiales que realizan cargas equilibradas entre las celdas. Esto significa que no cargan una celda y luego otra ya que puede acarrear una disminución del rendimiento de la batería y de su vida útil.

Figura 2-7 Batería Lipo de Turnigy

- **Módulo de potencia**

  El módulo de potencia es un componente que tiene como función principal suministrar a la controladora una tensión estable de 5 V y una corriente de 2,2 A. La necesidad de este componente se encuentra en la batería, que proporciona unos valores muy diferentes a los que requiere la controladora.

  Además tiene una segunda función, que consiste en la posible monitorización del estado de la batería. Mide el estado de la batería y la envía a la controladora de vuelo. Desde ahí puede ser monitorizada para su uso en componentes como zumbadores y demás.

  Se ha elegido el POWER MODULE APM, por la compatibilidad que tiene con la controladora y por supuesto por suministrar los valores que deseamos. Puede soportar hasta una tensión máxima de 18 V y una corriente de 90 A.

Figura 2-8 Módulo de potencia compatible con Ardupilot

- **Módulo de GPS.**

  El módulo GPS se encarga de conectar con los satélites disponibles e intercambiar dicha información con la controladora de vuelo. Es un componente que se adquiere dependiendo de la aplicación que se le dará al drone ya que por ejemplo en drones de carrera no tiene utilidad.

En nuestro caso es imprescindible ya que es quien suministrará la información necesaria para que el drone pueda permanecer quieto en un punto del plano horizontal y sin sufrir derivas. La elección del modelo es simple ya que solo buscamos que sea compatible con la controladora elegida y que sea económico.

Como adición al módulo GPS, se le instala un soporte vertical al drone para que el módulo permanezca fijo y un poco alejado del drone para evitar interferencias en las medidas del GPS. Este soporte suele ser de aluminio lo que incrementa el peso total del drone, pero la rigidez que proporciona nos ayuda a evitar posibles vibraciones en el módulo y así evitar cierto ruido en la medida.

Figura 2-9 Módulo GPS junto a la estructura de soporte

- **Emisor - Receptor**

  Son los elementos que permiten al usuario transmitir ordenes al drone. Dichas ordenes son comandos simples que sirven para inclinar o hacer rotar al drone, o darle más o menos potencia a los motores por igual entre otras.

  Para ello hay un emisor que controla el usuario remotamente, y un receptor instalado en el drone que se encarga de recibir las órdenes de la emisora por radiofrecuencia y transmitirlas por medio de pulsos en cables físicos a la controladora. Para cada "aspecto" del drone que se puede controlar desde la emisora hay un canal en el recepto, por lo que no se puede transmitir toda la información que queramos.

  Esos componentes elementos son muy complejos y hay una gran variedad de productos en el mercado, he seleccionado uno con al menos 6 canales y otras prestaciones muy económico: el Flysky FS-i6.

  Algunas de sus prestaciones por las cuales lo he elegido son:

  - 6 canales.
  - Acepta modelos Heli (Como el cuadricóptero).
  - Alcance de radiofrecuencia de 2.4 a 2.48 GHz y ancho de banda de 500KHz.
  - Posee una pantalla LCD
  - El receptor posee una capacidad anti-interferencias activa y pasiva.

Figura 2-10 Emisora Flysky Fs i6 y Receptor asociado

- **Soporte anti vibraciones**

  Consiste en dos plataformas, una atornillada al cuerpo del drone, y otra en la que va sujetada la controladora de vuelo. Entre las dos plataformas hay unos elementos de silicona que los une y absorbe todo tipo de vibraciones.

  Esto es beneficioso para la controladora ya que la vibración causada por el movimiento de los motores puede provocar una vibración en el cuerpo del drone. Dicha vibración puede acabar siendo transmitida por el cuerpo hasta la controladora y así interferir en las mediciones del acelerómetro y giroscopio instalados en la controladora.

  Se trata de un elemento no imprescindible, pero si queremos conseguir la mejor estabilidad posible puede ayudarnos a conseguirlo.

Figura 2-11 Plataforma anti vibraciones con juntas de silicona

- **Buzzer-Led**

Consiste en un elemento sonoro y otro visual que nos proporciona información directa del estado de algunas variables del dron.

Por un lado el Buzzer es un dispositivo que al aplicarle una tensión emite un sonido agudo bastante potente, y el Led al igual que el buzzer necesita una tensión para funcionar y así emitir radiación en nuestro espectro visible de color azul y rojo.

Cada elemento dispone de una tarea principal. El buzzer se programa para que responda a una serie de eventos descritos por el fabricante:

Tabla 1 Comportamiento del Módulo Buzzer-Led proporcionado por el fabricante

	APM Status	Indication
Red LED	ARM	solid
	DISARM	flash
Blue LED	GPS FIX	solid
	GPS UN-FIX	flash
Buzzer	ARM	2 Beep
	DISARM	1 Beep
	Low battery	Continue Beep..Beep..
	Crash	Fast continue Beep.Beep.

Se trata de un componente del drone que no es imprescindible, pero aporta una información muy útil directamente al usuario. Se ha elegido el módulo APM INDICATOR V1.0 por la compatibilidad con la controladora.

Figura 2-12 Módulo Buzzer-Led compatible con Ardupilot

En cuanto a los cálculos realizados para la elección general de la composición del drone, podemos destacar los más importantes que han determinado en gran medida el prototipo:

- Empuje. Se trata de la fuerza que ejercen las hélices para poder sustentar el drone en el aire y poder realizar maniobras en pleno vuelo. Para ello el fabricante nos proporciona el empuje máximo que puede realizar cada hélice con el correspondiente motor.

  En nuestro caso, la selección de motor y hélice nos proporciona un empuje de 810 gramos. Si esto lo multiplicamos por los 4 motores que compone el drone, obtenemos un empuje de 3240 gramos. Esto significa que el sistema puede llegar a sustentar en el aire un peso total de hasta 3240 gramos, sin embargo no sería correcto definir un sistema que alcance dicho peso.

  Dado de que se ha calculado el empuje máximo que puede producir, para asegurar un buen funcionamiento del drone ante cualquier circunstancia el empuje máximo debería considerarse un valor 2 o 3 veces más grande que el peso del sistema completo.

  Dicho esto, se estima que el drone pese alrededor de 2 Kg, por lo que se encuentra entre unos valores aceptables.

- Autonomía. Consiste en el tiempo que puede permanecer el drone funcionando durante un vuelo, ya que en reposo el drone no consume apenas energía comparado con la consumida en vuelo.

  La batería seleccionada dispone de una capacidad de 3000 mAh. Para tener una idea de la posible autonomía de la que se puede disponer se va a calcular el tiempo que tardaría en descargarse la batería en las condiciones en las que los motores se encuentren al 100% de su potencia.

  Según las especificaciones del fabricante, el motor consume un máximo de 13 A por hora. Si multiplicamos dicho consumo por el número de motores, resulta un consumo global de 52 A por hora. Dicho consumo se puede expresar como 866 mA por minuto, por lo que si dividimos los mA de capacidad de la batería entre el consumo obtenemos unos 3.46 minutos de autonomía.

  Sin embargo dicho dato nos da una imagen de la autonomía del drone a plena potencia, sin embargo en la práctica el drone estará alrededor de un 50% de su potencia. Para dichos niveles de potencia las especificaciones del fabricante estima un consumo de 6.8 A por hora por motor. Si realizamos los mismos cálculos realizados anteriormente se obtiene una autonomía de unos 7 minutos. Autonomía aceptable para un primer prototipo.

2.1.2 Controladora de Vuelo

La controladora de vuelo como he explicado en el apartado anterior es el cerebro del drone, es quien ordena a cada motor girar a una revolución específica para así llevar a cabo acciones como estabilizar el drone, o moverse.

El objetivo de este componente será el de proporcionarnos la capacidad de realizar vuelos estáticos y rutas definidas por GPS, por lo que muchas controladoras quedan fuera de la búsqueda. Controladoras a las que no se les pueda suministrar la información del medio necesaria para el vuelo estático, como las correspondientes a los drones de carrera ya que únicamente son capaces de estabilizarse a partir del acelerómetro y giroscopio de la controladora.

En el mercado hay varias controladoras que aún entran en nuestro criterio, por lo que se procederá a hacer una breve comparación entre las posibles candidatas que considero pueden ser las más adecuadas según el objetivo de este proyecto. Dicha información se refleja en la tabla 2, y se ha elaborado con los datos proporcionados por las correspondientes páginas oficiales y los precios de Amazon, además de [13] y [14].

Tabla 2 Comparativa de controladoras de vuelo

	APM 2.6	Pixhawk	Naza M Lite
Procesador	Microcontroladores ATMEGA2560 16-bit	ARM CortexM4 de 32-bit con FPU a 168MHz	NEO-M8N 32-bit
RAM	1MB	2MB	2MB
Sensores	-MPU6000 giroscopio y acelerómetro de 3 ejes -Brújula - Barómetro MEAS MS5611	-MPU6000 giroscopio y acelerómetro de 3 ejes -Giroscopio ST Micro 16-bit -Acelerómetro y compás magnético ST Micro 14 bit -Barómetro MEAS MS5611	-Giroscopio de 3 ejes -Acelerómetro de 3 ejes -Barómetro
Precio	43€	38€	95

Naza es un producto de la empresa DJI, muy destacada en el mercado de drones actual. El modelo ofrece una calidad muy buena para que puedas volar el drone sin problema sin tener que prácticamente configurar nada. El inconveniente es que se trata de un código muy cerrado, no ofrece muchas posibilidades a desarrolladores y es caro.

Por otro lado la APM de Ardupilot consiste en una controladora basada en la plataforma Arduino. Por ello se trata de un modelo totalmente opuesto a la Naza, necesita de una rigurosa configuración de parámetros y permite a los desarrolladores crear todo tipo de proyectos gracias al código abiero, la grán comunidad que la apolla y comparten ideas, capacidad de hasta 8 canales y sus 4 puertos series.

Ha sido una controladora de vuelo muy usada en todo tipo de vehículos aéreos o terrestres, y en el ámbito de la navegación autónoma. Sin embargo con la aparicion de la Pixhawk y otras platarformas se ha dejado de dar soporte y actualizaciones.

Pixhawk es la controladora que sigue a la APM de la misma empresa Ardupilot. Con ello implica llevar el mismo software que su antecesora pero con mejoras y actualizaciones tanto en hardware como software.

Figura 2-13 Pixhawk de la empresa Ardupilot

Figura 2-14 Naza Lite de la empresa DJI

Dado el grán historial que posee APM junto a la gran comunidad en internet donde se puede encontrar todo tipo de información acerca de proyectos u otros datos, me he decidido apostar por la APM a pesar de no tener

un software actualizado a día de hoy.

Decidido ya la controladora de vuelo, se procede a hablar acerca de las características de ésta:

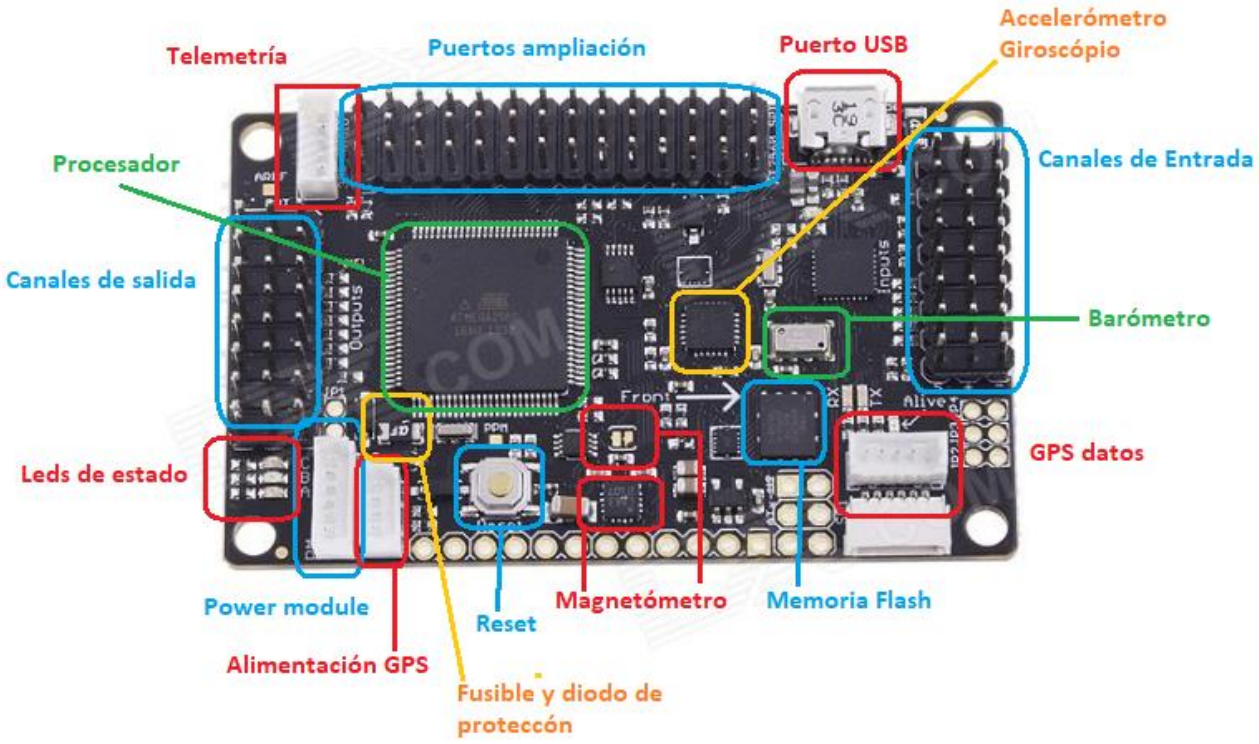


Figura 2-15 Esquema de la distribución de componentes en la APM 2.6

- Telemetría: Es el puerto de comunicación con módulos de telemtría o para comunicación serie. Serán las que usaremos para la comunicación con la Raspberry
- Puertos Ampliación: Son los pines asignados a los dispositivos que puede usar la APM ya sea de entrada o de salida. Pueden ser luces, ultrasonidos, cámaras…
- Puerto USB: Desde aquí podremos conectarnos al ordenador y así configurar parámetros del código de la controladora. También puede ser usado como comunicación.
- Acelerómetro Giróscopio: Intrumento para poder conocer la orientación y inercia del drone.
- Canales de Entradas: Son los canales por los que la controladora recive información de la emisora correspondiente a las órdenes dadas por el usuario mediante el receptor.
- Barómetro: Instrumento capáz de dar información acerca de la altura del drone a través de la lectura de la presión atmosférica. Es capáz de medir con un error de +/- 10 cm.
- GPS datos: Son los pines donde deberá conectarse el módulo de GPS ya que la controladora no dispone de un GPS propio.
- Memoria flash: Donde se guardan los parámetros de configuración.
- Magnetómetro 1 y 2: Instrumento que sirve como brújula, para conocer la orientación del drone en el plano horizontal.
- Fusible y diodo de protección: Medidas de protección para accidentes a la hora de conectar la fuente.
- Reset: Botón para dejar la controladora con la configuración de fábrica.
- Alimentación GPS: Pines correspondiente a la fuente del módulo de GPS.

- Power Module: Pines corresponditentes a los datos del módulo de Power Module.
- Leds de estado: Leds que aportan información del estado del drone.
    - Rojo fijo: drone armado
    - Rojo intermitente: drone desarmado
    - Amarillo intermitente: controladora calibrando
    - Verde: controladora encendida
- Canales de Salidas: pines donde deberán ir conectados los variadores, para el control del los motores.
- Procesador: donde la controladora ejecuta cálculos y demás operaciones.

Efectivamente, se trata de una controladora de vuelo muy completa y versatil, que dispone de los elementos necesarios para poder proporcionarnos un modo de vuelo estático.

Para ello, además de construir y ensamblar el drone, hay que conectar vía USB la controladora al ordenador para poder configurarla en un software que ardupilot proporciona llamado Mission Planner. Al tener la APM 2.6 hay que tener actualizado el programa a la versión 1.3.16 correspondiente a la última actualización correspondiente a la APM.

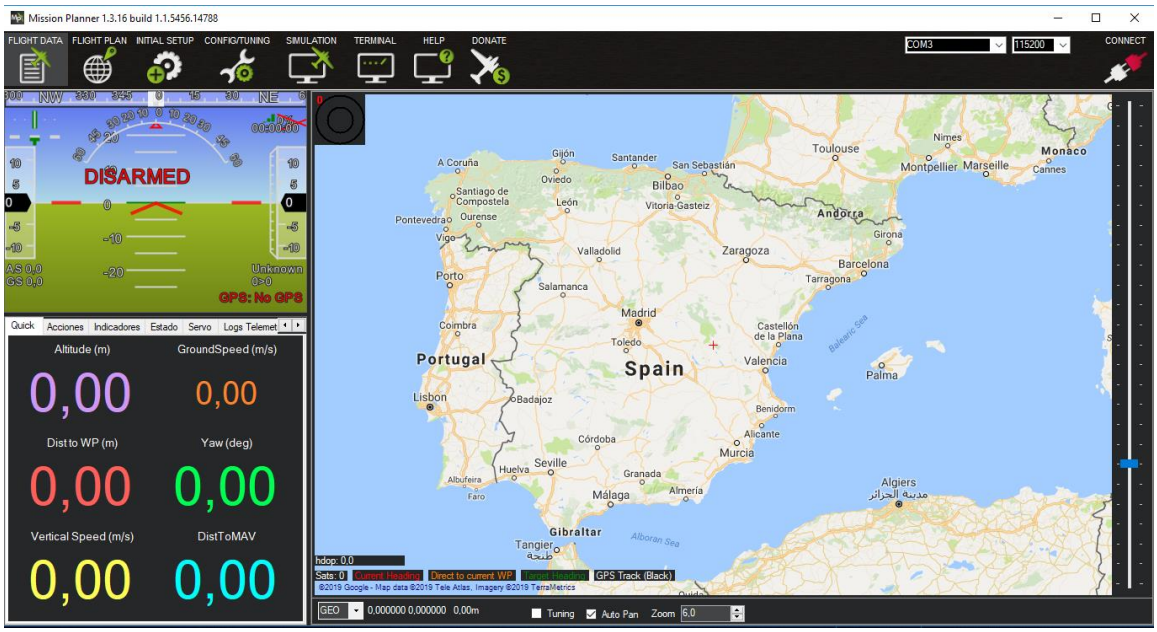


Figura 2-16 Visualización del Software Mission Planner

En este programa se pueden hacer una serie de tareas que voy a destacar de todas las que se pueden usar:

- Calibrar la controladora, es decir podemos definir el tipo de dron que tenemos (tipo de ala, numero de motores), calibrar el acelerómetro, el giroscopio, máximos y mínimos correspondiente a los canales de entrada procedientes de los potenciometros de la emisora…
- Configurar los modos de vuelos que queremos usar, así como los valores de los controladores que influyen en dichos vuelos.
- Configurar una gran lista de parámetros que influyen en el módo de reaccionar el drone a determinados eventos como lecturas del Power Module, comunicación por puerto serie, utilización del buzzer…

Existen muchos parámetros configurables y se puede acceder a la información acerca de ellos en su página web [15], pero nos centraremos en los más importantes para poder conseguir el vuelo estático que deseamos.

2.2 Raspberry

La controladora de vuelo es capaz de estabilizar el drone, y de incluso realizar vuelos semiautónomos a siguiendo una ruta ya definida mediante el GPS. Sin embargo no dispone de la capacidad de poder realizar control de una plataforma de servos de manera autónoma y mucho menos de un tratamiento de imágenes adecuado para el control de ésa.

Por lo que necesitaremos un dispositivo además de la controladora que pueda comunicarse con ella y que cumpla los requisitos de computación que necesita la tarea de analizar imágenes y controlar una proforma formada por servos por medio de un PID.

Dado que la APM usa la programación de los Arduino, la opción recomendada sería usar un Arduino. Sin embargo el dispositivo debe de ser capaz de realizar un tratamiento de imágenes previo a la toma de decisiones. Por ello se ha elegido la Raspberry Pi modelo 3 B+ dada su superioridad en muchos aspectos. Sobre todo la capacidad que tiene la Raspberry de poder usar la librería OPENCV para el tratamiento de imagen [16], [17].

A continuación enumeraré algunas de las características más importantes de la Raspberry Pi 3 B+:

Figura 2-1713 Raspberry Pi modelo 3 B+

- CPU + GPU: Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC @ a 1.4GHz
- RAM: 1GB LPDDR2 SDRAM
- Wi-Fi + Bluetooth: 2.4GHz y 5GHz
- Bluetooth 4.2, BLE
- Ethernet: Gigabit Ethernet sobre USB 2.0 (300 Mbps)
- GPIO de 40 pines
- HDMI
- 4 puertos USB 2.0
- Puerto CSI para conectar una cámara.
- Puerto DSI para conectar una pantalla táctil
- Salida de audio estéreo y vídeo compuesto
- Micro-SD
- Power-over-Ethernet (PoE)

Además tiene la capacidad de poder soportar sistemas operativos por lo que se puede considerar un microordenador. Es una máquina muy potente en relación con el poco peso y tamaño que tiene. Además en Amazon se puede encontrar por un precio muy económico, alrededor de 40€.

Raspberry dispone de otros modelos, pero con procesadores más lentos. Dada la importancia del ciclo del bucle en el que la Raspberry deberá de realizar una serie de acciones, he priorizado el modelo de mayor velocidad de procesamiento. Dichas tareas consisten en: tomar imagen, tratarla, calcular parámetros de actuación y realizar comunicación.

Los inconvenientes de este tipo de dispositivos es que se necesita de grandes nociones de informática y estar relacionado con el Linux para poder sacarle el potencial que posee. Para empezar no podemos usarla tras comprarla, es necesario instalar un sistema operativo para poder empezar a usarla. La empresa creadora de este dispositivo es consciente de ello y brinda una gran ayuda y foros donde la comunidad puede aportar información y proyectos realizados al público. En su página web oficial se pueden descargar una serie de imágenes de sistemas operativos para poder instalarlos en la SD y empezar a poder usarla. En mi caso he instalado el sistema operativo Raspian por su sencillez.

2.2.1 Otros dispositivos necesarios para el tratamiento de imágenes

Además de la Raspberry, necesitamos de una serie de elementos para poder realizar todas las tareas descritas anteriormente. Dichos elementos constituirán entradas o salidas de información o de energía al sistema, necesarios para que pueda interaccionar con el mundo físico.

Para ello disponemos de:

- Módulo de Cámara PICAMERA de Kuman.

  Consiste en una cámara de pequeño tamaño y prestaciones normales a un precio asequible (15€) compatible con la Raspberry. La cámara proporcionará a la Raspberry la posibilidad de poder analizar el entorno desde un punto de vista visual, por lo que es fundamental.

  Sus características son:

  - 5 megapíxeles.
  - Calidad de hasta 1080 píxeles.
  - Sensor OV5647.
  - Objetivo SLR full-frame.
  - Foco fijo de 1 m a infinito.
  - Conector plano de 15 pines MIPI para conectar a un puerto especial de la Raspberry dedicada a estos módulos.

  La Raspberry tiene la capacidad de también tomar imágenes de cámaras con prestaciones mucho mejores como por ejemplo Webcams, pero dado que el peso toma papel muy fuerte en el diseño, se ha optado por este módulo.

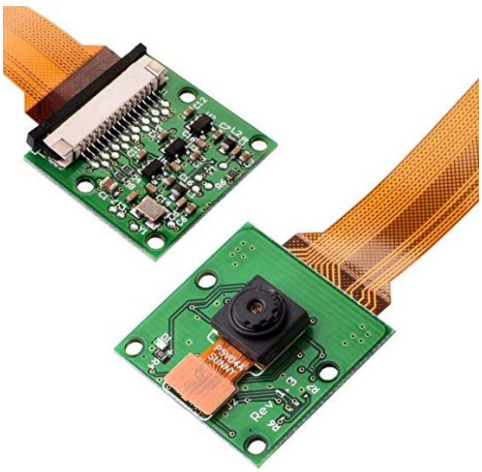


Figura 2-18 Picamera de la empresa Kuman

- Batería externa Omars.

  La Raspberry al ser un componente electrónico necesita de una fuente para poder funcionar. Mientras que se experimente con ella en tierra se puede conectar por el puerto micro-USB una fuente de 5 V con al menos 2.4 A. Sin embargo en el aire necesitamos disponer de una fuente independiente de la batería asignada a la controladora para no reducir más de lo necesario la autonomía del vuelo.

Por ello se ha elegido un tipo de batería muy usada en al ámbito de móviles. Un cargador portátil que suministra por medio de un puerto USB una tensión de 5 V y una corriente diferente dependiendo del puerto al que se conecte.

He elegido la Batería externa Omars que nos brinda una autonomía de 10000 mAh con un peso de sólo 207 gramos.

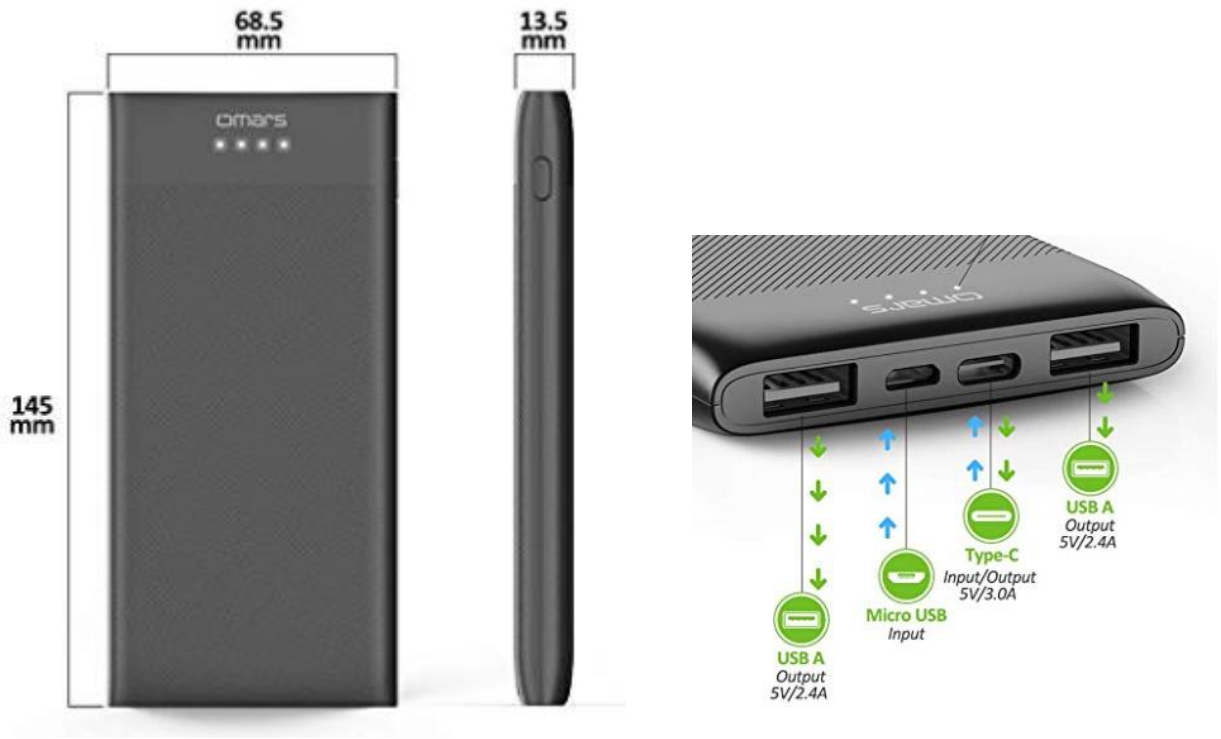


Figura 2-19 Batería externa de la empresa Omars

Como podemos ver en la figura 2-19, es capaz de proporcionar los 2.4 A con cualquier puerto que usemos.

También hay otros modelos que cumplen las especificaciones de salida que necesitamos, pero se ha encontrado este modelo particularmente poco pesado comparado con otros modelos con la misma capacidad. Además tiene un precio económico de 15€.

- Digital Servo PDI-4503HB

Dado que se van a realizar una serie de pruebas con la Raspberry en tierra con el tratamientos de imágenes, para que también se pueda implementar el PID y poder así realizar experimentos más completos es necesario que la salida del control de la Raspberry pueda controlar la posición de la cámara de alguna manera.

Para dicho objetivo, podemos usar un servo. Consiste en un elemento que contiene un eje que puede rotar hasta 180 grados, y que dicha posición del eje puede ser controlada mediante los pulsos emitidos por un canal de control.

Por ello, si instalamos la cámara en el eje del servo, de manera que podamos controlar el grado de inclinación del servo y por ende de la cámara en el eje Z, podríamos simular como si estuviéramos rotando el drone.

Existen todo tipo de servos: los hay muy baratos donde el control de posición del servo se puede volver un problema debido a la gran error deposición y también los hay con una gran precisión, y que

incluso puedan rotar el eje de manera continua. Aunque para nuestro caso, con que pueda girar 180 grados y tenga una precisión aceptable es suficiente.

Por ello, el servo elegido es el PDI-4503HB por 12€. Sus especificaciones son:

- Torque de hasta 3.95 Kg/cm
- Engranajes de alta precisión con anodizado duro
- Motor con engranajes de metal
- Dead band de 2 microsegundos
- Funcionamiento a 330Hz
- Velocidad de hasta 60 grados en 0.12 segundos

Figura 2-20 Servo PDI-4503HB

2.2.2 Programas a usar y lenguaje de programación

Una vez instalado el sistema operativo Raspian en la Raspberry, procedemos a realizar el programa que realizará el control a partir del tratamiento de imágenes.

En nuestro caso se programará en Python. En la versión 2.7 ya que las librerías a usar están más disponibles en cuanto a facilidad de instalar en la Raspberry como de encontrar ayuda.

Dado que en la Raspberry se puede programar y ejecutar archivos de Python desde la terminal no hace falta recurrir a un programa para realizar el programa. Aunque existen programas que nos ayudan a tener un mejor control del programa a la hora de la edición y la visualización, estos programas son los IDE.

Existen muchos tipos de IDE que están basados en diferentes versiones del lenguaje de Python y otros lenguajes y que nos ofrecen ciertas ayudas a la hora de programar. En mi caso he usado el programa IDLE ya que se encontraba de serie en el sistema operativo.

Usar un IDE puede ofrecerte servicios como:

- Menor manejo alrededor del código.
- Detección de errores durante la escritura del código, y análisis del código en busca de errores antes de ejecutar el código con su correspondiente error detallado en línea y concepto.

- Visualización de salidas de terminal, pantalla …
- Ayuda en el autocompletado de palabras.
- Se pueden abrir varios programas en diferentes ventanas.

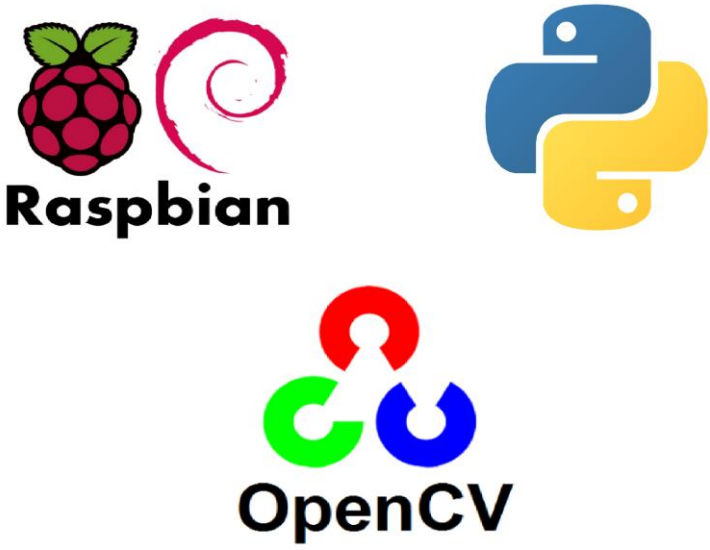


Figura 2-21 Logos de los software: Raspian, Python y OpenCV

2.3 Ensamble de componentes

A continuación se procederá a documentar el proceso de construcción de cada sistema por separado para documentar las conexiones realizadas y así poder dar una visualización tanto del sistema al completo como de la distribución de cada uno de sus componentes y así ayudar a su comprensión.

