Escola Politècnica Superior de Castelldefels
UNIVERSITAT POLITÈCNICA DE CATALUNYA

TRABAJO FINAL DE CARRERA

TITULO: Control de un Quadrotor mediante la plataforma Arduino

TITULACIÓN: Ingeniería Técnica de Telecomunicaciones, especialidad Sistemas de Telecomunicaciones

AUTOR: Christian Nadales Real

DIRECTOR: Gemma Hornero Ocaña

CO-DIRECTOR : Oscar Casas Piedrafita

FECHA: 27 de julio de 2009

TITULO: Control de un Quadrotor mediante la plataforma Arduino

Autor: Christian Nadales Real

Titulación: Ingeniería Técnica de Telecomunicaciones

Especialidad: Sistemas de Telecomunicaciones

Plan: 2000

Director: Gemma Hornero Ocaña

Co-director: Oscar Casas Piedrafita

Departamento: EEL

Director del TFC

Registro:

Titulo: Control de un Quadrotor mediante la plataforma Arduino

Autor: Christian Nadales Real

Director: Gemma Hornero Ocaña

Co-director: Oscar Casas Piedrafita

Fecha: 27 de julio de 2009

Resumen

El objetivo de este trabajo es diseñar un sistema UAV quadrotor. Con él se desea tener una plataforma abierta que permita en el futuro implementar y probar nuevos sistemas de control y navegación. Para conseguir este objetivo se diseñará el hardware y software necesario para tener una arquitectura completa y fácilmente escalable, así como un sencillo sistema de control que permita su estabilidad.

Title: Quad-rotors control through the Arduino platform

Author: Christian Nadales Real

Director: Gemma Hornero Ocaña

Co-director: Oscar Casas Piedrafita

Date: July,27 2009

Overview

The aim of this work is to design an UAV quadrotor system. With it we will have an open platform which will allow future implementing and testing of new navigation and control systems. To achieve this goal all necessary hardware and software will be designed for a complete and easily scalable architecture, as well as a simple control system which permits its stability.

Agradecimientos

En primer lugar quisiera agradecer a Gemma Hornero
la oportunidad que me ha brindado para realizar este proyecto
y aprender de él, y al grupo de investigación
de Instrumentación Sensores e Interfaces el permitirme realizarlo.

A Oscar.Casas por su gran ayuda y acertados
aportes durante el desarrollo de este proyecto.

A mi pareja por su comprensión durante
el tiempo que le dediqué a este proyecto.

INDICE

INDICE FIGURAS

INDICE TABLAS

INTRODUCCIÓN Y OBJETIVOS

En los últimos años ha habido un avance muy importante en el desarrollo de vehículos no tripulados (del inglés UAV- Unmanned Aerial Vehicle) sobre todo para su uso en futuras aplicaciones civiles. Este tipo de vehículo se está empleando en tareas de búsqueda y rescate, vigilancia comercial, exploración de edificios, entre otras. Los UAVs son muy útiles, principalmente, cuando nos encontramos en entornos de difícil acceso o con un cierto peligro. Los avances tecnológicos han hecho posible el desarrollo de dichos vehículos aéreos, gracias a los sensores de escala reducida (MEMS - Micro Electromechanical Systems), avances en microcontroladores, ofreciendo la posibilidad de cálculos más complejos en poco espacio así como la mejora en el almacenamiento de energía, a hecho posible la construcción de estos sistemas autónomos. Cada vez más, las líneas de investigación de estos sistemas va en aumento. Sin embargo, el desarrollo de sistemas de control para este tipo de vehículos no es trivial, debido principalmente a la dinámica tan compleja inherente en los sistemas aerodinámicos, los cuales son multivariables, y presentan diversas características no lineales. Por lo tanto, se requiere un modelo de control que proporcione una buena estabilidad en vuelo autónomo, o que ayude al menos al pilotaje del vehículo, proporcionando alta maniobrabilidad y robustez ante perturbaciones externas.

El objetivo de este proyecto es el desarrollo de un sistema quadrotor, estos sistemas tienen características únicas en el mundo de los UAV, como por ejemplo la gran maniobrabilidad y posibilidad de trabajar tanto en exteriores como en interiores. Se implementará un esquema de estabilización sencillo, ya que la idea principal de este trabajo es la de proporcionar una plataforma, en la cual en posteriores trabajos puedan seguir experimentando diferentes funciones de control y navegabilidad.

Esta plataforma es la primera versión de un conjunto de estudios llevados por la Escuela Politécnica Superior de Castelldefels. El nombre escogido para esta plataforma es " **FLISI** " del origen (FLights of Instrumentation Sensors and Interfaces group), como este trabajo es la primera versión, el nombre adquiere este sistema es **FLISI1**, con la idea que las siguientes versiones utilicen el mismo nombre (FLISI2, FLISI3, …).

La memoria está estructurada en cuatro capítulos. El primer capítulo trata de una introducción a los vehículos aéreos no tripulados, luego seguiremos con otro que profundiza en los sistemas quadrotores. El tercer capítulo se presenta el diseño del sistema quadrotor FLISI, por último acabaremos con un capítulo de pruebas realizadas en el laboratorio.

CAPITULO 1. Vehículos aéreos no tripulados

1.1. Sistemas UAVs

1.1.1. Definición

Se entiende por una aeronave no tripulada (UAV: *Unmanned Aerial Vehicle*) aquella que es capaz de navegar sin llevar a bordo ningún piloto, estas aeronaves pueden ser controladas desde una estación base o llevar una programación preestablecida.

Estos vehículos han sido utilizados en aplicaciones militares tales como reconocimiento de terreno y ataque. En los últimos años investigadores del ámbito de robótica e inteligencia artificial aeronáutica y control están enfocando sus esfuerzos hacia esta línea para aplicar dicho concepto a aplicaciones civiles. En la actualidad nos podemos encontrar con situaciones en las que la visión aérea sería de gran ayuda, por ejemplo en la detección de incendios, control policial en situaciones de riesgo, reconocimiento de desastres naturales. Estos son algunos ejemplos en los que podemos aplicar el concepto de UAV en misiones civiles.

Las ventajas de un UAV se pueden resumir en; menor coste que las aeronaves tripuladas, no se arriesgan vidas, capacidad de incorporar muchos sensores y la posibilidad de acceder a sitios peligrosos o de difícil acceso.

1.1.2. Clasificación

En la actualidad podemos clasificar a los UAV según sus características de vuelo como se observa en la tabla 1.1. Esta comparativa abarca tanto aplicaciones civiles como militares. [1]

Tabla 1.1. Clasificación UAV.

Categoría	Acrónimo	Alcance (km)	Altitud de vuelo (m)	Autonomía (h)	Carga máxima (kg)
Micro < 250 gr	Micro	< 10	250	1	< 5
Mini < 25 kg	Mini	< 10	150 y 300	< 2	< 30
Alcance cercano	CR	10 a 30	3.000	2 a 4	150
Alcance corto	SR	30 a 70	3.000	3 a 6	200
Alcance medio	MR	70 a 200	5.000	6 a 10	1.250
Altitud baja	LADP	> 250	50 a 9.000	0,5 a 1	350
Autonomía media	MRE	> 500	8.000	10 a 18	1.250
Autonomía alta Altitud baja	LALE	> 500	3.000	> 24	< 30

Autonomía alta Altitud media	MALE	> 500	14.000	24 a 48	1.500
Autonomía alta Altitud alta	HALE	> 2.000	20.000	24 a 48	12.000
Combate	UCAV	1500	10.000	2	10.000
Ofensivo	LETH	300	4.000	3 a 4	250
Señuelo	DEC	500	5.000	< 4	250
Estratosférico	STRATO	>2.000	20.000 y 30.000	>48	ND
EXO-estratosférico	EXO	ND	>30.000	ND	ND

La mayoría de los UAV civiles se encuentran entre la categoría Micro y alcance cercano.

Fig. 1.1 Aeroplano

Fig. 1.2 Dirigible

Fig. 1.3 Helicóptero (Dragranfly 2)

Fig. 1.4 Quadrotor (Draganflyer)

La principal ventaja de los aeroplanos (figura 1.1), es la facilidad de control y guiado, así como su alta fiabilidad ante condiciones extremas o difíciles. El problema más grande es su falta de maniobralidad en espacios reducidos.

Los dirigibles, figura 1.2, destacan por su estabilidad en el aire y su gran autonomía. En contra tienen una reducida relación capacidad de carga / volumen. Su principal problema es la poca maniobrabilidad. Son muy

adecuados para tareas de interiores, siempre que sean interiores de dimensiones grandes como por ejemplo un estadio o salas de conciertos.
Los vehículos más utilizados en condiciones de poco alcance (< 10 km) sin duda son los helicópteros (figura 1.3) ya que tienen una relación capacidad de carga / volumen muy grande, por contra presentan una gran dificultad de control. En los quadrotores (figura 1.4) las características que destacan sobre los otros vehículos son: tanto su dinámica como capacidad de vuelo lo hacen único en maniobrabilidad. La capacidad de vuelo en interiores lo diferencian del resto, consiguiendo así una de las características más importantes a la hora de elegir un UAV u otro.

En la tabla 1.2 [1] podemos ver una comparativa de estos UAV civiles de corto alcance (< 30 km).

Tabla 1.2. Comparativa UAV corto alcance.

Característica	Helicóptero	Aeroplanos	Dirigibles	Quadrotor
Capacidad de vuelo estacionario	***		****	***
Velocidad de desplazamiento	***	****	*	**
Maniobralidad	***	*	*	****
Autonomía de vuelo	**	***	****	*
Resistencia a perturbaciones externas	**	****	*	**
Auto Estabilidad	*	***	***	**
Capacidad de vuelos verticales	****	*	**	****
Capacidad de carga	***	****	*	**
Capacidad de vuelo en interiores	**	*	***	****
Techo de vuelo	**	****	***	*

Existe vehículos que son variaciones de los quadrotores clásicos, al utilizar tres ejes en vez de cuatro, lo que hace necesario utilizar 6 motores por cuestiones de dinámica y navegación, pero que tienen características similares.

En la figura 1.5 podemos ver un modelo comercial llamado Dragranflyer X6 [2]. Este modelo, que ganó el premio al mejor invento del año 2008 en el sector aeronáutico por la revista Popular Science, incorpora GPS, puede alcanzar velocidades hasta 30 km/h y se le pueden incorporar cámaras de video de alta definición (HD) con una capacidad de carga de unos 0.5 kg. Actualmente la policía canadiense lo ha adquirido para tareas de vigilancia, ya que posee características muy buenas de estabilidad, rápida carga de la batería y sobre todo capacidad para alcanzar espacios de difícil acceso.

Fig. 1.5 Draganflyer X6

1.1.3. Aplicaciones

Existen gran variedad de aplicaciones enfocadas a los UAV debido a su gran versatilidad. A continuación se describen alguna de ellas.[1]

Misiones militares, existe una gran diferencia a nivel económico entre el ámbito militar y el civil. En la tabla 1.1 hemos podido observar el gran número de tipos de UAV que discurren en el ámbito militar. En espionaje los UAV son bastante utilizados, desde los más pequeños (difícil de detectar por los radares) hasta los más grandes. Una de las grandes características a destacar a la hora de utilizar los UAV en misiones militares es que no arriesgamos vidas humanas y lo único que podemos perder es todo el sistema UAV.

Supervisión, una de las capacidades mas valoradas es la gran maniobrabilidad que presentan algunos sistemas UAV como puede ser los quadrotores. Esto puede ser utilizado para la inspección de obras civiles como grandes puentes, edificios o estructuras de gran envergadura.
Estos vehículos pueden ir dotados de cámaras de video y ayudar a la supervisión de la obra desde diferentes ángulos sin poner en riesgo las vidas humanas y reduciendo el tiempo de ejecución ya que no hay que montar ninguna estructura externa para poder acceder al sitio en concreto.

Fotografía aérea, la fotografía aérea o adquisición de datos aéreos es un área importante ya que podemos extraer bastante información del terreno. En este aspecto podemos utilizar tanto los vehículos más pequeños hasta los más grandes, ya que según la altitud en la que queramos hacer las fotografías elegiremos uno u otro.

Vigilancia, los equipos de seguridad tanto privados como los policiales pueden utilizar estos sistemas para la vigilancia de una zona en concreto o de soporte en alguna tarea especifica.

Reconocimiento de desastres, al ser vehículos no tripulados, puede servir para el reconocimiento inicial de desastres tales como explosiones nucleares o reconocimientos en lo que se pondría en peligro la integridad de los pilotos.

En resumen, los UAV son idóneos para aplicaciones donde sea necesario el sensado de parámetros y adquisición de datos, especialmente en lugares de difícil acceso y sin poner en peligro ninguna vida humana.

1.2. Quadrotor

1.2.1. Definición

Un quadrotor se puede definir como una aeronave que se eleva y se desplaza por el movimiento de cuatro motores colocados en los extremos de una estructura en forma de cruz. Normalmente se utiliza el nombre inglés quadrotor aunque también existe la traducción cuadricóptero. Tal como se presenta en la figura 1.6, el vehículo dispone de 4 motores con sus palas respectivas, se utiliza la velocidad de los motores para controlar la estabilidad y movimientos del vehículo aéreo.

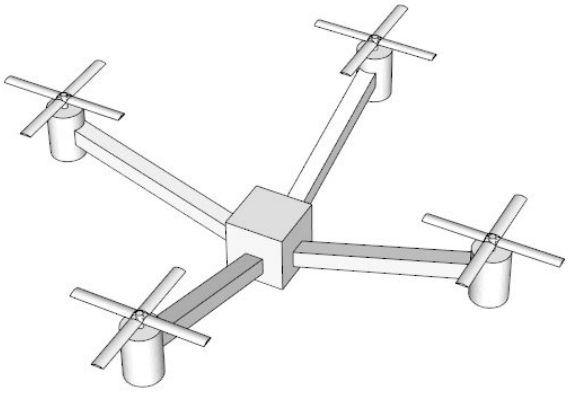

Fig 1.6 Sistema Quadrotor.

1.2.2. Principales características

Los sistemas quadrotores se sitúan en la categoría de mini UAVs (< 25 kg). Como se observa en la tabla 1.1 las características generales de vuelo de esta categoría son las siguientes:

- Alcance : < 10 km.
- Altitud de vuelo: < 300 m
- Autonomía: < 2 h
- Carga máxima: < 5 kg.

Una de las características a destacar es la gran **maniobrabilidad** que pose este tipo de vehículo. Al disponer de cuatro motores el control es bastante exacto, lo que ayuda a utilizarlo en aplicaciones donde la exactitud de vuelo estacionario sea muy importante. Una aplicación donde se aprecia esta característica es en la navegación de interiores y sitios de espacio reducido.

Como en el helicóptero, estos vehículos disponen de una **capacidad de vuelo vertical** que los hacen únicos, esta función es ventajosa cuando no queremos tener mucha velocidad horizontal y cuando queremos tener una buena capacidad de vuelo estacionario, lo que ayuda a elegir este tipo de sistemas cuando se quieren adquirir datos desde el vehículo.

El problema fundamental de los quadrotores es su **control**. El sistema debe de incorporar mecanismos de estabilización para ayudar a la navegación. La **capacidad de carga** es bastante alta con relación al peso de todo el sistema (5 kg / 25 kg) , podemos encontrar vehículos que soporten una carga superior al peso que tienen. Esta característica hace posible el incorporar un gran número de sensores.

Una de las características más importantes a tener en cuenta en los sistemas de vuelo es la **autonomía**. La autonomía de vuelo no suele ser muy buena (< 2 h), esta fue una de las limitaciones por la que los UAV tardaron un cierto tiempo en avanzar. Actualmente se están realizando avances importantes en las baterías, proporcionado más capacidad (mAh) y reduciendo los tamaños.

1.2.3. Modelos comerciales

Existen varias soluciones comerciales que podemos adquirir en el mercado. Destacamos dos grandes plataformas: Draganfly y Mikrokopter.son las más utilizadas y que presentan características diferentes que permiten ubicar el desarrollo realizado en este proyecto.

Draganfly

Existen 3 versiones de este quadrotor, el nombre comercial de este producto es Draganflyer V. Se compone de una estructura de fibra de carbono e incorpora una serie de sensores como acelerómetros, giroscopios y cámara de video. [3] Draganflyer utiliza motores de corriente continua de 12 V y el sistema de alimentación lo implementan con una batería de Lithium-Polymer de 3 celdas que proporciona una autonomía aproximadamente de 15 minutos.

Las dimensiones y características de vuelo de todo el conjunto son las siguientes;

- Tamaño: 76 cm. (30”) de diámetro.
- Peso: 483 g (17 ounces)
- Velocidad máxima: aproximadamente 32 km/h (20 mph)

Este modelo ha sido utilizado por el MIT (Massachusetts Institute of Technology) que está desarrollando un sistema llamado UAV Swarm Health Management Project. El multi-vehículo experimental desarrollado por el MIT utiliza varios Draganflyer [4], proporciona un equipo de seguimiento, un sistema de posicionamiento de vigilancia y control de múltiples vehículos aéreos no tripulados. En la figura 1.7 podemos ver la envergadura de este proyecto al estar trabajando 9 quadrotores a la vez en un espacio reducido.

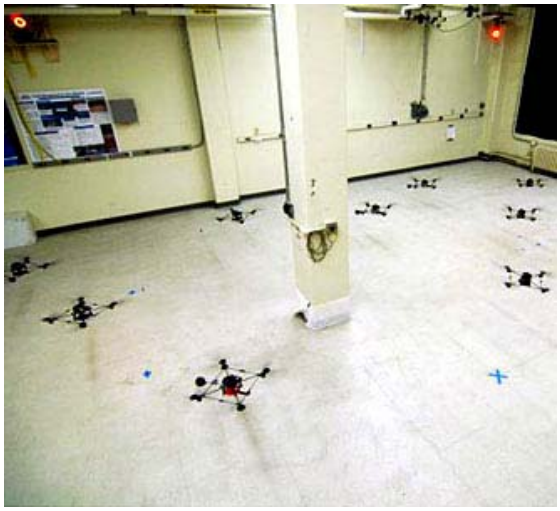

Fig. 1.7 MIT. UAV Swarm Health Management Project

El Draganflyer se distribuye por piezas o todo en conjunto, las diferentes versiones oscilan entre unos precios de 600 € hasta 1800 €, encontrando versiones más simples hasta versiones mas complejas con cámaras de video de calidad y sistemas de estabilización de la imagen.

Mikrokopter

El Mikrokopter fue creado por Holger Buss e Ingo Busker en Alemania el 26 de Octubre de 2006.[5]

Utilizan motores Brushless los cuales se comentaran en el siguiente apartado. Actualmente existen 4 versiones diferentes de mikrokopter, desde la más pequeña, 30 cm de distancia entre motores, hasta la más grande con 48 cm. El mikrokopter se suele distribuir con un kit básico, se puede complementar con tres placas adicionales. Estas placas nos pueden proporcionar características muy importantes como la posibilidad de mantener la posición en el aire (incluso luchando contra el viento) o que vuelva a la posición de partida ("vuelta a casa") mediante la posición GPS.

Al existir módulos, los precios pueden oscilar bastante. Podemos encontrar una versión reducida por 600 € hasta versiones más completas que pueden alcanzar precios que rondan los 1000 €.

