Facultad de Ingeniería de Sistemas y Electrónica

Carrera Profesional de Ingeniería Electrónica

IMPLEMENTACIÓN DE UN SISTEMA DE DETERMINACIÓN DE ORIENTACIÓN MEDIANTE DISEÑO SOPC EN UNA FPGA PARA VEHÍCULO AÉREO NO TRIPULADO DEL TIPO QUADROTOR

Tesis para la obtención del Título Profesional de

Ingeniero Electrónico

AUTOR:

RONNY MICHAEL HUERTA FIRMA

ASESOR:

ING. JUAN SANTIAGO VEGA MARTÍNEZ

LIMA - PERÚ

2015

Dedicatoria

Esta tesis se la dedico a mis padres Belsi y Moisés por todo su apoyo, consejos y por quienes ahora veo realizado uno de mis sueños. A mi hermano Anthony, quien a pesar de su corta edad, me enseña lo que es responsabilidad y perseverancia. A Katherine, por su comprensión, paciencia y animarme siempre a seguir adelante. A todos mis compañeros que me ayudaron cuando más lo necesitaba.

Agradecimientos

Principalmente a Dios, por bendecirme y brindarme una vida llena de aprendizajes y experiencias en compañía de mis familiares.

A mis padres, por todos sus consejos que me permitieron poder superar muchos momentos difíciles, pero sobre todo gracias por todo el amor que me dan.

A todos mis compañeros de la universidad, en especial a mis compañeros del proyecto Quadrotor, Julio Mejía, Carlos Quispe, Eduardo Pillco y Luis Rueda, que gracias al trabajo en conjunto se lograron los mejores resultados.

A Ricardo Campos, quien me apoyó en el trabajo tan complicado de calibración.

Al profesor Eleazar Sal y Rosas Celi, quien inició el proyecto y por quien ahora podemos ver los anhelados resultados.

Un especial agradecimiento al profesor Miguel Chicchon Apaza por su amistad, enseñanzas y paciencia en el desarrollo del presente trabajo, y cuyo apoyo me ha permitido alcanzar los objetivos de esta tesis.

A mi asesor el Ing. Juan Vega Martínez por su orientación y lecciones para la culminación de esta tesis y así poder subir un escalón más hacia la meta.

Al profesor Alberto Alvarado Rivera por todo su apoyo al proyecto Quadrotor.

Al Ing. Miguel Risco Castillo, por sus instrucciones y permitirnos el uso de la Arquitectura de Bus Simple (SBA), cuyo aporte fue de mucha utilidad en el proyecto.

"No hay distancia que no se pueda recorrer ni meta que no se pueda alcanzar."

– Napoleón Bonaparte.

Resumen

En la actualidad, los UAV (Vehículos Aéreos no Tripulados o conocidos también como Drones) se desarrollan con interés y en diversos ámbitos; debido a sus diversas áreas de aplicaciones como en la agricultura, reconocimiento en zonas de desastres, espionaje militar, etc. El sistema de control más importante de un UAV es el de orientación ya que sin éste no se lograría otro tipo de control como el de velocidad y posición. Para controlar la orientación es necesario un sistema de determinación. Debido a que los sistemas de determinación convencionales son costosos y pesados, en este trabajo se aborda este problema al desarrollar un sistema de determinación de orientación para un UAV del tipo Quadrotor utilizando sensores de tecnología MEMS de bajo costo y pequeño tamaño.

El sistema fue implementado en un dispositivo de lógica reconfigurable (FPGA) mediante dos módulos: hardware y software. En el módulo hardware se implementa el sistema SBA usando el lenguaje de descripción de hardware (VHDL) para la comunicación con los sensores. El módulo software se ejecuta en el procesador embebido SOPC NIOSII, el cual procesa los algoritmos de calibración y fusión de sensores (Filtro de Kalman Extendido) para la estimación de orientación.

Se compararon tres diferentes algoritmos de determinación de orientación: un filtro de Kalman lineal y dos filtros de Kalman Extendidos (EKF). A través de las pruebas se demostró que el último algoritmo Filtro de Kalman Extendido empleado una Matriz de Rotación para la estimación de los ángulos de Euler es el de mejor rendimiento para el Quadrotor.

Palabras claves: Quadrotor, Filtro de Kalman Extendido, tecnología MEMS, Sensores Inerciales, SOPC, VHDL, FPGA, Calibración, DCM, Cuaterniones, Ángulos de Euler.

Abstract

At present, UAVs (Unmanned Aerial Vehicles or also known as Drones) are developed in various fields because of its various application areas such as agriculture, recognition in disaster zones, military espionage, etc. The most important control is control attitude because without it no other control is achieved as the speed and position. To control the orientation requires a determination system. Conventional determination systems are expensive and heavy because in this thesis is developed a system for determining the orientation for a UAV type Quadrotor using low cost MEMS sensors of small size.

The system was implemented in a reprogrammable logic device (FPGA) by two modules: hardware and software. In hardware module the SBA system is implemented using hardware description language (VHDL) to communicate with the sensors. In software module running on the embedded processor SOPC NIOSII which processes the calibration algorithms and sensor fusion (Extended Kalman filter) to estimate orientation.

Three different determination algorithms were compared: A Kalman filter and two linear Extended Kalman filter (EKF). The tests showed that the latter Extended Kalman Filter algorithm employed a rotation matrix for estimation of the Euler angles is the best performance for Quadrotor.

Keywords: Quadrotor, Extended Kalman Filter, MEMS Technology, Inertial Sensors, SOPC, VHDL, FPGA, Calibration, DCM, Quaternions, Euler angles.

Índice General

Índice de Figuras

Figura **Página**

Índice de Tablas

Tabla **Página**

CAPÍTULO 1

INTRODUCCIÓN

Un Quadrotor es un vehículo aéreo no tripulado (UAV, por sus siglas en inglés de Unmanned Aerial Vehicle) de tamaño medio, conocido también como Drone; éste es propulsado por cuatro rotores que le permiten despegar y aterrizar verticalmente, además de una gran maniobrabilidad. En la actualidad, sus aplicaciones son múltiples y se desarrolla en diversos ámbitos como: la agricultura, reconocimiento en zonas de desastres, espionaje militar, entre otros. En la figura 1.1, se muestra el Quadrotor del proyecto.

En el diseño de control de un UAV, el control de orientación o actitud es el más importante, ya que sin éste no es posible estabilizarlo en el vuelo debido a perturbaciones externas como las ráfagas de viento, tampoco se podría realizar otro tipo de control como el de trayectoria. Para realizar el control de actitud del Quadrotor es fundamental sensar su movimiento y determinar su orientación en el espacio, el cual es el objetivo central de esta tesis.

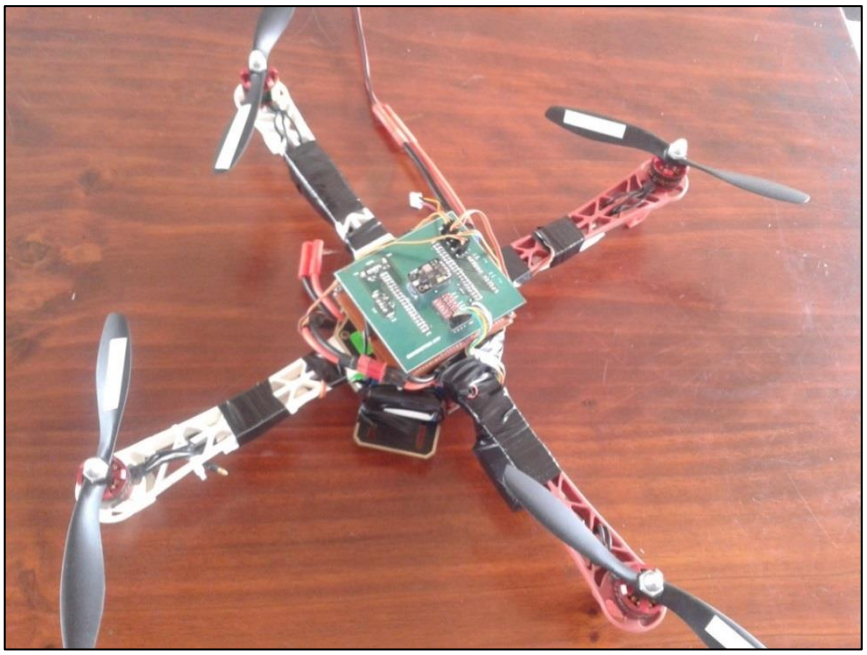

Figura 1.1: UAV-Quadrotor del proyecto[1]

La arquitectura de control implementado en este proyecto es un Sistema de Control Embebido (ECS, Embedded Control System), como se muestra en la figura 1.2. El movimiento del Quadrotor es sensado por la IMU (Acelerómetro, giroscopio y magnetómetro). Luego se realiza el control de actitud en el sistema embebido conformado por un dispositivo de lógica reconfigurable (FPGA), que captura la información de la IMU y recepciona los puntos de consigna para los ángulos de vuelo del mando RC, y para generar la señal de control. La señal de control se convierte a señales de modulación por ancho de pulso (PWM) mediante los ESC, que son los controladores de velocidad de los motores sin escobillas (brushless motor) del Quadrotor. Adicionalmente los datos obtenidos son enviados al sistema en tierra y son mostrados en un computador.

[1] UAV-Quadrotor utilizado en el proyecto. Para un mejor detalle de su estructura ver anexo D.

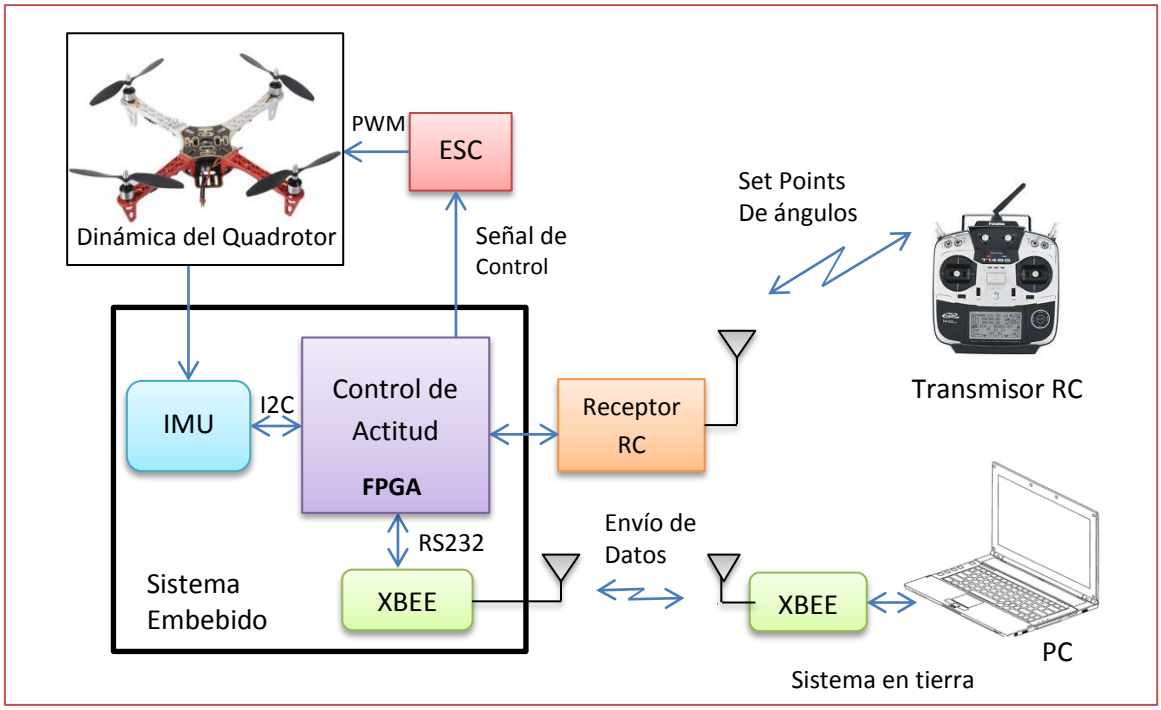

Figura 1.2: Arquitectura del sistema de control embebido

El presente trabajo se enfoca en el desarrollo de un sistema de determinación de orientación para el Quadrotor como se muestra en la figura 1.3. En este trabajo se desarrollan los siguientes temas:

- Detección de datos mediante los sensores.
- Obtención de los parámetros de calibración.
- Calibración de los sensores.
- Elaboración del algoritmo de fusión (Filtro de Kalman Extendido).

Existen diferentes productos comerciales robustos de determinación de orientación, como el ADIS16480 del fabricante Analog Devices, o el MTi-30 del fabricante Xsens, éstos son de elevado costo del orden de miles de dólares, por ello se ha realizado un diseño propio.

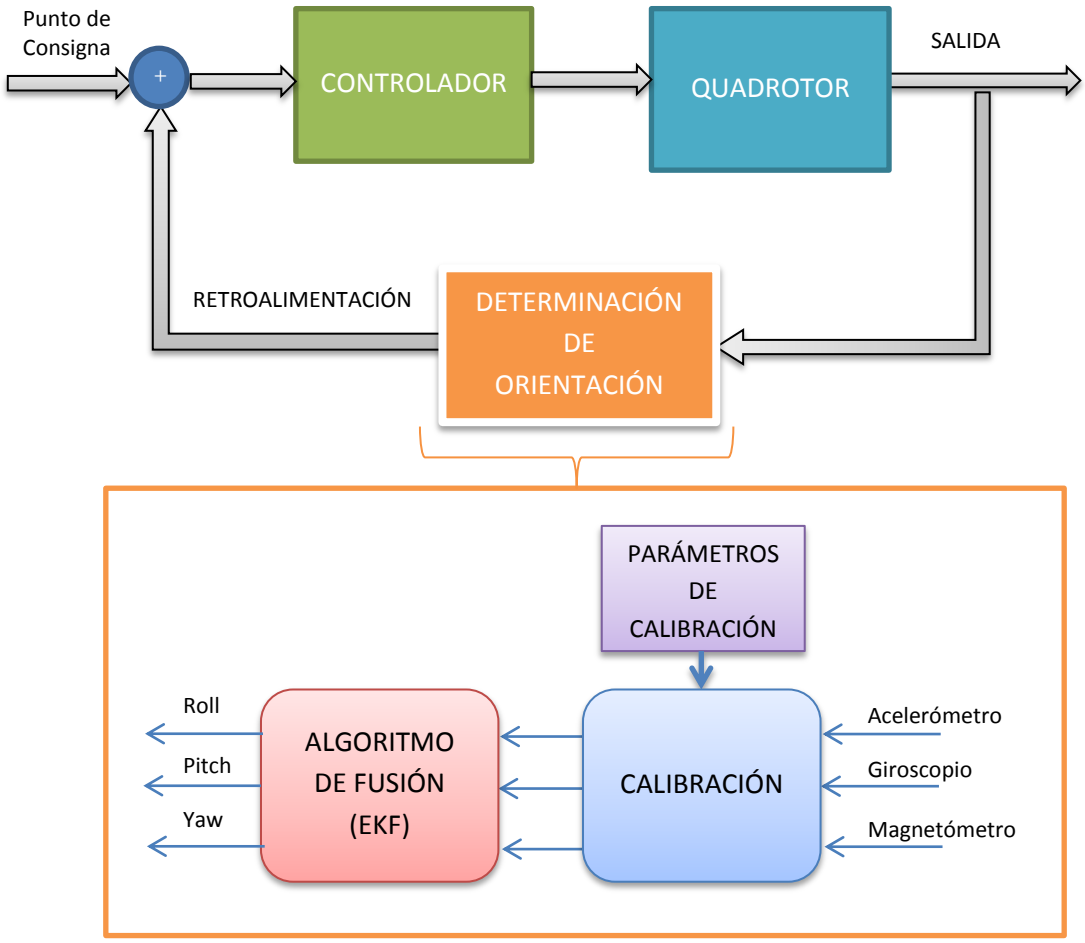

Figura 1.3: Delimitación del proyecto

Debido a la limitada carga útil del Quadrotor es necesario que este sistema sea de tamaño pequeño, peso ligero y de gran precisión, por ello que se utilizan sensores tipo MEMS (Sistemas Micro-Electromecánicos), que son de bajo costo. También se utiliza la tarjeta de desarrollo DE0-Nano con un FPGA Cyclone IV de la marca Altera.

Para determinar los parámetros de calibración se ha diseñado y construido una plataforma mecánica, en la cual se implementó el control de velocidad del motor DC en un FPGA. La precisión de los resultados obtenidos aumentó significativamente.

En el capítulo 1 de la tesis se inicia con una introducción al tema de investigación, luego se plantean los objetivos de la tesis, así como su justificación y el estado de arte. En el capítulo 2 se presenta un análisis de los alcances teóricos. En el capítulo 3 se muestran los diseños, las simulaciones y la implementación del sistema de determinación de orientación utilizando el FPGA. En el capítulo 4 se muestran los resultados de la comparación entre los tres algoritmos de fusión de sensores desarrollados, así como el descripción del sistema hardware-software. Finalmente, en capítulo 5 se enumeran y describen las conclusiones y los posibles trabajos futuros de la investigación realizada en la presente Tesis.

1.1 Descripción del Problema

Los UAV, como es el Quadrotor, actualmente están teniendo una mayor participación en el área civil y militar, en diversas aplicaciones. Por ejemplo se podría realizar vigilancia en un área sumamente peligrosa para un ser humano, en labores de búsqueda y rescate, agricultura, etc. (Nonami, Kendoul, Suzuki, Wang, & Nakazawua, 2010). Disminuyendo significativamente los costos en estas aplicaciones, en comparación respecto de utilizar vehículos tripulados como helicópteros, ya que éstos consumen combustible, requieren de un piloto y además tienen un costo elevado de compra o alquiler.

Para controlar al UAV autónomamente es fundamental sensar su movimiento y determinar la orientación o actitud en el espacio en tres dimensiones, tema en el cual se centra la Tesis. Debido a la dinámica del Quadrotor y la necesidad de enlazarse con diferentes sensores como son: el acelerómetro, giroscopio, magnetómetro para el sistema de determinación de orientación, y adicionalmente con un GPS, cámara de video, rayos ultrasonidos, infrarrojos entre otros; por ello es necesario utilizar un dispositivo de alta capacidad de procesamiento y rendimiento, por lo que se consideró conveniente, utilizar un dispositivo de lógica reconfigurable (FPGA), debido que permite implementar

módulos de hardware utilizando un lenguaje de descripción de hardware (VHDL), además de uno o más procesadores embebidos para el desarrollo de algoritmos complejos.

Debido a la limitada carga útil del Quadrotor otro requisito indispensable es que los sensores deben ser de pequeño tamaño y ligeros de peso, por lo que se ha elegido que sean de tecnología MEMS (sistemas Micro-Electromecánicos) con un tamaño de menos de medio centímetro. En contra parte estos sensores MEMS tienen diversas deficiencias como su poca precisión y sensibilidad al ruido, que deben ser compensados mediante un proceso de calibración y del uso de un algoritmo óptimo de fusión. Si bien existen sistemas de determinación comerciales, con sensores de gran precisión y rendimiento, éstos pueden ser de gran tamaño o de costos elevados en varios miles de dólares, no siendo adecuados para este proyecto.

1.2 Objetivos

1.2.1 Objetivo general

- Implementar un sistema de determinación de orientación de pequeño tamaño, liviano y de bajo costo para un Quadrotor, que sea robusto frente a perturbaciones externas como las vibraciones e interferencias magnéticas, utilizando para ello un dispositivo de lógica reconfigurable (FPGA).

1.2.2 Objetivos específicos

- Obtener los datos procedentes de los dos sensores inerciales (acelerómetro, giroscopio) además del magnetómetro, utilizando el Protocolo de Comunicaciones I^2C en un lenguaje de descripción de hardware (VHDL).
- Diseñar un entorno SOPC (System-on-a-Programmable-Chip) con el procesador NIOSII embebido en el FPGA Cyclone IV de la tarjeta DE0-Nano. En donde se genere un módulo hardware y otro software. En el módulo hardware se

implementa el protocolo I2C en VHDL y el módulo software ejecute en el procesador NIOS II los algoritmos de calibración y fusión de sensores (EKF).

- Diseñar y construir una plataforma de calibración, para la obtención de los parámetros de calibración de los sensores.
- Implementar un control de velocidad para el motor DC de la plataforma de calibración, necesario para la calibración del giroscopio.
- Desarrollar y comparar diferentes tipos de algoritmos de determinación de orientación o fusión de sensores.
- Validar las implementaciones, mediante simulaciones computacionales además de pruebas reales con el Quadrotor.

1.3 Hipótesis

Para controlar de manera autónoma el vuelo de un Vehículo Aéreo no Tripulado como el Quadrotor, es fundamental un sistema de determinación de orientación robusto ante perturbaciones como las vibraciones e interferencias magnéticas. Además, debido a su limitada capacidad de carga útil, este sistema debe ser de peso ligero, pequeño y de un bajo costo para su viabilidad. Por lo tanto se plantea como hipótesis que para cumplir con los requerimientos establecidos se utilicen sensores del tipo MEMS (sistemas micro-electromecánicos), junto a un algoritmo calibración y de fusión de datos de los sensores (Filtro de Kalman Extendido), implementando en un dispositivo de lógica reconfigurable (FPGA) que tiene una alta capacidad de procesamiento, implementando dos módulos: hardware y software. El primer módulo implementado en un lenguaje de descripción de hardware (VHDL) y el segundo ejecutado por un procesador Sof-Core embebido, optimizando de esta manera el sistema.

1.4 Justificación

El correcto control y estabilización en vuelo de un Quadrotor, se logra con un autopiloto capaz de contrarrestar perturbaciones como las ráfagas de viento. Para lograr esto es necesario, un sistema de determinación de orientación que obtiene en tiempo real y en tres dimensiones el movimiento del vehículo, utilizando sensores inerciales (acelerómetro, giroscopio) además de un magnetómetro. Controlando la estabilidad en vuelo del Quadrotor, éste puede ser utilizado en diversas aplicaciones, brindando ayuda en diversos problemas existentes en la sociedad. Por ejemplo, mediante una cámara abordo se podrían realizar labores de búsqueda y rescate en zonas peligrosas o de difícil acceso para un ser humano, o en el control de obras, tránsito vehicular, control de multitudes, vigilancia de fronteras, en general las aplicaciones son múltiples.

El sistema de determinación de orientación se implementó en un dispositivo de lógica reconfigurable (FPGA) ya que asegura el tiempo necesario de estabilización del Quadrotor, por su gran capacidad de procesamiento y la posibilidad de generar hardware dedicado a tareas específicas las cuales se ejecutan de manera paralela. Por otro lado, los algoritmos como el de calibración y el Filtro de Kalman Extendido se ejecutan en un procesador Sof-Core implementado en el FPGA.

Si bien se podría utilizar un sistema de determinación de orientación comercial, los precios de estos sistemas son elevados, además al realizar un diseño propio no sólo se abaratan los costos, sino también genera la posibilidad de corregir errores, realizar mejoras y obtener un control adaptado a la aplicación del UAV.

Con este primer trabajo el sistema de determinación se centra en un UAV del tipo Quadrotor, también se podría implementar en otras aplicaciones sin considerables modificaciones, como pueden ser: entornos de realidad virtual, misiles guiados, robots, vehículos espaciales, entre otros.

1.5 Estado del Arte

1.5.1 Historia del Quadrotor y sus aplicaciones actuales

Un Quadrotor es un helicóptero propulsado por cuatro rotores, éste se controla variando la velocidad de los rotores aumentando o disminuyendo el empuje; son bastante maniobrables y de gran estabilidad.

Los primeros diseños de Vehículos tipo Quadrotor se realizaron en las décadas de los años 1920 como el Oechmichen N°2 (figura 1.4), hecho por el ingeniero francés Oechmichen, el cual llevaba a bordo a una persona con un despegue y aterrizaje exitoso. Otro de los primeros prototipos fue el Quadrotor De Bothezat (figura 1.5) del ejército de los Estados Unidos el cual se elevó a unos 15 pies de altura, pero debido a que los resultados no fueron los esperados no se prosiguieron con las investigaciones.

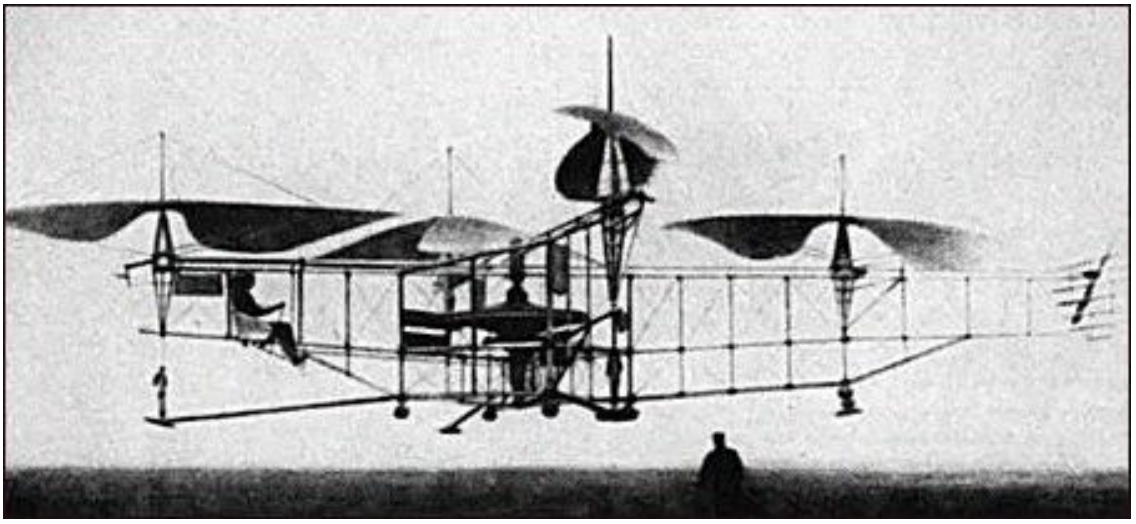

Figura 1.4: Oechmichen N°2, año 1922

Figura 1.5: Prototipo de Quadrotor De Bothezat, 1923

Los Quadrotores actuales existentes en el mercado son más pequeños como el Parrot AR.Drone (figura 1.6) que se puede controlar desde un dispositivo móvil.

Figura 1.6: Parrot AR.Drone, Quadrotor comercial controlado por un Smartphone

Actualmente los Quadrotores son de tamaño pequeño, y se usan en áreas de investigación militar y civil. En el mercado laboral por ejemplo, una aplicación es para

monitorear el tráfico en tiempo real mediante una cámara y con la información obtenida, realizar un sistema de control de los semáforos para que dé mayor fluidez al tránsito vehicular. Otra aplicación sería apoyar en la búsqueda y rescate de personas luego de una catástrofe o desastre natural o simplemente utilizarlo como medio de entretenimiento.

Algunas plataformas actuales de investigación con diversas estrategias de control, que se desarrollan en diferentes partes de mundo son:

1.5.2 Draganflyer

Es un producto comercial a radio control (figura 1.7), incorpora un microcontrolador como unidad de procesos, tres giroscopios piezoeléctricos que proporcionan la velocidad angular de sistema y amplificadores de potencia para los motores (Mckerrow, 2004). Tiene una longitud aproximada de 30 cm, y de medio kilogramo de peso con una carga útil de 100 gramos.

Figura 1.7: Quadrotor comercial Draganflyer

1.5.3 Universidad de Pensilvania

En esta investigación como primer paso se utilizó una cámara externa y sensores inerciales como el giroscopio para su control y determinación de su orientación; un computador externo recibe y procesa las imágenes de la cámara para luego enviar los datos al Quadrotor (Altug, Ostrowsky, & Taylor, 2004). Luego se le añadió cámaras a bordo con lo cual se lograron mejores resultados, ideales para los despegues y aterrizajes, así como el trabajo cooperativo (Shaojie, Yash, Nathan, & Vijay, 2013). En la figura 1.8, se muestra un Quadrotor elaborado en la universidad.

Figura 1.8: Quadrotor de la Universidad de Pensilvania

1.5.4 Universidad de Cornell

El proyecto se realizó como tesis de maestría (Brian, 2004), el cual es un Quadrotor de 6.2 Kg de peso, con fuente de alimentación y sensores abordo. Para la estimación de la actitud se implementó un Sigma Point Filter (Van der Merwe & Wan, 2004), la imagen del Quadrotor se muestra en la figura 1.9.

Figura 1.9: Quadrotor diseñado en la Universidad de Cornell

1.5.5 Escuela Politécnica Federal de Zúrich

Desarrollado en una universidad de Suiza, donde se estudió el modelo y control de un robot OS4 (figura 1.10), del tipo VTOL con el estudio del despegue y aterrizaje vertical (Bouabdallah, Murrieri, & Siegwart, 2004). El sistema consiste en enviar órdenes desde un computador vía comunicación RS232, luego éste pasa por un adaptador RS232 a I2C, para controlar el sistema. Se implementa un control PID en un microcontrolador PIC16F876 y un filtro de Kalman para la estimación de la orientación.

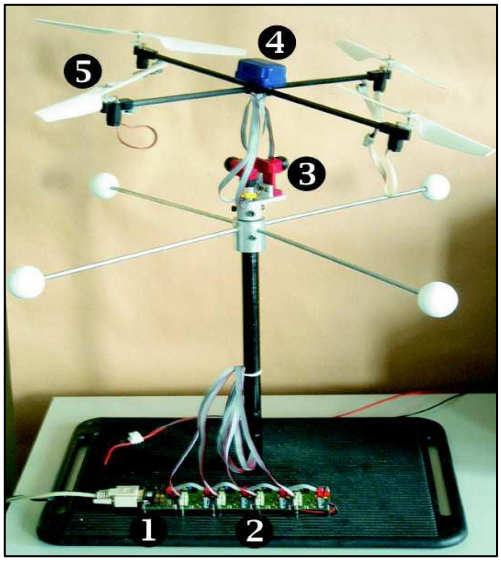

Figura 1.10: Diseño del Quadrotor en la Escuela Politécnica de Zúrich

CAPÍTULO 2

FUNDAMENTO TEÓRICO

2.1 FPGA (Field Programmable Gate Array)

2.1.1 Introducción

Un FPGA (Field Programmable Gate Array) es un circuito integrado digital que está compuesto por una matriz con una gran cantidad de bloques de lógica digital e interconexiones reconfigurables, rodeados además de bloques dedicados de entrada y salida; como se muestra en la figura 2.1 (Maxifield, 2004). Dependiendo de la tecnología de fabricación, algún FPGA sólo pueden ser programadas una vez (OTP, one-time programmable), mientras que otros pueden ser programados de manera múltiple. Los bloques lógicos pueden ser de tres tipos: basados en PLAs (Programmable Logic Array), basados en multiplexores, y basados en TLU (Look-up Table).

El primer fabricante de estos dispositivos fue la empresa Xilinx en 1985, el XC2064; y surgen como una evolución de los CPLDs (del acrónimo inglés, Programmable Logic Device). Hoy en día los FPGA's se han convertido en una industria multimillonaria,

por ejemplo, según un estudio del *Gran View Research*[2], se espera que su mercado alcance los 9,882.5 millones de dólares para el 2020.

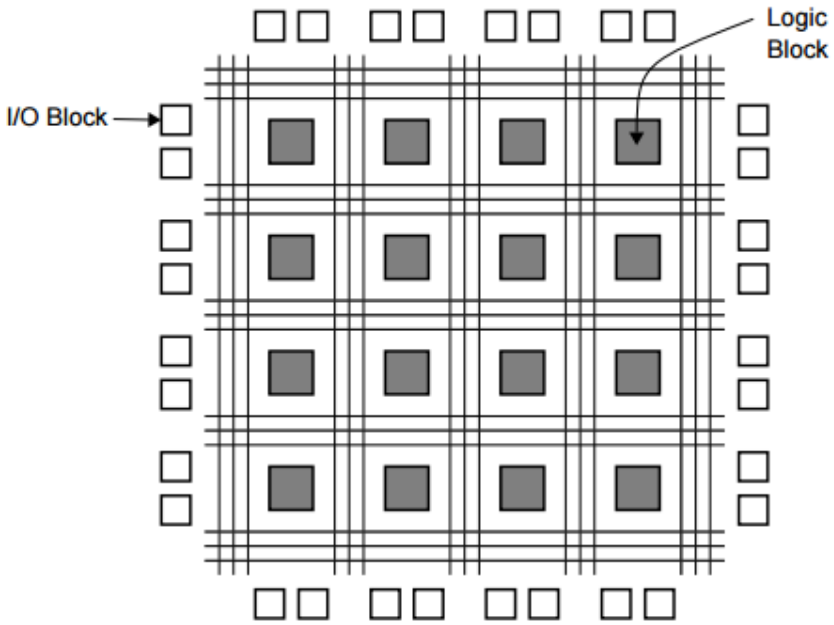

Figura 2.1: Estructura de un FPGA[3]

Mediante los FPGA se pueden implementar diversas aplicaciones específicas, tales como: decodificadores, unidades lógicas aritméticas, redes neuronales (Kumar & Pinjare, 2012), filtros FIR, buses de comunicaciones, procesadores Sof-Core, entre muchos otros.

Debido a sus diversas áreas de aplicaciones, ha abarcado un amplio mercado, como son: Aeroespacial y defensa, electrónica médica, industria automotriz, sistemas de TV digital, servidores, comunicaciones Wireless, instrumentación científica, etc.

Entre las ventajas del uso de estos dispositivos, se tienen:

---

[2] Grand View Research es una empresa de consultoría con sede en EE.UU. que proporciona informes sindicados de investigación, informes de investigación personalizados y servicios de consultoría.
[3] Brown, S., & Rose, J. Architecture of FPGA and CPLDs: A Tutorial, pp. 7-8

- Son reconfigurables: permiten modificar la funcionalidad del diseño, además de implementar funcionalidades personalizadas en hardware, lo que otorga una gran flexibilidad al flujo de diseño.
- Rendimiento: a diferencia de los procesadores, los FPGA realizan diferentes operaciones de manera paralela, esto quiere decir, que a cada tarea de procesos independientes se le asigna una sección dedicada del chip, y puede ejecutarse de manera autónoma sin ser afectada por otros bloques de lógica. En conclusión, el rendimiento de una parte de la aplicación no se ve afectado cuando se agregan otros procesos (National Instruments (NI), 2011).
- Los costos de desarrollo y adquisición son menores para pequeñas cantidades de dispositivos y el tiempo de desarrollo es menor, en comparación a los ASICs (Circuito Integrado de Aplicación Especifica).
- Fiabilidad: Mientras que las herramientas de software ofrecen un entorno de programación, los circuitos de un FPGA son una implementación segura de la ejecución de un programa.

2.1.2 Arquitectura de un FPGA

Existen 3 principales fabricantes en la distribución de FPGA y software de soporte, estos son: Xilinx, Altera y Actel. Pero también se pueden encontrar otros de menor producción como QuickLogic, Cypress, Atmel, Texas Instruments, entre otros. Como ejemplo se van a mostrar las arquitecturas en las FPGA de Xilinx y Altera.

a) Arquitectura del FPGA de Xilinx

Su estructura consta de unos módulos lógicos llamados, CLBs (Configurable Logic Blocks) o en español Bloques Lógicos Configurables, estos contienen circuitos que permiten realizar una amplia gama de funciones lógicas, dentro de estos se encuentran las tablas de búsqueda llamados LookUp Tables – LUTs, que son elementos basados en

memoria RAM. Además de los CLBs, se encuentran los bloques IOBs (Input/Output Blocks) o Bloques de Entrada y Salida que permiten el flujo de datos; de manera que, un pin pueda actuar como salida, entrada o un tercer estado. En la figura 2.2 se puede apreciar la arquitectura del FPGA XC4003E de Xilinx.