2.3.1 Construcción del drone

La construcción o ensamblaje de todos los componentes del drone consiste en una tarea bastante simple si se realiza con cierto cuidado y atendiendo a unos detalles importantes que comentaré a lo largo del documento. Para simplificar la documentación se complementará el proceso con imágenes esquematizadas y simbólicas para las explicaciones, aunque también se aportarán fotos visualizando como queda el resultado en el drone.

Es necesario disponer de ciertas herramientas que nos ayudarán en ciertos procesos del montaje, lo más importantes son:

- Soldador, estaño y pasta para soldar.
- Elementos de corte como tijeras, pelacables o cúter.
- Elementos de aislamientos como plásticos termo-retráctiles o silicona caliente.
- Gama amplia destornilladores.
- Bridas de varios tipos de longitudes.
- Voltímetro.

Una vez que disponemos de los componentes que forman el drone, además de los materiales necesarios podemos empezar a construir el drone. Se dividirá el proceso en pasos considerados más determinantes:

1. **Ensamble del cuerpo del drone y soldado de terminales.**

   En una de las placas que componen el cuerpo del drone, hay unos terminales para que se pueda distribuir la potencia sin tener que recurrir a tanto cableado. Dicha placa es la inferior del drone.

   Procederemos a estañar los terminales con el objetivo de tenerlos listos para futuras soldaduras mediante un soldador.

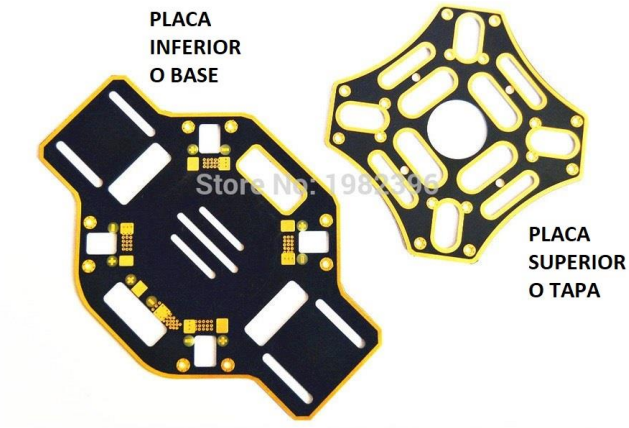


Figura 2-22 Placas inferior y superior del cuerpo del drone

Una vez soldada la placa inferior procedemos mediante un destornillador Allen y los tornillos suministrados por el vendedor, a atornillar las 4 patas correspondientes. El color o la disposición de éstas no importa, son todas iguales.

Y para finalizar se coloca la placa superior (la restante) y se atornilla a las patas mediante los mismos tornillos.

Una vez finalizado ya tenemos la estructura del dron donde irán instalados todos los drones, deberá quedar así:

Figura 2-23 Cuerpo ensamblado

2. Instalación del distribuidor de potencia Power Module.

El Módulo de potencia será el intermediario entre la batería y el resto de las componentes. Suministra tensión y corriente mediante los cables de Vcc (Rojo) y Ground (Negro) a los variadores, y a la controladora por medio de una conexión formada por varios conductores ya que el módulo no solo aporta la tensión corriente necesarias para funcionar a la controladora, como se ha dicho antes también le proporciona información acerca del estado de la batería a la controladora para su monitorización.

Sin embargo el cuerpo o estructura posee una manera de distribuir la potencia suministrada por la batería, por lo que hay que pelar los cables del módulo correspondiente a la alimentación de componentes para soldar los terminales con la placa base del cuerpo del dron (Vcc como "+" y Ground como "- "). Los terminales a los que deberá ir soldado se muestran en la siguiente figura:

Figura 2-24 Conexiones del módulo de potencia

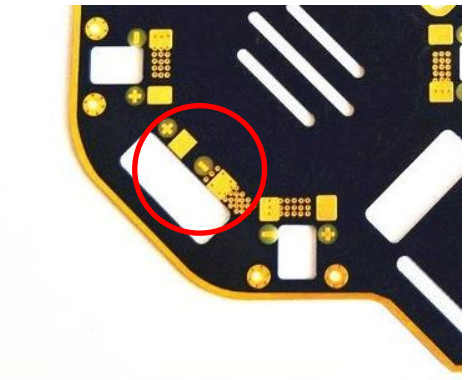


Figura 2-25 Terminales para el módulo de potencia

La posición del módulo no importa mucho pues es un elemento bastante pequeño. En este caso se ha colocado en un lateral posicionado de manera que la conexión a la batería está orientado a la parte delantera del drone.

3. Instalación de motores y variadores.

Los motores así como los variadores son todos iguales, no hay que hacer una selección a la hora de posicionarlos en unas patas u otras. Es cierto que en un drone tipo cuadricóptero los motores no giran todos en el mismo sentido, pero eso es configurable en la conexión de los terminales, los elementos son todos iguales entre ellos.

Dicho esto, se procede a atornillar los motores a los extremos de las patas, en los huecos que la propia estructura de la pata nos facilita para afianzar el motor. Si se necesita un ajuste para afianzar más el motor al cuerpo del drone y así evitar posibles vibraciones indeseadas, se pueden montar sobre unas estructuras en forma de cruz suministradas por el fabricante. En este caso no han sido necesarias.

Una vez instalados los motores, se procede a la instalación de los variadores. Para ello podemos ayudarnos de la siguiente figura para entender las conexiones.

Los cables de alimentación irán soldados a los terminales de la placa del cuerpo base de la misma manera que se hizo con el Power Module. El cable formado por 3 conductores (Negro Ground, Rojo Vcc, blanco señal) que va conectado a la controladora no lo tocaremos. Los conectores azules que van a los motores no se soldarán, puesto que habrá que determinar el orden correcto de las fases para conseguir el sentido del giro deseado.

Figura 2-26 Esquema de los terminales de un variador

Por ello, y adicionalmente para poder ser sustituidos en caso de algún desperfecto tanto en el motor como en el variador, se instalarán unos conectores tipo "Bullet" a los terminales de las conexiones azules del variador y sus correspondientes a los terminales de los motores.

Figura 2-27 Conectores tipo "Bullet"

Es muy importante realizar una buena soldadura y gruesa cuando se suelda los terminales de las fuente de los variadores a las placas. Como comprobación de la conexión se puede usar el voltímetro para asegurar el contacto eléctrico entre ellos para detectar errores de soldadura.

4. Colocación del soporte anti vibraciones y la controladora de vuelo.

Primeramente deberemos decidir dónde instalaremos la controladora de vuelo: Sobre la tapa, o entre la tapa y la base. En este caso, se ha decidido instalarla sobre la tapa ya que es un elemento de pequeño peso y las baterías son mucho más pesadas. Así conseguiremos bajar el centro de gravedad del sistema contribuyendo un poco a una mejor estabilidad.

La propia estructura anti vibraciones posee una serie de pegatinas de doble cara para afianzarla a superficies. En el momento de posicionarla, hay que tener en cuenta que la controladora medirá por acelerómetros y giroscopios los movimientos del drone, por lo que deberá de colocarse justo en el

centro geométrico formado por la estructura entera para asegurar una buena lectura y así seguir contribuyendo a una mejor estabilidad.

Una vez instalada, podemos colocar con las mismas pegatinas la controladora a la estructura anti vibraciones, pero se ha considerado que una fijación mucho más sólida se podría realizar con bridas y así se ha realizado.

5. Conexión de terminales a la controladora.

Una vez instalada la controladora, podemos proceder a la conexión de los terminales de los variadores y del Power Module. Para ello se ha creado la siguiente representación esquemática:

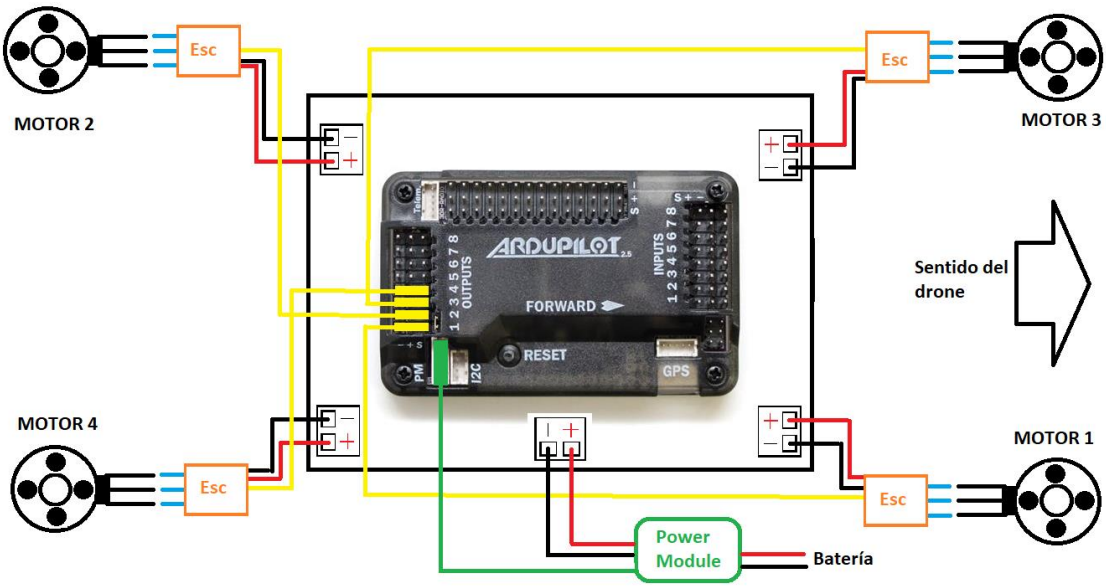


Figura 2-28 Esquema de conexiones de variadores y módulo de potencia a la controladora de vuelo

Tal y como se muestra en la figura y en los capítulos anteriores, cada terminal tiene su inequívoca conexión en el puerto indicado en salidas o "OUTPUTS". Aun así para despejar cualquier duda, hay dos cuestiones que pueden ser conflictivas:

- Cada variador tiene su puerto designado. Para ello hay que fijarse a qué motor está controlando. El número del motor que controle definirá al puerto al que debe estar conectado.
- Para realizar la conexión del puerto con los variadores hay que tener en cuenta código de colores y las indicaciones de la controladora.

Ahora que están instalados los variadores, podemos afianzarlos en los brazos del drone mediante bridas.

6. Instalación del módulo GPS y conexión a la controladora.

El módulo GPS se suele instalar a una altura del cuerpo del drone para disponer de la mejor cobertura posible, ya que el ruido electromagnético de los componentes electrónicos puede afectar a las medidas. Para ello se puede ensamblar el módulo a una estructura constituida por una base y una varilla de fibra de vidrio, aportando dicha altura de seguridad al módulo respecto al resto del dron.

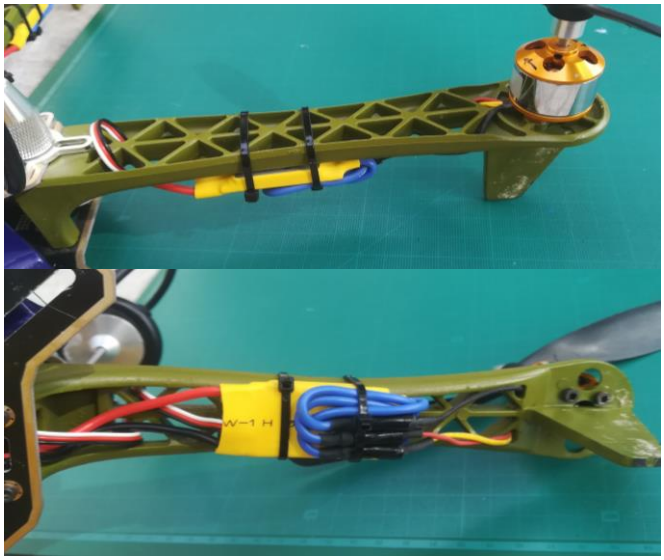


Figura 2-29 Motor y variador instalado en el drone

Sin embargo esto puede ser contraproducente ya que la estructura puede aportar cierto balanceo al módulo GPS si no es suficientemente rígida y así aportarnos lecturas con bastantes perturbaciones. Por lo que hay que aportar una altura suficiente sin llegar a perder consistencia.

La mayoría de los módulos GPS poseen en la cubierta que protegen los circuitos electrónicos una flecha indicadora del sentido en el que tiene que estar el módulo. En el caso en el que no posea dicha flecha como es mi caso, nos guiaremos por la posición de la salida del cable, ya que indica la parte trasera del módulo.

Durante este proceso únicamente hay que atornillar las correspondientes partes y pegar mediante un adhesivo el módulo GPS a la estructura siguiendo las instrucciones del fabricante. Para asegurar la plataforma al drone tendremos que usar los tornillos usados para ensamblar el cuerpo superior del drone a las patas, por lo que hay 3 distintas posiciones donde instalarlo. En este caso por mera cuestión visual se ha instalado en la parte trasera derecha.

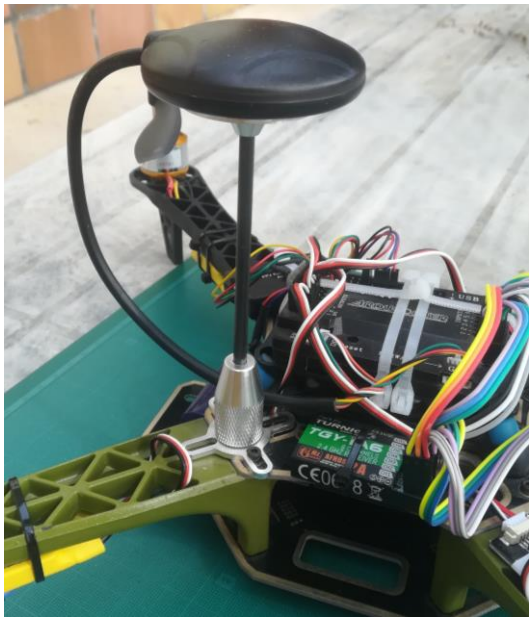


Figura 2-30 Módulo GPS instalado en el drone

En cuanto a la conexión del módulo con la controladora puede existir una complicación, y es la de que del módulo parten 2 terminales o puertos:

- La agrupación de 2 conductores corresponde a los pines designados para la alimentación del módulo. El puerto al que debe conectarse es el designado como I2C, el terminal que se encuentra justo enfrente del designado para el Power Module.
- La agrupación de 4 o 5 conductores corresponde a los pines designados para la transmisión de datos a la controladora. El puerto al que debe conectarse es el designado como GPS.

7. Instalación de la receptora y conexión a la controladora.

La receptora es el dispositivo que "escuchará" las órdenes que emitamos desde la emisora y las transmitirá a la controladora. Por lo que posee un terminal para cada canal que queramos transmitir. En nuestro caso disponemos de hasta 6 canales pero únicamente usaremos 5 canales.

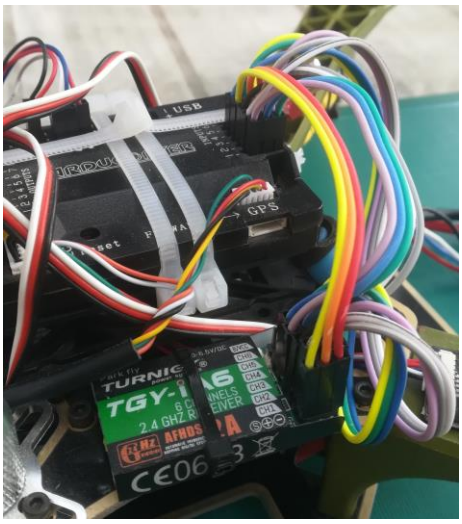


Figura 2-31 Receptora instalada en el drone con su correspondiente conexión a la controladora

Por ello en este caso lo único que debemos de realizar es la correcta conexión de cada canal al puerto correspondiente. En este caso se conectarán a los puertos de entrada o "INPUTS", y siguiendo las especificaciones de color o señalización de los puertos.

8. Instalación del módulo Buzzer-Led y conexión a la controladora.

Consiste en un módulo de ampliación, por lo que estará conectado a los puertos de ampliación que provee la controladora para controlar dispositivos extras instalados en el drone como servos, Buzzer, luces…

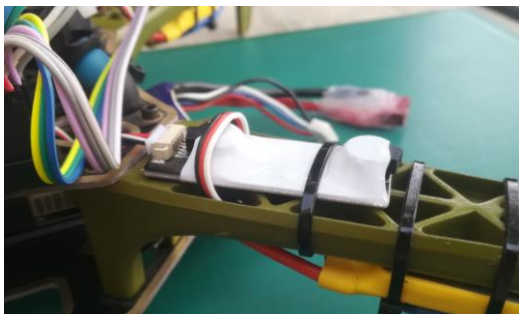


Figura 2-32 Módulo Buzzer-Led instalado en el drone

En cuanto a la disposición del módulo en el drone es indiferente, y los puertos correspondientes están mostrados en la información que da el fabricante. Una vez instalado se puede programar el pin correspondiente desde Mission Planner.

9. Preparación de terminales para la batería y colocación de ésta.

Por último, debemos de preparar la batería para su uso. Esto puede suponer un problema porque hay diferentes tipos de conexiones que se usan en los puertos de la batería. Si resulta ser el mismo que tenemos en el Power Module no hay problema, pero si son distintos hay que quitar las conexiones y reemplazarlas con unas compatibles. En este paso puede resultar interesante afianzar la conexión a los terminales con silicona caliente ya que puede servir como aislador y como protector en cuestiones de tirones de los terminales a la hora de desconectar la batería.

Una vez hecho esto, podemos pegar en el hueco designado para la batería unos velcros así como el correspondiente en la batería para poder asegurarla durante el vuelo, aunque también se puede afianzar mediante bridas. Sin embargo en el caso en el que se disponga de varias baterías, la opción del velcro es una solución muy eficiente para poder sustituirlas de manera mucho más eficiente.

Importante instalar la batería de manera que no aporte momentos en el vuelo, eso significa que se coloque alineando su centro de gravedad con el del drone para intentar ayudar a la estabilidad.

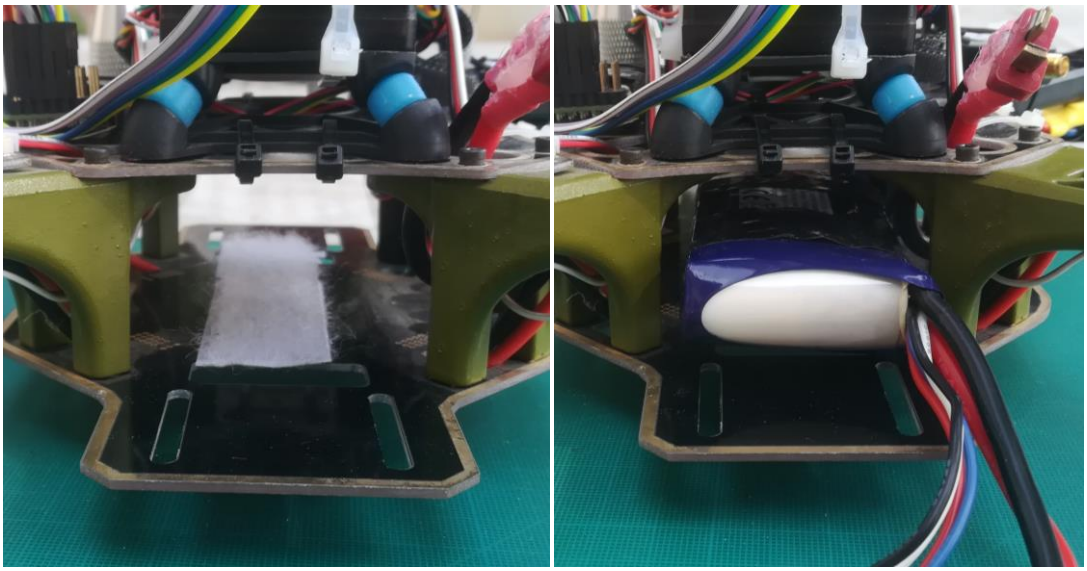


Figura 2-33 Posición de la batería en el cuerpo del drone

Por último antes de disponer del drone listo para empezar a realizar pruebas en él es necesario una serie de comprobaciones y de calibraciones en el software Mission Planner. Para ello conectaremos la controladora al ordenador vía USB.

Si recordamos el paso para instalar los motores, éstos no se encuentran aún conectados a los terminales del variador. Esto se debe a que en el modelo de drone cuadricóptero los motores no giran todos en el mismo sentido. Si así fuera se produciría empuje, pero el par total de los motores acabaría generando un momento que produciría un giro en torno el eje Z del drone de forma incontrolada. Para solventar esto, los motores se distribuirán en pares que girarán en sentidos diferentes.

Para determinar el giro de cada motor, procederemos a conectar de manera arbitraria los motores con sus respectivos variadores y después conectaremos la batería. Es importante no poner aún las hélices para evitar cualquier posible accidente.

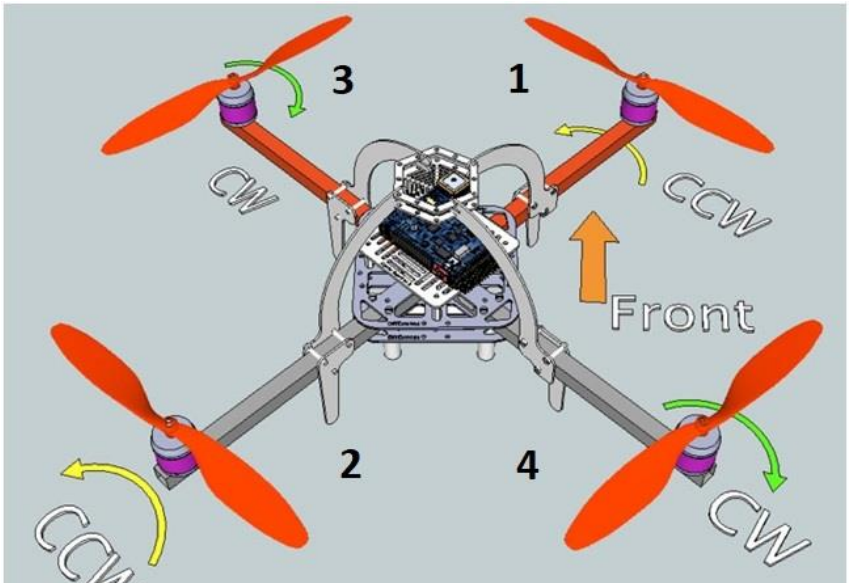


Figura 2-34 Distribución de hélices

Una vez que estemos dentro del software Mission Planner podemos realizar una prueba de motores que realizará un bucle infinito en el que se le aplicará una pequeña tensión a cada motor de manera separada durante un unos momentos. En este momento, y siguiendo el la figura de distribución de motores con sus respectivos giros de antes tenemos que asegurarnos de que efectivamente los motores giran en el sentido apropiado.

En el caso en el que el motor gire en el sentido opuesto, simplemente debemos de alternar la conexión de dos terminales del motor correspondiente. Podemos asegurar la conexión mediante tubos termo retráctiles para que no se suelten en pleno vuelo.

Una vez que los motores están bien configurados, procedemos a colocar las hélices en sus correspondientes motores. Normalmente la propia hélice tiene inscrito si es tipo CV o CWW, pero si no es el caso podemos fijarnos en el ángulo de ataque.

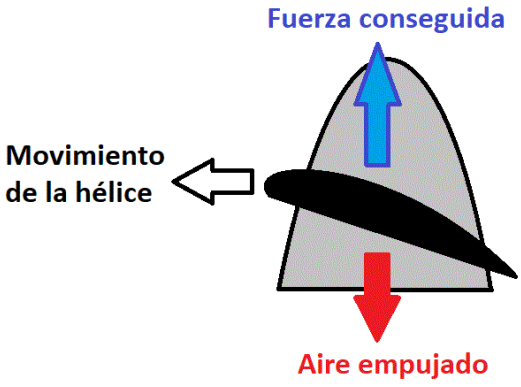


Figura 2-35 Funcionamiento de la hélice

Ahora que tenemos todo los elementos bien ensamblados, podemos realizar la calibración de sensores. Para ello, con el dron conectado vía USB iniciamos la conexión. El programa leerá la configuración de todos los parámetros presentes en la controladora y nos permitirá configurarlo.

Y al final, tendremos el drone operativo listo para su uso y configuración.

2.3.2 Construcción de plataforma para seguimiento de objetivo

Al tratarse de un experimento para poder poner a prueba el programa de seguimiento y el tratamiento de imágenes, es decir que será temporal ya que tanto la Raspberry como la cámara irán instalados en el drone, no es necesario diseñar una compleja estructura al detalle.

La estructura consistirá en dos servos que se encontrarán instalados uno en el brazo giratorio de otro, y en el último irá la cámara:

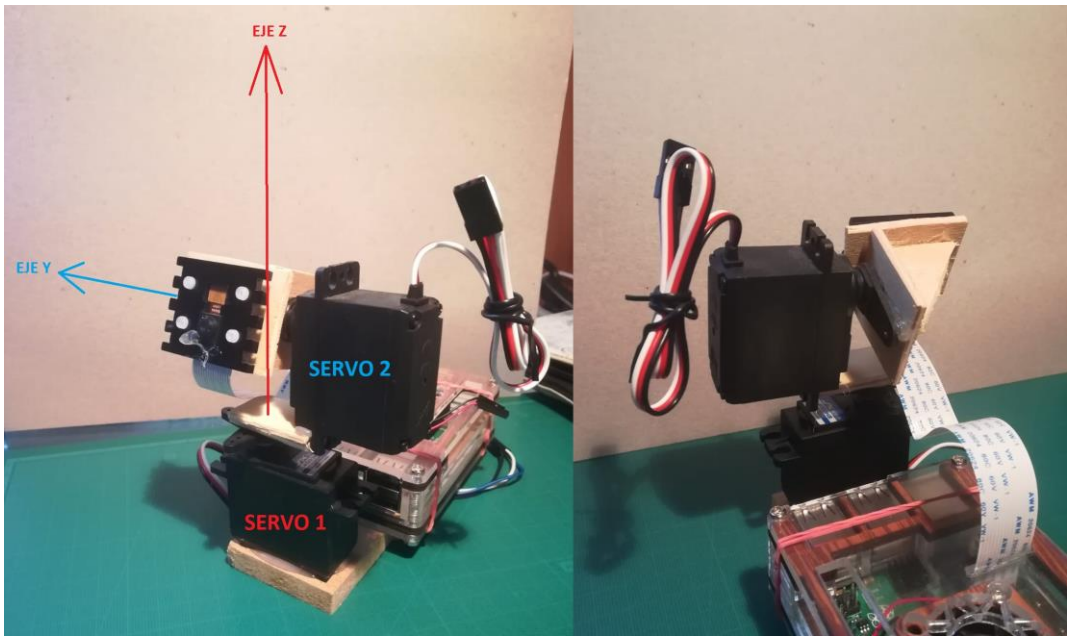


Figura 2-36 Ejes de la plataforma con servos

De esta manera podemos controlar la posición angular de la cámara del eje Z mediante los pulsos que se transmitan al servo 1 por el programa, y de igual manera podemos hacer con el servo 2 para controlar la posición angular del eje Y de la cámara.

Importante denotar que para el sistema funcione lo mejor posible, hay que instalar los servos de manera que los ejes rotatorios coincidan con la cámara. De lo contrario cuando un servo rote, la cámara podría no solo rotar también, podría desplazarse.

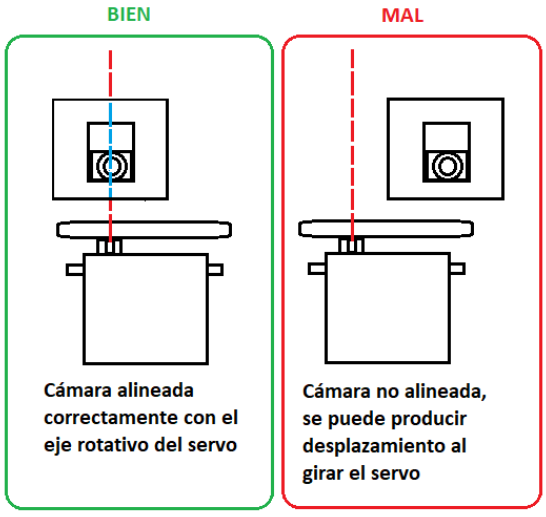


Figura 2-37 Colocación de cámara sobre servo

Dado que es un proyecto temporal, la estructura para adecuar los servos, y de soporte para la cámara puede realizarse con un panel de madera de contrachapado de unos 2 mm. Se trata de un material muy barato y ligero en el que no se le puede exigir una gran exactitud en los cortes puesto que el material puede desprenderse. Sin embargo para esta aplicación se ajusta perfectamente a nuestras necesidad y además su ligereza y porosidad la hace muy compatible con silicona caliente para ensamblar piezas.

El diseño de la estructura para el soporte de la cámara que irá en el servo 2 es la siguiente:

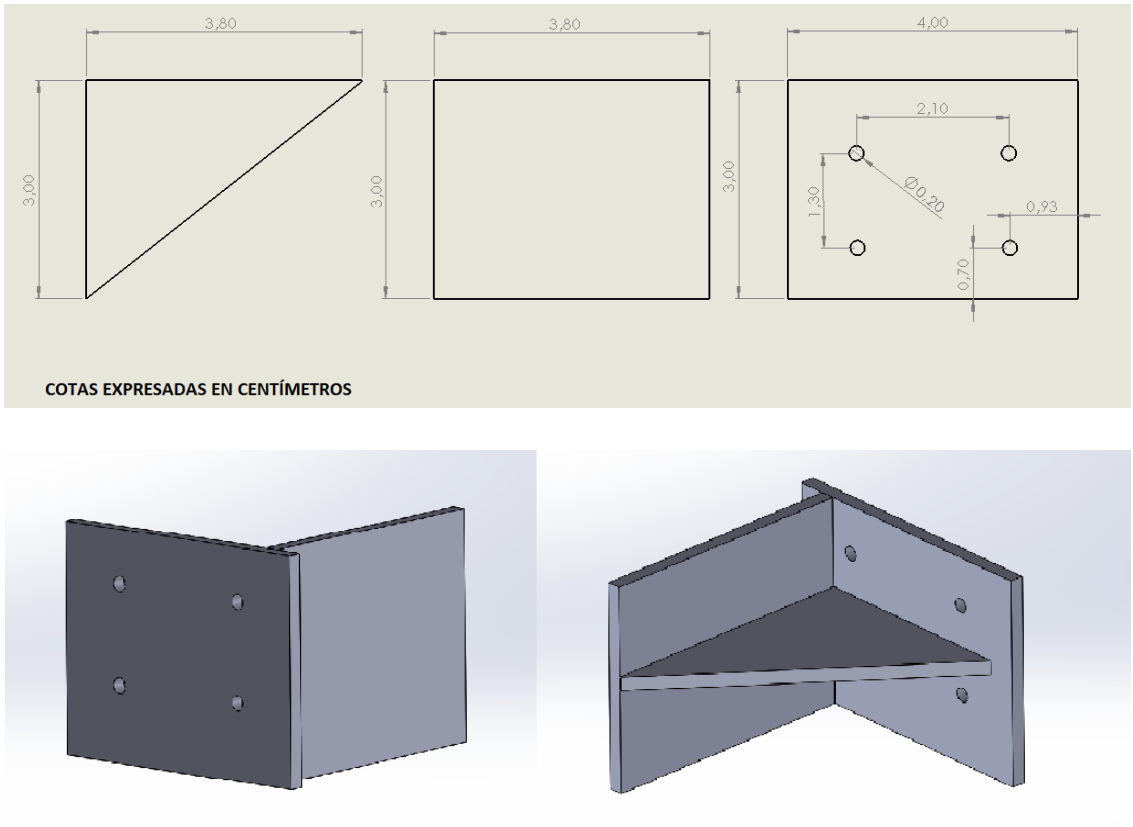


Figura 2-38 Planos del soporte de cámara para servo y visualización 3D

Tanto el diseño gráfico en 3D como la realización de los planos correspondientes han sido creados a partir del programa de modelado en 3D Solid Work. En dicho programa se ha modelado cada elemento para su ensamblaje y diseño por separado. Una vez tenemos la estructura definida en 3D, se puede realizar un plano en el mismo programa para acotarlo de manera normalizada.

Una vez terminado la estructura, tenemos que conectar los pines de los servos a los pines correspondientes de la Raspberry. Para ello nos fijaremos en la distribución de pines del modelo elegido de Raspberry y en la de los servos.