En el grupo de investigación de Instrumentación, Sensores e Interfaces (ISI) se quiere tener plataformas de ambos sistemas por lo que en este proyecto se tomará como base la estructura del sistema de Draganfly y en un trabajo de investigación paralelo se trabajará con la plataforma Mikrokopter. [6]

CAPITULO 2. ARQUITECTURA DEL SISTEMA QUADROTOR

En este capítulo se presenta la arquitectura de un sistema quadrotor, para ello se ha divido todo el sistema en varias partes. Comenzaremos por una breve introducción a la dinámica de quadrotor para dar un enfoque práctico del funcionamiento de vuelo de dicho sistema. El segundo bloque se compone de las partes mecánicas, desde los motores hasta los sensores pasando por el sistema de procesado de información como podemos ver en la figura 2.1.

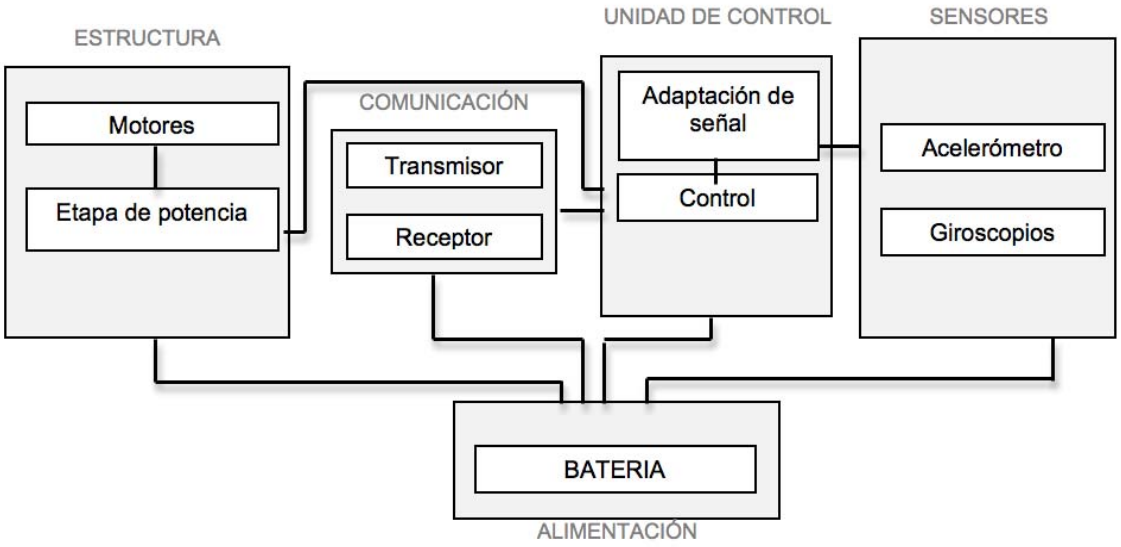

Fig 2.1 Arquitectura genérica del Quadrotor

2.1. Dinámica

La dinámica de un sistema quadrotor se puede explicar a partir del momento de inercia de cada rotor. Cada rotor consta de un motor, un mecanismo de engranaje y un rotor de palas con ángulo fijo. Una de las características más importantes del quadrotor es, que el motor delantero y el motor trasero rotan en sentido anti-horario (2,4), los otros dos motores rotan en sentido horario (1,3) como observamos en la figura 2.2. Con ello, los efectos giroscópicos y los momentos aerodinámicos tienden a cancelarse en vuelo estacionario.

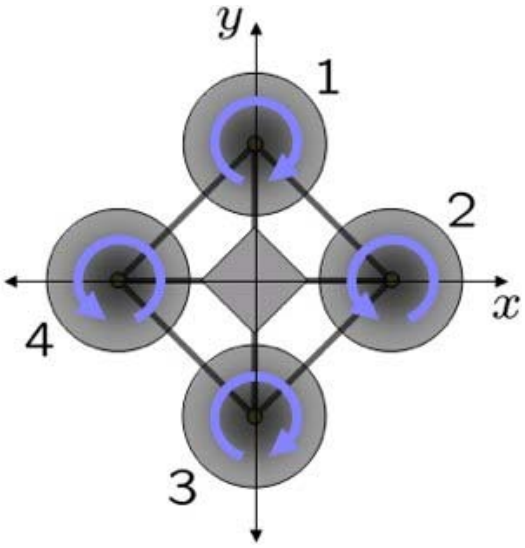

Fig 2.2 Momentos rotores Quadrotor

Para lograr movimiento hacia adelante la velocidad del rotor trasero (4) debe ser aumentada y, simultáneamente, la velocidad del rotor delantero (2) debe ser disminuida. El desplazamiento lateral se ejecuta con el mismo procedimiento, pero usando los rotores de la derecha y de la izquierda. El movimiento de rotación sobre el mismo se obtienen con cada par de rotores, es decir, se acelera los dos rotores con sentido horario mientras se desacelera los rotores con sentido anti-horario, y viceversa.

2.2. Motores

Lo dos motores más utilizados en aeromodelismo son los motores DC y los motores Brushless.

Motores DC

Los motores de corriente continua son muy utilizados en mini robots. Podemos encontrar motores de este tipo en prácticamente todos los coches teledirigidos y/o coches de juguete, los ventiladores de mano y accionamientos varios. En la figura 2.3 podemos observar el aspecto físico y tan conocido de estos motores. El rotor es el dispositivo que gira en el centro del motor DC y está compuesto de arrollados de cable conductores de corriente. Esta corriente es suministrada al rotor por medio de las "escobillas" generalmente fabricadas de carbón.La fuerza con la que el motor gira es proporcional a la corriente que hay por los conductores. A mayor tensión, mayor corriente y mayor par motor.

Fig. 2.3. Motor DC

Motor Brushless

Actualmente se utilizan mucho los motores brushless, o trifásicos. Estos motores son muy superiores a los motores dc en dos aspectos fundamentales: relación potencia-peso (también menor tamaño para la misma potencia) y eficiencia. Esto implica que la cantidad de energía eléctrica que se transforma en energía mecánica es mucho mayor.

La palabra brushless se puede traducir como "sin escobillas", estos motores carecen de colector y escobillas o carbones como podemos ver en la figura 2.4. En vez de funcionar en DC funcionan en AC, la mayoría se alimentan con una señal trifásica, esta señal idealmente debería ser sinusoidal, pero en la practica son pulsos.

Fig 2.4. Motor Brushless

Las ventajas de los motores Brushless son las siguientes:

- Mayor eficiencia (menos perdida por calor)
- Mayor rendimiento (mayor duración de las baterías para la misma potencia)
- Menor peso para la misma potencia
- Requieren menos mantenimiento al no tener escobillas
- Relación velocidad/par motor es casi una constante
- Mayor potencia para el mismo tamaño
- Rango de velocidad elevado al no tener limitación mecánica.

Las desventajas;

- Mayor coste de construcción
- El control es caro y complejo
- Siempre hace falta un control electrónico para que funcione, que a veces duplica el coste.

La tendencia en los sistemas quadrotores es utilizar los motores brushless, ya que ofrecen mayor rendimiento ante las baterías y mayor potencia. Como veremos en el siguiente capítulo para el diseño del quadrotor se ha utilizado motores DC por su sencillez de uso y el precio económico que tienen respecto a los brushless.

2.3. Etapa potencia / Driver

Esta etapa del sistema es muy importante, ya que va a ser la etapa que nos proporcione el control de los motores, es decir, nos va a "traducir" la información de nuestro centro de control / emisora, y va a proporcionar la señal adecuada para el control de los motores. Un aspecto muy importante a tener en cuenta en esta etapa es la corriente que vamos a tener circulando por nuestro sistema.

Según el motor que se utilice deberemos de elegir un diseño u otro. Podemos encontrar bastantes esquemas de control, pero todos se basan en la misma idea pero ofreciendo características diferentes como por ejemplo la potencia máxima que pueden soportar.

Explicaremos la etapa de potencia para motores DC, ya que es la que se ha utilizado en el diseño del quadrotor que veremos en el siguiente capítulo. Podemos encontrar circuitos integrados que nos proporcionen todo lo necesario o podemos optar por circuitos con componentes discretos.
Con los motores brushless existen controladores que proporcionan todo lo necesario para utilizar estos motores.

Para conseguir variaciones de velocidad, ya sea desde nuestra emisora radiocontrol o desde una unidad de control se utiliza habitualmente una modulación por anchura de pulso (PWM).
Dicha señal la podemos generar desde un circuito integrado o lo más usual desde un microcontrolador.

En la figura 2.5 podemos observar la forma de una señal PWM

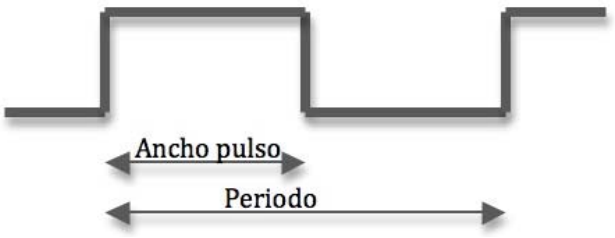

Fig. 2.5. Señal PWM

La señal PWM se utiliza como técnica para controlar circuitos analógicos. El periodo y la frecuencia del tren de pulsos pueden determinar la potencia entregada a dicho circuito. Si, por ejemplo, tenemos un voltaje de 9 V y lo modulamos con un ciclo de trabajo (*duty cycle*) del 10%, obtenemos 0.9 V de media. Para entenderlo mejor, realmente estamos alimentando el motor y apagándolo en períodos de tiempo del orden de milisegundos. Gracias a la inercia del propio motor estos cambios provocan que se mueva a una velocidad constante. Así pues, cuanto mayor sea el ancho del pulso, mayor velocidad obtendremos.

$$\text{DutyCycle} = \frac{\text{AnchoPulso}}{\text{Periodo}} \qquad (\mathbf{2.1})$$

Control Motor DC

Este control nos lo puede proporcionar circuitos integrados como por ejemplo el L298 [7], uno de los más utilizados a la hora de controlar motores DC. Este circuito integrado incluye dos puentes h. Se llama puente h, por la configuración que forman los transistores (figura 2.6)

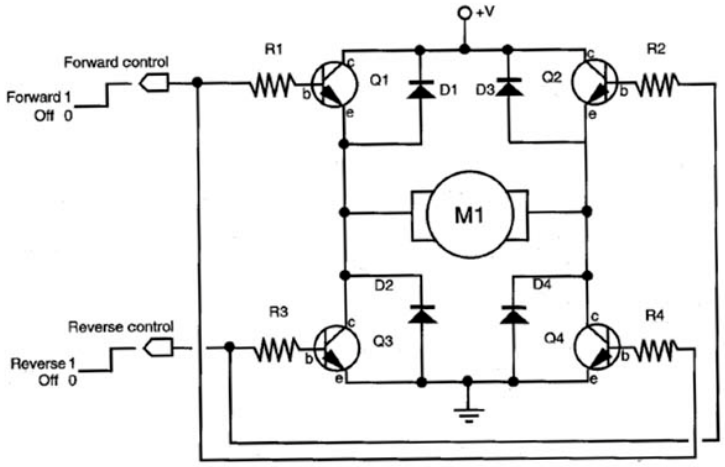

Fig. 2.6. Controlador L298

El modelo L298 soporta hasta un máximo de 4 A, esto puede ser un problema ya que alcanzar ese consumo es relativamente fácil utilizando motores dc de cierto tamaño.

Cuando necesitamos trabajar con corrientes importantes como es nuestro caso (> 4 A) la mejor solución es realizar un circuito con componentes discretos.

Esta solución puede comportar ventajas pero a la vez algunos inconvenientes como por ejemplo tener que diseñar todo el circuito específico, realizar un estudio más exhaustivo de todos los componentes y asumir una probabilidad de fallo más alta en algunos componentes.

2.4. Alimentación

Uno de los principales problemas de los sistemas portables es la alimentación, la solución a este problema es la utilización de almacenadores.

Existen dos clases de almacenadores: el primario (pila), cuya carga no puede renovarse cuando se agota, excepto reponiendo las sustancias químicas de las que está compuesta, y el secundario (baterías), que si es susceptible de reactivarse sometiéndola al paso más o menos prolongado de una corriente eléctrica continua.

La elección de nuestro sistema quadrotor ha sido el utilizar baterías, ya que al tener consumos considerables (> 4 A) nuestro almacenador debe de ser capaz de proporcionar picos de corrientes y una duración adecuada. La pila se utiliza cuando tenemos consumos pequeños.

Para proporcionar un rápida aceleración y duraciones considerables, las baterías de cualquier sistema autónomo tiene que proporcionar una gran potencia y energía. Además, han de ser de bajo precio, fáciles de mantener, seguras bajo condiciones de funcionamiento externo y tolerantes a los abusos a los que los vehículos aéreos están expuestos (aire libre, caídas, etc.).

La capacidad de una batería es la cantidad total de carga producida en la reacción electroquímica y se define en unidades de coulombs (C) o amperios-hora (Ah), que es la más usada como especificación de las baterías.

Los parámetros a considerar de una batería son los siguientes:

- La tensión de salida, medida en voltios.
- La capacidad eléctrica, se mide por referencia a los tiempos de carga o de descarga en amperios hora (Ah) En las baterías se utiliza un múltiplo ,el miliamperio hora (mAh). Este parámetro es muy importante y debemos fijarnos en él cuando compremos la batería pues cuanto mayor sea, más tiempo tardará en descargarse por el uso. El precio esta en relación directa con este parámetro.

A continuación veremos los diferentes tipos de baterías que se utilizan en los equipos autónomos o de radio control.

Ni-Cd (baterías de níquel-cadmio)

Durante mucho tiempo las baterías de níquel-cadmio fueron la única opción para propulsar a los vehículos hasta que en 1990 aparecieron la baterías de níquel-metal-hidruro y las de ion-litio. El voltaje de una célula de Ni-Cd es de 1.2 V. Poseen efecto memoria. El efecto memoria se produce cuando cargamos las baterías sin haberlas previamente descargado por completo, se crean unos cristales en el interior de dichas baterías que hace que no se carguen en toda su capacidad. Para prevenir el efecto memoria basta con realizar de vez en cuando una descarga-carga completa. Otro inconveniente es que el cadmio es muy tóxico.

Ni-MH (baterías de níquel-metal-hidruro)

En estas baterías, debido a la alta toxicidad del cadmio, éste se sustituye por hidruros metálicos.Tienen mayor densidad de carga (capacidad), no contienen Cd (menos contaminantes), no poseen efectos de pérdida de capacidad por mal uso (efecto memoria). Soportan un menor número de cargas durante su vida útil que las de Ni-Cd. Tienen una resistencia interna superior lo que las limita para alimentar motores de alta potencia. Aceptan cargas rápidas.

Li-ion (baterías de iones de litio)

La capacidad de una batería de Ion-Litio es aproximadamente el doble de la capacidad de una batería de Níquel-Cadmio. El litio es el metal más ligero que existe por lo que a igualdad de capacidad las baterías resultan mucho más ligeras. El voltaje de una célula de Ion-Litio es de 3.7 V. No tienen mantenimiento, no poseen efecto memoria. Tienen como desventaja que requieren de un circuito de control que se emplean para limitar el voltaje máximo de cada célula de la batería.,

Li-Po (baterías de polímero de litio)

Son una variación de las Baterías Litio-Ion (Li-ion. Tienen una densidad de energía entre 5 y 12 veces las de Ni-Cd o las de Ni-MH. Son baterías mucho más ligeras y que pueden adoptar cualquier forma. No poseen efecto memoria. El voltaje de cada elemento es de 3.7 V. La desventaja es que necesitan una carga mucho más lenta que las de Ni-Mh. Este tipo de baterías son ideales para alimentar motores muy potentes.

Fig 2.7. Batería Li-Po

Se debe cuidar de que el consumo máximo del motor sea menor que la descarga máxima de la batería para evitar que la vida de ésta se acorte demasiado. Su reducido tamaño y ligereza también las hace ideales para aeromodelos y helicópteros como se observa en la figura 2.9. En la tabla 2.1 podemos observar una comparativa de los 4 modelos estudiados.

Tabla 2.1 Comparativa baterías

	Voltaje por célula	Efecto memória	Resistencia interna	Cargas vida útil	Velocidad de carga
Ni-Cd	1.25	Si	*	**	Lenta
Ni-MH	1.25	No	**	*	Rápida
Li-ion	3.7	No	***	***	Rápida
Li-Po	3.3	No	****	****	Lenta

En el diseño del quadrotor se han utilizado unas baterías tipo Li-Po igual que utiliza el UAV comercial Draganflyer, se ha elegido este tipo de batería principalmente por la gran densidad de carga que tienen en poco espacio y peso. Las mejores baterías desde el punto de vista capacidad / peso, muy importante en vehículos aéreos, son las de zinc-aire, pero aún no están comercializadas, aunque el ejército Israelí ya las utiliza en sus UAV. Actualmente se pueden encontrar baterías primarias, para aplicaciones sobretodo en audífonos, pero no secundarías.

2.5. Unidad de control

La unidad de control es un elemento de los sistemas quadrotores bastante peculiar. Como todo sistema aéreo debe de disponer de unos sensores para medir parámetros como la aceleración de los ejes o la velocidad angular, esto ayuda al sistema a proporcionar la estabilización del mismo. La función más compleja de la unidad de control es la estabilización del sistema, aunque debe de realizar otras funciones como el almacenamiento de datos entre otras.

Las funciones principales de una unidad de control son las siguientes:

- Estabilizar el sistema mediante los sensores.
- Proporcionar comunicación estación base.
- Interpretar comandos de control enviados desde la estación base.
- Guardar / enviar datos recogidos de los sensores.

2.5.1. Microcontrolador

Los microprocesadores y los microcontroladores son los dispositivos programables más importantes de los sistemas digitales, los cuales son la base para el control de procesos en la industria, adquisición y procesado de datos.

Todo desarrollo tecnológico moderno sustentado en la microelectrónica, que posee una unidad central de control, comprende a un microprocesador como el corazón mismo de dicha unidad; o bien a un dispositivo programable simplificado llamado microcontrolador. También podemos encontrarnos con sistemas más avanzados como DSP, FPGA o PSOC.

Un microcontrolador es un tipo de computadora en miniatura que podemos encontrar en pequeños dispositivos electrónicos (radios, teléfonos, etc..).
No debemos de confundir los microcontroladores con los microprocesadores, ya que estos últimos sólo es un unidad central de procesos, que de manera externa, hay que conectarle la memoria, puertos y unidades de almacenamiento, tiene la ventaja de ser más escalable que un microcontrolador, lo que nos da la capacidad de poder agregarle más componentes según sea nuestra necesidad.

Para el control de un sistema quadrotor es suficiente utilizar un microcontrolador, la utilización de los conversores A/D , los contadores y la pequeña memoria (RAM y ROM) que incorporan hacen que estos dispositivos sean los mas utilizados en este ámbito. En el diseño del quadrotor se ha utilizado un microcontrolador del fabricante ATMEL utilizando los conversores A/D de 10 bits y las 6 salidas PWM .