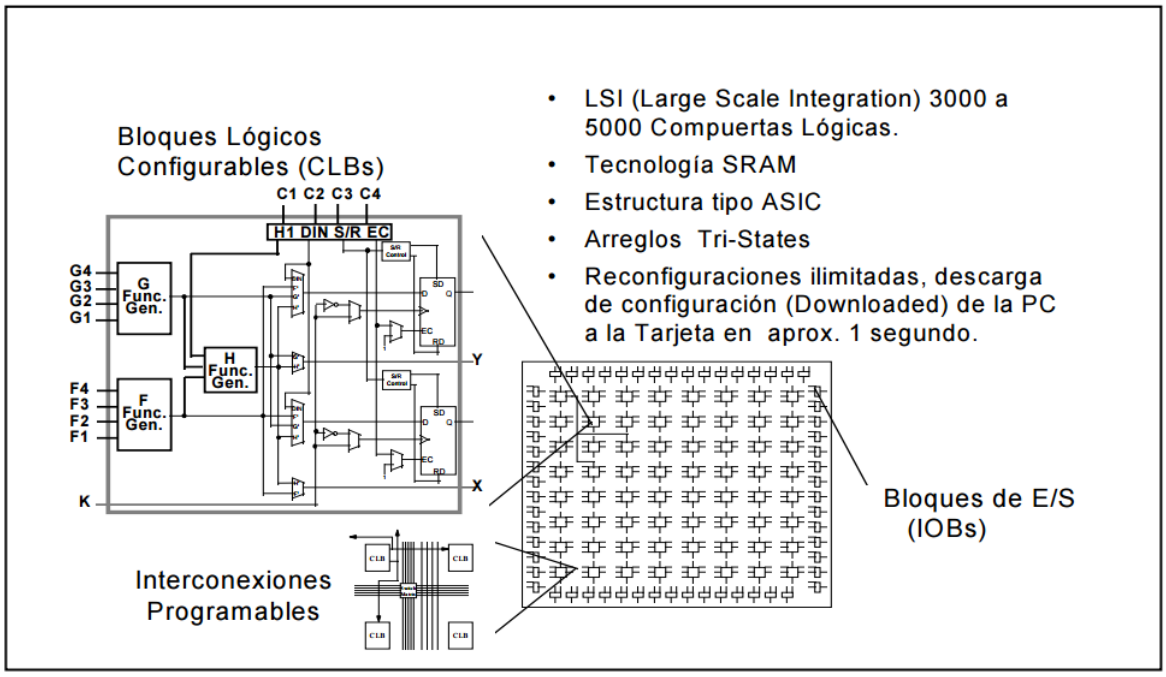

Figura 2.2: Arquitectura de un FPGA XC4003E de Xilinx[4]

b) Arquitectura del FPGA de Altera

La estructura de los FPGA de Altera están compuestos básicamente por ALMs (Adaptive Logic Modules); por ejemplo, en la serie Stratix V un ALM tiene 8 entradas con una tabla de búsqueda o LookUp Table (LUT), dos sumadores integrados y cuatro registros dedicados (Altera Corp., 2015), como se muestra en la figura 2.3.

[4] *Arquitectura FPGA*: http://embebidos-cidetec.com.mx/profesores/jcrls/doctos/fpga_jchl.pdf

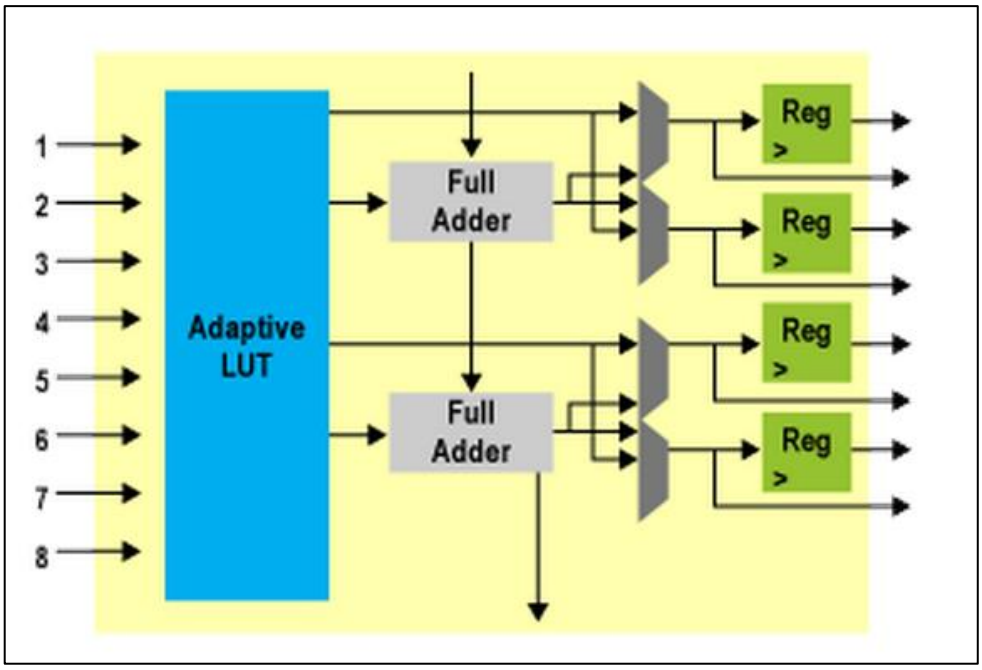

Figura 2.3: Diagrama de bloques de un ALM de la serie Stratix V de Altera[5]

Un ALB (Logic Array Block) está compuesto por estos bloques ALM, que se pueden configurar para implementar registros de funciones, funciones aritméticas y funciones lógicas (Altera Corp., 2015). Altera ofrece diferentes tipos de familias, como se indica en la tabla 2.1.

Recientemente se ha introducido una nueva generación de dispositivos SoC (System on Chip) y SoPC (System on a Programmable Chip) en los FPGA, para mejorar los diseños de sistemas y el rendimiento de los procesos; en el cual, se integra hardware con software, los cuales trabajan para realizar una función específica creando de esta manera un sistema embebido. Una manera de crear este sistema es embeber un núcleo procesador que se pueden implementar en un FPGA de dos maneras: en hardware y software.

[5] *High-Performance FPGA Architecture*:https://www.altera.com/products/general/devices/

Tabla 2.1: Arquitectura de dispositivos Altera[6]

Familia	Estructura del Bloque Lógico	Estructura de Interconexión	Tecnología de programación
Stratix	LUT	Continua	SRAM
Cyclone	LUT	Continua	SRAM
APEX 20K	LUT y termino producto	Continua	SRAM
ACEX 1K	LUT	Continua	SRAM
FLEX 6000	LUT	Continua	SRAM
MAX 9000	Termino producto	Continua	EEPROM
MAX 7000	Termino producto	Continua	EEPROM
MAX 5000	Termino producto	Continua	EPROM
Classic	Termino producto	Continua	EPROM

Procesadores en hardware (Hard-Core): usan un procesador embebido en silicio, por ejemplo se tiene los FPGAs SoC de Altera de la familia Cyclone V con un procesador duro (HPS) que se basa en el procesador ARM Cortex-A9 de núcleo doble.

Procesadores en software (Sof-Core): para implementar la lógica del procesador, usan elementos lógicos programables existentes en el FPGA. Como ejemplos se tienen el NIOS II de Altera, el Microblaze de Xilinx, LatticeMicro32 de la marca Lattice[7].

[6] *Introducción a los dispositivos FPGA*: http://www.ing.unlp.edu.ar/islyd/Trabajo%20Final.pdf
[7] Lattice Semiconductor Corporation, es un fabricante con sede en EE.UU. de dispositivos lógicos programables (FPGAs, CPLDs, SPLDs).

2.1.3 Lenguajes de Descripción de Hardware (HDL)

Un lenguaje de descripción de hardware (HDL, Hardware Description Languaje) es un lenguaje de programación especializado que describe el comportamiento de un circuito o sistema electrónico, y que posteriormente dicho circuito o sistema puede ser implementado físicamente.

A principios de los 80's surgen dos principales lenguajes de descripción de hardware: el VHDL y el Verilog. Estos lenguajes son normalizados por el IEEE[8] y es por ello su importancia (IEEE, 2015).

VHDL surgió del programa VHSIC (Very High Speed Integrated Circuits) del gobierno de los Estados Unidos; habiendo luego, la necesidad de crear un lenguaje estándar que pueda describir su estructura y funcionalidad, es por ello, que se desarrolla el lenguaje VHDL (acrónimo de la combinación de VHSIC y HDL).

El HDL Verilog tuvo su origen como un lenguaje desarrollado en 1983 por la compañía Gateway Design Automation, y a mediados de los 90's se normaliza este lenguaje por la IEEE.

Ambos lenguajes de programación permiten describir la estructura de un diseño y su partición en bloques jerárquicos, así como también, la conexión entre estos bloques. Además hacen posible realizar la simulación del sistema completo o de cada bloque de manera independiente, para finalmente, pasar a su implementación.

Las empresas fabricantes de FPGA proporcionan diseños tanto en software como en hardware para el soporte de sus dispositivos. Mediante estas herramientas se pueden realizar principalmente:

---

[8] IEEE acrónimo en ingles de: Institute of Electrical and Electronics Engineers.

- Una entrada de diseño
- Simulaciones
- Síntesis/Place and Route
- Programación

La empresa Xilinx ofrece el software llamado ISE, mientras que Altera tiene por su lado el software Quartus II (Altera Corp., 2014); también, existen otros desarrolladores aunque de menor popularidad en el mercado, como es el software Libero IDE de la empresa Microsemi.

Un flujo de diseño típico en VHDL con software Quartus II, se muestra en la figura 2.4, en la cual se pueden apreciar los siguientes pasos:

- **Entrada de diseño:** Se refiere al comportamiento o descripción del diseño estructural. Se puede realizar mediante captura esquemática o mediante lenguajes de descripción HDL; como VHDL, Verilog.
- **Síntesis:** Traduce el diseño a las especificaciones primitivas del dispositivo (compuertas lógicas, flip-flops, etc.), así como también, lo optimiza para satisfacer con la área requerida.
- **Place &Route:** Mapea los lugares específicos dentro de la tarjeta con referencia del área y restricciones de rendimiento.
- **Análisis de temporización:** Se verifica si las especificaciones de rendimiento se cumplieron y también se realiza un análisis de tiempo estático.
- **Simulación:** Se verifica el modelo lógico y flujo de datos. Se pueden realizar simulaciones pre-síntesis, post-síntesis y post-Place and Route.
- **Programación y configuración:** Se programa el dispositivo y se prueba su funcionalidad en la tarjeta.

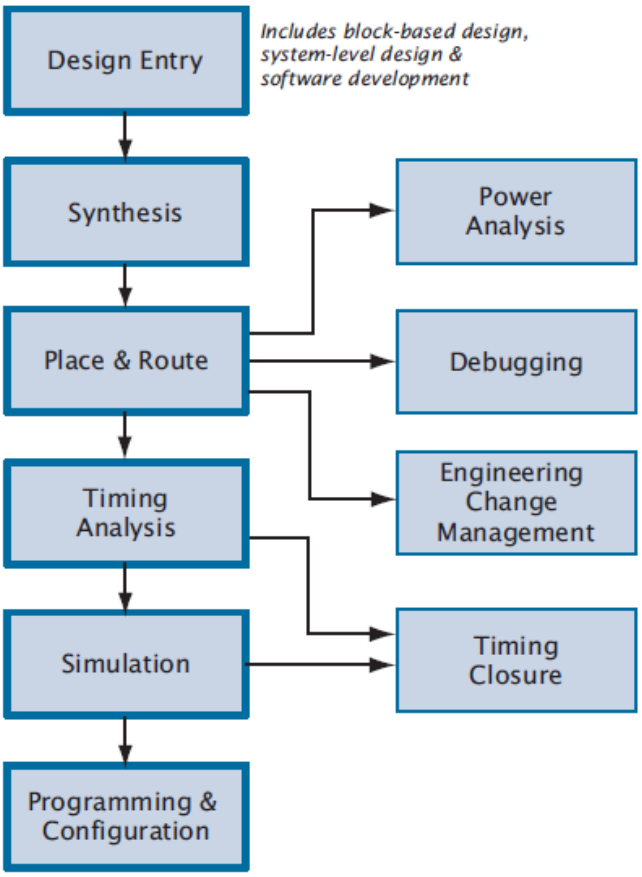

Figura 2.4: Flujo de diseño del software Quartus II[9]

2.1.4 Procesador NIOS II

Como se mencionó en la sección 2.1.2, es posible implementar un sistema embebido en un FPGA, en el cual, hardware y software trabajan en conjunto para poder realizar una función específica. Esto se logra, al embeber un procesador ya sea Hard-Core o Soft-Core. El objetivo de crear este sistema es que el procesador o CPU se encargue del procesamiento de programas secuenciales y de los cálculos, mientras que, la lógica del FPGA se encargue de las interfaces además de facilitar el procesamiento paralelo (Pereyra, Gallia, Alasia, & Micolini, 2013). La empresa Altera cuenta con el

[9] *Introduction to the Quartus II Software:* https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/manual/intro_to_quartus2.pdf

procesador Soft-Core NIOS II, éste es muy eficiente debido a las bondades que proporciona (Altera Corp.), entre sus principales características están:

- Es un procesador con un conjunto de instrucciones de 32 bits para ruta de datos y espacio de direcciones.
- 32 registros de propósito general.
- 32 fuentes de interrupciones.
- Opcional instrucciones de punto flotante.
- Acceso a diversos periféricos e interfaces como memorias.
- Opcional unidad de protección de memoria (MPU), etc.

Un diagrama de bloques del procesador NIOS II se muestra en la figura 2.5:

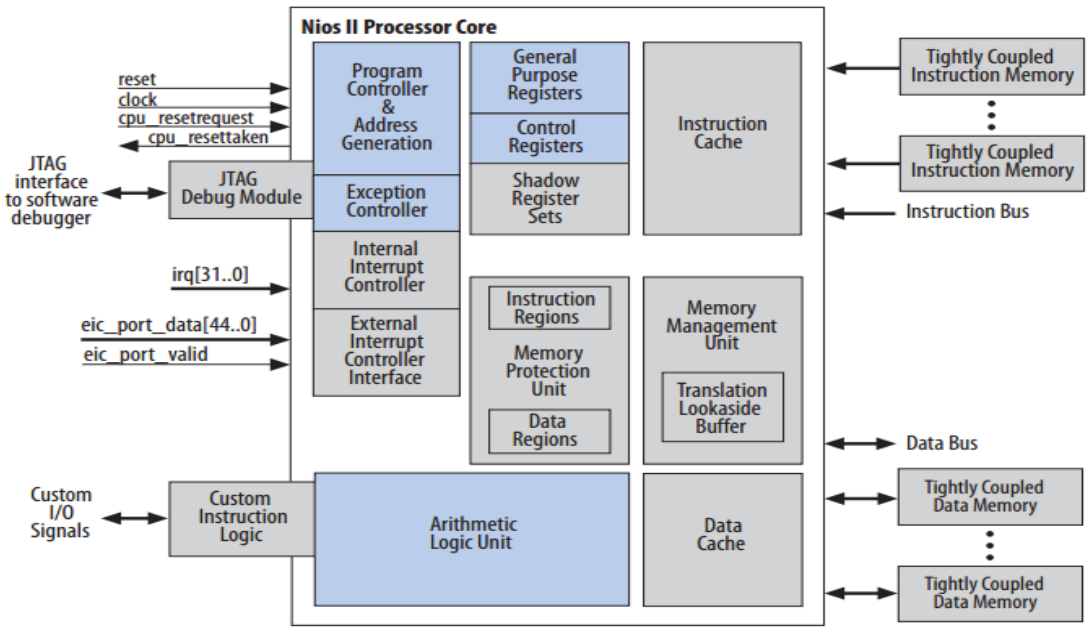

Figura 2.5: Diagrama de bloques del procesador NIOS II[10]

[10] Nios II Processor, http://www.altera.com/literature/hb/nios2/n2cpu_nii5v1.pdf

SOPC Builder

El embeber un procesador dentro de un FPGA permite construir una plataforma de desarrollo SOPC (System on a Programmable Chip), que es un sistema completo en un chip utilizando elementos lógicos programables del FPGA.

El software que permite el desarrollo de este sistema es el SOPC Builder, que es una herramienta del software Quartus II creado por la empresa Altera, esta herramienta incluye librerías y componentes pre-diseñadas; como son: el procesador NIOS II, controladores de memoria, interfaces y periféricos, para poder así crear un sistema completo. Las interconexiones de todos estos componentes se realizan a través del Bus Avalon.

SOPC Builder también maneja automáticamente el arbitraje del bus, la coincidencia del ancho de bus, del dominio del cruce de reloj (CDC, Clock Domain Crossing), entre otros.

En la figura 2.6, se muestra el diagrama de bloques de un diseño mediante el SOPC Builder. Algunos de los componentes que se pueden generar son:

- Procesadores como el NIOS II.
- Periféricos de microcontroladores, timers.
- Interfaces de comunicación serial, tal como el UART.
- Interfaces de propósitos general I/O.
- Periféricos de comunicaciones Ethernet, etc.

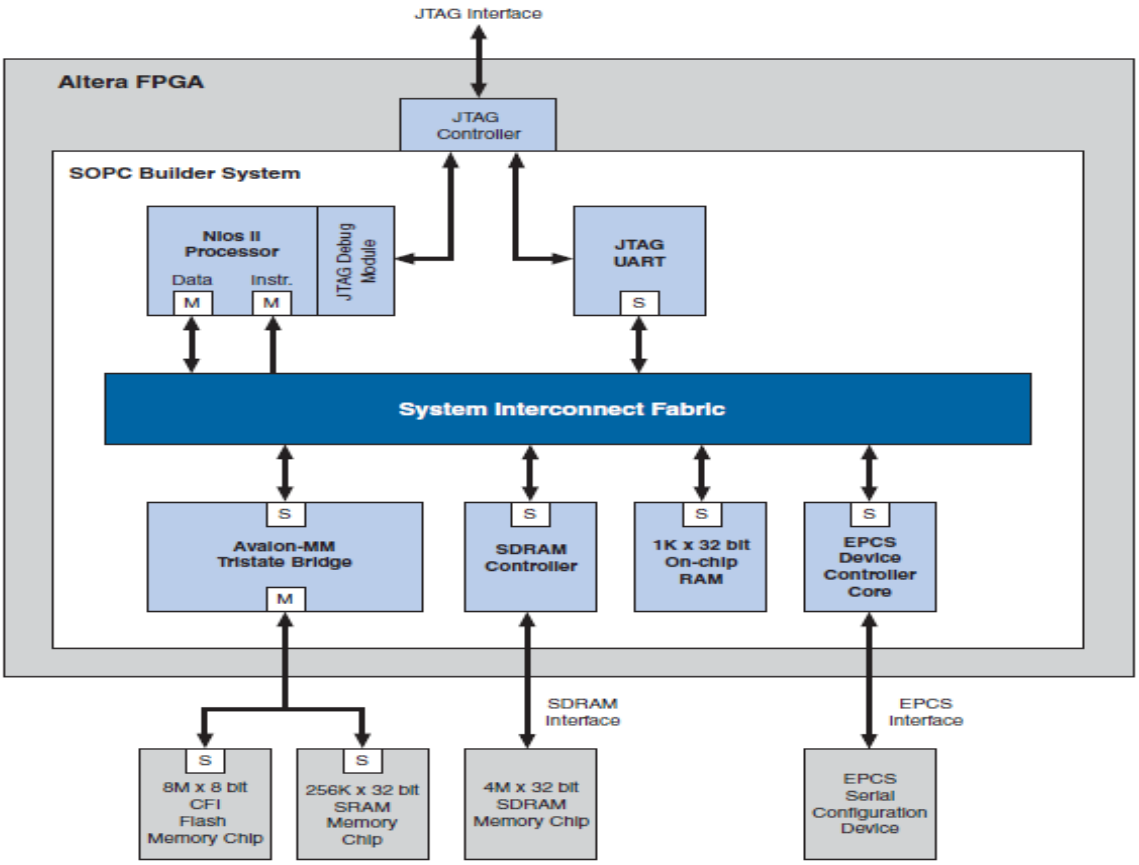

Figura 2.6: Diagrama de bloques de un diseño con el SOPC Builder[11]

Eclipse para NIOS II

Eclipse es un entorno de desarrollo integrado (IDE), multiplataforma y de código abierto. El software Eclipse para el NIOS II es una plataforma que permite realizar tareas como: la creación, edición, ejecución y hasta la depuración de los programas. Se puede programar tanto a bajo nivel (ensamblador), así como también, en alto nivel (C/C++). Este entorno presenta además una interfaz de uso fácil (Altera Corp.).

Para programar el FPGA se utiliza el QUARTUS II Programmer. El archivo que contiene el diseño del hardware tiene como extensión .sof, mientras que, el ejecutable

[11] Processor Architecture, https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/hb/nios2/n2cpu_nii51002.pdf

que se graba en el NIOSII es el archivo .elf; con estos dos archivos se puede grabar el programa. En la figura 2.7, se muestra el entorno de interfaz gráfica de eclipse para el NIOS II en este caso del proyecto Quadrotor.

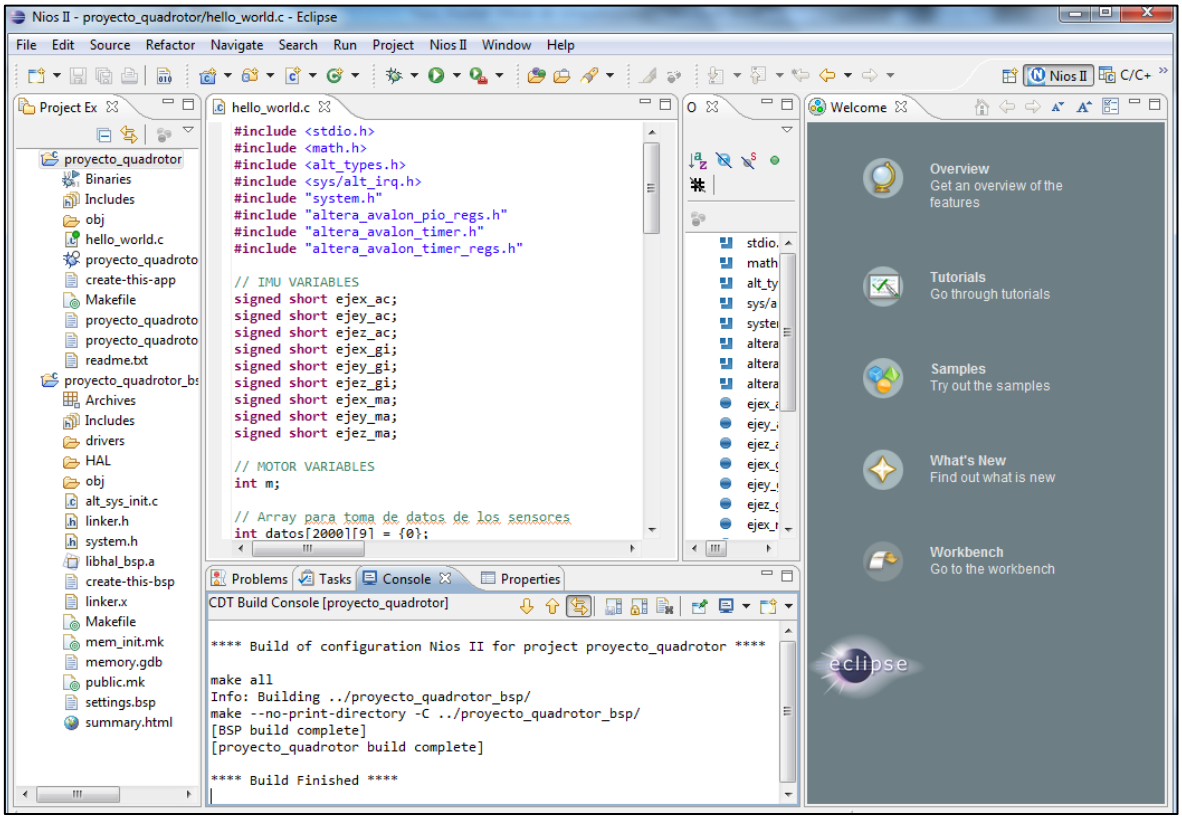

Figura 2.7: Interfaz gráfica en eclipse para el procesador NIOS II

2.1.5 Arquitectura de Bus Simple (SBA)

La Arquitectura de Bus Simple (SBA, Simple Bus Architecture), está compuesta por herramientas de software y núcleos de propiedad intelectual, interconectados de una manera práctica y sencilla; esto permite un diseño rápido, además de lograr la implementación de un sistema embebido en chip (SoC), con una estructura de valor educativo inherente (Risco, 2014). El código utilizado en esta arquitectura es el VHDL por lo que tiene una alta portabilidad. La SBA es una aplicación y derivación simplificada de la especificación Wishbone, que es un estándar de interconexión de núcleos IP (Núcleos de Propiedad Intelectual).

La SBA define 3 tipos de núcleos IP: maestros, esclavos y auxiliares.

a) Núcleo IP Maestros

Los núcleos maestros generaran e inician los ciclos del bus y de controlar el flujo de datos (figura 2.8). Este núcleo IP se denomina SBAController y en todo sistema debe haber al menos uno.

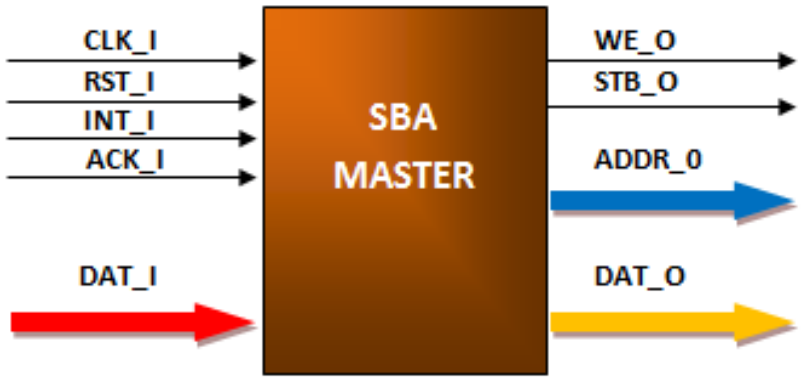

Figura 2.8: Interconexión de un núcleo maestro[12]

b) Núcleos IP Esclavos

Son núcleos con una funcionalidad específica (figura 2.9); tales como, una unidad de procesamientos de datos o como un adaptador entre el SoC y un dispositivo externo al chip. Cada IP esclavo tiene una dirección en el mapa de direcciones del SBA.

[12] Imagen disponible en: http://sba.accesus.com/home

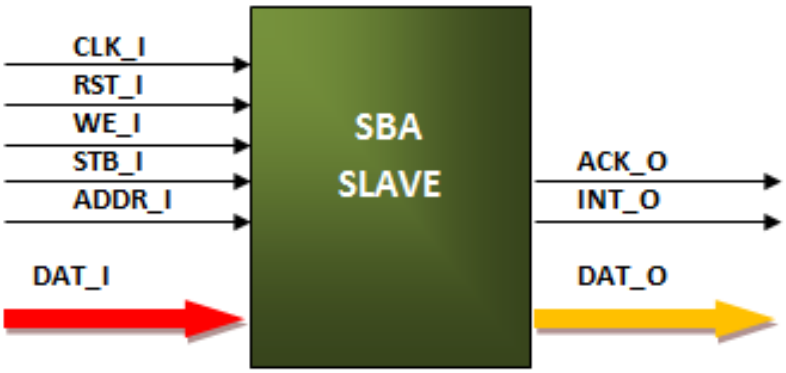

Figura 2.9: Interconexión de un núcleo esclavo[13]

c) Núcleos IP Auxiliares

Los núcleos auxiliares operan junto al núcleo maestro en tareas para la administración del bus y de las señales de control. Entre estos están: El núcleo AddrSpace que determina la posición de cada esclavo en el mapa de direcciones y además genera las señales de habilitación; el Syscon, que controla el reloj del sistema y sincroniza la señal de inicialización; el núcleo DataIntf que hace posible compartir el bus de datos de entrada del maestro con múltiples esclavos; entre otros. En la figura 2.10 se muestra, como ejemplo, el esquema de interconexión del núcleo auxiliar AddrSpace.

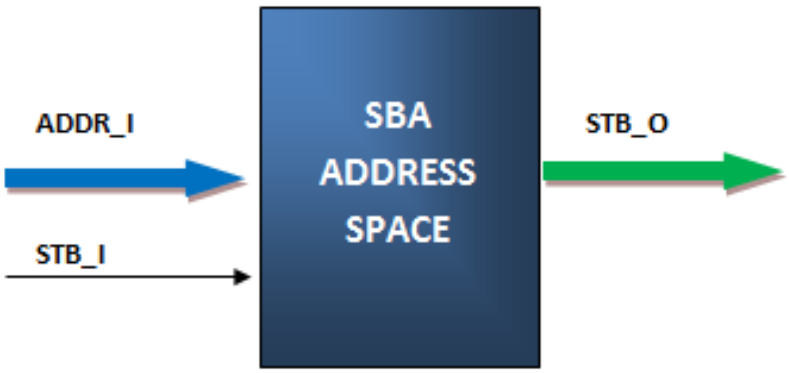

Figura 2.10: Interconexión del AddrSpace[14]

[13] Imagen disponible en: http://sba.accesus.com/home

2.1.6 Tarjeta de desarrollo FPGA DE0-Nano

La tarjeta DE0-Nano es una plataforma de desarrollo, de la empresa Terasic Technology, que contiene un FPGA Cyclone IV de la marca Altera. El tamaño y peso de esta tarjeta son reducidos, pero con una gran capacidad de procesamiento y poco consumo de energía; características requeridas para ser transportada a bordo del Quadrotor (Terasic Technologies Inc., 2011). Las características más importantes de la tarjeta DE0-Nano se muestran en la tabla 2.2. En la figura 2.11 se muestra una imagen del FPGA.

Tabla 2.2: Características del FPGA DE0-nano

Característica	Descripción
FPGA	Cyclone IV EP4CE22 con EPCS16 16-Mbit serial
Interfaces I/O	• Cable USB para configuración del FPGA • Acelerómetro ADXL345 de tres ejes con 13-bit de resolución • Convertidor A/D, 8 canales, 12-bit de resolución
Memoria	• 32 MB SDRAM • 2 Kb I2C EEPROM
Interruptores y LEDs	• 8 LEDs verdes • 4 interruptores

[14] Imagen disponible en: http://sba.accesus.com/home

Figura 2.11: tarjeta DE0-Nano[15]

2.2 Navegación Inercial

En este subcapítulo se presenta una serie de descripciones necesarias sobre navegación inercial para el desarrollo de este proyecto.

Un sistema de navegación inercial tiene como objetivo determinar la posición, velocidad y actitud de una aeronave, mediante una Unidad de Medida Inercial (IMU, Inertial Measurement Unit), compuesto por acelerómetros, giroscopios y magnetómetros. El presente proyecto se enfoca en la determinación de actitud u orientación, en este caso, para un UAV Quadrotor (Jiong, Lei, Jiangping, Rong, & Jianyu, 2010).

En primer lugar, se describen los diferentes sensores a utilizar, luego se revisan los conceptos estudiados en el desarrollo del sistema de medición inercial para la determinación de orientación.

[15] Tarjeta DE0-Nano. Imagen disponible en: http://www.terasic.com.tw/cgi-bin/page/archive.pl?CategoryNo=139&No=593

2.2.1 Sensores Inerciales

La selección de sensores a utilizar es muy importante, ya que tienen que cumplir varias características importantes para ser colocados a bordo del Quadrotor.

En este caso se han elegido sensores inerciales de tecnología MEMS (Sistemas Micro-Electromecánicos), que se caracterizan por su reducido tamaño menor a un centímetro, su bajo costo en comparación con otro tipo de tecnologías, su poco peso, ya que el Quadrotor tiene una capacidad limitada de carga útil, poco consumo de energía y eficiencia; características muy importantes para el UAV.

Acelerómetro ADXL345

Un acelerómetro tiene la capacidad de medir aceleraciones estáticas como es la gravedad de la Tierra, así como también, aceleraciones dinámicas (las producidas por movimientos). El modelo usado es el ADXL345 del fabricante Analog Devices, de tecnología MEMS de tres ejes, que tiene las siguientes características (Analog Devices, 2012):

- Bajo consumo de energía, 23 μA en modo medida y 0.1 μA en modo espera.
- Resolución de 13 bits en un rango de hasta $\pm$ 16g.
- Fuente de alimentación 2.0 V a 3.6V.
- Interfaces de comunicación SPI e I2C.

En la figura 2.12 se aprecia el tamaño reducido del sensor, el cual tiene una dimensión de 3mm x 5mm x 1mm; que está colocado en un tablero de la marca Sparkfun Electronics.

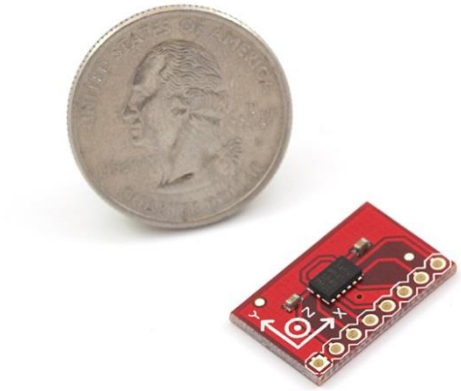

Figura 2.12: Acelerómetro ADXL345[16]

Giroscopio ITG-3200

Un giroscopio tiene la capacidad de medir velocidades angulares, en este caso el modelo usado es el ITG-3200 de tecnología MEMS (figura 2.13) del fabricante InvenSense, de triple eje con salida digital (InvenSense, Inc., 2012).

Para poder determinar el ángulo de giro en cualquiera de sus ejes se realiza una integración, pero con esto se genera un error o deriva que se incrementa con el tiempo, que se tiene que corregir mediante software. Entre las características más importantes de este giroscopio están:

- Salida digital de 3 ejes (X, Y, Z) de velocidades angulares.
- Tres convertidores de analógico - digital (ADC) de 16 bits, para digitalizar las salidas.
- Bajo consumo de energía: En funcionamiento 6.5 mA y 5 μA en modo espera.
- Sensor de temperatura de salida digital.
- Interfaz serial I2C (400kHz).
- Rango de voltaje 2.1 – 3.6V.

[16] Acelerómetro ADXL345. Obtenido de https://www.sparkfun.com/

Figura 2.13: Giroscopio ITG-3200[17]

Magnetómetro HMC5883L

Este magnetómetro mide la intensidad de campo magnético en tres ejes. La dirección del campo magnético de la tierra se define mediante la inclinación y declinación. Para el Perú la inclinación, que es el ángulo entre plano horizontal y el vector de campo magnético, es de aproximadamente 0°5' y la declinación que es el ángulo entre el norte verdadero y el norte magnético, es alrededor de -1°8' (NOAA, 2007).

Las características más importantes del HMC5883L (Honeywell International Inc., 2011) son:

- Incorpora un ADC de 12 bits el cual proporciona una resolución de 2mili-gauss en un rango $\pm$ 8 gauss de campo magnético.
- Voltaje de alimentación de 2.16 – 3.6 V con un bajo consumo de energía de 100 μA.
- Interfaz de comunicación I2C.