Los servos poseen 3 pines que tienen un color asignado para identificarlos con mayor facilidad.

- Pin correspondiente al cable negro: Indica la tierra o Ground (0V).
- Pin correspondiente al cable rojo: Indica la fuente o VCC (5V).
- Pin correspondiente al cable amarillo o blanco: Indica la señal de control PWM que se usará para controlar la posición del servo.

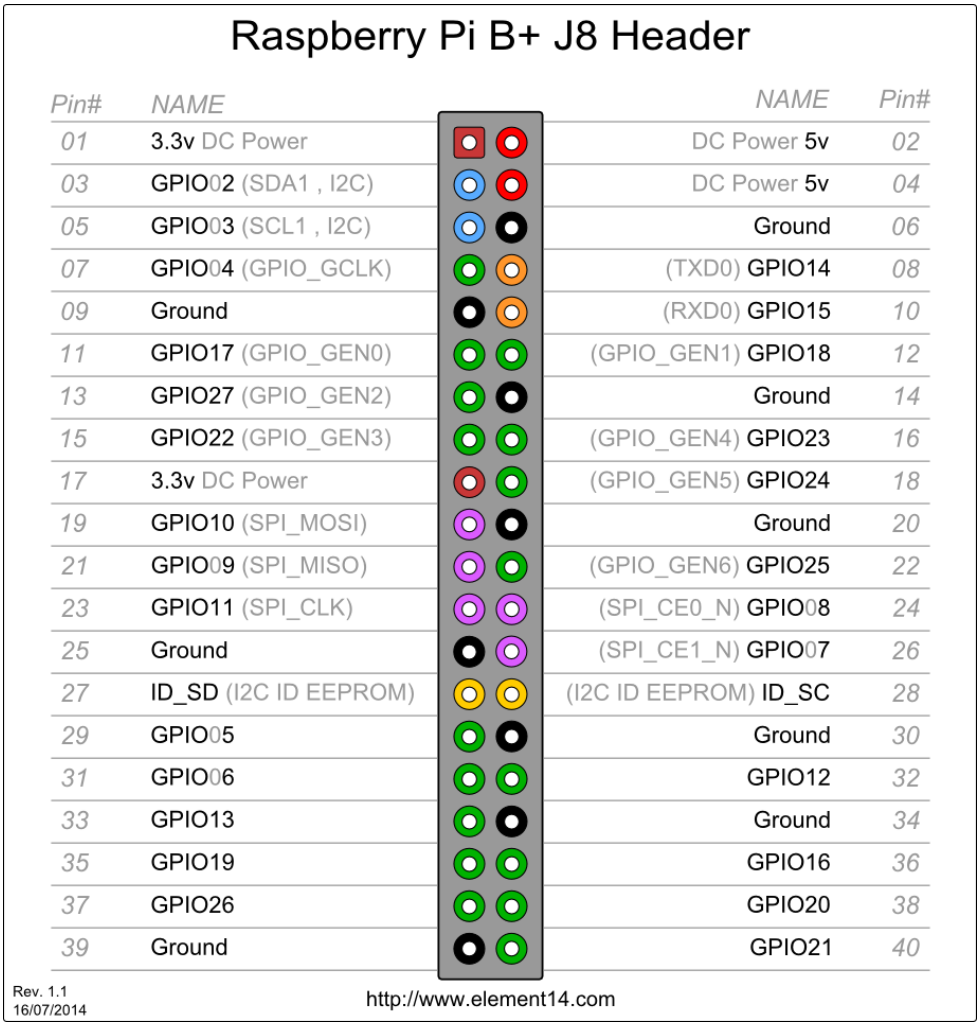


Figura 2-39 Distribución de pines en Raspberry [18]

Una vez identificados los pines, procederemos a unir los pines:

- Los cables negros de ambos servos los uniremos en uno, y lo conectaremos en un pin de Raspberry correspondiente de Ground. Como por ejemplo el pin 6.
- Los cables rojos de ambos servos los uniremos en uno, y lo conectaremos en un pin de Raspberry correspondiente de VCC. Como por ejemplo el pin 4.
- En nuestro caso el cable de señal de ambos servos es blanco, y lo conectaremos en puertos diferentes para poder controlar de manera independiente cada servo.
    - El servo 1 lo conectaremos al pin 13 correspondiente con el GPIO 27.
    - El servo 2 lo conectaremos al pin 15 correspondiente con el GPIO 22.

3 SEGUIMIENTO DE OBJETIVO: RASPBERRY

3.1 Propuesta de ensayo

Como se ha comentado anteriormente, se ha propuesto un experimento ajeno al proyecto del drone autónomo para poder realizar pruebas con el código de seguimiento de objetos. Es muy normal que en las primeras fases de implementación del código las cosas no funcionen como se espere y el programa se descontrole en algunos sentidos. Esto sumado a lo peligroso que puede ser entrar en contacto con las hélices de un drone como el de este proyecto o por seguridad del proyecto mismo para no exponer a posibles accidentes el drone (Sistema bastante frágil), se ha propuesto la idea de hacer el problema del seguimiento de objetivo mediante la actuación de servos. Y así poder centrarnos en el problema del control de la plataforma con mucha más eficiencia y seguridad.

Otra gran ventaja de este experimento consiste en la posibilidad visualizar el tratamiento de imágenes durante el proceso en un monitor conectándolo por puerto HDMI. De esta manera podemos entender mucho mejor paso a paso el proceso de detección del objetivo, y encontrar de una manera mucho más sencilla típicos errores que desde una terminal costaría un tiempo adicional encontrar la raíz del problema.

3.2 Programación: Librerías usadas

Python consiste en un lenguaje de programación de alto nivel que se puede usar para crear una gran diversidad de proyectos. Sin embargo Python no proporciona todas las funciones que queramos, para ello dispone de librerías que se pueden descargar e incluir en el código. Dichas librerías congregan una serie de funciones orientadas a una aplicación para que podamos llamarlas desde el código y Python las pueda usar. Existen librerías que nos ayudan de manera que podamos ahorrar código escribiendo procesos ya que ellas las contienen. Y hay otras que nos proporcionan nuevas opciones para programas que no disponíamos o no éramos capaces de llegar a ellas.

En nuestro código usaremos las siguientes librerías:

- picamera.

  En esta librería podremos acceder a las funciones relacionadas con el módulo de la cámara que hemos instalado en la Raspberry. Se eligió dicha cámara por la compatibilidad que tenía con Raspberry no solo por su conexión sino por las librerías que existen para poder usar la cámara. Esta consiste en una de ellas [19].

  Para ello extraeremos los módulos que necesitemos para el proyecto:

  - PiCamera: Con este módulo podemos asignar una variable con la ruta correspondiente a la cámara y usarla dentro del programa ya sea para cambiar la configuración de esta o bien para capturar imágenes.
  - PiRGBArray: Con este módulo podemos realizar una captura de imagen y guardarla en un vector RGB de 3 dimensiones (fila, columna, color).

- cv2.

  Esta librería corresponde con la librería de OpenCV (Open Source Computer Vision Library). Tal y como su nombre indica, es una librería destinada a la visión artificial de libre uso tanto comercial como académica. Posee librerías para varios tipos de lenguaje de programación (C++, Python, Java…) y nos asegura una eficiencia computacional incluso para aplicaciones de tiempo real.

  Con esta librería podremos realizar todo lo referente al tratamiento y análisis de imágenes. Nos proporciona una cantidad inmensa de funciones programables mediante parámetros con varios sistemas teóricos de tratamientos de imágenes implementados. [20]

Por destacar algunas de las funciones que posee esta librería numerare de forma breve algunas que usaremos en el proyecto:

- Conversión de imágenes a color RGB a color HSV.
- Descomposición de imágenes RGB en 3 imágenes correspondientes a cada color en escalas de grises.
- Aplicación de máscaras.
- Aplicación de filtros.
- Detección de Círculos en imágenes.

- Pigpio.

  Mediante esta librería definida especialmente para Raspberry podremos comunicarnos con el "pigpio Daemon" con el fin de controlar y usar los pines GPIO de la Raspberry.

  Nos permitirá poder configurar si un pin es salida o entrada, y usarla con todo tipo de funciones para cada caso. [21]

  Algunas de las funciones más importantes son:

  - Uso de pulsos para controlar servos.
  - Lectura PWM de los pines de entrada.
  - Comunicación por puerto serie.

- Numpy.

  Una potente librería necesaria para operar con vectores de N dimensiones y matrices. También es capaz de proporcionar funciones para crear datos algebraicos, y aleatorios. [22]

3.3 Programación: Estructura del programa

Durante todo el experimento, el programa estará en ejecución. Su estructura estará formada por una declaración de variables seguida de un bucle infinito del que no saldrá hasta que se presione la tecla "Esc".

Tal y como se muestra en la siguiente figura, primero el programa realiza una serie declaraciones de variables e inicializaciones. Estas tareas comprenden:

- Inicializa variable asignada a la cámara.
- Configura los pines correspondientes a los servos en modo salida.
- Inicializar los ventanas donde mostraremos diferentes pasos del tratamiento de imágenes.
- Inicializar los rangos de máximos y mínimos de los colores que vamos a detectar.
- Inicializar los parámetros del PID.

Una vez finalizada la inicialización se realizará un bucle infinito en el que se ejecutarán una serie de tareas que podemos englosar en dos grandes bloques que explicaremos más detalladamente más adelante:

1. Tratamiento de imágenes: que engloba las tareas correspondientes a la captura y tratamientos de imágenes proporcionando datos de interés como la posición del objetivo en píxeles de la imagen.
2. PID: que consiste en el uso de los datos proporcionados por el tratamiento de imágenes para su uso por un PID implementado y la correspondiente corrección de posición se los servos en relación con la salida del PID.

Después de ejecutar estas tareas, el programa consultará el estado de la tecla "Esc" de manera que si no se ha pulsado, volverá a ejecutar el mismo código sin volver a inicializar, volverá justo antes de volver a tomar una nueva captura. Si por el contrario se ha pulsado la tecla, desconectará la cámara y el servo, y cerrará el programa.

.

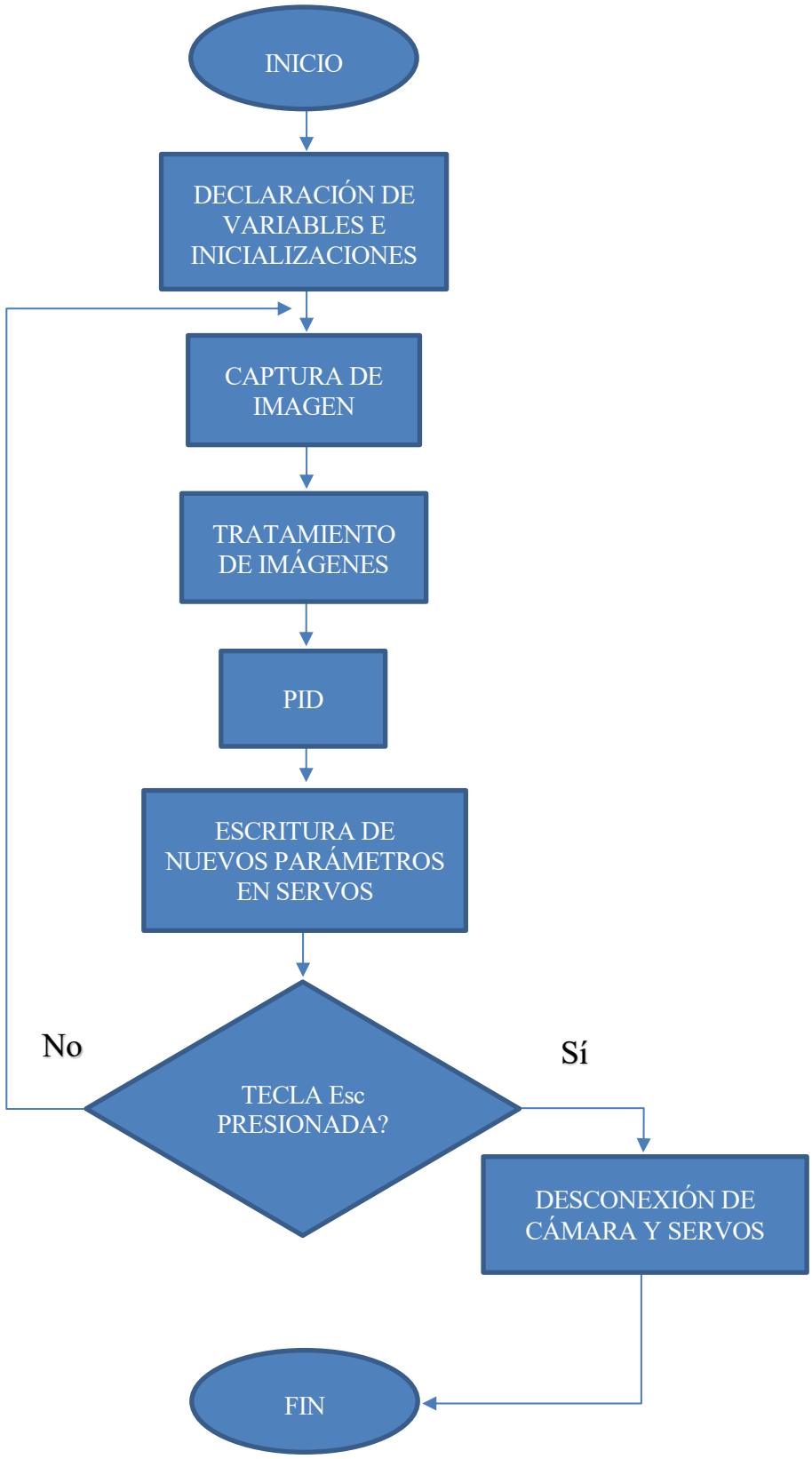


Figura 3-1 Diagrama de flujo del prototipo de seguimiento de objetos con servos

3.4 Programación: Tratamiento de imágenes

Antes de proceder a la captura de imágenes, durante el proceso de inicialización de variables tal y como se ha explicado anterior se le asigna a una variable la dirección de la cámara para poder así tomar imágenes a partir de ésta.

Una vez definida la variable, podemos ajustar los parámetros del proceso de captura de la imagen para ajustarlo a nuestros requisitos.

- La cámara dispone de la capacidad de tomar imágenes de hasta una resolución de hasta 1080x1920 píxeles y así lo hace por defecto. Sin embargo es posible cambiar dicha resolución.
- Con la variable se puede tomar una captura, pero se puede configurar el parámetro "framerate" para que usado en un bucle, tome capturas de pantalla cada cierto tiempo.

Dado que nuestra aplicación está definida como un sistema de control para un sistema en tiempo real, se ha priorizado el framerate para ofrecer un control con un periodo de muestra lo más corto posible. Por lo que se ha optado por el framerate más alto que la cámara puede soportar: 70.

Sin embargo la Raspberry no es capaz de realizar un control continuo con dicho framerate con la resolución máxima por lo que he tenido que reducirla a unos 320x480 píxeles.

Ahora la cámara está configurada lista para tomar fotos a una frecuencia dada, y un "while" definido por dicha frecuencia se procede a tratar las imágenes que se tomen en cada bucle.

Lo siguiente es transformar la imagen capturada en un modelo que podamos usar. La estrategia a seguir será la de transformar la imagen RGB en otro tipo de variable a la que podamos aplicarle un filtro de color, después otro para eliminar el ruido y al resultado aplicarle un método de identificación de círculos.

Ante el dilema de la transformación a realizar se sugieren los siguientes casos, donde se expondrá ejemplos partiendo de una foto donde se muestren diferentes tipos de fruta [23]:

- **Modelo RGB.**

  El color de cada píxel de la imagen está formado por un modelo aditivo de 3 tipos de colores superpuestos: el rojo (R de Red), el verde (G de Green) y el azul (B de blue). Cada color se representa en una escala de intensidad que va desde el 0 (ausencia de color) hasta el 255 (color en toda su totalidad). Y de esta manera es posible representar todos los colores que se presentan en el espectro visible.

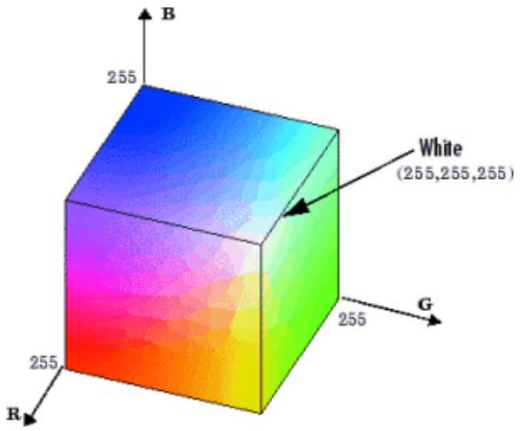


Figura 3-2 Espacio del modelo de color RGB

El color conseguido por el modelo RGB se puede definir como un cubo en el que cada eje representa la intensidad de cada color. Asignándose el color negro al (R=0, G=0, B=0) y el blanco al (R=255,

G=255, B=255).

Cabe decir que en este modelo y en todos, estamos tratando con variables digitales y el mundo real es analógico por naturaleza. Mientras que los instrumentos de iluminación puedan representar con mejor diferencia dicha escala de intensidades mejor definición de color se puede conseguir. Actualmente se puede diferenciar la intensidad de cada color RGB en una escala del 0 al (16*10^6), pero teóricamente en informática usaremos la escala de intensidades explicada anteriormente.

Mediante funciones presentes en la librería de OpenCV podemos usar las imágenes que forman cada color representándolas en una escala de grises. De esta manera se puede hacer un filtro más exhaustivo analizando 3 aspectos a la vez de la imagen.

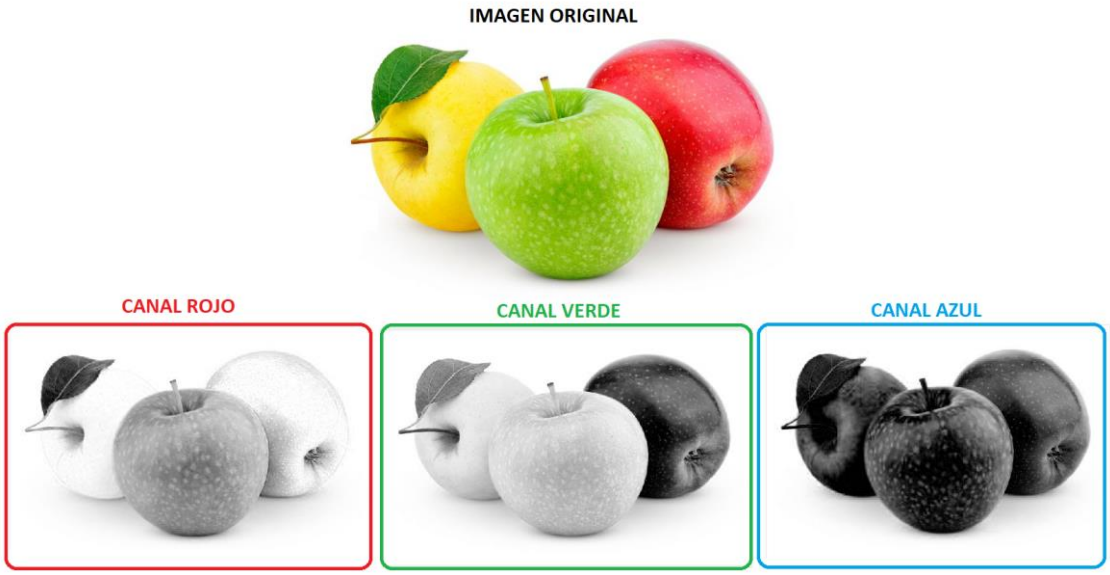


Figura 3-3 Separación de colores RGB

- **Modelo de Grises.**

  Es la transformación más simple donde se unifican todos los colores en una misma imagen. En cada píxel se realiza una media en la que se tienen en cuenta las 3 máscaras. El color en este modelo simplemente consiste en una línea de intensidad del color negro siendo el origen negro y el final blanco.

  Resulta útil para simplificar imágenes pero tiene el gran inconveniente en el que es mucho menos ineficiente a la hora de pasarles filtros de gradientes ya que filtros de color no serían posibles pues es posible identificar varios colores a la vez. Por lo que se descarta.

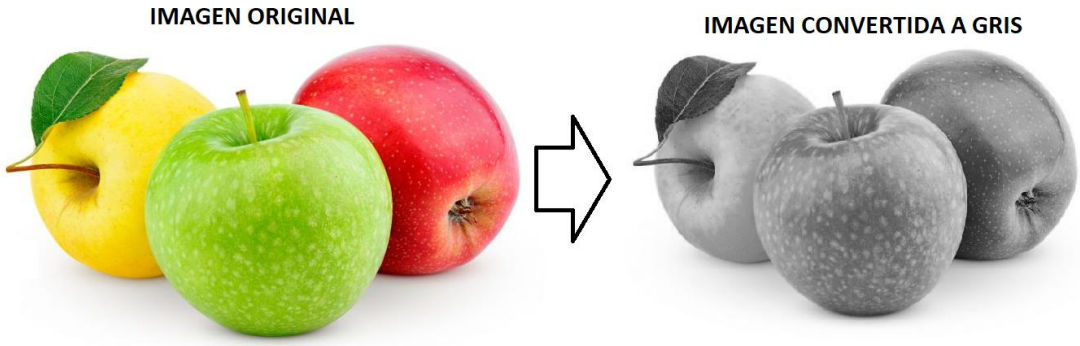


Figura 3-4 Transformación de RGB a gris

- **Modelo HSV.**

  En este modelo se adapta el espacio del modelo RGB para tratarlo de una manera más intuitiva desde el punto de vista del ojo humano con una transformación no lineal. De manera similar a la RGB, separa la imagen en un conjunto de 3 máscaras formadas por: matiz, saturación y brillo.

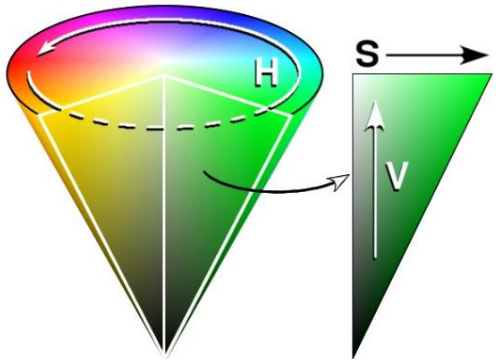


Figura 3-5 Espacio de color del modelo HSV

El color en este modelo se puede representar como un cono donde el perímetro de la base se puede recorrer en función del matiz o Hue (H), para escoger la cromaticidad o clase de color. Después de puede recorrer el plano definido por la saturación o Saturation (S) y el brillo o Value (V). La saturación implica la cantidad de blanco aportado al color e implica por así decirlo la pureza del color, y el brillo denota la apreciación subjetiva de claridad y oscuridad.

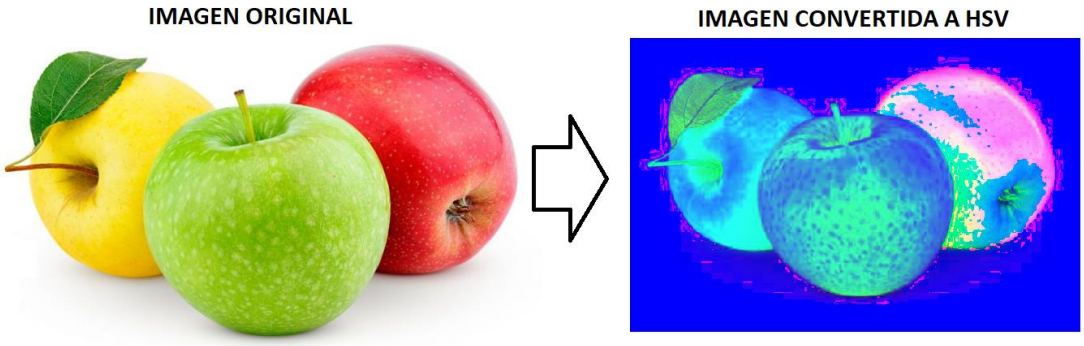


Figura 3-6 Transformación de RGB a HSV

Resulta ser un modelo muy usado en el campo de la visión artificial, pero dada su complejidad a primera vista y la familiarización con el modelo RGB, se descartó para finalmente elegir el RGB.

Una vez definido el modelo de color con el que trabajaremos, se procede a implementar un filtro de color. El filtro creará una matriz con el mismo tamaño de filas y columnas que la imagen, el valor de cada elemento no irá de 0 a 255, será lógico (0 o 1). La explicación de dicho valor lógico consiste únicamente en el resultado de si dicho elemento ha pasado el filtro o no.

Siguiendo el ejemplo de las manzanas, se va a intentar aplicar un filtro de color con el objetivo de poder separar la manzana del resto de la foto.

Para ello se implementará a la imagen un filtro a cada una de las máscaras con el siguiente intervalo:

- En la máscara roja solo se dejará pasar los píxeles con un valor de intensidad comprendido en el intervalo 120 y 255.

- En la máscara verde solo se dejará pasar los píxeles con un valor de intensidad comprendido en el intervalo 0 y 180.
- En la máscara azul solo se dejará pasar los píxeles con un valor de intensidad comprendido en el intervalo 0 y 180.

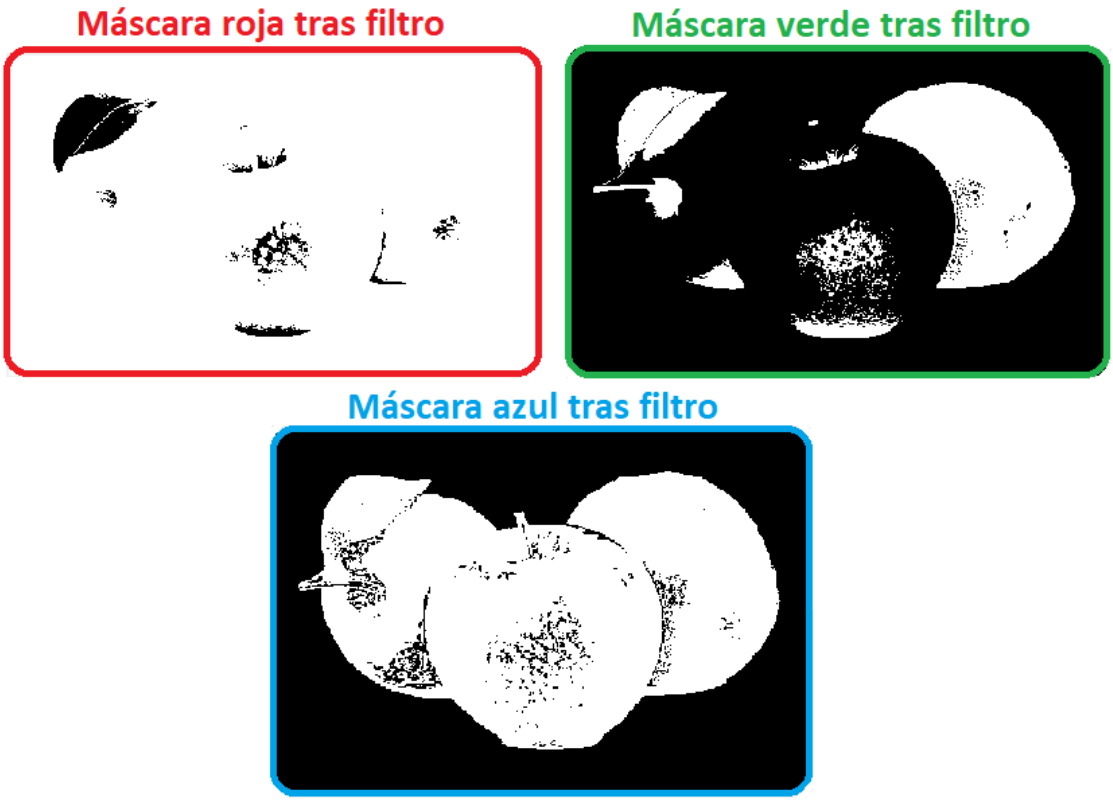


Figura 3-7 Resultado al aplicar filtro de color de manera independiente

El objetivo de realizar un filtro con las 3 máscaras es el de poder discernir un color en concreto usando la combinación de cada filtro. De esta manera si multiplicamos el valor de cada píxel todas los filtros entre sí, conseguiremos un nuevo filtro en el que solo sobrevivirán los píxeles que cumplan todos los requisitos impuestos.

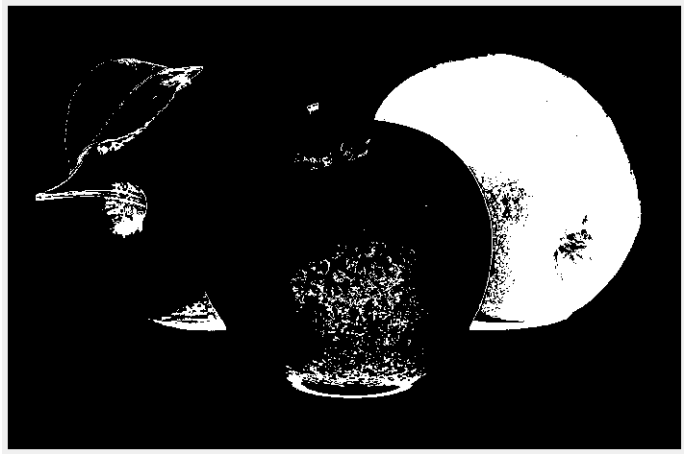


Figura 3-8 Resultado al aplicar filtro de color

Sin embargo en la práctica los objetos presentan muchas tonalidades y resulta imposible poder separar un objeto mediante filtros de color. Por lo que es necesario recurrir a métodos morfológicos para poder eliminar del filtro final de color todos aquellos elementos que han conseguido sobrevivir al filtro y sin embargo no pertenecen al objeto. De igual manera se pueden incluir los píxeles que pertenecen al objeto en cuestión a separar y que no han pasado el filtro.

Para dicho filtro, que podemos considerar filtro de ruido, se emplean 2 funciones:

- Close: Consiste en una acreción seguida de una erosión. Es una operación morfológica usada para rellenar los huecos o ruido negro de figuras con siluetas definidas. Si se usa en exceso puede hacernos perder la silueta del objeto en concreto.
- Open: Consiste en una erosión seguida de una acreción. Es una operación morfológica usada para eliminar motas o ruido blanco sobre un fondo. Si se usa en exceso puede crear nuevos objetos en el fondo a partir del pequeño ruido existente.

Dichos métodos morfológicos se basan en las operaciones básicas de erosión y acreción. En ambas operaciones se les pasa un elemento o matriz que sirve de modelo para realizar las transformaciones pertinentes sobre la imagen. En el caso de la acreción se busca para cada elemento de la imagen el valor máximo para los valores consultados en la imagen dependiendo de la posición actual y del elemento modelo que se use, y lo sustituye por el actual (Para la erosión es el mínimo). Para un mejor entendimiento de cada operación se puede observar la siguiente figura:

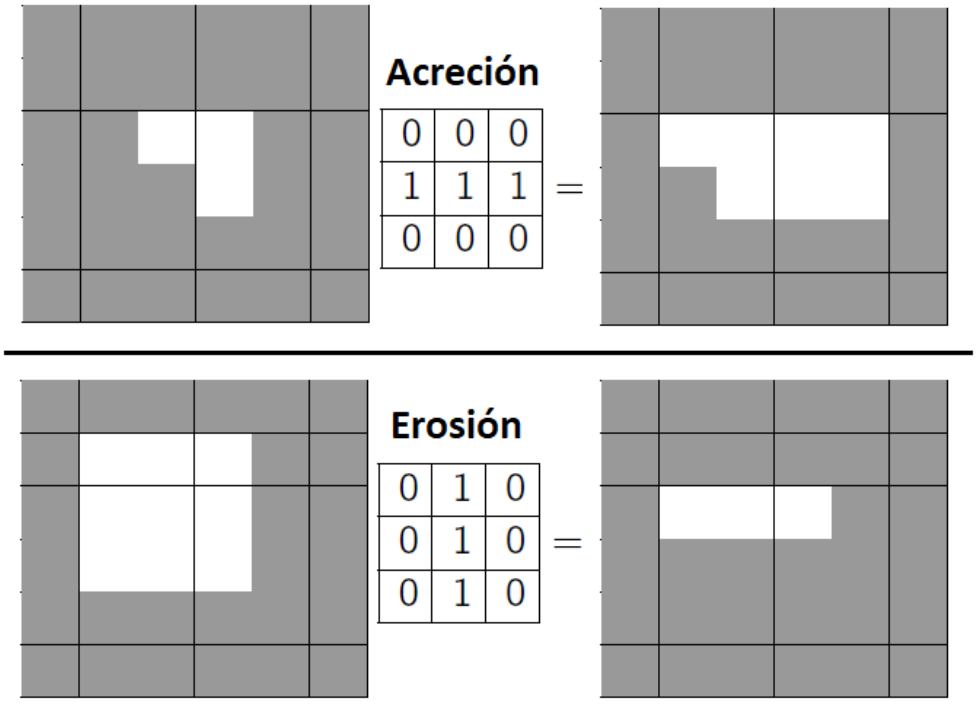


Figura 3-9 Ejemplo de aplicación de Acreción y Erosión

Mediante estas dos funciones podemos ir adecuando el filtro en varias etapas más suaves o menos etapas más fuertes para tener al final un resultado decente. No es un resultado exacto ya que hay partes de la manzana correspondientes al borde que se han perdido, pero se ha conseguido separar la manzana roja exitosamente.