2.5.2. Sensores

Un sensor es todo dispositivo que, a partir de la energía del medio donde se mide (temperatura, altitud, inclinación, etc..), genera una señal de salida que podemos interpretar mediante algún parámetro que se modifica en función a la variable medida.

En un sistema quadrotor podemos encontrar un gran número de sensores enfocados a la estabilización del sistema y a la navegación. Algunos de los sensores más utilizados en estos sistemas son los acelerómetros y giroscopios, aunque en algunos sistemas más avanzados, también podemos encontrar sensores de presión. Por otra parte, tenemos los sensores o equipos utilizados para la navegación, que van desde sensores de proximidad hasta receptores GPS.

A continuación se describirán los sensores utilizados como los giroscopios y acelerómetros ya que para realizar el control de la estabilización son esenciales.

Giroscopio

Foucault en 1851, poco después de su famosa experiencia del péndulo en el Panteón de París intentó demostrar la rotación de la Tierra mediante otro artefacto: el giróscopo. Aunque el experimento no contó con tanto éxito, el nombre de giróscopo está aún vigente (gyros, rotación; scopos, verse o poner de manifiesto), es decir aparato para mostrar la rotación.

Un giróscopo o giroscopio es un sólido rígido en rotación alrededor de un eje principal de inercia como vemos en la figura 2.8. Habitualmente se monta sobre anillas en suspensión. De esta manera, ningún movimiento que realizase el conjunto causaría momento externo. Por tanto, el momento angular o momento cinético se conservaría y el eje de rotación mantendría una dirección fija en el espacio.

Fig. 2.8 Giroscopio.

Los movimientos giroscópicos han tenido un gran número de aplicaciones. La tendencia a mantener fija la orientación en el espacio del eje de rotación del giróscopo se emplea para estabilizar barcos, en los sistemas de navegación automática de los aviones, en el sistema de dirección de torpedos y misiles, etc.

Si bien existen giróscopos cuyo funcionamiento se basa en un elemento mecánico, la realidad es que la mayoría de los sensores actuales de pequeño tamaño, como los que se utilizan en modelos de helicópteros y robots, están basados en integrados con pequeñas lengüetas vibratorias, construidas directamente sobre el chip de silicio. Su detección se basa en que las piezas cerámicas en vibración están sujetas a una distorsión que se produce por el efecto Coriolis (cambios de velocidad angular). Como resultado de esta deformación el giroscopio genera un voltaje de salida que es proporcional a esta velocidad angular de rotación.

Estos sensores son pequeños, un ejemplo es el ADXRS150 de Analog Devices, con un tamaño de 7 x 7 x 3,2 mm. En el proyecto se ha utilizado el Gyrostar (ENC-03), fabricado por Murata [8], con una cápsula que mide 12,2 x 7 x 2,6 mm como vemos en la figura 2.9.

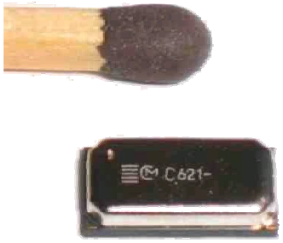

Fig.2.9 Giroscopio Murata.

Acelerómetro

Las técnicas convencionales para detectar y medir aceleraciones se basan en el principio descubierto por Isaac Newton y que se encuentran descritos en su Principio de Newton (1687).

Este principio es también conocido como la segunda ley de Newton la cual nos dice: “que la fuerza neta aplicada sobre un cuerpo es proporcional a la aceleración que adquiere dicho cuerpo”.

Los sensores utilizados para medir aceleraciones son los denominados acelerómetros. Estos son sensores inerciales que miden la segunda derivada de la posición. Por tanto un acelerómetro mide la fuerza de inercia generada cuando una masa u objeto es afectado por un cambio de velocidad.

Existen diferentes tipos de acelerómetros basados en varios tipos de tecnologías como son [9] :

Tecnología capacitiva: se basan en variar la capacidad entre dos o más conductores entre los cuales existe siempre un material dieléctrico.

Tecnología piezoresistiva: Estos dispositivos basan su funcionamiento en la propiedad que tienen las resistencias eléctricas de cambiar su valor cuando el material se deforma mecánicamente

Tecnología piezoeléctrica: Al sufrir una deformación física del material se produce un cambio en la estructura cristalina y por consecuencia cambian sus características eléctricas.

Tecnología mecánica: Estos acelerómetros utilizan bobinas, imanes para medir aceleración. Los componentes utilizados son una masa y resortes.

Tecnología micromecanizada: Se denomina MEMS (Sistemas Micro Electro-Mecánicos) o microsistemas electromecánicos a una tecnología de base que se utiliza para crear dispositivos diminutos. Este tamaño puede oscilar en pocas micras pudiendo llegar hasta un milímetro de diámetro.
En los sistemas quadrotores una de las características importantes es el peso de todo el sistema, por ello se utilizan los acelerómetros con tecnología MEMS. Un ejemplo de acelerómetros MENS muy conocida son los acelerómetros de la familia ADXL de Analog Devices.[10]

Los acelerómetros de esta familia pueden ser utilizados en una gran variedad de aplicaciones de bajas "g", entre ellas para mostrar la inclinación u orientación, vibración del sistema y detección de movimientos. Estos sensores están formados por una superficie micromecanizada, formada por un pequeño sensor (aprox. 1 mm^2) de aceleración en un circuito integrado de silicio. Utilizando los mismos pasos que para hacer circuitos electrónicos convencionales, la superficie micromecanizada permite crear estructuras libres de movimiento cerca de la superficie de silicio.

En sistemas avanzados de quadrotores podemos encontrar sensores de presión, dichos sensores se emplean para limitar la altura del vehículo, ya sea para corregir un posible fallo o para limitar alguna acción intencionada. Otro sensor muy utilizado son los sensores de temperatura, este sensor se utiliza para corregir los pequeños errores que puedan producir los cambios de temperatura en los giroscopios, la desviación puede ser de mayor o menor importancia según el modelo utilizado de giroscopio. En cualquier sistema en que se utilicen motores, hemos de tener monitorizado la temperatura de dichos motores. El sensor de temperatura nos puede ayudar a detectar posibles excesos de temperatura y así poder avisar al usuario o en casos extremos realizar alguna maniobra de aterrizaje y evitar el "accidente" del sistema.

Al tratarse de un sistema aéreo en todos los casos, incorporan sensores más o menos complejos que ayudan a la navegación. Podemos encontrarnos sensores de proximidad hasta receptores GPS, muchos de estos sistemas aéreos pueden programarse para que funcionen de forma autónoma gracias a estos sensores.

Una de las aplicaciones más utilizadas en los sistemas quadrotores es la captura de imágenes aéreas, esta línea de aplicación se está investigando gracias al gran avance de las cámaras de video o fotografía. Podemos encontrar usuarios que incorporan a sus sistemas aéreos cámaras más sencillas para obtener una visión en directo de la imagen o usuarios que incorporan cámaras más sofisticadas que pueden proporcionar un video en alta definición HD.

En la figura 2.10 podemos ver las diferencias entre dos plataformas que utilizan una cámara muy sencilla (derecha) o utilizar una cámara más sofisticada (izquierda).

Fig. 2.10 Comparativa entre plataformas que incorporan cámaras.

2.5.3. Sistema de comunicación

El control más utilizado en estos sistemas son las emisoras de radio control. La gran ventaja de utilizar emisoras de radio es la expansión que ha tenido durante mucho años en el mundo de aeromodelismo.

La principal limitación y causa de problemas es el uso de la misma frecuencia por más de un usuario. La segunda causa de problemas es la falta de estabilidad de emisoras y receptores, a los que pueden influir canales adyacentes. Utilizando buenos receptores este problema se minimiza mucho, pero si una emisora adyacente emite con mucha potencia y con bastante desviación del canal, entonces terminará influyendo y cuando el vehículo se aleje de nuestra emisora puede tener serios problemas. Esto es una tecnología muy simple y antigua, comercializada hace varias décadas buscando la simplicidad para poder limitar el precio, tamaño y peso de los receptores. En ningún momento se esperaba que un microprocesador fuera a estar involucrado así que no tenía sentido hacerla más compleja.

El avance en las comunicaciones inalámbricas abre un gran abanico en lo que se refiere a sistemas de comunicación. Podemos encontrar codificaciones muy eficientes y que consigan velocidades de transmisión importantes. Al tratarse de vehículos aéreos tendremos que tener en cuenta la distancia máxima que pueden alcanzar con las diferentes codificaciones. Podemos encontrar protocolos como Zigbee que proporcionen una distancia de comunicación en campo abierto de 1 km.

CAPITULO 3. Diseño Sistema Quadrotor

En este capitulo lo que se expone es el diseño propio de un sistema quadrotor. El resultado final llevará el nombre de **FLISI1** cómo la primera versión de un conjunto de sistemas aéreos, diseñados por el grupo de investigación ISI.

FLISI1 es un sistema quadrotor que utiliza la estructura de fibra de carbono del quadrotor comercial Draganflyer V. El sistema de propulsión se consigue a través de cuatro motores dc y está equipado con un acelerómetro de tres ejes y tres giroscopios que ayudan a la estabilización de todo el sistema. El procesado de todos los datos se realiza mediante la plataforma Arduino.

A continuación se va a describir de forma detallada, todas las partes que componen el **FLISIS1** y los resultados obtenidos a partir de las pruebas realizadas sobre él. En la figura 3.1 podemos observar el escenario inicial utilizado en el proyecto.

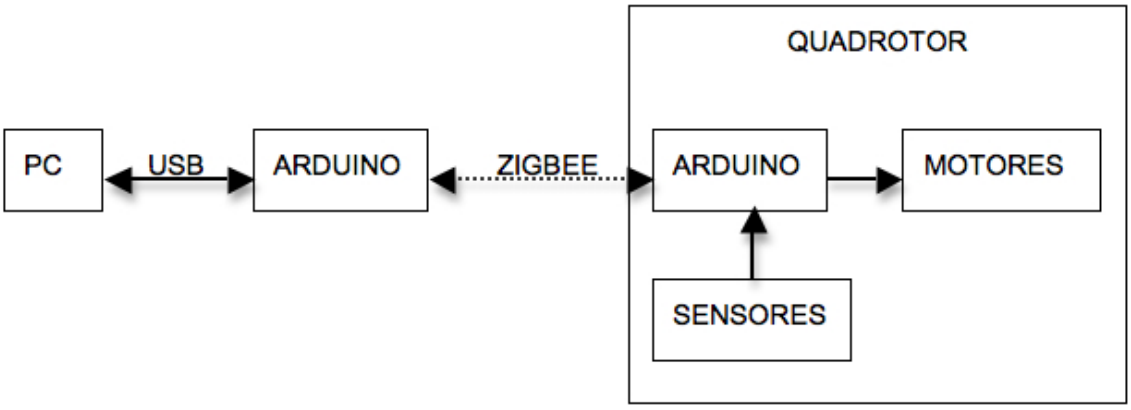

Fig 3.1. Escenario inicial controlado por PC.

Se comenzó utilizando el control del quadrotor mediante PC porque proporciona sencillez a la hora de programar Arduino y ayuda a realizar las pruebas iniciales de comunicación. Una vez programado el quadrotor y definida toda la comunicación él siguiente paso fue controlar el vehículo mediante Nunchuk (mando inalámbrico de la Wii) y se pasó al escenario que se muestra en la figura 3.2.

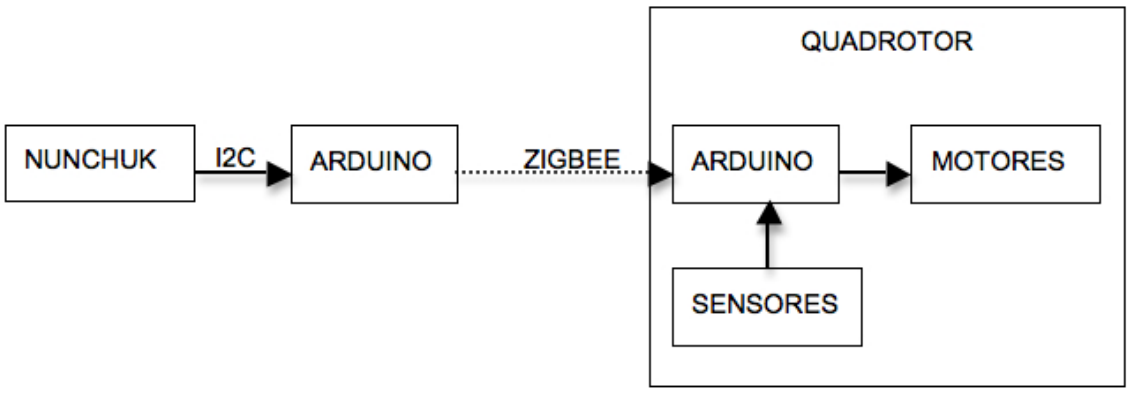

Fig 3.2. Escenario final controlado por Nunchuk.

3.1. Estructura

La estructura es una de las primeras partes a elegir cuando realizamos el diseño quadrotor. En este caso se ha elegido la misma estructura que posee el quadrotor comercial Draganflyer V. Esta estructura es de fibra de carbono con un peso de 123 gr y con unas dimensiones de 42 cm x 42 cm como se observa en la figura 3.3.

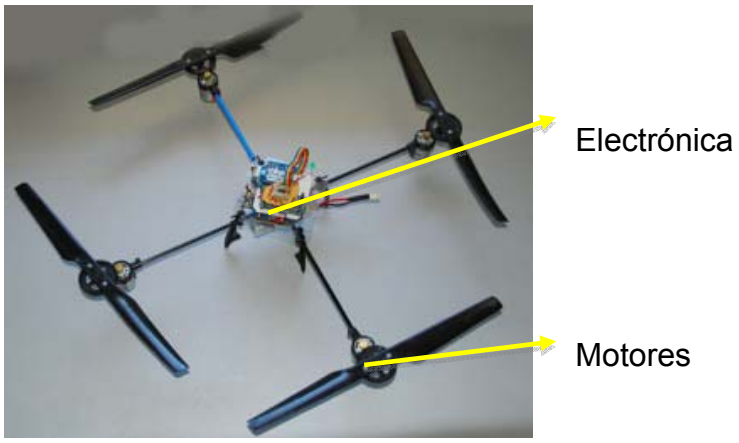

Fig 3.3. Estructura FLISI1.

La estructura se compone de dos ejes perpendiculares formando una cruz simétrica. En su extremos incorpora unos pequeños soportes para los motores. Toda la electrónica y batería está colocado en el centro de la estructura para ayudar a su ensamblaje.

3.2. Motores

Los motores escogidos son motores dc de la casa Mabuchi Motor con las siguientes características [11]:

- Modelo RC 280SA
- Voltaje de trabajo 4.5 – 12 V
- Consumo sin carga 0.14 A
- Consumo medio 4 A
- Peso 47 gr

En la siguiente figura 3.4 podemos observar la caracterización de uno de estos motores [12]. Se observa la relación de voltaje de entrada en función de la corriente que consume y el peso que es capaz de elevar. La línea de color rojo es una aproximación lineal de los valores obtenidos (color azul).

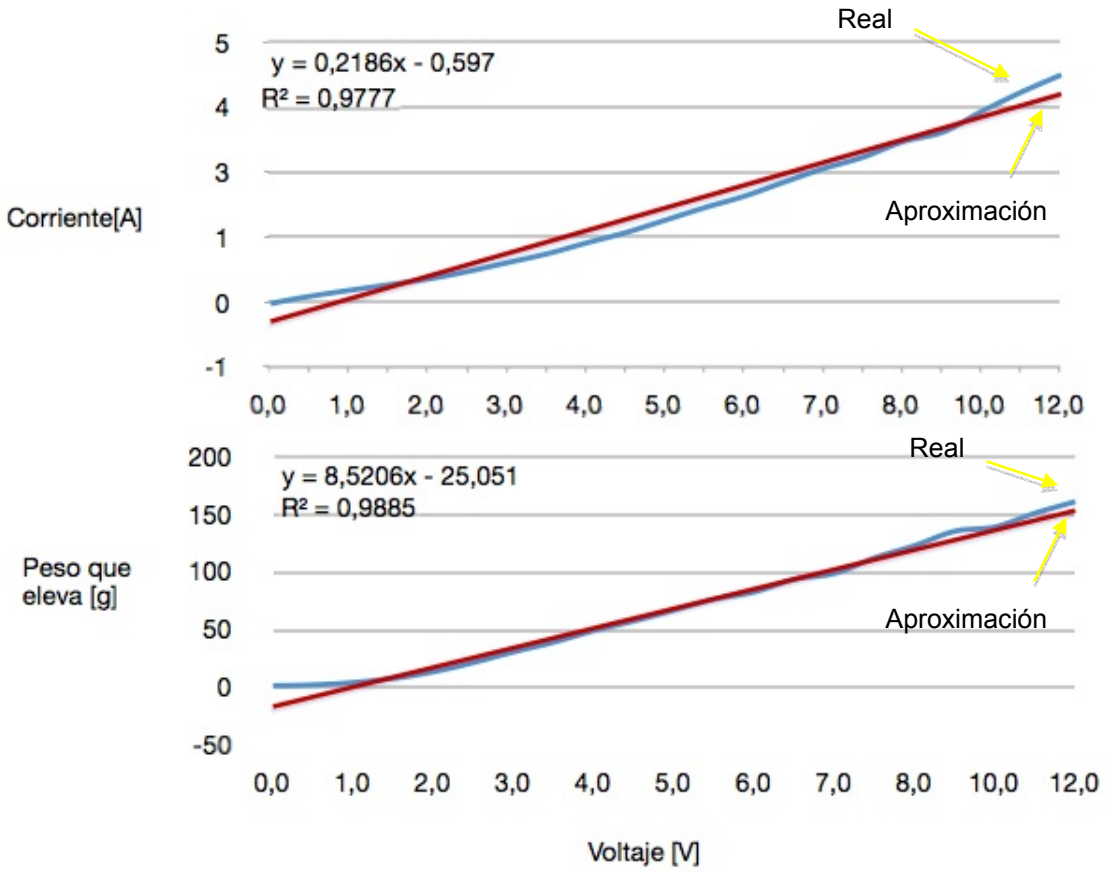

Fig 3.4. Caracterización motor: consumo y peso que eleva en función de la alimentación.

Para realizar estas gráficas se sujetó un motor a un báscula de precisión, se invirtió el aspa y gradualmente se le fue aumentando el voltaje aplicado a la entrada del motor. El peso que eleva el motor es el peso del motor menos lo que está midiendo en ese momento la báscula.

Podemos observar como a 11 V (máxima tensión que se va a suministrar a nuestro motor) tenemos un consumo aproximadamente de 4 A y una capacidad de elevar una carga equivalente a 140 gr.

Hay que tener en cuenta que este estudio se ha realizado únicamente a un motor. Como primera aproximación podemos suponer que tanto el consumo como la capacidad de elevar cierta carga la multiplicaremos por 4, número de motores que lleva equipado el quadrotor.

Con esta aproximación podemos decir que el quadrotor consumirá 16 A referentes al sistema de propulsión (motores) y tendrá una capacidad de elevar 560 gr.

Los motores van equipados de un sistema de engranajes para dar más fuerza a las aspas. En la figura 3.5 se puede observar el sistema de engranajes del que va equipado el sistema.