En la figura 2.14 se muestra el magnetómetro HMC5883L en el tablero de la marca Sparkfun Electronics.

[17] Giroscopio *ITG-3200*. Obtenido de https://www.sparkfun.com/

Figura 2.14: Magnetómetro HMC5883L[18]

2.2.2 Sistema de coordenadas para el Quadrotor

Para determinar la orientación en el espacio del Quadrotor es necesario establecer dos sistemas de coordenadas ortogonales, lo que permiten mediante los sensores a bordo del vehículo, conocer su orientación respecto a un sistema global que en este caso es el planeta Tierra.

Coordenadas de Navegación

Este sistema se define como “M”; en el cual, su origen es un punto sobre la superficie de la Tierra, con el plano X, Y tangente a este punto; donde, el eje X apunta hacia al norte, el eje Y hacia el este y el eje Z hacia el centro de la Tierra, como se muestra en la figura 2.15.

Coordenadas Body

Este sistema definido como “B” tiene como origen de coordenadas el centro de masa del Quadrotor como se muestra en la figura 2.16, en el cual el eje X apunta hacia la dirección de movimiento principal, el cual puede ser cualquiera de los dos ejes del

[18] Magnetómetro HMC5883L. Obtenido de https://www.sparkfun.com/

Quadrotor, ya que consiste de una estructura simétrica; el eje Y apunta hacia la derecha y el eje Z hacia abajo.

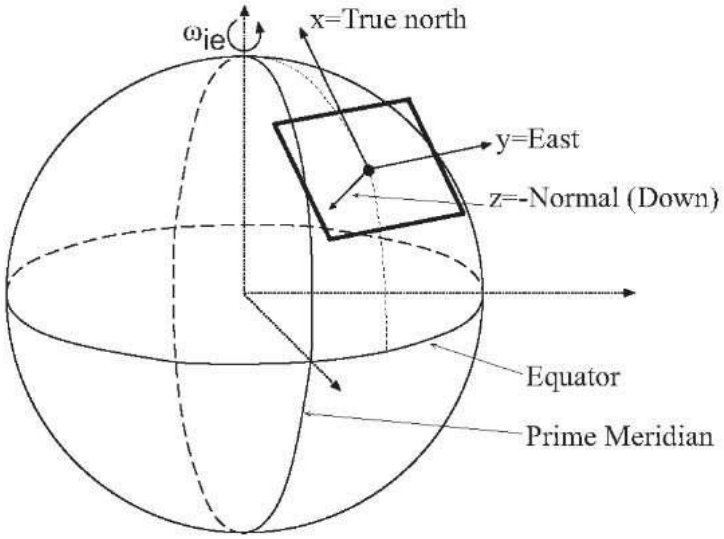

Figura 2.15: Coordenadas de Navegación[19]

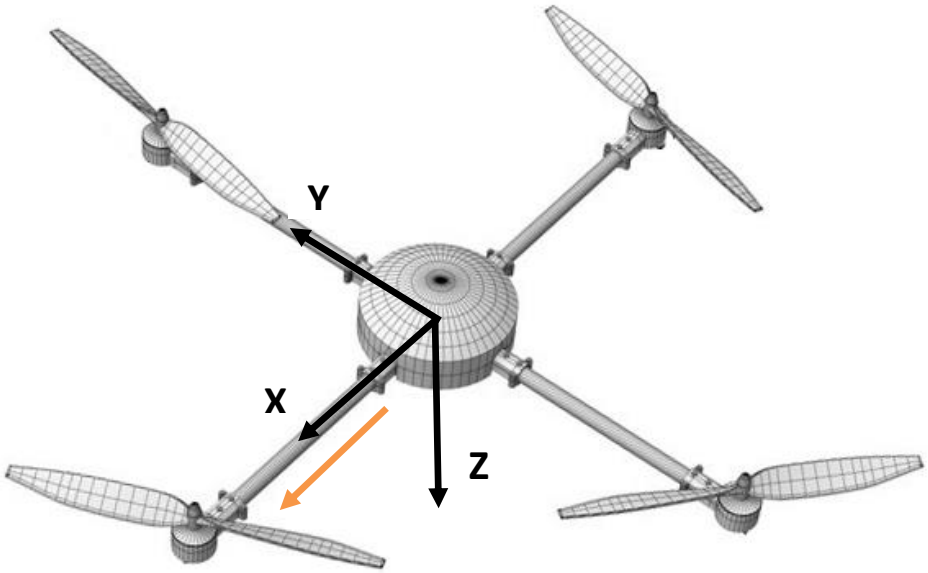

Figura 2.16: Coordenadas Body

[19] Coordenada de navegación, imagen disponible en: http://what-when-how.com/gps-with-high-rate-sensors/reference-frame-definitions-gps/

2.2.3 Métodos de Representación de Orientación

Existen diferentes maneras para conocer la orientación del Quadrotor en cada instante respecto a un sistema de referencia fijo; en este trabajo se abordó dos métodos de parametrización: los Ángulos de Euler y los Cuaterniones (Diebel, 2006).

Ángulos de Euler

Este método sostiene que se puede representar cualquier rotación en el espacio mediante tres rotaciones sucesivas e independientes, en cada eje de coordenadas del sistema (Figura 2.17), donde:

ϕ : representa el ángulo respecto del eje x (ángulo de giro).

θ : representa el ángulo respecto del eje y (ángulo de inclinación).

ψ : representa el ángulo respecto del eje z (ángulo de desviación).

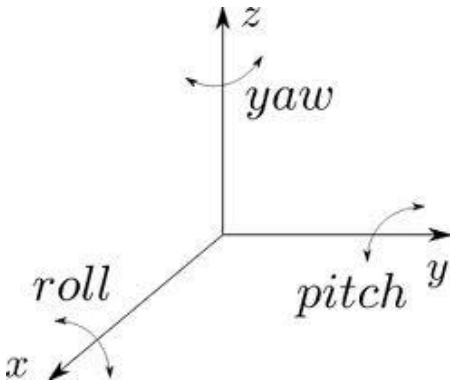

Figura 2.17: Rotaciones de los ángulos de Euler

Para realizar estas rotaciones se utiliza una Matriz de Rotación asociada a cada eje del sistema de coordenadas.

Matriz de Rotación

Es la matriz mediante la cual se pueden realizar rotaciones alrededor de uno o más ejes, transformando vectores de un sistema de coordenadas a otro. En este caso se desarrollan las matrices de rotación de un sistema de Coordenadas Body a Coordenadas de Navegación, como se muestran a continuación:

$$R_b^n(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -sen\phi \\ 0 & sen\phi & cos\phi \end{bmatrix} \quad (2.1)$$

$$R_b^n(\theta) = \begin{bmatrix} cos\theta & 0 & sen\theta \\ 0 & 1 & 0 \\ -sen\theta & 0 & cos\theta \end{bmatrix} \quad (2.2)$$

$$R_b^n(\psi) = \begin{bmatrix} cos\psi & -sen\psi & 0 \\ sen\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2.3)$$

A manera de ejemplo se muestra una rotación alrededor de un solo eje, en este caso del eje z con un ángulo α y con un cambio de sistema de coordenadas (x, y, z) a otro (x', y', z') como se ilustra en la figura 2.18, en el cual se aprecia un mismo punto en dos sistemas de coordenadas diferentes (Diebel, 2006).

Al multiplicar dos matrices de rotación produce una tercera, la cual tiene la misma rotación que la secuencia generada de las dos por separado. Por lo tanto se puede obtener una matriz de Rotación general para los tres ejes como:

$$R_b^n(\varphi,\theta,\psi) = R_b^n(\psi)R_b^n(\theta)R_b^n(\phi) = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

$$= \begin{bmatrix} cos\psi cos\theta & cos\psi sen\theta sen\phi - sen\psi cos\phi & cos\psi sen\theta cos\phi + sen\psi sen\phi \\ sen\psi cos\theta & sen\psi sen\theta sen\phi + cos\psi cos\phi & sen\psi sen\theta cos\phi - cos\psi sen\phi \\ -sen\theta & cos\theta sen\phi & cos\theta cos\phi \end{bmatrix} \quad (2.4)$$

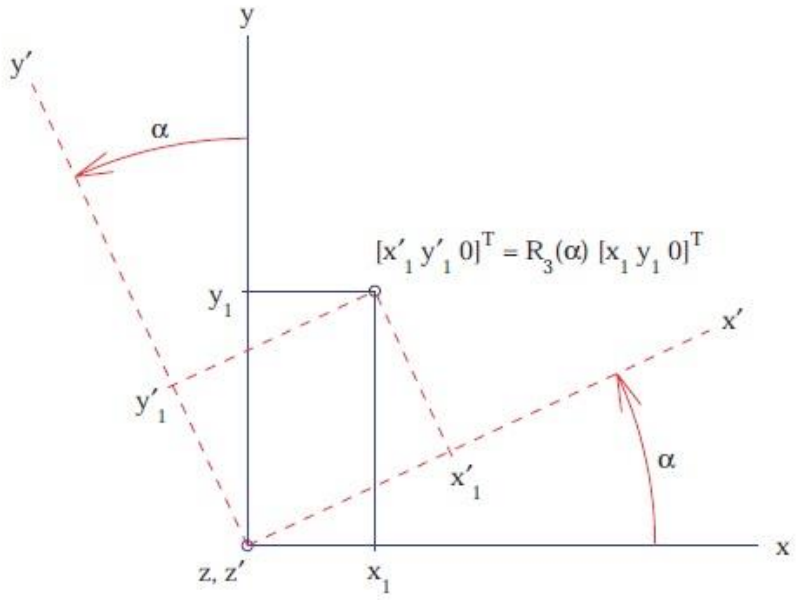

Figura 2.18: Rotación alrededor del eje z con un ángulo ($\boldsymbol{\alpha}$)[20]

Mediante las siguientes fórmulas se pueden obtener los ángulos de Euler a partir de la matriz de rotación, donde:

$$\phi = atan2(r_{32}, r_{33}) \tag{2.5}$$

$$\theta = -asin(r_{31}) \tag{2.6}$$

$$\psi = atan2(r_{21}, r_{11}) \tag{2.7}$$

Los ángulos ϕ, θ y ψ se definen en los cuatro cuadrantes:

[20] Diebel, J. (2006). *Representing Attitude Euler Angles, Unit Quaternions, and Rotation Vectors.* California.

$$atan2(y,x) = \begin{cases} \text{atan}\left(\frac{y}{x}\right) & si\ x > 0 \\ \text{atan}\left(\frac{y}{x}\right) - \pi & si\ x < 0\ \Lambda\ y < 0 \\ \text{atan}\left(\frac{y}{x}\right) + \pi & si\ x < 0\ \Lambda\ y > 0 \end{cases} \quad (2.8)$$

Mediante la siguiente ecuación matricial se obtiene la velocidad angular (ω) del Quadrotor:

$$\begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = R_w(\phi,\theta,\psi) \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \quad (2.9)$$

Con la matriz:

$$R_w(\phi,\theta,\psi) = \begin{bmatrix} 1 & 0 & -sen\theta \\ 0 & cos\phi & sen\phi cos\theta \\ 0 & -sen\phi & cos\phi cos\theta \end{bmatrix} \quad (2.10)$$

Con los sensores a bordo del Quadrotor, se puede obtener la velocidad angular entonces es conveniente obtener la matriz inversa. Para $\theta \neq \pm 90°$ se obtiene:

$$R_w(\phi,\theta,\psi)^{-1} = \begin{bmatrix} 1 & sen\phi tan\theta & cos\phi tan\theta \\ 0 & cos\phi & -sen\phi \\ 0 & sen\phi/cos\theta & cos\phi/cos\theta \end{bmatrix} \quad (2.11)$$

Como el método más fácil y común de representar la orientación en el espacio de un objeto es mediante ángulos de Euler, estos tienen diversas desventajas entre los cuales están:

- Presentan singularidad cuando el ángulo de inclinación $\theta = \pm 90°$, ya que en esta orientación los ejes yaw y roll coinciden perdiendo un grado de libertad, este fenómeno también es conocido como bloqueo de ejes (en inglés, Gimbal lock).

- En comparación con los cuaterniones son menos precisos, cuando se utiliza para determinar la actitud u orientación del Quadrotor, además genera un mayor cálculo computacional.

Cuaterniones

Los cuaterniones fueron descubiertos por William Rowan Hamilton en 1843, éstos son una extensión de los números reales, parecido a los números complejos con la diferencia de que en lugar de añadirse un número imaginario, los cuaterniones constan de tres números imaginarios (Sánchez, 2011). El cuaternión $q \in \mathcal{H}$ se puede definir mediante la siguiente expresión:

$$q = q_0 + q_1 i + q_2 j + q_3 k \tag{2.12}$$

También se puede representar como un vector, mediante la ecuación:

$$q = [q_0\ q_1\ q_2\ q_3]^T \tag{2.13}$$

Donde:

$$i^2 = j^2 = k^2 = -1, \qquad ij = k, \qquad jk = i, ki = j$$

Como el tema de los cuaterniones es extenso, en este trabajo sólo se muestran definiciones específicas para representar la actitud del Quadrotor (Nonami, Kendoul, Suzuki, Wang, & Nakazawua, 2010).

Norma

Para usar los cuaterniones y representar rotaciones de un cuerpo rígido es necesario que su norma sea igual a uno. Se define la norma como:

$$\|q\| = \sqrt{{q_0}^2 + {q_1}^2 + {q_2}^2 + {q_3}^2} \tag{2.14}$$

Si el cuaternión no tiene como norma uno, se puede normalizar definiendo un nuevo cuaternión mediante la siguiente expresión:

$$q = \frac{q}{\|q\|} \tag{2.15}$$

Conjugada

La conjugada de un cuaternión se obtiene al cambiar los signos de las componentes imaginarias.

$$\bar{q} = [q_0 \ -q_1 - q_2 - q_3]^T \tag{2.16}$$

Inversa

Se define el cuaternión inverso como sigue:

$$q^{-1} = \frac{\bar{q}}{\|q\|} \tag{2.17}$$

Multiplicación de cuaterniones

La multiplicación de cuaterniones es no conmutativa y para multiplicarlos se pueden usar matrices, en el cual, el segundo cuaternión se multiplica por una matriz que está en función del primer cuaternión; como se muestra en la siguiente expresión:

$$q \otimes p = Q(q)p = \bar{Q}(p)q \tag{2.18}$$

Donde:

$$Q(q) = \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & q_3 & -q_2 \\ q_2 & -q_3 & q_0 & q_1 \\ q_3 & q_2 & -q_1 & q_0 \end{bmatrix} \tag{2.19}$$

Y la conjugada del matriz cuaternión es:

$$\bar{Q}(q) = \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_0 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{bmatrix} \tag{2.20}$$

Rotación

Como se mencionó anteriormente, para usar los cuaterniones representando rotaciones en el espacio es necesario que la norma sea igual a uno; en estas condiciones se considera un vector $\boldsymbol{t} \in \mathbb{R}^3$ en sistema de coordenadas de Navegación y $\boldsymbol{t'} \in \mathbb{R}^3$ que es el mismo vector en el sistema de coordenadas Body; y se cumple que:

$$\begin{bmatrix} 0 \\ t' \end{bmatrix} = q \otimes \begin{bmatrix} 0 \\ t \end{bmatrix} \otimes \bar{q} = \bar{Q}(q)^T Q(q) \begin{bmatrix} 0 \\ t \end{bmatrix} \tag{2.21}$$

La expresión (2.21) se puede reducir a una multiplicación de una sola matriz por el vector t como sigue:

$$t' = R(q)t \tag{2.22}$$

Donde;

$$R(q) = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1 q_2 + q_0 q_3) & 2(q_1 q_3 - q_0 q_2) \\ 2(q_1 q_2 - q_0 q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2 q_3 + q_0 q_1) \\ 2(q_1 q_3 + q_0 q_2) & 2(q_2 q_3 - q_0 q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \tag{2.23}$$

Pero si se requiere rotar de un sistema Body al sistema de Navegación se formula:

$$t = R(q)^T t' \tag{2.24}$$

Con;

$$R(q)^T = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \quad (2.25)$$

Derivada

Existe una función que relaciona la derivada en el tiempo de una unidad de cuaternión con la velocidad angular (ω), que se podría definir en el sistema de coordenadas de navegación o en el Body, en este caso se toma el último sistema, ya que el giroscopio está montado sobre el Quadrotor. Entonces se tiene:

$$\dot{q_{\omega'}}(q, \omega') = \frac{1}{2}\begin{bmatrix} 0 \\ \omega' \end{bmatrix} \otimes q = \frac{1}{2}\bar{Q}(q)\begin{bmatrix} 0 \\ \omega' \end{bmatrix} = \frac{1}{2}\begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_0 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{bmatrix}\begin{bmatrix} 0 \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}$$

$$= \frac{1}{2}\begin{bmatrix} -q_1 & -q_2 & -q_3 \\ q_0 & -q_3 & q_2 \\ q_3 & q_0 & -q_1 \\ -q_2 & q_1 & q_0 \end{bmatrix}\begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \quad (2.26)$$

Conversión de Cuaterniones en Ángulos de Euler

Mediante la siguiente expresión se puede realizar la conversión de cuaterniones en ángulos de Euler:

$$\phi = atan2(2(q_0q_1 + q_2q_3), 1 - 2({q_1}^2 + {q_2}^2)) \quad (2.27)$$

$$\theta = -asin(2q_1q_3 - 2q_0q_2) \quad (2.28)$$

$$\psi = atan2(2(q_1q_2 + q_0q_3), 1 - 2({q_2}^2 + {q_3}^2)) \quad (2.29)$$

El uso de cuaterniones tiene como desventaja cierta complejidad matemática, entre sus ventajas está que no tiene el problema de bloqueo de ejes ni singularidades, de esta manera, se puede realizar cualquier tipo de rotación; además, su implementación no requiere demasiada capacidad de procesamiento, debido que las operaciones son más simples que los ángulos de Euler o rotación de matrices. Además, representar la actitud mediante cuaterniones proporciona un algoritmo más robusto.

2.2.4 Modelo dinámico del Quadrotor

El modelo dinámico del Quadrotor (Das, Subbarao, & Lewis, 2009) en términos de posición (x, y, z) y rotación (ϕ, θ, ψ) utilizando la formulación Euler-Lagrange se define mediante las ecuaciones siguientes (Bouabdallah & Siegwart, 2007):

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + \frac{1}{m} \begin{bmatrix} -\sin\theta \\ \cos\theta \sin\phi \\ \cos\theta \cos\phi \end{bmatrix} u \tag{2.30}$$

$$\begin{bmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} = f(\phi, \theta, \psi) + g(\phi, \theta, \psi)\tau \tag{2.31}$$

Donde:

$$f(\phi, \theta, \psi) = \begin{bmatrix} \dot{\theta}\dot{\psi}\left(\frac{I_y - I_z}{I_x}\right) - \frac{J_p}{I_x}\dot{\theta}\Omega \\ \dot{\phi}\dot{\psi}\left(\frac{I_z - I_x}{I_y}\right) - \frac{J_p}{I_y}\dot{\phi}\Omega \\ \dot{\phi}\dot{\theta}\left(\frac{I_x - I_y}{I_z}\right) \end{bmatrix} \tag{2.32}$$

$$g(\phi, \theta, \psi) = \begin{bmatrix} \frac{l}{I_x} & 0 & 0 \\ 0 & \frac{l}{I_y} & 0 \\ 0 & 0 & \frac{l}{I_z} \end{bmatrix} \tag{2.33}$$

$$\tau = \begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} \in R^3 \tag{2.34}$$

La variable $\tau_{(\phi,\theta,\psi)}$ representa los torques en los ángulos de giro, inclinación y desviación respectivamente, $I_{(x,y,z)}$ representa las inercias del Quadrotor, J_p es la inercia del rotor, $\Omega = \omega_2 + \omega_4 - \omega_1 - \omega_3$, u es el empuje total y ω la velocidad angular del motor. Este sistema está conformado por seis salidas y cuatro entradas (Raffo, 2007) como se muestra en la figura 2.19.

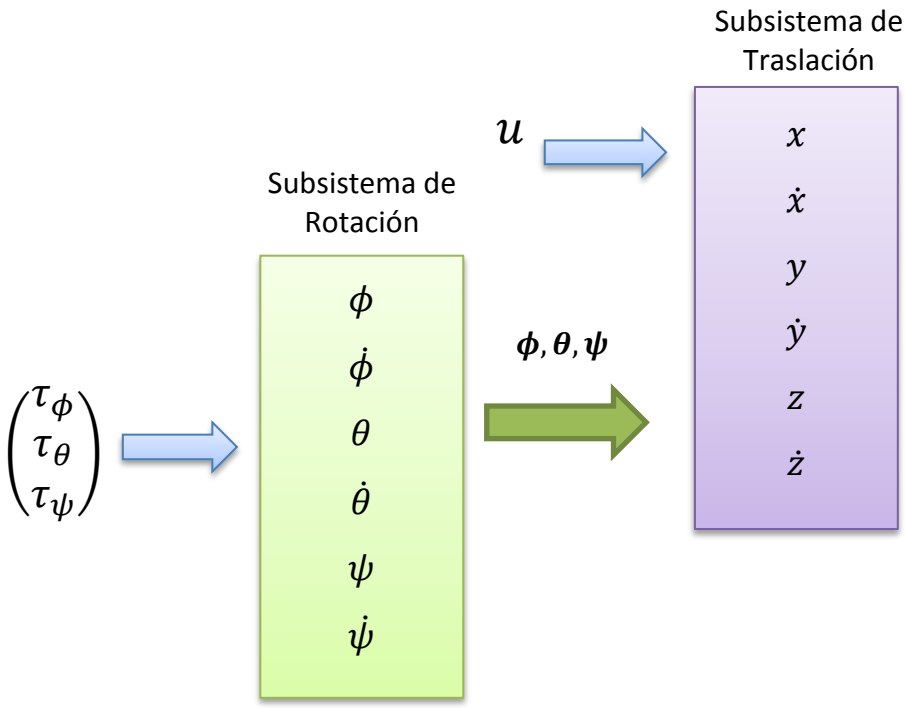

Figura 2.19: Sistema dinámico del Quadrotor

En la tabla 2.3 se muestran los valores de los parámetros determinados para el Quadrotor.

Tabla 2.3: Valores de los parámetros del Quadrotor

Parámetro	Abreviación	Valor	Unidades
Masa	m	0.9	Kg
Longitud de brazo	l	0.025	m
Momento de inercia en el eje X	I_x	$9.657x10^{-3}$	$Kg.m^2$
Momento de inercia en el eje Y	I_y	$10.161x10^{-3}$	$Kg.m^2$
Momento de inercia en el eje Z	I_z	$18.863x10^{-3}$	$Kg.m^2$
Constante de arrastre de torque	b	$1.14x10^{-7}$	$\frac{kg.m^2}{rev^2}$
Momento de inercia del rotor	J_p	$6.5x10^{-5}$	$kg.m^2$

2.2.5 Filtro de Kalman

Para describir completamente el comportamiento de un sistema, se determina primero un modelo matemático que permite representar un sistema físico, basado en leyes fundamentales y pruebas empíricas; sin embargo, es imposible tomar en cuenta las innumerables variables para representar a dicho sistema, por lo cual sólo se toman aquellas de mayor interés, introduciendo de esta manera, una incertidumbre en el modelado del sistema. Incluso las variables de control tienen cierta incertidumbre debido a perturbaciones externas, así como también, los sensores utilizados para un control de lazo cerrado presentan ruido e imprecisiones. Por estas razones, un modelo solamente

determinístico no permitirá describir, ni tener un conocimiento completo del sistema, como tampoco un control óptimo de éste (Maybeck, 1979).

Por esta razón, se introduce un modelo estocástico que toma en cuenta las incertidumbres introducidos al sistema y mejora las deficiencias de un modelo solamente determinístico; un ejemplo de este modelo óptimo es el Filtro de Kalman.

El Filtro de Kalman es un algoritmo óptimo y recursivo de procesamiento de datos. Es óptimo porque proporciona la mejor estimación de una variable deseada, minimizando el error estadísticamente, para ello tiene en cuenta la dinámica del sistema y del dispositivo de medida, la descripción estadística de los ruidos del sistema, errores de medida e incertidumbre del sistema dinámico, además de información sobre las condiciones iniciales de las variables. Es así que, teniendo las mediciones de diferentes dispositivos de medida, cada uno con sus propias dinámicas y errores característicos, se pueda utilizar cada una de estas salidas y combinarlas de una manera óptima. El Filtro de Kalman es recursivo ya que no es necesario almacenar todos los datos anteriores, a diferencia de otros algoritmos, ni tener que procesarlos cada vez que se toma una nueva medición; debido a esta característica hace que sea un algoritmo adecuado y práctico para su implementación en un procesador.

Filtro de Kalman Discreto

En esta sección se describe el Filtro de Kalman mediante puntos discretos del tiempo para la estimación de estado y mediciones.

El modelo de Proceso y la Medida

El objetivo de aplicar el Filtro de Kalman es estimar el estado $x \in \mathbb{R}^n$ de un proceso discreto con una ecuación diferencial lineal estocástica descrito en el espacio de estados (Welch & Gary, 2006). Entonces se expresa:

$$x_k = Ax_{k-1} + Bu_k + w_{k-1} \tag{2.35}$$

Con una ecuación de medida expresada por:

$$z_k = Hx_k + v_k \tag{2.36}$$

Donde:

x_k: Variable de estado del proceso a estimar

u_k: Vector de control

A: Matriz de estado

B: Matriz de entrada

H: Matriz de medida

La variable aleatoria w_k representa el ruido del proceso y v_k representan el ruido de la medida. Para obtener la mejor estimación del estado existen ciertas restricciones, el primero es, que el modelo del sistema sea lineal y que el ruido del sistema y de la medición sean del tipo blancos y gaussianos, en otras palabras que el ruido no este correlacionado en el tiempo y que tengan una distribución de probabilidad normal o forma de campana:

$$p(w) \backsim N(0, Q) \tag{2.37}$$

$$p(v) \backsim N(0, R) \tag{2.38}$$

Donde:

Q_k: Matriz de covarianza de ruido del proceso

R_k: Matriz de covarianza de ruido de la medición

Algoritmo del filtro de Kalman Discreto

El proceso de estimación es un proceso en el cual, dos grupos de ecuaciones se repiten en forma de bucle; estos grupos son: las ecuaciones de predicción y las ecuaciones de corrección. En la etapa de Predicción se toman en cuenta la dinámica e incertidumbre del proceso y se proyecta el estado actual junto con el error de covarianza, para así obtener la estimación "a priori" o anterior de la siguiente medición. En la etapa de Corrección una vez tomado el siguiente dato, éste se fusiona de la manera más óptima, reduciendo la probabilidad de error al mínimo con la estimación a priori, para finalmente obtener la estimación deseada o "a posteriori".

En la Figura 2.20, se muestra el diagrama del proceso de operación del Filtro de Kalman.

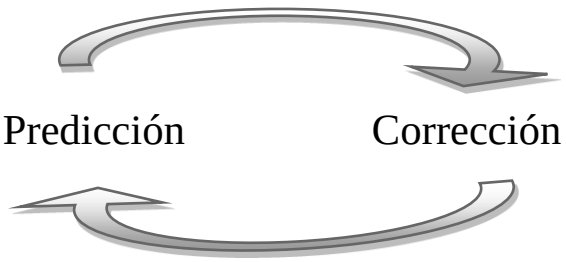

Figura 2.20: Ciclo de operación del Filtro de Kalman

Las ecuaciones en tiempo discreto de las etapas de predicción y corrección son:

Ecuaciones de Predicción:

$$\hat{x}_k^- = A\hat{x}_{k-1}^- + Bu_k \tag{2.39}$$

$$P_k^- = AP_{k-1}A^T + Q \tag{2.40}$$

Mediante la ecuación (2.39) se determina el estado estimado a priori ($\hat{x}_k^-$) en el tiempo *k*, a partir de la estimación anterior en el tiempo *k-1* ($\hat{x}_{k-1}^-$), el cual tiene toda la información de los estados anteriores. Luego, en la ecuación (2.40) se obtiene el error de

covarianza estimado a priori (P_k^-) a partir del error de covarianza anterior del tiempo *k-1* y de la covarianza de ruido del proceso.

Ecuaciones de Corrección

$$K_k = P_k^- H^T (H P_k^- H^T + R)^{-1} \tag{2.41}$$

$$\hat{x}_k = \hat{x}_k^- + K_k (z_k - H\hat{x}_k^-) \tag{2.42}$$

$$P_k = (I - K_k H) P_k^- \tag{2.43}$$

En esta etapa el primer paso es obtener la Ganancia de Kalman (K_k), para esto se toma en cuenta la covarianza del ruido de la medición (R); una vez hallado la ganancia éste se multiplica por el resultado de la resta $(z_k - H\hat{x}_k^-)$ o también llamado residuo de la medida, en el cual, se calcula la diferencia entre la predicción de la medida $(H\hat{x}_k^-)$ y la toma de la medida actual (z_k); para después, obtener mediante la ecuación (2.42) el estado estimado a posteriori ($\hat{x}_k$). Por último se actualiza el error de covarianza a posteriori (P_k) con la ecuación (2.43).

En la figura 2.21, se muestra el diagrama completo del proceso de estimación de estados del Filtro de Kalman. En este diagrama se muestra el bucle entre las dos etapas de predicción y corrección con sus respectivas ecuaciones; básicamente, primero se realiza una predicción del estado estimado, a través del estado a posteriori anterior; para luego, una vez tomado el nuevo dato de medida corregir la predicción, obteniendo una nueva estimación a posteriori óptima.

2.2.6 Filtro de Kalman Extendido

Como se mostró en la sección anterior, se asumió para el Filtro de Kalman una estimación del proceso con una ecuación diferencial estocástica lineal, pero que sucede si el modelo del proceso o de la medida es *no lineal*. Estas situaciones se presentan

frecuentemente, como en este caso, por lo que es necesario aplicar una linealización previa mediante derivadas parciales que se van a detallar en este capítulo. A este nuevo enfoque del filtro se le denomina “Filtro de Kalman Extendido” o EKF (por sus iniciales en inglés de, Extended Kalman Filter).

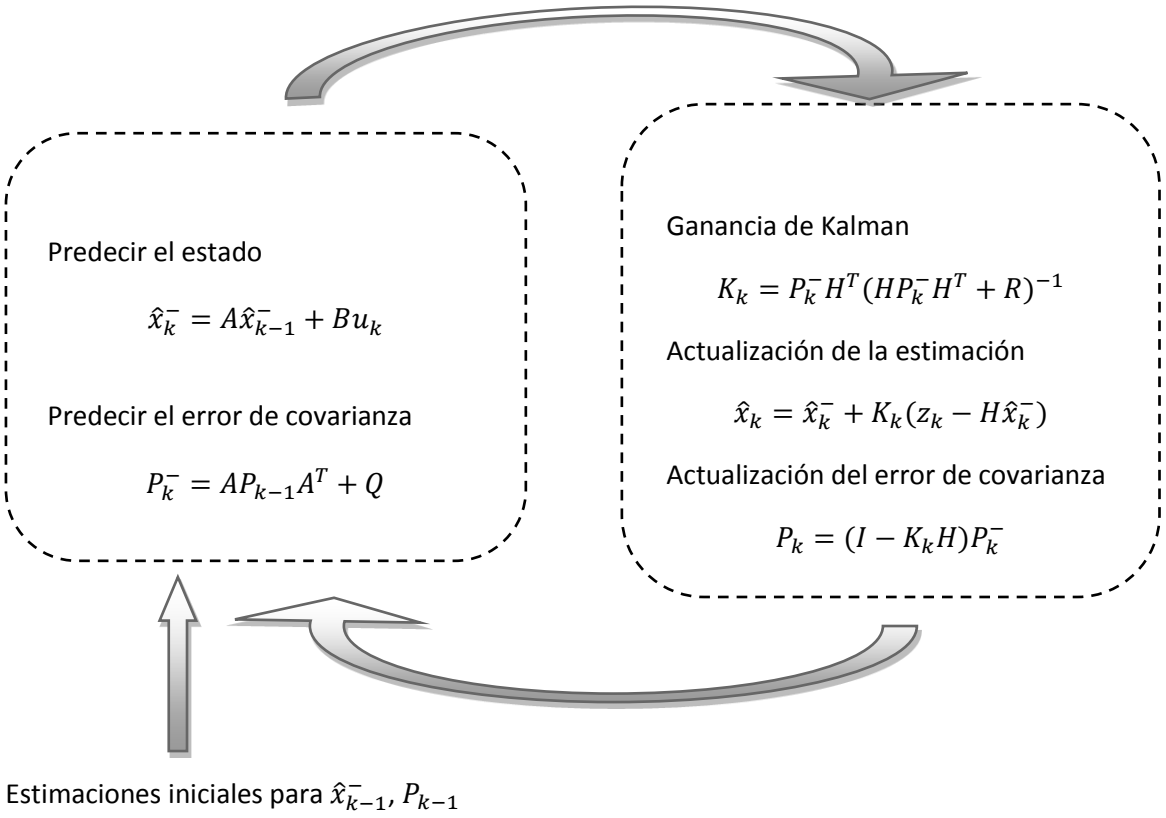

Figura 2.21: Diagrama completo de la operación del Filtro de Kalman

Modelo del Proceso

Se asume un proceso con una ecuación diferencial estocástica *no lineal*, cuyo vector de estado $x \in \mathbb{R}^n$ se define como:

$$x_k = f(x_{k-1}, u_k, w_{k-1}) \tag{2.44}$$

Y con una medición $z \in \mathbb{R}^m$ igual a:

$$z_k = h(x_k, v_k) \tag{2.45}$$

En este caso se representa una función no lineal denotada por f, el cual relaciona el estado anterior *k-1* con el estado actual *k*. Así también se establece h como una función no lineal el cual relaciona el estado x_k con la medida z_k.

Para simplificar se puede realizar una aproximación del proceso ($\tilde{x}_k$) y de la medida ($\tilde{z}_k$), igualando a cero el ruido del proceso w_k y el de la medición v_k.

$$\tilde{x}_k = f(\hat{x}_{k-1}, u_k, 0) \tag{2.46}$$

$$\tilde{z}_k = h(\tilde{x}_k, 0) \tag{2.47}$$

Luego finalmente las ecuaciones que linealizan y estiman el proceso son:

$$x_k \approx \tilde{x}_k + F(x_{k-1} - \hat{x}_{k-1}) + W w_{k-1} \tag{2.48}$$

$$z_k \approx \tilde{z}_k + H(x_k - \tilde{x}_k) + V v_k \tag{2.49}$$

En donde x_k y z_k son los vectores actuales de estado y de medida respectivamente; $\tilde{x}_k$ y $\tilde{z}_k$ son los valores aproximados, $\hat{x}_k$ es el estado estimado a posteriori. Para poder aproximar linealmente a una función alrededor de la actual estimación, se utiliza la matriz Jacobiana formado por las derivadas parciales de la función.

