



Proyecto Final de Carrera
INGENIERÍA INDUSTRIAL
Modelado, simulación, construcción y
control de un Quadcopter
Memoria

Resumen

En el presente proyecto se ha tenido como meta el control de un Quadcopter. Para tal propósito se ha linealizado el modelo no lineal obtenido y controlado mediante un Regulador Lineal Cuadrático (LQR). El contenido de este trabajo está organizado en 8 capítulos y 4 anexos.

En el capítulo 1 se presentan los motivos por los que se ha elegido este tema. También se mencionan requerimientos previos que se han requerido para no tener insalvables dificultades.

En el capítulo 2, como Introducción, se hace un estudio del arte, con breves pinceladas del basto campo que hoy en día se encuentra sobre este tema. Además se describen los objetivos de este proyecto. Éstos no son más que el resolver todo problema que haya surgido durante esta aventura.

Se obtiene el modelo a partir de los parámetros característicos del sistema físico en el capítulo 3. En linealizar el sistema en el punto de equilibrio se llega a la conclusión de que las variables que lo forman se desacoplan y se comportan como un conjunto de dobles integradores. Ésto permite que la ley de control sea sencilla y justifica el que se hayan usado controladores PID tan asiduamente. También se presenta el modelo implementado en Simulink, con el que se probará el control obtenido posteriormente para verificar su eficiencia.

El capítulo anterior permite que en el 4 se obtengan las acciones de control $u = -k \cdot X$ por medio de un regulador LQR (Linear Quadratic Regulator) y se ataca al sistema con la ley obtenida.

En el capítulo 6 se describen tanto los componentes que se utilizan como el proceso de montaje paso a paso. Se justifica la elección de cada elemento tomando como criterio tanto las características de otros elementos como el dimensionado mismo del Quadcopter.

En los capítulos 7, 8 y 9 se tiene un análisis económico, un estudio de impacto ambiental y las conclusiones respectivamente. En el primero se desglosan los costes materiales y se valoran las horas invertidas, considerando que el trabajo lo realiza un estudiante. En el estudio del impacto ambiental se describen brevemente los principales efectos en el medio ambiente de los componentes que se utilizan. En el último capítulo se relata con qué se queda uno en limpio tras acabar esta empresa y cómo se espera continuar para futuros desarrollos.

En el Anexo A se muestra mediante códigos Matlab de dónde se ha extraído el vector de fuerzas del modelo. De esta manera se justifica toda otra opción a la linealización, en estas circunstancias, inviable. También en este código se linealiza el sistema y se obtienen las constantes K de la realimentación.

En el Anexo B se detalla el proceso de grabado y posteriores modificaciones realiza-

das en el sistema operativo utilizado (Raspbian), para condicionar la Raspberry Pi tal según se ha requerido.

En el Anexo C se describe el filtro de Kalman y cómo se ha implementado.

En el Anexo D se describe cómo se han encontrado los parámetros que representan al sistema físico que se tiene. Para ello se han caracterizado los motores con sus curvas de fuerza y consumo, además la masa, inercias y distancias características propias del aparato.

Índice general

Resumen	1
1. Prefacio	7
1.1. Motivación	7
1.2. Requerimientos previos	7
2. Introducción	9
2.1. Estudio del arte	9
2.2. Objetivos del proyecto	10
2.3. Alcance	11
3. Definición del modelo	13
3.1. Definición de las variables	13
3.2. Obtención del modelo	16
3.3. Representación del modelo con Simulink	22
4. Diseño del controlador	27
4.1. Regulador Quadrático Lineal	27
4.2. Obtención de la ley de control	28
4.3. Aplicación en el modelo	29
5. Implementación del control	33
6. Construcción del Quadcopter	35
6.1. Descripción de los componentes	35
6.2. Montaje	42
7. Análisis económico	45
8. Impacto ambiental	47
9. Conclusiones	49
9.1. Futuras aplicaciones	49

Índice de figuras

2.1. Quadcopter de ejemplo con cámara	9
2.2. Ejemplo de Quadcopter con 6 motores	10
3.1. Marcos de referencia en el quadcopter	13
3.2. Modelo en Simulink de la dinámica del Quadcopter	22
4.1. Modelo en Simulink del lazo cerrado aplicado en el Quadcopter	30
4.2. Respuesta del ángulo ϕ en la simulación	32
4.3. Acciones de los motores en la simulación	32
5.1. Flowchart del código a ejecutar	33
6.1. Raspberry Pi a utilizar	35
6.2. Conectores de la Raspberry Pi	36
6.3. IMU MPU-6050	36
6.4. Emisora y Receptor	37
6.5. Batería LiPo empleada	37
6.6. Esquema del Regulador Step-Down	38
6.7. Esquema de un Variador como inversor trifásico	39
6.8. Tabla de Set up de un ESC	40
6.9. Piezas que componen la estructura del Quadcopter	40
6.10. Par de hélices 8x4 simétricas	41
6.11. Representación de la Longitud y Pitch de una hélice de dos aspas	41
6.12. Ensamblaje del esqueleto del Quadcopter	42
6.13. Ensamblaje de la montura de la RPi y el Receptor	42
6.14. Detalle del cruce de fases en los motores	43
6.15. Ensamblaje y detalle de una de las patas.	43
6.16. Vehículo finalizado	44
9.1. Detección del sensor MPU-6050	62
9.2. Función de densidad de la variable de estado $\hat{x}_{k k-1}$	64
9.3. Comparación con función de densidad de la medición z_k	64
9.4. Bancada para la determinación de las inercias	68

Capítulo 1

Prefacio

1.1. Motivación

La principal motivación de este proyecto es la de aplicar por uno mismo los conocimientos básicos adquiridos en la carrera, y más en particular en el área del control en haber cursado la intensificación de automática.

En plantear un tema para el proyecto rápidamente surgió la idea de realizarlo sobre el control de un sistema mecánico, y más en particular sobre uno que estuviera actualmente emergiendo tanto en mercados como en el campo de la investigación.

De entre las diferentes alternativas, un quadcopter es la más atractiva para el proyecto tanto por su simplicidad constructiva como por su no excesivo coste. Existen actualmente en el mercado infinidad de proveedores para los componentes necesarios para construir un quadcopter, con un gran abanico de opciones de entre las que escoger cada elemento como motores, baterías, electrónica, etc.

Las posibles aplicaciones son numerosas, tanto que aún no se han explotado todas las posibles. Como ejemplo: vigilancia de superficies abiertas, transporte de pequeños paquetes, herramienta de ocio, entre otras.

Entender el funcionamiento y familiarizarse con el mecanismo es una inquietud que se ha mantenido insatisfecha durante largo tiempo. Con este pretexto se pretende acabar de justificar la elección del tema de esta pequeña gran empresa.

1.2. Requerimientos previos

Es necesario tener ciertos conocimientos mínimos en automática para poder controlar el quadcopter, así como la inquietud de aprender lo que sea necesario para cumplir, en la medida de lo posible, los objetivos iniciales del presente proyecto.

Nociones de mecánica son necesarias para elaborar e interpretar el sistema sobre el que se trabaje, así como sus resultados. Conocimientos del lenguaje *C* son imprescindibles ya que el código del programa que realiza los cálculos está escrito en este mismo lenguaje.

Para adaptar los elementos que componen el aparato y afianzarlos a la estructura (Frame) del Quadcopter ha sido de gran ayuda contar con el servicio de una impresora 3D de creación también propia. Finalmente, estar familiarizado con el argot y elementos de un Quadcopter, así como motores, emisoras de radio, entre otros es de ayuda.

Superar las dificultades que han surgido, teniendo en cuenta que el fin no justifica los medios, es la praxis que se ha aplicado a la hora de realizar este trabajo. Finalmente, recordar reminiscencias de la carrera es siempre útil e interesante, así que conceptos adquiridos pero no utilizados se esperan aplicar.

Capítulo 2

Introducción

Un Quadcopter es un vehículo volador no tripulado (*Unmanned Aerial Vehicle* o *UAV*) que se caracteriza por tener cuatro motores como actuadores en vez de dos como en el caso de los helicópteros. Este tipo de autogiro intenta obtener una flotabilidad más estable y vuelo preciso balanceando las fuerzas producidas por los cuatro motores y surge de la necesidad de tener un aparato de vuelo de dimensiones reducidas y gran estabilidad, con prestaciones superiores a las de un helicóptero convencional.



Figura 2.1: Quadcopter de ejemplo con cámara

Una de las ventajas que se obtiene con el cambio es la mayor capacidad de carga ya que tiene 4 motores para soportar el peso. La estabilidad del vehículo mejora en permitir aterrizajes y despegues verticales con una mayor maniobrabilidad. También puede trabajar en áreas de difícil acceso o más agresivas, como con lluvia y viento.

Esquemáticamente se puede representar como una estructura en *X* con su centroide coincidiendo con el centro de masas y cuatro actuadores en las puntas de cada brazo, todos ellos apuntando en la misma dirección y sentido pero con giros de aspa en sentido contrario, pero igual en lados opuestos.

2.1. Estudio del arte

En la actualidad el campo referente a estos aparatos se ha diversificado tanto que se pueden encontrar muchas variedades y tipos de Quadcopters. Se ha hecho mención

del modelo de cuatro motores, pero bien pueden encontrarse de tres hasta 8 motores, sino más en casos más concretos.



Figura 2.2: Ejemplo de Quadcopter con 6 motores

Con el objetivo de desarrollar un *UAV* que realice tareas de forma totalmente autónoma, se han diseñado sensores para estimar los estados del vehículo de manera más rápida y eficiente, así como diferentes estrategias para controlar la estabilidad y orientación del mismo. Se ha utilizado visión por computador para realizar aterrizajes, detección de objetivos y navegación autónoma. También pueden combinarse diferentes técnicas con el uso del GPS para asegurar una buena localización del aparato.

Para controlar esta clase de aparatos se utilizan diferentes leyes de control, así como PID, LQR, Redes Neuronales, Machine Learning, entre otros. La más utilizada es sin duda el PID por su fácil implementación y poca carga computacional.

2.2. Objetivos del proyecto

La meta que se persigue es la de hacer planear un Quadcopter en direcciones horizontales. A la vez, éste es un excelente subterfugio para cumplir con otros objetivos relacionados con este fin como el control de motores, el inteligente uso de la electrónica y correcto diseño mecánico del aparato. Además, se espera haber ampliado la visión que uno tiene en cada campo.

De entre todas las disciplinas involucradas, se espera acabar familiarizado con las facetas relacionadas con la presente empresa para clarificar la relación entre los elementos que componen un Quadcopter. A saber, se espera poder acabar entendiendo la electrónica, mecánica y conocimientos de control que reinan en el sistema.

En el caso de la electrónica se espera entender y poder implementar sencillos protocolos de comunicación, en este caso *I2C* entre los sensores y la Raspberry Pi. Ésta misma es el centro neurálgico del Quadcopter, así que es esencial saber utilizarla como se requiere para que pueda recibir consignas, estados y enviar la correspondiente acción

de control a los motores para que se llegue al estado final deseado.

Por parte de la mecánica, es primordial tener conocimientos sólidos de conceptos como el Teorema de la Cantidad de Movimiento y el de la conservación del Momento de Inercia, así como poder calcular el Lagrangiano (\mathcal{L}) para generar un modelo verosímil y que represente fidedignamente la realidad para que se desarrolle de manera efectiva el control.

Finalmente, y como más serio objetivo de toda esta aventura, es el de haber afianzado lo que se creía entendido en relación al control. Asimilar otros métodos de control aplicables a sistemas multilineales no contemplados en la carrera es un excitante reto que se espera haber superado con éxito.

Entonces, se entiende este trabajo como la causa de la motivación a cumplir con un objetivo multidisciplinar, como puede serlo cualquiera en el campo de la robótica, en el que se debe tener constancia de muchos aspectos relevantes, como el de la economía. Poder tener idea de lo que cuesta un robot es de gran importancia si uno se embarca en un proyecto de envergadura, ya que se quiere que los pronósticos sobre el presupuesto cuadren con la futura realidad.

2.3. Alcance

En este proyecto se ha perseguido el objetivo de controlar un Quadcopter mediante un regulador LQR. Para ello se ha obtenido un modelo lineal parametrizado con la información representativa propia del sistema real que representa: masas, inercias, distancias y caracterización de los actuadores, al que se le ha aplicado la misma ley de control que al sistema real.

La construcción del mismo se ha llevado a cabo mediante piezas compradas por separado sin recurrir a kits montados o modelos completos, como Ardupilot o un AR Drone. No se ha pretendido construir un vehículo totalmente robusto, sino una plataforma de trabajo sobre la que se comparan comportamientos con su simulación y a la que se intenta abordar un control.

No se analizan ni comparan los resultados con otros posibles controles a implementar, tales como el PID. Se juzgará la calidad del vuelo según su aparente estabilidad y si se mantiene la posición asignada como consigna.

Capítulo 3

Definición del modelo

3.1. Definición de las variables

Para caracterizar la planta con la que es trabajará, es necesario obtener un modelo del Quadcopter. Las constantes propias del modelo se dejarán en forma de parámetros a calibrar una vez se tenga el objeto físico. De esta manera el modelo será general para todo quadcopter que comparta la misma familia de parámetros. Es necesario considerar dos marcos de referencia: el inercial formado por los ejes x, y, z y el del cuerpo (Body) formado por los ejes x_B, y_B, z_B . El primero tiene la perspectiva de el observador en tierra, estático, mientras que el segundo es solidario a la estructura. Según la orientación de los ejes del cuerpo con esta referencia se pueden dar los siguientes dos casos:

- **Cross type:** Los ejes de coordenadas coinciden con los brazos de la estructura ya que se tienen los actuadores en las puntas de cada brazo.
- **X-type:** Los ejes y la estructura forman 45. Se tienen entonces dos motores en la parte delantera y dos en la trasera.

Por ser más usual la primera opción, se decide utilizar la configuración *Crosstype* tal y como se tiene en la figura 3.1.

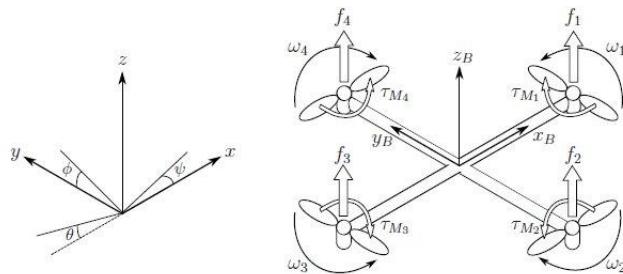


Figura 3.1: Marcos de referencia en el quadcopter

Se supone que el objeto es un rotor esférico, y por tanto su tensor de inercia es diagonal:

$$\mathbf{I} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \quad (3.1)$$

Se define la posición lineal absoluta con las coordenadas x, y, z con vector $\boldsymbol{\xi}$ e igualmente para la posición angular a partir de $\boldsymbol{\eta}$ según:

$$\boldsymbol{\xi} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \quad \boldsymbol{\eta} = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix}, \quad \mathbf{q} = \begin{bmatrix} \boldsymbol{\xi} \\ \boldsymbol{\eta} \end{bmatrix} \quad (3.2)$$

donde ϕ es el ángulo de cabeceo (Pitch), θ es el de alabeo (Roll) y ψ el de guiñada o deriva (Yaw).

Para la orientación angular entre los dos marcos se tiene un sistema de referencia con ángulos Tait-Bryan, donde la matriz de transformación es:

$$\mathbf{R}(\phi, \theta, \psi) = \begin{bmatrix} C_\psi C_\theta & C_\psi S_\theta S_\phi - S_\psi C_\phi & C_\psi S_\theta C_\phi + S_\psi S_\phi \\ S_\psi C_\theta & S_\psi S_\theta S_\phi + C_\psi C_\phi & S_\psi S_\theta C_\phi - C_\psi C_\phi \\ -S_\theta & C_\theta C_\phi & C_\theta C_\phi \end{bmatrix} \quad (3.3)$$

con $C_\phi = \cos(\phi)$ i $S_\phi = \sin(\phi)$. Ésto se obtiene del sistema de orientación intrínseco basado en las rotaciones fundamentales respecto ϕ, θ y ψ :

$$\mathbf{R}(\phi, \theta, \psi) = \mathbf{R}_1(\psi) \cdot \mathbf{R}_2(\theta) \cdot \mathbf{R}_3(\phi) \quad (3.4)$$

donde las matrices de rotación son de la forma:

$$\mathbf{R}(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi) & \cos(\phi) \end{bmatrix} \quad \mathbf{R}(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \quad (3.5)$$

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

Las velocidades lineales en el marco de referencia del cuerpo (Body Frame) se representen con el vector v_B y las velocidades angulares con γ según:

$$\mathbf{v}_B = \begin{bmatrix} v_{x,b} \\ v_{y,b} \\ v_{z,b} \end{bmatrix} \quad \boldsymbol{\gamma} = \begin{bmatrix} p \\ n \\ r \end{bmatrix} \quad (3.6)$$

En cambio, las velocidades en el marco de referencia inercial (Inertial Frame) se representan por $\dot{\xi}$ para las velocidades lineales y con $\dot{\eta}$ para las angulares:

$$\dot{\xi} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \quad \dot{\eta} = \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \quad (3.7)$$

Ya que la derivada de los ángulos ϕ , θ y ψ no es igual al vector de velocidades angulares $\gamma \neq \dot{\eta}$, es necesario tener un cambio de base para relacionar las velocidades en el marco de referencia inercial con las del cuerpo. Para ello se calcula la matriz de transformación de velocidades angulares entre las referencias Inercial y Body como:

$$\begin{bmatrix} p \\ n \\ r \end{bmatrix} = \mathbf{I}_3 \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} + \mathbf{R}(\phi) \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + \mathbf{R}(\phi) \cdot \mathbf{R}(\theta) \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} = [\mathbf{W}_{\eta}] \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \quad (3.8)$$

y se obtiene la matriz \mathbf{W}_{η} :

$$\mathbf{W}_{\eta} = \begin{bmatrix} 1 & 0 & \sin(\theta) \\ 0 & \cos(\phi) & -\sin(\phi)\cos(\theta) \\ 0 & \sin(\phi) & \cos(\phi)\cos(\theta) \end{bmatrix} \quad \text{con} \quad \gamma = [\mathbf{W}_{\eta}] \dot{\eta} \quad (3.9)$$

Las fuerzas de sustentación y velocidades angulares de los cuatro actuadores son f_1, f_2, f_3, f_4 y w_1, w_2, w_3, w_4 respectivamente. Siguiendo la orientación de la figura 3.1, con el objetivo de poder anular los momentos producidos en el eje z_B (en el marco del cuerpo B) el sentido de giro de los actuadores 4 y 2 son en el de las agujas del reloj (clockwise) y el de los 1 y 3 en sentido contrario (counterclockwise).