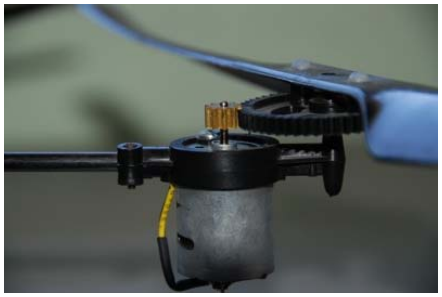

Fig. 3.5 Engranajes.

Una parte muy importante, son las características de las aspas utilizadas. Existen muchos diseños de aspas y dimensiones a utilizar. En este caso se utilizan unas aspas de plástico de dimensiones 18 cm de longitud, al estar hechas de dicho material ayudan a alargar la vida útil de ellas.

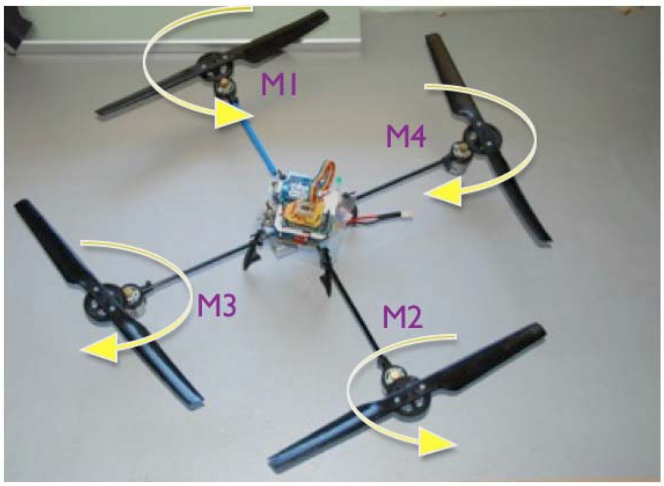

Fig. 3.6 Sentido aspas FLISI1.

En la figura 3.6 podemos ver el sentido de las aspas. Para lograr movimiento hacia adelante la velocidad del rotor trasero (M2) debe ser aumentada y, simultáneamente, la velocidad del rotor delantero (M1) debe ser disminuida. El desplazamiento lateral se ejecuta con el mismo procedimiento, pero usando los rotores de la derecha y de la izquierda.
El movimiento de rotación sobre el mismo se obtienen con cada par de rotores, es decir, se acelera los dos rotores con sentido horario (M3,M4) mientras se desacelera los rotores con sentido anti-horario (M1,M2), y viceversa. Lo que conseguimos con cada par de motores es cancelar el movimiento sobre él, ya que cada par de motores como se ha comentado anteriormente rota en sentido contrario al otro par.

3.3. Etapa potencia

La etapa de potencia elegida es un diseño propio construido a través de componentes discretos.

Para el diseño de la etapa de potencia, la principal característica que se ha buscado es que soporte los 4 A por motor. Este problema se ha solucionado eligiendo componentes discretos que soporten dicho amperaje con soltura.

La estrategia de diseño que se ha empleado es elegir transistores mosfet, dichos transistores nos ofrecen las siguientes ventajas:

- Consumo en modo estático muy bajo.
- Tamaño muy inferior al transistor bipolar (actualmente del orden de media micra).
- Gran capacidad de integración debido a su reducido tamaño.
- Funcionamiento por tensión, son controlados por voltaje por lo que tienen una impedancia de entrada muy alta. La intensidad que circula por la puerta es del orden de los nanoamperios.
- La velocidad de conmutación es muy alta, siendo del orden de los nanosegundos.

Como podemos ver, las características de los mosfet los hacen adecuados para ser utilizados en nuestro sistema, ya que necesitamos velocidades de conmutación rápidas y que el consumo no sea excesivo.

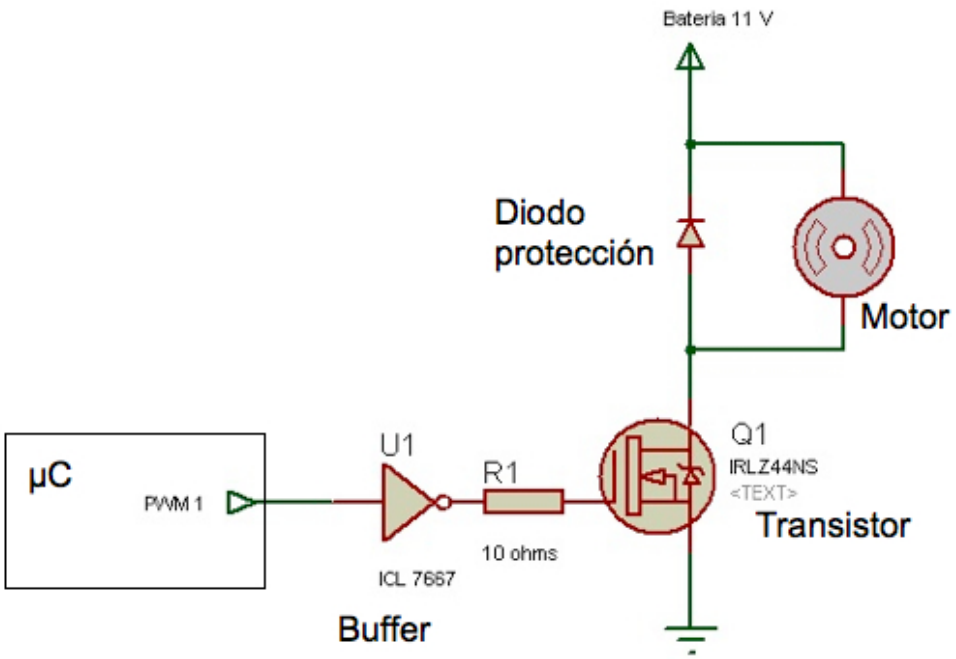

Fig. 3.7 Esquema control motor.

En la figura 3.7 vemos el diseño que se ha utilizado, es simple de analizar, ya que la función principal del transistor es abrir y cerrar el circuito del motor, para que por este circule corriente o no. Lo que se realiza es colocar la señal PWM a la entrada del transistor. El transistor al ser bastante rápido conmutará la señal, provocando conmutaciones del circuito cerrado del motor según el estado alto del la señal PWM. Lo que conseguimos con este esquema es que el motor vea una señal continua (media de la señal PWM) y variando el duty cycle variaremos dicha señal que recibe el motor. En la figura 3.8 podemos ver la variación en amplitud de la señal equivalente que verá el motor en función de las variaciones de la señal PWM generada por el microcontrolador.

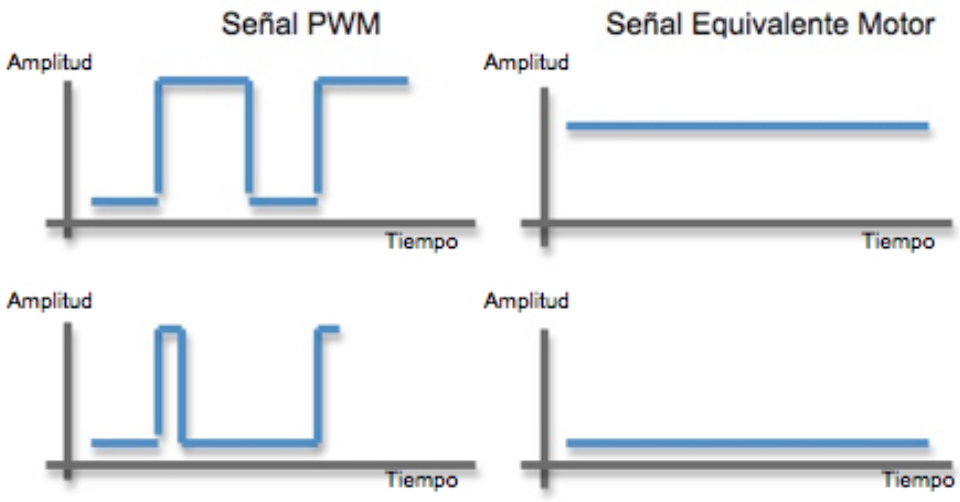

Fig.3.8 Señal PWM / Señal equivalente que llega al motor.

El transistor mosfet elegido es el IRLZ44NS [13], cuyas características más relevantes son las siguientes:

- V_{DSS}: 55 V
- R_{DS} : 0.022 Ω
- ID : 47A

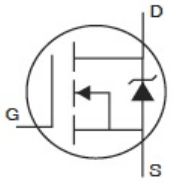

Fig.3.9 Esquema transistor.

Una característica a destacar de este transistor es la R_{DS} (resistencia entre Drain y Source) que es muy pequeña. En la figura 3.9 podemos observar el esquema proporcionado por el fabricante y llegar a la conclusión que la R_{DS} estará en paralelo con el motor. Con esta resistencia y el consumo del motor podemos calcular la caída de tensión máxima que tendremos en el transistor cuando apliquemos los 11 V a los motores.

$$V = R * I \qquad (3.1)$$

$$V = 0.022[\Omega] * 4\,[A] = \mathbf{88\ mV}$$

Mediante la formula 3.1 obtenemos el cálculo teórico de la caída de tensión en el transistor, el resultado es bastante bueno ,la caída de tensión es insignificante gracias a la pequeña resistencia, lo que ayuda a aprovechar toda la tensión de la batería para el motor.

La señal PWM que proviene del microcontrolador no se introduce directamente en la entrada del transistor, si no, como vemos en la figura 3.7, pasa por un buffer que nos ayuda a proporcionar una señal con una corriente suficientemente grande para realizar la conmutación del transistor. Otra función del buffer es proteger el microcontrolador de las corrientes inversas. Al utilizar componentes con bobinas (motores) debemos tener cierto cuidado con las corriente inversas que se puedan generar. El buffer utilizado es el ICL7667 [14] que puede alimentarse hasta 15 V y es suficiente ya que nuestras baterías son de 11 V.

El diodo que se observa en la figura 3.7 es para proteger al circuito, no al motor. Este diodo se le conoce como diodo de rodada libre. Está inversamente polarizado porque al conmutar un circuito con una inductancia, la corriente se dice que "se invierte" en la bobina, regresando esa corriente al circuito. Si tenemos el diodo, esta corriente lo polariza directamente y se descarga a través de él, sin afectar al resto del circuito.

El esquema de la figura 3.7 hay que utilizarlo para cada motor, el quadrotor utiliza 4 motores por lo que se duplicará el mismo esquema en cada motor.

Todo el esquema e ha diseñado en una placa de circuito impreso para facilitar su uso (anexo 1). Se ha añadido un condensador a la entrada de la batería para filtrar posibles fluctuaciones y dos interruptores para controlar el suministro de alimentación a la etapa de potencia y al microcontrolador. Estos interruptores los podemos identificar en la figura 3.10.

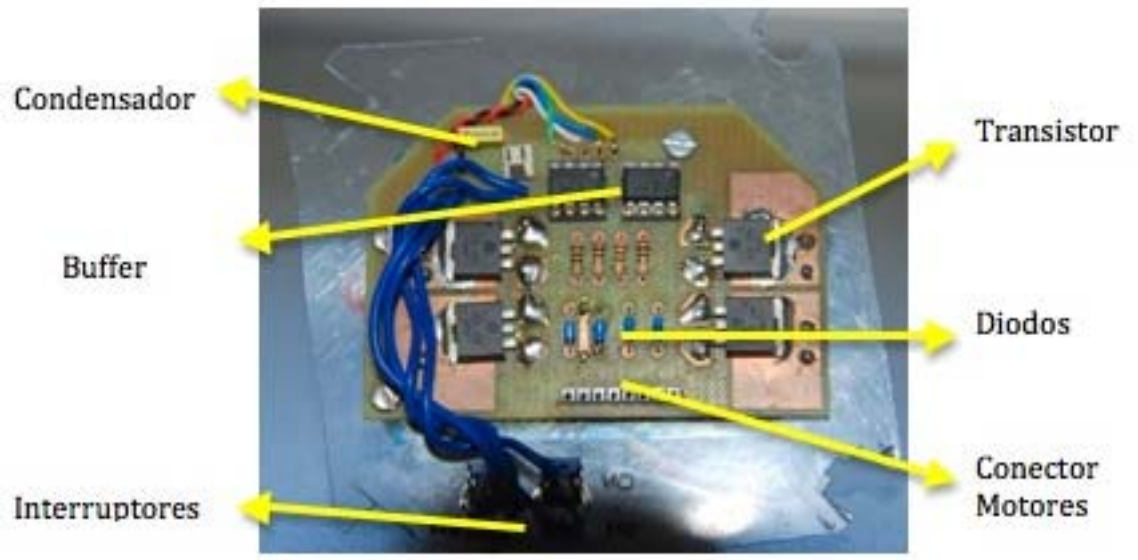

Fig 3.10 Circuito impreso donde se recogen los elementos de la etapa de potencia del sistema quadrotor.

3.4. Batería

Como se ha visto en el capítulo 2, una de las baterías más utilizadas en los sistemas aéreos o móviles, son las baterías Li-Po. Estas baterías permiten una mayor densidad de energía, así como una tasa de descarga bastante superior. Estas baterías tienen un tamaño más reducido respecto a las de otros componentes.

Casi todos los aeromodelistas utilizan baterías Li-po, aunque hay que tener en cuenta que pueden ser peligrosas si se utilizan mal, pudiendo provocar alguna explosión.

En el quadrotor se han utilizado dos baterías Li-po de capacidades diferentes, una de 1300 mAh y otra de 1800 mAh. Se han elegido estas capacidades porque ofrecen la relación óptima entre peso y duración.

Más adelante se realizará un cálculo aproximado del tiempo de duración de las baterías en el **FLISI1**.

Batería 1300 mAh

Las características principales de esta batería son las siguientes [15]:

- Marca: Dualsky
- Tensión: 11.1 V
- Capacidad: 1300 mAh
- Amperaje máx. continuo: 32,5 A
- Peso : 109 gr

Fig 3.11 Dualsky 1300 mAh.

Batería 1800 mAh

Las características principales de esta batería son las siguientes [16]:

- Marca: FlightPower
- Tensión: 11.1 V
- Capacidad: 1800 mAh
- Amperaje máx. continuo: 50,4 A
- Peso: 146 gr

Fig 3.12 FlightPower 1800 mAh.

En las figuras 3.11 y 3.12 podemos ver el aspecto físico que adquieren estas baterías. El funcionamiento de estas con el quadrotor es idéntico y las únicas diferencias que existe entre ellas es el peso y la duración. Esta diferencia se debe básicamente a que tienen capacidades (mAh) diferentes. Se han utilizado dos baterías para poder realizar las pruebas mientras se esta cargando la otra batería.

3.5. Unidad de control

A causa de las no linealidades de los elementos, ya sean eléctricos o físicos tendremos que realizar una tarea de estabilización. La estabilización se llevará acabo desde una unidad de control equipada con un microprocesador y diferentes sensores.

3.5.1. Microprocesador

Se ha elegido utilizar la plataforma Arduino para el diseño del quadrotor porque proporciona una facilidad de uso muy grande y la programación que exige es bastante sencilla e intuitiva. Una de las grandes ventajas de la plataforma Arduino es el precio asequible que tiene. A diferencia con otros microcontroladores esta plataforma facilita la configuración del microcontrolador gracias a unas librerías que podemos encontrar en la pagina oficial del Arduino.

Arduino sigue la línea de código abierto pero en su versión hardware, esto quiere decir que toda la comunidad tiene acceso a toda la plataforma de manera libre.

Arduino está basada en una sencilla placa con entradas y salidas analógicas y digitales, en un entorno de desarrollo que implementa el lenguaje de programación Processing. Está basado en el microcontrolador ATMEL, un chip sencillo y de bajo coste que permite el desarrollo de múltiplos diseños. Según ha ido avanzando el tiempo, el procesador se ha ido actualizando hasta llegar al modelo Atmega1280 con mejores prestaciones, como por ejemplo más memoria flash (128 kb) y 16 puertos de entrada y salida.[17]

Al ser hardware libre, tanto su diseño como su distribución es libre, es decir, puede utilizarse libremente para el desarrollo de cualquier tipo de proyecto sin haber adquirido ninguna licencia.

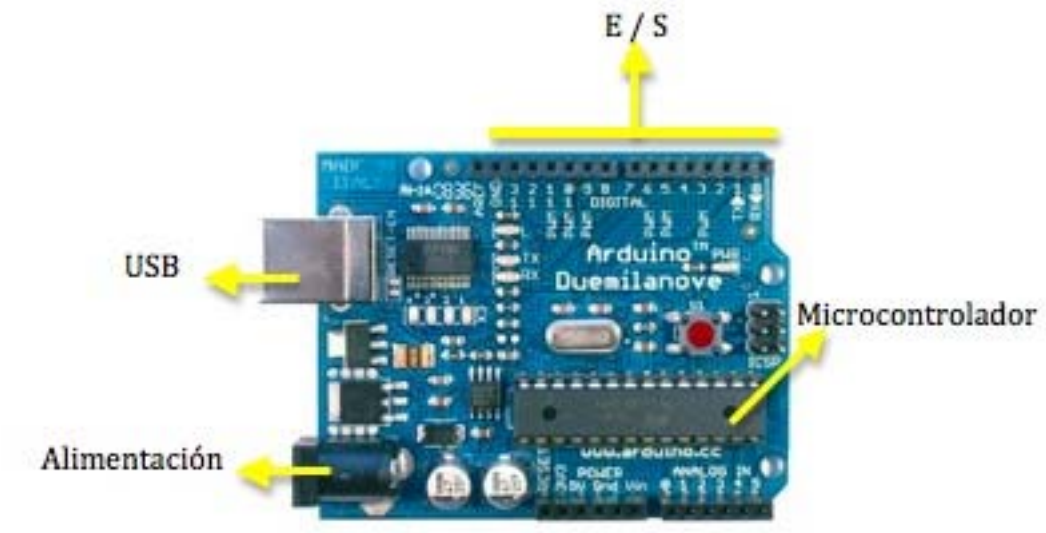

Fig 3.13 Arduino Duemilanove.

Una de las peculiaridades de la plataforma Arduino es que poseen placas que ayudan a la configuración del microcontrolador. En al figura 3.13 tenemos un ejemplo de una de las placas más sencillas y más utilizada por los usuarios, esta placa incorpora un conector USB ya instalado, un regulador de tensión, un conversor puerto serie / USB , sistemas de protección y una fácil accesibilidad a todos los puertos del microcontrolador.

Actualmente existe una gran variedad de placas diferentes, nos podemos encontrar con placas que incorporan tecnología bluetooth hasta placas llamadas nano que tienen unas dimensiones muy reducidas.

Se ha elegido una Arduino Duemilanove y no otra porque esta ofrece una gran compatibilidad con todos los módulos que existen en el mercado enfocados a Arduino, además es de bajo coste y fácil de programar.

En la tabla 3.1 se muestran características de la placa utilizada.