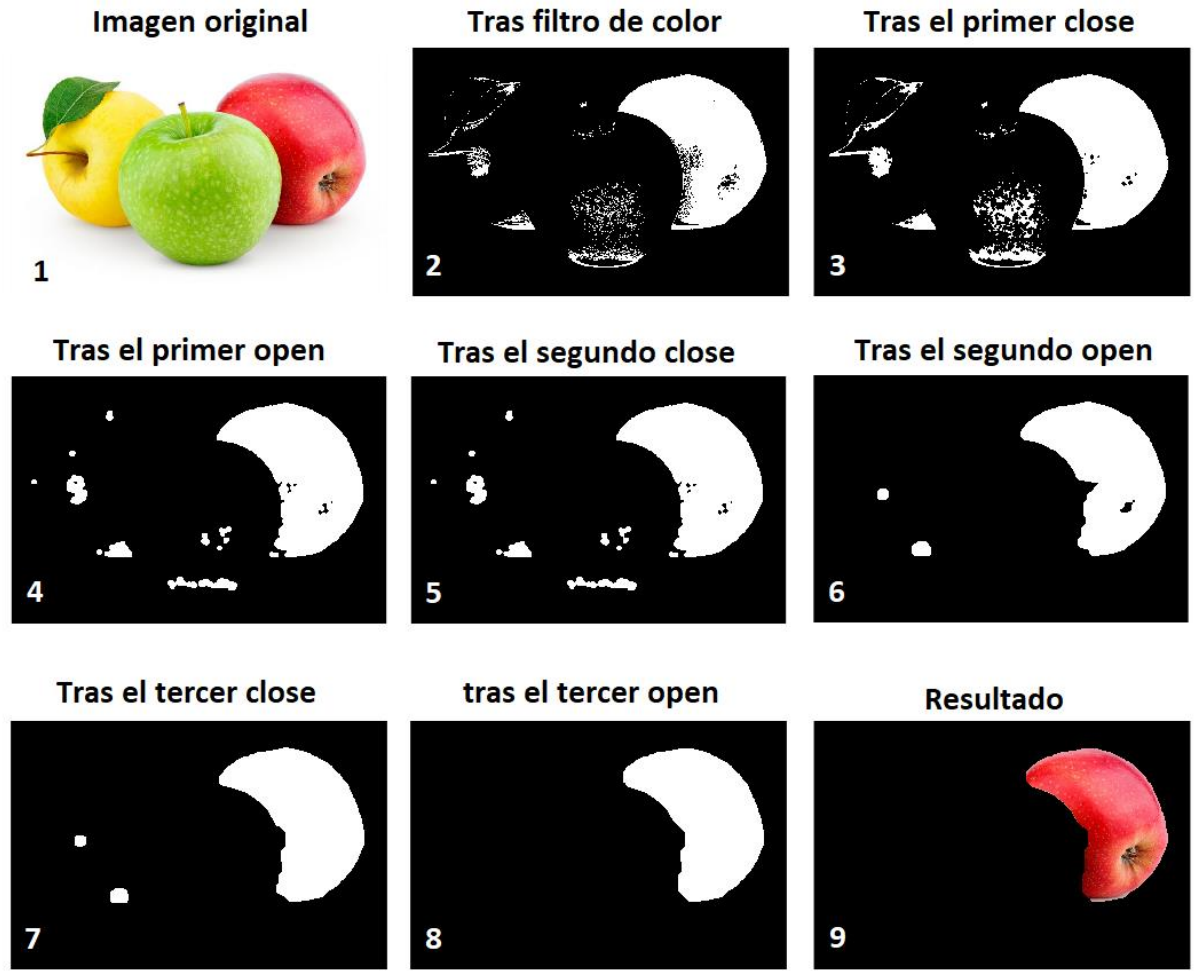


Figura 3-10 Aplicación de filtros de ruido

En el ejemplo se ha realizado varias veces un filtro contra el ruido hasta que hemos considerado que se había conseguido dar con éxito la separación de la manzana. Sin embargo en la práctica, necesitamos definir un único filtro de ruido o varios pero con las etapas definidas ya que se pretende que el tratamiento de imágenes sea autónomo [24].

Por ello, es normal que nos encontremos en la pantalla un poco de ruido. Por lo que hay que proceder con la identificación del objeto teniendo en cuenta la posibilidad de encontrar ruido en la imagen.

Si el objetivo de este proyecto es diseñar el sistema de manera que pueda seguir un objetivo, podemos en cierta manera definir el objetivo. De esta manera si consideramos que el objetivo a seguir posee una pelota de un cierto color podríamos no solo definir de antemano un filtro de color adecuado al color de la pelota, sino que podríamos buscar en el filtro final elementos con forma circular y así poder identificar el objetivo en una imagen con posibles ruidos.

Esto no elimina la posibilidad de que el ruido pueda tener forma de circulo también y esto produzca cierto error de detección por parte del programa, pero para ello trataremos de elegir un color no presente en el campo de pruebas y así poder elegir el mejor filtro de color.

La librería OpenCV nos proporciona una función con la cual podemos identificar elementos en la imagen usando el método de la transformada de Hough. Es un método en el que se suele usar para contornos rotos o duplicados, Pero también para identificar elementos con dichos contornos. Primeramente realiza un gradiente para calcular el contorno y seguidamente realiza una votación de todas las posibles funciones a buscar que pasen por dicho punto. Siendo las funciones: rectas, círculos, elipses…

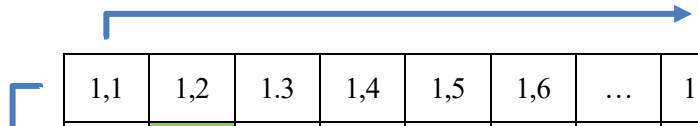
En nuestro caso buscaremos círculos, por lo que en cada elemento del contorno calculado anteriormente vota los posibles parámetros que forman el círculo (Centro (x,y), Radio r), y una vez calculados todos se determina la circunferencia con los parámetros más votados [25].

Por lo que después de todo el tratamiento de imágenes correspondiente, podemos disponer de la circunferencia obtenida con las posiciones en X e Y del centro y del radio de ésta. A partir de aquí, podemos utilizar dichos datos adecuándolos a nuestra conveniencia.

En nuestro caso:

- Usaremos el centro de la pantalla como referencia. Por lo que podemos usar la diferencia de píxeles entre el centro de la pantalla y el centro de la circunferencia. Dado que mediante los servos podemos hacer que la cámara gire en dos ejes, descompondremos la distancia de los centros en distancias verticales y horizontales.

  Durante todo el código, las imágenes se guardan y se tratan como matrices. Por lo que los índices de cada píxel se comportan como tal:



1,1	1,2	1.3	1,4	1,5	1,6	…	1,M
2,1	2,2	2,3	2,4	2,5	2,6	…	2,M
3,1	3,2	3,3	3,4	3,5	3,6	…	3,M
4,1	4,2	4,3	4,3	4,5	4,6	…	4,M
5,1	5,2	5,3	5,4	5,5	5,6	…	5,M
6,1	6,2	6,3	6,4	6,5	6,6	…	6,M
…	…	…	…	…	…	…	…,M
N,1	N,2	N,3	N,4	N,5	N,6	N,…	N,M

Tabla 3 – Distribución de píxeles en matriz

  Si suponemos que la imagen consta de 8 filas y 6 columnas, el centro de la imagen será el píxel 4,6. Suponiendo que el tratamiento de imágenes nos indique que la posición del centro de la circunferencia esté en la posición 2,2 podremos identificar que el objetivo se encuentra a (6-2=4) píxeles arriba y (4-2=2) píxeles a la izquierda. Por lo que podríamos modificar los servos para que la cámara se pueda alinear con el centro de la circunferencia [24].

- El radio de la circunferencia que el tratamiento de imágenes calcula, también puede aportarnos información acerca del área del objetivo. Aunque en el sistema formado por servos resulta inútil, en un sistema donde se pueda modificar la lejanía respecto al objetivo puede suponer una medida de lo cerca (radio mayor) o lejos (radio menor) que se encuentra el objetivo respecto a la cámara [25].

3.5 Programación: PID

El PID constituye un método de control muy usado para sistemas lineales en todo tipo de aplicaciones. Se trata de un control muy robusto además de simple, dichas características las consigue mediante una realimentación del estado final para tenerlo en cuenta en el propio control (control de lazo cerrado). En adición, un PID se

puede descomponer en varios controles en paralelos que tienen en cuenta diferentes aspectos de control, dándole la capacidad de poder adaptarse a cualquier tipo de situación.

Un controlador PID se puede modelar de la siguiente manera [26]:

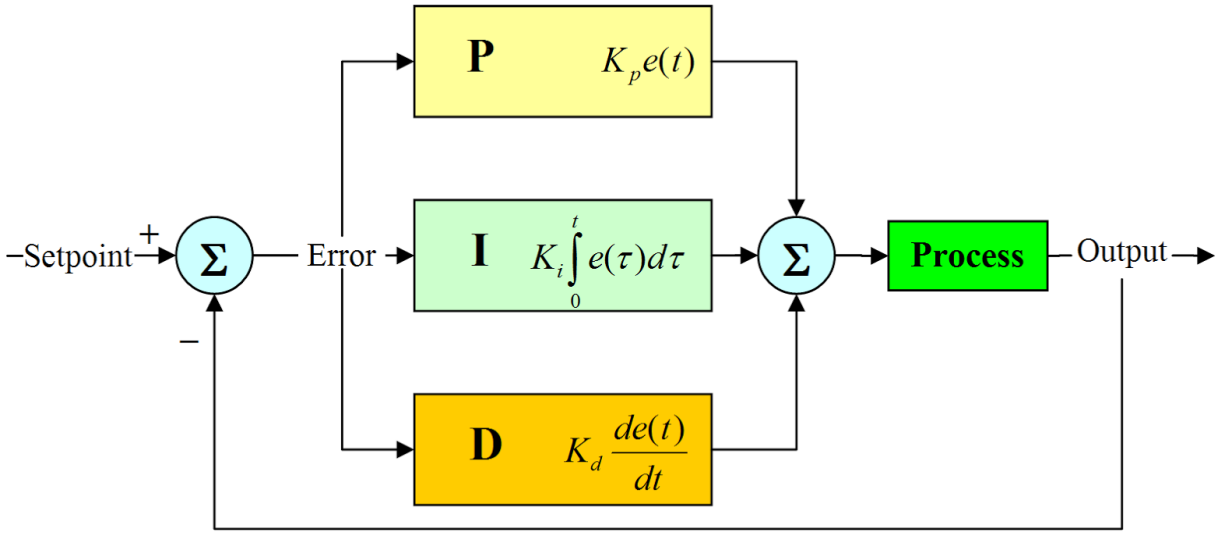


Figura 3-11 Modelo de un controlador PID

Cada rama del PID afecta al sistema de manera diferente, atendiendo a los valores que definen cada rama:

- Término proporcional (Kp): Este valor está relacionado con la velocidad o fuerza en la que el sistema cambia de estado para dirigirse a uno nuevo. Cuando Kp es muy pequeña el sistema no es capaz de alcanzar el nuevo estado, pero si Kp es muy grande irá con tanta fuerza el nuevo estado que se producirá una sobre oscilación.
- Término integral (Ki): Esta rama del controlador tiene en cuenta el historial de errores del sistema para poder estudiar su evolución y así ser capaz de corregirlo. El término integral corrige el error que el proporcional no puede solucionar, pero si se hace muy grande puede crear muchas sobre oscilaciones alrededor del punto objetivo.
- Término derivativo (Kd): En esta rama se intenta realizar una anticipación para intentar evitar sobre oscilaciones u otras variaciones abruptas que puedan ser nocivas. Para ello se hace una comparación del error actual con el pasado para obtener una idea de cómo seguirá respondiendo el sistema. Kd representa la fuerza con la que el controlador se anticipará ante tales variaciones.

Sin embargo el PID que necesitamos implementar en el programa no puede ser un PID en el dominio continuo, debe de ser discreto ya que la toma de imágenes no es continua. Necesitamos que el controlador se ejecute cada vez que se tome una captura y se analice, por lo que hay que recurrir a otra forma de implementar un PID pero en el dominio discreto.

Para ello disponemos de la siguiente fórmula:

$$S(k) = Kp * e(k) + Ki * I(k) + Kd * (e(k) - e(k-1))$$

Siendo:

- Kp: término proporcional.
- Ki: término integral.

- Kd: término derivativo.
- S(k): salida del PID de la iteración k.
- e(k): entrada al PID de la iteración k.
- I(k): historial de todos los valores de entradas hasta la iteración k.

$$I(k) = \sum_{i=0}^{k} e(i)$$

En nuestro caso, el PID implementado controlará la posición angular de los servos. Esto provocará que la cámara rote y por tanto que la imagen a tratar cambie. Lo que proporcionará nueva información al PID realimentándolo tal y como lo hemos visto anteriormente.

Dado que este apartado consiste simplemente en una prueba para poder implementar un PID y comprobar su buen funcionamiento junto al tratamiento de imágenes, no se implementará el control completo en el que se usan los dos servos para el seguimiento del objetivo. Se usará únicamente el servo que controla el grado ce inclinación de la cámara en el eje Y y se implementará un único PID para su control. Como anotación, si se quisiera implementa el control respecto al segundo servo, el código es exactamente el mismo, se ha obviado para el mejor sintonizado y mejores resultados experimentales del control del primer servo.

Para poder controlar el servo mediante el PID, primeramente tenemos que entender cómo funciona el servo.

Un servo o servomotor, consiste en un motor DC que se implementa junto a unos engranajes y un sistema de control electrónico. La finalidad del servo no es la de conseguir velocidad de giro, sino ofrecer un par de resistencia en cierta posición angular, o llevar a cabo giros con un torque mayor.

El sistema electrónico, es capaz de controlar el servo ya que éste tiene un potenciómetro que le permite leer la posición de éste y así mediante dicha realimentación alcanzar de manera más eficaz dicha posición. La entrada al sistema de control consta de una señal PWM. La señal PWM es un modo de modulación de una onda cuadrada por ancho de pulsos. Es decir, que durante un periodo de tiempo, el sistema electrónico leerá el ancho de pulso de la onda cuadrada, y en su medida determinará una posición para el servo.

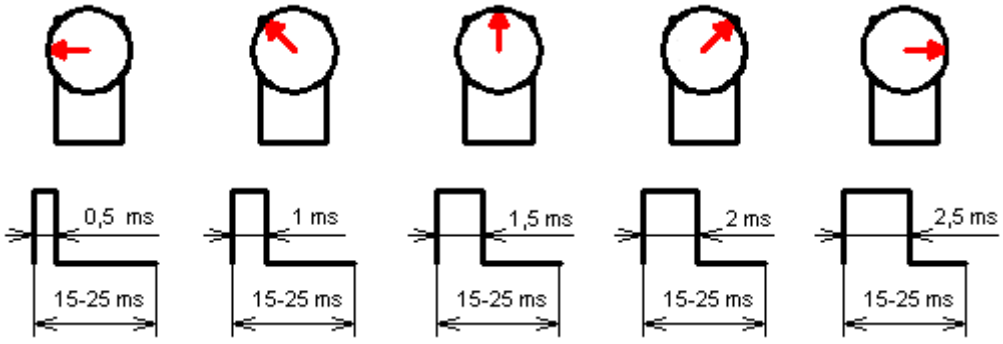


Figura 3-12 Relación entre pulsos y posición del servo

El periodo más normal es de 25 ms, de los cuales el ancho de pulso máximo será de 2,5 ms y un mínimo de 0,5 ms. Esto es así para poder asegurarnos de que dichos pulsos no se mezclen y leer mejor la señal de entrada.

Mediante Python, podemos enviar una señal PWM con el ancho de banda que queramos, por lo que dicho valor será la salida del PID. Sin embargo hay que tener una serie de consideraciones a la hora de asignar señales:

- Como acabamos de ver, hay unos máximos y mínimos anchos de pulsos que podemos mandar al

servo. Para ello saturaremos el valor de salida del PID siempre que calculemos la actuación.

- Para no sobrecargar el servo, denotaremos un rango de error para el cual el PID no actuará. En nuestro caso, denotaremos un rango de 15 píxeles para el cuál si el objetivo se encuentra en dicho rango, el PID no actuará, sin embargo en el momento en el que abandone dicho rango el PID volverá a actuar.

  Como solo implantaremos el PID en el eje X, podemos definir como la zona en la que el PID no actuará como: Valor de X del centro de la pantalla +- rango de seguridad.

- Dado que el sistema de reconocimiento del objetivo no es perfecto, habrá instantes en los que no exista objetivo, y por tanto no podamos usar el PID para tomar medidas al respecto. La mejor manera de actuar en esta situación será la de desactivar el PID ya que el servo seguirá girando mientras busca el objetivo ya que al no disponer de las nuevos parámetros del objetivo actualizado, el error no decrementa y la actuación por ende no se estabiliza.

Figura 3-13 Pruebas del PID con pelota

Por último, una vez que tenemos todo el código implementado es hora de sintonizar el PID. Para ello podemos ir probando en orden valores de Kp, Ki y Kd. De manera que podamos ver como el sistema va adaptándose a nuestra aplicación.

Sin embargo generalmente se espera de un PID que controle un aspecto en el menor tiempo posible, sin error, y sin sobre oscilaciones. Para ello hay un método de sintonización simple llamado el método Ziegler-Nichols.

Para ello tenemos que estudiar cómo responde el sistema ante un escalón cualquiera. Una vez tenemos la gráfica, se traza una tangente a la curva resultante del escalón. Los puntos donde tal recta alcance los valores de los estados inicial y final delimitan una variable de tiempo que llamaremos Ts. El tiempo desde que se inicia el escalón y el primer punto de la recta que hemos descrito anteriormente forma otra variable que llamaremos Td.

Una vez que tenemos dichas variables, según el método de Ziegler-Nichols los parámetros del PID serán:

$$Kp = 1.2 * \frac{Ts}{Td}$$

$$Ki = \frac{Kp}{2 * Td}$$

$$Kd = 0.5 * Kp * Td$$

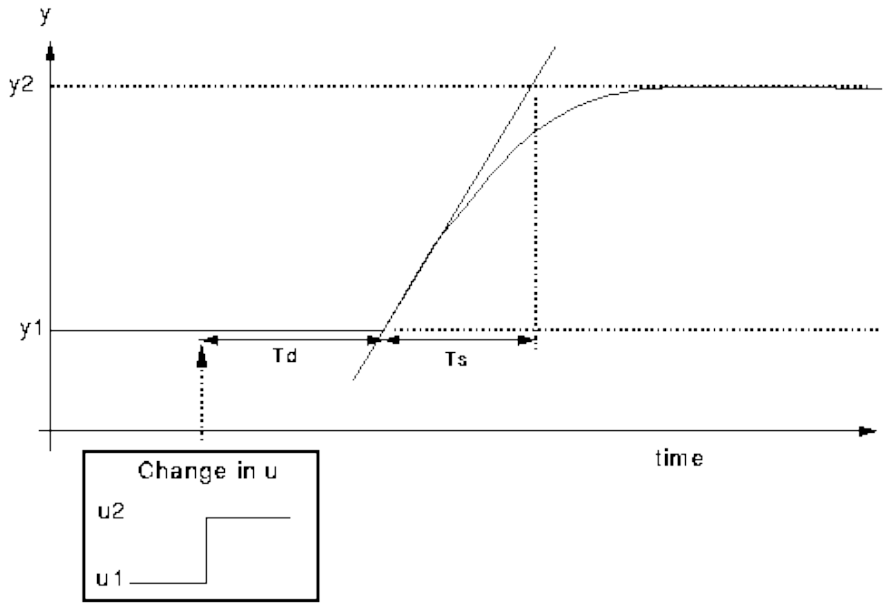


Figura 3-14 Representación de variables Td y Ts

Sin embargo, en nuestro proyecto no disponemos de servomotores electrónicos los cuales se puede conocer el estado real del servo en cualquier momento. Disponemos de unos servos mucho más simples y económicos que no disponen de tal función, por lo que es imposible realizar dicha gráfica.

En tal caso, se procede a sintonizar el PID de manera visual a base de prueba y error.

4 CONTROL DEL DRONE

4.1 Control final, Control en cascada

Respecto al control del drone, no se estudiará los modelos matemáticos de éstos ya que no es el objetivo del proyecto. Sin embargo se procederá a explicar los mecanismos necesarios para que a partir de la lectura de ciertos sensores, la controladora es capaz de elegir la mejor estrategia de control (voltaje asignado a cada motor) para alcanzar unos estados objetivos.

Por un lado tenemos un control de estabilidad. El drone necesita una gran estabilidad no solo para realizar las medidas de la radicación solar con fiabilidad, sino para no comprometer su estructura física además del vuelo en sí. Supone un control fundamental ya que es la encargada de responder ante perturbaciones como ráfagas de vientos o mecánicas de fluidos que suponen que las hélices no consigan el empuje supuesto.

Por otro lado tenemos un control de órdenes, necesario para la interpretación y ejecución de órdenes y el seguimiento de rutas. Para ello se busca un estado objetivo del drone con las características que definan el modo de vuelo que se esté ejecutando y la controladora a partir de los sensores disponibles como el módulo GPS o el barómetro determinará la acción que debe realizar el drone así como la transformación de éstos en voltajes sobre cada motor.

Dichos controles se encuentran ya definidos de fábrica con el modelo matemático correspondiente de manera que el usuario solo debe configurar los parámetros geométricos correspondientes al drone y las variables referentes a la sintonización del control para cada caso.

Entonces nuestro sistema al completo forma parte de dos subsistemas que contienen una finalidad propia con un control independiente, pero que relacionados consiguen alcanzar el objetivo final. Dicho concepto de control se llama Control en Cascada. El control en cascada se da cuando la salida de un controlador de realimentación es usada por un segundo controlador de realimentación, de esta manera el control de un sistema se puede descomponer en una serie de controladores con tareas independientes pero que conjuntamente alcanzan un objetivo común [27].

De manera esquemática, nuestro sistema se podría representar de la siguiente manera:

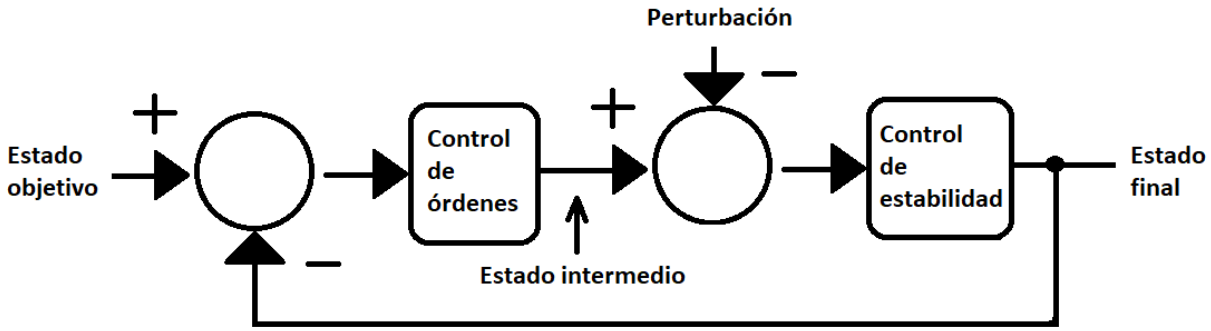


Figura 4-1Control correspondiente al sistema Drone-Raspberry

Recordando lo explicado en el primer capítulo, podemos notar que el sistema también se trata de un control en lazo cerrado y no solo el sistema en completo, sino cada control dispone de una realimentación independientemente que no se ha reflejado en el esquema pero que es fundamental para su funcionamiento. Este tipo de control se caracteriza por implementar la opción de que el control pueda considerar también la salida del sistema como información adicional para poder así ayudar a elegir la mejor estrategia de control. Dicha implementación se realiza realimentando la salida a la entrada negativamente y así conseguir una diferencia de estado, pero para ello tenemos que adecuar la salida transformándola en una información que podamos contrastar con la de entrada.

Este tipo de control es necesario para aplicaciones reales donde ocurren una serie de complicaciones técnicas:

- Los sensores no son perfectos, y nos dan una información discreta donde puede existir una tolerancia de error más o menos amplia en relación con la calidad de dichos sensores. Incluso también pueden aportar información defectuosa. De igual manera ocurren con los actuadores
- Los modelos matemáticos usados en el control no son un reflejo exacto del sistema real, por lo que puede dar a ciertas desviaciones en el objetivo final.
- En la práctica hay muchos otros aspectos que pueden actuar sobre el sistema que no se tienen en cuenta, incluso algunos pueden espontáneamente producir una perturbación en el sistema, dado como inválido el estado final de control.

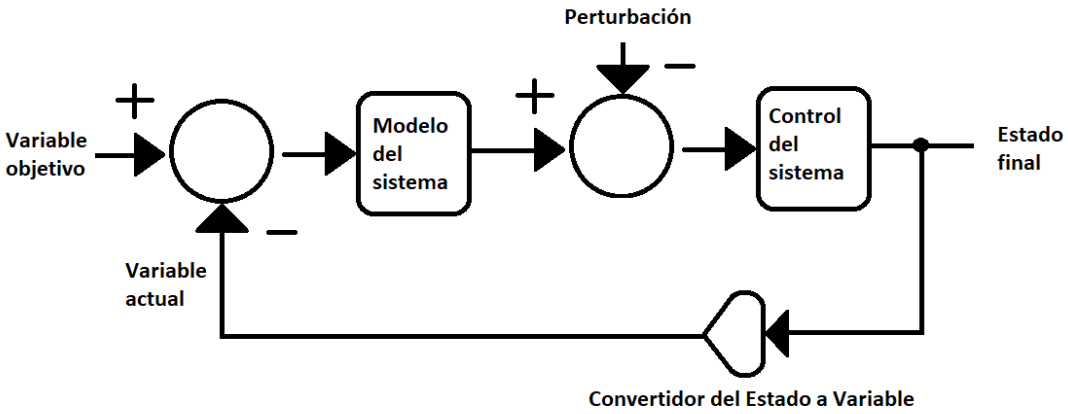


Figura 4-2 Control de lazo cerrado

Con el control en lazo cerrado, se pueden corregir estos problemas y adecuar el control para a pesar de todo los inconvenientes que se pueden producir, alcanzar un estado final satisfactorio [3].

4.2 Control del drone

4.2.1 Movimientos característicos de un drone

Para entender qué decisiones debe acometer la controladora para dar una estabilidad al sistema, vamos a describir los movimientos más importantes que definen el comportamiento del drone y como mediante la combinación de los cuatro motores disponibles es capaz de realizarlos.

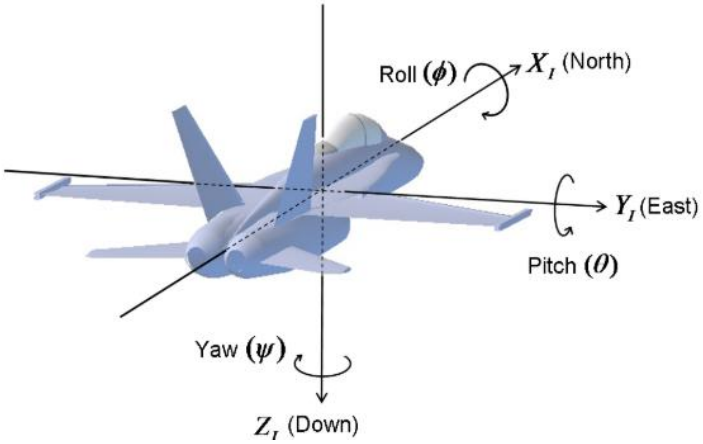


Figura 4-3 Ángulos de Euler en una aeronave [28]

Lo primero serán entender los ángulos de Euler correspondientes con el drone [12]:

- La rotación respecto al eje Z se denota como Yaw (ψ). Este movimiento se puede crear haciendo rotar con más intensidad los motores 1 y 3 mientras que reducimos un de igual manera los 2 y 4 o viceversa. De esta manera como ya se ha explicado en el apartado de construcción del dron se puede manipular el momento angular y conseguir aumentar el Yaw sin reducir el empuje neto. Corresponde con la guiñada de una aeronave.

  La matriz de rotación que denota dicha transformación sería:

$$R(\psi) = \begin{pmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

- La rotación respecto al eje X se denota como Roll (θ). Este movimiento se puede crear haciendo girar los motores del lado derecho más rápido que los izquierdos. De esta manera se puede hacer inclinar el drone hacia la izquierda ya que la parte derecha tiene más impulso o viceversa. Corresponde con el alabeo de una aeronave y de así podemos desplazar el drone en el eje Y.

  La matriz de rotación que denota dicha transformación sería:

$$R(\theta) = \begin{pmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{pmatrix}$$

- La rotación respecto al eje Y se denota como Pitch (ϕ). Este movimiento se puede crear haciendo girar los motores del lado trasero más rápido que los delanteros. De esta manera se puede hacer inclinar el drone hacia delante ya que la parte trasera tiene más impulso o viceversa. Corresponde con el cabeceo de una aeronave y así podemos desplazar el drone en el eje X.

  La matriz de rotación que denota dicha transformación sería:

$$R(\phi) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & \sin(\phi) \\ 0 & -\sin(\phi) & \cos(\phi) \end{pmatrix}$$

Al tratar un drone de ala rotatoria, podemos conseguir un tipo de movimiento no característico de las aeronaves de ala fija, aunque no pertenezca a los ángulos de Euler:

- El desplazamiento a lo largo del eje Z se denota como Throttle. Este movimiento se puede crear haciendo girar todos los motores del drone más rápido para conseguir más empuje y así alcanzar mayor altura o viceversa.

4.2.2 Controlador implementado en Ardupilot

Dentro de Mission Planner solo tenemos la opción de configurar los parámetros de control referentes a los distintos modos de vuelo. Sin embargo es necesario estudiar la implementación de éste para poder sintonizarlo si fuera necesario una modificación [29]. En la implementación de control por parte de Ardupilot optan por usar un filtro de Kalman sumado a un PID.

El filtro de Kalman se puede considerar como un procedimiento matemático que opera por medio de un mecanismo de predicción y corrección. Lo que hace el filtro es predecir el nuevo estado a partir de una estimación previa añadiendo un término de corrección proporcional al error de predicción para minimizarlo.

Tiene su origen en el documento de Kalman (1960) donde da una solución recursiva al problema de filtrado lineal de datos discretos. Todo en un contexto de modelos de estado de espacios donde se estima por mínimos cuadrados.

Es muy usado en sistemas dinámicos ya que puede evitar la influencia de cambios estructurales en la estimación. La estimación usada parte de una muestra inicial y las actualiza incorporando sucesivamente una nueva observación con todos los datos disponibles. Utiliza el método de mínimos cuadrados para ir generando recursivamente un estimador de estado en cada momento, que es lineal, insesgado y de varianza mínimo. De esta manera puede predecir el estado de un modelo y adaptarse incluso a modelados de sistemas poco precisos.

Dado que es un método en el que se tiene en cuenta el historial pasado de estados y variables, depende fundamental de ello. Por lo que puede volverse difícil establecer unas condiciones iniciales en las que se pueda vasar el filtro.

A pesar de todo ello, el filtro de Kalman es un método con una enorme capacidad de poder resolver todo tipo de problemas, incluso con inferencias estadísticas.