Interesa conocer qué fuerza y momento aportará cada motor dada una velocidad angular conocida. Se supone poder controlar ambos y cómo aproximación inicial se considera que están relacionados con la velocidad angular de la forma:

$$\begin{aligned} f_i &= kw_i^2 \\ \tau_{M_i} &= bw_i^2 \end{aligned} \quad (3.10)$$

Por tanto el empuje total \mathbf{T}_B proporcionado en la dirección z_B y los momentos generados τ_B por los motores son:

$$\mathbf{T}_B = k \left(\sum_{i=1}^4 w_i^2 \right) e_{z_B} = \begin{bmatrix} 0 \\ 0 \\ \sum_{i=1}^4 f_i \end{bmatrix} \quad \tau_B = \begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \begin{bmatrix} l(f_4 - f_2) \\ l(f_3 - f_1) \\ \sum_{i=1}^4 \tau_{M_i} \end{bmatrix} \quad (3.11)$$

3.2. Obtención del modelo

Las ecuaciones que gobiernan el sistema se obtienen con el método de Euler-Lagrange, por lo que se empieza obteniendo el Lagrangiano del sistema:

$$\mathcal{L} = E_{cinetica} - E_{potencial} = (E_{translacion} + E_{rotacion}) - E_{potencial} \quad (3.12)$$

Substituyendo cada componente por su expresión:

$$\mathcal{L}(\mathbf{q}, \dot{\mathbf{q}}) = \frac{m}{2} \dot{\boldsymbol{\xi}}^T \dot{\boldsymbol{\xi}} + \frac{1}{2} \boldsymbol{\gamma}^T \mathbf{I} \boldsymbol{\gamma} - mgz \quad (3.13)$$

Es encuentra el vector de fuerzas y momentos:

$$\mathbf{F} = \begin{bmatrix} \mathbf{f} \\ \boldsymbol{\tau}_B \end{bmatrix} = \frac{d}{dt} \left(\frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}} \right) - \frac{\partial \mathcal{L}}{\partial \mathbf{q}} \quad (3.14)$$

$$\text{amb } \mathbf{q} = [x \ y \ z \ \phi \ \theta \ \psi]^T \quad i \quad \dot{\mathbf{q}} = [\dot{x} \ \dot{y} \ \dot{z} \ \dot{\phi} \ \dot{\theta} \ \dot{\psi}]^T.$$

En calcular F es necesario hacer el cambio de variables de $\boldsymbol{\gamma}$ a $\dot{\boldsymbol{\eta}}$ con el cambio de base $\dot{\boldsymbol{\eta}} = [W_{\boldsymbol{\eta}}]^{-1} \boldsymbol{\gamma}$ para poder derivar el Lagrangiano respecto $\dot{\boldsymbol{\eta}}$, que son las variables propias del marco de referencia inercial:

$$\frac{1}{2} \boldsymbol{\gamma}^T \mathbf{I} \boldsymbol{\gamma} = \frac{1}{2} (\mathbf{W}_{\boldsymbol{\eta}} \dot{\boldsymbol{\eta}})^T \mathbf{I} (\mathbf{W}_{\boldsymbol{\eta}} \dot{\boldsymbol{\eta}}) = \frac{1}{2} \dot{\boldsymbol{\eta}}^T (\mathbf{W}_{\boldsymbol{\eta}}^T \mathbf{I} \mathbf{W}_{\boldsymbol{\eta}}) \dot{\boldsymbol{\eta}} = \frac{1}{2} \dot{\boldsymbol{\eta}}^T \mathbf{J} \dot{\boldsymbol{\eta}} \quad (3.15)$$

donde la matriz J queda como

$$\mathbf{J} = \mathbf{W}_{\boldsymbol{\eta}}^T \mathbf{I} \mathbf{W}_{\boldsymbol{\eta}} = \begin{bmatrix} I_{xx} & 0 & I_{xx} S_\theta \\ 0 & I_{yy} C_\phi^2 + I_{zz} S_\phi^2 & (I_{zz} - I_{yy}) C_\phi S_\phi C_\theta \\ I_{xx} S_\theta & (I_{zz} - I_{yy}) C_\phi S_\phi C_\theta & I_{xx} S_\theta^2 + I_{yy} S_\phi^2 C_\theta^2 + I_{zz} C_\phi^2 C_\theta^2 \end{bmatrix} \quad (3.16)$$

y por tanto el Lagrangiano queda:

$$\mathcal{L}(\mathbf{q}, \dot{\mathbf{q}}) = \frac{m}{2} \dot{\boldsymbol{\xi}}^T \dot{\boldsymbol{\xi}} + \frac{1}{2} \dot{\boldsymbol{\eta}}^T \mathbf{J} \dot{\boldsymbol{\eta}} - mgz \quad (3.17)$$

Los componentes lineales y angulares del vector de fuerzas no dependen unos de los otros y por tanto se pueden estudiar por separado obteniendo dos ecuaciones: una para las fuerzas lineales y otro para los momentos. Ésto quiere decir que la fuerza que ejercen los actuadores no depende de las velocidades angulares que se tengan ni tampoco se tendrán aceleraciones angulares diferentes según la altura a la que se encuentre el quadcopter: el objeto girará de la misma manera sea cual sea la posición en la que se encuentre en el espacio.

Entonces, calculando la derivada parcial respecto $\dot{\boldsymbol{\eta}}$ se obtiene que

$$\mathbf{F} = \frac{d}{dt} \left(\frac{m}{2} (1 \cdot \dot{\boldsymbol{\xi}} + \dot{\boldsymbol{\xi}} \cdot 1) + \frac{1}{2} \frac{\partial}{\partial \dot{\boldsymbol{\eta}}} (\dot{\boldsymbol{\eta}}^T \mathbf{J} \dot{\boldsymbol{\eta}}) \right) - \frac{\partial \mathcal{L}}{\partial \mathbf{q}} \quad (3.18)$$

Como que \mathbf{J} es una matriz simétrica, se puede decir que $\frac{\partial}{\partial \dot{\boldsymbol{\eta}}}(\dot{\boldsymbol{\eta}}^T \mathbf{J} \dot{\boldsymbol{\eta}}) = 2 \frac{\partial}{\partial \dot{\boldsymbol{\eta}}}(\dot{\boldsymbol{\eta}}^T \mathbf{J}) \dot{\boldsymbol{\eta}}$.

Demostración: Para probar ésto se verá para el caso $\frac{\partial}{\partial \mathbf{x}}(\mathbf{x}^T \mathbf{A} \mathbf{x}) = 2 \frac{\partial}{\partial \mathbf{x}}(\mathbf{x}^T \mathbf{A}) \mathbf{x}$ con

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad \mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \quad (3.19)$$

Llavors

$$\frac{\partial}{\partial \mathbf{x}}(\mathbf{x}^T \mathbf{A} \mathbf{x}) = \frac{\partial}{\partial \mathbf{x}} \left([x_1 x_2 x_3] \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \right) = \quad (3.20)$$

$$\frac{\partial}{\partial \mathbf{x}}(x_1^2 a_{11} + x_1 x_2 a_{21} + x_1 x_3 a_{31} + x_1 x_2 a_{12} + x_2^2 a_{22} + x_2 x_3 a_{32} + x_1 x_3 a_{13} + x_2 x_3 a_{23} + x_3^2 a_{33}) = \quad (3.21)$$

Com que \mathbf{A} es simétrica $a_{12} = a_{21}$, $a_{13} = a_{31}$ i $a_{23} = a_{32}$, y en hacer la derivada direccional resulta

$$\frac{\partial}{\partial \mathbf{x}}(\mathbf{x}^T \mathbf{A} \mathbf{x}) = \begin{bmatrix} 2x_1 a_{11} + x_2 a_{21} + x_3 a_{31} + x_2 a_{12} + x_3 a_{13} \\ x_1 a_{21} + x_1 a_{12} + 2x_2 a_{22} + x_3 a_{32} + x_3 a_{23} \\ x_1 a_{31} + x_2 a_{32} + x_1 a_{31} + x_2 a_{23} + 2x_3 a_{23} \end{bmatrix} = 2 \cdot \begin{bmatrix} x_1 a_{11} + x_2 a_{12} + x_3 a_{13} \\ x_1 a_{12} + x_2 a_{22} + x_3 a_{23} \\ x_1 a_{13} + x_2 a_{23} + x_3 a_{23} \end{bmatrix} \quad (3.22)$$

Y en evaluar el otro costado de la igualdad se tiene el mismo resultado

$$2 \frac{\partial}{\partial \mathbf{x}}(\mathbf{x}^T \mathbf{A}) \mathbf{x} = 2 \frac{\partial}{\partial \mathbf{x}} \left(\begin{bmatrix} x_1 a_{11} + x_2 a_{12} + x_3 a_{13} \\ x_1 a_{21} + x_2 a_{22} + x_3 a_{23} \\ x_1 a_{13} + x_2 a_{32} + x_3 a_{33} \end{bmatrix}^T \right) \mathbf{x} = \quad (3.23)$$

$$= 2 \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = 2 \cdot \begin{bmatrix} x_1 a_{11} + x_2 a_{12} + x_3 a_{13} \\ x_1 a_{12} + x_2 a_{22} + x_3 a_{23} \\ x_1 a_{13} + x_2 a_{23} + x_3 a_{23} \end{bmatrix} \quad (3.24)$$

■

Como que $2 \frac{\partial}{\partial \dot{\boldsymbol{\eta}}}(\dot{\boldsymbol{\eta}}^T \mathbf{J}) \dot{\boldsymbol{\eta}} = 2 \mathbf{J} \dot{\boldsymbol{\eta}}$ se tiene, aplicando la regla de la cadena en el producto $\mathbf{J} \dot{\boldsymbol{\eta}}$ que:

$$\mathbf{F} = \frac{d}{dt} (m \dot{\boldsymbol{\xi}} + \mathbf{J} \dot{\boldsymbol{\eta}}) - \frac{\partial \mathcal{L}}{\partial \mathbf{q}} = m \ddot{\boldsymbol{\xi}} + \mathbf{J} \ddot{\boldsymbol{\eta}} + \mathbf{J} \dot{\boldsymbol{\eta}} - \left(\frac{1}{2} 2 \frac{\partial}{\partial \dot{\boldsymbol{\eta}}}(\dot{\boldsymbol{\eta}}^T \mathbf{J}) \dot{\boldsymbol{\eta}} - mg \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right) \quad (3.25)$$

Para llegar a este resultado se ha aplicado la derivada direccional a mgz :

$$D_q(mgz) = D_{\boldsymbol{\xi}}(mgz) = mg \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad (3.26)$$

Separando las componentes lineales y angulares en dos ecuaciones:

$$\begin{cases} \mathbf{f} = m\ddot{\xi} + mg \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \mathbf{R}\mathbf{T}_B \\ \boldsymbol{\tau} = \mathbf{J}\ddot{\boldsymbol{\eta}} + \underbrace{\left(\mathbf{J} - \frac{\partial}{\partial \dot{\boldsymbol{\eta}}}(\dot{\boldsymbol{\eta}}^T \mathbf{J}) \right) \dot{\boldsymbol{\eta}}}_{\mathbf{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}})} = \mathbf{J}\ddot{\boldsymbol{\eta}} + \mathbf{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}}) \cdot \dot{\boldsymbol{\eta}} \end{cases} \quad (3.27)$$

Donde $\mathbf{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}})$ es la matriz de Coriolis. Para obtener el sistema de ecuaciones del modelo se deben aislar las aceleraciones y se obtiene:

$$\begin{cases} \ddot{\xi} = \frac{1}{m}\mathbf{R}\mathbf{T}_B - g \cdot \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T \\ \ddot{\boldsymbol{\eta}} = \mathbf{J}^{-1}(\boldsymbol{\tau} - \mathbf{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}}) \cdot \dot{\boldsymbol{\eta}}) \end{cases} \quad (3.28)$$

Reescribiendo estas ecuaciones de la forma $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$, donde \mathbf{u} es el conjunto de fuerzas ejercidas por los motores, se tiene

$$\frac{\partial}{\partial t} \begin{bmatrix} \xi \\ \dot{\xi} \\ \eta \\ \dot{\eta} \end{bmatrix} = \begin{bmatrix} \dot{\xi} \\ \frac{1}{m}\mathbf{R}\mathbf{T}_B - g \cdot \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T \\ \dot{\eta} \\ \mathbf{J}^{-1}(\boldsymbol{\tau} - \mathbf{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}}) \cdot \dot{\boldsymbol{\eta}}) \end{bmatrix} \quad (3.29)$$

Para realizar el control será útil representar el sistema 3.28 en forma de espacio de estados, y por tanto el vector de estados será de la forma $\mathbf{X} = [x \ \dot{x} \ y \ \dot{y} \ z \ \dot{z} \ \phi \ \dot{\phi} \ \theta \ \dot{\theta} \ \psi \ \dot{\psi}]$. El punto de equilibrio de este sistema es aquél que hace que las componentes angulares del vector de estados no varíe y no haya desplazamientos lineales.

Este caso se puede dar para toda posición ξ dada. Ésto se tiene si todos los motores hacen exactamente la misma fuerza y entre todos cuatro la misma al peso del Quadcopter, además de tener una cantidad de movimiento nulo y evitar que el ángulo ψ (yaw) varíe. Se trata de un punto de equilibrio forzado. El punto de equilibrio será entonces:

$$\begin{cases} \mathbf{X}_0 = [x \ 0 \ y \ 0 \ z \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0]^T \\ \mathbf{U}_0 = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} = \frac{m \cdot g}{4} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \end{cases} \quad (3.30)$$

El modelo se expresa en forma lineal como

$$\begin{aligned}\dot{\mathbf{X}} &= [\mathbf{A}] \cdot \mathbf{X} + [\mathbf{K}] + [\mathbf{B}] \cdot \mathbf{U} \\ \mathbf{Y} &= [\mathbf{C}] \cdot \mathbf{X}\end{aligned}\tag{3.31}$$

donde \mathbf{K} es el término que incluye a la constante de la gravedad, separada de la función $\mathbf{f}(\mathbf{x}, \mathbf{u})$. Se calculan las matrices \mathbf{A} y \mathbf{B} con los respectivos jacobianos en el punto de equilibrio:

$$\mathbf{A} = \frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{x}} \Big|_{\mathbf{X}_0, \mathbf{U}_0} \quad \mathbf{B} = \frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{u}} \Big|_{\mathbf{X}_0, \mathbf{U}_0}\tag{3.32}$$

En considerar que las posiciones angulares de ϕ y θ no diferirán significativamente respecto de zero y que el ángulo ψ no es relevante, se trabaja con una matriz de rotación R igual a la identidad

$$\mathbf{R} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\tag{3.33}$$

Se ha mantenido el término independiente de la gravedad a fuera de la función $\mathbf{f}(\mathbf{x}, \mathbf{u})$ para no eliminarlo en la derivada del jacobiano. Las matrices \mathbf{A} y \mathbf{B} quedan como

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1/m & 1/m & 1/m & 1/m \\ 0 & 0 & 0 & 0 \\ 0 & -l/I_{xx} & 0 & l/I_{xx} \\ 0 & 0 & 0 & 0 \\ -l/I_{yy} & 0 & l/I_{yy} & 0 \\ 0 & 0 & 0 & 0 \\ b/I_{zz} & -b/I_{zz} & b/I_{zz} & -b/I_{zz} \end{bmatrix}\tag{3.34}$$

y las matrices \mathbf{K} i \mathbf{C} quedan como

$$\mathbf{K} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ -9,81 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad \mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \quad (3.35)$$

Éstos son los resultados obtenidos del código comentado en el *Anexo A*, en donde se detalla el procedimiento.