Tabla 3.1 Características Arduino Duemilanove

Microcontrolador	ATmega328
Voltaje (recomendado)	7-12V
Digital E/S	14 (6 salidas PWM)
Entradas analógicas	6
Corriente por I/O	40 mA
Memoria Flash	32 kB con 2 kB para el bootloader
SRAM	2 kB
EEPROM	1 kB
Velocidad de reloj	16 MHz

Este microcontrolador nos proporciona 6 entradas analógicas con conversores A/D de 10 bit de resolución y 6 salidas PWM. Dicha placa será utiliza para leer la información de los sensores y procesarla para el posterior control de los motores. La facilidad de utilizar esta placa con otros módulos es una sus características más importantes, ya que podemos encontrar una gran variedad de módulos (gps, comunicación inalámbrica ,control motores, comunicación Ethernet, etc.)

3.5.2. Sensores

Los sensores utilizados por el sistema quadrotor son 3 giroscopios y un acelerómetro de 3 ejes. Estos son los sensores básicos para conseguir una primera estabilización.

El acelerómetro ADXL330 del fabricante Analog Devices es un dispositivo de aceleración capacitivo de tres ejes (X, Y, Z) creado sobre un único circuito integrado “IC” y cuyas dimensiones son de 4 mm x 4 mm x 1.45 mm como se observa en la figura 3.14.

Este sensor es capaz de medir aceleraciones de ± 3 g y cambios de aceleración de hasta 10.000 g. Esto se debe a que en la mayoría de aplicaciones estos dispositivos son utilizados como medidores de desaceleraciones.

Fig. 3.14 Acelerómetro ADXL330 de Analog Devices

Según pruebas realizadas en el laboratorio, este acelerómetro se puede utilizar como inclinómetro, ya que la ecuación 3.2 que rige el funcionamiento del sistema nos dice que el valor de tensión en cada una de las salidas será igual a la mitad del valor de alimentación (offset), más un término que dependerá de la aceleración registrada.

$$V_{out}(x, y, z) = \frac{V_s}{2} \pm (\text{f.a}) \qquad (\mathbf{3.2})$$

Donde:

- V_s es la tensión de alimentación que en este caso es de 3.3 V.
- f.a es el factor de aceleración (sensibilidad * aceleración).

Este factor de aceleración podrá sumar o restar si la aceleración detectada es positiva o negativa. La sensibilidad del acelerómetro es de 300 mV / g, lo que quiere decir si tenemos una aceleración registrada de (±1 g) obtendremos una variación que la salida que oscilará desde 1.35 V hasta 1.95 V.

$$V_{out}(x,y,z) = \frac{3.3}{2} \pm 0.3 = (1.35\ \text{V},\ 1.95\ \text{V}) \quad \textbf{(3.3)}$$

Por indicaciones del fabricante, para el correcto funcionamiento del acelerómetro tenemos que añadir unos condensadores colocados en cada salida (C_x,C_y,C_z) como se observa en la figura 3.15.

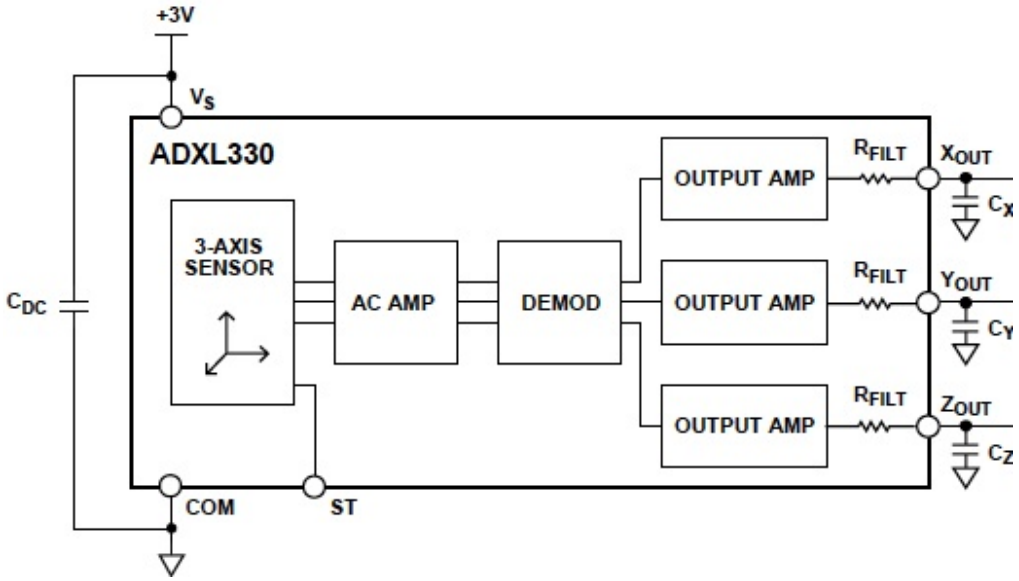

Fig. 3.15 Esquema ADXL330.

Según el fabricante podemos obtener diferentes anchos de banda según el valor que escojamos de dichos condensadores. El valor escogido es de 22 nF, con lo que conseguimos un ancho de banda de 200 Hz como observamos en la tabla 3.2 proporcionada por el fabricante del acelerómetro.

Tabla 3.2 Ancho de banda ADXL330.

Ancho de banda (Hz)	Capacidad (µC)
1	4.7
10	0.47
50	0.10
100	0.05
200	0.022
500	0.01

La elección de este valor se ha realizado en base a que no podemos tener un ancho de banda muy pequeño porque las variaciones rápidas se transmitirían con un cierto retardo.

Al disponer en la salida una variación de 600 mV con un offset de 1.65 V se ha diseñado un circuito de adaptación para mejorar el margen dinámico y así tener una mejor resolución en la medida. Para ello se ha diseñado el siguiente circuito.

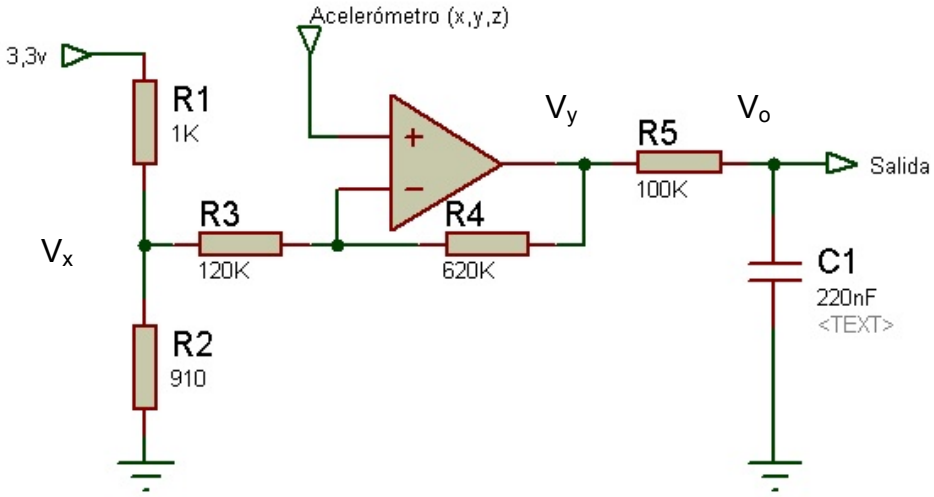

Fig. 3.16 Circuito adaptación acelerómetro.

El circuito de adaptación es un amplificador diferencial. Con este esquema lo que conseguimos es amplificar por 5 la diferencia que existe entre la V_x y la salida del acelerómetro, consiguiendo eliminar el offset (1.65 V) de la señal del acelerómetro.

$$V_y = V_{acelerometro}\left(1+\frac{R_4}{R_3}\right)-V_x\frac{R_4}{R_3} \quad (\mathbf{3.4})$$

$$V_x \approx \left(\frac{3.3}{R_1+R_2}\right)R_2 = 1.57\ \text{V} \quad (\mathbf{3.5})$$

Se eligió un valor de diseño de V_x igual a 1.57 V, para que el valor mínimo del acelerómetro (1.35 V) correspondiera a una V_y mínima de 0.21 V y a una V_y máxima de 3.9 V. Con estos valores no saturamos el amplificador.

$$V_y(\min) = 1.35\left(1+\frac{620}{120}\right)-1.57\frac{620}{120} = 0.2\ \text{V}$$

$$V_y(\max) = 1.95\left(1+\frac{620}{120}\right)-1.57\frac{620}{120} = 3.9\ \text{V} \quad (\mathbf{3.6})$$

Este mismo esquema se ha implementado en los tres ejes que incorpora el acelerómetro, ya que el comportamiento es mismo para los tres. Este diseño incorpora a la salida (V_y) un filtro paso bajo RC que nos ayuda a eliminar todo el ruido provocado por las vibraciones de todo el sistema cuando esta encendido. Experimentalmente se han encontrado los valores idóneos de dicho filtro para que la vibración de los motores afecte lo menos posible a la medida. La frecuencia de corte del filtro paso bajo es de 7,2 Hz.
En la ecuación 3.6 podemos encontrar el cálculo de la frecuencia de corte con un valor de resistencia igual a 100 KΩ y un condensador de 220 nF.

$$f_c = \frac{1}{2 \cdot \pi \cdot R \cdot C} = \frac{1}{2 \cdot \pi \cdot 100 \cdot 10^3 \cdot 220 \cdot 10^{-9}} = 7.2 \text{ Hz} \quad (\mathbf{3.7})$$

Una vez tenemos la información de la inclinación de nuestro sistema, debemos obtener la velocidad angular de cada eje (x,y,z) para poder realizar una estrategia de estabilización idónea. Los giroscopios elegidos son de la casa Murata , exactamente el modelo ENC-03 con las siguientes características:

Tabla 3.3 Características del giroscopio Murata ENC-03

Voltaje de alimentación	2.7~5.5 Vdc
Consumo de corriente	5 mA máx.
Velocidad angular máxima	±300 deg./sec.
Factor de escala	0.67 mV/deg./sec.
Respuesta	50 Hz máx.
Min. Tº de operación	-5°C
Max. Tº de operación	75°C
Peso	1.0 g máx.

El mismo fabricante propone el esquema de adaptación que se muestra en la figura 3.17.

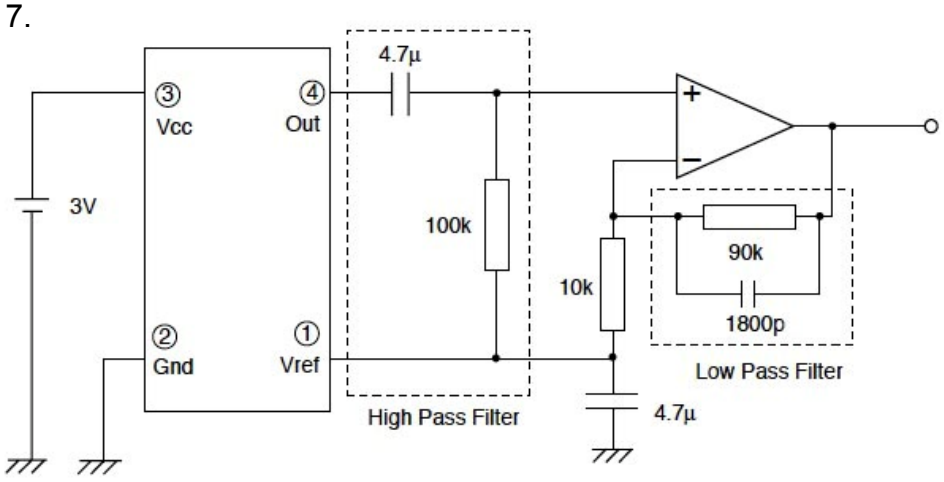

Fig. 3.17 Circuito adaptación giroscopio ENC-03.

El esquema se trata de un circuito amplificador y un filtro paso alto y uno paso bajos, con una frecuencia de corte de 0,3 Hz (para eliminar el offset) correspondiente al filtro paso alto y una frecuencia de corte de 1 khz para el paso bajos. El circuito consigue amplificar con una ganancia de 10 las pequeñas variaciones que se obtienen a la salida del giroscopio.

En el diseño del quadrotor se han utilizado tres giroscopios, una para cada eje, lo que quiere decir que la orientación de estos debe de ser la correcta. En la figura 3.18 vemos la posición en que se debe de colocar el giroscopio para su correcto funcionamiento respecto al eje x.

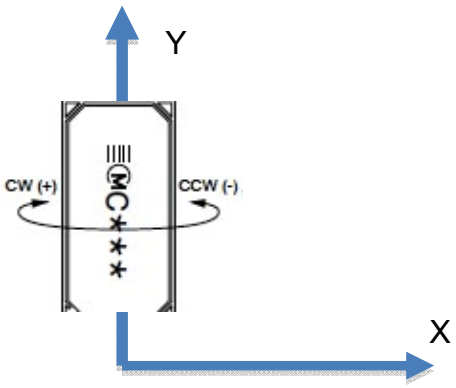

Fig. 3.18 Posición giroscopio.

Todo el sistema de sensores y circuitos adaptadores se han diseño en una placa de topos como se observa en la figura 3.19. Esta placa es totalmente adaptable a la placa de Arduino para minimizar el número de cables y así poder ensamblar todas las placas una encima de otra.

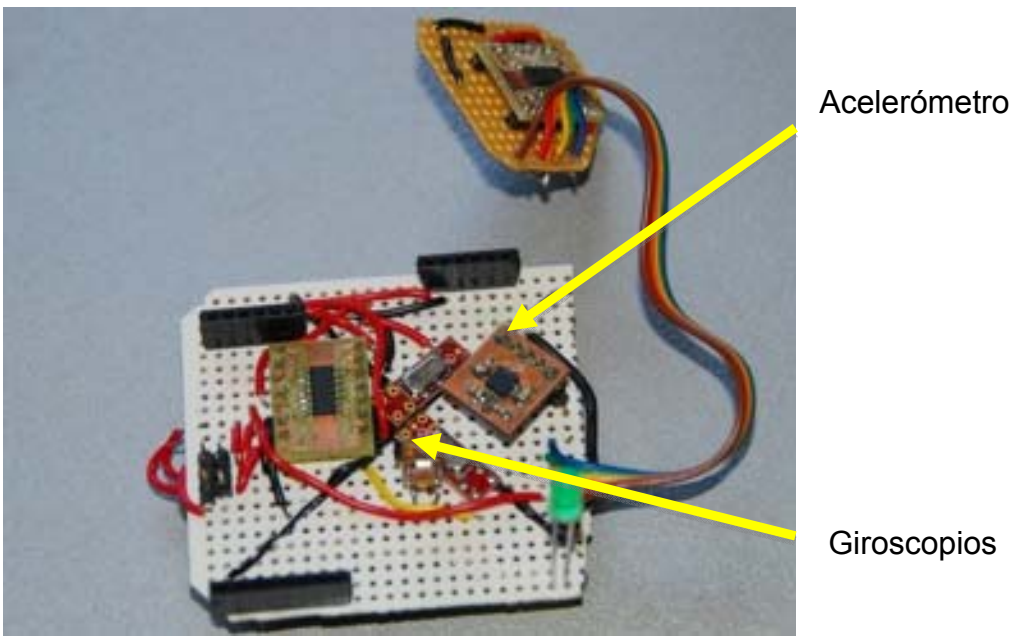

Fig. 3.19 Placa sensores.

3.5.3. Sistema de comunicación

El protocolo de comunicación escogido para el sistema quadrotor es Zigbee. Es un protocolo de comunicaciones inalámbrico basado en el estándar de comunicaciones para redes inalámbricas IEEE_802.15.4. Creado por Zigbee Alliance, una organización, teóricamente sin ánimo de lucro, de más de 200 grandes empresas (destacan Mitsubishi, Honeywell, Philips, ODEM do, Invensys, entre otras), muchas de ellas fabricantes de semiconductores.

Zigbee permite que dispositivos electrónicos de bajo consumo puedan realizar sus comunicaciones inalámbricas. Es especialmente útil para redes de sensores en entornos industriales, médicos y, sobre todo, domóticos.

Las comunicaciones Zigbee se realizan en la banda libre de 2.4 GHz. El alcance depende de la potencia de transmisión del dispositivo así como también del tipo de antenas utilizadas (cerámicas, dipolos, etc) El alcance normal con antena dipolo en línea vista es de aproximadamente de 100m y en interiores de unos 30 m. La velocidad de transmisión de datos de una red Zigbee es de hasta 256 kbps..

Entre las necesidades que satisface el protocolo se encuentran:

- Bajo coste.
- Ultra-bajo consumo de potencia.
- Uso de bandas de radio libres y sin necesidad de licencias.
- Instalación barata y simple.
- Redes flexibles y extensibles.

Existen unos módulos del fabricante Maxstream [18] que utiliza el protocolo Zigbee llamados Xbee. Cada módulo Xbee, al igual que ocurre con las direcciones MAC de los dispositivos ethernet, tiene una dirección única. En el caso de los módulos Xbee cada uno de ellos tiene una dirección única de 64 bits que viene grabada de fábrica. Por otro lado, la red Zigbee, utiliza para sus algoritmos de ruteo direcciones de 16 bits. [19]

Fig. 3.20 Módulo Xbee utilizado.

Estos módulos Xbee, pueden ser ajustados para usarse en redes de configuración punto a punto, punto-a-multipunto o peer-to-peer (todos usuarios y servidores a la vez).

En el vehículo quadrotor el método de configuración elegido es punto a punto, ya que no necesitamos una configuración multipunto. Se ha elegido esta tecnología Xbee ya que la función punto-multipunto no se descarta en un futuro en la comunicación de varios sistemas quadrotores a la vez.
La configuración punto a punto es la conexión ideal para reemplazar comunicación serial por un cable. Sólo se debe configurar la dirección. Para ello se utilizan los comandos MY y el DL. La idea, es definir arbitrariamente una dirección para un módulo, usando el comando MY, el cual se va a comunicar con otro que tiene la dirección DL, también definida arbitrariamente. Con esto cada módulo define su dirección con MY, y escribe la dirección del módulo al cual se desea conectar usando DL.

El comando MY, define un número de 16 bit como dirección del módulo dentro de la red. El rango se encuentra entre 0x0 y 0xFFFE (la dirección 0xFFFF y 0xFFFE son para habilitar la dirección de 64-bit, por lo que si se desea utilizar direccionamiento de 16 bits, estos valores no deben ser usados). Para definirla se introduce ATMY y el número en formato hexadecimal, pero sin el '0x'. Por ejemplo si a un módulo se le quiere asignar la dirección 0x3BF1 (15345 en decimal), entonces se debe ingresar el comando ATMY3BF1. El comando DL, permite definir un número de 16 bit como dirección del módulo de destino dentro de la red al cual se va a realizar la comunicación.

Para realizar la configuración debemos trabajar en el modo comando, este modo permite ingresar comandos al módulo Xbee, para configurar, ajustar o modificar parámetros. Permite ajustar parámetros como la dirección propia o la de destino, así como su modo de operación entre otras cosas. Para poder ingresar los comandos es necesario utilizar el Hyperterminal de Windows, el programa X-CTU4 o algún microcontrolador que maneje UART y tenga los comandos guardados en memoria o los adquiera de alguna otra forma.

Para ajusta la tasa de transmisión entre el módulo y su cliente conectado a través de la interfaz serial debemos configurar los módulos mediante el comando BD'Valor' según se muestra en la siguiente tabla:

Tabla 3.4 Tasas transmisión Xbee.