Las ecuaciones generales que componen el filtro son las siguientes:

$$x_k = \xi_{k-1}(x_{k-1}) + w_{k-1} \qquad w_k \sim \aleph(0, \mathbf{Q}_k) \qquad \text{Modelo Dinámico no Lineal}$$

$$z_k = h(x_k) + v_k \qquad v_k \sim \aleph(0, \mathbf{R}_k) \qquad \text{Modelo de Medición no Lineal}$$

Etapa de predicción

$$\hat{x}_k^{(-)} = \mathbf{\Phi}_{k-1}\hat{x}_{k-1}^{(+)} \qquad \mathbf{\Phi}_{k-1} = \frac{\partial \xi_{k-1}(\hat{x},k)}{\partial x}|_{x=\hat{x}_{k-1}} \qquad \text{Predicción a } \textit{“priori”}^{(-)} \text{ de } x_k$$

$$\mathbf{P}_k^{(-)} = \mathbf{\Phi}_{k-1}\mathbf{P}_{k-1}^{(+)}\mathbf{\Phi}_{k-1}^T + \mathbf{Q}_{k-1} \qquad \text{Predicción a } \textit{“priori”}^{(-)} \text{ de } \mathbf{P}_k$$

Etapa de corrección

$$\hat{z}_k = \mathbf{H}_k\hat{x}_k^{(-)} \qquad \mathbf{H}_k = \frac{\partial h(\hat{x},k)}{\partial x}|_{x=\hat{x}_{k-1}} \qquad \text{Linealización de la medición}$$

$$\mathbf{K}_k = \mathbf{P}_k^{(-)}\mathbf{H}_k^T[\mathbf{H}_k\mathbf{P}_k^{(-)}\mathbf{H}_k^T + \mathbf{R}_k]^{-1} \qquad \text{Ganancia de Kalman}$$

$$\hat{x}_k^{(+)} = \hat{x}_k^{(-)} + \mathbf{K}_k[z_k - \mathbf{H}_k\hat{x}_k^{(-)}] \qquad \text{Obtención a } \textit{“posteriori”}^{(+)} \text{ de } x_k$$

$$\mathbf{P}_k^{(+)} = [\mathbf{I} - \mathbf{K}_k\mathbf{H}_k]\mathbf{P}_k^{(-)} \qquad \text{Obtención a } \textit{“posteriori”}^{(+)} \text{ de } \mathbf{P}_k$$

Figura 4-4 Ecuaciones generales del filtro de Kalman [31]

Siendo:

- X(k) el estado.
- Z(k) la medición.
- P(k) la matriz de covarianza del error estimada.
- Q(k) y R(k) las matrices de acoplamiento del ruido tanto del modelo como de los sensores.

- K(k) la matriz de ganancia de Kalman.

Tal como sugiere la tabla de ecuaciones, las fases de este filtro se pueden descomponer en 3 fases. En una primera fase el modelo se inicializa. Una vez tenemos el estado predice el siguiente el siguiente estado a partir de la información disponible. Antes de predecir el siguiente estado se corrigen la influencia de la información disponible a partir de la veracidad de la predicción realizada y así ir adaptándose a cualquier cambios.

En cuanto al método de cómo se implementa el filtro de Kalman en la controladora, de manera simplificada:

- Se calcula el estado del sistema a partir de una serie de mediciones, algunas de ellas pueden ser:
  i. Orientación angular a partir de la velocidad angular que siente el sistema.
  ii. Posición en el espacio a partir de la velocidad del sistema, que a su vez se calcula a partir de la aceleración sentida por el sistema teniendo en cuenta la gravedad.
  iii. Posición en el espacio en referencias de longitud y altitud por medio de las lecturas de los satélites por triangulación a partir del módulo GPS.
  iv. Altura a partir de valores que proporciona el barómetro.

  Al final obtenemos el estado del sistema correspondiente a los ángulos de Euler (roll, pitch, Yaw), altura, posición en el espacio, orientación magnética.

  Para su posterior uso, también se mide el sesgo de los sensores de medida como los acelerómetros, giroscopios, barómetros, brújula magnética y GPS.

  Esta etapa corresponde con la fase de predicción de estado.

- Se usa los ruidos en los giroscopios y acelerómetros así como los errores de diferencia en los estados objetivos y actuales como el de la posición de GPS para estimar el crecimiento de error de las variables que constituyen el estado del sistema. De esta manera podemos constituir la matriz de covarianza del error estimado, así como las de acoplamiento de ruido.

  Y con estos datos podemos ir variando la matriz de ganancia para ir adaptando la estimación siguiente estimación.

  Dicha etapa corresponde con la fase de corrección.

Dichas fases se irán sucediendo constantemente, realizando un control adaptativo en todo momento [30] [31].

Sin embargo para el control de estabilidad es necesario implementar un segundo elemento de control que pueda adaptar la velocidad de los motores acorde con los parámetros que el filtro de Kalman nos proporciona. Para ello se implementa un PID (Proporcional Integral Derivativo).

4.2.3 Calibración de sensores

Una correcta calibración de los sensores es totalmente imprescindible antes de volar el drone, ya que se trata de un sistema de control que nos permite cierta autonomía en aspectos de vuelo y sin ellos el drone se puede volver totalmente incontrolable.

Para ello hay una serie de pasos en los que el software Mission Planner va calibrando algunos de los sensores más importantes usados en el vuelo.

1. Primeramente el drone deberá de estar conectado por USB desde la controladora hasta el ordenador y desde Mission Planner realizar la conexión.
2. Como primera configuración se elige el tipo de cuerpo que constituye el drone. En nuestro caso tenemos un cuerpo tipo X, pero hay otros más disponibles atendiendo al número de motores y la distribución de éstos a lo largo del cuerpo del drone.
3. La siguiente calibración es muy importante, ya que constituye la calibración del acelerómetro. Dicho sensor es el encargado de proporcionar al drone la información referente a los ángulos del drone en cada momento. Por lo que si se desea que el drone adquiera cierta estabilidad, es de vital importancia realizar un buen calibrado en superficies rectas para que el drone pueda disponer de dicha información

en cualquier momento. Un caso de mal calibrado puede provocar que el drone a pesar de estar controlando bien cada motor, adquiera una deriva en el plano horizontal ya que para sí mismo se encuentra estabilizado pero en la realidad se encuentra inclinado produciendo un movimiento.

Para calibrar el sensor, basta con colocar el drone en las distintas posiciones que se muestran en pantalla y confirmarlas pulsando cualquier botón.

4. Otro sensor importante a calibrar es el compás, ya que le proporciona al drone la información acerca de la dirección en la que el drone se encuentra alineado. Dependiendo de la aplicación este sensor adquiere menor o mayor importancia, en el nuestro es también imprescindible ya que proporciona información muy útil en el seguimiento de puntos en el mapa de manera autónoma.

   Para calibrarlo, hay que hacer rotar el drone 360 grados en cada uno de los 6 ejes, tanto en un sentido como en el otro.

5. A continuación, indicaremos si disponemos de un Power Module para la monitorización de la batería, indicando el tipo que disponemos. Además también debemos de aportar información acerca del tipo de batería que usaremos en el drone.

6. Lo siguiente es la calibración de la emisora. La calibración no es una configuración de la conexión, ese paso es anterior a éste ya que si no está conectada con el drone la emisora no se puede calibrar. Entonces, una vez calibrada, procederemos a llevar los joysticks a las posiciones extremas para que así la controladora conozca qué valores son los máximos y mínimos de cada canal.

7. Por último, se configura de manera preventiva la distribución de modos de vuelo que se instalarán en la controladora de vuelo, y el failsafe.

   El failsafe consiste en un procedimiento por el cual, cuando la controladora detecte niveles críticos de batería, procederá a ejecutar el programa failsafe. Dicho programa puede consistir en una vuelta autónoma a casa, o un aterrizaje suave en el sitio, o simplemente no hacer nada. Además se puede configurar el nivel de batería en el que saltará el modo failsafe.

4.2.4 Modos de vuelo y sintonización del control

Dependiendo del modo de vuelo que seleccionemos, la controladora ejecutará un control sobre el drone u otro. El modo en el que este control afecte al modo de vuelo es configurable desde el software Mission Planner así como la sensibilidad del drone a realizar las órdenes que les llegue desde la emisora.

Existen varios modos de vuelo atendiendo a la aplicación a usar. En cada modo de vuelo se hace uso de diferentes módulos de sensores que dispone la controladora.

Los diferentes modos de vuelo que ofrece Ardupilot son los de la figura 4-7.

Para dar un idea del proceso por el cual se ha sintonizado correctamente el drone en varias etapas, se procederá a indicar los modos de vuelo que se han usado a lo largo de todo el proyecto y los parámetros que se han cambiado.

También existe un control que afecta en menor o mayor medida a los modos de vuelo, y es el control implementado para la estabilidad del drone en el aire. Configurable en el panel 1 de la figura 4-5. Los modos de vuelo son:

- Modo de vuelo **STAVILIZE**.

  En este modo de vuelo tenemos un control semi total del drone. La controladora únicamente implementará un control de corrección en cuando el usuario no esté controlando el Roll y el Pitch de manera que sea más fácil de estabilizar. También impone unos límites en dichos grados de Euler de manera que el usuario no pueda sobrepasarlos.

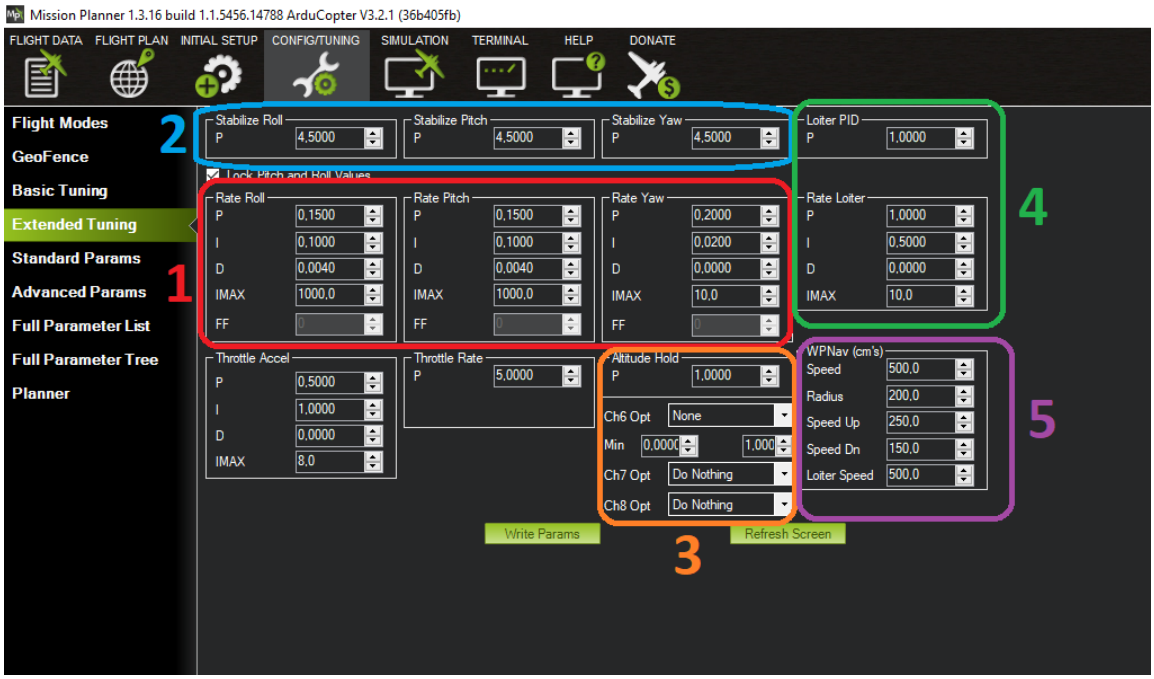


Figura 4-5 Panel de control para sintonizado de controladores del drone

Se trata de un modo de vuelo muy agresivo en el que requiere de mucha práctica y soltura para poder pilotar adecuadamente el drone ya que el usuario debe de controlar los 4 aspectos del movimiento del drone de manera simultánea. Este tipo de modo de vuelo tiene su aplicación en situaciones donde se requiera de una rápida respuesta, como por ejemplo en carreras de drones.

En nuestro proyecto se usará dicho modo de vuelo al principio, durante las pruebas ya que en cuando el control de estabilidad no funcione correctamente, podremos pasar a este modo de vuelo y así evitar derivas que desemboquen en algún accidente.

Los módulos que se usan en este modo de vuelo son: acelerómetros y giroscopios. Por lo que una buena calibración anterior al vuelo es imprescindible.

El panel en la figura 4-5 podemos configurar el control que afecta a este modo en el panel 2.

- Modo de vuelo **ALT HOLD**.

  En este modo de vuelo se parte del modo de vuelo anterior. Posee todas las características del modo de vuelo STAVILIZE y además añade un control de altura. Esto significa que el usuario puede controlar la altura del drone mediante el Joystick asignado al Throttle pero una vez lo coloque en posición neutral el drone mantendrá dicha altura. También se le puede asignar una altura máxima a la cual el drone no podrá sobrepasar.

  A pesar de todo el dron sigue teniendo la posibilidad de sufrir una deriva en el plano horizontal causadas por un error de estabilidad en el Roll o en el Pitch, o causada por ráfagas de viento.

  En nuestro proyecto se usará dicho modo de vuelo para probar la estabilización del sistema en cuando a PID. Esto quiere decir que ya podemos hacer pruebas en los que determinaremos los proporcionales correspondientes al Roll, el Pitch, y el control de altura.

  Los módulos que se usan en este modo de vuelo son: acelerómetros, giroscopios y el barómetro. Además de la calibración inicial, también hay que asegurarse de tener encapsulada la controladora para que las ráfagas de viento no afecten al barómetro. Si no se dispone del encapsulado Ardupilot aconseja colocar un trozo de cinta de celo encima de la placa para solventar el problema.

El panel en la figura 4-5 podemos configurar el control que afecta a este modo en el panel 3

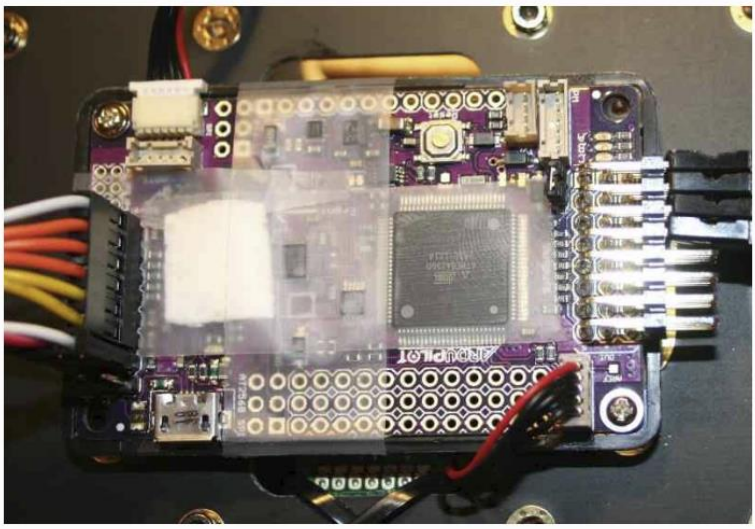


Figura 4-6 Remedio casero para el barómetro según Ardupilot.org

- Modo de vuelo **LOITER**.

  En este modo de vuelo, al igual que modo de vuelo ALT HOLD, lo usaremos para configurar el PID correspondiente al control del drone en cuanto a la posición en el plano horizontal. Esto es muy importante ya que el sensor que usaremos en adición en este modo de vuelo será el módulo GPS. Esto dejará lista la configuración necesaria en el drone para su uso en el siguiente modo de vuelo.

  Cuando el usuario decida modificar alguno de los 4 movimientos del drone descritos anteriormente, la controladora dejará de controlar dicha variable hasta que el usuario suelte los controles. De esa manera se puede controlar el drone en todos sus aspectos dentro de un límite pero con la seguridad de que una vez soltados los mandos el drone permanecerá en el mismo lugar sin sufrir ningún tipo de deriva. A pesar de cualquier perturbación ocasionada por ejemplo del viento.

  Los módulos que se usan en este modo de vuelo son: acelerómetros, giroscopios, barómetro, compás magnético y el módulo GPS. Además de la calibración inicial, también hay que asegurarse de realizar una exhaustiva del PID dada la complejidad del control de la posición del GPS.

  El panel en la figura X.X podemos configurar el control que afecta a este modo en el panel 4.

- Modo de vuelo **AUTO**.

  En este modo de vuelo, se utiliza la configuración del modo de vuelo LOITER para hacer cumplir los objetivos de una ruta ya definida. En dicha ruta o misión, se definen unos puntos en el espacio por los cuales el drone debe de pasar y quedarse un tiempo definido en cada punto. Esto ya no corresponde con el control de estabilidad sino del control de órdenes.

  De esta manera, aplicada al objetivo del proyecto, podríamos definir una ruta en la que el drone pueda recorrer todas las instalaciones de la planta correspondientes a los paneles de espejos, y pararse durante un tiempo determinado en cada zona de interés para realizar unas medidas.

  Por lo que una vez lleguemos a este modo de vuelo, además de tener todo el drone ya configurado tendremos que haber definido con anterioridad la misión mediante el software Mission Planner.

  El panel en la figura X.X podemos configurar el control que afecta a este modo en el panel 5.

Mode	Alt Ctrl	Pos Ctrl	GPS	Summary
Acro	-	-		Holds attitude, no self-level
Alt Hold	s	+		Holds altitude and self-levels the roll & pitch
Auto	A	A	Y	Executes pre-defined mission
AutoTune	s	A	Y	Automated pitch and bank procedure to improve control loops
Brake	s	A	Y	Brings copter to an immediate stop
Circle	s	A	Y	Automatically circles a point in front of the vehicle
Drift	-	+	Y	Like stabilize, but coordinates yaw with roll like a plane
Flip	A	A		Rises and completes an automated flip
FlowHold	s	A		Position control using Optical Flow
Follow	s	A	Y	Follows another vehicle
Guided	A	A	Y	Navigates to single points commanded by GCS
Land	A	s	(Y)	Reduces altitude to ground level, attempts to go straight down
Loiter	s	s	Y	Holds altitude and position, uses GPS for movements
PosHold	s	+	Y	Like loiter, but manual roll and pitch when sticks not centered
RTL	A	A	Y	Retruns above takeoff location, may aslo include landing
Simple/Super Simple			Y	An add-on to flight modes to use pilot's view instead of yaw orientation
SmartRTL	A	A	Y	RTL, but traces path to get home
Stabilize	-	+		Self-levels the roll and pitch axis
Sport	s	s		Alt-hold, but holds pitch & roll when sticks centered
Throw	A	A	Y	Holds position after a throwing takeoff

Symbol	Definition
-	Manual control
+	Manual control with limits & self-level
s	Pilot controls climb rate
A	Automatic control

Figura 4-7 Modos de vuelo que aporta la controladora APM 2.6 [32]

4.2.5 Configuración de la ruta en Mission Planner

La ruta que el drone seguirá de forma autónoma está compuesta por puntos en el espacio definidos, de manera que el drone pueda realizar un control de posición en relación con la posición actual y la objetivo. Dicha secuencia de puntos en el espacio forman una ruta que el drone seguirá en orden proporcionando el vuelo autónoma por los puntos de interés, sin embargo es un modo de vuelo autónomo muy básico ya que tiene únicamente en cuenta la posición en GPS sin tener en consideración elementos físicos en el camino como muros, árboles, pilares, aves…

Para ello, el software Mission Planner nos proporciona un mapa en el que podemos ir seleccionando los puntos de la ruta además de varias opciones a la hora de que el drone realice ciertas operaciones. Una consideración importante a la hora de realizar la ruta es la de asignar adecuadamente la posición de home o retorno, ya que si el drone detecta que se ha alcanzado un nivel de batería crítico o ha perdido la señal de la emisora, procederá a volver a dicho punto de manera autónoma por seguridad. Dado que en dicho estado no se podrá controlar el drone es de vital importancia definirlo para no extraviar el drone o evitar posibles accidentes.

Dada nuestra aplicación, no se utilizarán todas las órdenes disponibles. Las usadas en este proyecto serán:

- TAKEOFF: Mediante esta orden el drone irá suministrando poco a poco voltaje de igual manera a todos los motores para poder despegar de manera suave. Dicha orden viene asociada a una posición en el espacio definida por una latitud, longitud, y altura. Hasta que el drone no alcance dichos valores no avanzará en el código. La manera adecuada de posicionar dicha posición es definirla en la posición donde el drone vaya a despegar, ya que si la posición se encuentra a una distancia no hay seguridad en que durante el camino el drone evite cualquier obstáculo.
- WAYPOINT: Mediante esta orden el drone se dirigirá al punto definido en esta orden. Se pueden realizar varias operaciones de este tipo para ir aumentando o disminuyendo la altura, o para moverse en el plano horizontal o ambas. Si se quisiera realizar una ruta en un entorno con varios obstáculos en la zona lo más sensato sería usar varías operaciones consecutivas para ir variando solo la altura o posición en el plano para definir de mejor manera la ruta en el espacio. Si no fuese el caso, con una sola operación el drone es capaz de alcanzarla de manera más eficiente. Una vez que alcance dicha posición el drone avanzará en el código.
- LOITER_TIME: Dado que el drone necesita disponer de un tiempo para realizar las medidas de radiación solar en los puntos de interés, necesitamos que el drone se mantenga estático en el espacio. Para dicha situación usaremos esta orden en la que el drone permanecerá estático en el punto definido usando el modo de vuelo loiter anteriormente definido. De esta manera el drone permanecerá un tiempo definido en la operación y contrarrestará cualquier perturbación para mantener dicho punto en el espacio.
- LAND: Mediante esta orden el drone irá descendiendo hasta la posición definida en la orden de manera suave. La velocidad de aterrizaje irá disminuyendo a medida que baje la altura y así poder asegurar la integridad física del aparato. De igual manera que se ha explicado en el comando TAKEOFF, para mayor seguridad es necesario definir un punto de aproximamiento a la zona de aterrizaje para que el drone solo tenga que adecuar la altura y no la posición en el plano horizontal y así evitar posibles trayectorias con obstáculos.

El conjunto de puntos que define tanto la trayectoria como las acciones a desarrollar lo definiremos como misión. Para el objetivo del proyecto, además de todas las consideraciones referentes a trayectorias seguras en los despegues y aterrizajes, se puede definir un WAYPOINT en cada zona de espejos en la que se quiera disponer de una medida de radiación, seguido de un LOITER_TIME con un tiempo suficiente para que el sistema de medición de radiación pueda realizar una medida. A modo de ejemplo, en el programa podría visualizarse la misión de la manera mostrada en la figura 4-8.

Las misiones se escriben en la memoria de la controladora de vuelo, y solo puede existir una misión escrita en la controladora. De igual manera se puede leer la misión existente en la controladora. Para solventar el problema de una única misión disponible en la controladora, el programa nos ofrece la posibilidad de realizar misiones y guardarlos en el ordenador para disponer de ellos a la hora de escribirlos en la controladora.

Para realizar las pruebas y así comprobar la funcionalidad de rutas programadas de esta manera, se ha decidido definir como campo de pruebas el terreno del solar de un particular con su consentimiento. De esta manera no se infringe las leyes existentes de privacidad y dada su lejanía con los aeropuertos de Sevilla y de Morón la hace adecuada para ciertas pruebas. Aunque dada su tamaño, se realizará una misión muy básica en la que se pueda poner de manifiesto todos los aspectos comentados anteriormente.

En dicha misión se puede apreciar como los primeros y últimos puntos son los correspondientes a los de despegue, aterrizaje y aproximaciones. También se puede apreciar como en cada punto de interés (extremos de la parcela en forma triangular) hay dos operaciones, una correspondiente a la orden de llegar al punto y otra de permanencia en el sitio para realizar la medida.

Una vez definida la misión, es importante ajustar cierto parámetros en la configuración extendida de la controladora de vuelo. Dichos ajustes afectan a las maniobras que el drone realiza durante la misión.

1. Velocidad de vuelo entre trayectos. Para asegurar la estabilidad del drone he seleccionado una velocidad máxima de 100 cm/s, ya que a velocidades superiores el drone adquiría una inercia tal que al llegar al punto deseado realiza una gran sobre oscilación alrededor del punto deseado.
2. Velocidad de vuelo a la hora de subir o bajar altura. Este valor no afecta mucho a nuestra aplicación, está configurada a una máxima de 500 cm/s.
3. Radio eficaz en el que se considera que el drone ha alcanzado el punto objetivo. Dado que se trata de un prototipo, con un radio de 2 metros es suficiente para comprobar su buen funcionamiento.

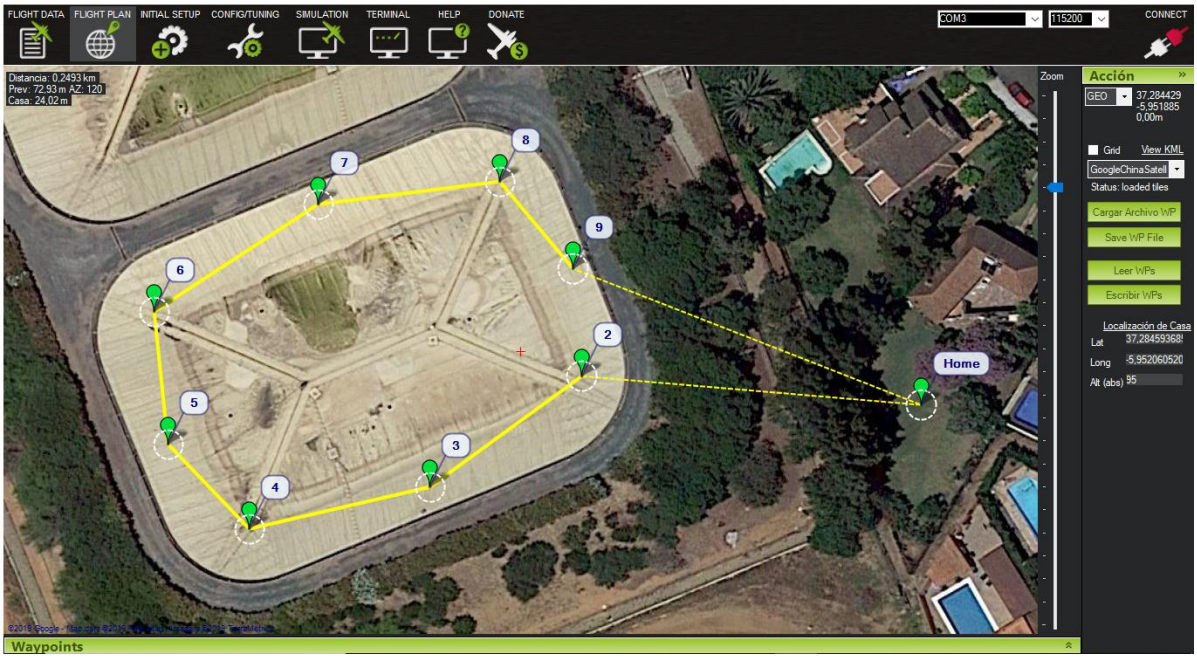


Figura 4-8 Posible misión simulando una ruta alrededor de un recinto

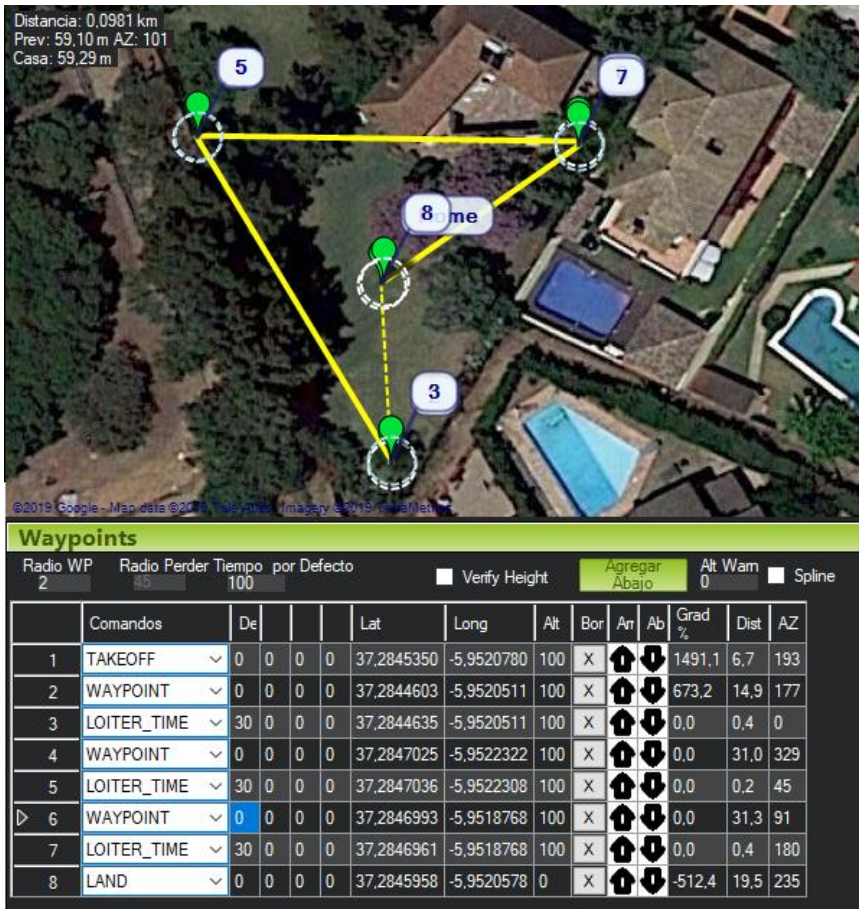


Figura 4-9 Misión para pruebas detallada con cada punto

4.3 Implementación de ambos sistemas en uno solo

Hasta ahora hemos abordado el proyecto mediante dos subproblemas con los cuales hemos alcanzado una serie de objetivos.

Por un lado tenemos un sistema de control que es capaz de realizar un seguimiento de objetivos. Está compuesto por una cámara montada en una plataforma controlada por servos y una unidad de control formada por la Raspberry encargada de recibir la información de la cámara y en función de ella enviar señales de control a los servos.

Por otro lado tenemos un drone capaz de realizar vuelos siguiendo rutas que consisten en varios puntos en el espacio formados por latitud, longitud y altura. Además también tiene la capacidad de realizar vuelos estáticos en un punto de la ruta durante un tiempo. Todo ello de manera autónoma junto a todo lo que ello implica.

Llegados a este punto, únicamente queda unificar cada sistema en uno solo y modificar un poco el sistema correspondiente al seguimiento de objetivo.

4.3.1 Instalación de sistema de seguimiento en drone

Dado que el drone posee una estructura rígida y definida, podemos usarla para instalar el sistema de seguimiento en ella readecuando los componentes del drone en la medida que lo permita el componente.

El elemento principal del sistema de seguimiento, la Raspberry, posee varios puertos distribuidos a lo largo de todo su perímetro además de los pines correspondientes a los GPIO. Si también añadimos el hecho de que se trata de un dispositivo bastante frágil, llegamos a la conclusión de que no se puede instalar en la parte inferior de la base. Por ello se ha decidido reutilizar la posición en la que se encuentra la batería para adecuar la Raspberry en dicho hueco, proporcionando una accesibilidad total a sus puertos además de una gran protección frente a cualquier golpe.

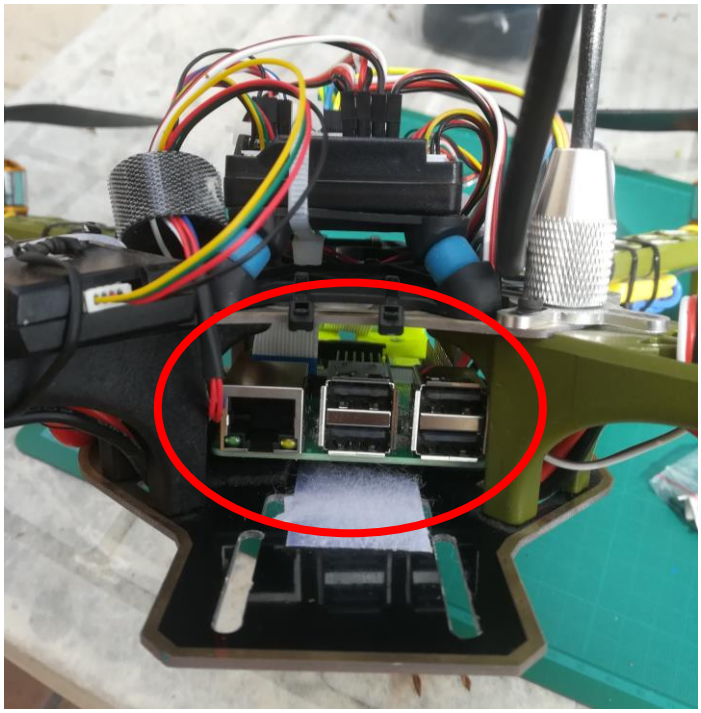


Figura 4-10 Nueva ubicación para Raspberry en el cuerpo del drone

Para poder instalar la Raspberry, ha sido necesario desmontar algunas partes del cuerpo del drone dadas las dimensiones del dispositivo.