Como se desea implantar un control sobre el equilibrio del Quadcopter en vez de la posición, se tendrán las variables pertinentes a la orientación, y por tanto el vector de estados estará formado por $[\dot{\phi} \ \ddot{\phi} \ \theta \ \ddot{\theta}]$.

Queda entonces

$$\begin{bmatrix} \dot{\phi} \\ \ddot{\phi} \\ \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \phi \\ \dot{\phi} \\ \theta \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -l/I_{xx} & 0 & l/I_{xx} \\ 0 & 0 & 0 & 0 \\ -l/I_{yy} & 0 & l/I_{yy} & 0 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} \quad (3.36)$$

$$\mathbf{Y} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \phi \\ \dot{\phi} \\ \theta \\ \dot{\theta} \end{bmatrix}$$

donde el vector \mathbf{K} se ha obviado porque las componentes que se hubieran considerado de éste son todas nulas.

El eliminar estas componentes se hace porque no hay una referencia zero para el ángulo ψ , mientras que en el caso de los ángulos ϕ y θ la referencia siempre será la horizontal. No supone un infranqueable problema a la hora de hacer mover el Quadcopter, pues éste ya se puede desplazar en todo su plano horizontal, sin necesidad de girar en la dirección del *yaw*.

Para abordar el control de este sistema reducido se evalúa su controlabilidad y obser-

vabilidad. Se espera poder llevar el vector de estados de una posición a otra mediante una entrada adecuada, en un tiempo finito. La matriz de controlabilidad $\mathbf{W}_c = [B A B A^2 B \dots]$ es de la forma:

$$\mathbf{W}_c = \left[\begin{array}{cccc|cccc|c} 0 & 0 & 0 & 0 & 0 & -l/I_{xx} & 0 & l/I_{xx} & \dots \\ 0 & -l/I_{xx} & 0 & l/I_{xx} & 0 & 0 & 0 & 0 & \dots \\ 0 & 0 & 0 & 0 & -l/I_{yy} & 0 & l/I_{yy} & 0 & \dots \\ -l/I_{yy} & 0 & l/I_{yy} & 0 & 0 & 0 & 0 & 0 & \dots \end{array} \right] \quad (3.37)$$

Con las 8 primeras columnas es suficiente para ver que el rango de la matriz es máximo, y el resto de componentes de la matriz son nulas. Por lo tanto, no hay subsistema inaccesible des de la entrada y es entonces controlable:

$$\text{rango}(\mathbf{W}_c) = 4 \text{ (max.)} \rightarrow \text{Controlable} \quad (3.38)$$

Para evaluar la observabilidad se calcula la matriz de observabilidad \mathbf{W}_o y como en el caso anterior, se evaluará su rango:

$$\mathbf{W}_o = \left[\begin{array}{c} C \\ CA \\ CA^2 \\ \dots \end{array} \right] = \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ \hline \dots \end{array} \right] \quad (3.39)$$

Otra vez no es necesario evaluar todos los términos de la matriz, ya que con las primeras 4 filas se ve que el rango vuelve a es máximo:

$$\text{rango}(\mathbf{W}_o) = 4 \text{ (max.)} \rightarrow \text{Observable} \quad (3.40)$$

Ésto arroja lo que ya era sabido del sistema: de un Quadcopter se pueden observar sus estados y se puede gobernar (controlar) mediante los actuadores provistos.

3.3. Representación del modelo con Simulink

Obtenidas las ecuaciones del modelo, se quiere poder trabajar con éste para comprobar que la ley de control que se tenga es válida como para controlar el aparato real. Se intenta que el modelo sea lo más preciso posible, por lo que no se tienen en cuenta linearizaciones ni aproximaciones de tipo alguno.

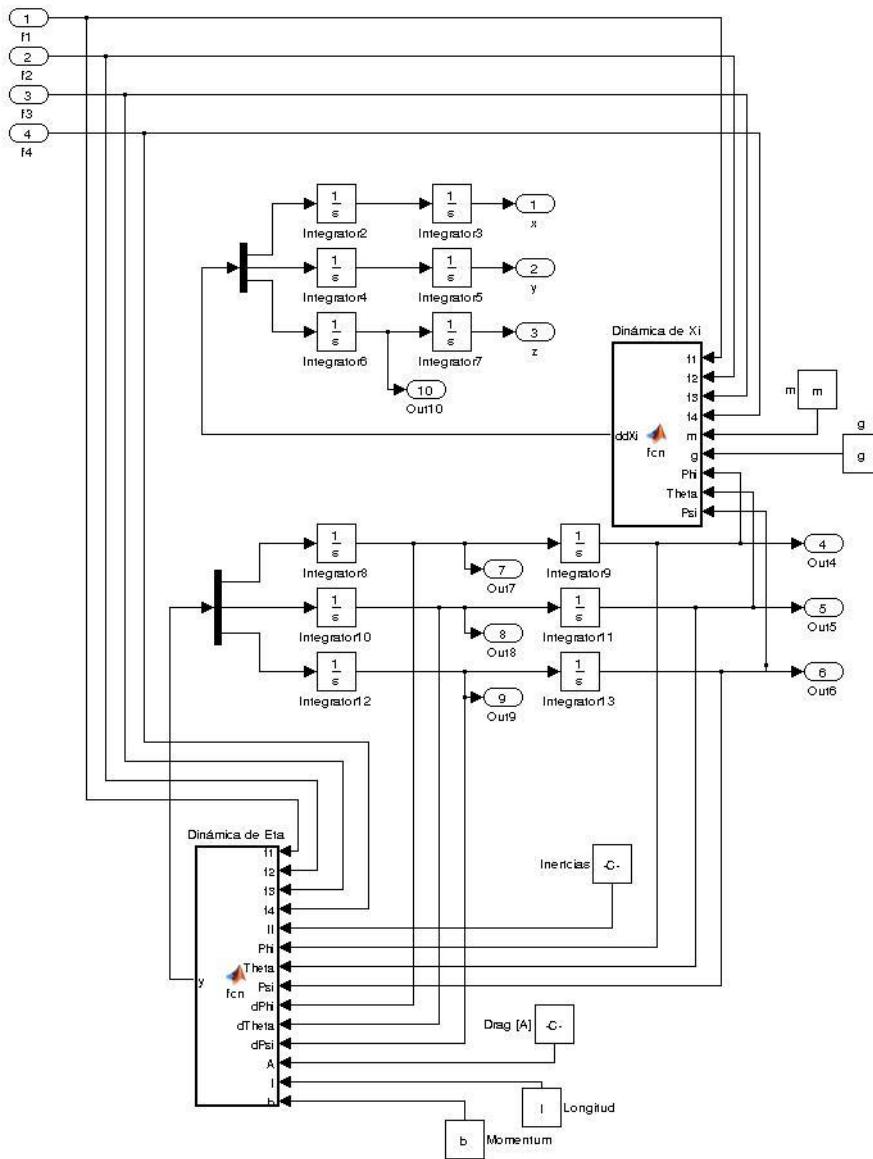


Figura 3.2: Modelo en Simulink de la dinámica del Quadcopter

Como se ha visto en el apartado anterior, no se hayan tenido en cuenta el modelo los respectivos coeficientes de fricción con el aire ya que la determinación de estos parámetros no es fácil y de incluirse podrían aportar un considerable error a los resul.

Tal y como se explica a la hora de caracterizar los motores, no se pudo encontrar la relación existente entre la fuerza y momento ejercido por cada actuador, por lo que en este caso se ha supuesto que varía proporcionalmente con b .

Los dos bloques presentes en el modelo (*Function Matlab*) definen su dinámica. La función *Dinmica de Xi* determina las aceleraciones lineales. El código incluido en ésta es:

```

1 function ddXi = fcn(f1,f2,f3,f4,m,g,Phi,Theta,Psi)
2 %codegen
3
4 R=[[cos(Psi)*cos(Theta) cos(Psi)*sin(Theta)*sin(Phi)-sin(Psi)*
5   cos(Phi) cos(Psi)*sin(Theta)*cos(Phi)+sin(Psi)*sin(Phi)];
6   [sin(Psi)*cos(Theta) sin(Psi)*sin(Theta)*sin(Phi)+cos(Psi)*
7     cos(Phi) sin(Psi)*sin(Theta)*cos(Phi)-cos(Psi)*sin(Phi)];
8   [-sin(Theta) cos(Theta)*sin(Phi) cos(Theta)*cos(Phi)]];
9
10 ddXi = R*T/m+[0; 0; -g];

```

En ella se calcula la matriz de rotación **R** en función de la posición angular(ϕ, θ y ψ) y con el vector de fuerzas en la base móvil (del cuerpo *Body*) se calcula en vector de aceleraciones lineales, es decir, $\ddot{\xi} = [\ddot{\phi} \quad \ddot{\theta} \quad \ddot{\psi}]^T$.

En la función *Dinmica de Eta* se calculan las aceleraciones angulares. Su código corresponde a:

```

1 function y = fcn(f1,f2,f3,f4,II,Phi,Theta,Psi,dPhi,dTheta,dPsi,
2 A,l,b)
3 %codegen
4
5 Ixx=II(1);
6 Iyy=II(2);
7 Izz=II(3);
8
9 TauB=[[ 1*(f4-f2)];
10   [ 1*(f3-f1)];
11   [ b*(f1+f3-f2-f4) ]];
12
13 Weta=[[ 1 0 sin(Theta)];
14   [ 0 cos(Phi) -cos(Theta)*sin(Phi) ];
15   [ 0 sin(Phi) cos(Theta)*cos(Phi) ]];
16 J=transpose(Weta)*[[ Ixx 0 0];[0 Iyy 0];[0 0 Izz]]*Weta;
17

```

```

18 CdE1=(Iyy*dTheta^2*sin(2*Phi))/2 - (Izz*dTheta^2*sin(2*Phi))/2
      - Ixx*dPsi*dTheta*cos(Theta) - (Iyy*dPsi^2*sin(2*Phi)*cos(
      Theta)^2)/2 + (Izz*dPsi^2*sin(2*Phi)*cos(Theta)^2)/2 - Iyy*
      dPsi*dTheta*cos(2*Phi)*cos(Theta) + Izz*dPsi*dTheta*cos(2*
      Phi)*cos(Theta);
19
20 CdE2=-dPhi*(Ixx*dPsi*cos(Theta) + 2*Iyy*dTheta*cos(Phi)*sin(Phi)
      ) - 2*Izz*dTheta*cos(Phi)*sin(Phi) + Iyy*dPsi*cos(Phi)^2*cos(
      Theta) - Izz*dPsi*cos(Phi)^2*cos(Theta) - Iyy*dPsi*cos(
      Theta)*sin(Phi)^2 + Izz*dPsi*cos(Theta)*sin(Phi)^2) - Ixx*
      dPsi^2*cos(Theta)*sin(Theta) + Iyy*dPsi^2*cos(Theta)*sin(Phi)
      )^2*sin(Theta) + Izz*dPsi^2*cos(Phi)^2*cos(Theta)*sin(Theta)
      ;
21
22 CdE3= dPhi*(Ixx*dTheta*cos(Theta) - Iyy*dTheta*cos(Phi)^2*cos(
      Theta) + Izz*dTheta*cos(Phi)^2*cos(Theta) + Iyy*dTheta*cos(
      Theta)*sin(Phi)^2 - Izz*dTheta*cos(Theta)*sin(Phi)^2 + 2*Iyy*
      dPsi*cos(Phi)*cos(Theta)^2*sin(Phi) - 2*Izz*dPsi*cos(Phi)*
      cos(Theta)^2*sin(Phi) + Iyy*dTheta^2*cos(Phi)*sin(Phi)*sin(
      Theta) - Izz*dTheta^2*cos(Phi)*sin(Phi)*sin(Theta) + 2*Ixx*
      dPsi*dTheta*cos(Theta)*sin(Theta) - 2*Izz*dPsi*dTheta*cos(
      Phi)^2*cos(Theta)*sin(Theta) - 2*Iyy*dPsi*dTheta*cos(Theta)*
      sin(Phi)^2*sin(Theta);
23
24 CdEta=[CdE1; CdE2; CdE3];
25
26 y = inv(J)*(TauB-CdEta);

```

En esta función se calcula el vector de momentos (**TauB**), se calcula la actual matriz de cambio de base de velocidad angular a derivadas de ángulos de Euler (\mathbf{W}_η), se actualiza el tensor de inercias según este cambio y se obtiene el producto $\mathbf{C}(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}}) \cdot \ddot{\boldsymbol{\eta}}$. Finalmente devuelve las aceleraciones según se obtienen en la ecuación 3.29.

Los parámetros que se cargan en Matlab necesarios para definir este modelo son

```

1 % Declaracion simbolica de las variables
2
3 syms x dx ddx y dy ddy z dz ddz Phi dPhi ddPhi Theta dTheta
      ddTheta Psi dPsi ddPsi
4 syms f1 f2 f3 f4
5 syms Ixx Iyy Izz
6 syms Ax Ay Az
7 syms k m g l b
8
9 l=0.165                      % Longitud de los brazos (m)
10 Ixx=0.007931                  % Producto de inercia en x (kg*m2)
11 Iyy=0.007664                  % Producto de inercia en y (kg*m2)
12 Izz=0.009741                  % Producto de inercia en z (kg*m2)

```

```
13 m=0.96          % Masa del Quadcopter (kg)
14 b=1.2*10^-7      % Relacion Momento-Fuerza
15 g=9.81           % Aceleracion gravedad (m/s^2)
16
17 Ax=0            % Friccion aire (no usado)
18 Ay=0
19 Az=0
```


Capítulo 4

Diseño del controlador

Con el modelo linealizado del sistema ya se tiene lo que emulará una respuesta verosímil y acorde a la realidad. Para que haga lo que uno espera se decide aplicar un control óptimo mediante un regulador LQR (Linear Quadratic Regulator).

4.1. Regulador Quadrático Lineal

Dado un sistema lineal de tiempo continuo (*LTI*) discreto definido por

$$\begin{aligned}\dot{\mathbf{X}}_{k+1} &= [\mathbf{A}] \cdot \mathbf{X}_k + [\mathbf{B}] \cdot \mathbf{U}_k \\ \mathbf{Y}_{k+1} &= [\mathbf{C}] \cdot \mathbf{X}_k\end{aligned}\tag{4.1}$$

se quiere operar este sistema mediante un coste mínimo según un criterio determinado. Este criterio se caracteriza por tener una función de costes \mathbf{J} que depende tanto de los estados como de las entradas del sistemas [8]

$$\mathbf{J} = \sum_{k=0}^{\infty} (\mathbf{X}_k^T \mathbf{Q} \mathbf{X}_k + \mathbf{U}_k^T \mathbf{R} \mathbf{U}_k + 2 \mathbf{X}_k^T \mathbf{N} \mathbf{U}_k) \tag{4.2}$$

y se definen los pesos \mathbf{Q} , \mathbf{R} y \mathbf{N} como matrices definidas positivas y simétricas, según la penalización que se le quiera dar a los estados (\mathbf{X}), acciones (\mathbf{U}) o el efecto de ambos (\mathbf{N}) respectivamente, tal que se minimice el coste \mathbf{J} . Se obtiene la ley de control $\mathbf{U}_k = -\mathbf{K} \cdot \mathbf{X}_k$ de

$$\mathbf{K} = (\mathbf{B}^T \mathbf{S} \mathbf{B} + \mathbf{R})^{-1} (\mathbf{B}^T \mathbf{S} \mathbf{A} + \mathbf{N}^T) \tag{4.3}$$

donde \mathbf{S} es la solución de la ecuación de Riccati asociada

$$\mathbf{A}^T \mathbf{S} \mathbf{A} - \mathbf{S} - (\mathbf{A}^T \mathbf{S} \mathbf{B} + \mathbf{N})(\mathbf{B}^T \mathbf{S} \mathbf{B} + \mathbf{R})^{-1} (\mathbf{B}^T \mathbf{S} \mathbf{A} + \mathbf{N}^T) + \mathbf{Q} = \mathbf{0} \tag{4.4}$$

Finalmente, de este procedimiento se obtiene, además de las constantes de realimentación \mathbf{K} y la solución \mathbf{S} , los polos del sistema: $\mathbf{P} = \text{eigenvalue}(\mathbf{A} - \mathbf{B} \cdot \mathbf{K})$.