Entonces, se define F como la matriz Jacobiana de las derivadas parciales de f respecto de x.

$$F_{[m][n]} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}_{(\hat{x}_{k-1}, u_k)} \tag{2.50}$$

A W como la matriz Jacobiana de las derivadas parciales de f respecto de w.

$$W_{[m][n]} = \frac{\partial f_{[m]}}{\partial w_{[n]}}(\hat{x}_{k-1}, u_k, 0) \tag{2.51}$$

La matriz Jacobiana H con las derivadas parciales de h respecto de x.

$$H_{[m][n]} = \begin{bmatrix} \frac{\partial h_1}{\partial x_1} & \cdots & \frac{\partial h_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_m}{\partial x_1} & \cdots & \frac{\partial h_m}{\partial x_n} \end{bmatrix}_{(\tilde{x}_k, 0)} \tag{2.52}$$

Y la matriz Jacobiana V con las derivadas parciales de h respecto de v.

$$V_{[m][n]} = \frac{\partial h_{[m]}}{\partial v_{[n]}}(\tilde{x}_k, 0) \tag{2.53}$$

Redefiniendo Ww_k , Vv_k como ε_k y η_k respectivamente, como las nuevas variables aleatorias, se tiene la siguiente distribución de probabilidad:

$$p(\varepsilon_k) \backsim N(0, WQ_kW^T) \tag{2.54}$$

$$p(\eta_k) \backsim N(0, VR_kV^T) \tag{2.55}$$

Al linealizarse el sistema se aplica el algoritmo Filtro de Kalman lineal descrito en la sección anterior, obteniendo las siguientes ecuaciones para la etapa de predicción y corrección.

Ecuaciones de Predicción del EKF

Haciendo el cambio de $\tilde{x}_k$ por $\hat{x}_k^-$ para una mejor apreciación de la notación *a priori*, se obtiene la expresión:

$$\hat{x}_k^- = f(\hat{x}_{k-1}, u_k, 0) \tag{2.56}$$

$$P_k^- = F_k P_{k-1} F_k^T + W_k Q_{k-1} W_k^T \tag{2.57}$$

Los Jacobianos F, W al igual que H y R que se muestran en la etapa de corrección, tienen subíndice k lo que indica que se recalculan en cada paso de tiempo. Con la ecuación (2.56) se predice el estado a estimar y con la ecuación (2.57) el error de covarianza, tomando en cuenta el ruido aleatorio del proceso para este último.

Ecuaciones de Corrección del EKF

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + V_k R V_k^T)^{-1} \tag{2.58}$$

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - h(\hat{x}_k^-, 0)) \tag{2.59}$$

$$P_k = (I - K_k H_k) P_k^- \tag{2.60}$$

Luego de la etapa de predicción, como primer paso, se determina la Ganancia de Kalman mediante la ecuación (2.58); posteriormente, se corrigen tanto el estado con la medición z_k de los sensores, así como también, el error de covarianza; mediante las ecuaciones (2.59) y (2.60) respectivamente. En la figura 2.22 se muestra el diagrama completo del bucle de operación entre las etapa de predicción y corrección junto con las

variables iniciales. Este diagrama ilustra la secuencia en la que se implementa en el NIOS II de la DE0-Nano.

Sintonización de parámetros

Hallar el valor de la covarianza de ruido de la medida (R) es relativamente sencillo, ya que se realiza calculando la varianza del ruido de las mediciones de los sensores. Pero, obtener el valor de la covarianza del ruido del proceso (Q) no es directamente medible, por el hecho, de que no se puede observar directamente el proceso a estimar. Para esto se insertar cierta incertidumbre en el proceso, para luego una vez obtenido valores para las covarianzas, a éstos aplicarles un ajuste de valores que se realiza fuera de línea o con otro Filtro de Kalman distinto.

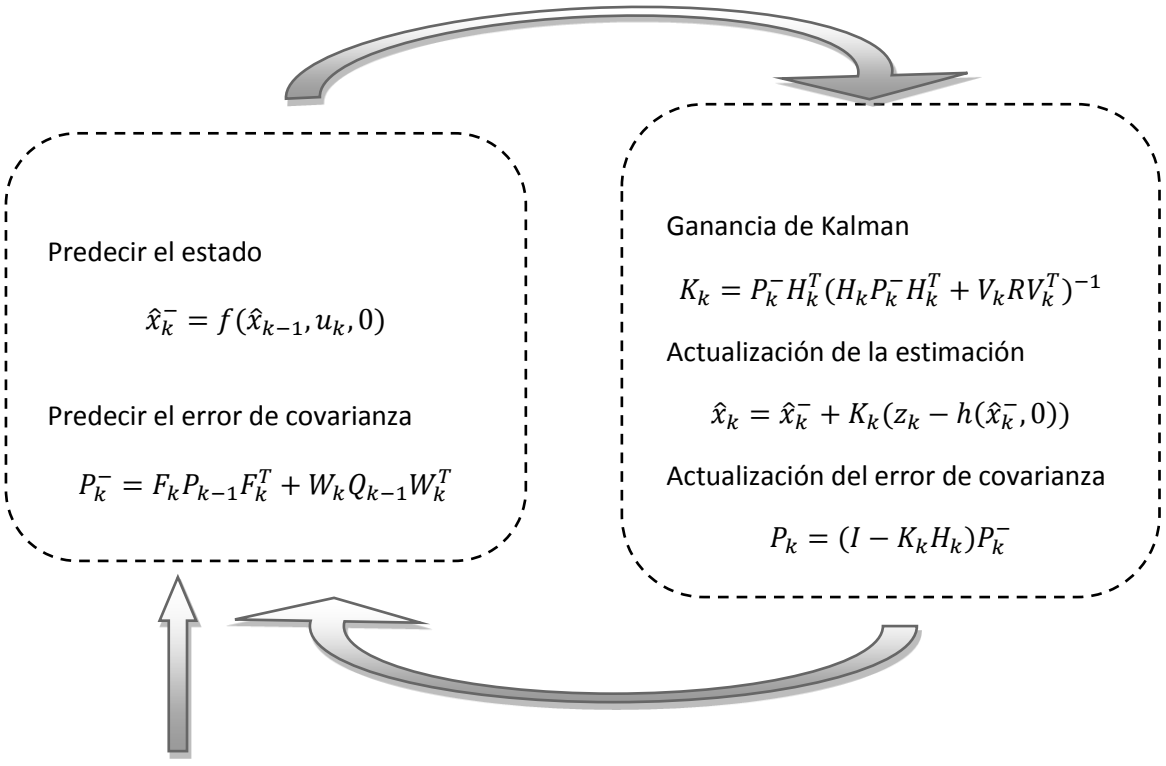

Figura 2.22: Diagrama completo de operación del Filtro de Kalman Extendido

2.2.7 Índices de rendimiento

Los índices de rendimiento son datos mediante los cuales se pueden juzgar la eficacia y eficiencia de un sistema para alcanzar su objetivo.

En este caso se utiliza: la media, varianza y error RMS de los datos obtenidos del Filtro de Kalman, para analizar su rendimiento (Pierre & Denis, 2004).

a) Error Medio

Es el error promedio de un conjunto de valores estimados. Mediante el error medio se analiza la exactitud de las mediciones.

Entonces dado un conjunto de muestras $\{x_i\}$ se tiene la siguiente expresión matemática:

$$\bar{e} = 1/N \sum_{i=1}^{N} \hat{e}_i$$

Donde N es el número de muestras, $\hat{e}_i = \widehat{x_i} - y_i$ y:

$\widehat{x_i}$: es el valor estimado.

y_i : es el valor verdadero o referencial.

b) Varianza del error

Mediante la varianza se obtiene información de la precisión de las medidas o del nivel de dispersión de la variable a analizar. Este índice tiene la siguiente expresión matemática:

$$VarError(x) = 1/N \sum_{i=1}^{N} (e_i - \bar{e})^2$$

c) Error RMS

El error RMS (Root Mean Square) se define como el error cuadrático medio de una variable. Mediante este índice se obtiene información de la magnitud del error existente entre el valor estimado con el valor verdadero o referencial. Se define como:

$$errorRMS = \sqrt{\frac{\sum_{i=1}^{N}(\widehat{x_i} - y_i)^2}{N}}$$

Donde:

$\widehat{x_i}$: es el valor estimado.

y_i : es el valor verdadero o referencial.

CAPÍTULO 3

DISEÑO, SIMULACIÓN E IMPLEMENTACIÓN DEL SISTEMA DE DETERMINACIÓN DE ORIENTACIÓN

En este capítulo se explica el desarrollo de un sistema de determinación de orientación o actitud para un UAV del tipo Quadrotor, el cual es implementado en un dispositivo de lógica reconfigurable (FPGA), utilizando un lenguaje de descripción de hardware (VHDL) y un diseño SOPC (System On a Programmable Chip) en el FPGA.

3.1 Diagrama de bloques del sistema

En la figura 3.1, se muestra el diagrama de bloques general del sistema. Este diseño se divide principalmente en dos partes, en los cuales se centra el desarrollo de este proyecto:

a) Diseño del sistema SBA.
b) Diseño SOPC NIOS II.

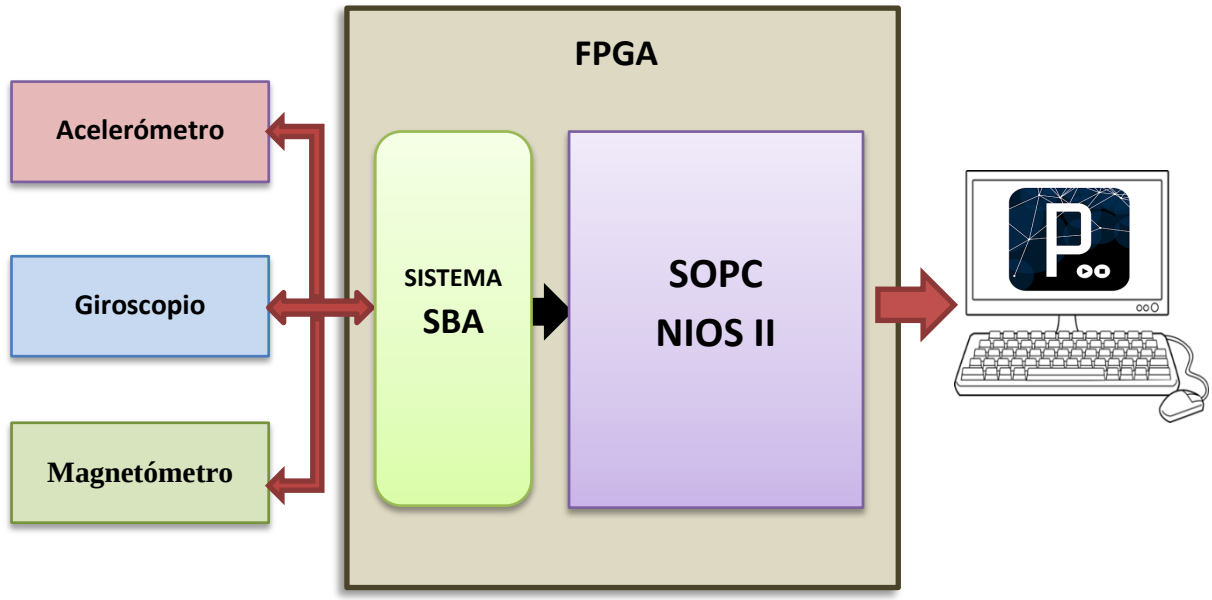

Figura 3.1: Diagrama de bloques general del sistema

En el diagrama de bloques de la figura 3.1, el bloque del sistema SBA tiene como finalidad, realizar la adquisición de datos de los sensores inerciales (acelerómetro, giroscopio), además del magnetómetro, utilizando el protocolo I^2C. El bloque SOPC NIOS II, tiene como función realizar los cálculos matemáticos de los algoritmos para la calibración de los sensores y filtro de Kalman Extendido, a partir de los datos entregados por el sistema SBA, obteniendo de esta manera la orientación en el espacio del Quadrotor. Los datos se envían a un computador mediante el protocolo serial (RS232), en el cual se muestran de manera gráfica los cambios de movimiento en el entorno de Processing[21].

[21] Processing es un lenguaje de programación y entorno de desarrollo integrado basado en Java, diseñado para la producción de proyectos multimedia y aplicaciones gráficas.

3.2 Etapa de diseño

En esta sección se describe principalmente el diseño del bloque sistema SBA y SOPC NIOS II.

3.2.1 Diseño del sistema SBA

Este bloque implementa un sistema SBA (Risco, 2014), en el cual se utiliza el IP Core I^2C para configurar y leer información de aceleración, velocidad angular y campo magnético de los sensores inerciales (acelerómetro, giroscopio) y del magnetómetro.

En la figura 3.2 se visualiza el bloque sistema SBA de la figura 3.1. Se observa que este bloque tiene como señal de entrada, los datos de los sensores, utilizando el protocolo de comunicaciones I^2C configurado a una velocidad de 400 KHz. El sub-bloque que recibe estos datos es el I^2C ADAPTER, cuya salida está conectada al controlador SBA, mediante el cual se pueden realizar las configuraciones de los registros internos de los sensores. Como señales de salida se tienen: los datos de los tres sensores en sus tres ejes (X, Y, Z) cada uno de 16 bits. Estos datos se envían al procesador SOPC NIOS II.

En la tabla 3.1, se muestran las configuraciones que necesitan tener los sensores digitales para que tengan un funcionamiento óptimo, utilizando para ello el SBA Controller. Los sensores son: el acelerómetro ADXL345, giroscopio ITG-3200 y el magnetómetro HMC5883L. Para un mayor detalle, en el Anexo A se muestra un diagrama de flujo del proceso de configuración, mediante el proceso de escritura I^2C.

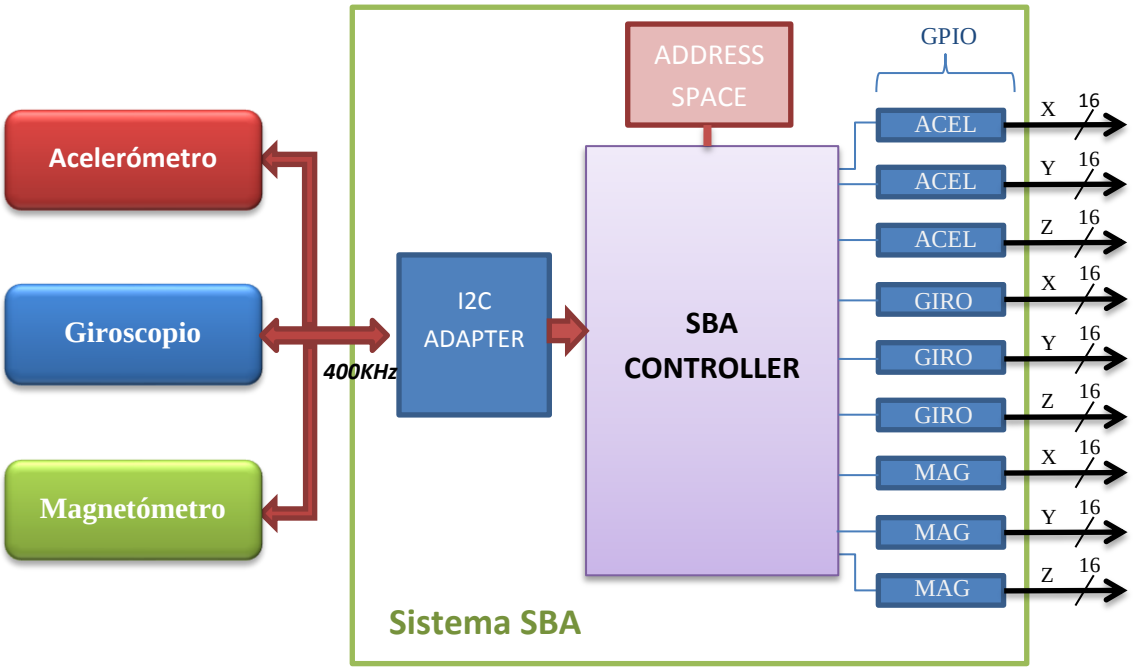

Figura 3.2: Comunicación con los sensores mediante el IP Core SBA I2C

Tabla 3.1: Configuraciones de los sensores mediante el SBAController

SENSOR	Rango de medición	Resolución	Velocidad de salida de datos
Acelerómetro ADXL345	$+/- \ 16g$	$3.9 \ mg/LSB$	$100 \ Hz$
Giroscopio ITG3200	$+/- \ 2000°/s$	$14.375 \ LSB/(°/s)$	$62.5 \ Hz$
Magnetómetro HMC5883L	$+/- \ 1.3Gauss$	$0.92 \ mGauss/LSB$	$15 \ Hz$

3.2.2 Diseño SOPC NIOSII

El diseño del SOPC NIOSII (ver figura 3.1) involucra un desarrollo en hardware y software. La parte Hardware está compuesto por bloques desarrollados en un lenguaje de descripción de hardware (VHDL), y la parte software está conformado por el desarrollo

de los cálculos necesarios en la calibración de sensores y Filtro de Kalman Extendido, implementados en el núcleo del procesador SOPC NIOS II.

Diagrama de bloques

Como se observa en el diagrama de bloques de la figura 3.3, el sistema SOPC NIOS II consta de dos partes: diseño de la arquitectura hardware y diseño software.

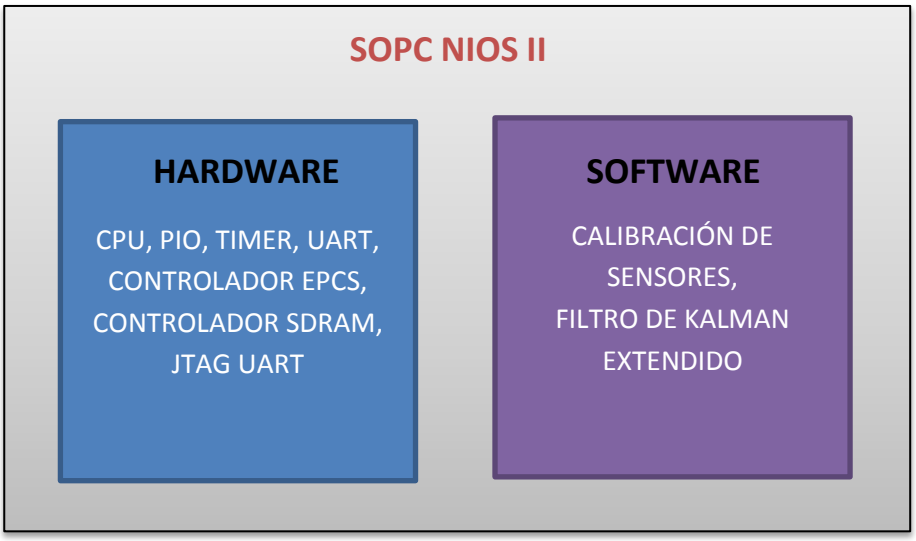

Figura 3.3: Hardware y software del SOPC NIOS II

a) Arquitectura Hardware

La arquitectura hardware se detalla en el diagrama de bloques de la figura 3.4, cada parte se implementa en un bloque de silicio dentro del FPGA y fue desarrollado mediante el constructor SOPC Builder del software Quartus II. Detallando el diagrama de bloques de la figura 3.4 se tiene:

- CPU: Es el núcleo del procesador, de una arquitectura RISC de 32 bits, personalizado a NIOSII/f. Se encarga de realizar los cálculos para los algoritmos de calibración y Filtro de Kalman Extendido. Contiene además una unidad de punto flotante.

- JTAG UART: Interfaz de comunicaciones entre el ordenador y el procesador SOPC NIOSII.
- Controlador SDRAM: Controlador que hace posible la comunicación del procesador con la memoria externa SDRAM de 32Mbytes.
- PIO: son entradas y salidas digitales, en este caso se requieren 9 entradas para los datos de los 3 sensores en sus 3 ejes, cada uno de 16 bits. Como salidas se tienen 4 señales PWM de 10 bits cada uno.
- TIMER: Timer interno para la generación de interrupción en cada periodo de muestreo.
- Controlador EPCS: Interfaz de comunicación entre el NIOS II con el chip EPCS16. Mediante este chip se logra la grabación permanente en el FPGA de los archivos que contienen el diseño desarrollado. Estos son: el archivo **.sopcinfo* que contiene toda la información de la arquitectura Hardware; y el archivo **.elf* que contiene la información del software.

b) Arquitectura software

El desarrollo del software se realiza en el CPU del procesador SOPC NIOS II de la figura 3.4. Los algoritmos matemáticos están desarrollados en el lenguaje de programación C/C++, utilizando el IDE (entorno de desarrollo integrado) de Eclipse.

El software realiza las siguientes tareas:

- Desarrollo del algoritmo de calibración para el acelerómetro, giroscopio y magnetómetro.
- Desarrollo del algoritmo de determinación de orientación (Filtro de Kalman Extendido).

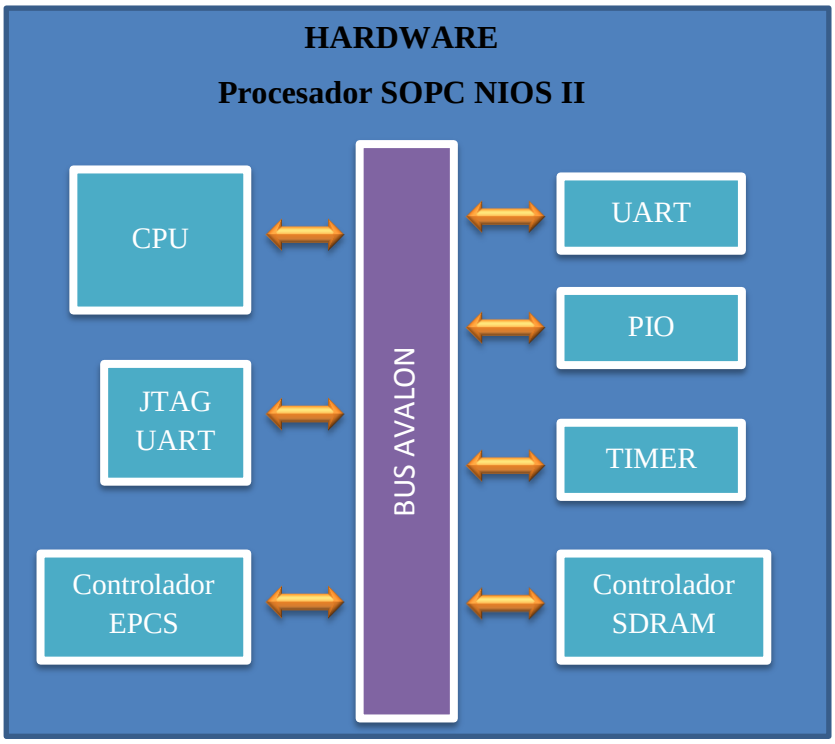

Figura 3.4: Arquitectura hardware del SOPC NIOS II

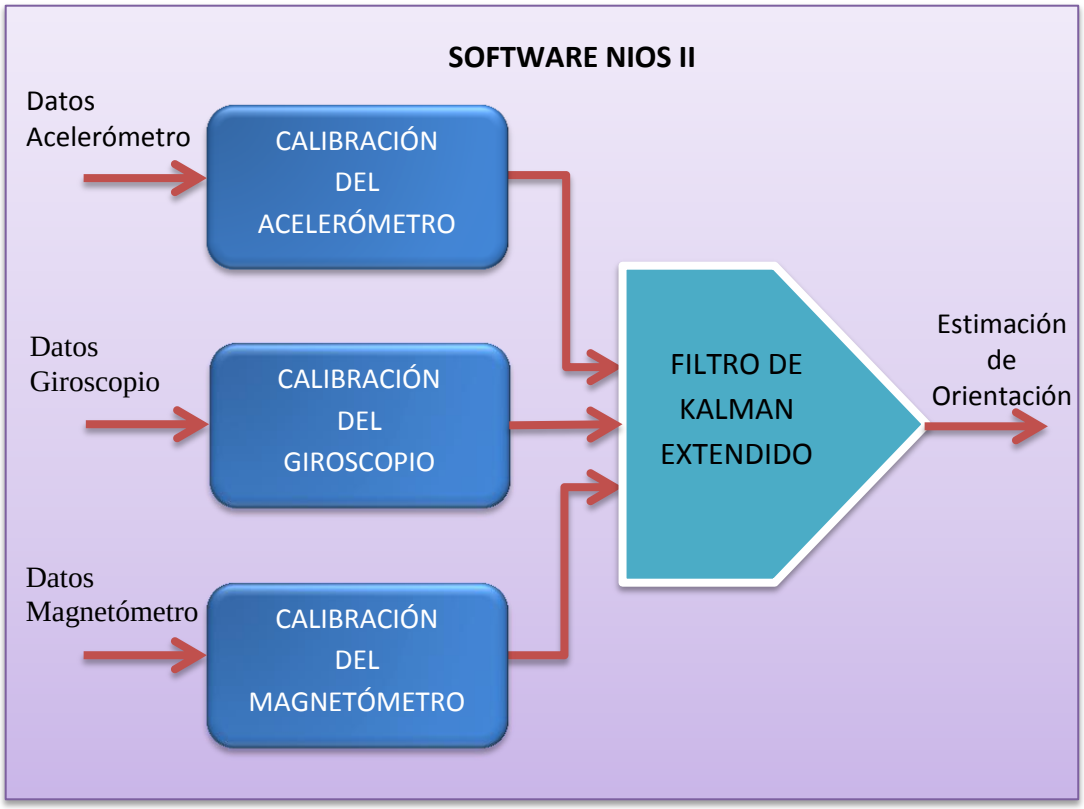

Figura 3.5: Diagrama del software en el procesador NIOS II

En la figura 3.5, se muestra el diagrama de bloques del software a ejecutarse en el procesador NIOS II. En primer lugar, se realiza la calibración de los sensores con los datos obtenidos del sistema SBA (figura 3.2). Los datos ya calibrados son utilizados en el en el desarrollo del algoritmo de fusión de sensores (Filtro de Kalman Extendido), obteniendo la estimación de orientación del Quadrotor expresado en ángulos de Euler (roll, pitch, yaw).

A continuación se detalla el proceso de calibración y fusión de sensores.

Calibración de sensores

La salida de datos del IMU (Unidad de Medida Inercial) que está conformado por el acelerómetro, giroscopio además de un magnetómetro; contienen errores del tipo determinístico, que son propios y constantes de cada sensor o en el caso del magnetómetro errores causados por campos magnéticos externos; estos errores, que son mayores en sensores de bajo costo, como son los del tipo MEMS, pueden ser determinados y compensados mediante un proceso llamado calibración. Este proceso puede mejorar significativamente la precisión de las mediciones.

Los errores que se generan son a menudo producidos por los diversos esfuerzos mecánicos a los que han sido sometidos dichos sensores, como pueden ser: el proceso de fabricación, soldadura en el PCB o montaje de agujeros; otros factores que también influyen son la temperatura y el envejecimiento.

Los errores que se consideran en este modelo de calibración son los siguientes: el desalineamiento, offset, factor de escala y la no ortogonalidad. Existen diferentes métodos de calibración, pero para este proyecto se eligió, aquel en el cual se comparan las salidas de datos de instrumentos de precisión con la de los sensores, que se ponen en diferentes orientaciones o actitudes, para realizar finalmente un ajuste de curvas mediante el algoritmo de mínimos cuadrados y hallar los parámetros de calibración. Para

realizar pruebas, fue necesario diseñar y construir una plataforma mecánica que permitiese obtener dichos parámetros de calibración.

A. Modelo de error de los sensores y algoritmo de calibración

En este modelo de error se toman en cuenta cuatro parámetros: el desalineamiento, offset, factor de escala y la no ortogonalidad (Skog & Handel, 2006). Estos parámetros son expresados mediante la siguiente ecuación de matrices para los tres sensores:

$$\begin{bmatrix} A_x \\ A_y \\ A_z \end{bmatrix} = [T]_{3x3}[A\_D]_{3x3} \begin{bmatrix} 1/FS_x & 0 & 0 \\ 0 & 1/FS_y & 0 \\ 0 & 0 & 1/FS_z \end{bmatrix} \begin{bmatrix} a_x - OS_x \\ a_y - OS_y \\ a_z - OS_z \end{bmatrix} \quad (3.1)$$

Donde:

$[T]_{3x3}$: Alinea la ortogonalidad del sensor con el sistema de coordenadas Body

$[A\_D]_{3x3}$: Vector de desalineamiento entre los ejes del sensor y los ejes del sistema

$FS_i(i = x, y, z)$: Factor de Escala

$OS_i(i = x, y, z)$: Offset

$A_i(i = x, y, z)$: Salida con los errores compensados en cada eje

$a_i(i = x, y, z)$: Datos de los sensores en cada eje

La ecuación (3.1) puede ser reescrita, obteniendo:

$$[A_x \quad A_y \quad A_z] = [a_x \quad a_y \quad a_z \quad 1] \begin{bmatrix} A_{11} & A_{21} & A_{31} \\ A_{12} & A_{22} & A_{32} \\ A_{13} & A_{23} & A_{33} \\ A_{10} & A_{20} & A_{30} \end{bmatrix} \quad (3.2)$$

El proceso de calibración se puede reducir a encontrar doce parámetros desde A_{10} hasta A_{33} de la ecuación (3.2). Que también puede ser expresado como:

$$Y = W.X \tag{3.3}$$

Se aplica entonces el algoritmo de mínimos cuadrados donde se determina los doce parámetros de calibración representada por X:

$$X = [W^T * W]^{-1} * W^T * Y \tag{3.4}$$

B. Plataforma de calibración

Para calibrar los sensores (acelerómetro, giroscopio, magnetómetro) se introducen los doce parámetros de calibración en la ecuación 3.2 y se implementa dicha ecuación en el software del procesador NIOS II.

Para calcular los parámetros de calibración es necesario construir un sistema mecánico que ponga los sensores en diferentes posiciones, inclinaciones y a diferentes velocidades conocidas o consideradas referenciales. Estos datos referenciales luego son comparados con los datos de los sensores para poder hallar su error. Si bien existen sistemas de calibración comerciales, éstos son muy costosos y de gran tamaño, por lo que no son adecuados para este proyecto, es por ello que se realizó un diseño propio.

En la Figura 3.6 se muestra un esquema general del proceso de captura los datos tanto de los sensores, como los obtenidos mediante la plataforma mecánica.

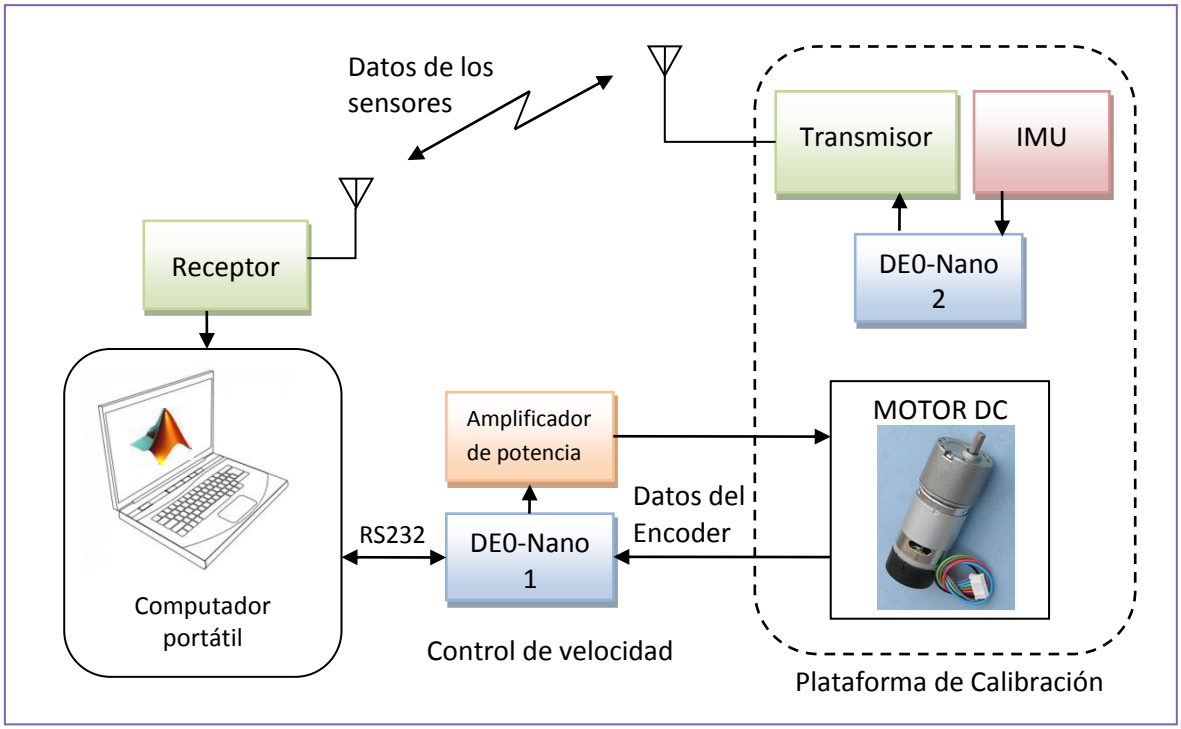

Figura 3.6: Esquema general del proceso de captura de datos de los sensores y plataforma de calibración

En el diagrama de la figura 3.6, se observa una primera tarjeta DE0-Nano 1, que permite el control de velocidad del motor DC que hace girar la plataforma de calibración. Esta DE0-Nano 1 recibe los diferentes puntos de consigna enviados desde el computador, para ejecutar el algoritmo del control PID y generar la señal PWM hacia el amplificador de potencia compuesto por el puente H, para controlar la velocidad y determinar el sentido de giro del motor DC. La tarjeta DE0-nano 1 también transmite al computador, vía comunicación RS232, la información de velocidad del motor DC obtenida mediante el codificador. Con la tarjeta DE0-nano 2, se obtienen los datos de los sensores (IMU), para luego transmitir de manera inalámbrica los datos al computador utilizando un transmisor Xbee.

Un computador recibe tanto los datos de los sensores como los obtenidos de la plataforma, para luego procesar el algoritmo de mínimos cuadrados y determinar los doce parámetros de calibración, utilizando el software MATLAB[22].

En la figura 3.7 se muestra el diseño de la plataforma mecánica utilizando el software Solidworks[23]. La estructura debe presentar ciertas características como: una velocidad de giro controlada y no generar distorsiones magnéticas. Para un mayor detalle de la estructura mecánica de la plataforma de calibración ver Anexo B.

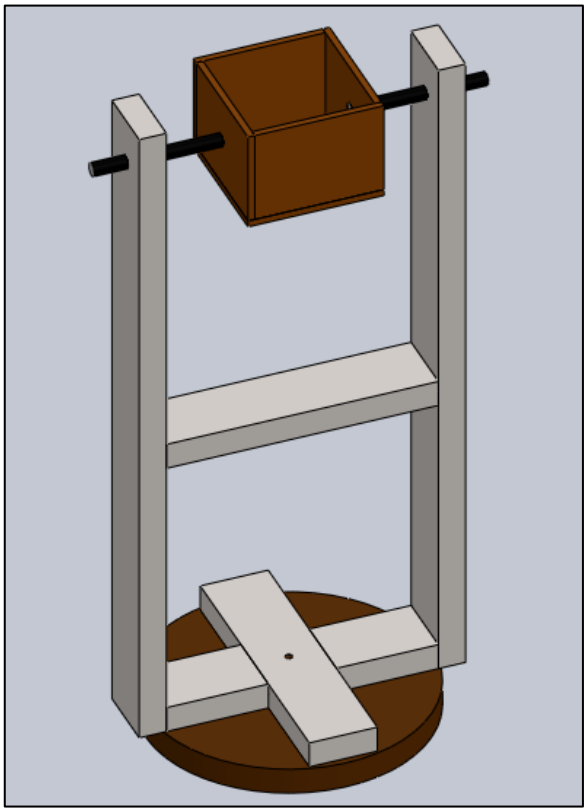

Figura 3.7: Diseño de la plataforma en Solidworks para determinar el valor de los parámetros de calibración

[22] MATLAB, abreviatura de MATrix LABoratory o laboratorio de matrices, es un entorno de desarrollo (IDE) que permite la manipulación de matrices, implementación de algoritmos, creación de interfaces de usuario, entre otros.