Otra opción podría haber sido reubicar la controladora de vuelo en el hueco que acabamos de designar ahora para la Raspberry y esta última en el hueco de la controladora, de esta manera se puede conseguir una accesibilidad aún mayor de la Raspberry. Pero dado que toda la configuración del drone se ha realizado con la controladora de vuelo en dicha posición, para no alterar ningún parámetro he decido mantener dicha ubicación.

Otro gran elemento a ubicar es la plataforma de servos junto a la cámara. Dado que en el capítulo anterior se construyó un prototipo muy básico de plataforma mediante algunas placas de madera con poca precisión, hay que diseñar una nueva plataforma que nos pueda proporcionar mucho más robustez y precisión. De esta manera podemos incluir en el diseño los requisitos de espacio presentes en el cuerpo del dron y así poder adaptarlo mejor.

Para el diseño de la plataforma se ha optado por el programa Solid Work, no solo por su gran versatilidad y posibilidad de realizar planos, sino también por la posibilidad de guardar dichos diseños en archivos “.STL” y así poder imprimir las piezas en plástico mediante impresoras en una empresa designada para ello.

Este método de impresión en piezas de plástico en 3D se trata de una tecnología que ha surgido relativamente hace poco y que poco a poco se está empleando cada vez más en más ámbitos de trabajo, ya que mediante un proceso simple aunque lento, se puede conseguir un modelo real de un diseño en 3D a partir de dispositivos que no superan el cubo de 1 metro de largo de dimensiones. Dicha pieza dada la naturaleza de la impresión y del material usado (plástico) no sirve para ámbitos donde se emplean grandes fuerzas o tensiones, pero para todo lo demás es una gran oportunidad no solo de poder visualizar el modelo 3D en uno real, sino para cualquier otras aplicaciones que no impliquen grandes fuerzas. Aunque las últimas investigaciones en este campo han conseguido elaborar dispositivos de impresión de piezas de metal, innovación que abre aún más el

ámbito de aplicación.

Dado su gran impulso, la impresión en 3D ha abierto también la posibilidad de la aparición de empresas que trabajen con estas impresoras a un precio muy económico si vas con el diseño de la pieza a imprimir ya realizado. Como es justamente nuestro caso, se ha decidido usar este método para imprimir las piezas que constituyen la plataforma de servos.

En cuanto al diseño, en primera instancia hay que decidir dónde irá ubicada la plataforma. Dado que el drone posee unas ranuras en la parte delantera y trasera del cuerpo, se ha decidido ubicar la plataforma en la ranura frontal del cuerpo. Por lo que primeramente se ha modelado la ranura y los servos en Solid Work para poder trabajar sobre tamaños reales desde el primer momento.

Tras el proceso de diseño, las piezas que constituyen la plataforma y que hay que imprimir son:

- Una placa para adaptar un servo a la ranura del drone. El servo estará posicionado de manera que puede variar la rotación en el eje Z de la cámara, y su posición relativa al drone está designado por la posición en que se instale esta pieza. Dicha posición es variable gracias a unas ranuras horizontales que permiten una holgura mientras que los tornillos no estén apretados.
- La siguiente pieza se encuentra instalada sobre el eje rotatorio del primer servo, para que la pieza rote con él. Esta pieza constituye la unión del segundo servo con el primero, girando el eje de rotación del segundo servo de manera que coincida con el eje X de la cámara.
- La última pieza se encuentra instalada sobre el eje rotatorio del segundo servo, y es la encargada de llevar la cámara instalada en sí misma. De esta manera la cámara puede girar sobre sí misma con los giros de ambos servos simultáneamente.

Importante recordar que la disposición de los servos entre sí y entre la cámara se ha realizado de manera que cuando se realicen giros mediante los servos, estos le transmitan solo un giro angular a la cámara y no un desplazamiento. Para ello hay que alinear el centro de la última pieza con los ejes de revolución de cada servo.

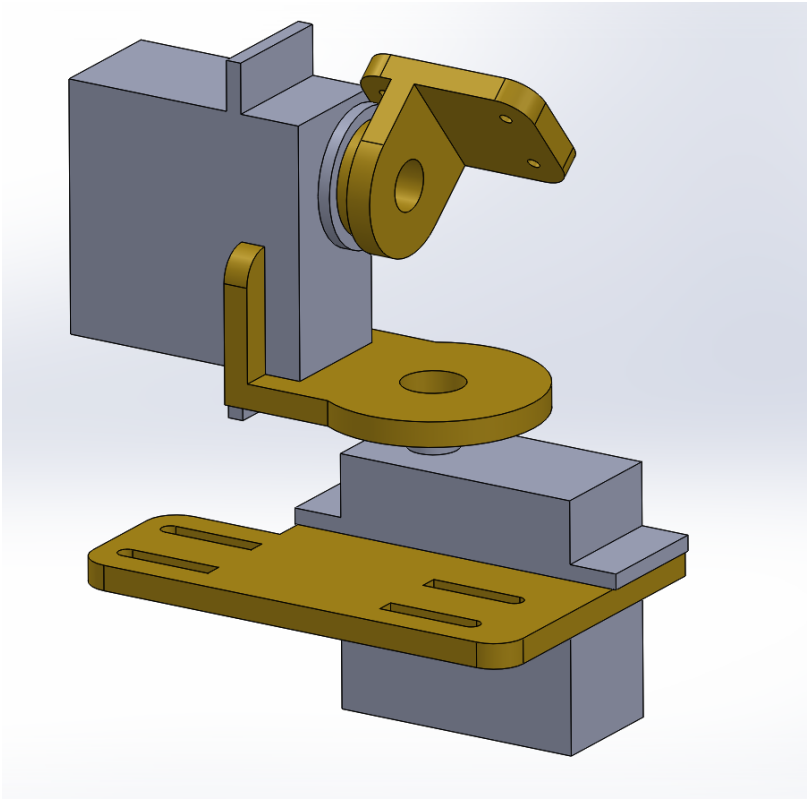


Figura 4-11 Diseño trasero en Solid Work de la nueva plataforma junto a los servos

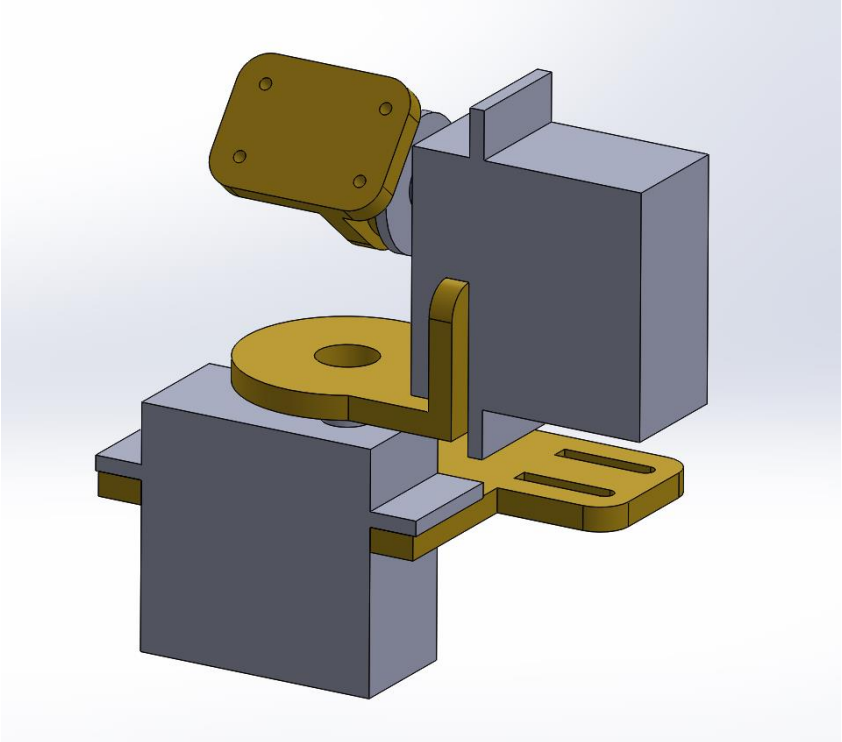


Figura 4-12 Diseño delantero en Solid Work de la nueva plataforma junto a los servos

Para unir las piezas, he usado silicona caliente. Aportan robustez, y se adhiere sin problemas.

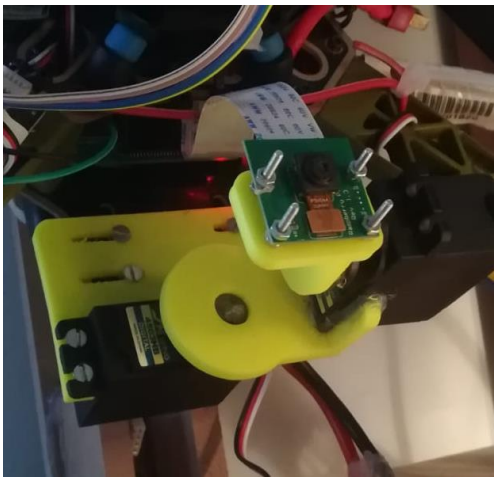


Figura 4-13 Instalación de la nueva plataforma para la cámara

Una vez que tenemos instalado la Raspberry y la plataforma de servos y cámara, el último elemento a colocar junto a la batería extraída al instalar la Raspberry será la batería de la Raspberry. En conclusión tenemos que colocar las 2 baterías a lo largo del drone.

La opción más inteligente sería colocarlo debajo del cuerpo, en la parte inferior. Ya que las baterías al ser los elementos que más masa tienen del sistema completo, al colocarlas en la parte inferior del drone hacen disminuir el centro de masa del sistema a lo largo del eje Z y así ayudar de la mejor manera a la estabilidad.

Sin embargo las baterías a pesar de ser un elemento menos frágil que la Raspberry, aún son consideradas un elemento bastante frágil. Un golpe podría romper una celda y hacer que entren en reacción con el aire haciéndolas estallar. Por ello se ha tener un gran cuidado a hora de realizar los aterrizajes del drone. Otra medida para ayudar a la protección de las baterías sería la de aportar unos pies al drone y así evitar cualquier contacto de las baterías con el suelo.

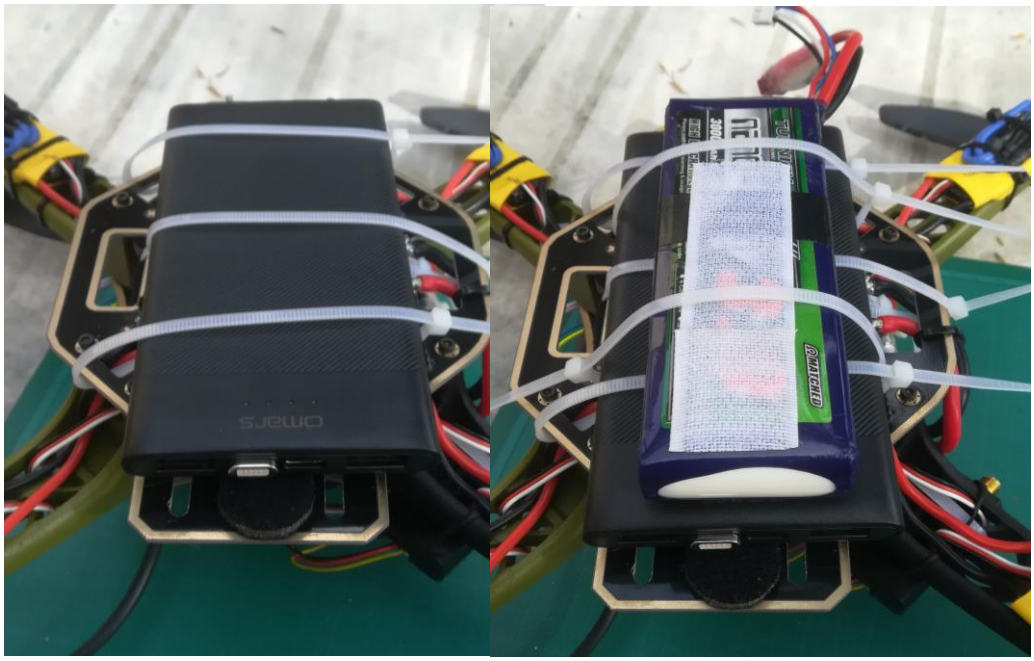


Figura 4-14 Nueva posición para las baterías

Por último, hay que conectar los servos a los correspondientes pines tal y como se explica en el capítulo anterior, y la fuente mediante un cable USB. Una vez que los cables estén conectados, afianzarlos de manera que no estén sueltos, ya que podrían rozar de manera accidental con las hélices produciendo así un error fatal.

4.3.2 Cambio en el programa de seguimiento

En cuanto a al programa de seguimiento que se ha realizado, necesita ciertos cambios. Al fin y al cabo, el programa era una mera prueba para poder comprobar el buen funcionamiento del tratamiento de imágenes y del PID. Pero sólo funciona con un servo, y solo busca círculos de color rosa.

Por ello se realizarán una serie de cambios en el código para adaptarlo a la aplicación final de este proyecto que consiste en realizar un seguimiento del sol. Para ello:

- La cámara necesita un ajuste de parámetros para que la imagen pueda ser tratada correctamente a pesar de estar realizando una foto a una fuente de luz como lo es el sol. Para ello:
    - ISO al mínimo: La ISO es una medida de la sensibilidad del sensor de la cámara a la luz. Cuando un objeto está poco iluminado, se necesita aumentar la ISO para que con dicha luz podamos conseguir una foto similar a la que se haría en condiciones normales. Pero mientras más ISO, más ruido electrónico aparecerá en la imagen.

      En nuestro caso, al querer fotografiar el sol, una fuente de luz, no tenemos en cuenta cualquier objeto de la imagen, solo nos interesa el objeto que produce luz. Por lo que tiene sentido que la sensibilidad se encuentre al mínimo.
    - Bajar el brillo de la cámara al 30%: De esta manera se puede evitar confundir ciertos espectros de colores, de manera que sólo lo más brillante adquiere un color y el resto de los elementos quedan oscuros.

- Velocidad de captura a 300000: Consiste en un valor relacionado con el tiempo de exposición que el sensor de la cámara que capta la imagen permanece expuesto a la luz, o también se puede relacionar con la velocidad que la cámara abre y cierra una película que oculta el sensor de la cámara.

  Dado que se quiere fotografiar un objeto que emite luz, un tiempo de exposición menor, o una velocidad de captura mayor, propiciará a que el sensor sólo adquiera información del objeto que emite luz que es justo lo que se busca.

- Filtro "solarize": Consiste en un filtro que el propio software aplica a la imagen para reducir reflejos no deseados, contrastes en fotos con gran luz. Se trata de un extra más.

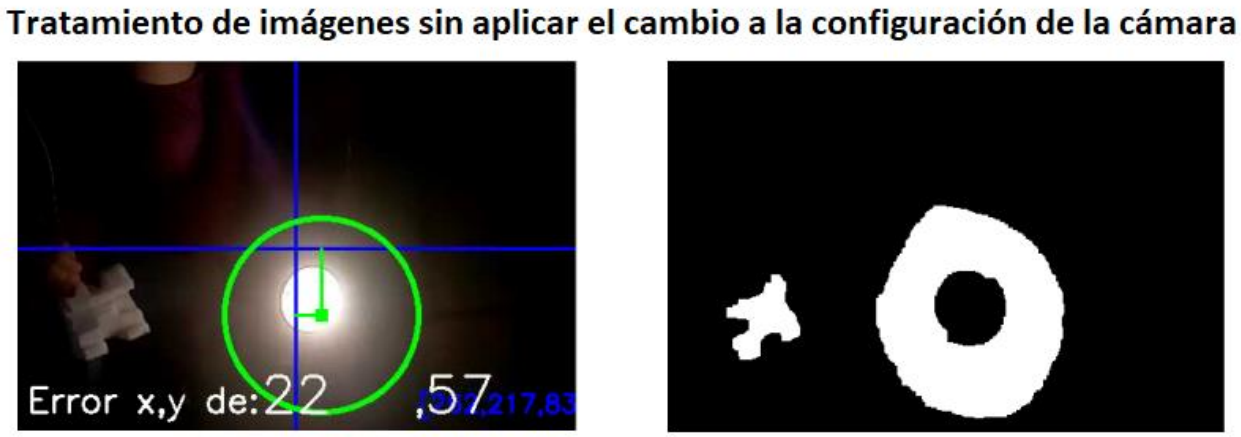


Figura 4-15 Tratamiento de imágenes para luces antes de modificación

En la figura se puede apreciar la necesidad de adecuar la configuración de la cámara. Ya que no solo siguen siendo visibles los elementos que a pesar de no emitir luz al ser blancos son captados por el filtro de color, sino por la detección del halo de luz que se crea en vez del objeto luminoso en sí.

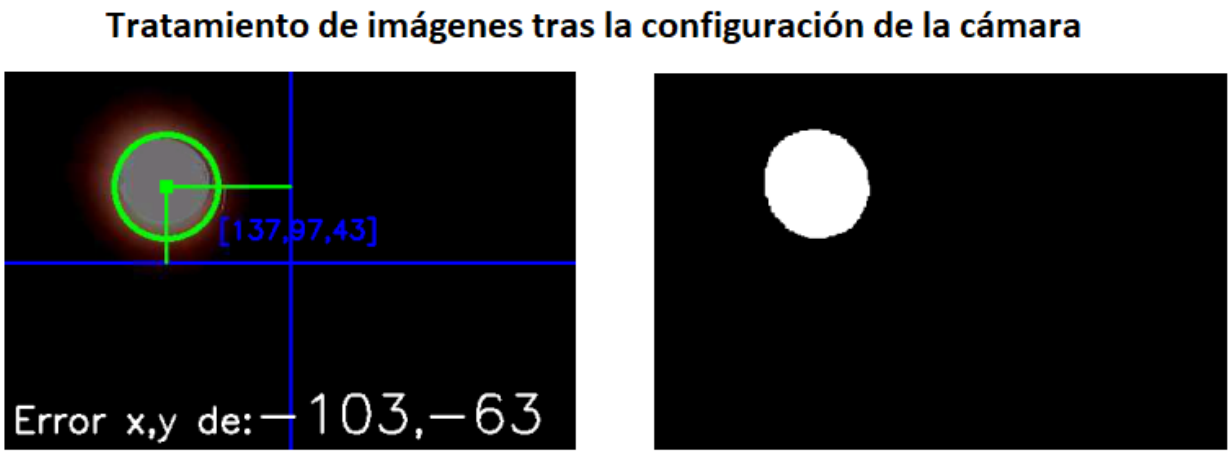


Figura 4-16 Tratamiento de imágenes para luces después de modificación

Después de aplicar la nueva configuración de la cámara, se puede notar enormemente la mejora. Los elementos blancos del fondo han desaparecido de la imagen y el programa es capaz de obviar el halo de luz que se produce, identificando únicamente el objeto luminoso en cuestión.

- Adecuar el filtro de color usado en el tratamiento de imágenes. Ya que anteriormente se buscaba el color rosa, y ahora buscamos el color blanco. Por lo que nuestro filtro estará compuesto por los siguientes valores:

Tabla 4 Valor del color en RGB para localizar objetos luminosos

Máscara de la imagen	*Valor mínimo*	*Valor máximo*
R (Rojo)	200	255
G (Verde)	200	255
B (Azul)	200	255

- Implementar un modo de búsqueda para que cuando el sol no se encuentre en la imagen durante un tiempo definido, entre en dicho modo en el que los servos harán recorrer la cámara por todo el espacio de trabajo para intentar localizar el sol. Una vez que lo encuentre, pasará al modo de seguimiento. Y si no lo encuentra podemos asumir que el sol se encuentra tapado por las nubes, ya que asumiremos que no realizaremos medidas en el ocaso o en el amanecer.
- Implementar un segundo PID para el segundo servo, exactamente de la misma manera que el primero pero con variables independientes para poder sintonizarlo de manera independiente al primero.

Las modificaciones en el programa reflejan un esquema diferente de pseudocódigo comparada con el prototipo del capítulo anterior, la figura 4-16.

Tras varias pruebas, se llegó a la conclusión de 2 aspectos que hacían la plataforma poco estable y no consistía en el sintonizado de los PID. Por un lado en el tratamiento de imágenes no se llegaba a encontrar siempre el objetivo a pesar de realizar un buen filtrado de color y de ruido, el método de Hough fallaba a menudo y encontraba el objetivo un 70% de las veces. Situación válida para el caso en el que tenga que encontrar al objetivo, pero en un PID la intermitencia hace reiniciar la actuación varias veces, por lo que no es lo que buscamos.

A pesar de ello, el espacio de trabajo no se trata del mismo que en el capítulo anterior. Se trata de la semiesfera superior en vez de la frontal. Esto hace que el seguimiento se vuelva más complicado ya que en el capítulo anterior una diferencia en las X se solventaba con un único servo y las Y igual. Sin embargo en nuestro caso cuando la cámara rota alrededor del eje Z la posición del objetivo cambia en las dos variables. No se trata de un problema que el conjunto de dos PID no puedan resolver, sin embargo esto sumado al problema de la interferencia de la localización del objetivo en pantalla hizo que tuviera que replantear el mecanismo por el cual tenía que identificar el objetivo.

Dado que en la imagen se ha adaptado para poder fotografiar objetos que despiden luz, y en el cielo suponemos que solo está el sol, tendremos muchos menos problemas con el ruido o posibles objetos que consigan pasar los filtros. Solo tendremos el sol, o en el peor de los casos, un sol con la superficie separada. Entonces, dado que solo tendremos un objeto para el tratamiento de imágenes, no es necesario recurrir a una identificación de formas. Podemos directamente trabajar con el objeto conseguido tras los filtros.

Para ello, primeramente calcularemos mediante la función cv2.edge los bordes del objeto. Que en este caso es trivial ya que la imagen está compuesta por elementos de valor 0 o de valor 255 y el gradiente para calcular los bordes no producirá ningún borde erróneo al objeto a estudiar. Una vez tenemos el contorno podemos realizar un paso intermedio a la identificación del objeto, que consiste en elegir siempre el contorno más grande de todos los contornos calculados para prevenir casos defectuosos con mucho ruido o superficies del objeto desprendidas.

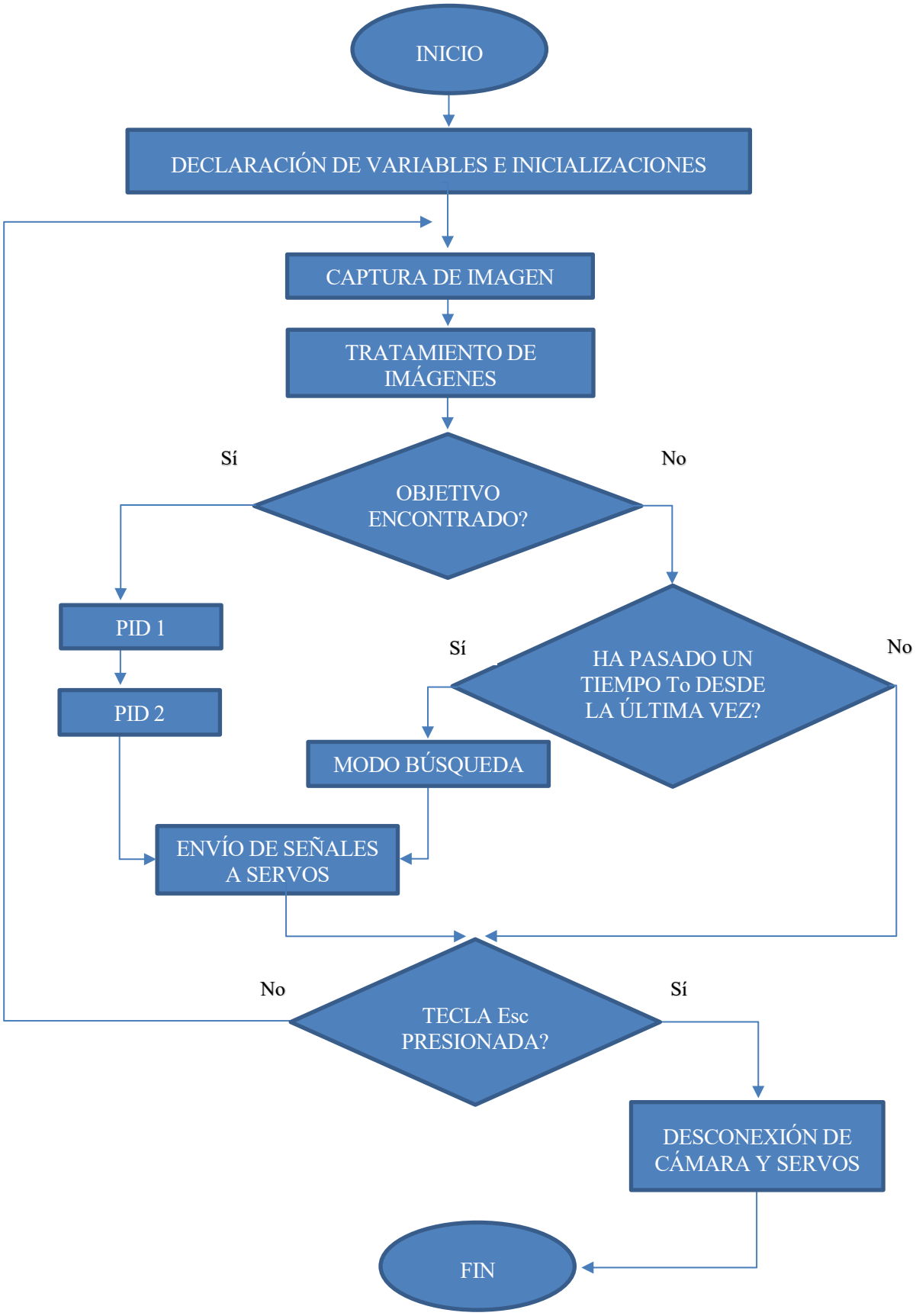


Figura 4-17 Diagrama de flujo del proyecto final de seguimiento de objetos con servos

Para la identificación del objeto, nos basaremos en los momentos de inercia producidos por su contorno de la siguiente manera [24]:

- Centro del objetivo en X:

$$Pos(x) = \frac{\mu 10}{\mu 00}$$

- Centro del objetivo en Y:

$$Pos(y) = \frac{\mu 10}{\mu 00}$$

- Área:

$$Area = \mu 00$$

- Radio:

$$Radio(r) = \sqrt{\frac{Área}{\pi}}$$

De esta manera se consigue en todo momento la localización correcta del objetivo con una gran exactitud, lo que hace funcionar correctamente el PID y en consecuencia el seguimiento de objetivo. Sin embargo es un método de identificación poco robusto frente al ruido u otros errores que sobrevivan el proceso de filtrado.

5 RESULTADOS EXPERIMENTALES

Las pruebas realizadas para comprobar el buen funcionamiento del sistema en global se han realizado a los 2 subsistemas existentes de manera independiente para tener un mejor entendimiento: el drone y el sistema de seguimiento. Por lo que se refiere a cada subsistema, se expondrán las pruebas que se han ido realizando a medida que el proyecto iba desarrollándose.

5.1 Pruebas con el sistema de seguimiento de objetos

En el capítulo 3 se creó el primer prototipo formado por la Raspberry y una plataforma improvisada para montar la cámara en 2 servos mediante pequeños paneles de contrachapado. El objetivo de esta primera prueba es configurar las bases de un buen tratamiento de imágenes con la implementación en un PID. Puesto que el objetivo era comprobar de manera más eficaz su funcionamiento, sólo se implementó el seguimiento a un servo y por tanto sólo se podía corregir la diferencia de posiciones en coordenadas X en la imagen.

El objetivo a seguir en final instancia del proyecto es el sol, por lo que se buscó un objeto esférico y así poder sentar la base de la identificación del objetivo en elementos redondos mediante el método de Hough. Dado que el campo de pruebas se estableció en una pequeña habitación donde se podía mantener la iluminación bajo control identifiqué un elemento esférico con un color no presente en la habitación: una pelota de ping pong naranja.

Mediante pruebas en el filtro de color, se comprobó que la calidad de la cámara era bastante baja con una muy baja gama de colores. Esto presenta un problema ya que el color de la pelota puede coincidir con el color de la piel humana u otros elementos y entorpecer la identificación. Mediante un mejor control de la iluminación del objeto y de un filtro de color más riguroso se consiguió solventar el problema.

Sin embargo, cuando el tratamiento de imágenes funcionaba correctamente se identificó un segundo problema. La identificación del objetivo era un 70% efectiva. A pesar de que el filtrado de color y de ruido trabajara correctamente, había intermitencia en la identificación del objetivo. Dado que se trata de un sistema en tiempo real y que podría haber fotogramas que no percibiera, se llegó a la conclusión en la que había fotogramas donde los filtros distorsionaran tanto el objeto que la identificación de círculos no funcionara correctamente, y dado que el procesamiento estaba al máximo podrían no verse reflejado en pantalla. A pesar de ello, el PID se implementó y se sintonizó correctamente.

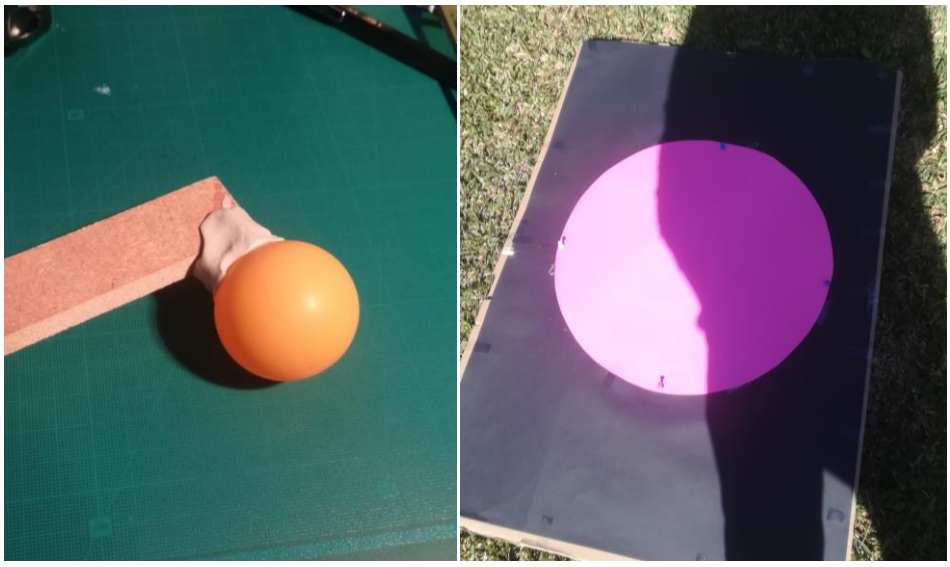


Figura 5-1 Elementos usados para el seguimiento de objetos

Pero para que la prueba fuera correcta, se cambió el campo de pruebas a uno más realista, el exterior en un día normal. Para ello también se adaptó el objeto a seguir dado que la pelota era muy pequeña, se construyó una plancha de cartón enfundada con cartulina negra con un círculo rosa en el centro. De esta manera podría asegurar un buen gradiente entre colores, y un seguimiento de objetivo a una distancia moderada (Dado que con la pelota se realizaba a unos 20 cm de la cámara).

Durante esta prueba, surgieron muchos problemas relacionadas con la iluminación: la cartulina creaba destellos en ciertos ángulos que a ojos de la cámara quedaba como blanca, la cartulina expuesta a la luz y con sombra tiene un color tan diferente que no podían entrar en un mismo filtro de color, el sol creaba una contraluz a ciertas horas que hacía bajar mucho la calidad de color captada por la cámara… Como medida se optó por elegir una hora del día de manera que el entorno creara una sombra estable para a ojos de la cámara poder tener un color estable y así realizar las pruebas.