'Valor'	Tasa (bps)
0	1200
1	2400
2	4800
3	9600
4	19200
5	38400

En el sistema de comunicación del quadrotor se ha elegido una tasa de transmisión de 19200 bps considerándolo suficiente ya que no se ha va transmitir mucha información desde las dos unidades (serán simples ordenes de navegación).

La configuración aplicada al módulo del quadrotor es la siguiente;

MY1, DL0, BD4

La configuración aplicada al modulo de comunicación es la siguiente:

MY0, DL1, BD4

En el mercado existe un módulo que incorpora el sistema Xbee y es totalmente compatible con la placa de Arduino. Como vemos en la figura 3.21 el módulo se adapta perfectamente a la placa Duemilanove Arduino.

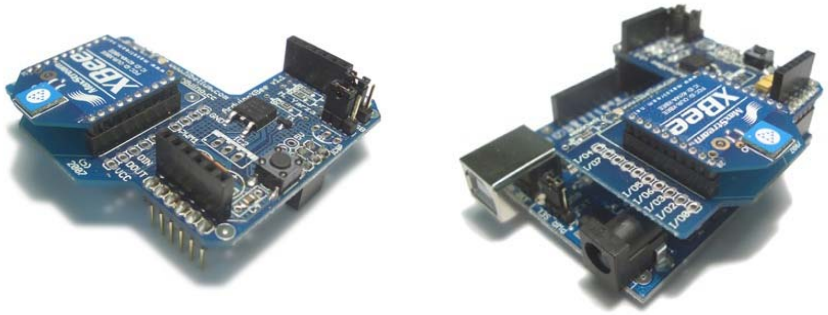

Fig. 3.21 Módulo Xbee compatible con Arduino

El control del quadrotor se ha realizado mediante otro modulo Xbee, este módulo va incorporado a una placa Arduino. Inicialmente se trabajó vía PC ya que es más fácil de programar y se puede utilizar para recibir datos desde el quadrotor. Una vez programado el quadrotor y definida toda la comunicación se paso a controlar el vehículo mediante nunchuk.

Control mediante PC

El control mediante PC se consigue conectado Arduino con el módulo Xbee al PC, la conexión entre ellos se realiza con el puerto USB que incorpora la placa Arduino. Este método de control se puede utilizar para controlar el quadrotor, pero en este diseño se ha destinado básicamente a leer los datos que se envían desde el quadrotor. Los datos se pueden interpretar con muchos programas, el más utilizado es el propio programa de Arduino con la opción serial monitor que encontraremos en la misma interface principal del programa. Mediante el serial monitor podemos tener acceso muy rápido al puerto de comunicaciones de la placa Arduino y facilitar su uso.

Control mediante Nunchuk

Este método es el que se ha utilizado principalmente para controlar el quadrotor. El Nunchuk es un mando del fabricante Nintendo, que se utiliza en la famosa consola Wii. Este dispositivo incorpora; un joystick centrar, un acelerómetro de 3 ejes y dos botones tal y como observamos en la figura 3.22.

Fig. 3.22 Mando Nunchuk desarrollado por Nintendo

Se ha elegido este dispositivo para el control del quadrotor porque nos proporciona todos los movimientos necesarios para realizar un control de todos los parámetros del quadrotor. El nunchuk utiliza un protocolo de comunicación I2C que podemos programar en el Arduino, existen librerías [20] de libre distribución que proporcionan toda la comunicación entre el nunchuk y el Arduino, la forma de conectar el nunchuk al Arduino es con una placa como se muestra en la figura 3.23.

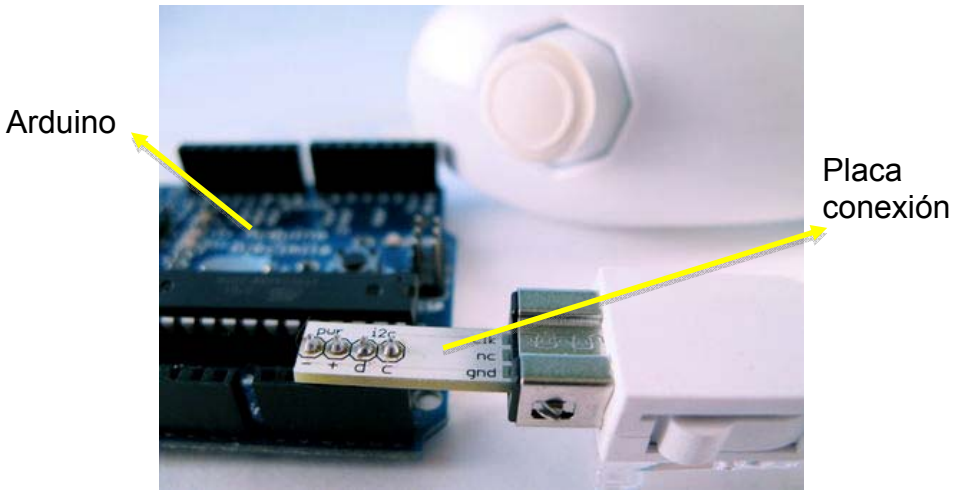

Fig. 3.23 Conector Nunchuk

El diagrama de programación que se ha seguido para controlar la potencia de todo el sistema es el que se muestra en la figura 3.24.

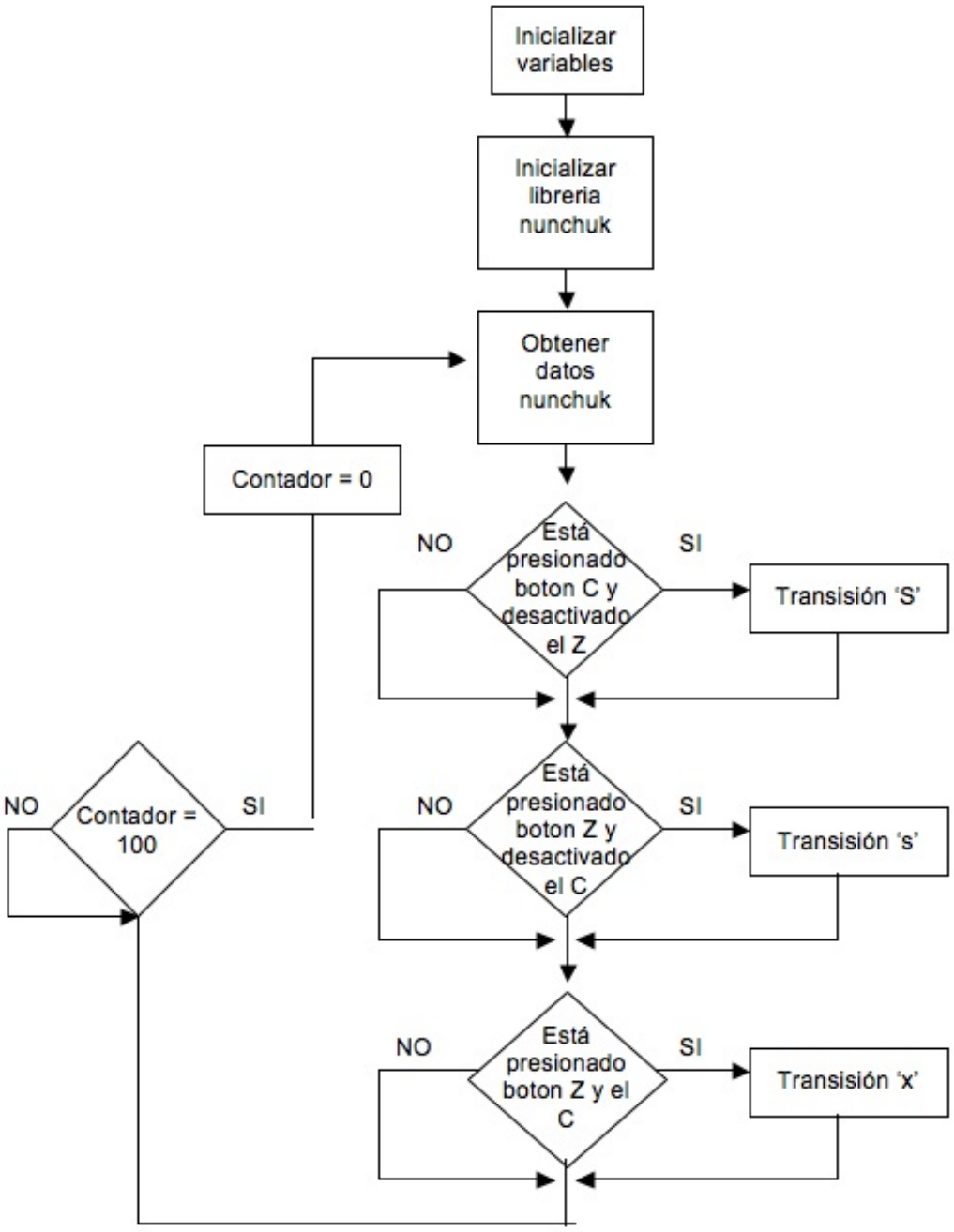

Fig. 3.24 Diagrama programación nunchuk

Con esta programación lo que se consigue es utilizar el nunchuk como una unidad de control de potencia de los motores, siendo totalmente inalámbrico y con un alcance aproximado de 100 m. Si transmitimos una 'S' el quadrotor incrementará su potencia y una 's' la decrementará. Si presionamos a la vez los dos botones se transmitirá una 'x' y significará parada de todos los motores.
En futuras versiones del FLISI1 se puede implementar la dirección del quadrotor mediante el movimiento del nunchuk gracias a los acelerómetros.

3.6. Estabilización

La estrategia elegida para realizar la estabilización del quadrotor es un sistema de control realimentado. Un sistema de control realimentado es un sistema que mantiene una relación determinada entre la salida y la entrada de referencia, comparándolas y usando la diferencia como medio de control [21]. Los sistemas realimentados se denominan también sistemas de control en lazo cerrado. En un sistema de control de lazo cerrado, se parte de la señal de error de actuación, que es la diferencia entre la señal de entrada y la señal de realimentación, con el fin de reducir el error y llevar la salida del sistema a un valor deseado.

Una ventaja del sistema de control en lazo cerrado es que el uso de la realimentación vuelve la respuesta del sistema relativamente insensible a las perturbaciones externas y a las variaciones internas en los parámetros del sistema.

Desde el punto de vista de estabilidad, el sistema de control en lazo abierto es más fácil de desarrollar pero menos eficiente a la hora de conseguir una buena estabilización. Cuando queremos controlar la estabilidad de un sistema la mejor solución es utilizar un control en lazo cerrado, en este caso se complica el diseño ya que es relativamente fácil corregir en exceso errores que nos producirán oscilaciones en todo el sistema.

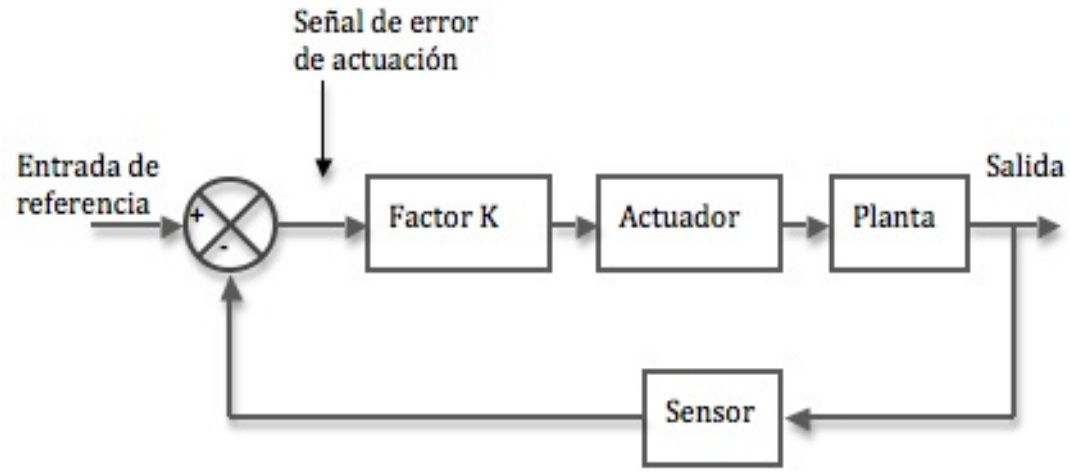

Fig. 3.25. Diagrama de estabilización

En la figura 3.25 observamos como se calcula el error, mediante la diferencia entre la entrada de referencia y la salida de nuestra planta.

Se describe como planta a todos los elementos del sistema que componen el quadrotor y tienen una relación entre la entrada y la salida. En estudios de dinámica se consigue modelar la planta con funciones de transferencia para su posterior desarrollo.

En el sistema quadrotor tenemos que controlar los tres movimientos posibles, como observa en la figura 3.26 [22]. La estrategia de control de la lazo cerrado es aplicada de igual forma para los tres ejes.

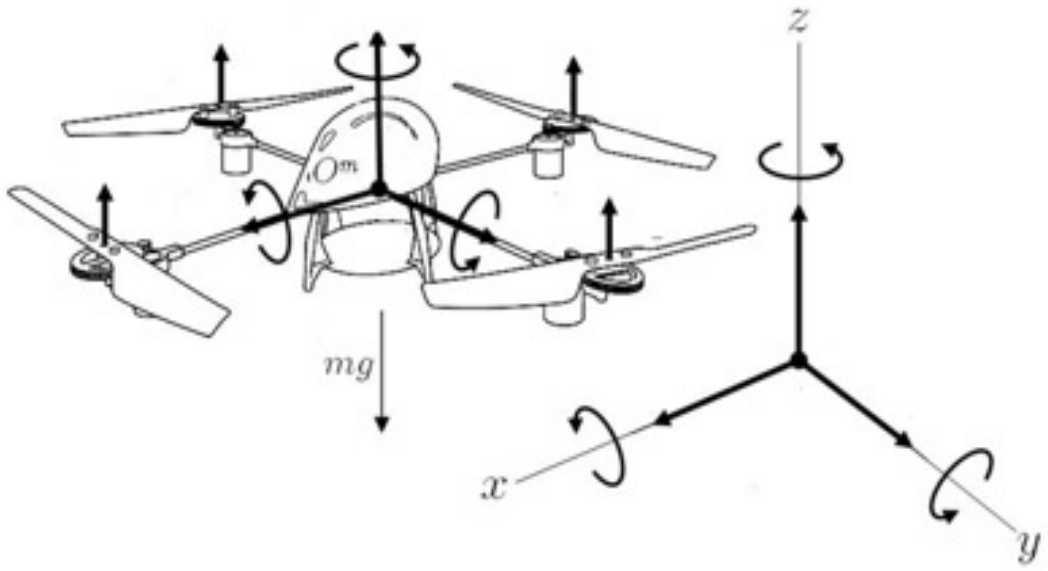

Fig. 3.26. Ejes Quadrotor

Para implementar el diagrama de estabilización que se muestra en la figura 3.25 se ha realizado una programación que podemos observar en el diagrama de bloques de la figura 3.27.

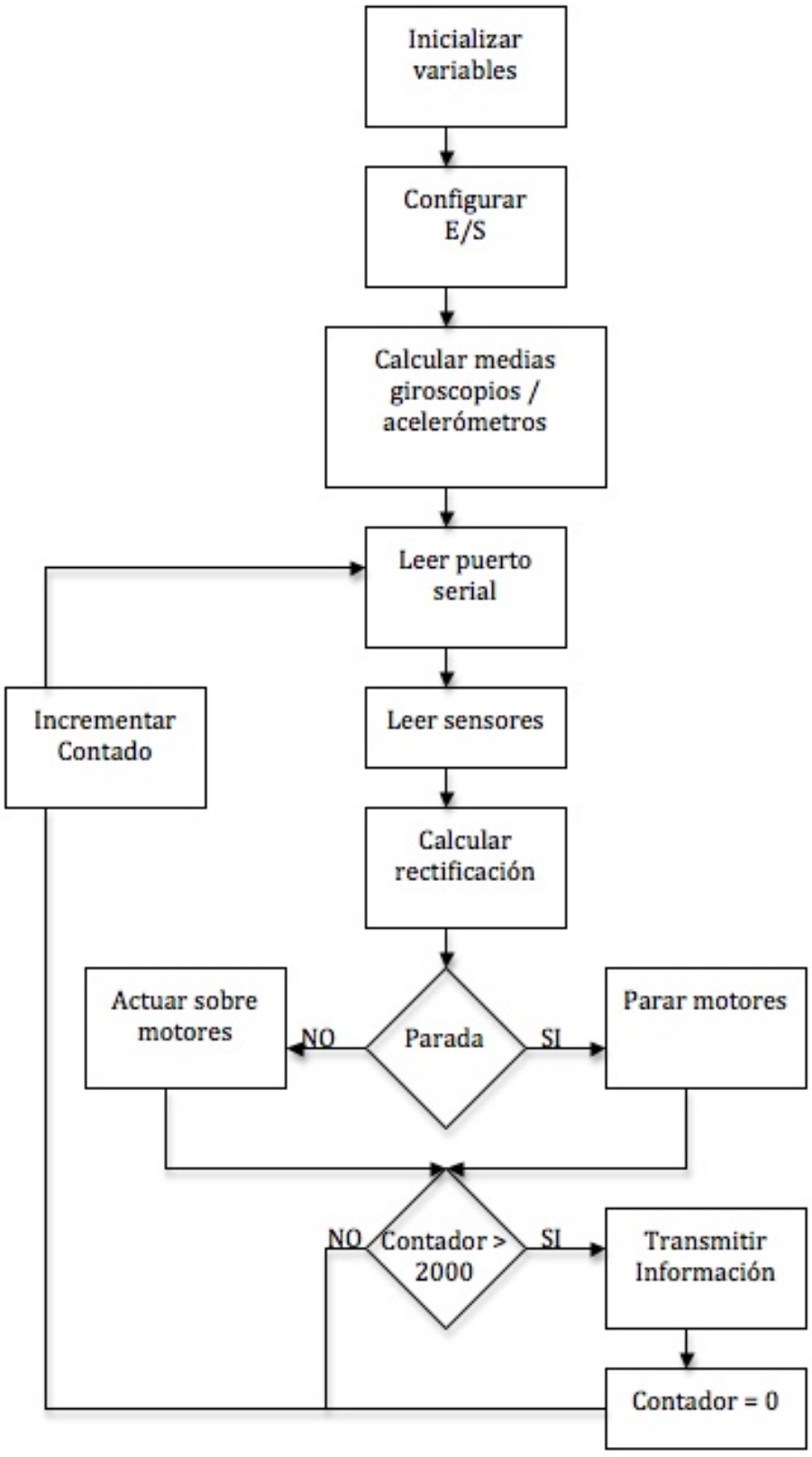

Fig. 3.27. Diagrama programación estabilización.

El diagrama de programación se puede descomponer en cuatro fases. Una primera fase de inicialización de variables donde se aprovecha para guardar los valores de nivel “0” de todos los sensores. Esta tarea se realiza calculando la media de 50 muestras cuando el quadrotor está en el suelo. En esta fase el led que tenemos en la placa parpadea 2 veces y a la tercera se quedará encendido, a partir de este momento quiere decir que ya podemos utilizarlo.