[23] Solidworks es un software CAD (Diseño asistido por computadora), que permite modelar piezas, extraer planos técnicos, etc. De una manera automatizada y sencilla.

C. Calibración del giroscopio

La magnitud de los parámetros de calibración del giroscopio se determinan, girando a una velocidad constante, y se realiza una comparación entre los datos obtenidos del sensor con las velocidades obtenidas del motor DC de la plataforma de calibración.

La rotación del sensor se realiza alrededor de cada uno de sus ejes (x, y, z) y se obtienen los datos durante un tiempo determinado. Con el fin de obtener una velocidad constante se ha implementado un control PID de velocidad para el motor DC.

(1) Control de velocidad de un motor DC mediante el FPGA DE0-Nano

Para el control de velocidad PID se utiliza el FPGA DE0-Nano, donde se han logrado resultados de gran precisión. La tarjeta DE0-Nano además de tener una alta capacidad de procesamiento, también permite generar hardware dedicado a tareas específicas, los cuales son: un contador de pulsos para el codificador del motor DC, y un generador de señal PWM, como se muestra en el diagrama de bloques de la figura 3.8.

El procedimiento realizado para el control de velocidad del motor DC, es el siguiente:

(a) Descripción del modelo matemático.

(b) Identificación de parámetros.

(c) control PID.

(d) Diseño del SOPC NIOS II para el control de velocidad.

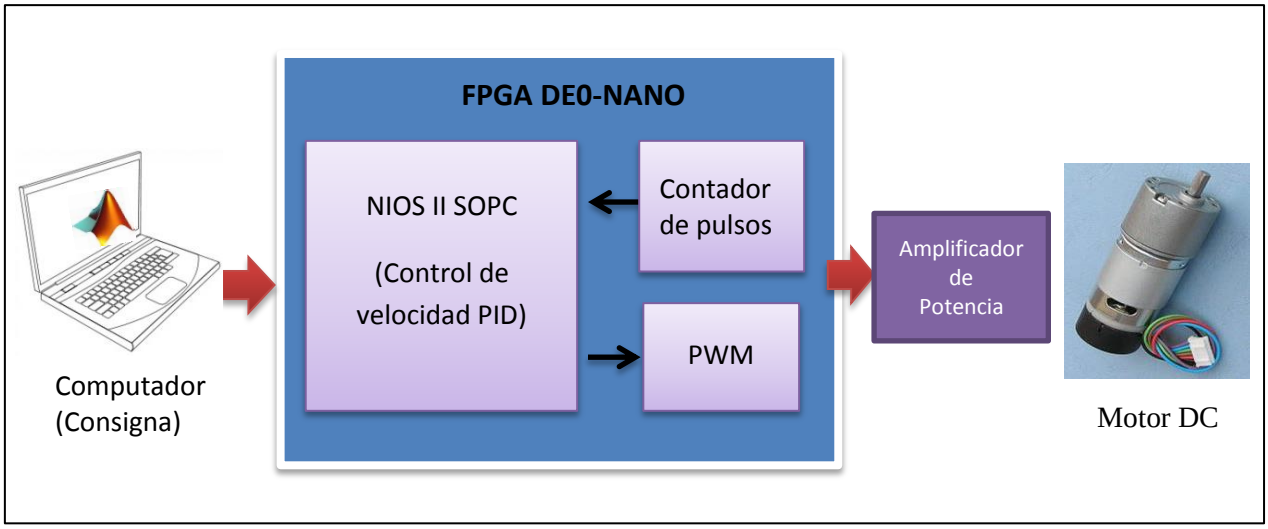

Figura 3.8: Diagrama de bloques del diseño para el control PID del motor DC

(a) Modelo matemático del motor DC

Para representar el comportamiento del motor DC se formulan las ecuaciones 3.5 - 3.9, que son mecánicas y eléctricas; donde la función de transferencia (ecuación 3.9) tiene como entrada la tensión aplicada y como variable de salida la velocidad de giro del eje del motor. En la figura 3.9 se muestra una gráfica con los diferentes parámetros del motor DC.

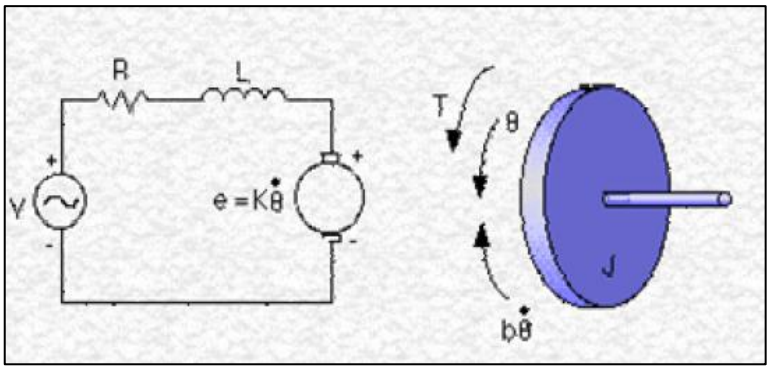

Figura 3.9: Esquema de modelamiento del motor DC

Donde:

V: Fuente de tensión

R: Resistencia eléctrica del motor

L: Inductancia eléctrica

J: Inercia del motor

K: Constante de fuerza electromotriz

b: Coeficiente de amortiguamiento

w: Velocidad angular del eje

Ecuación eléctrica del motor:

$$-V + Ri + L\frac{di}{dt} + e = 0 \tag{3.5}$$

Ecuación mecánica del motor:

$$J\frac{dw}{dt} + bw = Ki \tag{3.6}$$

Aplicando transformada de Laplace, y como $e = Kw$ se tiene:

$$-V + RI(s) + SLI(s) + KW(s) = 0 \tag{3.7}$$

$$SJW(s) + bW(s) = KI(s) \tag{3.8}$$

Relacionando las ecuaciones (3.7) y (3.8) se obtiene la función de transferencia de la velocidad angular respecto del voltaje aplicado:

$$\frac{W}{V} = \frac{K}{(JS + b)(R + SL) + K^2} \tag{3.9}$$

(b) Identificación del sistema

Para poder determinar la función de transferencia, que relaciona la variable de salida del sistema (velocidad angular) con respecto a la variable de entrada (voltaje aplicado), se analiza su comportamiento dinámico. Para ello, se introduce al sistema una entrada tipo escalón de voltaje controlado por PWM, para analizar la respuesta transitoria. El resultado de la entrada tipo escalón y la respuesta transitoria en la salida se muestra en la figura 3.10.

Al obtener los datos, voltaje y velocidad, se introducen en el software ToolBoxIDENT que es una interfaz gráfica de usuario (GUI) del software Matlab (MathWorks, Inc.), con el que se obtiene un modelo del sistema real con precisión, en este caso se utilizó para determinar la función de transferencia del motor DC.

Analizando los resultados mediante el identificador de parámetros del sistema, se concluye que aproximando el sistema a uno de primer orden, se obtienen casi los mismos resultados que aproximándolo a uno de segundo orden; por lo que se eligió la primera alternativa para reducir el tiempo computacional. En la figura 3.11 se muestra el resultado de la aproximación del modelo a una función de primer orden mediante el ToolBoxIDENT.

Según el resultado mostrado en la figura 3.11, se obtiene un modelo con un índice de adecuación o ajuste (Best Fits), de 98.52 %, lo que indica que la función de transferencia obtenida representa de manera casi precisa el sistema real.

La función de transferencia del motor DC a utilizar queda entonces como:

$$G(s) = \frac{3.8913}{1 + 0.067113s} \tag{3.10}$$

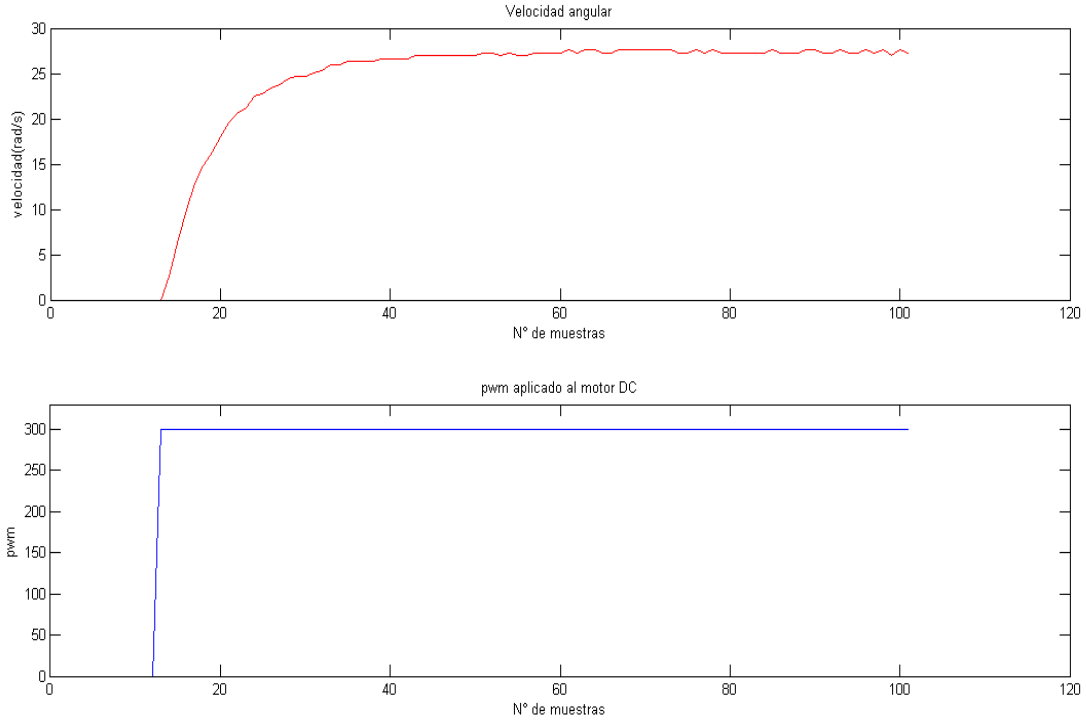

Figura 3.10: Respuesta transitoria del sistema

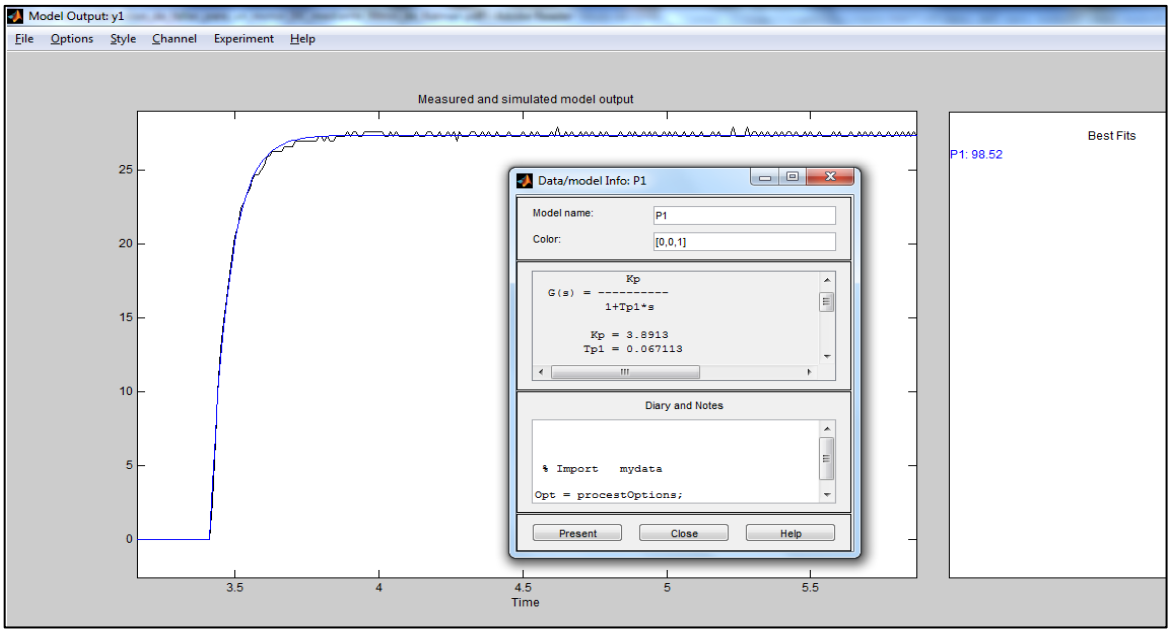

Figura 3.11: Modelo del proceso aproximado a una función de primer orden

(c) Control PID

Con la función de transferencia del motor obtenido, entonces se puede diseñar un controlador con parámetros que cumplan las especificaciones del estado estacionario y transitorio del sistema en lazo cerrado. Un controlador en general compara el valor real de la respuesta con el valor deseado, determina la desviación y produce una señal de control que tienda a cero o a un valor pequeño esta desviación (Ogata, 2002). La acción de control a utilizar es un control PID (Proporcional-Integral-Derivativo) que presenta la siguiente ecuación diferencial:

$$u(t) = K_p e(t) + K_i \int_0^t e(t) + K_d \frac{de(t)}{dt} \tag{3.11}$$

Donde, $e(t)$ es la variable de error, $u(t)$ es la variable de control y K_p, K_i, K_d son las ganancias proporcional, integral y derivativa, respectivamente.

(d) Diseño de un SOPC NIOS II para el control de velocidad

Con este diseño (ver figura 3.8) se ha generado hardware dedicado a tareas específicas mediante un lenguaje de descripción de hardware (VHDL), que consiste en:

- Contador de pulsos: Este bloque implementa en VHDL un contador que cambia su estado interno con los pulsos del codificador del motor DC, cuyos datos son almacenados y transmitidos cuando el algoritmo de control lo requiera. De esta manera el procesador NIOS II se dedica únicamente a procesar el control PID, sin generarse interrupciones.
- Generador PWM: Este bloque permite generar la señal PWM con un ciclo útil calculado a partir del controlador PID.

D. Calibración del acelerómetro

Para obtener los parámetros de calibración del acelerómetro, se inclina este sensor obteniendo los datos de manera simultánea, los cuales se comparan con los datos obtenidos de un instrumento referencial. En este caso se utilizó un transportador con un sistema sexagesimal y amplitud de 360°, como instrumento para medir los ángulos de las inclinaciones.

E. Calibración del magnetómetro

En este sistema de determinación de orientación, el magnetómetro se utiliza únicamente para el movimiento en el eje yaw, ya que en este eje los datos del acelerómetro no son suficientes. Este sensor presenta dos tipos de distorsiones, llamados hierro duro y hierro suave (Hard iron y Soft iron). Las distorsiones por Hard iron producen un aumento constante del campo magnético en la salida de datos del magnetómetro, éstos son causados por materiales magnetizados abordo o cercanos al Quadrotor. Las distorsiones Soft iron no son aditivas ya que producen una distorsión en el campo magnético por metales como el hierro o el níquel.

Por esta razón, en el diseño de la maqueta de calibración (ver figura 3.7), se consideró que el material utilizado no fuese de metal, y que además el motor DC esté lo suficientemente alejado de sensor.

Para determinar los doce parámetros de calibración del magnetómetro, se hace girar e inclinar simultáneamente parar obtener datos en la mayor cantidad de orientaciones posible. Para esto se utilizó también la maqueta de calibración.

Algoritmo de determinación de orientación

En el diseño de control automático del Quadrotor, se requiere un control eficiente de actitud con la capacidad de mantener al vehículo en un vuelo seguro a pesar de

perturbaciones como el viento. Para esto es fundamental primero una estimación de actitud u orientación eficaz.

En este capítulo se muestra el desarrollo de tres algoritmos de estimación de orientación, teniendo en cuenta el uso de sensores de bajo costo y ligeros. El algoritmo utilizado en este trabajo es el llamado MARG (Magnetic force, Angular Rate and Gravity) el cual requiere de un sensor giroscópico, un sensor magnético y un acelerómetro, cada uno de ellos de triple eje. Mediante este método se obtiene una medición completa de la actitud y rumbo del Quadrotor usando la velocidad angular, fuerza magnética y la gravedad de la Tierra (Nonami, Kendoul, Suzuki, Wang, & Nakazawua, 2010).

Los tres algoritmos de estimación de orientación desarrollados son:

- Filtro de Kalman Lineal
- Filtro de Kalman Extendido
- Filtro de Kalman Extendido y Matriz de Rotación

El propósito es comparar estos algoritmos e identificar el de mayor rendimiento en un Quadrotor.

Los dos Filtros de Kalman extendidos tienen el mismo modelo del proceso pero difieren en el modelo de medición. También, los EKF utilizan una representación de la orientación mediante cuaterniones, lo cual disminuye la carga computacional, y evita los problemas de singularidad asociados a los ángulos de Euler.

Esta sección se divide en cuatro partes, que son:

A. Modelado de sensores.
B. Filtro de Kalman Lineal.
C. Filtros de Kalman Extendido.
D. Matrices de error de covarianza.

A. Modelado de sensores

Primero se realizó un modelado matemático del giroscopio, acelerómetro y magnetómetro, como se detalla a continuación:

a) Modelado del Giroscopio

El giroscopio proporciona datos de las velocidades angulares en tres ejes en el sistema de coordenadas Body con respecto al sistema de coordenadas de navegación. Debido al sesgo (bias) o error en la medición una simple integración no es suficiente para predecir la actitud del vehículo. A causa de estas integraciones en lazo abierto este sesgo incrementa el error de estimación al pasar el tiempo. Por lo tanto, el modelo de medición del giroscopio se expresa mediante:

$$\omega_m = \omega_T + b(t) + \eta \tag{3.12}$$

Donde; ω_m es la velocidad angular medida, ω_T es la verdadera velocidad angular, $b(t)$ es el sesgo que varía lentamente sobre un valor promedio, por lo que en este diseño se supuso constante. Por último, se tiene η que representa el ruido gaussiano blanco.

b) Modelado del Acelerómetro

El acelerómetro permite medir aceleraciones lineales, así como también, la aceleración gravitacional del vehículo con respecto al sistema de navegación. Si las aceleraciones lineales en el Quadrotor, ya sean por ráfagas de viento, transiciones de vuelo o ruido introducido por las vibraciones, son pequeñas; entonces los ángulos de inclinación (pitch) y de giro (roll) se pueden obtener directamente. La lectura del acelerómetro que proporciona los componentes del vector gravedad está dado por:

$$a_m = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = gR_n^b \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} + \mu \tag{3.13}$$

Donde, a_m es la aceleración medida, $g = 9.81 m/s^2$ es la aceleración de la gravedad y μ es el ruido blanco gaussiano introducido en la medición.

Para el cálculo del ángulo de giro e inclinación se tienen las ecuaciones:

$$\phi = atan2(a_y, a_z) \tag{3.14}$$

$$\theta = -\operatorname{asin}\left(\frac{a_x}{|a_m|}\right) \tag{3.15}$$

Donde, $|a_m| = \sqrt{{a_x}^2 + {a_y}^2 + {a_z}^2}$ es la magnitud de la aceleración resultante.

c) <u>Modelado del Magnetómetro</u>

Mediante el magnetómetro de 3 ejes, se puede obtener el ángulo de desviación (ψ), ángulo en el cual el acelerómetro no es suficiente. El magnetómetro mide la intensidad de campo magnético en el sistema Body. Entonces para hallar las mediciones en coordenadas Body, se tiene la siguiente ecuación:

$$m_m = \begin{bmatrix} m_x \\ m_y \\ m_z \end{bmatrix} = R_n^b h + \mu \tag{3.16}$$

Donde; μ es ruido blanco gaussiano, h es el vector del campo magnético de la tierra que tiene la siguiente expresión:

$$h = \begin{bmatrix} h_x \\ h_y \\ h_z \end{bmatrix} = |h| \begin{bmatrix} \cos(I) \\ 0 \\ sen(I) \end{bmatrix} \tag{3.17}$$

Donde, I es la inclinación del vector del campo magnético. En el Perú, Lima $I = 0°5'$. En la figura 3.12 se muestra una representación gráfica del vector gravedad y del campo magnético de la tierra.

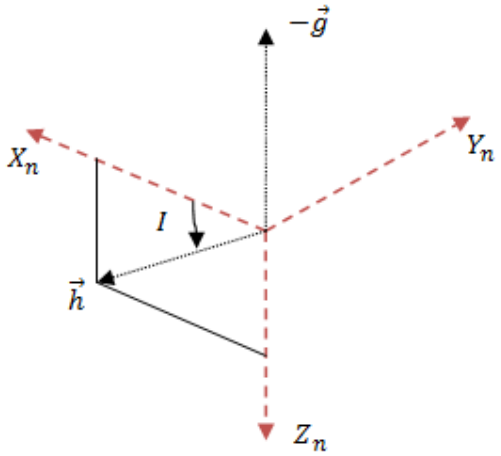

Figura 3.12: Vector de la gravedad y del campo magnético de la tierra en coordenadas de navegación

La ecuación para determinar el ángulo de desviación (yaw), que compensa las inclinaciones en el eje de giro e inclinación, es el siguiente:

$$\psi = atan2(-m_{y1}, m_{x1}) \tag{3.18}$$

Donde;

$$m_{y1} = m_y \cos\phi - m_z sen\phi$$

$$m_{x1} = m_x \cos\theta + m_y sen\phi sen\theta + m_z cos\phi sen\theta$$

B. Algoritmo Filtro de Kalman Lineal

El Filtro de Kalman consiste de un algoritmo de fusión de sensores, que en este caso compensa la deriva resultante en la integración de las mediciones del giroscopio, con las lecturas ruidosas del acelerómetro y magnetómetro (Pflimlin, Hamel, Soueres, & Metni, 2005).

Para que la estimación obtenida mediante el filtro de Kalman sea óptimas, se considera que el modelo del sistema sea completamente lineal y que sólo esté afectado por ruido con distribución gaussiana. Por ello se desarrolló primero un Filtro de Kalman Lineal, cuyo esquema se muestra en la figura 3.13, cuyo rendimiento será comparado mediante pruebas con los Filtros de Kalman Extendido.

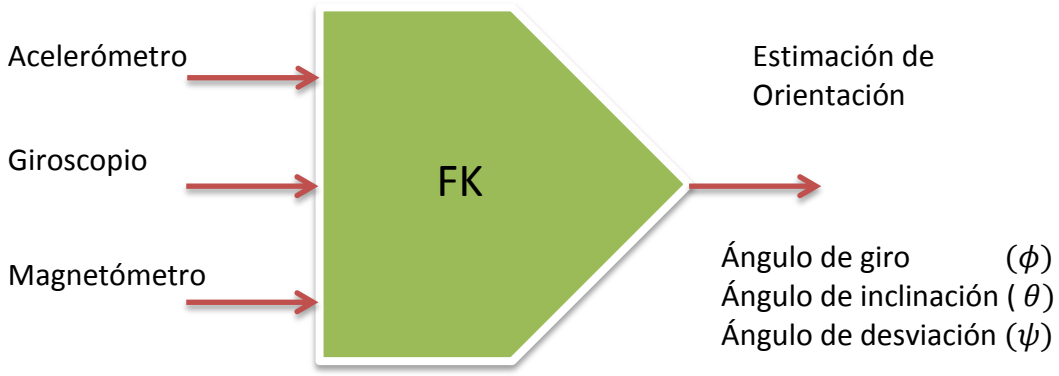

Figura 3.13: Esquema del Filtro de Kalman Lineal

En el filtro de Kalman se desarrolló el sistema dinámico giroscopio-acelerómetro que toma en cuenta el modelo del giroscopio con un sesgo y ruido blanco gaussiano con media cero.

El sistema fue analizado para un eje, en este caso para el ángulo de giro (ϕ), ya que de manera semejante se puede aplicar para los dos ejes restantes y de manera independiente.

Vector de estados

Como en este filtro se plantea estimar el ángulo de giro (ϕ) y el sesgo del giroscopio (ω_b), entonces el vector de estados se define como:

$$x = \begin{bmatrix} \phi \\ \omega_b \end{bmatrix}_k \tag{3.19}$$

Modelo del proceso y medición

Las ecuaciones para el modelo del proceso y de medición en tiempo discreto son:

$$\begin{bmatrix} \phi \\ \omega_b \end{bmatrix}_{k+1} = \begin{bmatrix} 1 & -T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \phi \\ \omega_b \end{bmatrix}_k + \begin{bmatrix} T \\ 0 \end{bmatrix} \omega_{k+1} + N(0, Q_{k+1}) \tag{3.20}$$

$$\phi_{A,k+1} = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \phi \\ \omega_b \end{bmatrix}_{k+1} + N(0, R_{k+1}) \tag{3.21}$$

Donde T es el periodo de muestreo, ω_{k+1} son las mediciones de las velocidades angulares del giroscopio y $\phi_{A,k+1}$ son las mediciones del ángulo de giro calculado con los datos del acelerómetro mediante la ecuación (3.14). Quedando la expresión siguiente:

$$\phi_A = atan2(a_y, a_z) \tag{3.22}$$

Con estos modelos obtenidos, se puede aplicar el filtro de Kalman usando las ecuaciones (2.39) – (2.43).

En la sección siguiente se desarrollan los dos Filtros de Kalman Extendido que tienen algunas ventajas sobre el Filtro de Kalman Lineal en el cálculo de orientación del Quadrotor.

C. Algoritmo Filtro de Kalman Extendido

Existen diferentes enfoques y alternativas de algoritmos de fusión de datos de sensores, como son: El EKF (Extended Kalman Filter), UKF (Unscented Kalman Filter), filtro complementario entre otros (Wan & Van der Merwe, 2000). Para realizar una elección correcta primero se describen las ventajas y desventajas del Filtro de Kalman Extendido (EKF), así como las comparaciones con otros algoritmos.

Las ventajas del Filtro de Kalman Extendido son:

- Es un método de estimación óptimo y robusto ya que minimiza el error cuadrático medio (MMSE).
- De relativa simplicidad y tratabilidad.
- Capacidad de fusionar diferentes tipos de sensores cada uno con su propia dinámica y modelo de errores.

Las desventajas del Filtro de Kalman Extendido son:

- El proceso de linealización de la matriz de transición de estado y de observación, mediante la expansión de series de Taylor, puede producir inestabilidad en el filtro (Julier & Uhlmann, 1997).

Comparando el EKF con el filtro complementario; el filtro complementario representa una solución efectiva, pero sólo tiene un buen rendimiento para una IMU (acelerómetro, giroscopio) (Madgwick, 2010).

Comparando el EKF con el UKF, éste último muestra un nivel superior de exactitud en diversos sistemas no lineales, pero en aplicaciones particulares de estimación de orientación, movimiento y sistemas de navegación mediante el uso de cuaterniones, el EFK y el UKF han demostrado tener rendimientos equivalentes; sin embargo, el UKF presenta un mayor cálculo computacional. En algunas aplicaciones el

cálculo de la matriz Jacobiana del EKF puede traer dificultades de implementación, pero en este caso las matrices son simples debido a la estructura de las funciones del proceso y de medición (La Viola, 2003). Por estas razones es que el EKF es la elección más apropiada para la estimación de orientación del Quadrotor.

En los Filtros de Kalman Extendido implementados no se consideró el modelo dinámico del Quadrotor en el modelo de proceso del filtro. Este enfoque tiene la ventaja, de que al no estar sujeto a las dinámicas de la planta del Quadrotor se podría utilizar este sistema en otros tipos de vehículos, como: vehículos aéreos de ala fija (aviones), vehículos terrestres o acuáticos, cohetes, entre otros. Además, este enfoque tiene un menor cálculo computacional, por lo que se puede implementar en un microprocesador de baja capacidad de procesamiento, y en consecuencia de bajo costo.

Para este trabajo se desarrollaron dos filtros de Kalman Extendidos (EKF), los cuales difieren en el modelo de medición pero con el mismo modelo matemático del proceso, además este último con una Matriz de Rotación adicional para el cálculo del ángulo de desviación, los filtros desarrollados son:

a) Filtro de Kalman Extendido.
b) Filtro de Kalman Extendido y Matriz de Rotación.

En el primer Filtro de Kalman Extendido (EKF) se utilizan directamente las lecturas del acelerómetro y magnetómetro para estimar el ángulo de giro, inclinación y desviación en la etapa de corrección del filtro (figura 3.14). En el segundo EKF se utiliza la medición del acelerómetro para calcular el ángulo de giro e inclinación, mientras que para el ángulo de desviación se utilizó una Matriz de Rotación con ángulos de Euler, como se muestra en la figura 3.15. El primer EKF es el modelo típico en la tarea de estimación de orientación, pero en este proyecto se añadió este segundo enfoque, con una Matriz de

Rotación, con el fin de obtener mejoras en las estimaciones, lo cual se analizará mediante pruebas.

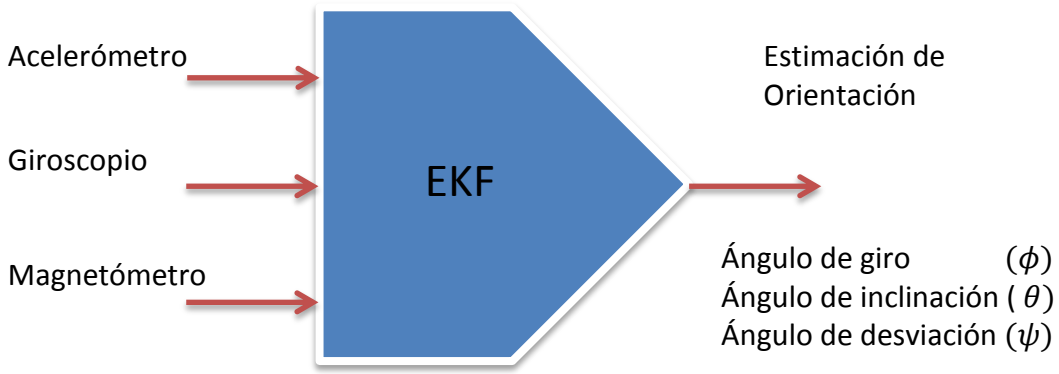

Figura 3.14: Esquema del Filtro de Kalman Extendido

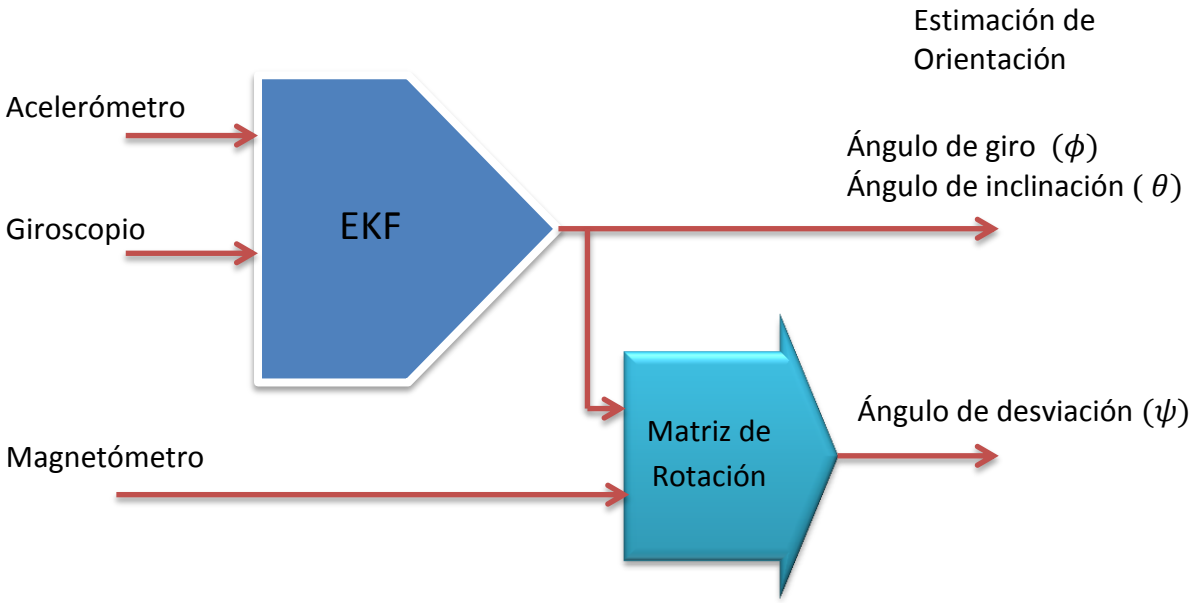

Figura 3.15: Esquema del Filtro de Kalman Extendido y Matriz de Rotación

a) Filtro de Kalman Extendido

Este filtro se muestra en el esquema de la figura 3.14, en la cual se utilizaron las mediciones del giroscopio para la etapa de predicción, mientras que para la etapa de corrección o actualización del vector de estados se utilizaron el acelerómetro y magnetómetro. La representación de orientación del Quadrotor se realizó mediante cuaterniones.