Sin embargo todo lo referente a los problemas de iluminación no concierne a este proyecto ya que el objetivo final será hacer el seguimiento del sol, por lo que todos los problemas de iluminación no tendrán lugar en dicha situación. El motivo de tomar todas las molestias de hacer pruebas de seguimiento es para probar la implementación del PID en el servo y así poder sintonizarlo. A pesar de ello, la iluminación acarreará problemas que se reflejan en las pruebas que a continuación se mostrarán.

A continuación se mostrarán unas gráficas en las que se mostrará los pulsos enviados al servo a lo largo del tiempo, mientras se realizaron pruebas de seguimiento para poder sintonizar bien el PID.

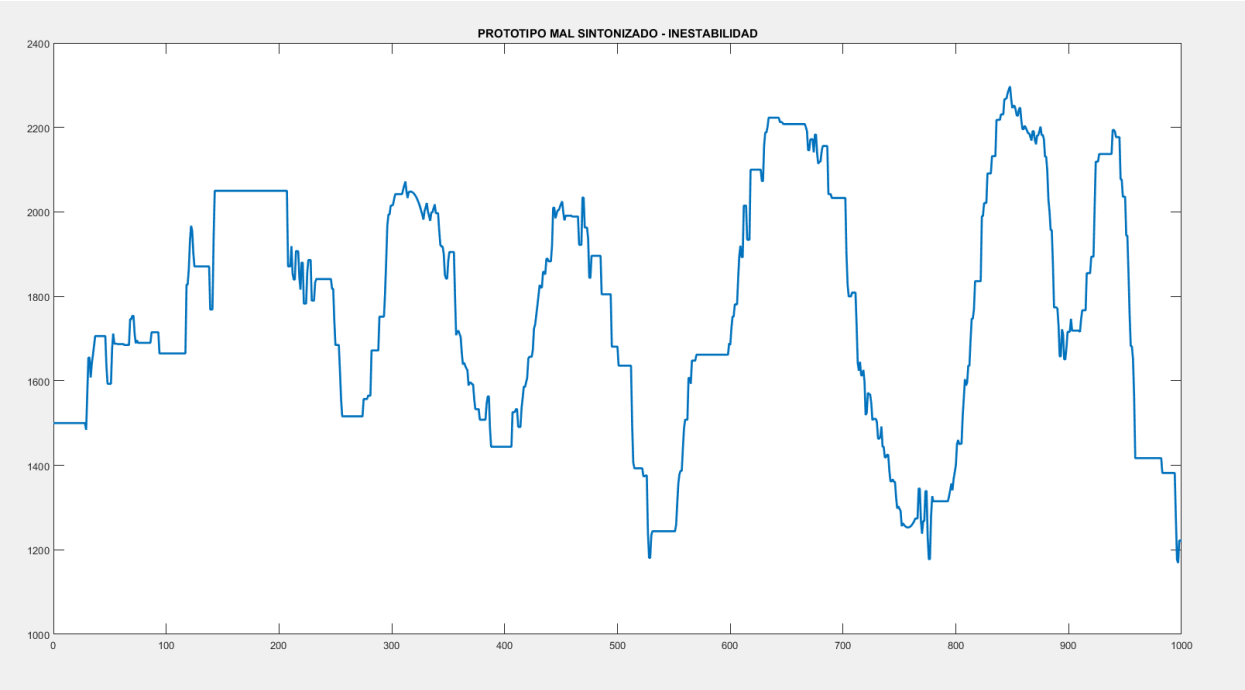


Figura 5-2 Gráfica de los pulsos transmitidos al servo a lo largo del tiempo 1

En la gráfica de la figura 5-2 se puede observar perfectamente como el sistema es inestable. Únicamente teniendo el objetivo en una posición fija el sistema intenta alcanzar el estado de error nulo pero las sobre oscilaciones hacen que cada vez se aleje más del objetivo. Supone un estado inicial de la sintonización de los PID totalmente normal y solucionable.

En la siguiente gráfica de la figura 5-3 tenemos un sistema ya estable, pero no eficiente ya que ciertos fallos de identificación del objetivo cuando éste se mueve (mala lectura) produce un salto muy grande en el sistema. El PID funciona, y se consigue el objetivo, pero no es óptimo.

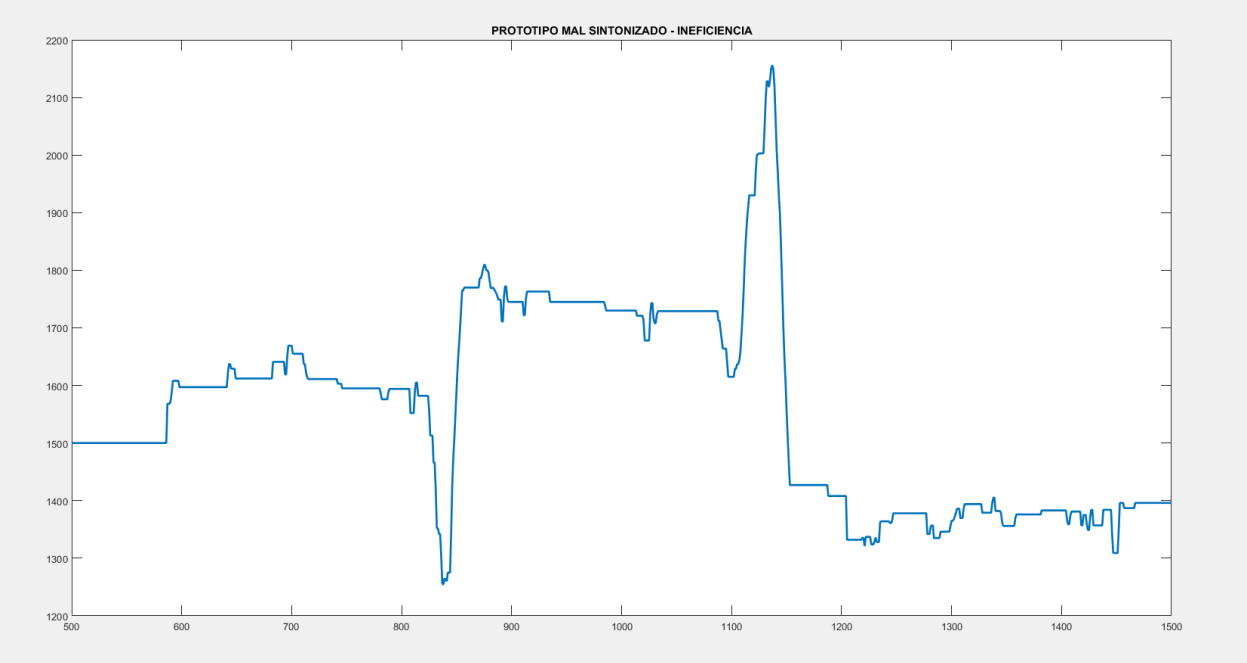


Figura 5-3 Gráfica de los pulsos transmitidos al servo a lo largo del tiempo 2

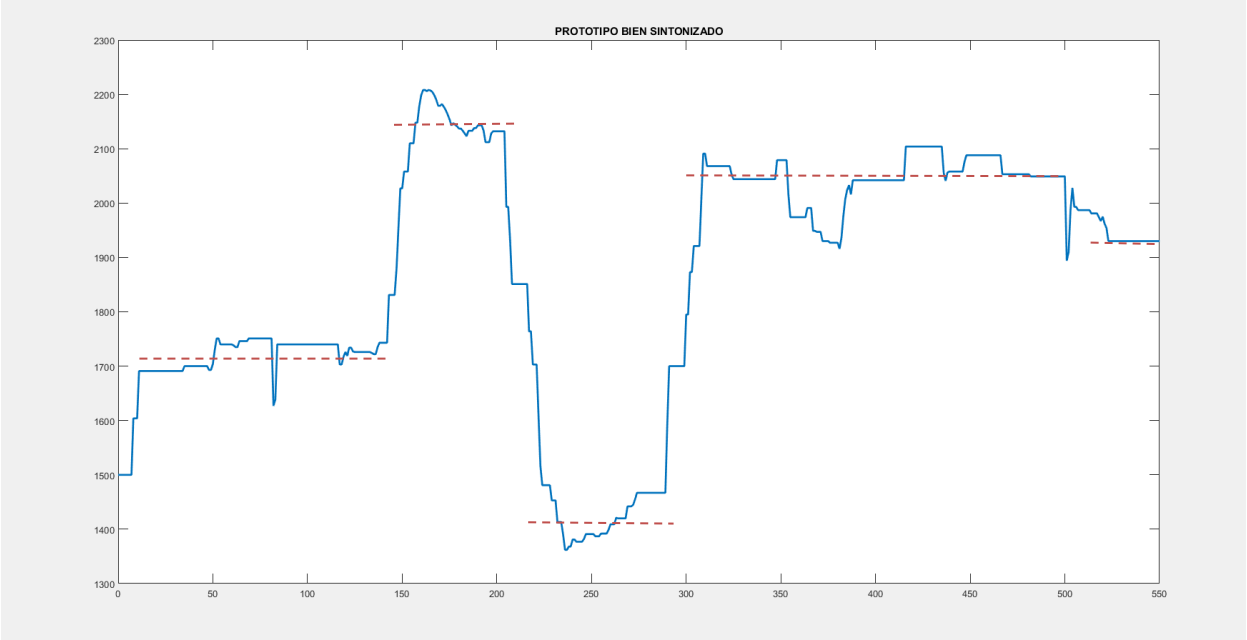


Figura 5-4 Gráfica de los pulsos transmitidos al servo a lo largo del tiempo 3

En la última gráfica de la figura 5-4 podemos ya asumir que se ha optimizado el PID en cuando a estabilidad y eficiencia respecto a las pruebas anteriores, ya que además de poseer un PID bien sintonizado tenemos un sistema de reconocimiento eficiente sin errores de lectura desorbitados. Se trata de un sistema estable, que responde bien tanto a los cambios de posición del objetivo a seguir, como a fallos en la identificación del objetivo.

Una vez que se ha conseguido un sistema de seguimiento básico, se procede a enfocar el sistema a la aplicación para la que se ha planteado este proyecto: un seguimiento autónomo del sol.

Para ello se elabora una estructura más firme para los servos y cámara y que se pueda ensamblar en el drone. Se realiza mediante impresión 3D, aportando firmeza y precisión al sistema. Dicha estructura se ensambla con el drone y a partir de entonces las siguientes pruebas se harán con el sistema montado en el drone.

En este segundo modelo, se implementa un segundo PID para controlar el otro servo. La implementación en el código es exactamente igual pero con variables que lo definen independientes por si se necesita volver a sintonizar por separado los PID a pesar de ser los mismos servos.

También es necesario retocar los ajustes de la cámara para poder realizar la foto al sol sin deslumbrar toda la imagen y modificar el filtro de color.

Una vez realizados estas modificaciones se empieza a realizar pruebas dentro de una habitación, usando una lámpara como símil al sol.

Sin embargo vuelve a surgir el problema que tuvimos anteriormente con la identificación del objetivo. La lámpara tras el filtro de color y de ruido queda como un halo alrededor de la bombilla, pero sigue siendo circular y aun así hay problemas de identificación. Dado lo bien que resultaba la imagen tras los filtros se decidió cambiar el método de identificación a uno de puro detección de contornos e identificación de características por momentos de inercia. Después de implementar este método de identificación, la plataforma empezó a funcionar mostrando una mejora enorme.

En las siguientes gráficas donde se muestran los pulsos enviados a los servos que formaban la plataforma, se puede observar la enorme mejora que supuso la implementación del nuevo método de identificación. En el método usado por Hough la plataforma se vuelve incontrolable dado la gran cantidad de tiempo en el que el objetivo no se encuentra localizado.

Sin embargo en el método usado por contornos y momentos, no hay momento en el que el objetivo no se encuentre no detectado, y esto se traduce en un control estable y preciso mediante una buena sintonización de PIDs.

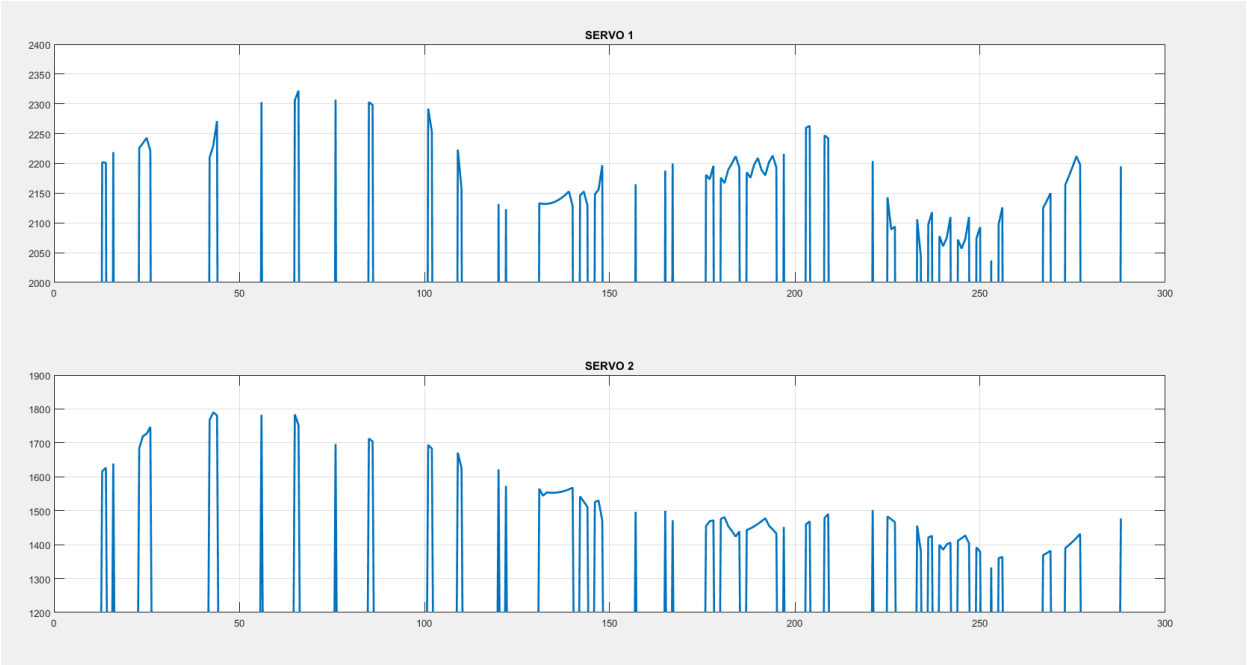


Figura 5-5 Gráfica de los pulsos transmitidos a los servos a lo largo del tiempo con método de Hough

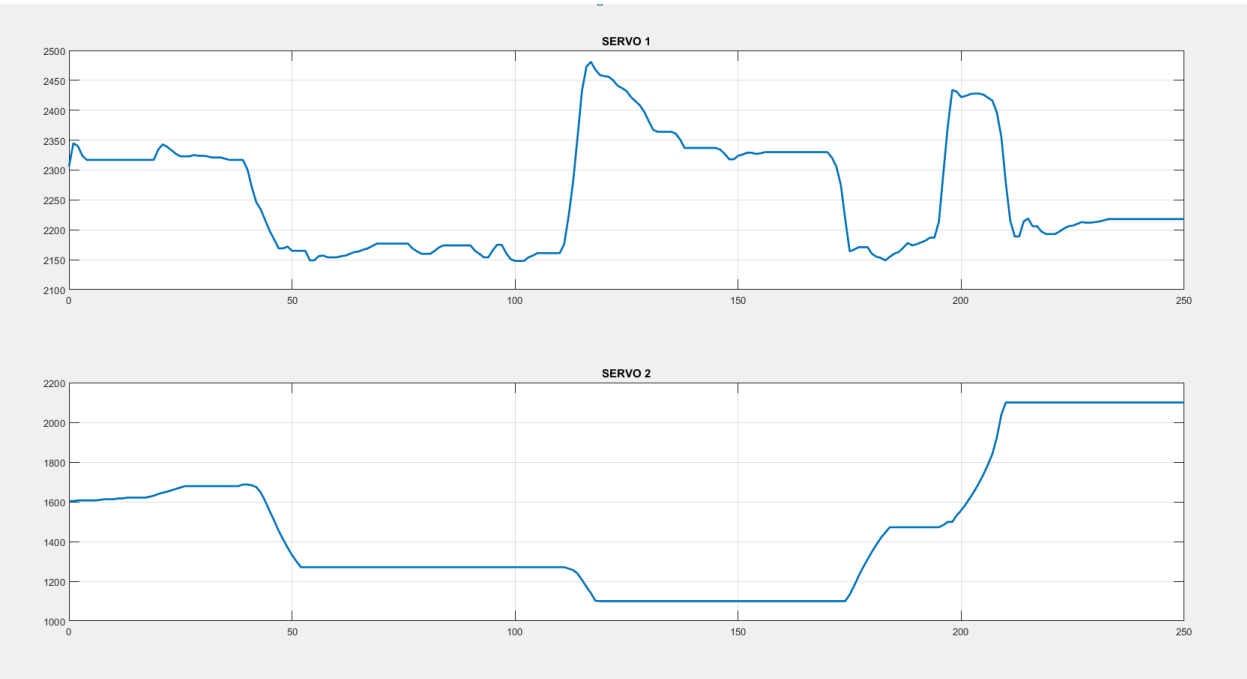


Figura 5-6 Gráfica de los pulsos transmitidos a los servos a lo largo del tiempo con método de contornos

Con esta configuración el sistema de seguimiento funciona perfectamente cumpliendo todos los objetivos que se le exigía como primer prototipo. Sin embargo presenta ciertos aspectos que podrían ser mejorables:

- La conexión entre la cámara y la Raspberry se realiza mediante un cable especial de una serie de cables planos. La longitud del cable es bastante corta y además es bastante frágil. En mi caso este hecho reduce bastante el espacio de trabajo ya que en vez de disponer de toda la semiesfera superior tal y como la plataforma permite, ésta se ve más reducida para no romper el cable. Es un suceso bastante probable ya que durante las pruebas ya sucedió. Como medida, he acotado las variables a escribir en los servos acorde a unas posiciones extremas en el que el cable alcanza una tensión crítica. Durante las pruebas se puede observar si se ha alcanzado dicha posición extrema ya que la distancia al centro del objetivo se torna de un color rojo indicando la imposibilidad de corregir dicho error.

- Dada la naturaleza de la plataforma, existe un punto conflictivo de la plataforma en el que el control se vuelve más complicado. Dicho punto es en el que la cámara se encuentra alineada con el eje de rotación de eje Z del sistema. Se trata de un punto en el que si el objetivo se encontrase allí, habría muchas soluciones posibles ya que al rotar el primer servo, no influiría nada en cuanto a distancias de centros.

  Sin embargo ese no es el problema. Es correcto asumir que el sol se encuentre en puntos cercanos a éste dado que en las horas donde se produce más calor a lo largo del día sea cuando el sol se encuentre en el punto más alto del cielo. No tiene que ser con exactitud el punto descrito anteriormente, pero si cercano, y en dichos puntos un movimiento del primer servo no equivale a una simple traslación del objetivo en cuanto al eje X de la imagen. También supone una traslación del objetivo en cuanto al eje Y, y eso puede suponer un problema si no se tratara con 2 PID independientes. Ya que aunque con el primer PID se modifique la distancia en cuando a Y de la imagen, una vez que tenga un error en las X adecuado, el segundo PID podrá corregir sin problemas la diferencia en Y causada por el movimiento del objetivo y por el primer servo.

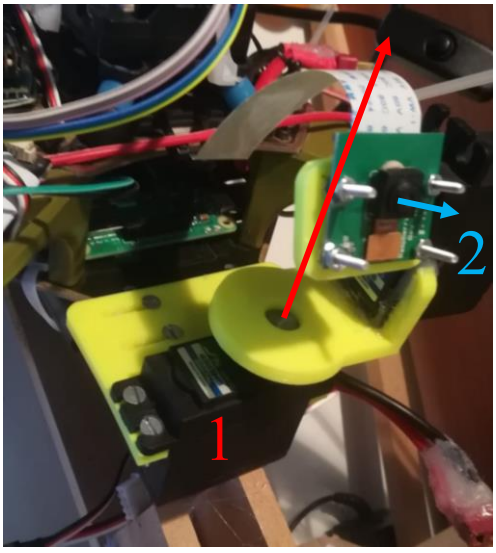


Figura 5-7 Orientación de cámara respecto a los servos 1

En la posición de la figura 5-7 no existe el problema anteriormente comentado. Un giro del primer servo implicará una diferencia del centro del objetivo en coordenadas X y un giro del segundo servo producirá un efecto similar pero en coordenadas Y.

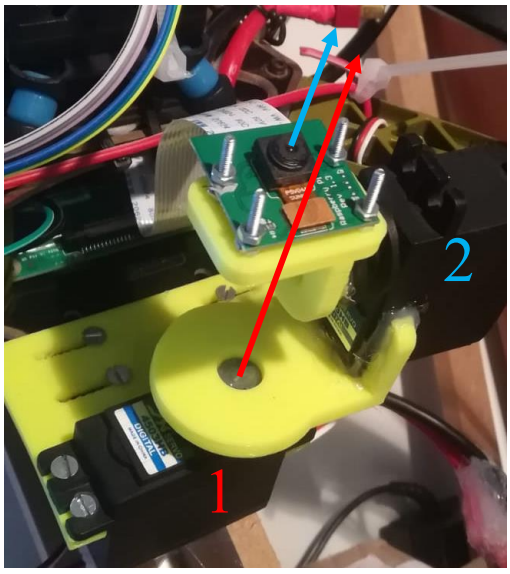


Figura 5-8 Orientación de cámara respecto a los servos 2

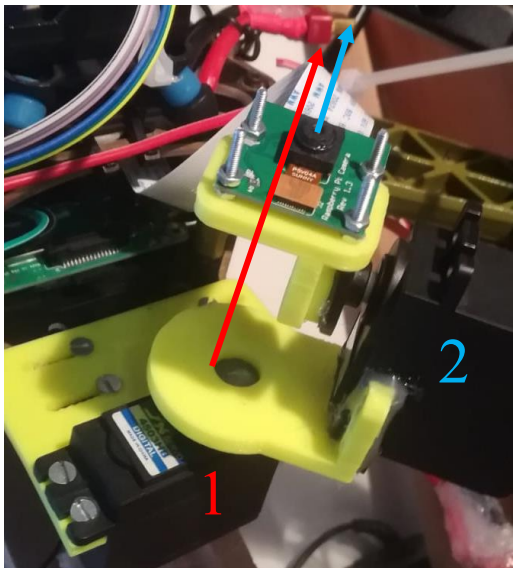


Figura 5-9 Orientación de cámara respecto a los servos 3

En esta otra posición de la figura 5-8 o 5-9, se puede apreciar como al estar claramente en posiciones diferentes la plataforma, siguen apuntando a un mismo objetivo. Posiciones cercanas a ésta tal y como se ha explicado antes, un movimiento del servo 1 puede acarrear que el objetivo se mueva en la imagen tanto en posición X como en Y. Esto se produce ya que al girar el servo 1 estamos rotando la imagen, y dicha rotación mientras más cerca del punto mostrado en la anterior figura, más fuerte será y más cambios producirá en cuanto a coordenadas Y del objetivo. Por ello cuando se encuentra en posiciones alejadas de dicho punto esta rotación no afecta.

5.2 Pruebas con el drone

Antes de empezar las pruebas de vuelo, hay que establecer un campo de pruebas apto para que se puedan realizar las pruebas de vuelo sin infringir ciertos aspectos legales correspondientes a la ley vigente sobre los drones.

Para ello, el campo de pruebas será el jardín de la parcela de un particular con una cantidad de metros cuadrados suficientes para realizar unas pruebas simples. De esta manera no se invade la privacidad de nadie ya que el drone sólo sobrevolará la extensión del campo de pruebas a una altura menor de 5 metros. Además se ha comprobado que la distancia de seguridad con los aeropuertos más cercanos a la zona (el aeropuerto de Sevilla y la base aérea de Morón) sea superior a la de la reglamentaria para poder volar un drone (zona coloreada).

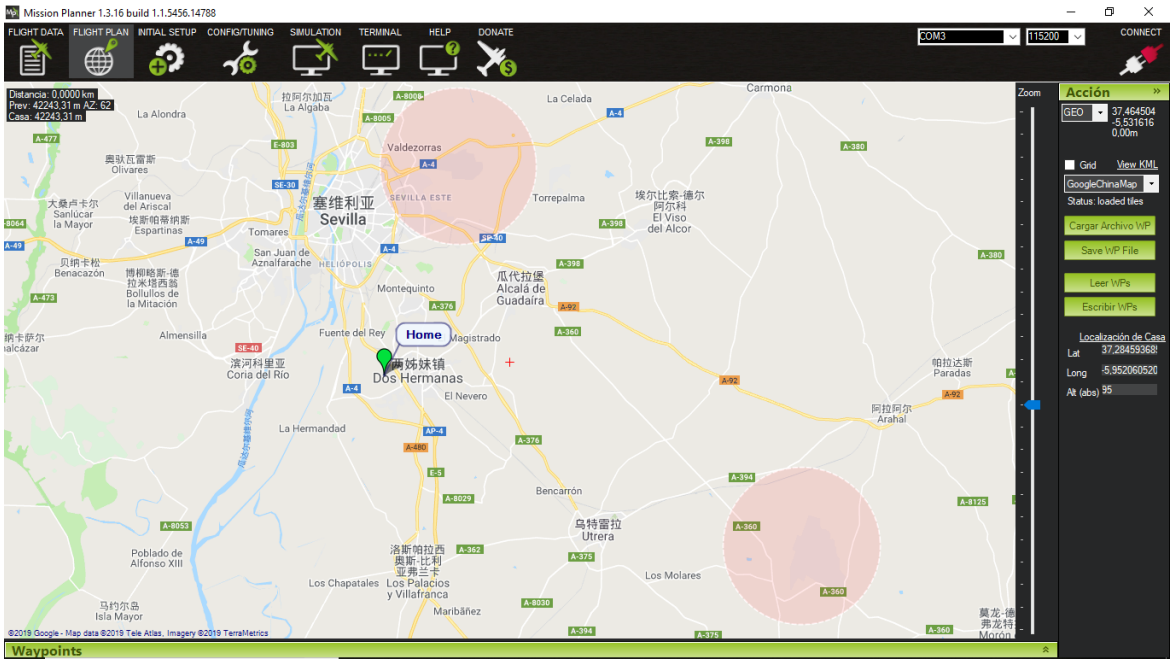


Figura 5-10 Localización de zonas restringidas para volar drones

Figura 5-11 Campo de pruebas

Otra opción hubiera sido realizar las pruebas en algún recinto cerrado donde el viento no podría interferir de ninguna manera asegurando así una mejor estabilidad, además de teniendo una iluminación totalmente controlada. Sin embargo esos aspectos del ambiente que nos proporciona sería sacrificando la posibilidad de realizar vuelos por GPS ya que las paredes y techo entorpecerían la conexión con satélites. Dado que la funcionalidad GPS es fundamental, se desechó tal idea.

Después de determinar el campo de pruebas, y del calibrado de sensores, se procede a realizar las primeras pruebas de vuelo.

Las primeras pruebas se realizan en modo Stabilize con dos propósitos: para que el usuario de adapte a los controles de la emisora y así ser capaz de manipular el drone en dicho modo de vuelo, y para asegurar las funciones básicas del drone. Por ejemplo, si un motor girara para un sentido erróneo o directamente no funcionara, o que un canal no emitiera las señales correspondientes… Todo este tipo de errores es mejor visualizarlos en este modo ya que al ser tan simple podemos identificar la raíz del problema más fácilmente.

Una vez aseguradas las funciones básicas, se realizan pruebas con el modo Alt-Hold y con el modo Loiter (En dicho orden) para sintonizar los PID correspondientes a cada aspecto de control. Para ello, siempre se ha de modificar el proporcional (P) hasta encontrar un valor suficiente que visiblemente sea capaz de realizar cambios en el drone, después modificaremos el integral (I) para que el error se minimice y por último el derivativo (D) para que las acciones a tomar por el drone sean suaves y precisas en vez de bruscas y con sobre oscilaciones.

Una vez sintonizado los PID se procede a realizar las pruebas de seguimiento de rutas autónomas. El drone realizará una misión simple en la que únicamente deberá ir de un punto A a un punto B y volver a hacer el mismo ciclo durante varias iteraciones. También deberá mantenerse en el punto unos 10 segundos cada vez que lo alcance.

Por último, se ha investigado acerca de la posibilidad de realizar un vuelo totalmente autónomo aportando la función de despegue y aterrizaje a la misión autónoma.

Es importante denotar que todo este proceso se ha llevado a cabo una vez que el sistema de control estuviera instalado en el drone, ya que al aportar un peso extra a éste, puede modificar el modelado dentro del control.

Los resultados en general fueron positivos. Hubo ciertos aspectos para tener en cuenta que produjeron un mal funcionamiento o perturbaciones importantes en el sistema en las primeras etapas de pruebas, pero al final se ha conseguido tener un drone que puede realizar vuelos autónomos con cierta exactitud y que fue capaz de realizar todas las maniobras con todo el peso que la totalidad de los componentes generaron.

Dado que se trata de una controladora de vuelo donde el modelado se encuentra ya instalado, no he podido monitorizar los datos del drone en pleno vuelo, pero si he podido estudiar elementos que han supuesto un factor importante a influir en todo el proceso:

- Calibrado de sensores: El buen calibrado de sensores es totalmente fundamental para la estabilidad del drone. Una vez que el drone se encuentre en el aire, necesita tomar una referencia para poder dirigirse a ese estado de referencia que le proporciona la estabilidad que busca. Si la referencia es errónea, por mucho que el drone alcance la referencia con éxito, se producirán derivas y malfuncionamientos.

  Es bien cierto que una vez que se implementen PID para mantener posiciones, éstos son capaces de corregir las derivas a pesar de que la referencia sea errónea, pero para un funcionamiento óptimo y no sobrecargar el PID con operaciones que podemos evitar para que así se encargue de las funciones que queremos, es necesario un buen calibrado para aportar una buena referencia al sistema.

- Calibrado de hélices: Otro calibrado muy importante es el de las hélices, ya que éstas de fábrica pueden traer cierto desperfecto que puede afectar en gran medida a la estabilidad del drone. El desperfecto del que hablo es el hecho de que las dos palas de la hélice no pesan exactamente lo mismo.

  La diferencia en peso puede ser mínima, de miligramos, pero a pesar de ello, es un desperfecto que a la hora girar a grandes revoluciones, produce ciertas vibraciones. Dichas vibraciones pueden incrementarse si son transmitidas a lo largo de todo el cuerpo hacia la controladora de vuelo y así transmitir una perturbación en los sensores que puede comprometer la estabilidad de vuelo.

En la práctica, la diferencia es tan mínima que no puede hacer que el drone sea totalmente incontrolable. Pero para la aplicación para la que estamos diseñando el drone, la estabilidad es primordial, una especificación del sistema que tenemos que maximizar y no comprometer, por lo que hay que tomar medidas para abordar dicho problema.

La solución es simple, mediante un dispositivo ya mencionado en el capítulo 2 podemos sostener con imanes la hélice desde su eje de rotación de manera que la fricción sea mínima. En dicha situación dado que una pala pesa más que la otra, la hélice se inclinará hacia dicha pala. De esta manera podemos ver qué hélice es la que pesa más. Una vez que sabemos qué hélice es la que pesa menos, le podemos pegar pequeñas tiras de cinta de celo para aportar un pequeño peso, o limar un poco la pala que pese más.

De esta manera podemos equilibrar el peso de las palas y eliminar la posible vibración que causarían. Es un proceso necesario a realizar en todas las hélices del drone.

Como dato importante a destacar, de la misma manera que un pequeño peso en una pala puede producir un desequilibrio y causar vibraciones, cualquier otro desperfecto producirá efectos similares. Por ejemplo la rotura de la hélice en una esquina de la pala causada por una caída brusca lateral. La hélice necesita ser reemplazada. Se trata de un hecho del que se ha sido testigo durante las pruebas.