4.2. Obtención de la ley de control

Como se ha visto en la ecuación 3.36, las matrices que representan el sistema son

$$\begin{aligned} \mathbf{A} &= \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} & \mathbf{B} &= \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -l/I_{xx} & 0 & l/I_{xx} \\ 0 & 0 & 0 & 0 \\ -l/I_{yy} & 0 & l/I_{yy} & 0 \end{bmatrix} & (4.5) \\ \mathbf{C} &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{aligned}$$

En el apartado anterior se ha visto que se deben asignar valores a \mathbf{Q} y \mathbf{R} , y como tanto \mathbf{X} como \mathbf{U} son vectores 4×1 , las matrices a asignar serán 4×4 . La matriz \mathbf{Q} se puede escribir como

$$\mathbf{Q} = 1 \cdot \mathbf{C}^T \mathbf{C} \quad (4.6)$$

por lo tanto

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (4.7)$$

Para el caso de \mathbf{R} se consideran toda un mismo peso para todas las entradas f_i y muy superior a los pesos de \mathbf{Q} :

$$\mathbf{R} = 100 \cdot I = \begin{bmatrix} 1000 & 0 & 0 & 0 \\ 0 & 1000 & 0 & 0 \\ 0 & 0 & 1000 & 0 \\ 0 & 0 & 0 & 1000 \end{bmatrix} \quad (4.8)$$

Se hayan \mathbf{K}, \mathbf{S} y \mathbf{E} mediante la toolbox de Matlab para calcular el control óptimo según el regulador LQR[10] en tiempo discreto:

```
[K,S,E] = lqr(A,B,Q,R,Ts)
```

El tiempo de muestreo se tiene que es $T_s = 20 \text{ ms}$ porque las acciones de control tienen una frecuencia de 50 Hz. Teniendo en cuenta que los parámetros del sistema físico obtenidos en el Anexo 4 son $I_{xx} = 7,931 \cdot 10^{-3} \text{ kg} \cdot \text{m}^2$, $I_{yy} = 7,664 \cdot 10^{-3} \text{ kg} \cdot \text{m}^2$ y $l = 16,5 \text{ cm}$ se tiene que las constantes K en la realimentación son:

$$\mathbf{K} = \begin{bmatrix} 0 & 0 & -0,0224 & -0,0322 \\ -0,0224 & -0,0328 & 0 & 0 \\ 0 & 0 & 0,0224 & 0,0322 \\ 0,0224 & 0,0328 & 0 & 0 \end{bmatrix} \quad (4.9)$$

En ser la ley de control $\mathbf{U} = -\mathbf{K} \cdot \mathbf{X}$, el vector de estados $\mathbf{X} = [\phi \ \dot{\phi} \ \theta \ \dot{\theta}]^T$ y $\mathbf{U} = [f_1 \ f_2 \ f_3 \ f_4]$, el ángulo ϕ tendrá efecto en los motores f_1 y f_3 mientras que θ afectará a f_2 y f_4 . La matriz \mathbf{S} , solución de la ecuación de Riccati, queda

$$\mathbf{S} = \begin{bmatrix} 1,4662 & 1,0748 & 0 & 0 \\ 1,0748 & 1,5758 & 0 & 0 \\ 0 & 0 & 1,4415 & 1,0390 \\ 0 & 0 & 1,0390 & 1,4978 \end{bmatrix} \quad (4.10)$$

y los polos del sistema, considerando que es un sistema *LTI*, son

$$\mathbf{E} = \begin{bmatrix} -0,6821 + 0,6821i \\ -0,6821 - 0,6821i \\ -0,6937 + 0,6937i \\ -0,6937 - 0,6937i \end{bmatrix} \quad (4.11)$$

4.3. Aplicación en el modelo

Como primera prueba para la ley de control extraída, se aplica ésta al modelo simulado en Matlab explicado anteriormente. Se encierra el modelo de la dinámica del sistema en un bloque (*Quadcopter*) y se usan sus salidas para realimentar la acción de control de control a través de los motores.

No se considera el control de la altura en el sistema real, pero se implementa un PID para evaluar la respuesta también en la estabilización mientras se alcanza una altura determinada. Ésta disposición de bloques permite implementar otros controles sobre el sistema también usados en el mundo físico tales como PID. El sistema que se tiene en Simulink es

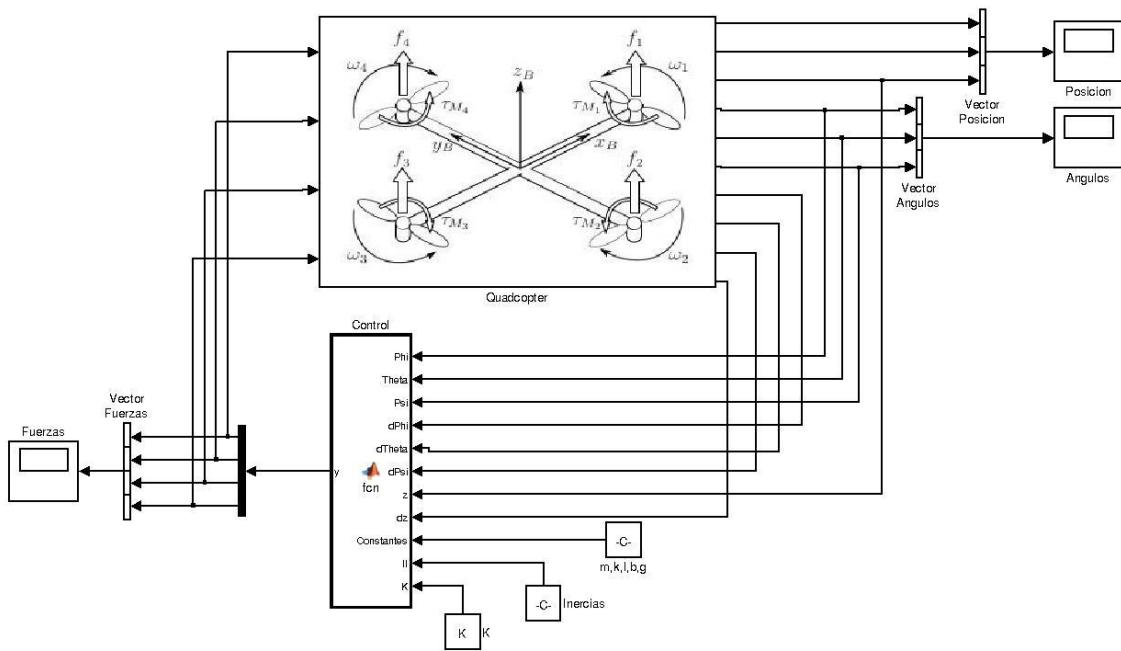


Figura 4.1: Modelo en Simulink del lazo cerrado aplicado en el Quadcopter

Mediante los tres *Scopes* se pueden observar la posición, ángulos y acciones de los motores del Quadcopter. La *Function Matlab* proporciona la acción de control según sea el estado del sistema. A ésta se importan las constantes de realimentación **K** desde el *Workspace* de Matlab y su contenido es

```

1 function y = fcn (Phi , Theta , Psi , dPhi , dTheta , dPsi , z , dz , Constantes
2 %codegen
3
4
5 m=Constantes(1) ;
6 % k=Constantes(2) ;
7 l=Constantes(2) ;
8 b=Constantes(3) ;
9 g=Constantes(4) ;
10
11 Ixx=II(1) ;
12 Iyy=II(2) ;
13 Izz=II(3) ;
14
15 zD=10;
16 dzD=0;
```

```

17
18 KzP=6;
19 KzD=1.75;
20
21 vector=-K*[ Phi ; dPhi ; Theta ; dTheta ]+(g+KzD*(dZD-dz)+KzP*(zD-z)
   )*m;
22
23 if( vector(1)<0)
   vector(1)=0;
25 elseif( vector(1)>10)
   vector(1)=10;
26
27 end
28 if( vector(2)<0)
   vector(2)=0;
30 elseif( vector(2)>10)
   vector(2)=10;
31
32 end
33 if( vector(3)<0)
   vector(3)=0;
35 elseif( vector(3)>10)
   vector(3)=10;
36
37 end
38 if( vector(4)<0)
   vector(4)=0;
40 elseif( vector(4)>10)
   vector(4)=10;
41
42 end
43
44 y=[vector(1) ; vector(2) ; vector(3) ; vector(4) ];

```

Se calcula la acción de control $\mathbf{U} = -\mathbf{K} \cdot \mathbf{X}$ en la línea 21 mediante los estados pero se añade un control para la altura para simular un caso más real. Se saturan las acciones de control para que no puedan ser negativas con los cuatro *if* entre las líneas 23 y 34 porque los motores no pueden cambiar de sentido, ni tampoco superar la fuerza máxima que puedan aportar que se considera que es de 10 N. De esta manera se tiene una representación de la realidad más verosímil.

A modo de ejemplo, para las $\mathbf{K}'s$ calculadas según los parámetros obtenidos del modelo construido:



$$\mathbf{K} = \begin{bmatrix} 0 & 0 & -0,0224 & -0,0322 \\ -0,0224 & -0,0328 & 0 & 0 \\ 0 & 0 & 0,0224 & 0,0322 \\ 0,0224 & 0,0328 & 0 & 0 \end{bmatrix} \quad (4.12)$$

y partiendo de unas condiciones iniciales de $\mathbf{q} = [\xi \quad \eta]^T = [0 \quad 0 \quad 0 \quad 45^\circ \quad 0 \quad 0]$ se tiene como respuesta para el ángulo ϕ :

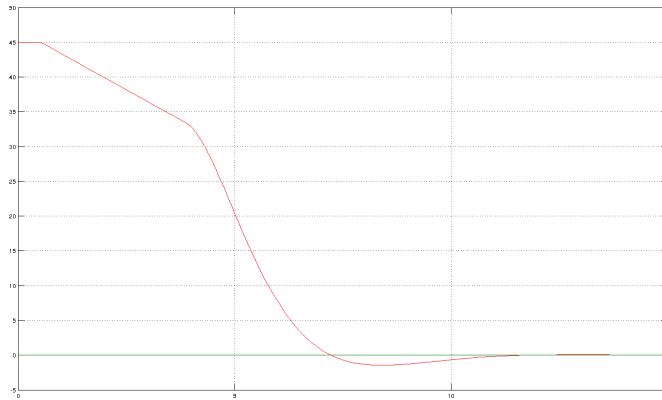


Figura 4.2: Respuesta del ángulo ϕ en la simulación

y la acción de control que se ha aplicado por parte de los motores hasta llegar a la posición de equilibrio es

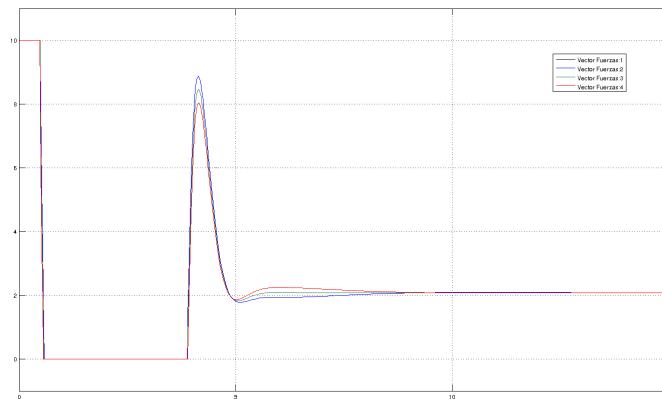


Figura 4.3: Acciones de los motores en la simulación

Se hace notar que las acciones de los actuadores están saturadas con límites tanto inferior como superior y el control implementado no contempla tales no linealidades. Por ello el control no es óptimo y se tarda más aún siendo estable, en alcanzar la consigna. Se detecta también un comportamiento extraño en la evolución del mismo ángulo, que se atribuye a la no linealidad del modelo.

Capítulo 5

Implementación del control

El implementar un control implica hacer capaz al sistema en cuestión de poder recibir la información necesaria para calcular las acciones de control apropiadas y aplicar con éstas al sistema.

Las funciones que debe realizar la Raspberry Pi en este caso son las de captar tanto la consigna como los sensores, así como calcular qué señal PWM se les debe enviar a los cuatro motores que se tienen, a partir del error entre estado y consigna:

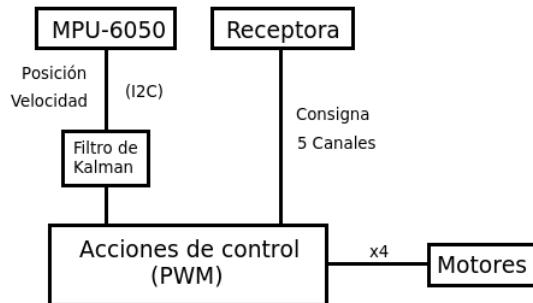


Figura 5.1: Flowchart del código a ejecutar

A los sensores giróscopo y acelerómetro se accede mediante el protocolo de comunicaciones *I2C* con la librería *wiringPiI2C.h* y se hace pasar la información por un filtro de Kalman (*Anexo C*) para quitar el ruido existente en las lecturas. De éstos se obtiene la posición y velocidad.

La consigna se obtiene de la Receptora. Ésta emite 5 señales PWM de iguales características que las que se envían a los motores, es decir, de 50 Hz con un ciclo de trabajo de entre el 5% y 10% y se leen del GPIO (General Purpose Input-Output) las 5 señales mediante interrupciones por cambio de flanco y flanco de bajada creadas con la librería *wiringPi.h*.

Mediante la consigna y la lectura de las estimaciones de los estados $\hat{x}_{k|k}$, se calcula

con la matriz \mathbf{K} obtenida mediante el regulador LQR, la señal a enviar a los motores. Ésta tiene dos componentes: una está influenciada por el error entre los estados $\mathbf{X} = [\phi \quad \dot{\phi} \quad \theta \quad \dot{\theta}]$ y la consigna, es decir, los valores que se desean que tomen. La otra componente viene dada por la persona que controla el aparato en cuestión: el *throttle*, es decir, cuánta fuerza se quiere que apliquen los actuadores (por igual) al sistema, que influirá en el impulso mecánico total del vehículo.

Entonces, si $K_{i,j}$ es cada constante de la matriz \mathbf{K} (4.9) y si f_i es la fuerza que requiere ser ejercida por cada motor i –ésimo en gramos (pondios), éstas se obtienen en el bloque *Acciones de control* de la figura 5.1 como:

$$\begin{aligned} f_1 &= (K_{1,3} \cdot (-\theta + \theta_c) + K_{1,4} \cdot (\dot{\theta})) \cdot \frac{1000}{9,81} + \text{throttle} \\ f_2 &= (K_{2,1} \cdot (-\phi + \phi_c) + K_{2,2} \cdot (\dot{\phi})) \cdot \frac{1000}{9,81} + \text{throttle} \\ f_3 &= (K_{3,3} \cdot (-\theta + \theta_c) + K_{3,4} \cdot (\dot{\theta})) \cdot \frac{1000}{9,81} + \text{throttle} \\ f_4 &= (K_{4,1} \cdot (-\phi + \phi_c) + K_{4,2} \cdot (\dot{\phi})) \cdot \frac{1000}{9,81} + \text{throttle} \end{aligned} \quad (5.1)$$

donde ϕ_c y θ_c son las consignas. Finalmente, para hallar el ciclo de trabajo que deben tener las señales que se envían a los motores se usa la relación encontrada en el *Anexo D* en la caracterización de los motores:

$$f_i = 119 \cdot PWM_i - 613 \quad \rightarrow \quad PWM_i = \frac{f_i + 613}{119} \quad (5.2)$$

Para enviar la información se escribe según la especificación de la librería *pi-blaster* en el archivo FIFO */dev/pi-blaster* (creado en la instalación de la misma) el PWM al que funcionará cada motor.

Capítulo 6

Construcción del Quadcopter

El conjunto de piezas que forman este aparato están conectadas entre sí según la función que realicen. Esencialmente la Raspberry Pi controla los motores según las señales que recibe de la IMU y el Receptor. Todo el conjunto es alimentado por una batería LiPo y se adapta el voltaje de 11.1V a 5V por medio de un Regulador para poder alimentar a la Raspberry. Todos los componentes están sujetos a una estructura (Frame) que también sobre las fuerzas y momentos.

6.1. Descripción de los componentes

Se describe seguidamente cada componente y el criterio de selección que se ha aplicado.

Raspberry Pi

Abreviado com a RPi, es un pequeño ordenador integrado en una sola placa (Single-Board Computer o SBC en inglés) del tamaño de una tarjeta de crédito, es dir, con unas dimensiones de 85.6cm x 53.98cm, desarrollada por la Fundación Raspberry Pi con la intención de promocionar las ciencias computacionales en las escuelas [2].



Figura 6.1: Raspberry Pi a utilizar

Se ha optado por esta opción por su económico precio, la velocidad de procesamiento y bajo consumo. Además, se han querido ampliar los conocimientos de este pequeño

monstruo. En particular se utiliza la segunda revisión del modelo B:

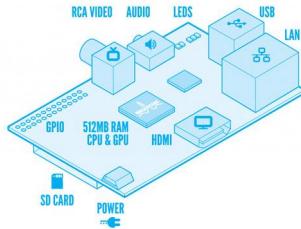


Figura 6.2: Conectores de la Raspberry Pi

Tiene un System-on-Chip (SoC) Broadcom BCM2835 con un ARM1176JZF-S a 700 Mhz, una GPU VideoCore IV y 512 MB de memoria RAM. Dispone de dos puertos USB, una salida mini-jack 3.5mm, salida de audio/vídeo HDMI, una salida RCA y un puerto RJ45 10/100 de Ethernet.

La alimentación se realiza por medio de un mini USB a 5V/700mA, con un consumo de 3.5W. El sistema operativo es un Raspbian, grabado en una tarjeta SD de 4GB. Dispone de un conjunto de pines que permiten comunicación con periféricos de bajo nivel UART, I2C, SPI y 8 pines de propósito general (General Purpose Input Output o GPIO).