(i) Modelo del proceso

Para este modelo se toma en cuenta el sesgo (bias) del giroscopio, que está incluido en el vector de estados. Dicho vector de estados está conformado por la estimación de la actitud mediante cuaterniones y los sesgos del giroscopio en cada eje, como se muestra en la ecuación (3.23) definido en tiempo discreto.

$$x_k = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \\ \omega_{xbias} \\ \omega_{ybias} \\ \omega_{zbias} \end{bmatrix}_k = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix}_k \tag{3.23}$$

En este modelo se considera que el sesgo no depende del tiempo, por lo tanto, se asume que tiene un valor constante. Entonces, si no hay un cambio de temperatura considerable en el ambiente, se tiene la siguiente ecuación del sesgo:

$$\dot{\omega}_{bias} = \begin{bmatrix} \dot{\omega}_{xbias} \\ \dot{\omega}_{ybias} \\ \dot{\omega}_{zbias} \end{bmatrix}_k = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}_k \tag{3.24}$$

Como ecuación fundamental en la dinámica del sistema del filtro se considera la derivada en el tiempo del cuaternión, que está representado en la ecuación (2.26). Por lo tanto, el cuaternión se determina mediante una integración, representado por la ecuación

(3.25), en donde se tienen en cuenta: el sesgo del giroscopio, la estimación anterior y la velocidad angular del giroscopio.

$$q_k = q_{k-1} + T * \dot{q_k} \tag{3.25}$$

Donde, T es el periodo de muestreo y

$$\dot{q_k} = \frac{1}{2}\begin{bmatrix}0\\ \omega\end{bmatrix} \otimes q_k \tag{3.26}$$

Operando el producto cuaternión de la ecuación (2.19), se obtiene:

$$\dot{q_k} = \frac{1}{2}\begin{bmatrix}0 & -\omega_x & -\omega_y & -\omega_z\\ \omega_x & 0 & \omega_z & -\omega_y\\ \omega_y & -\omega_z & 0 & \omega_x\\ \omega_z & \omega_y & -\omega_x & 0\end{bmatrix}\begin{bmatrix}q_0\\ q_1\\ q_2\\ q_3\end{bmatrix} \tag{3.27}$$

Como se considera el sesgo en la estimación de actitud, se resta de las mediciones del giroscopio:

$$\dot{q_k} = \frac{1}{2}\begin{bmatrix}0 & -(\omega_x - \omega_{xbias}) & -(\omega_y - \omega_{ybias}) & -(\omega_z - \omega_{zbias})\\ (\omega_x - \omega_{xbias}) & 0 & (\omega_z - \omega_{zbias}) & -(\omega_y - \omega_{ybias})\\ (\omega_y - \omega_{ybias}) & -(\omega_z - \omega_{zbias}) & 0 & (\omega_x - \omega_{xbias})\\ (\omega_z - \omega_{zbias}) & (\omega_y - \omega_{ybias}) & -(\omega_x - \omega_{xbias}) & 0\end{bmatrix}\begin{bmatrix}q_0\\ q_1\\ q_2\\ q_3\end{bmatrix} \tag{3.28}$$

Operando la multiplicación se obtiene la derivada en el tiempo del cuaternión, como sigue:

$$\dot{q_k} = \frac{1}{2}\begin{bmatrix}-q_1(\omega_x - \omega_{xbias}) - q_2(\omega_y - \omega_{ybias}) - q_3(\omega_z - \omega_{zbias})\\ q_0(\omega_x - \omega_{xbias}) + q_2(\omega_z - \omega_{zbias}) - q_3(\omega_y - \omega_{ybias})\\ q_0(\omega_y - \omega_{ybias}) - q_1(\omega_z - \omega_{zbias}) + q_3(\omega_x - \omega_{xbias})\\ q_0(\omega_z - \omega_{zbias}) + q_1(\omega_y - \omega_{ybias}) - q_2(\omega_x - \omega_{xbias})\end{bmatrix}_k \tag{3.29}$$

Por lo tanto, la ecuación en el espacio de estados que representa el modelo del proceso f en tiempo discreto, se calcula reemplazando la ecuación (3.29) en la ecuación

(3.25) cuyo resultado se muestra en la ecuación (3.30), y se expresa en términos de vector de estado como:

$$x_k = f(x_{k-1}, u_k) = \begin{bmatrix} x_1 + \frac{T}{2}\left[-x_2(\omega_x - x_5) - x_3(\omega_y - x_6) - x_4(\omega_z - x_7)\right] \\ x_2 + \frac{T}{2}\left[x_1(\omega_x - x_5) + x_3(\omega_z - x_7) - x_4(\omega_y - x_6)\right] \\ x_3 + \frac{T}{2}\left[x_1(\omega_y - x_6) - x_2(\omega_z - x_7) + x_4(\omega_x - x_5)\right] \\ x_4 + \frac{T}{2}\left[x_1(\omega_z - x_7) + x_2(\omega_y - x_6) - x_3(\omega_x - x_5)\right] \\ x_5 \\ x_6 \\ x_7 \end{bmatrix}_k \tag{3.30}$$

Para aplicar el filtro de Kalman, el modelo del proceso debe ser linealizado mediante derivadas parciales o Jacobiano, obteniendo como la matriz de transición de estados la siguiente expresión:

$$\left[\frac{\partial f}{\partial x}\right]_k = F_k \tag{3.31}$$

$$F_k = \begin{bmatrix} 1 & -\frac{T}{2}(\omega_x - x_5) & -\frac{T}{2}(\omega_y - x_6) & -\frac{T}{2}(\omega_z - x_7) & \frac{T}{2}x_2 & \frac{T}{2}x_3 & \frac{T}{2}x_4 \\ \frac{T}{2}(\omega_x - x_5) & 1 & \frac{T}{2}(\omega_z - x_7) & -\frac{T}{2}(\omega_y - x_6) & -\frac{T}{2}x_1 & \frac{T}{2}x_4 & -\frac{T}{2}x_3 \\ \frac{T}{2}(\omega_y - x_6) & -\frac{T}{2}(\omega_z - x_7) & 1 & \frac{T}{2}(\omega_x - x_5) & -\frac{T}{2}x_4 & -\frac{T}{2}x_1 & \frac{T}{2}x_2 \\ \frac{T}{2}(\omega_z - x_7) & \frac{T}{2}(\omega_y - x_6) & -\frac{T}{2}(\omega_x - x_5) & 1 & \frac{T}{2}x_3 & -\frac{T}{2}x_2 & -\frac{T}{2}x_1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}_k$$

(ii) Modelo de medición

En este modelo de medición $h(x_k)$, se realizan las lecturas del acelerómetro y del magnetómetro para la etapa de corrección del Filtro de Kalman Extendido, en la cual se

comparan la predicción obtenida mediante el modelo del proceso, con los valores obtenidos de los sensores. Se define entonces el vector de medición como:

$$z_k = \begin{bmatrix} a_x \\ a_y \\ a_z \\ m_x \\ m_y \\ m_z \end{bmatrix}_k \tag{3.32}$$

El vector debe estar normalizado, para que realice una correcta representación mediante cuaterniones.

Medición del acelerómetro

Para determinar los componentes del modelo de medición mediante el acelerómetro se hace rotar, usando cuaterniones, el vector fuerza de gravedad hacia el sistema de coordenadas Body, como se indica en la ecuación (3.13). De esta manera, se obtienen los tres primeros componentes del modelo de medición.

En términos del vector de estado, se obtiene la siguiente expresión:

$$h(x_k) = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = R_n^b \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 2(x_2x_4 - x_1x_3) \\ 2(x_1x_2 + x_3x_4) \\ 1 - 2({x_2}^2 + {x_3}^2) \end{bmatrix}_k \tag{3.33}$$

Medición del magnetómetro

Para los tres siguientes componentes del modelo de medición, se hace rotar el vector de intensidad de campo magnético desde el sistema de navegación hacia el sistema de coordenadas Body. Se obtiene entonces la siguiente ecuación:

$$h(x_k) = \begin{bmatrix} m_x \\ m_y \\ m_z \end{bmatrix} = R_n^b . h \tag{3.34}$$

De acuerdo a la ecuación (3.17):

$$h = \begin{bmatrix} h_x \\ h_y \\ h_z \end{bmatrix} = |h| \begin{bmatrix} \cos(I) \\ 0 \\ sen(I) \end{bmatrix} \tag{3.35}$$

En Lima-Perú la inclinación del vector de campo magnético es $I = 0°5'$, con magnitud $|h| = 0.25G$ y normalizando el vector, se obtiene:

$$h = \begin{bmatrix} h_x \\ h_y \\ h_z \end{bmatrix} = |h| \begin{bmatrix} \cos(0°5') \\ 0 \\ sen(0°5') \end{bmatrix} \approx \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{3.36}$$

Reemplazando h en la ecuación (3.34) y con R_n^b de la ecuación (2.23), el modelo matemático en términos de vector de estado se expresa como:

$$h(x_k) = \begin{bmatrix} m_x \\ m_y \\ m_z \end{bmatrix} = \begin{bmatrix} 1 - 2({x_3}^2 + {x_4}^2) \\ 2(x_2x_3 - x_1x_4) \\ 2(x_2x_4 + x_1x_3) \end{bmatrix}_k \tag{3.37}$$

Por lo tanto el modelo de medición obtenido mediante la medición de la aceleración gravitacional y del campo magnético de la tierra, se calcula uniendo las ecuaciones (3.33) y (3.37), quedando finalmente:

$$h(x_k) = \begin{bmatrix} 2(x_2x_4 - x_1x_3) \\ 2(x_3x_4 + x_1x_2) \\ 1 - 2({x_2}^2 + {x_3}^2) \\ 1 - 2({x_3}^2 + {x_4}^2) \\ 2(x_2x_3 - x_1x_4) \\ 2(x_2x_4 + x_1x_3) \end{bmatrix}_k \tag{3.38}$$

La matriz de medición se obtiene del Jacobiano de h :

$$H_k = \left[\frac{\partial h}{\partial x}\right]_k = \begin{bmatrix} -2x_3 & 2x_4 & -2x_1 & 2x_2 & 0 & 0 & 0 \\ 2x_2 & 2x_1 & 2x_4 & 2x_3 & 0 & 0 & 0 \\ 0 & -4x_2 & -4x_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & -4x_3 & -4x_4 & 0 & 0 & 0 \\ -2x_4 & 2x_3 & 2x_2 & -2x_1 & 0 & 0 & 0 \\ 2x_3 & 2x_4 & 2x_1 & 2x_2 & 0 & 0 & 0 \end{bmatrix}_k \tag{3.39}$$

Una vez obtenidos los modelos del proceso y de la medición éstos se pueden aplicar en las ecuaciones de predicción y corrección del EKF desde las ecuaciones (2.56) hasta el (2.60).

b) <u>Filtro de Kalman Extendido y Matriz de Rotación</u>

En este segundo modelo de EKF se utilizan solamente las lecturas del acelerómetro en el modelo de medición para el cálculo del roll y pitch. Para el cálculo del yaw se usó una Matriz de Rotación mediante ángulos de Euler con compensación de inclinaciones. Este planteamiento, es con el objetivo de obtener mejores estimaciones de orientación en ambientes que pudieran generar interferencias magnéticas.

Analizando el modelo de ambos filtros estas interferencias magnéticas afectaría al primer EKF, en los ángulos de giro, inclinación y desviación, mientras que en este segundo (EKF y Matriz de Rotación) sólo afectaría al ángulo de desviación. Lo que se busca entonces es que a pesar de interferencias magnéticas en el Quadrotor se mantenga estable en vuelo. Este nuevo enfoque en contraparte supondría un mayor cálculo computacional.

(i) Modelo de proceso

El modelo del proceso es el mismo que el desarrollado para el primer Filtro de Kalman Extendido, así como también su vector de estado.

(ii) Modelo de medición

En este modelo solamente se toman las lecturas del acelerómetro para el cálculo de los ángulos de giro e inclinación. El vector de medición se expresa como:

$$z_k = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}_k \tag{3.40}$$

El modelo de medición queda como se indicó en la ecuación (3.33), utilizando solamente el acelerómetro. Finalmente se obtiene el Jacobiano de h, con la siguiente expresión:

$$H_k = \left[\frac{\partial h}{\partial x}\right]_k = \begin{bmatrix} -2x_3 & 2x_4 & -2x_1 & 2x_2 & 0 & 0 & 0 \\ 2x_2 & 2x_1 & 2x_4 & 2x_3 & 0 & 0 & 0 \\ 0 & -4x_2 & -4x_3 & 0 & 0 & 0 & 0 \end{bmatrix}_k \tag{3.41}$$

(iii) Cálculo del ángulo de desviación mediante Rotación de Matrices

Para la estimación del ángulo de desviación se utilizaron las estimaciones de los ángulos de giro e inclinación, obtenidos del algoritmo anterior desarrollado. Los valores de los ángulos se ingresan en una matriz de rotación que utiliza ángulos de Euler, como se muestra en la ecuación (3.18), en la cual se calcula el ángulo de desviación compensando además las inclinaciones en los ejes de los ángulos de giro e inclinación.

Por consiguiente, el ángulo de desviación se expresa como:

$$\psi = atan2(-m_{y1}, m_{x1}) \tag{3.42}$$

Donde,

$$m_{y1} = m_y \cos\phi_{EKF} - m_z sen\phi_{EKF} \tag{3.43}$$

$$m_{x1} = m_x \cos\theta_{EKF} + m_y sen\phi_{EKF} sen\theta_{EKF} + m_z cos\phi_{EKF} sen\theta_{EKF} \quad (3.44)$$

D. Matrices de covarianzas de ruido del sistema y de la medición

El valor de la matriz de covarianza de ruido de la medición R_k se determina directamente mediante muestras evaluadas fuera de línea del filtro, en pruebas experimentales con el Quadrotor. Y los valores de la matriz de covarianza de ruido del sistema Q_k se calculan realizando un ajuste manual a fin de obtener la mejor estimación.

3.3 Etapa de simulación

En esta sección se muestran los resultados de las diferentes simulaciones realizadas al sistema desarrollado, los cuales son:

- Simulación del control de velocidad PID para el motor DC.
- Simulación del Filtro de Kalman Lineal para el eje roll.
- Simulación del Filtro de Kalman Extendido para el eje roll.

3.3.1 Simulación del control PID de velocidad para el motor DC

En la figura 3.16 se muestra la gráfica de las simulaciones con tres diferentes tipos de control, los cuales son: Proporcional (P), proporcional-Integral (PI) y Proporcional-Integral-Derivativo (PID), eligiendo la ecuación (3.10) como expresión de la planta.

Como se muestra en la figura 3.16, el resultado del control proporcional (color azul) no se realiza un buen seguimiento a la señal deseada. Por otro lado, el control PID genera inestabilidad en la señal de la planta de primer orden. Sin embargo, el control PI genera una respuesta favorable ante la entrada escalón. En conclusión, el controlador

más adecuado a implementar resulta ser el Proporcional-Integral (PI). Afín de mejorar su sintonización, se utilizó el ToolBox PID Tuning de Matlab (MathWorks, Inc.).

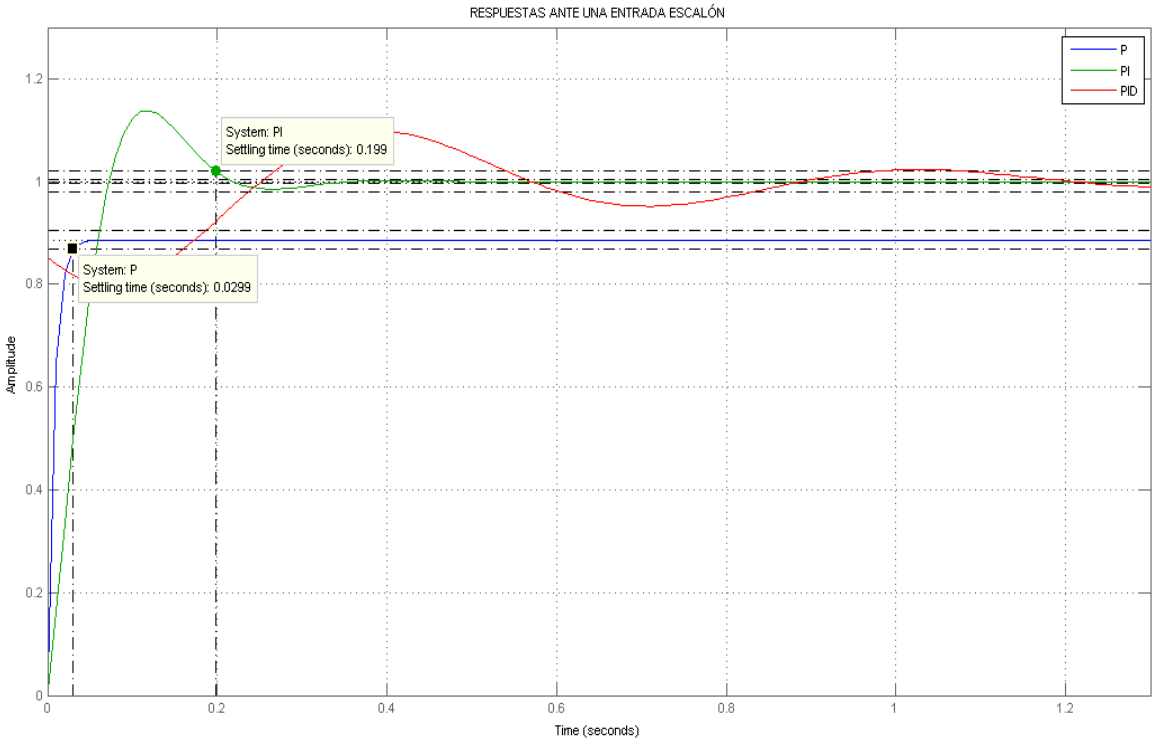

Figura 3.16: Simulación de los controladores: P, PI, PID

La sintonía del controlador es el proceso de establecer sus parámetros para que cumpla con las especificaciones de comportamiento requeridas. En este caso se utilizó el ToolBox PID Tuning de Matlab, con el modelo de primer orden a la que fue aproximada la planta (ecuación (3.10)).

En la figura 3.17 se observa que la respuesta del sistema ante una entrada tipo escalón, con un controlador PID con parámetros K_p , K_i y K_d igual a 0.26788, 11.9791 y 0 respectivamente. El valor de cero en la ganancia derivativa muestra que sólo es necesario un control PI para un control óptimo, como se había mencionado. Los resultados de la sintonización PID se muestran en la tabla 3.2.

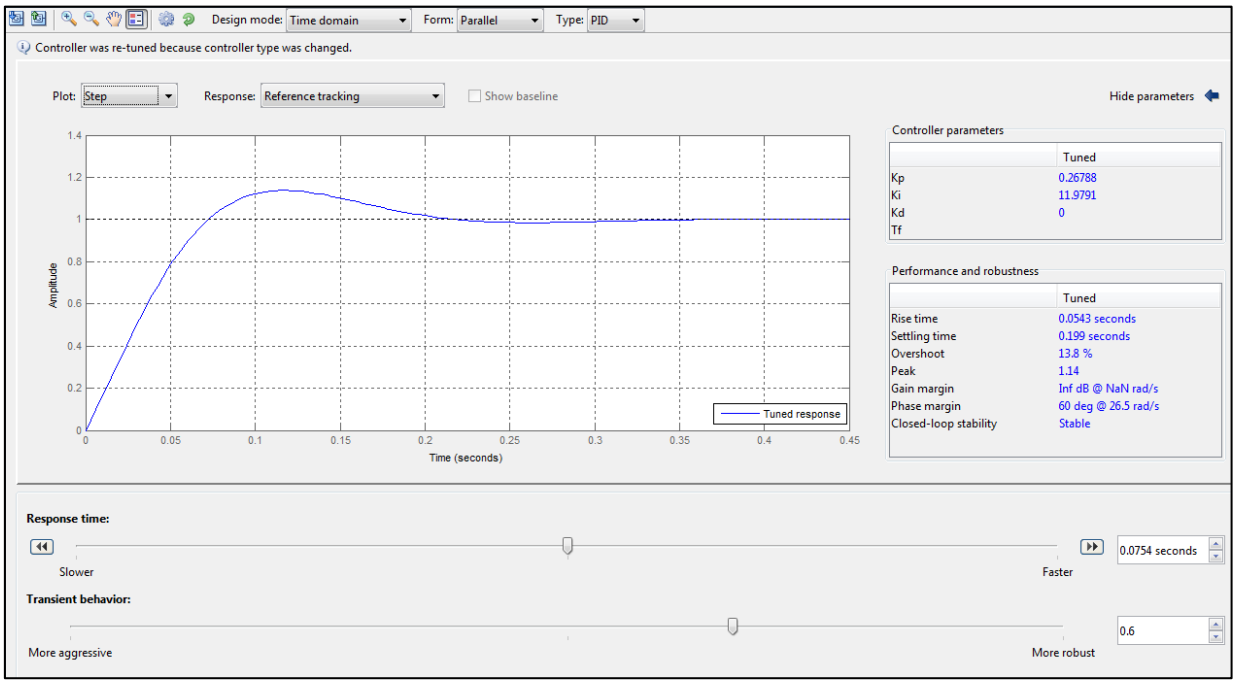

Figura 3.17: Sintonización con el PID Tuning de Matlab

Tabla 3.2: Valores de sintonización del control PI

ESTABILIDAD	SINTONIZADO
Tiempo de subida	0.0543 (s)
Tiempo de establecimiento	0.199 (s)
Sobreimpulso	13.8 (%)
valor pico	1.14
Kp	0.26788
Ki	11.9791
Kd	0

3.3.2 Simulación del Algoritmo del Filtro de Kalman Lineal y Extendido

En esta sección muestran las simulaciones tanto del Filtro de Kalman Lineal, como del Extendido. Las simulaciones se realizan solamente en el eje del ángulo de giro ya que para los otros dos ejes el procedimiento es semejante. Los resultados fueron suficientes para analizar el rendimiento de cada uno de ellos y realizar una comparación. También se analizó sólo el primer Filtro de Kalman Extendido de la sección C, ya que el segundo filtro difiere del primero, principalmente en el cálculo del ángulo de desviación.

Simulación de datos del acelerómetro y giroscopio

Antes de analizar los filtros, se tiene que simular los datos de los sensores, a los cuales se les añadió ruido del tipo gaussiano blanco. Para analizar el ángulo de giro, se simuló la información del acelerómetro y del giroscopio.

Los parámetros establecidos para la simulación de los sensores, se visualizan en la tabla 3.3. y en la figura 3.18 se muestra el resultado gráfico de dichas simulaciones.

Tabla 3.3: Parámetros de simulación para los sensores

PARÁMETRO	VALOR
Periodo de muestreo	0.01 (s)
Bias giroscopio	2 (°/s)
Desviación estándar del acelerómetro	19.97 (°)
Desviación estándar del giroscopio	2.9572 (°/s)

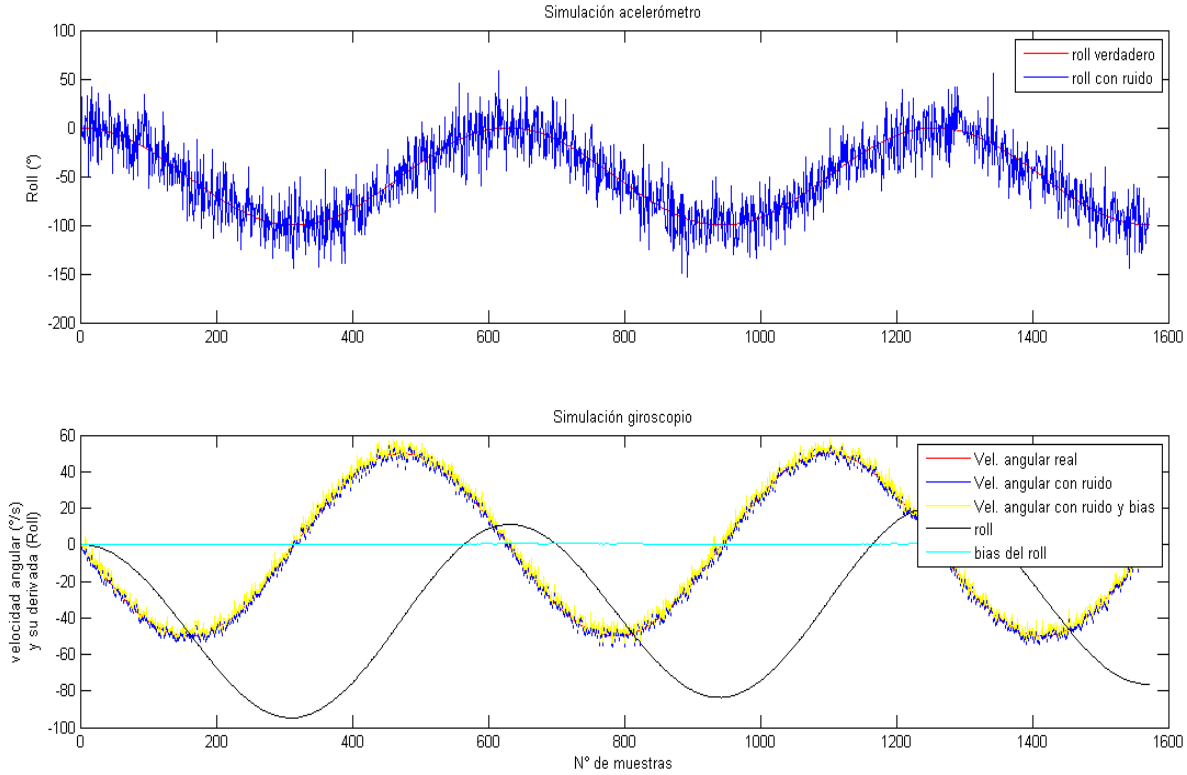

Figura 3.18: Simulación de datos del acelerómetro y giroscopio

En la figura 3.18, se observa que en la simulación de datos generados por el acelerómetro, se muestra la señal ruidosa del ángulo de giro (de color azul), y la señal verdadera o sin ruido (de color rojo). En la simulación del giroscopio se muestra de color rojo la velocidad angular verdadera o sin ruido ni sesgo, y de color negro el ángulo de giro obtenido tras realizar la integración de la velocidad angular.

Simulación del Filtro de Kalman Lineal

Los parámetros que se establecieron para este filtro son:

- Covarianza de la medición: 398.800
- Matriz de covarianza del proceso:

$$Q = \begin{bmatrix} 0.00087 & 0 \\ 0 & 0.0001 \end{bmatrix}$$

En la figura 3.19 se muestra el resultado de la simulación del filtro de Kalman Lineal desarrollado en la sección B, utilizando los datos simulados por los sensores. Se observa que una señal bastante ruidosa (color azul), representa los ángulos obtenidos del acelerómetro. La señal de color magenta muestra los ángulos calculados a partir del giroscopio, los cuales tienen bajo nivel de ruido pero tienen un sesgo o bias el cual incrementa su error conforme transcurre el tiempo. La señal verdadera o real es de color negro y el Filtro de Kalman Lineal de color verde, el cual realiza un buen seguimiento a dicha señal verdadera; concluyendo que este algoritmo genera resultados favorables en simulación.

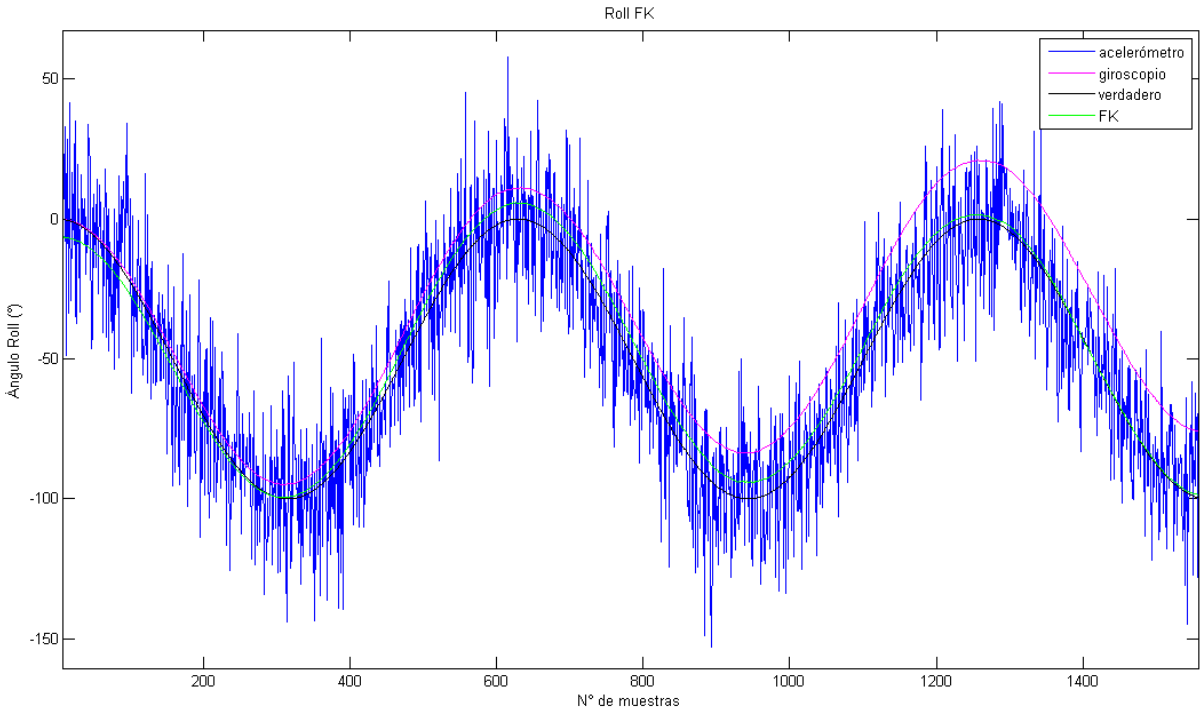

Figura 3.19: Simulación del Filtro de Kalman Lineal (FK)

Para evaluar el rendimiento del filtro numéricamente se utilizan índices de rendimiento, lo cuales son indicativos del error, precisión y exactitud, como se detallan en la sección 2.2.7.

De los resultados mostrados en la tabla 3.4, se observa que el error RMS, índice del error existente entre la estimación y el valor verdadero, tiene un valor de 4.5264°, lo que es indicativo de un rendimiento favorable para analizar este filtro en su implementación. La media y la varianza son indicativos de la precisión y exactitud del filtro.

Tabla 3.4: Índices de rendimiento de la simulación del Filtro de Kalman Lineal

ÍNDICE DE RENDIMIENTO DEL FILTRO DE KALMAN LINEAL	**RESULTADO**
Media del error $[grados]$	5.0295
Varianza del error $[grados^2]$	5.3395
Error RMS $[grados]$	**5.5350**

Simulación del Filtro de Kalman Extendido

Para la simulación de este filtro se eligen los mismos datos de los sensores (acelerómetro giroscopio), utilizados en la simulación del Filtro de Kalman Lineal, con la finalidad de realizar una comparación entre los resultados de los filtros. La covarianza de la medición y del proceso, para el Filtro de Kalman Extendido (EKF), son:

- Matriz de covarianza de ruido de la medición

$$R = \begin{bmatrix} 1e6 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1e6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1e6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1e9 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1e9 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1e9 \end{bmatrix}$$

- Matriz de covarianza de ruido de proceso del filtro de Kalman Extendido

$$Q = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.2 \end{bmatrix}$$

En la figura 3.20 se muestra el resultado de la simulación del Filtro de Kalman Extendido (EKF). Como se puede observar el EKF (color rojo) también realiza un buen seguimiento a la señal verdadera indicativo de buen rendimiento. Los índices de rendimiento para el Filtro de Kalman Extendido se muestran en la tabla 3.5.

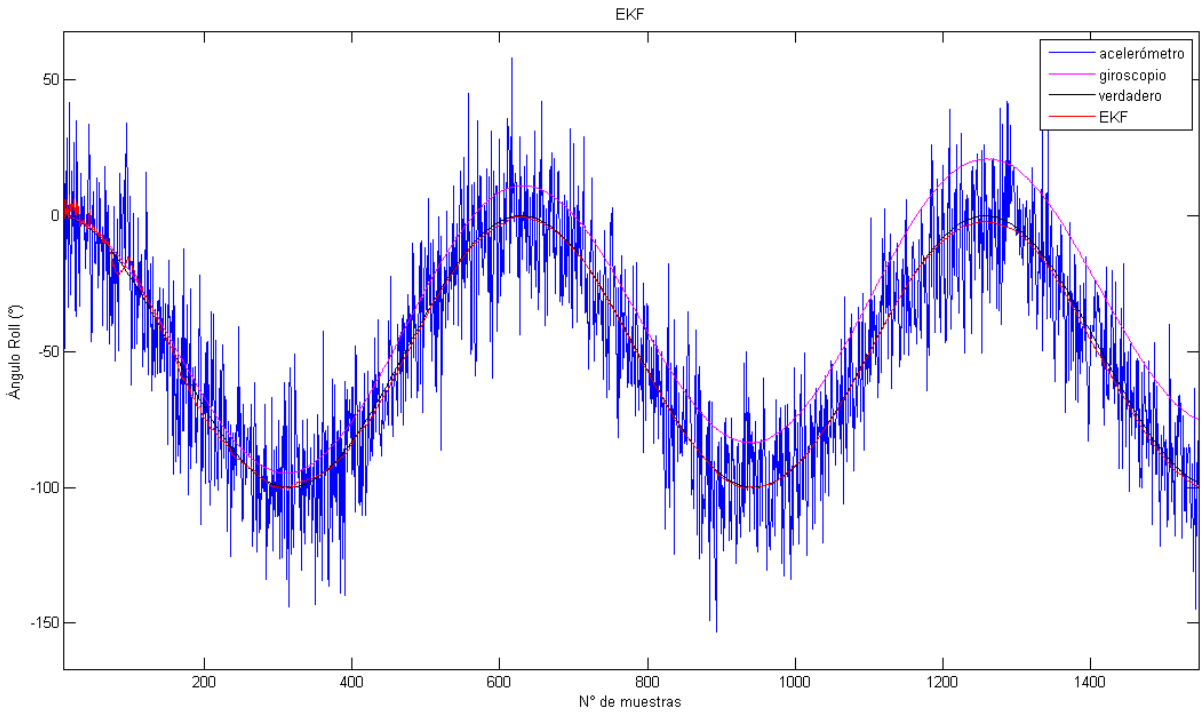

Figura 3.20: Simulación del Filtro de Kalman Extendido (EKF)

Tabla 3.5: Índices de rendimiento de la simulación del Filtro de Kalman Extendido

ÍNDICE DE RENDIMIENTO DEL FILTRO DE KALMAN EXTENDIDO	RESULTADO
Media del error $[grados]$	-0.6196
Varianza del error $[grados^2]$	0.6725
Error RMS $[grados]$	**1.0278**

Comparación entre el Filtro de Kalman Lineal (FK) y el Filtro de Kalman Extendido (EKF)

El Filtro de Kalman Lineal (FK) y el Filtro de Kalman Extendido (EKF) tienen rendimientos favorables para este proyecto, por lo que se realizó una comparación para conocer el mejor de los dos filtros según la simulación.

Como se observa en la figura 3.21, el EKF (señal color rojo) presenta un mejor seguimiento a la señal verdadera (color negro), que el Filtro de Kalman Lineal (color verde), lo cual se corrobora numéricamente mediante el error RMS de las tablas 3.4 y 3.5, en los cuales el Filtro de Kalman Lineal tiene un mayor error RMS de 4.5264° que el Filtro de Kalman Extendido con 1.2521°. Finalmente, se concluye mediante las simulaciones que el Filtro de Kalman Extendido es el más adecuado para la estimación de la orientación, pero todavía es necesario analizar sus rendimientos mediante pruebas reales con el Quadrotor.

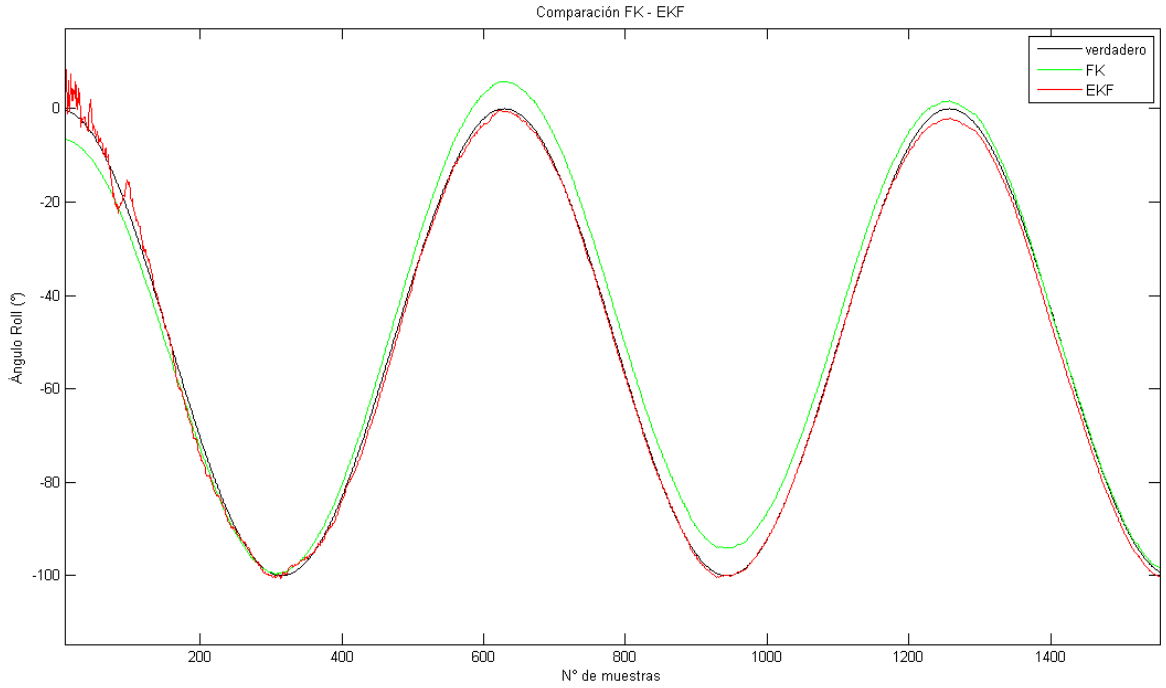

Figura 3.21: Comparación entre las simulaciones del Filtro de Kalman Lineal y Extendido

3.4 Implementación

Luego de realizar el diseño del sistema de determinación de orientación para el Quadrotor y las simulaciones necesarias de su funcionamiento, el siguiente paso consiste en la implementación en el FPGA DE0-Nano, para la evaluación de su rendimiento mediante pruebas reales con el Quadrotor.