La segunda fase consiste en leer los datos que provienen del emisor, es decir, como ya se ha comentado anteriormente se utiliza otro Arduino que envía información para subir o bajar, también podemos decir al quadrotor que apague los motores de golpe. La tercera fase se ocupa de captar los datos de los sensores. Arduino tiene una frecuencia de muestreo de 10 khz y un conversor analógico digital de 10 bits, con lo que se obtiene una resolución de 4.8 mV ($5\ V / 2^{10}$). En esta fase se realiza una pequeña media de los 4 valores anteriores para evitar el ruido provocado por los motores en funcionamiento.

A partir de esta fase ya tenemos toda la información necesaria :

- Inclinación X , Y
- Velocidad angular X, Y ,Z

Cuando ya tenemos los valores de inclinación se realiza otro cálculo utilizando la velocidad angular para darle más peso al resultado o menos. Esta etapa se identifica en la figura 3.25 como “factor K”. Hay que tener en cuenta que no es lo mismo que el vehículo se incline lentamente o muy rápidamente, lo que se consigue a través de la velocidad angular es que la respuesta de los motores sea más fuerte o menos según dicha velocidad.

La última fase se encarga de enviar pequeñas informaciones a través del canal, esta tarea se realiza cada 2000 pasos por el bucle, ya que no hace falta tener una información más inmediata porque podemos saturar el canal con tanta información. Hay que tener cierto cuidado con esta fase, ya que si utilizamos mucho la instrucción “Serial.print()” podemos ralentizar todo el bucle provocando un retardo en todo el cálculo lo que implica que el circuito no se estabilice bien y tienda a oscilar. Si tuviéramos que enviar mucha información, lo idóneo es introducirlo todo en una misma trama y utilizar sólo una vez la instrucción

CAPITULO 4. CARACTERÍSTICAS FUNCIONALES

El quadrotor desarrollado es el que se muestra en la figura 4.1

Fig. 4.1 Fotografía del Quadrotor FLISI1.

Se puede observar como toda la electrónica se aloja en el centro para ayudar a mantener un sistema totalmente simétrico. En la figura 4.2 podemos observar las partes que forman la unidad de control.

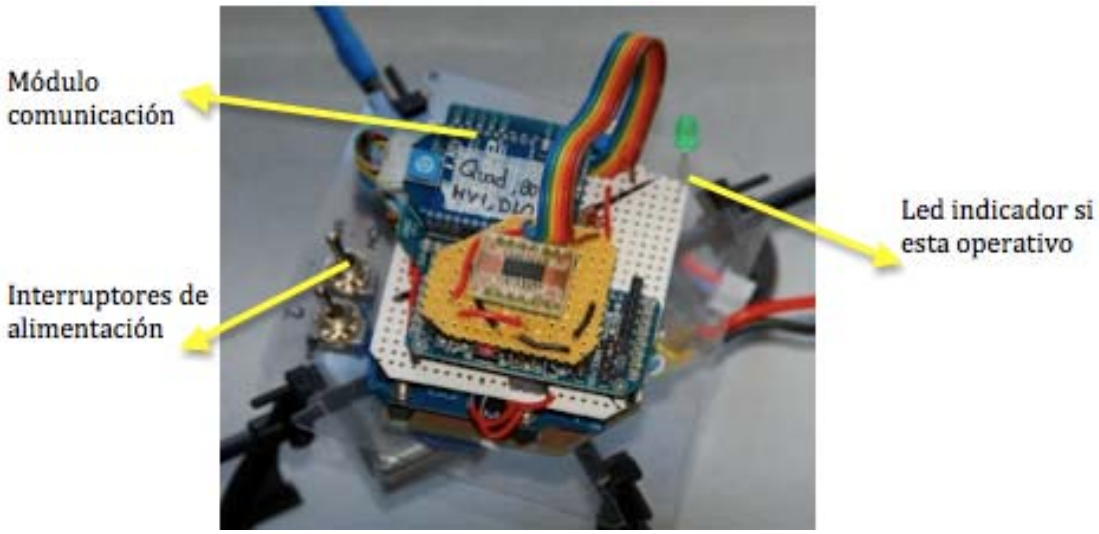

Fig. 4.2 Unidad de control FLISI1.

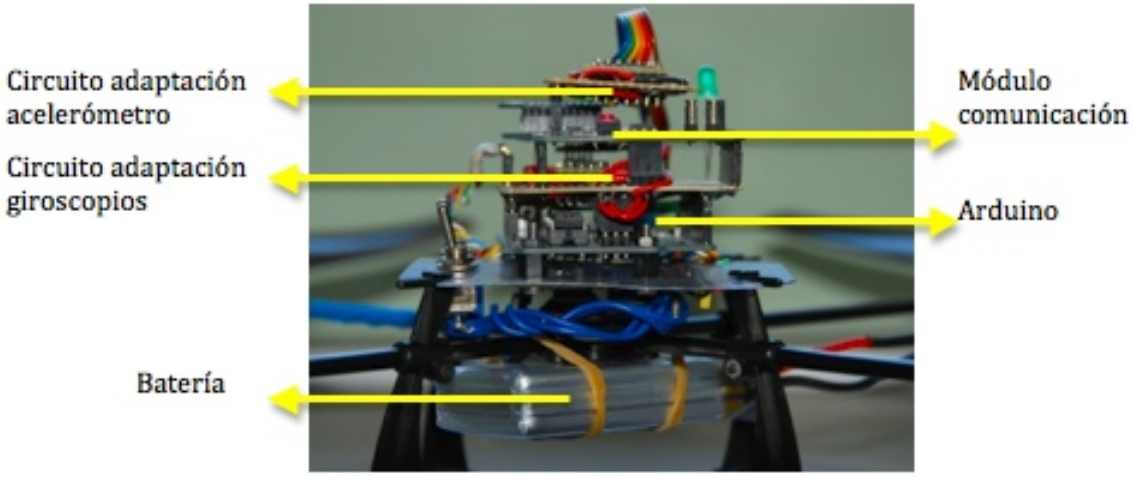

Fig. 4.3 Unidad de control y batería FLISI1.

En la figura 4.3 podemos ver la unidad de control como se compone de todos los modulos mencionados en los anteriores capítulos. Se ha utilizado un diseño modular, permitinedo la facilidad de configurarlo y poder añadir más sensores con facilidad. En la figura 4.4 se observa todos los módulos por separado.

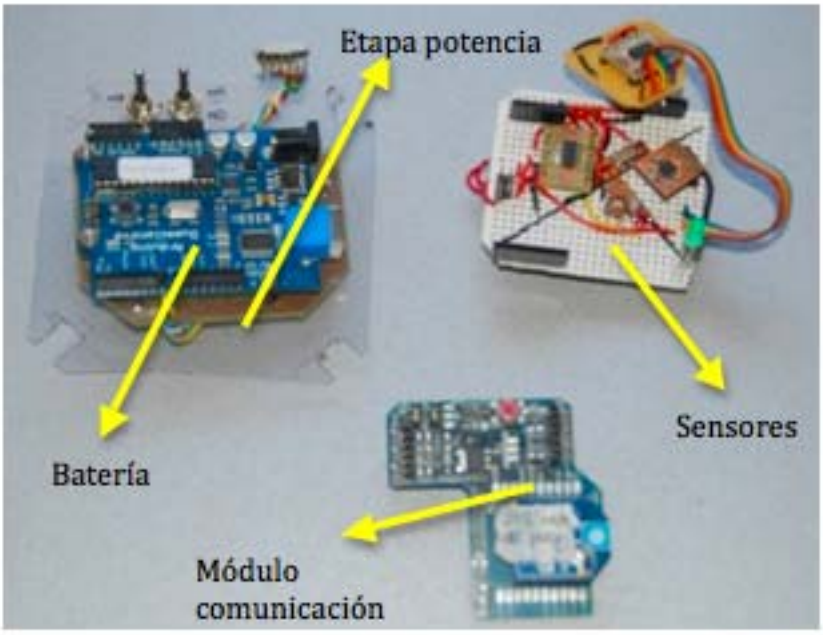

Fig. 4.4 Partes unidad de control FLISI1.

En la tabla 4.1 podemos observar los diferentes pesos de todos los componentes que forman el quadrotor FLISI1. Se puede observar que el 50 % del peso corresponde a la estructura y los motores. Durante todo el diseño del quadrotor se ha tenido en cuenta el peso de los componentes a la hora de elegir uno u otro.

Tabla 4.1 Peso componentes FLISIS1.

Módulo / Componente	Unidades	Peso unitario (gr)	Peso total (gr)
Motor	4	47.7	190.8
Estructura	1	123.2	123.2
IMU	1	35.3	35.3
Xbee	1	17.7	17.7
Arduino	1	28	28
Etapa potencia	1	54	54
Total sin Bateria			449
Total + Batería 1300mAh			**558**
Total + Batería 1800 mAh			**595**

Como se observar, el peso de todo el sistema no supera los 600 gr lo que sitúa al UAV en la categoría mini como se observa en la tabla 1.1 del capítulo 1.

El peso que es capaz de levantar es de 1112 gr por lo tanto tenemos una relación carga / peso de 1.86. Este resultado significa que el quadrotor es capaz de levantar un 53 % más que su peso. En la figura 4.5 podemos ver el esquema que se ha utilizado para medir el peso que levanta, se han invertido las aspas colocando el quadrotor encima de una báscula de precisión y se ha suministrado la máxima potencia a los motores. El peso que es capaz de levantar (1112 gr) es el resultado de realizar la resta del lo medido por la báscula menos el peso del quadrotor (595 gr).

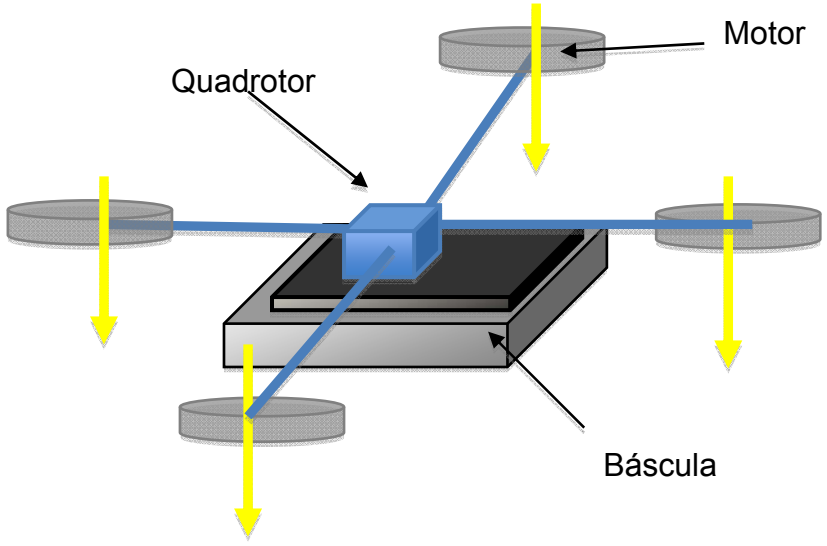

Fig. 4.5 Set up de medida para el cálculo de la carga que eleva el quadrotor.

La duración teórica aproximada de las baterías se calculó a partir del consumo de todo el sistema. Este consumo es aproximado ya que depende principalmente de la fuerza ejercida por los motores y de la resistencia de giro que tengan. El consumo medio de todo el sistema medido en el laboratorio es de 7.12 A.

A continuación se realiza el cálculo de la duración a partir del consumo medio del quadrotor con las dos baterías que se han utilizado en el proyecto:

Batería 1300 mAh

$$Duracion = \frac{1300 \cdot 10^{-3^{\circ}}\ \text{Ah}}{7.12\ \text{A}} = 0.182\ \text{h}$$
$$0.182\ \text{h} \times \frac{60\ \text{min}}{1\ \text{h}} \approx 10\ \text{min} \qquad (4.1)$$

Batería 1800 mAh

$$Duracion = \frac{1800 \cdot 10^{-3^{\circ}}\ \text{Ah}}{7.12\ \text{A}} = 0.252\ \text{h}$$
$$0.252\ \text{h} \times \frac{60\ \text{min}}{1\ \text{h}} \approx 15\ \text{min} \qquad (4.2)$$

Como podemos observar en los cáculos realizados, aproximadamente tenemos una duración de 10 min con la batería de 1300 mAh y de 15 min con la de 1800 mAh.

PRESUPUESTO

Componente	Unidades	Precio Unitario €	Precio Total €
Estructura + Motores + Aspas	1	141.90	141.90
Arduino Duemilanove	2	24.00	48.00
Módulo Xbee	2	42.90	85.80
Mando nunchuk	1	20.00	20.00
	Sub Total		**295.7**
Unidad de control			
Giroscopio ENC-03	3	24.95	74.85
Acelerómetro ADXL330	1	23.90	23.90
A.O OPA3555	2	4.40	8.80
Resistencias smd	24	0.2	4.8
Led	1	0.3	0.3
Condensadores smd	12	0.2	2.4
Placa + Cables + Conectores	1	5	5
	Sub Total		**120.05**
Etapa Potencia			
IRLZ44NS	4	2.46	9.84
ICL7667	2	2.38	4.76
Diodos BAT47	4	0.2	0.8
Resistencia 100 Ω	4	0.2	0.8
Condensador 100nF	1	0.2	0.2
Placa + Cables + Conectores	1	5	5
	Sub Total		**20.6**
Batería 1800 mAh	1	58.20	**58.2**
TOTAL MATERIAL			**494.55**
Horas Ingeniero técnico *	360	6	**2160**
TOTAL PROYECTO			**2654.55 €**

* Media de 4 horas por día durante 4 meses y medio de lunes a viernes. El precio de 6 € / hora es el estipulado para un estudiante de ingeniería técnica en prácticas de empresa.

CONCLUSIONES

El quadrotor es un vehículo aéreo no tripulado que posee características que destacan sobre los otros UAV. Tanto su dinámica como capacidad de vuelo lo hacen único en maniobrabilidad proporcionando una gran capacidad de vuelo en interiores.

Actualmente se están desarrollando sistemas muy avanzados en el mundo de los UAV proporcionando capacidades como evitar obstáculos y trabajo en grupo con otros UAV.

Este trabajo ha cumplido con los objetivos previsto de construir una plataforma en la cual poder trabajar y diseñar diferentes esquemas de estabilización.

Se ha demostrado que utilizando la plataforma Arduino podemos realizar proyecto de gran envergadura a bajo coste.

Se ha conseguido una estabilización en un eje (con dos motores). Las pruebas realizadas con los cuatro motores a la vez, han dado como resultado un sistema no totalmente estable. La posible causa de este problema proviene de diseñar la estabilización por separado en un eje y luego aplicarlo por simetría a todo el sistema. Que el sistema sea inestable con los 4 motores a la vez es debido a que la caracterización del sistema no es igual con dos motores que con cuatro. La solución a este problema es diseñar un esquema de estabilización PID (Proporcional Integral Derivativo), esto podría formar parte de las líneas a seguir en el futuro.

Un segundo reto de cara el futuro es conseguir que el sistema pueda desplazarse controlando en todo momento su posición, por ejemplo, mediante un sistema GPS.

REFERENCIAS

[1] A.Barrientos, J.del Cerro, P.Gutiérrez, R.San Martín, A.Martínez, C.Rossi,"Vehículos aéreos no tripulados para uso civil. Tecnología y aplicaciones", Grupo de Robótica y Cibernética, Universidad Politécnica de Madrid,

[2] http://www.draganfly.com/uav-helicopter/draganflyer-x6/, 30 de junio 2009.

[3] http://www.rctoys.com/rc-products-catalog/RC-HELICOPTERS-DRAGANFLYER-VTI.html, 1 de julio 2009.

[4] http://vertol.mit.edu/index.html, 1 de julio 2009.

[5] http://mikrokopter.com/ucwiki/, 1 de julio 2009.

[6] Trabajo de investigación subvencionado por UPC, "Seguimiento de objetos mediante quadrotores con navegación visual". Participantes C.Díaz,J.Bracke,S.Zicarelli, tutor O.Casas, 2009

[7] http://www.st.com/stonline/, 9 de julio 2009.

[8] http://www.murata.com/index.html, 7 de julio 2009.

[9] Fernandez Oltra, Ruben,"Sistema de adquisición y posicionamiento geográfico",PFC, Ecuela Politécnica Superior Ingenieria Vilanova i la Geltrú, 2007

[10] http://www.analog.com, 2 de julio 2009.

[11] http://www.mabuchi-motor.co.jp, 2 de julio 2009

[12] E.Arnal,F.Carlos, "Diseño, construcción y control de un MAV bimotor", TFC, Escuela Politécnica Superior de Castelldefels, director O.Casas, 2009.

[13] http://www.irf.com/indexsw.html, 9 de julio 2009.

[14] http://www.intersil.com/html/, 9 de julio 2009

[15] http://www.dualsky.com, 9 de julio 2009.

[16] http://www.flightpower.co.uk/, 9 de julio 2009.

[17] http://www.arduino.cc/ ,9 de julio 2009

[18] http://www.maxstream.fr, 9 de julio 2009.

[19] Oyarce, Andrés "Guia de usuario Xbee Serie1"

[20] http://todbot.com/arduino/sketches/WiichuckDemo/nunchuck_funcs.h, 9 de julio 2009

[21] Ogata,K, *Ingeniería de Cotrol*, Pearson educación, S.A, Madrid, 2003 edición 4º.

[22] T. Madani and A. Benallegue, "Backstepping Sliding Mode Control Applied to a Miniature Quadrotor Flying Robot", Laboratoire d'Ingénierie des Systèmes de Versailles

ANEXOS

ANEXO 1. Placa control motores

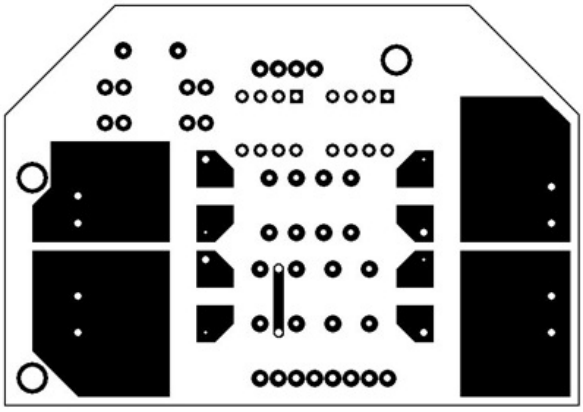

Figura A1.1. Capa Top.

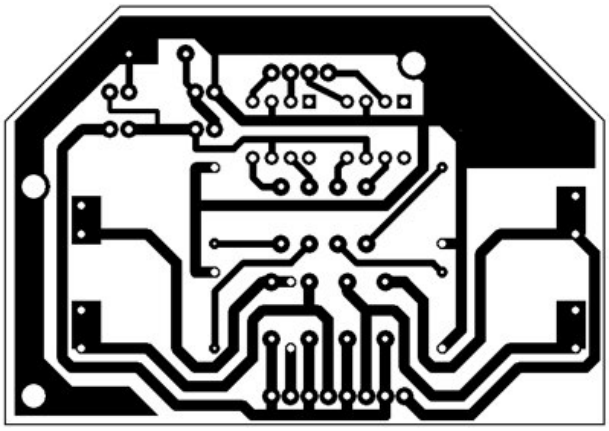

Figura A1.2. Capa Botton.