GY-521 MPU-6050

Se trata de una Unidad de Medida Inercial (IMU en inglés) que integra en un mismo encapsulado de $4x4x0,9\text{mm}$ un acelerómetro y un giróscopo, ambos de 3 ejes. Dispone de un conversor ADC de 16 bits para cada eje y se comunica mediante un protocolo de comunicación *I2C*. Se ha optado por utilizar este dispositivo por su bajo coste y la fácil implicación con la RPi.



Figura 6.3: IMU MPU-6050

Como características de los sensores: el giróscopo tiene un rango de $\pm 250, \pm 500, \pm 1000, \pm 2000$ grados/segundo, y el acelerómetro de $\pm 2g, \pm 4g, \pm 8g, 16g$. La tensión de alimentación es del rango de $2,375V - 3,46V$ y cabe la posibilidad de utilizar un módulo *DMP* (Digital Motion Processor), pero se ha decidido implementar un filtro de Kalman para leer los datos en crudo (*raw*) de la cola *FIFO* del sensor.

Emisor-Receptor

Con este aparato de se transmite la consigna generada des del transmisor hacia el receptor mediante ondas de radio. El modelo que se utiliza es el *Turnigy5X5ChMini*, porque es fácil de utilizar y además económico. Las especificaciones técnicas más relevantes son:



Figura 6.4: Emisora y Receptor

El transmisor tiene unas dimensiones de $156 \times 152 \times 50\text{mm}$, un peso de 265g y se alimenta a 6V (4 baterías AA). El receptor tiene unas medidas de $33,5 \times 20,5 \times 13\text{mm}$ y se alimenta a 4.8-6V. Dispone de 5 canales de radio, con transmisión segura a 2.4GHz con el método FHSS. Puede configurarse para trabajar con dos modos(mode1-mode2).

La señal que se recibe en cada canal en el receptor es un *PWM* de 50Hz con un ciclo de trabajo entre $1ms$ y $2ms$. Las señales que se reciben son : Aileron, Elevator, Throttle, Rudder y el Channel 5.

Batería LiPo

Para alimentar a todo el conjunto se utiliza una LiPo *Turnigy2200mAh3S1P25C*:



Figura 6.5: Batería LiPo empleada

Por tanto, es capaz de entregar 2.2A durante una hora, y como la capacidad es de 25C, la descarga puede ser de $2,2 \times 25 = 55\text{A}$ con un pic de descarga de 35C, es dir, con un pico de $2,2 \times 35 = 77\text{A}$ durante 10 segundos. Esta batería está formada por tres celdas que proporcionan un voltaje total de unos 11,1V.

El conector de carga es el *JST-XH* y el de descarga el *XT60*. Para tener en cuenta respecto al Quadcopter es interesante saber que pesa 188g y respecto a la estructura (frame) que sus dimensiones son de 105x33x24mm.

Regulador Step-Down

Para adaptar la tensión de 11.1V de la batería LiPo a 5V para alimentar la Raspberry Pi, es necesario un regulador. En este caso se tiene el Interruptor-Modo Máximo BEC LM2576S, que proporciona hasta 3A de corriente.

El esquema del componente [6]:

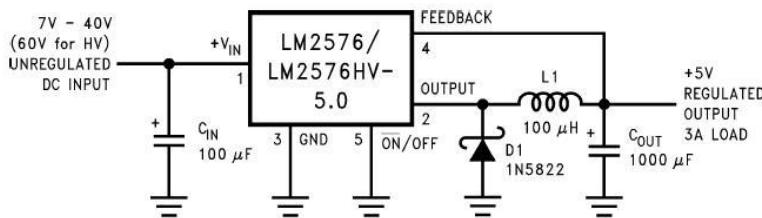


Figura 6.6: Esquema del Regulador Step-Down

Motores Brushless

Los actuadores que se utilizan son motores brushless (BL-DC). En particular el Turnigy 2213 20turn 1050kv 19A Outrunner. Se trata de máquinas síncronas de imanes permanentes, en vez de utilizar electroimanes, donde el campo magnético está uniformemente repartido en el entrehierro. Se conocen también de fuerza electromotriz (FEM) trapezoidal porque a velocidad constante la FEM tiene esta forma. Para controlarlo se utiliza un inversor (inverter), y como la commutación deja de ser mecánica la armadura de la máquina puede ser el estator. Ésto permite llegar a voltajes de uso y velocidades de funcionamiento más altas. Sus especificaciones más relevantes son:

- Kv: 1050rpm/V
- Corriente de trabajo: 6A 16A
- Corriente de pico: 19A
- Peso: 56g
- Dimensiones: 27.6 x 32mm
- Medida de el eje: 3.175mm

El primer parámetro hace referencia a cuántas rpm's funcionará el motor por Voltio aplicado. A este valor se le aplica un porcentaje $\%NLS$ (Porcentaje de Velocidad sin Carga, típicamente de 70 %) para saber cuántas revoluciones por minuto funcionará:

$$Kv \cdot V \cdot \%NLS = 1050 \cdot \frac{rpm}{V} \cdot V \cdot 0,70 = 8085 \text{ rpm} \quad (6.1)$$

Esta velocidad es teórica, y como que no hay un lazo de control sobre ésta, no se tendrá en cuenta. En cambio, se atacará al motor con la señal de PWM que se envía al Variador que le controla. La relación entre la señal de PWM y la fuerza, así como el consumo que se desarrolla se encuentra en los Anexos.

Variador ESC

Para controlar un motor brushless la etapa de potencia a utilizar es un ESC (Electronic Speed Controller) que es un variador que se compone de un inversor trifásico: El

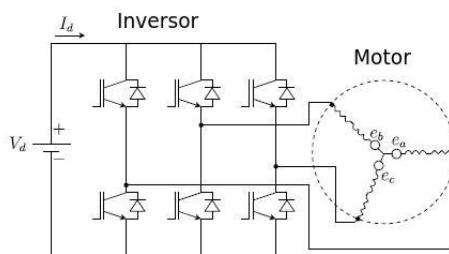


Figura 6.7: Esquema de un Variador como inversor trifásico

modelo a utilizar es el Turnigy AE-20A Brushless ESC. El uso de este tipo se justifica con el amperaje que soporta, en cuanto los motores consumen un pico de hasta 19A, que es menor al máximo de la corriente de este Variador (como límite se puede quemar el componente si se hacen circular 25A durante 10 segundos). Se debe asegurar la compatibilidad con la batería también, ya que soporta tensiones de entrada correspondientes a Baterías LiPo de entre 2 y 4 celdas.

Incluye un circuito BEC (Battery Eliminator Circuit) para poder alimentar también el receptor de 2A de máxima corriente. La máxima velocidad de funcionamiento del motor depende de sus polos: si es de 2 polos será de 210000 rpm, con 6 polos de 70000 rpm y de 12 polos a 35000 rpm.

Tanto el peso de 19g, como las dimensiones de 50x26x12mm son parámetros a tener en cuenta en el montaje, en parte a que se hubican en los brazos del Quadcopter, aumentando la inercia por eje e influyendo considerablemente en el modelo. Antes de usar cada Variador se ha tenido que programar para que éste responda con un mejor rendimiento al caso en cuestión.

Para los 5 parámetros a configurar hay tres opciones, y las marcadas con un punto son las que se han aplicado.

Throttle stick position Menu	Low	Medium	High
1. Brake setting	• off	soft	hard
2. Timing setting	Low	• Medium	High
3. Battery Protection Voltage Threshold	High cut-off threshold	• Medium cut-off threshold	—
4. Plane Mode	• Fixed-wing aircraft	Helicopter mode 1 (with Soft start)	Helicopter mode 2 (Super-Soft+Governor Mode)
5. Throttle response speed setting	• normal	Medium	High

Figura 6.8: Tabla de Set up de un ESC

Marco

Todos los elementos que componen el Quadcopter se sujetan en una estructura que mantiene la integridad y rigidez necesarias para el vuelo. Para el caso tiene el modelo *F330 Wheelbase 330mm*:

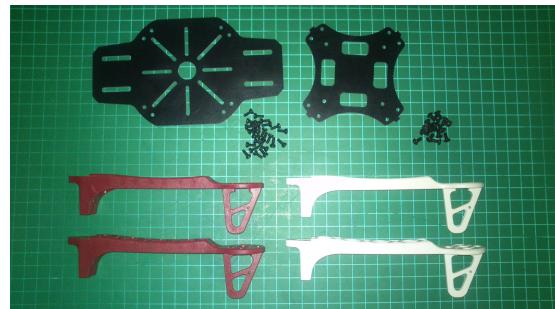


Figura 6.9: Piezas que componen la estructura del Quadcopter

La mayor parte del peso se concentra en el centro de la estructura e interesa que sea lo más ligera posible sin comprometer su integridad, de tal manera que los motores, situados en las puntas de cada brazo, deban compensar el menor peso posible. En este caso, tiene un peso de 190g, y las piezas están hechas de fibra de vidrio las piezas centrales (de color negro en la imagen) y de Nylon los brazos (rojo y blanco en la imagen). La distancia entre dos motores de una diagonal es de 33cm.

Hélices

En girar el motor, las aspas solidarias a su eje utilizan la energía cinética del rotor para proporcionar sustentación propulsando aire en sentido contrario. El tipo de aspa

condiciona el rendimiento que se obtenga, así que se utilizan las que optimizan la fuerza de propulsión respecto al consumo de corriente. El modelo de hélice utilizado es el 8x4



Figura 6.10: Par de hélices 8x4 simétricas

Los parámetros que definen una hélice son su Longitud (X), es decir, diámetro de superficie recorrida por las aspas en pulgadas, y el Pitch (Y), que es la distancia que se avanzaría la hélice en una revolución de moverse en una masa blanda. Se representan en el siguiente gráfico:

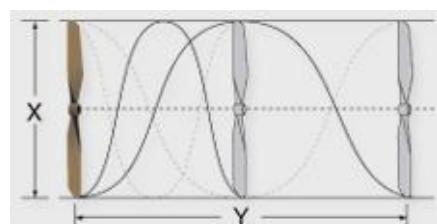


Figura 6.11: Representación de la Longitud y Pitch de una hélice de dos aspas

6.2. Montaje

En la unión de todos los componentes se ha optado por hacer uso de la impresión 3D para crear las piezas que se han necesitado en la unión de las mismas. En el ensamblaje de los componentes, se ha empezado con el Frame:

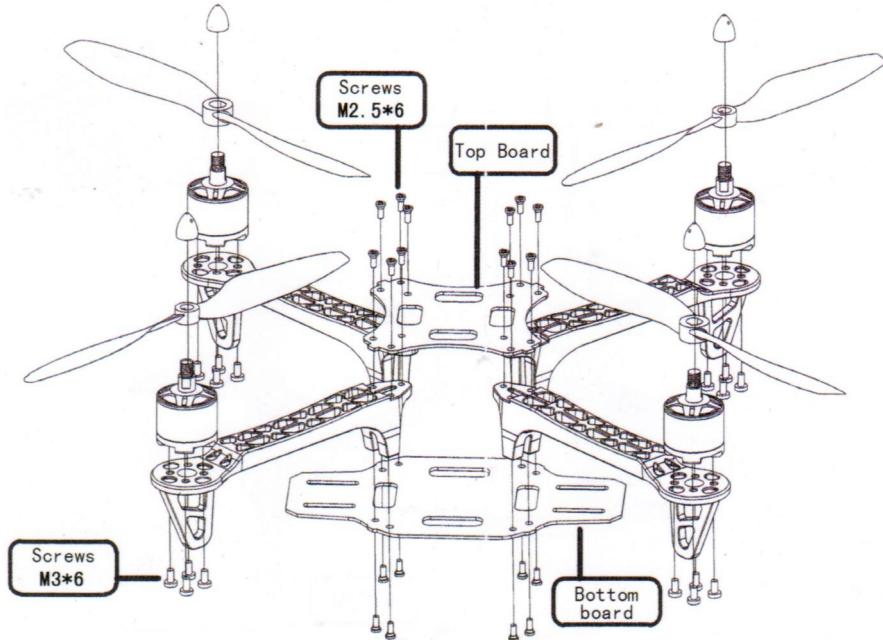


Figura 6.12: Ensamblaje del esqueleto del Quadcopter

Dado que no se tiene una montura para la Raspberry Pi y demás piezas, se diseñan dos piezas a modo de soporte para la electrónica, con el fin también de proteger tales componentes. Se sujetarán a estas piezas la Raspberry Pi y el Receptor:

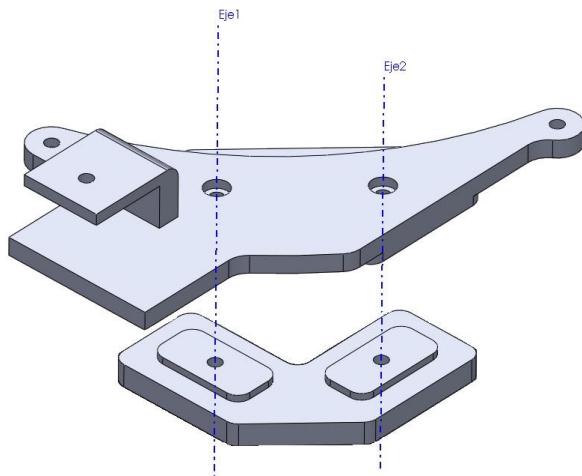


Figura 6.13: Ensamblaje de la montura de la RPi y el Receptor

Atornillados los motores, se sueldan a éstos los variadores, sujetándolos con bridales y teniendo en cuenta que se quiere que el sentido de rotación de éstos sea opuesto en las diagonales:



Figura 6.14: Detalle del cruce de fases en los motores

Para tener un vuelo estable es necesario asegurar la sujeción de la batería al marco. En ser ésta el elemento más pesado de todos de los que componen el aparato, de moverse sería el que más contribuiría en un cambio de los parámetros del sistema, i.e.: los momentos de inercia I_{xx} y I_{yy} . Se montan entonces escuadras que se atornillan a la estructura para fijar su posición.

En despegar se tiene un efecto aerodinámico no considerado en el modelo denominado "*Efecto suelo*". Se trata de las perturbaciones del aire que, en encontrarse con el suelo, afectan al funcionamiento de las hélices. Para evitar este efecto no deseado se decide elevar la altura inicial del Quadcopter con una extensión de las patas mediante piezas imprimidas:

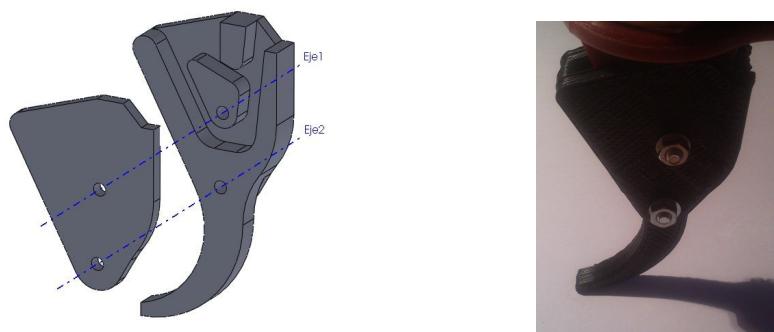
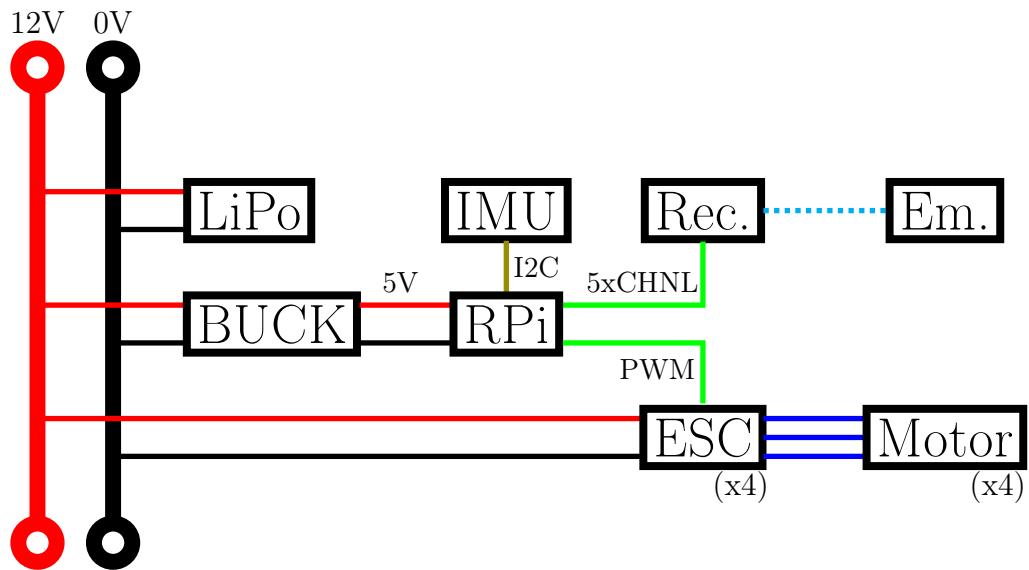


Figura 6.15: Ensamblaje y detalle de una de las patas.