Esta sección se divide en dos partes:

- Implementación de la calibración de sensores
- Implementación del sistema SOPC NIOS II

3.4.1 Implementación de la calibración de sensores

El objetivo principal consiste en determinar los doce parámetros de calibración para cada sensor, para implementar en el procesador NIOS II el algoritmo de calibración

o compensación de errores. Para este proceso se utilizó la plataforma mecánica de calibración construida.

Plataforma mecánica de calibración

En la figura 3.22 se muestra la comparación entre el diseño y la plataforma construida. Esta plataforma se utilizó para calibrar los tres sensores: acelerómetro, giroscopio y magnetómetro. El objetivo de usar esta plataforma es que mediante ésta se determinen los doce parámetros de calibración en cada sensor, los cuales contienen información del sesgo (offset), factor de escala, desalineamiento y no ortogonalidad.

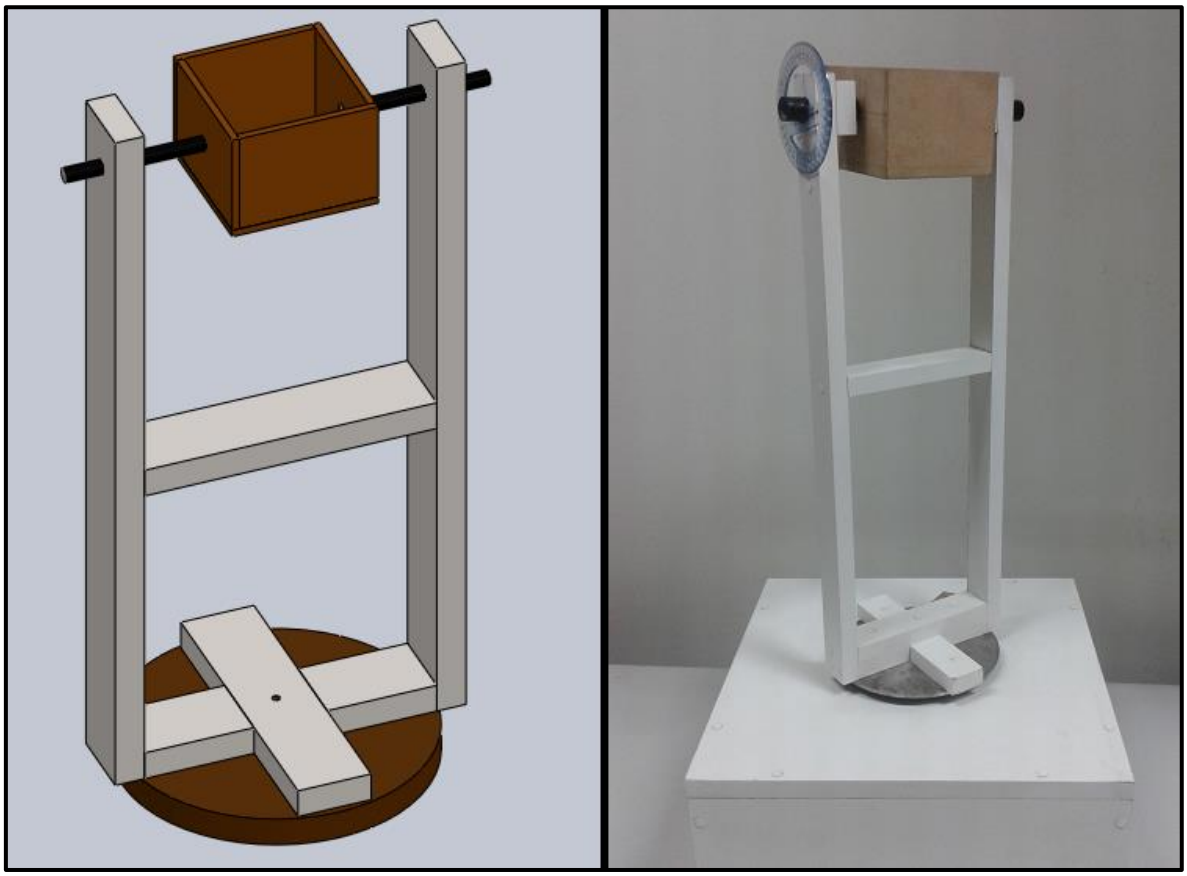

Figura 3.22: Comparación entre el diseño de la plataforma de calibración con la construida.

Parámetros de calibración del Giroscopio

El proceso para calcular los parámetros de calibración de este sensor se guía mediante el esquema de la figura 3.6.

Pero antes de realizar este proceso de determinar sus parámetros de calibración, es necesario implementar el control de velocidad PID en la tarjeta DE0-Nano.

a) Implementación del control de velocidad PID en la DE0-Nano

En la figura 3.23 se muestra la implementación del control de velocidad PID para el motor DC con el diseño que se muestra en la figura 3.8. En la figura 3.23 se muestran las conexiones entre el Procesador Nios II y los bloques: contador de pulsos y PWM, en el diagrama esquemático del software Quartus II. Estos bloques se desarrollaron utilizando un lenguaje de descripción de Hardware (VHDL) y el constructor SOPC Builder.

b) Implementación en la plataforma de calibración

Para encontrar los doce parámetros de calibración del giroscopio se rota este sensor a una velocidad constante, mientras que simultáneamente se obtienen datos y se cambia dicha velocidad cada cierto intervalo de tiempo. Este procedimiento se realiza en cada eje del sensor. Los datos que se obtienen mientras el sensor gira, son: la velocidad angular que indica el giroscopio y la velocidad de giro de la plataforma.

Para conocer la magnitud de la velocidad de giro de la plataforma se utilizan los datos del codificador del motor DC que se controla mediante la etapa de potencia y el control de velocidad PID implementado en la tarjeta DE0-Nano.

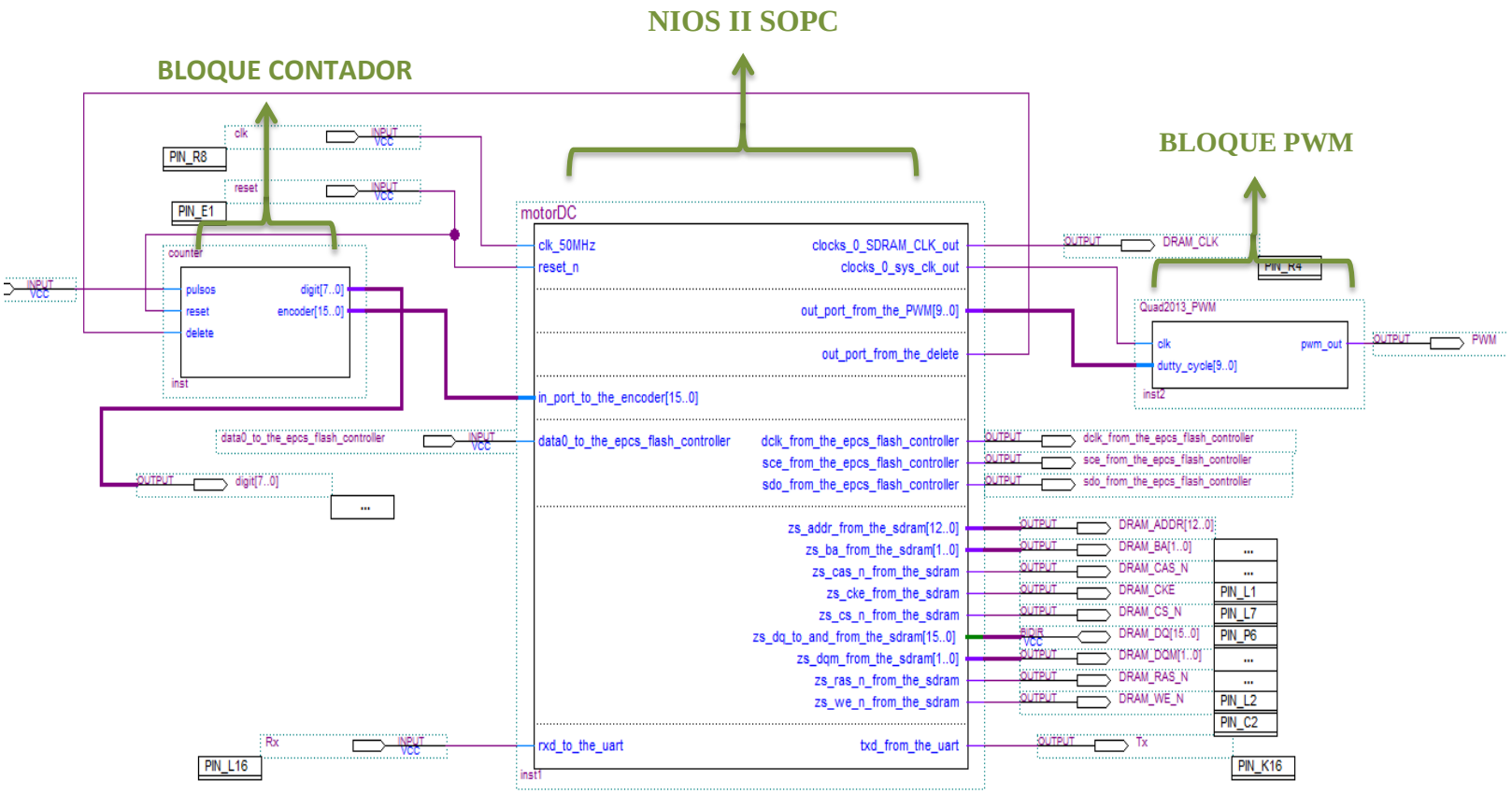

Figura 3.23: Esquema de conexiones entre el procesador Nios II con los bloques Contador y PWM para el control de velocidad del motor DC

Como las mediciones de velocidad debían ser de alta precisión debido a la alta precisión del sensor, solamente para la calibración del giroscopio se optó por separar de la plataforma el motor DC, ya que por el gran peso de la parte superior de la plataforma y la poca potencia del motor DC el control de velocidad bajaba de precisión.

En la figura 3.24 se muestra una fotografía en la cual se observa la implementación del sistema del proceso para determinar los parámetros de calibración del giroscopio.

Algunos de los resultados obtenidos mediante las pruebas realizadas se muestran en las figura 3.25 y 3.26, en las cuales se comparan los datos obtenidos mediante el codificador del motor DC (color rojo) considerado como referencia, del giroscopio (color azul) y los datos calibrados (color verde). Los puntos de consigna utilizados son de 5, 8, 13, 16, y 20 rad/s tomando 100 muestras por consigna. En las mismas figuras también se aprecia el cambio de coordenadas del sensor a coordenada Body.

Los resultados obtenidos son de buena precisión, obteniendo un error promedio de 1 $pulso/rev$ de un total de 100 $pulsos$ que genera el codificador en una vuelta. Después de procesar el algoritmo de mínimos cuadrados para determinar los doce parámetros de calibración, se obtuvo como resultado la siguiente matriz de compensación de errores (desalineamiento, factor de escala, no ortogonalidad y offset):

$$X = \begin{bmatrix} -1.0240 & 0.0551 & -0.0182 \\ 0.0227 & 1.0114 & -0.0059 \\ -0.0233 & 0.0319 & -1.0104 \\ -0.3138 & -0.1090 & 0.3443 \end{bmatrix}$$

Los resultados fueron favorables ya que se logró reducir el error de velocidad angular del giroscopio sin calibrar de 10.5388 °/s a un error con la calibración de 1.5504 °/s, logrando con esto una mejora en la precisión.

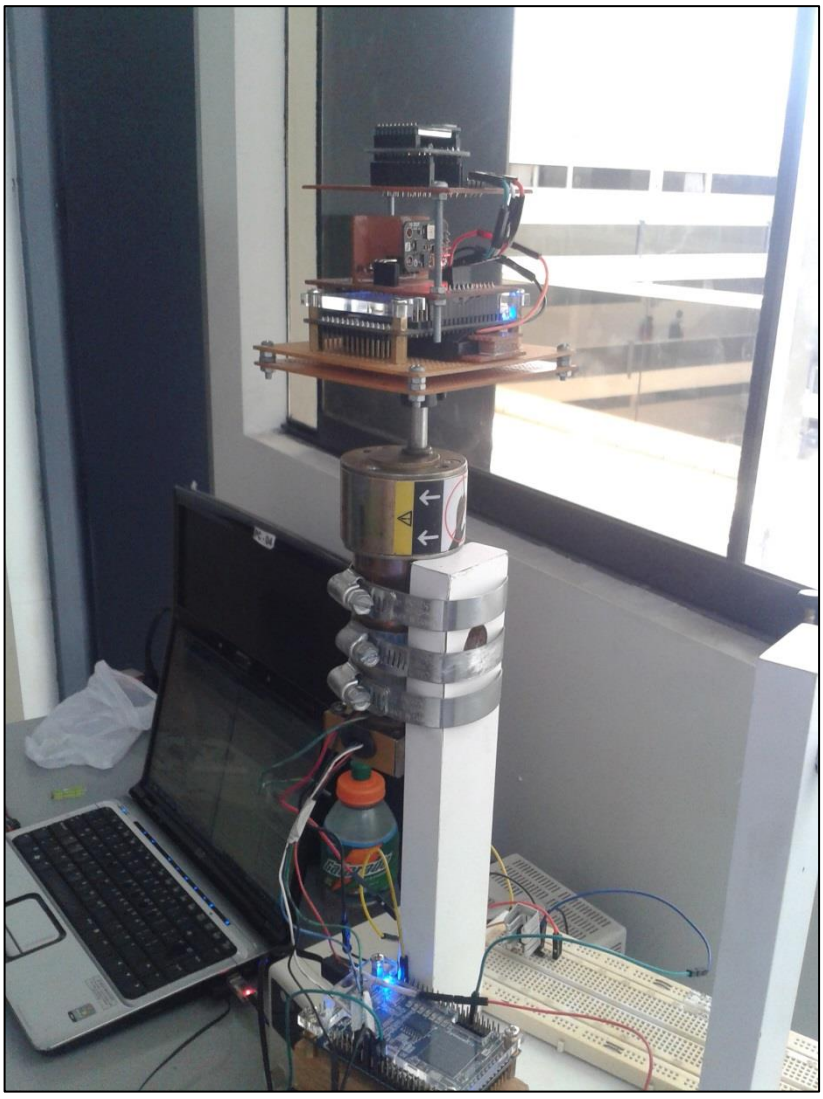

Figura 3.24: Foto de la implementación para encontrar los parámetros de calibración del giroscopio

Parámetros de calibración del Acelerómetro

Para encontrar los parámetros de calibración del acelerómetro se inclina en sus ejes y realiza las mediciones de los ángulos de inclinación con el sensor y mediante la plataforma. En este caso el sensor fue inclinado cada 10 grados, teniendo un total de 19 posiciones por cada eje positivo y negativo, en cada posición se tomaron 100 muestras. Para esta experiencia se utilizó un transportador acoplado a la parte superior de la plataforma de calibración como se muestra en la figura 3.27.

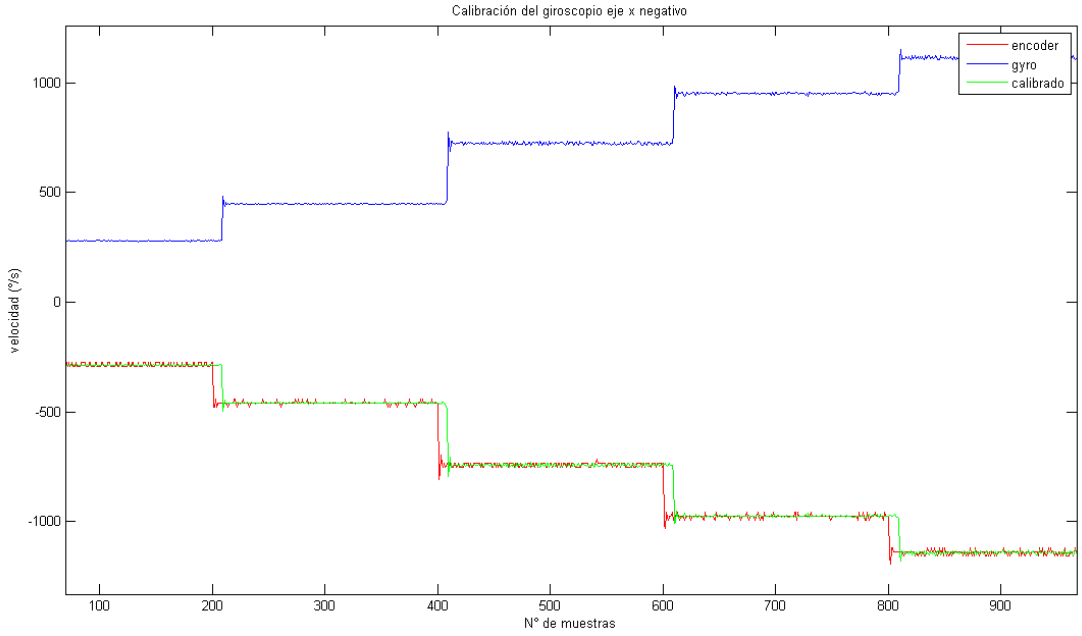

Figura 3.25: Calibración en el eje x negativo en coordenadas Body

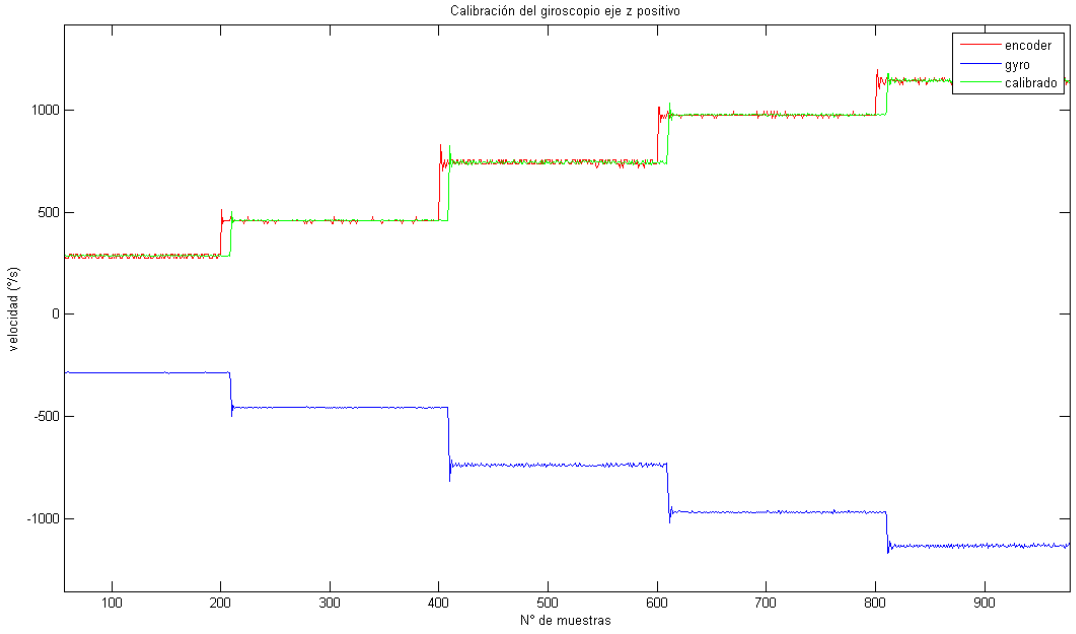

Figura 3.26: Calibración en el eje z positivo en coordenadas Body

Figura 3.27: Mecanismo para la obtención de los parámetros de calibración del acelerómetro

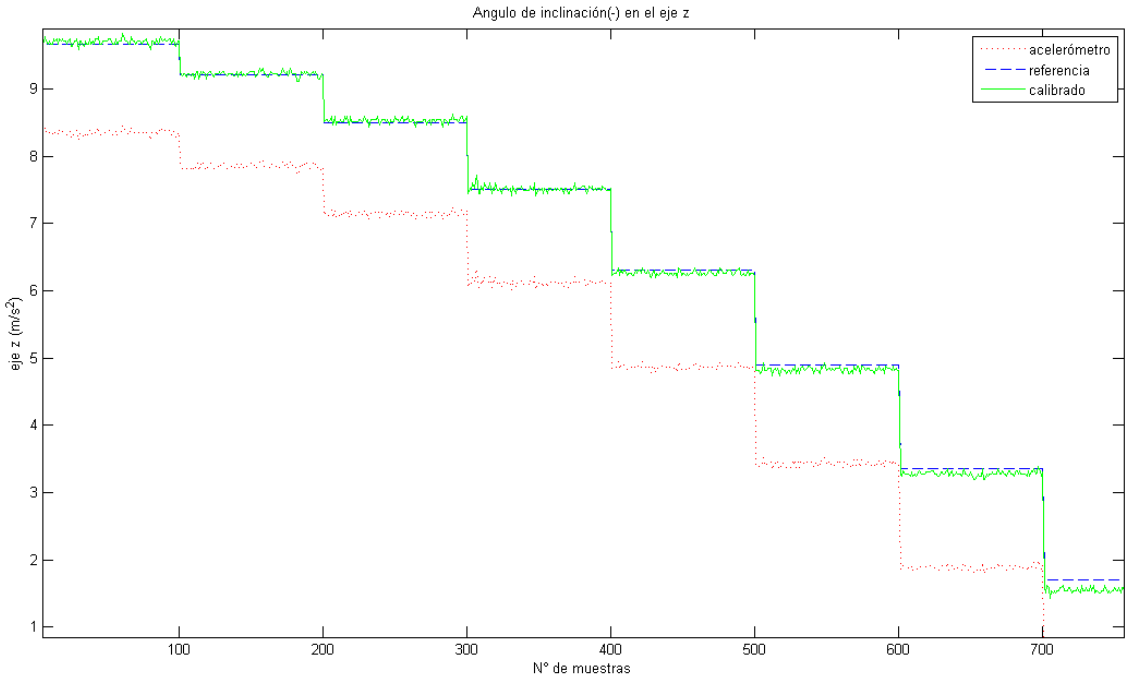

Figura 3.28: Calibración en el ángulo de inclinación negativo con la aceleración en el eje z

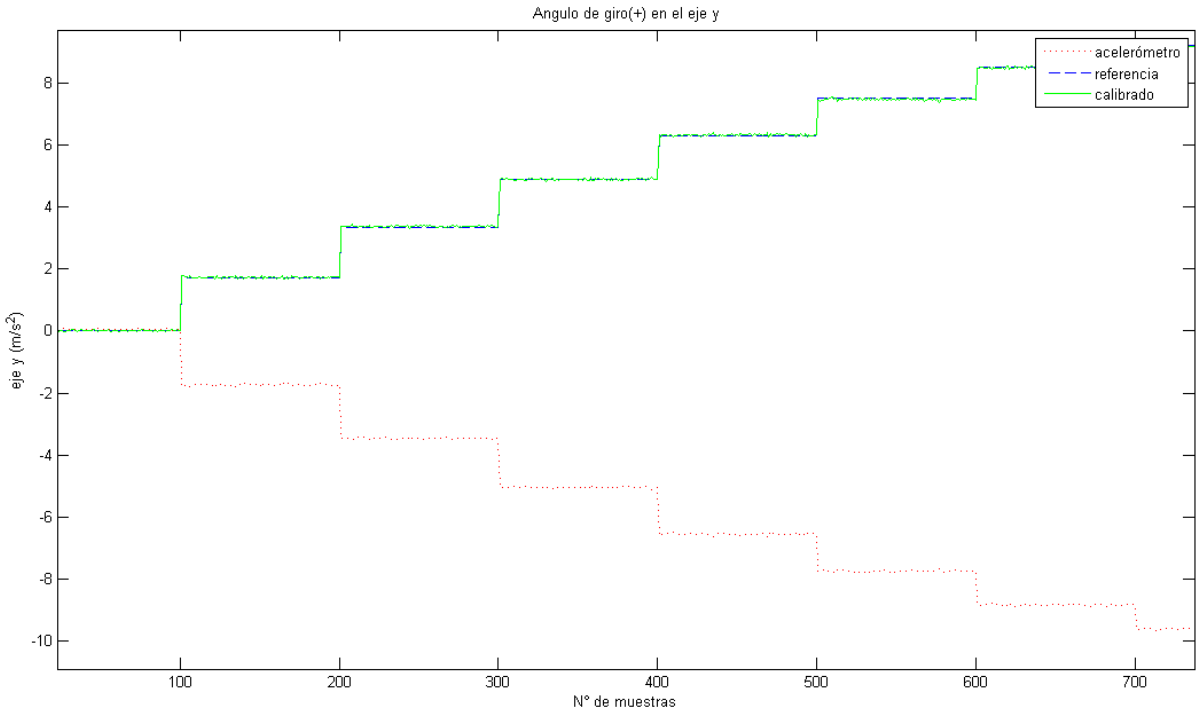

Figura 3.29: Calibración en el ángulo de giro positivo con la aceleración en el eje y

Los resultados del proceso de calibración, se muestran en las figura 3.28 y 3.29 en las cuales se comparan los datos de referencia (color azul), los datos del sensor (color rojo) y los datos calibrados (color verde) en los ángulos de inclinación negativo y de giro positivo, con la aceleración en el eje Z y eje Y respectivamente. De igual manera se calibraron en los demás ángulos y ejes.

Antes de calcular la matriz de calibración, los datos fueron normalizados con $g = 9.81\, m/s^2$ para procesar el algoritmo de mínimos cuadrados. La matriz de compensación obtenida es igual a:

$$X = \begin{bmatrix} 0.9859 & -0.0143 & 0.0112 \\ -0.0166 & -0.9608 & -0.0049 \\ -0.0023 & 0.0115 & 1.0066 \\ -0.0197 & -0.0038 & 0.1304 \end{bmatrix}$$

Con la calibración para las mediciones del acelerómetro se logró disminuir el error sin calibrar de 1.3739 m/s^2 a valores calibrados con un error de 0.0970 m/s^2, concluyendo que se logró disminuir el error del sensor satisfactoriamente.

Parámetros de calibración del Magnetómetro

El procedimiento de calibración consiste en hacerlo girar y simultáneamente inclinarlo con la finalidad de tomar muestras en la mayor cantidad de posiciones e inclinaciones. Para este proyecto se utilizó el código de Matlab (MathWorks, Inc., 2009) para graficar y calibrar este sensor. En este procedimiento se utilizó la maqueta completa como se muestra en la figura 3.30.

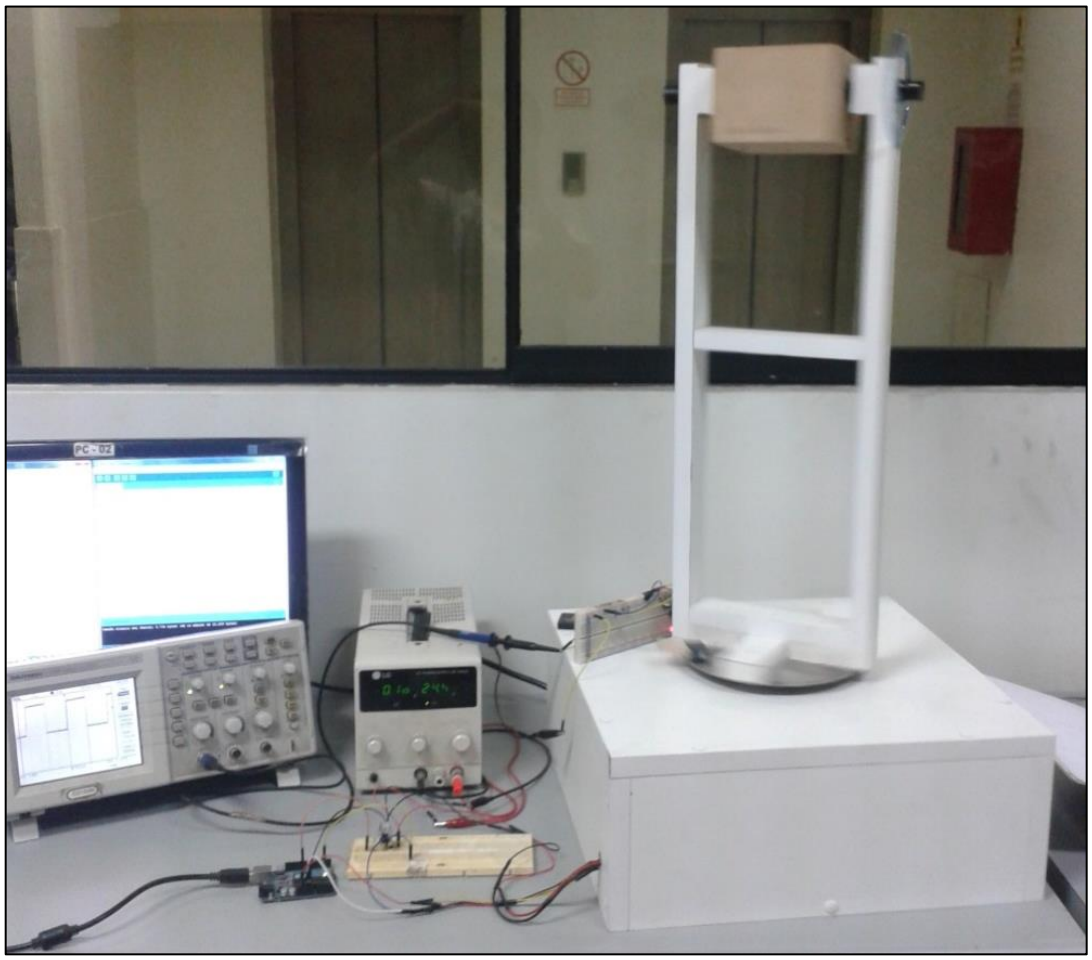

Figura 3.30: Implementación para determinar los parámetros de calibración del magnetómetro

En la figura 3.31 se muestran las comparaciones de los datos obtenidos del magnetómetro después de realizar las pruebas, graficados en tres dimensiones. De color rojo se muestran los datos sin calibrar del sensor, y de color azul los datos calibrados.

Idealmente si el sensor fuese perfecto y no existiesen interferencias magnéticas externas, con los datos del magnetómetro se debería graficar una esfera perfecta centrada en el origen de coordenadas, lo cual se logra mediante la calibración (color azul), sin embargo los datos del magnetómetro forman una figura ovalada y no centrada en el origen (color rojo).

Los datos del magnetómetro se normalizaron antes de calibrar tomando como valor del campo magnético en Lima-Perú igual a: $H = 0.25 gauss$.

Los valores de los resultados (doce parámetros de calibración) obtenidos son:

- Los errores respecto al centro (offset):

$$errorCentro_x = 0.0719749,$$

$$errorCentro_y = -0.412054,$$

$$errorCentro_z = -0.103984$$

- La matriz del factor de escala y del desalineamiento resultante es:

$$FactorEscalayDesalineamiento = \begin{bmatrix} 0.912879 & 0.00733372 & 0.0206488 \\ 0.00733372 & 0.921211 & 0.00647972 \\ 0.0206488 & 0.00647972 & 0.994242 \end{bmatrix}$$

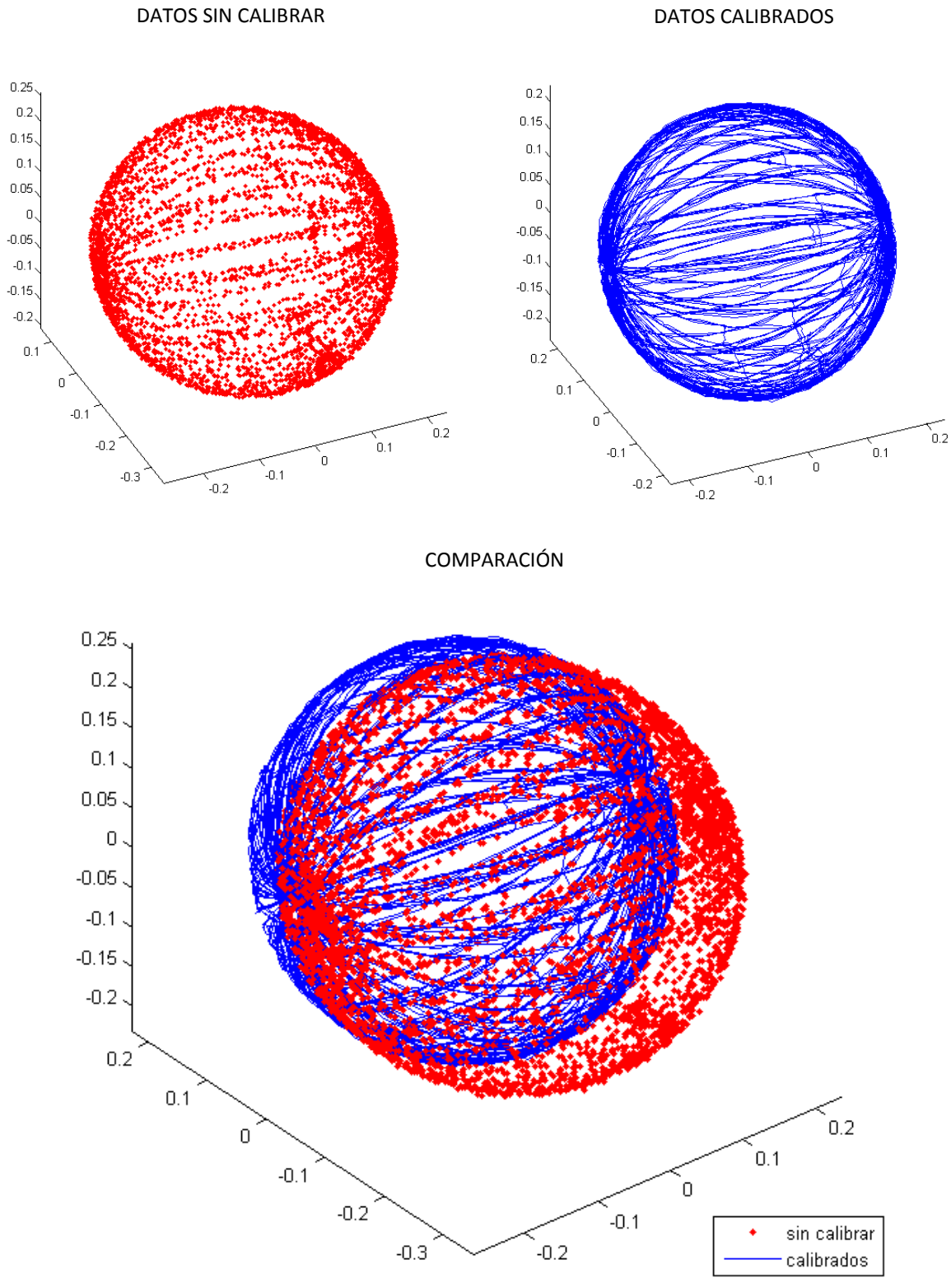

Figura 3.31: Comparación entre los datos del magnetómetro sin calibrar y calibrada

3.4.2 Implementación del sistema SOPC NIOS II

El procedimiento de implementación del sistema SOPC NIOS II se divide en dos, igual que en el diseño, y estos son:

- Implementación del hardware.
- Implementación del software.