- GPS: El GPS es un elemento muy importante en este proyecto ya que sin él, no serían posibles los vuelos autónomos o misiones programadas que se han usado. Sin embargo no se ha elegido un sistema que soporte una capacidad enorme de GPS con la mejor calidad del mercado. Dado que se trata de un prototipo se ha buscado el más económico dentro de unas prestaciones.

  Dada la calidad del módulo GPS que se dispone, hay una serie de aspectos para tener en cuenta para poder usar lo de manera óptima:

  - Para que el drone pueda conocer su posición real de manera exacta, es necesario que disponga de al menos 6 o 7 satélites siendo 7 lo máximos que es capaz de conectar a la vez. Sin embargo el proceso no es instantáneo, por lo que para poder usarlo de la mejor manera hay que encender el drone y una vez que éste suministre energía al módulo GPS esperar hasta que contacte con los satélites suficientes. El proceso de espera suele ser de unos 3 o 5 minutos, y dado que los motores estarán en modo reposo no supondrá un gran gasto de energía.
  - Reiterando en el aspecto de lo importante que es que el módulo GPS se conecte al mayor número de satélites, una vez que pase un tiempo para que se establezca la conexión con ellos, hay unos aspectos a seguir para que la conexión sea lo mejor posible: evitar obstáculos grandes alrededor del drone en los vuelos de manera que no puedan interferir en la conexión de satélites, y de la misma manera mientras más alto se vuele menos obstáculos y por tanto mejor conexión.

    Perder la conexión de algún satélite puede acarrear un balanceo en sentido al satélite perdido, y es impredecible lo fuerte que puede llegar a ser la deriva. En la práctica se ha experimentado ligeros balanceos y alguna deriva lo suficiente fuerte como para tener que pasar a modo manual para evitar la colisión del drone.

  Tal y como se ha comentado, una pérdida de conexión de satélites puede suponer un ligero balanceo o en casos de perder más de una conexión, perder la estabilidad total del drone.

  Durante las primeras pruebas, algunas veces pero no muchas, se ha tenido que cambiar al modo manual para evitar posibles colisiones del drone con el entorno. Una medida simple para contrarrestas este problema puede ser el reducir la velocidad con la que el drone se estabiliza para así afrontar mejor las malas lecturas bruscas. Justamente este aspecto se ve reflejado en el proporcional y el derivativo del PID.

  Sin embargo a pesar de todas las medidas que se tome para mejorar la función de GPS, éste siempre podrá aportar un error de posición debido a la poca calidad de éste. Esto se traduce como un balanceo en cualquier sentido, sin embargo un buen sintonizado de PID permitirá que a pesar de tener estos errores el drone corrija la posición cuando el módulo vuelva a aportar datos de manera correcta.

- Batería: La batería elegida para el proyecto consiste en una batería adecuada al tipo de drone que se ha construido. Esto quiere decir que hay baterías con una autonomía mucho mayores pero eso necesitaría de un drone más caro y grande (envergadura y número de motores mayores).

Sin embargo la batería elegida no es capaz de proporcionarnos más de 10 minutos de un vuelo de calidad, ya que a medida que pasan esos 10 minutos los motores pierden potencia y el drone pierde mucha movilidad.

Es un hecho que se tenía en cuenta, puesto que se conocía de una manera aproximada. Pero si a esto le sumamos que para que la batería se cargue de manera equilibrada y así preservar de la mejor manera sus características, hay que esperar unas 5 horas mínimo… Solo es posible 1 o 2 pruebas de vuelo al día.

En la práctica el proceso de sintonización de parámetros ha sido un proceso muy lento, ya que al cambiar las variables correspondientes a los PID y así ver el cambio respecto al anterior vuelo, había que esperar unas 5 horas o esperar al día siguiente… Y de esta manera es más difícil notar los cambios en los parámetros.

Para solventar este problema, se puede invertir más dinero para disponer de varias baterías de manera que se pueda disponer de todas cargadas al mismo tiempo y así realizar varias pruebas de vuelo seguidas.

6 CONCLUSIONES Y LÍNEAS FUTURAS

El proyecto se ha desarrollado como prototipo a un problema en plena investigación que se está desarrollando en el departamento de automática. El prototipo es un primer proyecto compuesto con componentes muy básicos que en conjunto sean capaces de lograr una serie de objetivos relacionados con la investigación vigente, pero adaptados a las posibilidades que el prototipo puede llevar a cabo.

Para entender el alcance del proyecto se volverá a presentar el objetivo de la investigación:

- Elaborar un dispositivo que sea capaz de manera autónoma de desplazarse a través de las zonas de los espejos repartidos por todo el solar.

  El vehículo elegido para satisfacer mejor los requerimientos ha sido un drone. Posee una gran capacidad de maniobrabilidad y es un hecho que en el aire el movimiento autónomo puede resultar mucho más simple dado la falta de obstáculos. Dicho drone ha sido dotado con una controladora de vuelo y un módulo GPS de manera que pueda realizar una serie de ordenes de manera autónoma ya definidas con anterioridad.

  El conjunto de órdenes se le llama misión, y entre ellas se encuentra tanto las órdenes de despegar, de aterrizar y de ir a un punto en el espacio. Dichas órdenes están configuradas junto a unas coordenadas en el espacio longitud, latitud, altura respecto al suelo.

  Dicho esto, cumplimos el objetivo al disponer de un dispositivo que se pueda mover por los espejos de manera autónoma. Solo habría que estudiar los puntos en el espacio y establecer una misión adecuada a ellos.

- Dicho dispositivo debe de ser capaz de realizar medidas de radiación solar en cada punto de interés de su ruta. Por lo que también debe ser capaz de pararse en dichos puntos y proporcionar una estabilidad necesaria para poder realizar las medidas.

  Otra orden que se puede implementar en la misión es la de permanecer en un punto estático. Dicha orden viene junto a unas coordenadas, sim embargo hay un control de estabilidad implementado en el drone que se encargará de realizar cualquier cambio en el drone para contrarrestar tanto diferencia de posiciones, como cambios en las variables geométricas del drone.

  Dicho control interno nos proporciona que el drone a pesar de cualquier perturbación como puede ser una ráfaga de viento, no solo no consiga desplazar al drone, sino que éste no se desestabilice. Por lo que podemos decir que hemos completado el objetivo impuesto.

- Dado que para la medición como bien se explicó en la introducción, es necesario que el sensor que realizará la medida de radiación esté alineado con el sol o la medición resultante no se considerará valida. Por lo que el dispositivo debe de ser capaz de poder alinear de manera autónoma el sensor al sol en cada medida.

  En este proyecto no se va a usar ningún sensor para medir la radiación solar, sin embargo se introduce el problema de alineamiento del sensor con el sol. Dado que El sensor tiene que estar alineado para que funcione correctamente, se ha diseñado una estructura con una cámara para simular al sensor y junto a 2 servos para poder controlar la estructura.

  La cámara emula al sensor ya que se procede a diseñar un controlador mediante una Raspberry para que así sea, pero también sirve como entrada al sistema para que éste pueda mover la plataforma mediante los servos y así mantener la cámara alineada.

  De esta manera podemos asumir que la solución del problema planteado es correcta y adecuada a la investigación ya que una vez que el sistema de alineamiento de la cámara funcione, únicamente debemos de incorporar el sensor en la plataforma y así nos aseguraremos de su buen funcionamiento también.

Para poder cumplir estos objetivos, se ha realizado un estudio muy amplio acerca de drones para entender conceptos de este campo así como el drone, como funciona y la funcionalidad de cada uno de sus

componentes, sin embargo a lo largo de todo el proyecto se han estado usando todo tipo de conocimientos aprendidos a lo largo de toda la carrera y de todo tipo de ámbitos. Esto refleja un efecto directo de la efectividad de la enseñanza impartida en el grado y de su utilidad en trabajos relacionados a estos como la elaboración de este proyecto.

También ha sido necesario el aprendizaje de lenguajes de programación que no se han dado a lo largo de la carrera como Python y Linux, además de distintos software como el diseño en 3D Solid Work. Destacar de este último programa la importancia de éste en la actualidad ya que se ha encontrado una gran facilidad para poder encontrar empresas que lleven a cabo impresiones en 3D, lo que hace de esta tecnología un recurso muy accesible y útil.

En conclusión podemos decir que hemos cumplido los objetivos con los que se diseñó este primer prototipo, sin embargo hay una serie de aspectos mejorables tanto de cara a las dificultades presentes en el prototipo como de cara a futuras líneas de investigación.

Dado que las dificultades de este proyecto se comentaron en el capítulo anterior de resultados experimentales, se procederá a especificar las posibles líneas futuras de investigación que se podrían realizar a partir de este proyecto:

- Una línea de estudio futura bastante evidente es el estudio del modelo de control de la cámara junto a los sensores de radiación solar. Dado que el sensor que se ha elegido para la investigación es un modelo muy caro, se puede probar con otros modelos más económicos y así empezar a realizar las pruebas con el sensor funcionando.

  Para adaptar las pruebas de cara al sensor ya especificado, se podría colocar en el sensor elegido para las pruebas una estructura cilíndrica de manera que se pueda simular la toma de medidas final.

  De esta manera podemos orientar el control de la plataforma de una manera mucho más objetiva, teniendo pruebas experimentales de las mediciones de radiación.

- También se puede investigar la opción de poder implementar en la estructura móvil unos servos que nos permiten conocer la posición angular real de éste en todo momento además de estudiar la posibilidad de incluir en el control dicha información.

- La cámara usada en este prototipo es muy básica con todas las restricciones que ello implica. Se podría investigar el uso de otro tipo de cámaras con más opciones a la hora de tomar fotos.

  Por ejemplo las cámaras que son capaces de captar la luz infrarroja. De esta manera se puede estudiar no solo si con dichas cámaras se puede conseguir un control más preciso, sino la posibilidad de poder localizar el sol a pesar de que se encuentra nublado. Dado que el sol emite luz infrarroja y parte de esta luz puede atravesar una cantidad finita de cúmulos de nubes.

- Puesto que en este proyecto se ha desarrollado un prototipo se ha mantenido un presupuesto muy reducido. Esto condiciona varios aspectos generales del drone: uno de ellos es la calidad de los componentes ya que en la mayoría de éstos mientras más prestaciones tenían más grandes y pesados eran además de caros.

  Esto supuso que además de diseñar un drone con los componentes más básicos y económicos resultó tener un peso para el cuál una configuración de 4 motores fueran suficientes. Una configuración de 6 u 8 motores hubiera sido mucho mejor de cara a la estabilidad del drone sin embargo por un criterio económico también se decidió usar la configuración de 4 motores.

  Para un proyecto con unos objetivos más grandes que el de este estudio, se necesitara una configuración de más motores. Dado que al tener más más motores se podría alcanzar:

  - Una mayor estabilidad.
  - Capacidad para disponer de una batería más pesada con mucho mayor autonomía.
  - Capacidad para disponer de más componentes como elementos de telemetría.
  - Capacidad para disponer de componentes más pesados y por ello con mejores prestaciones.
  - Capacidad para una plataforma de servos más robusta.

Todo ello sin sacrificar el empuje necesario para realizar cualquier maniobra que ya era capaz de hacer.

- Otro aspecto muy necesario para el estudio futuro de este proyecto es la posibilidad de conocer en tiempo real todos los aspectos del drone en pleno vuelo. Para ello existen ciertos dispositivos de telemetría que permiten modificar aspectos de la controladora de vuelo sin la necesidad de estar conectado por USB.

  De esta manera no solo podemos conocer los datos de vuelo, sino que podemos implementar nuevas rutas de vuelo o modificarlas en el aire. Otro gran conveniente de usar estos dispositivos es el de poder sintonizar de manera más eficiente ya que se pueden realizar varias pruebas con distintos sintonizados de los PID de manera muy seguida y así poder discernir mejor los cambios en este.

  También sería adecuado estudiar el poder emitir las emisiones tomadas por el sensor de radiación solar a distancia, ya que el objetivo del proyecto es conocer en qué puntos la radiación solar es menor para así tomar medidas y poder atenuar los efectos que esto provoca. Si hay que esperar a que el drone vuelva para conocer dichos datos se pierde eficiencia en el proceso.

- El prototipo ha sido diseñado para sea el usuario sea quien deba de realizar un estudio previo del terreno, para que pueda configurar una misión con unos puntos en el espacio tal que el drone no necesite disponer de un sistema de evasión de obstáculos.

  Sin embargo puede ser un punto de estudio dotar al drone de tal capacidad ya que pueden existir elementos que puedan cruzarse en su trayectoria que no formen parte del terreno.

7 BIBLIOGRAFÍA

- [1] Dyna, junio 2016, Magazine. «Evolución histórica de los vehículos aéreos no tripulados hasta la actualidad». Available: http://oa.upm.es/40803/1/INVE_MEM_2015_203893.pdf
- [2] Addati Gastón A. y Lance Gabriel, 2014. «Introducción a los UAV's, Drones o VANTs de uso civil». Available: https://www.econstor.eu/handle/10419/130802
- [3] Benjamin C. Kuo. «Sistemas de control automático».
- [4] Antoni Escrib Vidal, diciembre 2009. «La automática en la antigüedad». Available: https://www2.coitt.es/res/revistas/06d%20Rep%20Antiguedad%20OF9.pdf
- [5] Manuel Durán Fuentes, 2011. «Faros de Alenjadría y Brigantium». Available: http://www.traianvs.net/pdfs/2011_faro_brigantium.pdf
- [6] Mario Zamora Morillas «Estudio histórico tecnológico del molino hidráulico de rodezno de juan Tíscar». Available: http://tauja.ujaen.es/bitstream/10953.1/4053/1/TFG_Zamora_Morillas%2CMario.pdf
- [7] Julián Chaves Palacios. 2004. «Desarrollo tecnológico en la primera revolución industrial». Available: https://dialnet.unirioja.es/servlet/articulo?codigo=1158936
- [8] F. Torres. «Introducción a la automatización y el control». Available: https://rua.ua.es/dspace/bitstream/10045/18432/1/Tema%201_Introduccion.pdf
- [9] Jepherson Zapata Blandon y John Alquiber Sepúlveda Arbeláez, 2016. «Diseño de un drone para carga útil de 0.5 kg». Available: https://core.ac.uk/download/pdf/84108402.pdf
- [10] Sergio Moyano Díaz. «Diseño y construcción de un Quadcopter». Available: https://upcommons.upc.edu/bitstream/handle/2099.1/21902/102664.pdf
- [11] Sergio Vigorra Treviño, 2016. «Manual de montaje y configuración de multirrotores con autopiloto APM». Available: http://aeroclubmezquita.es/wp-content/uploads/2016/10/Manual-de-montaje-y-configuracio%CC%81n-de-multirrotores-con-autopiloto-APM.pdf
- [12] Emilio Marín Fernández. 2017. «Plataforma de telecontrol de un drone comercial».
- [13] Daniel Plaza Rey, junio 2017. «Navegación autónoma de drones y automatización de rutas aplicadas a la limpieza de edificios». Available: https://upcommons.upc.edu/bitstream/handle/2117/106479/MemoriaFinalDanielPlaza.pdf?sequence=1&isAllowed=y
- [14] Marco Andrés Banegas Guerrero y William Orlando Villa Villa, 2018. «Sistema de seguridad residencial en zona de expansión urbana y monitoreo autónomo empleando un vehículo aéreo no tripulado». Available: http://dspace.uazuay.edu.ec/bitstream/datos/8500/1/14218.pdf
- [15] http://ardupilot.org/
- [16] Lohit Ujjainiya y M. Kalyan Chakravarthi. «Raspberry – Pi based cost-effective vehicle collision avoidance system using image processing».
- [17] - https://www.raspberrypi.org/
- [18] - https://element14.com
- [19] - https://picamera.readthedocs.io/en/release-1.13/
- [20] - https://opencv.org/
- [21] - http://abyz.me.uk/rpi/pigpio/python.html
- [22] - http://www.numpy.org/

- [23] P. Gil, F. Torres, G. Ortiz. «Detección de objetos por segmentación multinivel combinada de espacios de color». Available: https://rua.ua.es/dspace/bitstream/10045/2179/1/Jornadas2004.pdf
- [24] M. R. Arahal, «Separación de regiones» de *Apuntes de la asignatura "Sistemas de Percepción"*, 2017.
- [25] https://docs.opencv.org/3.1.0/da/d53/tutorial_py_houghcircles.html
- [26] David M. Caruso, Salvador E. Tropea. «Controlador tipo PID, sobre microcontrolador embebido en FPGA». Available: https://www.researchgate.net/profile/Salvador_Tropea/publication/266852676_Controlador_tipo_PID_sobre_microcontrolador_embebido_en_FPGA/links/543d45610cf2d6934ebaf435/Controlador-tipo-PID-sobre-microcontrolador-embebido-en-FPGA.pdf
- [27] Jhon Jairo Zambrano 2013, diapositivas. «Automatización industrial». Available: Jhon Jairo Zambrano 2013. https://prezi.com/5i5kfq3igs4z/tema-control-en-cascada/
- [28] «Understanding Euler Angles». http://www.chrobotics.com/library/understanding-euler-angles
- [29] http://ardupilot.org/dev/docs/extended-kalman-filter.html#extended-kalman-filter
- [30] Ariel Peláez Valdés, 2016. «Análisis del filtro extendido de Kalman presente en el Pixhawk, profundizando en cuanto a su mecanismo de ajuste». Available: http://dspace.uclv.edu.cu/bitstream/handle/123456789/6342/Ariel%20Pel%C3%A1ez%20Vald%C3%A9s.pdf?sequence=1&isAllowed=y
- [31] Álvaro Solera Ramírez, Julio 2003. «El filtro de Kalman». Available: https://activos.bccr.fi.cr/sitios/bccr/investigacioneseconomicas/DocMetodosCuantitativos/Filtro_de_Kalman.pdf
- [32] http://ardupilot.org/copter/docs/flight-modes.html

8 ANEXO

Programa de seguimiento final del capítulo 4:

```
# HACER SIEMPRE ANTES    sudo pigpiod
# coding=utf-8

#////////////////////////////////////////////////////////////////
#----------------------------------------CABECERAS
#////////////////////////////////////////////////////////////////q

from picamera.array import PiRGBArray
from picamera import PiCamera

import time
import cv2
import numpy as np
import pigpio
import math

def nothing(x):
    pass


#////////////////////////////////////////////////////////////////
#---PRE-CONFIGURACION DEL MODO DE ACCION
#////////////////////////////////////////////////////////////////

aux_w = 0 #indica si se escriben parámetros en archivos o no (1/0)
aux_t = 1 #indica si se ejecuta el control en los servos o no (1/0)
aux_v = 1 #indica si se muestra por pantalla las ventanas de imágenes (1/0)
aux_p = 0 #indica si se muestra por pantalla el color elegido y la posibilidad de modificarlo (1/0)
aux_a = 1 #indica si se activa o no el seguimiento autónomo en la fase de inactividad (1/0)
aux_s = 0 #indica si se anula la detección de objetivo para probar la búsqueda autónoma (1/0)

aux_tipo_met = 2 #indica el tipo de método para identificar: 1-> por Hough / 2-> por momentos



#////////////////////////////////////////////////////////////////
#------------------------------INICIALIZACIONES
#////////////////////////////////////////////////////////////////

aux_c = 0   #necesaria para el control, indica si se ha detectado o no el círculo (1/0)

pi=pigpio.pi()  # Inicialización de los pines correspondientes a los servos y puesta de servos en posición de reposo
```

```
pi.set_mode(27, pigpio.OUTPUT)
pi.set_mode(22, pigpio.OUTPUT)
puls_x=1600
puls_y=2250
pi.set_servo_pulsewidth(27, puls_x)
pi.set_servo_pulsewidth(22, puls_y)

kernel_6 = np.ones((6,6),np.uint8) # Declaración necesaria para el tratamiento de imágenes
kernel_8 = np.ones((8,8),np.uint8)

nuevo_b = 1 # Declaración necesaria para la búsqueda autónoma
cont_p_22 = 0
cont_p_27 = 0
cont_v = 0
cont_t = 0
puls_p = 0
fin_s = 0

if aux_v is 1:  #declaración de las ventanas emergentes
   cv2.namedWindow('tracking')
   #cv2.namedWindow('rojo')
   #cv2.namedWindow('verde')
   #cv2.namedWindow('azul')
   #cv2.namedWindow('color hsv')
   #cv2.namedWindow('color_con_mask_0')
   cv2.namedWindow('MASK')



if aux_w is 1:  #declaración del archivo donde se escribirán los datos
   archivo = open('datos.txt', 'w');

#320, 240
camera = PiCamera()     #declaración y configuración de la cámara
camera.resolution = (480,320)
camera.framerate = 70
#Para trackear el sol
camera.led=False
camera.ISO=100
camera.brightness=30
camera.image_effect='solarize'
camera.shutter_speed= 300000

rawCapture = PiRGBArray(camera, size=(480,320))

# Constantes del PID_1
Kx = 0.5
Kdx = 0.3
```

```
Kix = 0.05

Ix = 0
dist_x = 0
dist_a_x = 0
Ax = 0

# Constantes del PID_2
Ky = 0.5
Kdy = 0.3
Kiy = 0.05

Iy = 0
dist_y = 0
dist_a_y = 0
Ay = 0

# Valores del color RGB
# Para el rosa:
Rmin=195
Rmax=255
Bmin=160
Bmax=255
Gmin=75
Gmax=195
# Para la luz:
Rmin=230
Rmax=255
Bmin=230
Bmax=255
Gmin=230
Gmax=255

# Creación de barra para modificar el filtro de color
if aux_v is 1 and aux_p is 1:
    cv2.createTrackbar('Rmin', 'tracking', 230, 255, nothing)
    cv2.createTrackbar('Rmax', 'tracking', 255, 255, nothing)

    cv2.createTrackbar('Gmin', 'tracking', 230, 255, nothing)
    cv2.createTrackbar('Gmax', 'tracking', 255, 255, nothing)

    cv2.createTrackbar('Bmin', 'tracking', 230, 255, nothing)
    cv2.createTrackbar('Bmax', 'tracking', 255, 255, nothing)
```

```
print("COMIENZA EL PROGRAMA")
#////////////////////////////////////////////////////////////////
#----------------------------------BUCLE PRINCIPAL
#////////////////////////////////////////////////////////////////

for frame in camera.capture_continuous(rawCapture, format="bgr", use_video_port=True):

   #-------------------------------------------------------------------TRATAMIENTO DE IMAGENES

   frame=frame.array
   cv2.line(frame, (0,160), (480,160), (255,0,0), 2)
   cv2.line(frame, (240,0), (240,320), (255,0,0), 2)

   if aux_v is 1 and aux_p is 1:  #Recogemos el valor del color seleccionado
      rmax = cv2.getTrackbarPos('Rmax','tracking')
      rmin = cv2.getTrackbarPos('Rmin','tracking')
      gmax = cv2.getTrackbarPos('Gmax','tracking')
      gmin = cv2.getTrackbarPos('Gmin','tracking')
      bmax = cv2.getTrackbarPos('Bmax','tracking')
      bmin = cv2.getTrackbarPos('Bmin','tracking')                    #////////////////////////////
   else:       #En caso contrario, usamos el color siguiente -CUIDADO AQUI, COLOR A DEFINIR
      rmax = 200                                                      #////////////////////////////
      rmin = 50
      gmax = 200
      gmin = 50
      bmax = 200
      bmin = 50

   #definimos el filtro de color
   bajos = np.array([gmin,bmin,rmin], dtype=np.uint8)
   altos = np.array([gmax,bmax,rmax], dtype=np.uint8)

   #aplicamos el filtro de color
   mask = cv2.inRange( frame , bajos, altos)

   #aplicamos filtros para reducir el ruido
   mask1 = cv2.morphologyEx( mask, cv2.MORPH_CLOSE, kernel_6)
   mask1 = cv2.morphologyEx( mask1, cv2.MORPH_OPEN, kernel_6)
   #x2
   mask1 = cv2.morphologyEx( mask1,cv2.MORPH_CLOSE, kernel_8)
   mask1 = cv2.morphologyEx( mask1, cv2.MORPH_OPEN, kernel_8)
```

```
#-------------------------------------------------------------------------------METODO DE HOUGH CIRCLES
    if aux_tipo_met == 1:
        #detectamos círculos en la imagen binaria resultante
        circles=cv2.HoughCircles(mask1, cv2.cv.CV_HOUGH_GRADIENT, 40, 1000)
        #en el caso en el que se detecte algún circulo
        if circles is not None:
            circles=np.round(circles[0, :]).astype("int") #coordenadas polares
            for (x,y,r) in circles:
                rad = r
                if rad < 60:    #y el radio del circulo conseguido sea menor de 60 pixeles
                    #esto es para anular la búsqueda y así poder hacer pruebas con la búsqueda autónoma
                    if aux_s == 0:
                        aux_c = 1   #notificamos que hemos encontrado el circulo
                        nuevo_b = 1
                    dist_x = x - 240
                    dist_y = y - 160
                    #dibujamos sobre la imagen el circulo conseguido y el centro
                    cv2.circle(frame, (x,y), r, (0,255,0), 3)
                    cv2.rectangle(frame, (x-5, y-5), (x+5, y+5), (0,255,0), -1)
                    cv2.putText(frame, "["+ str(x)+","+str(y)+","+ str(r)+"]", (x+r,y+r), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (255,0,0), 2)
                    cv2.line(frame, (x,y), (240,y), (0,255,0), 2)
                    cv2.line(frame, (x,y), (x,160), (0,255,0), 2)


#-------------------------------------------------------------------------------------------METODO DE MOMENTOS
    if aux_tipo_met == 2:
        #detectamos los contornos de la imagen una vez filtrada
        mask_c = mask1
        contours, hierarchy = cv2.findContours(mask1.copy(), cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
        #~ if contours:
            #~ cv2.drawContours( frame, contours, -1, (255,0,0), 2)
        a=len(contours)
        if a is not 0:
            #para evitar errores nos quedamos únicamente con el contorno mayor
            mayor_contorno = max( contours, key = cv2.contourArea)
            #calculamos los momentos del contorno elegido
            momentos = cv2.moments(mayor_contorno)
            #calculamos el centro de gravedad del contorno
            x = int(momentos['m10']/momentos['m00'])
            y = int(momentos['m01']/momentos['m00'])
            #calculamos el radio
            r= int( math.sqrt( momentos['m00']/3.141516 ) )
            #~ (x,y),radius = cv2.minEnclosingCircle(cnt)
            if aux_s == 0:
                aux_c = 1   #notificamos que hemos encontrado el circulo
                nuevo_b = 1
                dist_x = x - 240
                dist_y = y - 160
```

```
        #dibujamos sobre la imagen el circulo conseguido y el centro
        cv2.circle(frame, (x,y), r, (0,255,0), 3)
        cv2.rectangle(frame, (x-5, y-5), (x+5, y+5), (0,255,0), -1)
        cv2.putText(frame, "["+ str(x)+","+str(y)+","+ str(r)+"]", (x+r,y+r), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (255,0,0), 2)
        cv2.line(frame, (x,y), (x,160), (0,255,0), 2)
        if puls_x == 2100 or puls_x == 1100:
            cv2.line(frame, (x,y), (240,y), (0,0,255), 3)
        else:
            cv2.line(frame, (x,y), (240,y), (0,255,0), 2)
#-------------------------------------------------------------------------------------------------------------------------------------------------------

    if aux_c is 0:
        #en el caso en el que no se encuentre un circulo, lo notificamos
        cv2.putText(frame, "NO OBJETIVE", (40, 300), cv2.FONT_HERSHEY_SIMPLEX, 2, (0,0,255), 2)
    else:
        #en caso contrario, calculamos las distancias respecto al centro y las escribimos en la imagen
        cv2.putText(frame, "Error x,y de: ", (10, 300), cv2.FONT_HERSHEY_SIMPLEX, 1, (255,255,255), 2)
        cv2.putText(frame, str(dist_x), (210, 300), cv2.FONT_HERSHEY_SIMPLEX, 1.5, (255,255,255), 2)
        cv2.putText(frame, " , ", (325, 300), cv2.FONT_HERSHEY_SIMPLEX, 1, (255,255,255), 2)
        cv2.putText(frame, str(dist_y), (350, 300), cv2.FONT_HERSHEY_SIMPLEX, 1.5, (255,255,255), 2)



    if aux_v is 1:  #mostramos las imágenes conseguidas por pantalla
        cv2.imshow('tracking',frame)
        #cv2.imshow('rojo',r)
        #cv2.imshow('verde',g)
        #cv2.imshow('azul',b)
        #cv2.imshow('color hsv',hsv)
        #cv2.imshow('color_con_mask_0',mask)
        cv2.imshow('MASK',mask1)

    rawCapture.truncate(0)



    #-----------------------------------------------------------------CONTROL

    if aux_c is 1 and aux_t is 1: #solo si el control está activo y si se ha encontrado objetivo

        #-----------------------------------------------------------------PID_2
        #si no se encuentra en la zona central de seguridad:
        if dist_y < -5 or dist_y > 5:
            #cálculo de la salida del PID
            Ay = float( float(Ky * dist_y) + float(Kdy * (dist_y - dist_a_y))  + float(Kiy * Iy) )
            Iy = Iy + dist_y
            puls_y = puls_y + Ay
            dist_a_y = dist_y
```

```
        # limitaciones
        if puls_y > 2500:
            puls_y = 2500
            Iy = 0
        if puls_y < 2000:
            puls_y = 2000
            Iy = 0
        # escritura del movimiento del servo
        puls_y = int(puls_y)
        pi.set_servo_pulsewidth(22, puls_y)

    if aux_w is 1: #escritura de los datos en el archivo
        archivo.write('%s '%puls_y)
        archivo.write('%s '%dist_y)
        archivo.write('%s '%Ay)

    #-----------------------------------------------------------------PID_1
    #si no se encuentra en la zona central de seguridad:
    if dist_x < -5 or dist_x > 5:
        #cálculo de la salida del PID
        Ax = Kx * dist_x + Kdx * (dist_x - dist_a_x)  + Kix * Ix
        Ix = Ix + dist_x
        puls_x = puls_x + Ax
        dist_a_x = dist_x
        # limitaciones
        if puls_x > 2100:
            puls_x = 2100
            Ix = 0
        if puls_x < 1100:
            puls_x = 1100
            Ix = 0
        # escritura del movimiento del servo
        puls_x = int(puls_x)
        pi.set_servo_pulsewidth(27, puls_x)

    if aux_w is 1: #escritura de los datos en el archivo
        archivo.write('%s '%puls_x)
        archivo.write('%s '%dist_x)
        archivo.write('%s '%Ax)
        archivo.write('\n')

else:
    if aux_w == 1:
        archivo.write('0 ')
        archivo.write('0 ')
        archivo.write('0 ')
        archivo.write('0 ')
```

```
        archivo.write('0 ')
        archivo.write('0 ')
        archivo.write('\n')
    if aux_a == 1:

        if nuevo_b == 1: #Para reiniciar la espera cada vez que encuentre objetivo
            cont_t = 0
            nuevo_b = 0
            fin_s = 0

        cont_t = cont_t+1

        if cont_t > 50: #Numero de esperas para pasar a modo búsqueda

            if cont_v == 0:  #condiciones iniciales
              pi.set_servo_pulsewidth(22, 2500)
              pi.set_servo_pulsewidth(27, 1200)

            cont_p_27 = cont_p_27 +1

            if cont_p_27 > 41:
                if fin_s == 0:
                    cont_v = cont_v +1

                if cont_v == 6:
                    cont_v = 0
                    fin_s = 1

                if fin_s == 0:
                    cont_p_27 = 0
                    cont_p_22 = cont_v*10
                    puls_p_22 = int( 2500 - 10*(cont_p_22) )
                    pi.set_servo_pulsewidth(22, puls_p_22)

            if fin_s == 0:
                puls_p_27 = int( 1200 + 21*(cont_p_27) )
                pi.set_servo_pulsewidth(27, puls_p_27)
            else:
                pi.set_servo_pulsewidth(27, 1600)
                pi.set_servo_pulsewidth(22, 2250)

#------------------------------------------------------------------FINALIZACIÓN
aux_c = 0
k=cv2.waitKey(1) & 0xFF
if k==27:
    if aux_w is 1:
        archivo.close()
```

```
    break

 #//////////////////////////////////////////////////////////////////
#-------------------CIERRE SEGURO DEL SISTEMA
#//////////////////////////////////////////////////////////////////


pi.set_servo_pulsewidth(27,0)
pi.set_servo_pulsewidth(22,0)
pi.stop()
```