Y finalmente se fija el Regulador Step-Down a la estructura mediante bridales. En lo referente al conexionado, el esquema que se sigue es



Se han introducido conectores en las tomas positivas de cada ESC para poder programar cada uno por separado de ser necesario.

A cada motor se le atornillan la hélice adecuada según su sentido de giro, que viene dado por el orden en que se han soldado las fases del actuador, según 6.14.

El resultado final es:

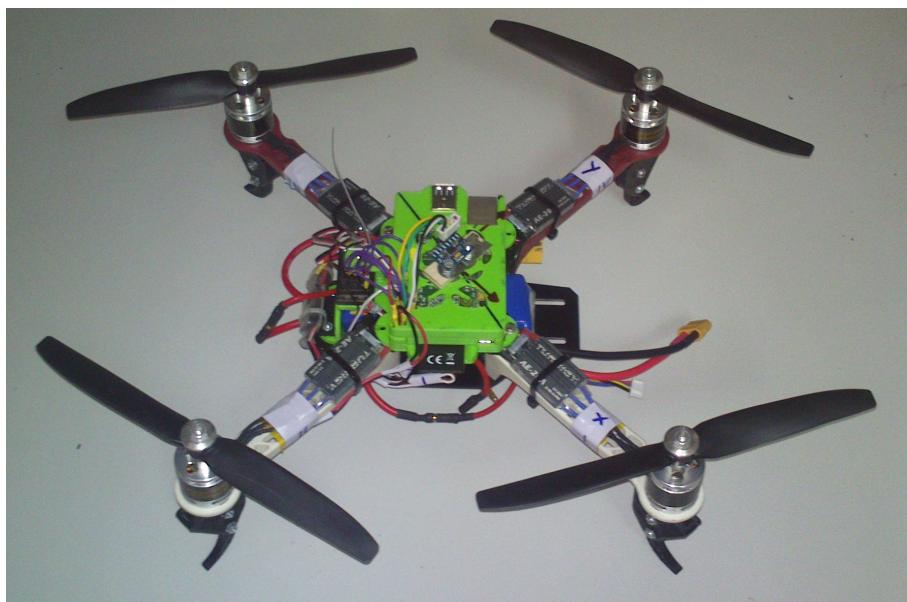


Figura 6.16: Vehículo finalizado

Capítulo 7

Análisis económico

Como en todo proyecto es importante conocer y justificar el gasto económico implicado. En no ser éste una excepción, se desglosan los costes asociados, tanto materiales como los de mano de obra.

Referente a los costes materiales, se incluye el de todos los elementos que componen el Quadcopter, así como otros igualmente necesarios como el cargador de las baterías. Se considera que el trabajo se ha realizado por un estudiante en desarrollo de su Proyecto Final de Carrera. El coste que se le supone al tiempo del digno obrero es de 6€/hora.

Componente	Precio(€ /u)	Cantidad	Coste(€)
Componente	Precio unitario	Cantidad	Coste(€)
Conector XT60 macho	0.36	1	0.36
Cable 12AWG (1 metro) Turnigy	2.2	1	2.20
Emisor-Receptor Mini Turnigy 5X 5Ch	24.99	1	24.99
Batería Turnigy LiPo 2200mAh 3S 25C	10.57	2	21.14
Motor Turnigy 2213 20turn 1050kv 19A	14.53	4	58.12
Variador Turnigy AE-20A Brushless	10.48	4	41.92
Pack 4 hélices 8x4	2.28	1	2.28
Pack 4 hélices 8x4R	2.28	1	2.28
Cargador LiPo iMAX B6	16.95	1	16.95
Cargador lipo	16.95	1	16.95
Total del Coste material		170.24 €	

Recursos Humanos	Coste(€/hora)	Horas	Coste(€)
Estudiante de PFC	6	540	3240
Total R.R. H.H.			3240 €
TOTAL			3410.24 €

Capítulo 8

Impacto ambiental

En el desarrollo del presente proyecto se ha construido un aparato compuesto por partes de diferentes materiales, de entre los que destacan el ABS, Nylon y fibra de vidrio. No se estudiará el impacto medioambiental del complejo proceso de producción, uso y reciclaje de los mismos, sino un comentario del impacto que éstos generan en general.

Por parte de la fibra de vidrio, se considera un material de baja toxicidad debido a sus diferencias con los cancerígenos asbestos. En poder fabricarse las fibras a partir de vidrio reciclado, el impacto de este material al medio se considera bajo.

Por otro lado, el Nylon es una poliamida de gran resistencia a la corrosión ambiental. Por este motivo éste es uno de los plásticos que más han contribuido en el incremento de los niveles de contaminación. Los tiempos de degradación del Nylon son muy largos y sus residuos muy perjudiciales para el medio ambiente. Por ello es de gran importancia tener una consolidada etapa de reciclado de éste material, también para otros polímeros parecidos a éste.

Respecto al ABS, se trata de un plástico de ingeniería que de la misma manera que el Nylon por su resistencia térmica y química, debe reciclararse y tratarse los residuos.

Como sistema de alimentación para el quadcopter se usan baterías del tipo LiPo, conocidas también como de ion de litio. Éstas son muy contaminantes de no tratarse adecuadamente en la fase de reciclado y requieren estar completamente descargadas antes de ser entregadas a un punto de recogida.

Capítulo 9

Conclusiones

9.1. Futuras aplicaciones

Agradecimientos

Un trabajo de esta índole no podría haber sido posible sin la ayuda y apoyo de quienes, sin o con la aportación directa en el proyecto, lo han hecho realmente posible.

A la familia, la verdadera *Alma Mater*, por su apoyo, soporte y paciencia en la convivencia durante mis peores momentos, que no han sido pocos. A Manel Velasco, tutor y mentor de este pobre *umpa lumpy* de la ciencia, por la guía, consejos y espacio de trabajo tan amablemente proporcionados en esta empresa. A Joan Guasch y Diego Muñoz, porque es importante hacer ameno el trabajo en los días poco fructíferos, que tan pesados son.

Bibliografía

- [1] Teppo Luukkonen (August 22,2011). *Modelling and control of quadcopter*
- [2] Wikipedia de la Raspberry Pi: http://en.wikipedia.org/wiki/Raspberry_Pi
- [3] Acelerómetro y giroscopio MPU-6050 para Arduino: <http://playground.arduino.cc/Main/MPU-6050#.UzhsVCK9jb4>
- [4] MPU-6050. Especificación del producto: <http://www.invensense.com/mems/gyro/documents/PS-MPU-6000A-00v3.4.pdf>
- [5] MPU-6050. Mapa de Registros y descripciones: <http://www.invensense.com/mems/gyro/documents/RM-MPU-6000A-00v4.2.pdf>
- [6] LM2576S Datasheet: <http://www.ti.com/lit/ds/symlink/lm2576.pdf>
- [7] Librería wiringPi: <http://wiringpi.com/>
- [8] Artículo sobre LQR de Wikipedia: http://en.wikipedia.org/wiki/Linear-quadratic_regulator
- [9] Documentación de la función LQR de Matlab: <http://www.mathworks.es/help/control/ref/lqr.html;jsessionid=8f737c267ee85a89cc2544a09a44>
- [10] Toolbox de Matlab del LQR : <http://www.mathworks.es/es/help/control/ref/lqr.html>
- [11] Ecuaciones de Lagrange en Matlab: <http://www.mathworks.com/matlabcentral/fileexchange/23037-lagranges-equations/content/Lagrange.m>
- [12] Github de la librería pi-blaster: <https://github.com/sarfata/pi-blaster/>

ANEXOS

Anexo A: Obtención del vector de fuerzas

El archivo que se ejecuta para obtener el modelo parte de los parámetros necesarios para caracterizar el sistema como variables simbólicas. A partir de ellas se extrae el modelo. Se comenta por partes el código Matlab del que se extrae toda la información.

Se inicializan las variables simbólicas implicadas en el Lagrangiano y las necesarias en el modelo:

```
1 clear all
2 clc
3
4 %%Para usar la funcion Laplace en el archivo Laplace.m se
5 % deben declarar las variables de esta manera:
6 syms x dx ddx y dy ddy z dz ddz Phi dPhi ddPhi Theta dTheta
7 ddTheta Psi dPsi ddPsi
8 syms f1 f2 f3 f4 %fuerza motores
9 syms Ixx Iyy Izz %Inercias
10 syms Ax Ay Az %No considerados
11 syms k m g l b
12 l=0.165 %longitud de los brazos (m)
13 Ixx=0.007931 %Momento de inercia del eje x (kg*m2)
14 Iyy=0.007664 %Momento de Inercia del eje y
15 Izz=0.009741 %Momento de Inercia del eje z
16 m=0.96 %Peso del Quadcopter (kg)
17 g=9.81 %gravedad (m/s2)
```

Se agrupan adecuadamente las variable en vectores, tal y como se especifica en el capítulo 3:

```
18 % El vector de variables que se usaran
19
20 v=[x dx ddx y dy ddy z dz ddz Phi dPhi ddPhi Theta dTheta
21 ddTheta Psi dPsi ddPsi]
22 Xi=[x; y; z]
```

```

23 dXi=[dx; dy; dz]
24
25 Eta=[Phi; Theta; Psi]
26 dEta=[dPhi; dTheta; dPsi]
27
28 Q=[Xi; Eta]
29 dQ=[dXi; dEta]
30
31 II=[[Ixx 0 0];
32      [0 Iyy 0];
33      [0 0 Izz ]]]
34
35 A=[[Ax 0 0];
36      [0 Ay 0];
37      [0 0 Az]]]
38
39 T=f1+f2+f3+f4           % Fuerza total (Total Thrust)
40 TB=[0; 0; T]             % Thrust en la referencia del Quadcopter
41
42 TauB=[l*(f4-f2);
43      l*(f3-f1);
44      b*(f1-f2+f3-f4)]

```

Se declaran las matrices de rotación, transformación de velocidad angular a derivada de ángulos de Euler, matriz de inercia y finalmente el Lagrangiano del sistema:

```

45 \begin{lstlisting}[language=Matlab, firstnumber=45]
46 R=[[cos(Psi)*cos(Theta) cos(Psi)*sin(Theta)*sin(Phi)-sin(Psi)*
47     cos(Phi) cos(Psi)*sin(Theta)*cos(Phi)+sin(Psi)*sin(Phi)];
48     [sin(Psi)*cos(Theta) sin(Psi)*sin(Theta)*sin(Phi)+cos(Psi)*
49     cos(Phi) sin(Psi)*sin(Theta)*cos(Phi)-cos(Psi)*sin(Phi)
50     ];
51     [-sin(Theta) cos(Theta)*sin(Phi) cos(Theta)*cos(Phi)]]
52
53 Weta=[[1 0 -sin(Theta)];
54      [0 cos(Phi) cos(Theta)*sin(Phi)];
55      [0 -sin(Phi) cos(Theta)*cos(Phi)]]
56 J=transpose(Weta)*II*Weta
57
58 L=m/2*transpose(dXi)*dXi+(1/2)*transpose(dEta)*transpose(Weta)*
59     II*Weta*dEta-m*g*z

```

Aquí se obtiene con el Lagrangiano el vector de fuerzas \mathbf{F} según la ecuación

$$\mathbf{F} = \frac{d}{dt} \left(\frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}} \right) - \frac{\partial \mathcal{L}}{\partial \mathbf{q}} \quad (9.1)$$

a partir de la función implementada en Lagrange.m implementada [11]

```
56 \begin{lstlisting}[language=Matlab , firstnumber=57]
57 F=Lagrange(L,v)
```

Para tener una expresión analítica del producto $C(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}}) \cdot \ddot{\boldsymbol{\eta}}$ se requiere haber derivado a mano hasta la ecuación 3.27 para ver que

$$\tau = J\ddot{\boldsymbol{\eta}} + C(\boldsymbol{\eta}, \dot{\boldsymbol{\eta}}) \cdot \ddot{\boldsymbol{\eta}} \quad (9.2)$$

y saber cómo aislar el producto

```
57 \begin{lstlisting}[language=Matlab , firstnumber=58]
58 %%Derivando a mano se obtiene que TauB=J*ddEta+CdEta
59 %%Y como TauB=[F(4); F(5); F(6)], se tiene que el termino de
60 %%Coriolis es:
61 CdEta=[F(4); F(5); F(6)]-J*[ddPhi; ddTheta; ddPsi]
62
63 %%Hasta se tiene el producto Coriolis(Eta,dEta)*dEta. Pero
64 %%como matlab
65 %%no elimina las aceleraciones se hace a mano, y queda:
66 CdE1=(Iyy*dTheta^2*sin(2*Phi))/2 - (Izz*dTheta^2*sin(2*Phi))/2
   - Ixx*dPsi*dTheta*cos(Theta) - (Iyy*dPsi^2*sin(2*Phi)*cos(
   Theta)^2)/2 + (Izz*dPsi^2*sin(2*Phi)*cos(Theta)^2)/2 - Iyy*
   dPsi*dTheta*cos(2*Phi)*cos(Theta) + Izz*dPsi*dTheta*cos(2*
   Phi)*cos(Theta);
67
68 CdE2=-dPhi*(Ixx*dPsi*cos(Theta) + 2*Iyy*dTheta*cos(Phi)*sin(Phi)
   - 2*Izz*dTheta*cos(Phi)*sin(Phi) + Iyy*dPsi*cos(Phi)^2*cos(
   Theta) - Izz*dPsi*cos(Phi)^2*cos(Theta) - Iyy*dPsi*cos(
   Theta)*sin(Phi)^2 + Izz*dPsi*cos(Theta)*sin(Phi)^2) - Ixx*
   dPsi^2*cos(Theta)*sin(Theta) + Iyy*dPsi^2*cos(Theta)*sin(Phi)
   )^2*sin(Theta)+ Izz*dPsi^2*cos(Phi)^2*cos(Theta)*sin(Theta);
69
70 CdE3= dPhi*(Ixx*dTheta*cos(Theta) - Iyy*dTheta*cos(Phi)^2*cos(
   Theta) + Izz*dTheta*cos(Phi)^2*cos(Theta) + Iyy*dTheta*cos(
   Theta)*sin(Phi)^2 - Izz*dTheta*cos(Theta)*sin(Phi)^2 + 2*Iyy*
   dPsi*cos(Phi)*cos(Theta)^2*sin(Phi) - 2*Izz*dPsi*cos(Phi)*
   cos(Theta)^2*sin(Phi)) + Iyy*dTheta^2*cos(Phi)*sin(Phi)*sin(
   Theta) - Izz*dTheta^2*cos(Phi)*sin(Phi)*sin(Theta) + 2*Ixx*
   dPsi*dTheta*cos(Theta)*sin(Theta) - 2*Izz*dPsi*dTheta*cos(
   Phi)^2*cos(Theta)*sin(Theta) - 2*Iyy*dPsi*dTheta*cos(Theta)*
   sin(Phi)^2*sin(Theta);
71
72 CdEta=[CdE1; CdE2; CdE3];
```

Si se quiere econtrar los puntos de equilibrio del sistema planteado en 3.28 se parte del mismo sistema de ecuaciones y se impone que tanto las velocidades y aceleraciones sean

cero. Para facilitar al sistema a resolver el problema también se supone que se está en condiciones próximas al punto de equilibrio, y por tanto, si $\alpha \approx 0 \rightarrow \sin(\alpha) = 0$ y si $\alpha \approx 1 \rightarrow \cos(\alpha) = 1$.

```

68 \begin{lstlisting}[language=Matlab , firstnumber=69]
69 dPhi=0
70 dTheta=0
71 dPsi=0
72
73 R=eye(3)
74
75 sys1=(R*TB-[0; 0; m*g])/m
76 sys2=inv(J)*(TauB-CdEta)
77
78 eq1=sym(sys1(1))
79 eq2=sym(sys1(2))
80 eq3=sym(sys1(3))
81
82 eq4=sym(sys2(1))
83 eq5=sym(sys2(2))
84 eq6=sym(sys2(3))
85
86 syms dPhi dTheta dPsi      % Requerido para substituir con subs()
87
88 seq1=sym(subs(eq1,{dPhi,dTheta,dPsi,sin(Phi),sin(Theta),sin(Psi)
89     ),cos(Phi),cos(Theta),cos(Psi)},{0,0,0,0,0,1,1,1}))
90 seq2=sym(subs(eq2,{dPhi,dTheta,dPsi,sin(Phi),sin(Theta),sin(Psi)
91     ),cos(Phi),cos(Theta),cos(Psi)},{0,0,0,0,0,1,1,1}))
92 seq3=sym(subs(eq3,{dPhi,dTheta,dPsi,sin(Phi),sin(Theta),sin(Psi)
93     ),cos(Phi),cos(Theta),cos(Psi)},{0,0,0,0,0,1,1,1}))
94 seq4=sym(subs(eq4,{dPhi,dTheta,dPsi,sin(Phi),sin(Theta),sin(Psi)
95     ),cos(Phi),cos(Theta),cos(Psi)},{0,0,0,0,0,1,1,1}))
96 seq5=sym(subs(eq5,{dPhi,dTheta,dPsi,sin(Phi),sin(Theta),sin(Psi)
97     ),cos(Phi),cos(Theta),cos(Psi)},{0,0,0,0,0,1,1,1}))
98 seq6=sym(subs(eq6,{dPhi,dTheta,dPsi,sin(Phi),sin(Theta),sin(Psi)
99     ),cos(Phi),cos(Theta),cos(Psi)},{0,0,0,0,0,1,1,1}))
100 sol=solve(seq3,seq4,seq5,seq6,'f1','f2','f3','f4')
sol.f1
sol.f2
sol.f3
sol.f4