ANEXO 2. Código programación Arduino nunchuk

```
#include <Wire.h>
#include "nunchuck_funcs.h"

int loop_cnt=0;

byte accx,accy,zbut,cbut;
int joyx,joyy;
int ledPin = 13;


void setup()
{
   Serial.begin(19200);
   nunchuck_setpowerpins();
   nunchuck_init();

}

void loop()
{
   if( loop_cnt > 100 ) { // every 100 msecs get new data
      loop_cnt = 0;

      nunchuck_get_data();

      accx  = nunchuck_accelx();
      accy  = nunchuck_accely();
      zbut = nunchuck_zbutton();
      cbut = nunchuck_cbutton();
      joyx = nunchuck_joyx();
      joyy = nunchuck_joyy();


      if(cbut == 1 && zbut ==0) tx(111);
      if(cbut == 0 && zbut ==1) tx(222);
      if(cbut == 1 && zbut ==1) tx(100);

      if(joyx<40) tx(11);
      else if(joyx > 200) tx(10);
      if(joyy<60) tx(21);
      else if (joyy > 200) tx(20);

   }
   loop_cnt++;
   delay(1);
}

void tx (int val) {
  if(val == 11) Serial.print('A');
  else if (val == 21) Serial.print('B');
  else if (val == 31) Serial.print('C');
  else if (val == 41) Serial.print('D');
  else if (val == 10) Serial.print('a');
  else if (val == 20) Serial.print('b');
  else if (val == 30) Serial.print('c');
  else if (val == 40) Serial.print('d');
  else if (val == 111) Serial.print('S');     //Motor 1234 up
  else if (val == 222) Serial.print('s');     //Motor 1234 down
  else if (val == 100) Serial.print('x');     //parada
}
```

ANEXO 3. Libreria nunchuk

```
/*
 * Nunchuck functions  -- Talk to a Wii Nunchuck
 *
 * This library is from the Bionic Arduino course :
 *                      http://todbot.com/blog/bionicarduino/
 *
 * 2007 Tod E. Kurt, http://todbot.com/blog/
 *
 * The Wii Nunchuck reading code originally from Windmeadow Labs
 *   http://www.windmeadow.com/node/42
 */

#include <WProgram.h>

static uint8_t nunchuck_buf[6];   // array to store nunchuck data,

// Uses port C (analog in) pins as power & ground for Nunchuck
static void nunchuck_setpowerpins()
{
#define pwrpin PORTC3
#define gndpin PORTC2
    DDRC |= _BV(pwrpin) | _BV(gndpin);
    PORTC &=~ _BV(gndpin);
    PORTC |=  _BV(pwrpin);
    delay(100);  // wait for things to stabilize
}

// initialize the I2C system, join the I2C bus,
// and tell the nunchuck we're talking to it
static void nunchuck_init()
{
    Wire.begin();               // join i2c bus as master
    Wire.beginTransmission(0x52);// transmit to device 0x52
    Wire.send(0x40);// sends memory address
    Wire.send(0x00);// sends sent a zero.
    Wire.endTransmission();// stop transmitting
}

// Send a request for data to the nunchuck
// was "send_zero()"
static void nunchuck_send_request()
{
    Wire.beginTransmission(0x52);// transmit to device 0x52
    Wire.send(0x00);// sends one byte
    Wire.endTransmission();// stop transmitting
}

// Encode data to format that most wiimote drivers except
// only needed if you use one of the regular wiimote drivers
static char nunchuk_decode_byte (char x)
{
    x = (x ^ 0x17) + 0x17;
    return x;
}

// Receive data back from the nunchuck,
// returns 1 on successful read. returns 0 on failure
static int nunchuck_get_data()
{
    int cnt=0;
    Wire.requestFrom (0x52, 6);// request data from nunchuck
    while (Wire.available ()) {
        // receive byte as an integer
        nunchuck_buf[cnt] = nunchuk_decode_byte(Wire.receive());
        cnt++;
```

```
    }
    nunchuck_send_request();  // send request for next data payload
    // If we recieved the 6 bytes, then go print them
    if (cnt >= 5) {
        return 1;   // success
    }
    return 0; //failure
}

// Print the input data we have recieved
// accel data is 10 bits long
// so we read 8 bits, then we have to add
// on the last 2 bits.  That is why I
// multiply them by 2 * 2
static void nunchuck_print_data()
{
    static int i=0;
    int joy_x_axis = nunchuck_buf[0];
    int joy_y_axis = nunchuck_buf[1];
    int accel_x_axis = nunchuck_buf[2]; // * 2 * 2;
    int accel_y_axis = nunchuck_buf[3]; // * 2 * 2;
    int accel_z_axis = nunchuck_buf[4]; // * 2 * 2;

    int z_button = 0;
    int c_button = 0;

    // byte nunchuck_buf[5] contains bits for z and c buttons
    // it also contains the least significant bits for the accelerometer data
    // so we have to check each bit of byte outbuf[5]
    if ((nunchuck_buf[5] >> 0) & 1)
        z_button = 1;
    if ((nunchuck_buf[5] >> 1) & 1)
        c_button = 1;

    if ((nunchuck_buf[5] >> 2) & 1)
        accel_x_axis += 2;
    if ((nunchuck_buf[5] >> 3) & 1)
        accel_x_axis += 1;

    if ((nunchuck_buf[5] >> 4) & 1)
        accel_y_axis += 2;
    if ((nunchuck_buf[5] >> 5) & 1)
        accel_y_axis += 1;

    if ((nunchuck_buf[5] >> 6) & 1)
        accel_z_axis += 2;
    if ((nunchuck_buf[5] >> 7) & 1)
        accel_z_axis += 1;

    i++;
}

// returns zbutton state: 1=pressed, 0=notpressed
static int nunchuck_zbutton()
{
    return ((nunchuck_buf[5] >> 0) & 1) ? 0 : 1;  // voodoo
}

// returns zbutton state: 1=pressed, 0=notpressed
static int nunchuck_cbutton()
{
    return ((nunchuck_buf[5] >> 1) & 1) ? 0 : 1;  // voodoo
}

// returns value of x-axis joystick
static int nunchuck_joyx()
{
    return nunchuck_buf[0];
}
```

```
// returns value of y-axis joystick
static int nunchuck_joyy()
{
   return nunchuck_buf[1];
}

// returns value of x-axis accelerometer
static int nunchuck_accelx()
{
   return nunchuck_buf[2];   // FIXME: this leaves out 2-bits of the data
}

// returns value of y-axis accelerometer
static int nunchuck_accely()
{
   return nunchuck_buf[3];   // FIXME: this leaves out 2-bits of the data
}

// returns value of z-axis accelerometer
static int nunchuck_accelz()
{
   return nunchuck_buf[4];   // FIXME: this leaves out 2-bits of the data
}
```

ANEXO 4. Código programación Control motores

```
#define LEDPIN 13
#define MOTOR1 3
#define MOTOR2 5
#define MOTOR3 9
#define MOTOR4 6
#define BAUDIOS 19200
#define ACXPIN 1
#define ACYPIN 0
#define ACZPIN 2
#define GYXPIN 3
#define GYYPIN 4
#define GYZPIN 5
#define TAMZERO 50
#define FILTERSAMPLES 7
#define TAMMEDIA 4
//#define valor 20
#define margen 1

float valor = 5.2;
float maxg = 3.2;

char val;
int con = 0;
float m1 = 0;
float m2 = 0;
float m3 = 0;
float m4 = 0;
int inc = 5;
int pot = 0;

int RAWax = 0;
int RAWay = 0;
int RAWaz = 0;
int RAWgx = 0;
int RAWgy = 0;
int RAWgz = 0;
int SMOOTHax = 0;
int SMOOTHay = 0;
int SMOOTHaz = 0;
int SMOOTHgx = 0;
int SMOOTHgy = 0;
int SMOOTHgz = 0;
int SMOOTHarrayax [FILTERSAMPLES];
int SMOOTHarrayay [FILTERSAMPLES];
int SMOOTHarrayaz [FILTERSAMPLES];
int SMOOTHarraygx [FILTERSAMPLES];
int SMOOTHarraygy [FILTERSAMPLES];
int SMOOTHarraygz [FILTERSAMPLES];
int MEDIAarrayax [TAMMEDIA];
int MEDIAarrayay [TAMMEDIA];
int MEDIAarrayaz [TAMMEDIA];
int MEDIAarraygx [TAMMEDIA];
int MEDIAarraygy [TAMMEDIA];
int MEDIAarraygz [TAMMEDIA];

int acxzero = 0;
int acyzero = 0;
int aczzero = 0;
int gyxzero = 0;
int gyyzero = 0;
int gyzzero = 0;
int RAW_m_ax = 0;
int RAW_m_ay = 0;
int RAW_m_az = 0;
int RAW_m_gx = 0;
```

```
int RAW_m_gy = 0;
int RAW_m_gz = 0;

int RAW_media_ax = 0;
int RAW_media_ay = 0;
int RAW_media_az = 0;
int RAW_media_gx = 0;
int RAW_media_gy = 0;
int RAW_media_gz = 0;

//variables de prueba
float m11=0;
float m22=0;
float m33=0;
float m44=0;

float m1_ant = 0;
float m2_ant = 0;
float m3_ant = 0;
float m4_ant = 0;

float gxx;
float gyy;

int contador=0;


void setup(){
  //Configuración I/0
  pinMode(MOTOR1,OUTPUT);
  pinMode(MOTOR2,OUTPUT);
  pinMode(MOTOR3,OUTPUT);
  pinMode(MOTOR4,OUTPUT);
  pinMode(LEDPIN,OUTPUT);
  digitalWrite(LEDPIN,LOW);
  Serial.begin(BAUDIOS);
  digitalWrite(LEDPIN,HIGH);
  //Asegurar motores a 0 nada mas empezar /  seguridad
  analogWrite(MOTOR1,255);
  analogWrite(MOTOR2,255);
  analogWrite(MOTOR3,255);
  analogWrite(MOTOR4,255);
  delay(1000);
  digitalWrite(LEDPIN,LOW);
  delay(4000);
  digitalWrite(LEDPIN,HIGH);
  delay(1000);
  digitalWrite(LEDPIN,LOW);
  delay(5000);
  //Conseguir los nivels 0 de acelerometros y Gyros
  zeroGyros();
  zeroAcel();
  m1=0;
  m2=0;
  m3=0;
  m4=0;
  pot=0;
  digitalWrite(LEDPIN,HIGH);

}
```

```
void loop(){
 read_serial();
 read_sensors(1);
 media(0);                          // 1 no hace media 0 si que hace media
 control();

 if(pot != 0){
 m1 = pot + m11;
 m2 = pot + m22;
 m3 = pot + m33;
 m4 = pot + m44;}
 else{
   m1 = pot;
   m2 = pot;
   m3 = pot;
   m4 = pot;}

 m1 = constrain (m1,0,255);
 m2 = constrain (m2,0,255);
 m3 = constrain (m3,0,255);
 m4 = constrain (m4,0,255);

 analogWrite(MOTOR1,255 – m1);
 analogWrite(MOTOR2,255 – m2);
 analogWrite(MOTOR3,255 - m3);
 analogWrite(MOTOR4,255 - m4);

if(contador > 2000){
  Serial.print("valor: ");
  Serial.print(valor);
  Serial.print("   maxg: ");
  Serial.println(maxg);
  contador = 0;}

contador ++;
}


////////////////SUBRUTINAS//////////////////////////////////////////////////////////////////

void control(){
  float b;
  float a;

  //////////////        Y     ////////////////////////
  a = acxzero - RAW_media_ax;
  b = acyzero - RAW_media_ay;
  gxx =constrain(abs(float(gyxzero) - float(RAW_media_gx))/valor,0.1,maxg);
  gyy =constrain(abs(float( gyyzero) - float(RAW_media_gy))/10.4,0.1,47);


    ////////////////////////////  X ////////////////////////
  if (a>0+margen){
    m22 = (a/gxx);
    m11 = 0;
  }
  else if(a<0-margen){
    m22 = 0;
    m11 = (-a/gxx);}

  ////////////////////////////  Y ////////////////////////
  if (b>0+margen){
    m44 = (b/gyy);
    m33 = 0;
  }
  else if(b<0-margen){
    m44 = 0;
    m33 = (-b/gyy);}
```

```
}
void media(int op){
  int i;
  int sumax;
  int sumay;
  int sumgx;
  int sumgy;

  if(op == 1){
    RAW_m_ax = RAW_media_ax;
    RAW_m_ay = RAW_media_ay;
    RAW_m_gx = RAW_media_gx;
    RAW_m_gy = RAW_media_gy;}

  else
  {
    ////////////////////     AC      ////////////////////
     for (i=0; i< TAMMEDIA-1; i++) {
       MEDIAarrayax[TAMMEDIA-i-1] = MEDIAarrayax[TAMMEDIA-i-2];
       MEDIAarrayay[TAMMEDIA-i-1] = MEDIAarrayay[TAMMEDIA-i-2];
       MEDIAarraygx[TAMMEDIA-i-1] = MEDIAarraygx[TAMMEDIA-i-2];
       MEDIAarraygy[TAMMEDIA-i-1] = MEDIAarraygy[TAMMEDIA-i-2];
     }
  MEDIAarrayax[0]=RAW_m_ax;
  MEDIAarrayay[0]=RAW_m_ay;
  MEDIAarraygx[0]=RAW_m_gx;
  MEDIAarraygy[0]=RAW_m_gy;
  sumax=0;
  sumay=0;
  sumgx=0;
  sumgy=0;

  for (i=0; i< TAMMEDIA; i++) {
   sumax = sumax + MEDIAarrayax[TAMMEDIA-i-1];
   sumay = sumay + MEDIAarrayay[TAMMEDIA-i-1];
   sumgx = sumgx + MEDIAarraygx[TAMMEDIA-i-1];
   sumgy = sumgy + MEDIAarraygy[TAMMEDIA-i-1];
   }

  RAW_media_ax = sumax / TAMMEDIA;
  RAW_media_ay = sumay / TAMMEDIA;
  RAW_media_gx = sumgx / TAMMEDIA;
  RAW_media_gy = sumgy / TAMMEDIA;

}
}

void read_sensors(int op){              //si op == 1   RAW_m_xx = dato sin pasar por SMOOTH
  RAWax = analogRead(ACXPIN)*0.7 + RAWax*0.3;
  RAWay = analogRead(ACYPIN)*0.7 + RAWay*0.3;
 // RAWaz = analogRead(ACZPIN)*0.7+RAWaz*0.3;
  RAWgx = analogRead(GYXPIN);
  RAWgy = analogRead(GYYPIN);
  RAWgz = analogRead(GYZPIN);

  if (op==1){
    RAW_m_ax=RAWax;
    RAW_m_ay=RAWay;
  //  RAW_m_az=RAWaz;
    RAW_m_gx=RAWgx;
    RAW_m_gy=RAWgy;
    RAW_m_gz=RAWgz;}

  else {
      SMOOTHax = digitalSmooth(RAWax,SMOOTHarrayax);
      RAW_m_ax=SMOOTHax;
      SMOOTHay = digitalSmooth(RAWay,SMOOTHarrayay);
      RAW_m_ay=SMOOTHay;
      SMOOTHay = digitalSmooth(RAWay,SMOOTHarrayay);
      RAW_m_ay=SMOOTHay;
```

```
      SMOOTHaz = digitalSmooth(RAWaz,SMOOTHarrayaz);
      SMOOTHaz = digitalSmooth(RAWaz,SMOOTHarrayaz);
      RAW_m_az=SMOOTHaz;
      SMOOTHgx = digitalSmooth(RAWgx,SMOOTHarraygx);
      RAW_m_gx=SMOOTHgx;
      SMOOTHgy = digitalSmooth(RAWgy,SMOOTHarraygy);
      RAW_m_gy=SMOOTHgy;
      SMOOTHgz = digitalSmooth(RAWgz,SMOOTHarraygz);
      RAW_m_gz=SMOOTHgz;}

}
void read_serial(){
  val = Serial.read();

  if(val == 'x'){
                    m1 = 0;
                    m2 = 0;
                    m3 = 0;
                    m4 = 0;
                    pot = 0;}

  else if((val == 'S') & (pot <= 255-inc))  pot = pot + inc;
  else if((val == 's') & (pot >= inc))   pot = pot - inc;
  else if(val == 'A') valor = valor - 0.2;
  else if(val == 'a') valor = valor + 0.2;
  else if(val == 'B') maxg = maxg - 0.2;
  else if(val == 'b') maxg = maxg + 0.2;
}

void zeroGyros(){
 byte i;
 gyxzero=0;
 gyyzero=0;
 gyzzero=0;

   for(i=0; i<TAMZERO; i++)  gyxzero += analogRead(GYXPIN);
  gyxzero = gyxzero / TAMZERO;
   for(i=0; i<TAMZERO; i++)  gyyzero += analogRead(GYYPIN);
  gyyzero = gyyzero / TAMZERO;
   for(i=0; i<TAMZERO; i++)  gyzzero += analogRead(GYZPIN);
  gyzzero = gyzzero / TAMZERO;}

void zeroAcel(){
   byte i;
   acxzero=0;
   acyzero=0;
   aczzero=0;
   for(i=0; i<TAMZERO; i++)  acxzero += analogRead(ACXPIN);
  acxzero = acxzero / TAMZERO;
   for(i=0; i<TAMZERO; i++)  acyzero += analogRead(ACYPIN);
  acyzero = acyzero / TAMZERO;
   for(i=0; i<TAMZERO; i++)  aczzero +=  analogRead(ACZPIN);
  aczzero = aczzero / TAMZERO;}


//http://www.arduino.cc/playground/Main/DigitalSmooth
int digitalSmooth(int rawIn, int *sensSmoothArray) {
  // "int *sensSmoothArray" passes an array to the function - the asterisk indicates the array
name is a pointer
  int j, k, temp, top, bottom;
  long total;
  static int i;
  static int sorted[FILTERSAMPLES];
  boolean done;

  i = (i + 1) % FILTERSAMPLES; // increment counter and roll over if necc. -  % (modulo
operator) rolls over variable
  sensSmoothArray[i] = rawIn; // input new data into the oldest slot
```

```
  for (j=0; j<FILTERSAMPLES; j++){ // transfer data array into anther array for sorting and
averaging
    sorted[j] = sensSmoothArray[j];
  }

  done = 0;                // flag to know when we're done sorting
  while(done != 1){        // simple swap sort, sorts numbers from lowest to highest
    done = 1;
    for (j = 0; j < (FILTERSAMPLES - 1); j++){
      if (sorted[j] > sorted[j + 1]){     // numbers are out of order - swap
        temp = sorted[j + 1];
        sorted [j+1] =  sorted[j] ;
        sorted [j] = temp;
        done = 0;
      }
    }
  }
  // throw out top and bottom 15% of samples - limit to throw out at least one from top and
bottom
  bottom = max(((FILTERSAMPLES * 15) / 100), 1);
  top = min((((FILTERSAMPLES * 85) / 100) + 1), (FILTERSAMPLES - 1));   // the + 1 is to make
up for asymmetry caused by integer rounding
  k = 0;
  total = 0;
  for ( j = bottom; j< top; j++){
    total += sorted[j];  // total remaining indices
    k++;
  }
  return total / k;    // divide by number of samples
}
```