```

Para linealizar el sistema se deben calcular las matrices **A** y **B** como en 3.32. Para ello se definen los vectores de derivadas, estados y fuerzas, se calculan los jacobianos y se substituye por el punto de equilibrio

```

100 \begin{lstlisting}[language=Matlab , firstnumber=101]
101 fx=[dx;
102     eq1 ;
103     dy ;
104     eq2 ;
105     dz ;
106     eq3 ;
107     dPhi ;
108     eq4 ;
109     dTheta ;
110     eq5 ;
111     dPsi ;
112     eq6 ]
113
114 states=[x;
115     dx;
116     y;
117     dy;
118     z;
119     dz;
120     Phi;
121     dPhi;
122     Theta;
123     dTheta;
124     Psi;
125     dPsi ];
126
127 forces=[f1 ;
128     f2 ;
129     f3 ;
130     f4 ];
131
132 A=jacobian (fx , states )
133 Asub=sym( subs(A,{ Phi , Theta , Psi , dPhi , dTheta , dPsi , f1 , f2 , f3 , f4
134     } ,{ 0 ,0 ,0 ,0 ,0 ,0 ,m*g/4,m*g/4,m*g/4 ,m*g/4 }) )
135 B=jacobian (fx , forces )
136 Bsub=sym( subs(B,{ Phi , Theta , Psi , dPhi , dTheta , dPsi , f1 , f2 , f3 , f4
137     } ,{ 0 ,0 ,0 ,0 ,0 ,0 ,m*g/4,m*g/4,m*g/4 ,m*g/4 }) )

```

Como se busca un sistema como el que se ha obtenido en 3.36, se repite todo el proceso incluyendo únicamente las variables respectivas a los ángulos ϕ y θ y sus derivadas:

```

136
137 \begin{lstlisting}[language=Matlab , firstnumber=137]
138 fx=[dPhi ;
139     eq4 ;
140     dTheta ;
141     eq5 ];

```

```

142 states=[Phi ;
143     dPhi ;
144     Theta ;
145     dTheta ];
146
147 disA=jacobian (fx , states );
148 Ared=sym( subs (disA ,{ Phi ,Theta ,dPhi ,dTheta ,f1 ,f2 ,f3 ,f4
149     } ,{0 ,0 ,0 ,0 ,m*g /4 ,m*g /4 ,m*g /4 ,m*g /4 } ) )
150
151 disB=jacobian (fx , forces );
152 Bred=sym( subs (disB ,{ Phi ,Theta ,dPhi ,dTheta ,f1 ,f2 ,f3 ,f4
153     } ,{0 ,0 ,0 ,0 ,m*g /4 ,m*g /4 ,m*g /4 ,m*g /4 } ) )

```

Para calcular las $\mathbf{K}'s$ del control se procede calculando los parámetros \mathbf{Q} y \mathbf{R} necesarios para especificar el control por LQR

```

151 \begin{lstlisting}[language=Matlab , firstnumber=152]
152 Cred=[[1 0 0 0];[0 0 1 0]];
153 Dred=[[0 0 0 0];[0 0 0 0]];
154
155 Q=1*transpose (Cred)*Cred
156 R=1000*eye(4)
157
158 [K,S,e]=lqr ( double (Ared) ,double (Bred) ,Q,R)

```

Anexo B:Preparación de la Raspberry Pi

Para poner a punto la RPi se instala un Raspbian y se configura para que es pueda comunicar con la placa MPU-6050.

Instalación Raspbian

Habiendo introducido la tarjeta SD en un lector adecuado, se detecta en una terminal mediante la orden `df -h`. Suponiendo que haya estado gravada anteriormente:

```
/dev/sdb1      56M   19M   38M  33% /media/boot
/dev/sdb2     1,8G  1,4G  266M  85% /media/9c7e2035-df9b-490b-977b-d60f2170889d
```

Se deben desmontar las dos particiones, tanto la `/dev/sdb1` como la `/dev/sdb2`:

```
umount /dev/sdb1
umount /dev/sdb2
```

Descargada la imagen a instalar, por ejemplo en el Escritorio, se procede con:

```
sudo dd bs=4M if=/Escritorio/2013-07-26-wheezy-raspbian.img of=/dev/sdb
```

Pasados unos minutos ya se tiene la SD grabada con el sistema operativo Raspbian.

Configuración de la Raspberry

Para poder comunicar la RPi con la IMU MPU-6050 por I2C es necesario configurar el sistema. Primero es necesario instalar los drivers mas relevantes. Del archivo

```
sudo nano /etc/modules
```

se añaden las siguientes dos líneas al final del archivo:

```
i2c-bcm2708
i2c-dev
```

En el archivo blacklist:

```
sudo vi /etc/modprobe.d/raspi-blacklist.conf
```

las siguientes dos líneas deben empezar con un signo # (de comentario):

```
#blacklist spi-bcm2708
#blacklist i2c-bcm2708
```

Para conectar el sensor, se sigue el conexionado:

- Pin 1-3.3V se conecta a VCC.
- Pin 3-SDA se conecta a SDA
- Pin 5-SCL se conecta a SCL
- Pin 6-Ground se conecta a GND

Es necesario instalar el paquete `i2c-tools`:

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

Para ver el sensor se escribe:

```
sudo i2cdetect -y 1
```

El resultado es el dispositivo que corresponde al MPU-6050:



```
pi@raspberrypi ~ $ sudo i2cdetect -y 1
      0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00: --  --  --  --  --  --  --  --  --  --  --  --  --  --
10: --  --  --  --  --  --  --  --  --  --  --  --  --  --
20: --  --  --  --  --  --  --  --  --  --  --  --  --  --
30: --  --  --  --  --  --  --  --  --  --  --  --  --  --
40: --  --  --  --  --  --  --  --  --  --  --  --  --  --
50: --  --  --  --  --  --  --  --  --  --  --  --  --  --
60: --  --  --  --  --  68  --  --  --  --  --  --  --
70: --  --  --  --  --  --  --  --  --  --  --  --  --  --
```

Figura 9.1: Detección del sensor MPU-6050

Librería wiringPi

Se utiliza esta librería [7] como a interfaz del GPIO (General Purpose Input Output). Permite, por tanto, leer las consignas y sensores y generar las acciones de control. La instalación y descripción de la librería esta explicado en la página web incluida en la bibliografía. Ésta también incluye la librería *wiringPiI2C.h*, usada para la comunicación por I2C con los sensores.

Librería pi-blaster

Es necesaria esta librería [12] para enviar las señales de control a sus respectivos motores, ya que permite usar cada pin del GPIO como un PWM. Se instala y usa según se especifica en el repositorio.

Anexo C: Filtro de Kalman

Las lecturas del sensor están sometidas a ruido y por lo tanto no son tan precisas como convendría. Para evitar eso se usa un filtro de Kalman, que es un algoritmo recursivo para corregir los valores obtenidos a partir de la estimación de variables no conocidas, obteniendo valores más precisos que los que se leen.

Algoritmo del Filtro de Kalman

Sobre un sistema dinámico *LTI* representado en espacio de estados tal como

$$\begin{aligned}\mathbf{x}_k &= \mathbf{A}_{k-1} \cdot \mathbf{x}_{k-1} + \mathbf{B}_{k-1} \cdot \mathbf{u}_{k-1} + \mathbf{w}_{k-1} \\ \mathbf{z}_k &= \mathbf{H}_k \cdot \mathbf{x}_k + \mathbf{v}_k\end{aligned}\quad (9.3)$$

donde \mathbf{w}_{k-1} y \mathbf{v}_k son el ruido blanco de promedio cero y varianza \mathbf{Q}_{k-1} y \mathbf{R}_k en los instantes $k-1$ y k respectivamente, este algoritmo identifica los estados \mathbf{x}_k en dos fases: la **predicción o estimación** y la **corrección**.

Predicción

En la primera fase se estiman los estados del sistema. Una estimación se puede hacer respecto a un estado anterior y las estimaciones previas, en tal caso se tendrá un *estado previo*:

$$\hat{\mathbf{x}}_{k-1|k-1} \quad (9.4)$$

También se puede tener un *estado a priori*, es decir, estimar el estado actual del sistema basándose en el estado actual y las estimaciones previas:

$$\hat{\mathbf{x}}_{k|k-1} \quad (9.5)$$

o bien se puede obtener el *estado a posteriori*, que es la predicción del estado según el estado y las estimaciones actuales:

$$\hat{\mathbf{x}}_{k|k} \quad (9.6)$$

Se procede a hacer una estimación *a priori* del estado mediante

$$\hat{\mathbf{x}}_{k|k-1} = \Phi_k \cdot \mathbf{x}_{k-1|k-1} \quad (9.7)$$

donde Φ_k es la matriz de transición de estados. Ésta puede depender también de otras variables, tales como \mathbf{u}_k como se verá posteriormente y se calcula la covarianza del error asociada a la estimación *a priori* en función de la varianza \mathbf{Q}_k del estado como

$$\mathbf{P}_{k|k-1} = \Phi_k \cdot \mathbf{P}_{k-1|k-1} \cdot \Phi_k^T + \mathbf{Q}_k \quad (9.8)$$

En suponer que un estado está sometido a ruido blanco se puede expresar su distribución de probabilidad con una distribución normal con media igual a la estimación realizada $\mu = \hat{\mathbf{x}}_{k|k-1}$ y varianza $\sigma^2 = P_{k|k-1}$:

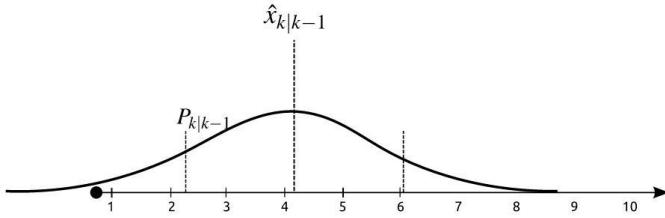


Figura 9.2: Función de densidad de la variable de estado $\hat{x}_{k|k-1}$

cuya función de densidad es:

$$f_{\mu, \sigma^2}(r) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(r-\mu)^2}{2\sigma^2}} \quad (9.9)$$

Corrección

En actualizar la medida, es decir, obtener una nueva observación del estado \mathbf{x}_k tras una acción cualquiera al sistema \mathbf{u}_{k-1} , ésta tendrá también ruido (\mathbf{v}_k):

$$\mathbf{z}_k = \mathbf{H} \cdot \mathbf{x}_k + \mathbf{v}_k \quad (9.10)$$

y se expresa mediante una nueva distribución normal de media z_k , que en el espacio de estados se representa como $H^{-1}z_k$:

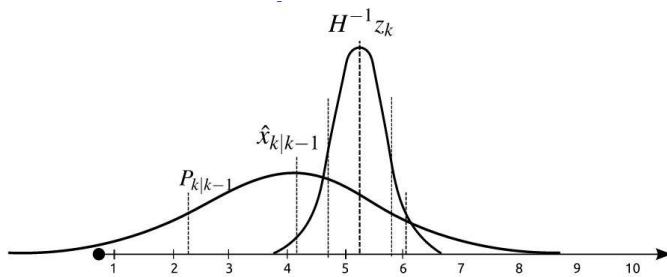


Figura 9.3: Comparación con función de densidad de la medición z_k

Como las distribuciones de la estimación $\hat{x}_{k|k-1}$ y la observación z_k no son iguales se combinan multiplicándolas para tener una mejor aproximación del estado *a posteriori* en el tiempo k . La multiplicación de dos funciones de densidad f_1 y f_2 tiene como resultado otra función de densidad

$$\begin{aligned} f_1(r) \cdot f_2(r) &= f_p(r) = \frac{1}{\sqrt{2\pi\sigma_1^2}} e^{-\frac{(r-\mu_1)^2}{2\sigma_1^2}} \cdot \frac{1}{\sqrt{2\pi\sigma_2^2}} e^{-\frac{(r-\mu_2)^2}{2\sigma_2^2}} = \\ &= \frac{1}{2\pi\sqrt{\sigma_1^2\sigma_2^2}} e^{-\left(\frac{(r-\mu_1)^2}{2\sigma_1^2} + \frac{(r-\mu_2)^2}{2\sigma_2^2}\right)} \end{aligned} \quad (9.11)$$

de media μ_p y varianza σ_p^2 :

$$\mu_p = \mu_1 + \frac{\sigma_1^2(\mu_2 - \mu_1)}{\sigma_1^2 + \sigma_2^2} \quad \sigma_p^2 = \sigma_1^2 - \frac{\sigma_1^4}{\sigma_1^2 + \sigma_2^2} \quad (9.12)$$

Substituyendo en este caso con las dos funciones anteriores se tiene que la estimación del estado *a posteriori* es

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \frac{\mathbf{P}_{k|k-1}\mathbf{H}_k^T(z_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1})}{\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^T + \mathbf{R}_k} \quad (9.13)$$

que se puede expresar como

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \cdot \tilde{\mathbf{y}}_k \quad (9.14)$$

donde \mathbf{K}_k es la ganancia de Kalman y se representa como

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^T \cdot (\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^T + \mathbf{R}_k)^{-1} \quad (9.15)$$

e $\tilde{\mathbf{y}}_k$ es la innovación:

$$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}_k \cdot \hat{\mathbf{x}}_{k|k-1} \quad (9.16)$$

Finalmente se actualiza la covarianza del error asociada a esta última estimación, substituyendo los valores de las varianzas tal y como se ha hecho para la media en 9.13:

Finalmente se actualiza la covarianza del error asociada a esta última estimación

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_{k|k-1} \quad (9.17)$$

Implementación

Se explicará paso por paso cómo se ha implementado el filtro de Kalman que se ha utilizado en el Quadcopter en código *C*.

Se determina el tiempo de muestreo mediante la diferencia de tiempo entre dos lecturas consecutivas del sensor y con ello se obtiene dt . Se considera en la estimación del estado sigue una dinámica tal que

$$\begin{aligned} \hat{\mathbf{x}}_{k|k-1} &= \mathbf{F}_{k-1} \cdot \hat{\mathbf{x}}_{k-1|k-1} + \mathbf{B}_{k-1} \cdot \mathbf{u}_{k-1} = \\ \left[\begin{array}{c} \theta \\ \dot{\theta}_b \end{array} \right]_{k|k-1} &= \left[\begin{array}{cc} 1 & -\Delta t \\ 0 & 1 \end{array} \right] \left[\begin{array}{c} \theta \\ \dot{\theta}_b \end{array} \right]_{k-1|k-1} + \left[\begin{array}{c} \Delta t \\ 0 \end{array} \right] \dot{\theta}_k \end{aligned} \quad (9.18)$$

A este Filtro se le conoce por Filtro de Schmidt ya que su segundo estado es el *bias* de la velocidad. Desarrollandolo se tiene que:

$$\left[\begin{array}{c} \theta \\ \dot{\theta}_b \end{array} \right]_{k|k-1} = \left[\begin{array}{c} \theta + \Delta t(\dot{\theta} - \dot{\theta}_b) \\ \dot{\theta}_b \end{array} \right] \quad (9.19)$$

Si se considera que $rate = (\dot{\theta} - \dot{\theta}_b)$ se expresa 9.19 como

```
1 rate=velocity-bias;
2 angle+=dt*rate;
```

Para estimar la varianza $\mathbf{P}_{k|k-1}$ se tiene

$$\begin{aligned} \begin{bmatrix} P_{00} & P_{01} \\ P_{10} & P_{11} \end{bmatrix}_{k|k-1} &= \begin{bmatrix} 1 & -\Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} P_{00} & P_{01} \\ P_{10} & P_{11} \end{bmatrix}_{k-1|k-1} \begin{bmatrix} 1 & -\Delta t \\ 0 & 1 \end{bmatrix}^T + \begin{bmatrix} Q_\theta & 0 \\ 0 & Q_{\dot{\theta}_b} \end{bmatrix} \Delta t = \\ &= \begin{bmatrix} P_{00} + \Delta t(\Delta t P_{11} - P_{01} - P_{10} + Q_\theta) & P_{01} - \Delta t P_{11} \\ P_{10} - \Delta t P_{11} & P_{11} + Q_{\dot{\theta}_b} \Delta t \end{bmatrix} \end{aligned} \quad (9.20)$$

que en código C se implementa como

```
3 P[0][0] += dt * (dt * P[1][1] - P[0][1] - P[1][0] + Q_angle);
4 P[0][1] -= dt * P[1][1];
5 P[1][0] -= dt * P[1][1];
6 P[1][1] += Q_gyroBias * dt;
```

La innovación \tilde{y}_k es

$$\tilde{y}_k = z_k - \mathbf{H}_k \cdot \hat{\mathbf{x}}_{k|k-1} = z_k - \theta_{k|k-1} \quad (9.21)$$

Que en el programa que se ejecuta se calcula como la diferencia de la lectura y el estado:

```
7 y=newangle-angle;
```

Para calcular la ganancia de Kalman se debe calcular antes \mathbf{S}_k :

$$\begin{aligned} \mathbf{S}_k &= \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{R}_k = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} P_{00} & P_{01} \\ P_{10} & P_{11} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \mathbf{R} = \\ &= P_{00k|k-1} + R_k \end{aligned} \quad (9.22)$$

y para la ganancia es

$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}^T \mathbf{S}_k^{-1} = \begin{bmatrix} K_0 \\ K_1 \end{bmatrix}_k = \frac{\begin{bmatrix} P_{00} \\ P_{10} \end{bmatrix}_{k|k-1}}{\mathbf{S}_k} \quad (9.23)$$

La implementación de la ganancia de Kalman queda como:

```
8 S = P[0][0] + R_angle;
9
10 K[0] = P[0][0] / S;
11 K[1] = P[1][0] / S;
```

Por último queda asignar los nuevos valores de las estimaciones. Para el caso del estado $\hat{\mathbf{x}}_{k|k}$ se tiene que

$$\hat{\mathbf{x}}_{k|k} = \begin{bmatrix} \theta \\ \dot{\theta}_b \end{bmatrix}_{k|k-1} + \begin{bmatrix} K_0 \tilde{y} \\ K_1 \tilde{y} \end{bmatrix}_k \quad (9.24)$$

y para la matriz de covarianzas del error

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}) \hat{\mathbf{x}}_{k|k-1} = \begin{bmatrix} P_{00} & P_{01} \\ P_{10} & P_{11} \end{bmatrix}_{k|k-1} - \begin{bmatrix} K_0 P_{00} & K_0 P_{01} \\ K_1 P_{00} & K_1 P_{01} \end{bmatrix} \quad (9.25)$$

El código que implementa este último paso del filtro es

```
12 angle += K[0] * y;  
13 bias += K[1] * y;  
14  
15 P[0][0] -= K[0] * P[0][0];  
16 P[0][1] -= K[0] * P[0][1];  
17 P[1][0] -= K[1] * P[0][0];  
18 P[1][1] -= K[1] * P[0][1];
```

Anexo D: Identificación de los parámetros del Quadcopter

Para poder trabajar en particular con el Quadcopter construido es necesario saber qué lo caracteriza o distingue de los demás. Parámetros relevantes y necesarios que se utilizan en el modelo implementado en Simulink deben ser, entonces, medidos para que se simule el mismo aparato con iguales características.

Se consideran relevantes la matriz de inercias, la longitud de los brazos l y la fuerza e intensidad (consumo) de los motores respecto a la señal PWM con que se controlan.

Con este conjunto de parámetros se considera que el Quadcopter está totalmente definido y puede obtenerse la ley de control.

Masa total

El peso que deberán contrarrestar los motores para permitir al Quadcopter planear se obtiene mediante una báscula pesando el conjunto. Resulta una masa m total de 0,96 kg.

Inercias

Como se ha comentado en la sección 3, se supone que el Quadcopter es un rotor esférico, y por tanto de su tensor de inercia todos los productos de inercia serán nulos. Se tendrá entonces una matriz diagonal como tensor. En este apartado, entonces, el objetivo será la medición de los momentos de inercia I_{xx} , I_{yy} y I_{zz} correspondientes a los ángulos *Roll*, *Pitch* y *Yaw* respectivamente. Para el cálculo de los dos primeros se usa una bancada compuesta por un muelle de constante de torsión D conocida del que cuelga el quadcopter. De querer determinar el momento de inercia I_{xx} , se cuelga desde uno de los extremos del eje x , tal y como se observa en la figura 9.4

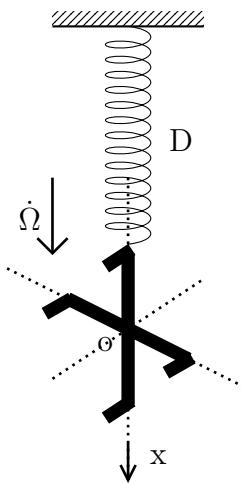


Figura 9.4: Bancada para la determinación de las inercias

Para un sólido rígido se tiene que

$$\mathbf{M} = \frac{d\mathbf{L}_o}{dt} \quad (9.26)$$

donde \mathbf{M} es el momento resultante y \mathbf{L}_o es el momento cinético o angular aplicado en el punto o , que en este caso corresponde al centro de gravedad del cuerpo. Éste se expresa como

$$\mathbf{L}_o = \mathbf{I}\dot{\boldsymbol{\Omega}} \quad (9.27)$$

El momento resultante puede expresarse entonces de la forma

$$\mathbf{M} = \frac{d(\mathbf{I}\dot{\boldsymbol{\Omega}})}{dt} = \mathbf{I}\ddot{\boldsymbol{\Omega}} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} \ddot{\Omega}_x \\ \ddot{\Omega}_y \\ \ddot{\Omega}_z \end{bmatrix} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{\Omega}} \\ 0 \\ 0 \end{bmatrix} = I_{xx}\ddot{\boldsymbol{\Omega}} \quad (9.28)$$

en suponer que el tensor de inercia no varía con el tiempo. Se asume que el eje de rotación $\ddot{\boldsymbol{\Omega}}$ pasa por el centro de masas, de lo contrario debería aplicarse el teorema de Steiner. Para el muelle se tiene que, según la ley de Hooke

$$M = -D \cdot \boldsymbol{\Omega} \quad (9.29)$$

donde $\boldsymbol{\Omega}$ es el ángulo girado en la dirección longitudinal del muelle. Igualando las expresiones 9.28 y 9.29 se llega a

$$-D \cdot \boldsymbol{\Omega} = I_{xx}\ddot{\boldsymbol{\Omega}} \implies \ddot{\boldsymbol{\Omega}} + \frac{D}{I_{xx}}\boldsymbol{\Omega} = 0 \quad (9.30)$$

La solución de la ecuación anterior es la de un oscilador armónico de periodo

$$T = 2\pi\sqrt{\frac{I_{xx}}{D}} \quad (9.31)$$

Ésto se cumple en oscilaciones pequeñas cercanas al punto de equilibrio, es decir, donde $\boldsymbol{\Omega} = \dot{\boldsymbol{\Omega}} = \ddot{\boldsymbol{\Omega}} = 0$. Por lo tanto, el periodo de una oscilación es suficiente para poder calcular el momento de inercia en estar directamente relacionados. Para obtener resultados más precisos se medirá el tiempo varias veces.

Para calcular la constante de torsión del muelle se le ha sometido a ensayo, sometiéndolo a diferentes deformaciones elásticas y observando el momento resultante:

$\boldsymbol{\Omega}(^{\circ})$	$\mathbf{M}(g \cdot 10cm)$	$\mathbf{M}(N \cdot m)$
30	33	0.0323
45	46	0.0451
60	62	0.0608
90	93	0.0912

En calcular la relación entre ambas magnitudes para obtener la constante de torsión del muelle, por regresión lineal se obtiene que la pendiente es de $1,0095 \text{ } g \cdot 10cm/\text{grad}$. La constante entonces puede expresarse en unidades del Sistema Internacional como

$$D = 1,0095 \frac{g \cdot 10cm}{\text{grad}} \cdot \frac{9,81 \text{ N}}{1000 \text{ g}} \cdot \frac{1 \text{ m}}{100 \text{ cm}} \cdot \frac{360 \text{ grad}}{2\pi \text{ rad}} = 0,0567 \frac{\text{Nm}}{\text{rad}} \quad (9.32)$$

En hacer oscilar el Quadcopter en el eje x se obtiene:

Eje	Muestras(s)				Media(s)	Momento de inercia($kg \cdot m^2$)
<i>x</i>	2.3	2.3	2.5	2.2	2.35	$7,932 \cdot 10^{-3}$
	2.3	2.4	2.5	2.4		
	2.4	2.3	2.3	2.4		
	2.3	2.4	2.3	2.3		
<i>y</i>	2.4	2.3	2.2	2.4	2.31	$7,664 \cdot 10^{-3}$
	2.2	2.4	2.3	2.2		
	2.4	2.3	2.3	2.4		
	2.3	2.3	2.3	2.3		

y por tanto se obtiene que $I_{xx} = 7,931 \cdot 10^{-3} \ kg \cdot m^2$ y $I_{yy} = 7,664 \cdot 10^{-3} \ kg \cdot m^2$.

Para el cálculo del I_{zz} se usa una bancada parecida a la anterior, pero eliminando el muelle, de tal manera que pueda oscilar desde uno de los agujeros de uno de los brazos. El periodo que resulta en hacer oscilar el Quadcopter es

$$T = 2\pi\sqrt{\frac{I}{mgd}} \quad (9.33)$$

donde d es la distancia entre el centro de masas y el punto del que se oscila. Mediante el teorema de Steiner se descompone la inercia que se obtiene no es respecto al eje que pasa por el su centro, sino mayor. Ésta es

$$I = I_{zz} + md^2 \quad (9.34)$$

Y por lo tanto se obtiene que

$$T = 2\pi\sqrt{\frac{I_{zz} + md^2}{mgd}} \quad (9.35)$$

En el experimento se tiene que $g = 9,81 \ m/s^2$, $d = 0,13 \ m$ y $m = 0,96 \ kg$. Se realizan 8 mediciones de 5 oscilaciones cada una, por lo que la media que se obtiene se divide entre 5

Eje	Muestras(s) (5 oscil.)				Media(s)	Momento de inercia($kg \cdot m^2$)
<i>z</i>	4.6	4.5	4.5	4.6	4.575	$9,741 \cdot 10^{-3}$
	4.6	4.6	4.6	4.6		

y por tanto se tiene que el tensor de inercia del Quadcopter es

$$I = \begin{bmatrix} 7,932 & 0 & 0 \\ 0 & 7,664 & 0 \\ 0 & 0 & 9,741 \end{bmatrix} \cdot 10^{-3} \quad (9.36)$$

Longitud de brazo

Queda especificado en el modelo del Frame elegido que es de 33 cm la distancia de motor a motor en la diagonal principal. La longitud que se considera es de $l = 16,5 \ cm$.

Motores

Para poder controlar la planta es necesario saber cómo responden los actuadores para realizar un control preciso. De éstos se debe conocer qué fuerza y momento proporcionan, así como el consumo de energía para un PWM dado. La respuesta varía según las aspas

que se utilicen, por lo que las curvas que se obtengan serán únicas para este caso. De cambiarlas tanto la fuerza, consumo y momentos generados se verían modificados.

Para obtener una curva de la fuerza respecto del PWM se pesa con una balanza la acción del motor. La rudimentaria bancada que se tiene para tal objetivo es un codo articulado en el punto en donde se crea el ángulo, de tal manera que la distancia del eje del motor al fulcro y de éste a la balanza sea la misma (en este caso de 12 cm). El consumo que se tiene se obtiene mediante un amperímetro conectado en serie al ESC.

Se transmite al ESC cómo se debe hacer funcionar el motor desde la Raspberry Pi por medio de una señal PWM. El código que se ejecuta para hacer las pruebas es:

```

1 #include <stdio.h>
2 #include <termios.h>
3 #include <wiringPi.h>
4 #include <unistd.h>
5
6 #define PIN 2 // pin 13
7
8 int main(void){
9
10    int i,p;
11    wiringPiSetup();
12
13    pinMode(PIN,OUTPUT);
14    digitalWrite(PIN,LOW);
15
16    for (i=0;i<=100;i+=5){
17        for (p=0;p<250;p++){
18            digitalWrite(PIN,HIGH);
19            delayMicroseconds(950+i*10);
20            digitalWrite(PIN,LOW);
21            delayMicroseconds(19050-i*10);
22
23            printf("PWM(%d) : %d\n",i,p);
24        }
25    }
26    return 0;
27 }
```

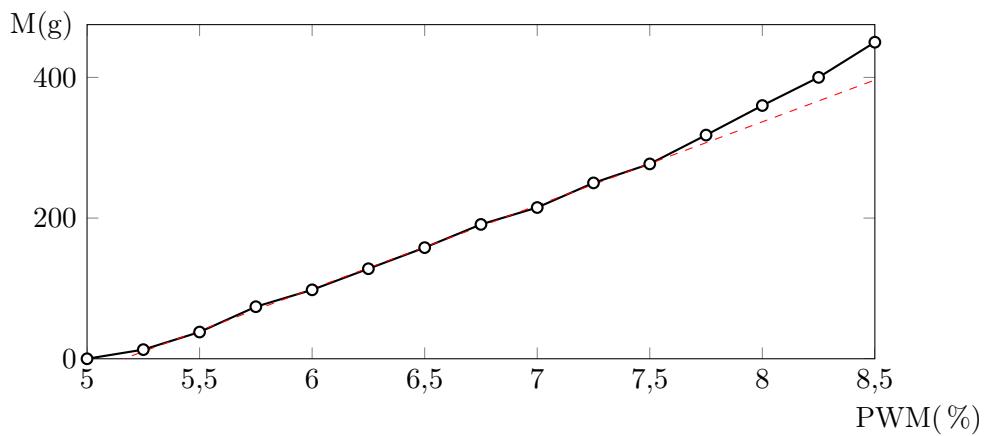
Se obtiene una lectura de fuerza y de intensidad por cada incremento del 0.25 % del PWM enviado al Variador. Las lecturas que se obtienen son:



PWM(%)	Intensidad(A)	Masa(g)
5	0	0
5,25	0,25	13
5,5	0,5	38
5,75	0,9	74
6	1,38	98
6,25	1,8	128
6,5	2,4	158
6,75	2,9	191
7	3,5	215
7,25	4	250
7,5	4,6	277
7,75	5,5	318
8	6,5	360
8,25	8	400
8,5	9,15	450

El PWM(%) que se envía está comprendido entre el 5 % y el 10 % del ciclo de trabajo, que corresponden al mínimo y máximo del motor. Para un PWM elevado la fuerza se considera demasiado grande y al no preverse que se haga funcionar al sistema en semejantes condiciones, no se estudia. Se detecta una relación proporcional del PWM con la fuerza generada.

Graficado:

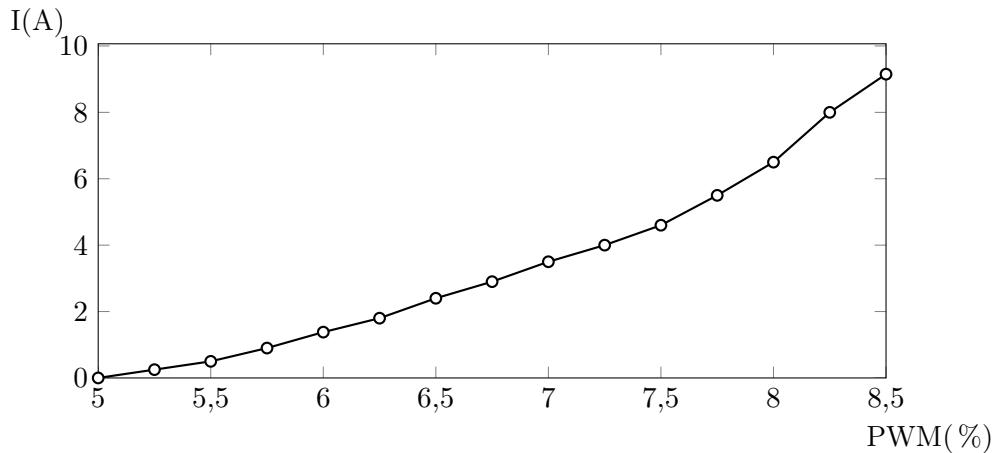


Se considera que una regresión lineal será suficiente en observarse una relación tan directamente proporcional. Para ello se decide utilizar los valores comprendidos entre los $PWM = 5,5$ y $PWM = 7,5$ porque no se espera hacer trabajar al sistema con los valores extremos. Se obtiene que la fuerza $f(PWM)$ del motor será:

$$f(PWM) = 119 \cdot PWM - 613 \quad (9.37)$$

con un coeficiente de determinación en la regresión de $R^2 = 0,999$. Este resultado facilita el problema en ser una relación entre las dos magnitudes lo suficientemente sencilla.

Respecto a la intensidad, será caso de estudio únicamente el saber cuánto consume en el punto de equilibrio, pues no es necesaria la ecuación de la curva para poder volar. Se verá una aproximación de lo que puede llegar a consumir cada motor para asegurar una autonomía de vuelo no irrisoria. Se tiene



Como en el punto de equilibrio cada motor soporta una cuarta parte del peso, el PWM debería corresponder al de una fuerza de $m_{Quadcopter} \cdot g/4 \approx 0,250\text{kg}$. El consumo correspondiente es de 4A aproximadamente. Entre los cuatro motores se consumirán $4 \cdot 4 = 16\text{A}$. En ser la batería de 2200mAh se supone una autonomía de 8.25 minutos aproximadamente:

$$\frac{2,2 \text{ A} \cdot \text{h}}{16 \text{ A}} \cdot \frac{60 \text{ minutos}}{1\text{h}} = 8,28 \text{ minutos} \quad (9.38)$$