Implementación del Hardware

La implementación del sistema SOPC NIOS II junto con el sistema SBA se realizó mediante el constructor SOPC Builder del software Quartus II. En la figura 3.32, se muestra el diagrama RTL[24] del sistema general una vez realizado el proceso de síntesis. En ella se muestran las conexiones entre el procesador Soft-Core NIOS II con el bloque Sistema SBA, además de los bloques auxiliares, los cuales son: el bloque controlador ADC[25], y los cuatro bloques PWM para el control de los motores brushless del Quadrotor. Estos bloques auxiliares fueron utilizados únicamente para la etapa de pruebas en la sintonización de filtro de Kalman Extendido y generación de disturbios, es por ello que no se consideran en el diagrama de bloques general del sistema a implementar (figura 3.1).

En la figura 3.33 se muestra un reporte del hardware generado por el sistema, por ejemplo se observa que se están usando 7434 elementos lógicos que representa el 33 % de la capacidad total del FPGA, generando de esta manera la posibilidad de añadir diseños futuros.

[24] RTL (Register Transfer Level) es un nivel de abstracción del sistema que representa el flujo de señales entre los registros hardware.
[25] ADC, de las siglas en inglés de Analog to Digital Converter, que se encarga de la transcripción de señales analógicas en señales digitales.

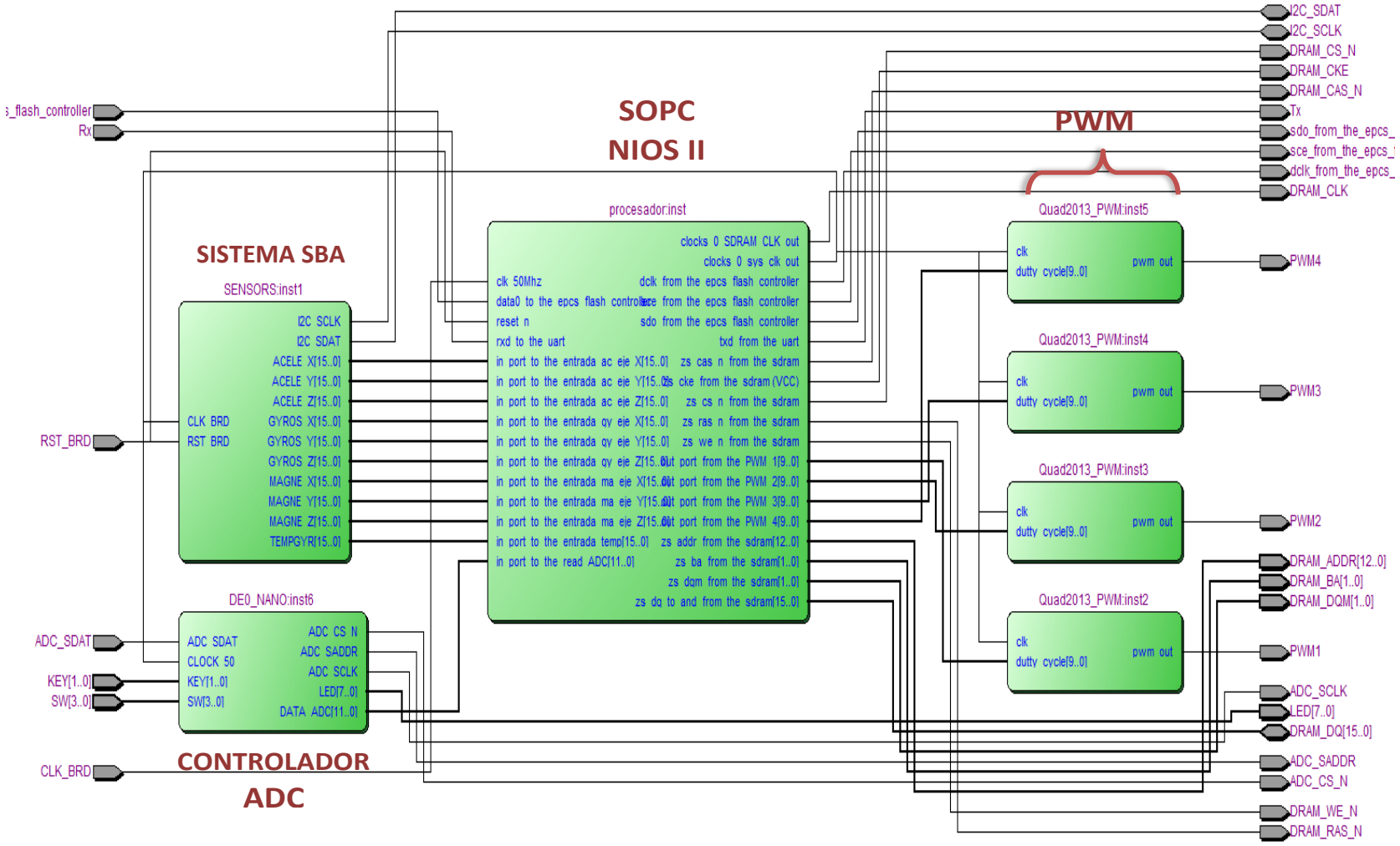

Figura 3.32: Diagrama RTL del Hardware desarrollado

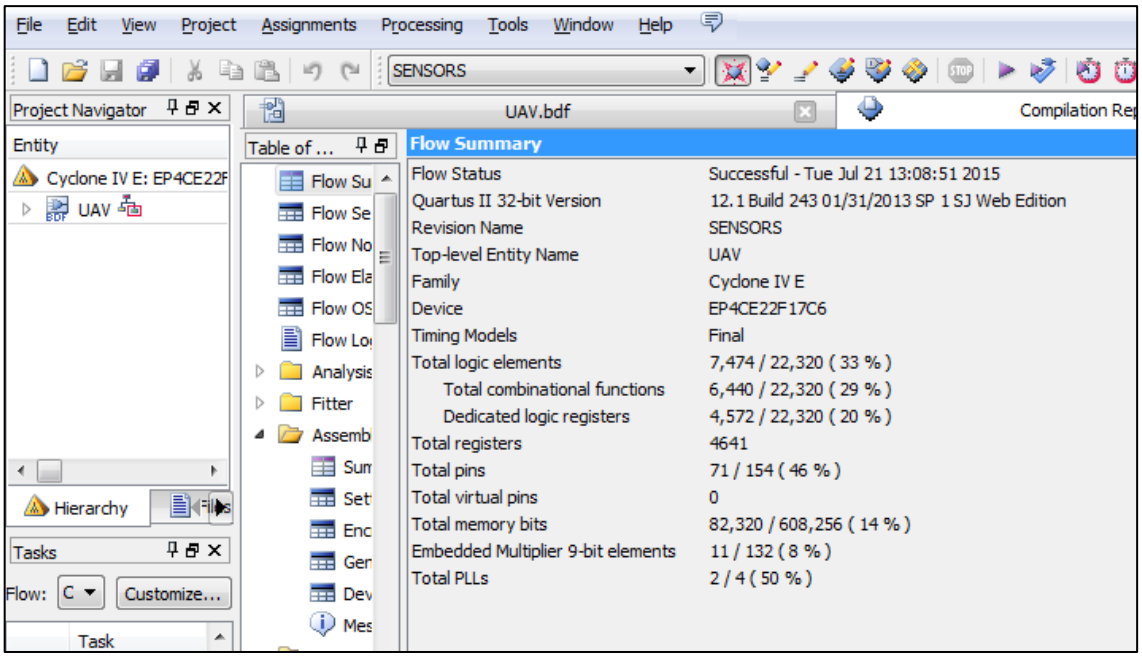

Figura 3.33: Reporte del Hardware implementado

Implementación en software

En el Procesador NIOS II se ejecuta el software a implementarse dentro de la FPGA DE0-Nano. Este software ejecuta principalmente dos algoritmos: calibración de sensores y el Filtro de Kalman Extendido. El lenguaje de programación utilizado es el C/C++.

El algoritmo de calibración se realiza implementado la ecuación (3.2) para cada sensor, con los parámetros calculados mediante la plataforma de calibración.

Con respecto al algoritmo de determinación de orientación (Filtro de Kalman Extendido), éste si requiere de una mayor cantidad de cálculos, por lo tanto una mayor capacidad de procesamiento.

Aplicación práctica

En esta aplicación se tiene como premisa que la matriz de covarianza del error de medición R_k es diagonal (Nonami, Kendoul, Suzuki, Wang, & Nakazawua, 2010).

Mediante esta aplicación se evita el cálculo de la matriz inversa para calcular la Ganancia de Kalman.

La matriz de covarianza R_k se considera que es una matriz simétrica de $m\ x\ m$. Luego se definen las matrices H_k, z_k y R_k como:

$$H = \begin{bmatrix} H(1) \\ \vdots \\ H(m) \end{bmatrix}, \qquad z = \begin{bmatrix} z(1) \\ \vdots \\ z(m) \end{bmatrix}, \qquad R = \begin{bmatrix} R(1) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & R(m) \end{bmatrix} \tag{3.45}$$

- **Paso (1): valor inicial**

Primero se considera $\bar{x} = \hat{x}_k^-$ y $\bar{P} = P_k^-$ que son el estado y el error de covarianza de la etapa de predicción del filtro de Kalman Extendido.

- **Paso (2): Etapa de corrección del filtro**

Igualando los valores de $\hat{x} = \bar{x}$ y $\hat{P} = \bar{P}$, se ejecutan las siguientes ecuaciones en un lazo para $i = 1\ hasta\ m$.

$$f = \hat{P}H(i)^T \tag{3.46}$$

$$\alpha = H(i)f + R(i) \tag{3.47}$$

$$K = f/\alpha \tag{3.48}$$

$$\hat{x} = \hat{x} + K(z(i) - h(i)) \tag{3.49}$$

$$\hat{P} = \hat{P} - Kf^T \tag{3.50}$$

- **Paso (3): Etapa de predicción del filtro**

Con los valores de $\hat{x}$ y $\hat{P}$ que representan el estado y el error de covarianza estimados, respectivamente; éstos se utilizan en las ecuaciones (2.56) y (2.57) de la etapa de predicción del filtro de Kalman Extendido.

Tiempo computacional

Para analizar el tiempo computacional de los algoritmos de fusión de sensores, éstos se implementaron en el procesador NIOS II, que opera a una frecuencia de reloj de 50MHz.

Los tiempos de ejecución de los algoritmos se midieron usando un osciloscopio. En la tabla 3.6, se comparan el computacional que se requiere ejecutar los algoritmos usando cofactores y la aplicación práctica.

Tabla 3.6: Comparación de los tiempos de ejecución

TIEMPO DE EJECUCIÓN (ms)		
MÉTODOS	**EKF**	**EKF y Matriz de Rotación**
Utilizando Cofactores	942.88	6.5
Aplicación práctica	**6**	**6.2**

Al Analizar el tiempo del Filtro de Kalman Extendido (EKF) en la Tabla 3.6 se observa que existe una diferencia significativa utilizando cofactores (942.88 ms) y la aplicación práctica (6.5 ms), se debe que en el primer caso, se debe resolver la inversa de una matriz de dimensión $[6x6]$, lo cual involucra una gran carga computacional,

mientras con la aplicación práctica se evita dicho cálculo. En el caso del EKF y Matriz de Rotación la diferencias de tiempos entre los dos métodos no es significativa (0.3 ms) debido que en este modelo sólo se resuelve la inversa de una matriz de $[3x3]$.

En general se observa que la aplicación del método práctico mejora los tiempos de ejecución de los dos algoritmos de determinación de orientación.

Para escoger un tiempo de muestreo adecuado se medió también el tiempo de ejecución del algoritmo de calibración y adquisición de datos de los sensores, cuyos resultados se muestran en las tablas 3.7 y 3.8.

El tiempo reducido de la tabla 3.8 en la adquisición de datos de los sensores (acelerómetro, giroscopio y magnetómetro) se debe a que se está utilizando el sistema SBA para la lectura de sensores, el cual ejecuta esta tarea de manera paralela a los algoritmos del procesador SOPC NIOS II.

Tabla 3.7: Tiempo de ejecución del algoritmo de calibración

TIEMPO DE EJECUCIÓN	
Algoritmo de calibración	0.089 ms

Tabla 3.8: Tiempo de ejecución de la adquisición de datos de los sensores

TIEMPO DE EJECUCIÓN	
Adquisición de datos de los sensores	0.6 μs

Para obtener un periodo de muestreo adecuado y teniendo en cuenta la dinámica rápida del Quadrotor, se concluye:

$$Periodo\ de\ muestreo > adquisición\ de\ datos + \ calibración + EKF$$

$$Periodo\ de\ muestreo > 0.6\ \mu s + \ 0.089\ ms + 6.2\ ms$$

$$Periodo\ de\ muestreo > 6.2896\ ms$$

Por lo tanto, se consideró adecuado un periodo de muestreo de:

$$Periodo\ de\ muestreo\ (Ts) = 10ms$$

CAPÍTULO 4

ANÁLISIS DE RESULTADOS

En este capítulo se muestran los resultados obtenidos tras las pruebas experimentales realizadas al sistema de determinación de orientación. Para ello se diseñó y construyó una plataforma de pruebas que permitió generar condiciones semejantes a un vuelo real.

Como resultados se obtuvieron los datos de la varianza de ruido de los sensores y la elección del algoritmo de determinación de orientación con mejor rendimiento entre los tres métodos desarrollados sobre un dispositivo lógico programable (FPGA).

4.1 Banco de pruebas

En la figura 4.1 se muestra la fotografía del Quadrotor en el banco de pruebas construido. Mediante esta plataforma se puede hacer girar el Quadrotor sobre un eje, con la posibilidad de generar de manera simultánea disturbios, como pueden ser: las vibraciones causadas por los motores brushless, aceleraciones lineales causadas por

movimientos bruscos, o interferencias magnéticas; simulando de esta manera un ambiente semejante a un vuelo real del Quadrotor.

El objetivo principal de la plataforma es comparar los datos obtenidos del sistema de determinación de orientación, con los datos obtenidos a través del banco de pruebas los cuales se consideran referenciales. Para obtener la actitud u orientación referencial se utilizó un potenciómetro lineal acoplado a un extremo del eje de la plataforma. Los datos analógicos del potenciómetro fueron capturados utilizando el ADC de la DE0-Nano.

Figura 4.1: Banco de pruebas del Quadrotor

4.2 Varianza de ruido de los sensores

Antes de analizar el rendimiento del sistema fue necesario conocer la varianza de ruido de los sensores, los cuales fueron introducidos en los algoritmos de determinación de orientación.

Para determinar los valores de la matriz de covarianza de ruido del sistema (Q_k) y de las mediciones (R_k), que modelan los ruidos en los sensores, se realizaron pruebas con los motores desactivados y con diferentes valores de potencia. Estas mediciones idealmente deberían tener como media cero y una distribución normal.

4.2.1 Varianza con los motores desactivados

En las tablas 4.1, 4.2 y 4.3 se muestran la varianza, desviación estándar y media de las mediciones del acelerómetro, giroscopio y magnetómetro respectivamente. Las pruebas se realizaron con los motores desactivados del Quadrotor e inmóvil sobre la línea horizontal. Los datos se obtuvieron durante 20 segundos a una velocidad de muestreo de 100 muestras/segundo con un total de 2000 muestras por prueba.

Tabla 4.1: Varianza, desviación estándar y media de las mediciones del acelerómetro y motores desactivados

ACELERÓMETRO			
t = 20s 100 muestras/seg	Motores desactivados		
	var	std	mean
eje x (g)	**7.43E-06**	0.0027	6.53E-04
eje y (g)	**5.73E-06**	0.0024	9.50E-03
eje z (g)	**1.35E-05**	0.0037	1.01E+00

Tabla 4.2: Varianza, desviación estándar y media de las mediciones del giroscopio y motores desactivados

GIROSCOPIO			
t = 20s 100 muestras/seg	Motores desactivados		
	var	std	mean
eje x (°/s)	**0.0016**	0.0404	1.9925
eje y (°/s)	**0.0019**	0.0433	1.3454
eje z (°/s)	**0.0015**	0.0389	-1.6903

Tabla 4.3: Varianza, desviación estándar y media de las mediciones del magnetómetro y motores desactivados

MAGNETÓMETRO			
t = 20s 100 muestras/seg	Motores desactivados		
	var	std	mean
eje x (H)	**5.89E-05**	0.0077	8.48E-02
eje y (H)	**2.53E-05**	0.005	5.83E-01
eje z (H)	**5.66E-05**	0.0075	2.96E-02

Las mediciones del acelerómetro fueron normalizados con $g = 9.81 m/s^2$. Las mediciones del magnetómetro fueron normalizados con $H = 0.25 gauss$ que es el valor del campo magnético en Lima – Perú.

Como se observa en las tablas 4.1 - 4.3, los valores de las varianzas son pequeños para los tres sensores (acelerómetro, giroscopio y magnetómetro) con la plataforma en reposo y los motores desactivados, esto es indicativo del poco nivel de

ruido introducido en el sistema en estas condiciones y resultando favorable para obtener estimaciones precisas.

4.2.2 Varianza con los motores prendidos

Estas pruebas se realizaron con los motores activados y puestas sus hélices, pero con el Quadrotor sobre la línea horizontal. La velocidad de los motores fue controlada con señal PWM con diferentes valores de potencias.

Las pruebas se realizaron primero calcular hallar las varianzas de ángulos del filtro de Kalman lineal (FK). También se realizaron pruebas fueron para calcular las varianzas de las mediciones de los sensores, cuyos resultados se utilizaron en los dos filtros de Kalman Extendido.

a) Varianza para el filtro de Kalman lineal

En este caso sólo se necesitaron los valores de las varianzas de la velocidad angular obtenidos del giroscopio y del ángulo de inclinación obtenidos de las mediciones del acelerómetro. En las tablas 4.4 y 4.5 se muestran los resultados de las pruebas para el ángulo de giro. Los resultados que se utilizaron en el algoritmo del filtro están resaltados en negrita. La modulación PWM tiene una frecuencia de 300 Hz y programado en VHDL con una resolución de 10 bits (0-1023).

Tabla 4.4: Varianza, desviación estándar y media de los ángulos (°) obtenidos del acelerómetro y los motores activados

ACELERÓMETRO			
t = 20s 100 muestras/seg	Varianza (σ^2)	Desviación estándar (σ)	Media
Potencia motor 45 %	399.9244	19.9981	0.7627
Potencia motor 50 %	**556.6971**	**23.5944**	**-1.6787**
Potencia motor 55 %	425.4754	20.6271	2.4023

Tabla 4.5: Varianza, desviación estándar y media de las velocidades angulares (°/s) obtenidos del giroscopio y los motores activados

GIROSCOPIO			
t = 20s 100 muestras/segundo	Varianza (σ^2)	Desviación estándar (σ)	Media
Potencia motor 45 %	0.0402	0.2005	2.2796
Potencia motor 50 %	0.0895	0.2991	2.2982
Potencia motor 55 %	**0.1314**	**0.3624**	**2.3653**

De los resultados mostrados de las tablas 4.4 y 4.5 se observa que para el acelerómetro se produce una mayor varianza cuando los motores funcionan a 50 % de su potencia máxima y que a mayores potencias se reduce dicha varianza. En el caso del giroscopio, la varianza alcanza su valor máximo a una potencia de 55% de la potencia máxima del motor.

Con los valores de la tabla 4.4, la covarianza de ruido de la medición (R_k) es:

$$R_k = \sigma^2 = (23.5944)^2 = 556.6971$$

La matriz de covarianza de ruido del sistema se calcula mediante la siguiente expresión:

$$Q_k = T^2 \begin{bmatrix} {\sigma_\omega}^2 & 0 \\ 0 & 1 \end{bmatrix} = 0.01^2 \begin{bmatrix} 0.3624^2 & 0 \\ 0 & 1.1 \end{bmatrix}$$

Donde, σ_ω es el mayor valor de desviación estándar del giroscopio de la tabla 4.5. El valor de 1.1 se determinó mediante prueba y ajuste, obteniendo con este valor el mejor resultado del filtro.

b) Varianza para el filtro de Kalman Extendido

Para calcular las matrices de covarianza de ruido de los filtros de Kalman Extendido desarrollados se realizaron mediciones directas de los sensores en pruebas con los motores activados y a diferentes potencias.

En las Tablas 4.6, 4.7 y 4.8 se muestran los resultados obtenidos con los valores de la varianza, media y desviación estándar de las mediciones tanto para el acelerómetro, giroscopio y el magnetómetro. Las varianzas a considerar en los EKF son los de mayores valores, los cuales son resaltados en negrita. Los datos del acelerómetro y magnetómetro se normalizaron con $g = 9.81 m/s^2$ y $H = 0.25 gauss$. Las gráficas de los resultados de las tablas 4.6 – 4.8 se muestran en el anexo C.

En las Tablas 4.6 – 4.8 se observa que las varianzas obtenidas son mucho mayores que cuando los motores estuvieron desactivados, esto evidencia el gran nivel de perturbación que se genera a causa de las vibraciones en la estructura.

Tabla 4.6: Varianza, desviación estándar y media de las mediciones del acelerómetro con los motores activados

t = 20s 100 muestras/seg	**ACELERÓMETRO**								
	PWM 450			PWM 500			PWM 550		
	var	std	mean	var	std	mean	var	std	mean
eje x (g)	0.0533	0.2308	0.0053	0.0109	0.1045	-0.005	**0.1432**	0.3784	2.2E-4
eje y (g)	0.1539	0.3923	0.0091	**0.1765**	0.4202	0.0128	0.0776	0.2786	0.0117
eje z (g)	0.0147	0.1211	1.0078	0.1849	0.43	1.01	**0.221**	0.4701	1.0232

Tabla 4.7: Varianza, desviación estándar y media de las mediciones del giroscopio con los motores activados

t = 20s 100 muestras/seg	**GIROSCOPIO**								
	PWM 450			PWM 500			PWM 550		
	var	std	mean	var	std	mean	var	std	mean
eje x (°/s)	**0.0402**	0.2005	2.2796	**0.0895**	0.2991	2.2982	**0.1314**	0.3624	2.3653
eje y (°/s)	**0.0054**	0.0734	1.5461	**0.016**	0.1267	1.5927	**0.005**	0.0704	1.6823
eje z (°/s)	**0.0179**	0.1337	-2.071	**0.0308**	0.1755	-2.125	**0.0276**	0.1661	-2.226

Tabla 4.8: Varianza, desviación estándar y media de las mediciones del magnetómetro con los motores activados

t = 20s 100 muestras/seg	**MAGNETÓMETRO**								
	PWM 450			PWM 500			PWM 550		
	var	std	mean	var	std	mean	var	std	mean
eje x (H)	7.9E-05	0.0089	0.0722	9.1E-5	0.0095	0.0683	**1.2E-4**	0.0109	0.0608
eje y (H)	1.3E-04	0.0113	0.5983	1.4E-4	0.0118	0.6079	**1.7E-4**	0.013	0.6281
eje z (H)	9.6E-05	0.0098	0.0223	8.3E-5	0.0091	0.0245	**1E-04**	0.01	0.0199

Se puede decir también que el aumento en la varianza de las mediciones del acelerómetro (>1g) es mucho mayor con respecto al del giroscopio (<1°/s) y del magnetómetro (<0.1mH), concluyendo que las vibraciones tienen un mayor efecto en el acelerómetro que en los otros sensores.

Con estos datos se pueden determinar directamente los valores de la matriz de covarianza de ruido de la medición R_k y mediante un ajuste de valores la matriz de ruido del proceso Q_k de los dos filtros de Kalman Extendido.

4.2.3 Matrices de covarianza de ruido del sistema y de la medición

Para sintonizar los Filtros de Kalman Extendido desarrollados, fue necesario establecer los valores de Q_k y R_k que son las matrices de covarianza de ruido del proceso y de la medición, respectivamente. Los valores de R_k se obtienen directamente de las varianzas del acelerómetro y magnetómetro de las tablas 4.6 y 4.8. Para esto, se eligen las varianzas de mayor valor en cada eje. Entonces la matriz de covarianza de ruido de la medición contiene los siguientes valores:

$$R = \begin{bmatrix} 0.1432 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.1765 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.221 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.2e-4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.7e-4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1e-4 \end{bmatrix}$$

El valor de la matriz Q_k se realizó introduciento cierta incertidumbre en el proceso para realizar un ajuste manual mediante prueba y error. Los valores obtenidos fueron:

$$Q = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2e-7 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2e-7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2e-7 \end{bmatrix}$$

4.3 Comparación entre los algoritmos de determinación de orientación

En esta etapa se utilizó el banco de pruebas (figura 4.1), primero en condiciones estáticas, sin movimiento y con los motores desactivados del Quadrotor. En estas condiciones se sintonizaron los valores de la matriz de covarianza de ruido del proceso y de la medición de los filtros de Kalman, además de medir también su exactitud y precisión. Posteriormente, se realizaron pruebas con los motores encendidos generando perturbaciones como: vibraciones, interferencias magnéticas y aceleraciones lineales.

Las pruebas se realizaron solamente en los ángulos de giro e inclinación, mas no en el ángulo de desviación debido a que no se consideró fundamental analizar a detalle su exactitud y precisión, ya que este primer trabajo se encuentra enfocado en el control de estabilidad del Quadrotor, y aún no de traslación. Además esto involucraría construir una plataforma de pruebas más compleja el cual se considera como trabajo futuro.

Para analizar la exactitud y precisión de los algoritmos de fusión de sensores se utilizaron la media, varianza y el error RMS como índices de rendimiento de los filtros (Pierre & Denis, 2004) los cuales han sido detallados en la sección 2.2.7. El índice de error RMS de las estimaciones fue el que se consideró con mayor prioridad.

4.3.1 Pruebas en condiciones Estáticas

En estas pruebas se mantiene en reposo y sobre la horizontal el Quadrotor con los motores desactivados. De manera simultánea se obtienen datos de los ángulos de giro, inclinación y desviación del sistema de determinación, utilizando primero el Filtro de Kalman Lineal (FK), y luego los dos Filtros de Kalman Extendido.

En las Tablas 4.9 y 4.10, se muestran los resultados de los índices de rendimiento (error RMS, media y varianza) de las pruebas realizadas. En la Tabla 4.9 se muestran los valores de los ángulos obtenidos en el eje para el ángulo de giro, y en la tabla 4.10 del eje para el ángulo de inclinación. Los mejores resultados se resaltan en negrita.

Se observa que el error RMS, que es un indicativo de la diferencia existente entre las estimaciones del sistema con el valor verdadero o referencial, tiene un menor valor aplicando el algoritmo Filtro de Kalman Extendido y una Matriz de rotación, tanto para movimientos en el ángulo de giro (0.5496°), como para el ángulo de inclinación (0.4965°), en comparación con los otros dos filtros, los cuales son: el Filtro de Kalman lineal (FK) y el Filtro de Kalman Extendido (EKF).

Tabla 4.9: Índices de rendimiento de los algoritmos de fusión de sensores en condiciones estáticas y sin perturbaciones para el ángulo de giro

CONDICIONES ESTÁTICAS	Algoritmos de Determinación de actitud ÁNGULO DE GIRO		
	FK	EKF	EKF y Matriz de Rotación
Error RMS $[grados]$	0.5946	1.6458	**0.5496**
Media del error $[grados]$	0.5939	1.6049	**0.5401**
Varianza del error $[grados^2]$	**8.7032e-04**	0.1328	0.0104

Tabla 4.10: Índices de rendimiento de los algoritmos de fusión de sensores en condiciones estáticas y sin perturbaciones para el ángulo de inclinación

CONDICIONES ESTÁTICAS	Algoritmos de Determinación de actitud ÁNGULO DE INCLINACIÓN		
	FK	EKF	EKF y Matriz de Rotación
Error RMS $[grados]$	0.5882	1.9671	**0.4965**
Media del error $[grados]$	0.5820	1.9155	**0.4917**
Varianza del error $[grados^2]$	0.0072	0.2004	**0.0047**

Como el Quadrotor en estas pruebas se encuentra sobre la línea horizontal, se obtendrán mejores resultados cuando la media de los ángulos esté cercana a cero.

Entonces, analizando el valor de la media en las tablas 4.9 y 4.10, se observa que también el EKF con una Matriz de Rotación proporciona los mejor resultados obteniendo los menores valores con 0.5401° en el ángulo de giro y 0.4917° en el ángulo de inclinación.

Por último, analizando el valor de la varianza, en la cual una mayor varianza representa una menor precisión en las estimaciones, se obtuvo un mejor resultado con el EKF y Matriz de Rotación (0.0047°) en el ángulo de inclinación, pero en el ángulo de giro el Filtro de Kalman Lineal (FK) es el que generó la menor varianza (8.7076e-04°). Esto se debe a que las estimaciones del Filtro de Kalman lineal (FK) inician del primer valor del ángulo obtenido por el acelerómetro, en cambio el segundo EKF obtiene el primer valor desde una estimación inicial de ángulo cero, como se puede observar en la figura 4.2. Sin embargo, esta diferencia es pequeña (< 0.01°) con respecto a los otros dos filtros.

En las figuras 4.2 y 4.3 se aprecia las gráficas de los resultados obtenidos de las pruebas realizadas. Se observa que las estimaciones del EKF y Matriz de Rotación junto con el Filtro de Kalman lineal (FK) generan señales con un seguimiento más preciso a la señal considerada verdadera o referencial. También se observa que tanto para ángulo de giro como para el ángulo de inclinación, la señal del EKF presenta el mayor rizado en su respuesta, indicativo de un menor rendimiento. Es decir que se corrobora de manera visual los resultados de las tablas 4.9 y 4.10.

Como conclusión del análisis realizado a los resultados obtenidos en condiciones estáticas y con los motores desactivados, se tiene que el sistema de determinación de orientación utilizando el Filtro de Kalman Extendido y una Matriz de Rotación es el de mejores resultados. A pesar del buen rendimiento del EKF y Matriz de Rotación en condiciones estáticas, se observa que el nivel de exactitud y la precisión son cercanos a la respuesta de los otros dos filtros, por lo que fue necesario analizar su comportamiento

en condiciones dinámicas y con perturbaciones (motores activados), para realizar una mejor elección.

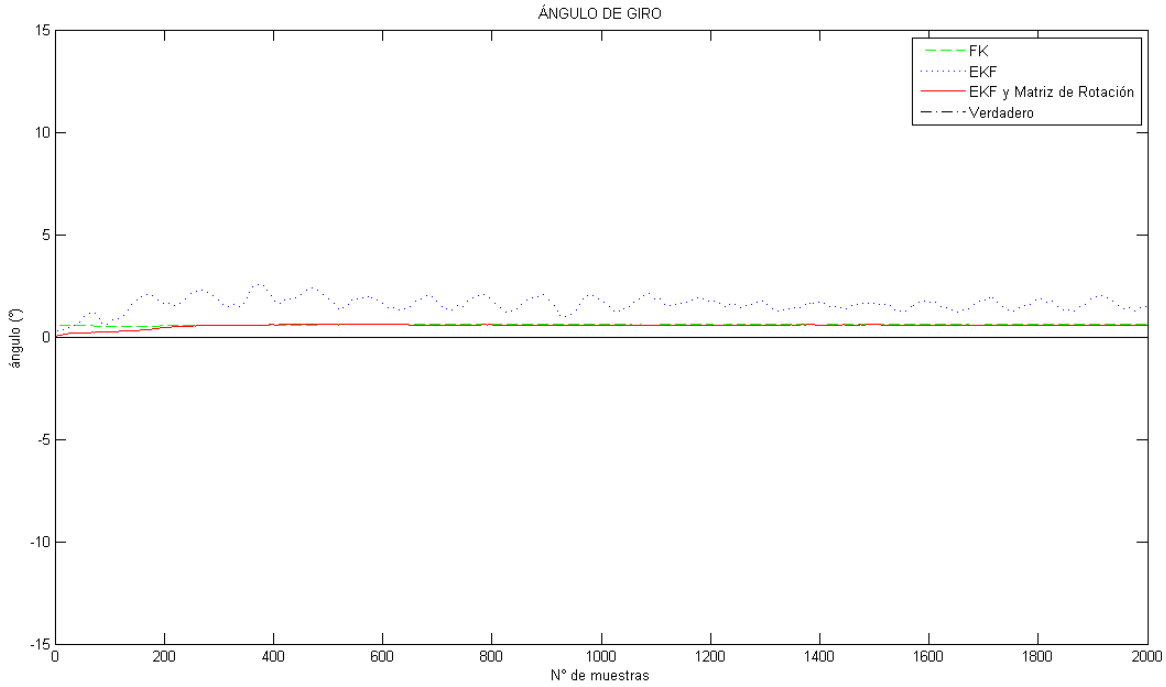

Figura 4.2: Ángulo de giro obtenido de los filtros, en condiciones estáticas

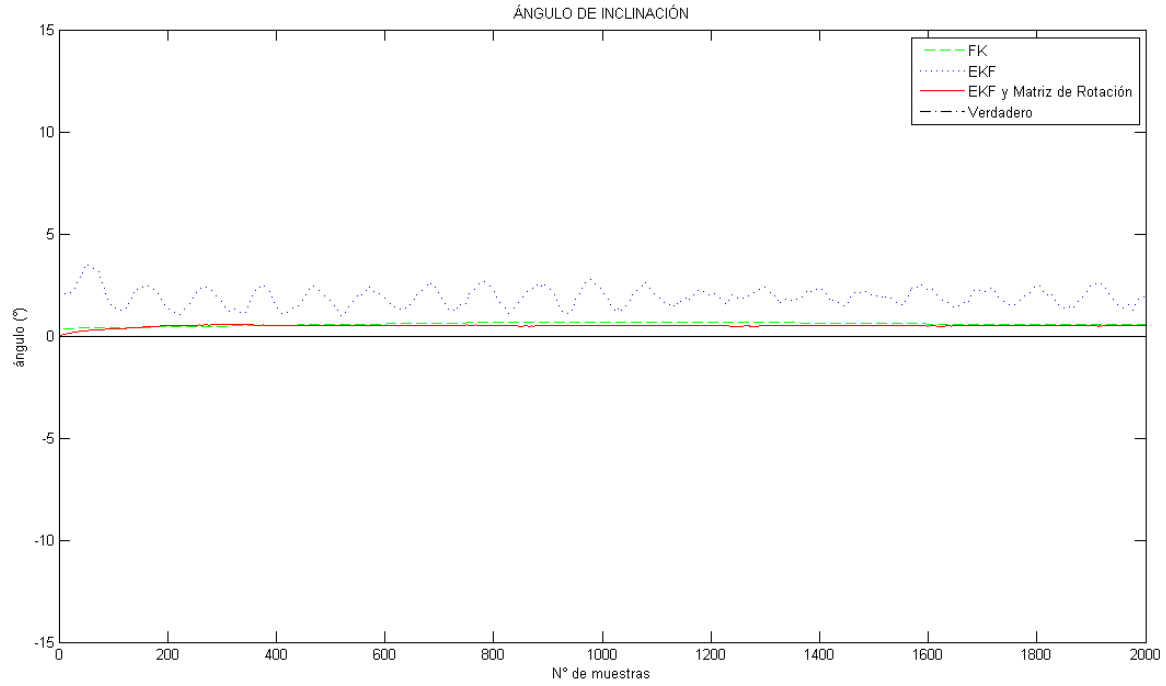

Figura 4.3: Ángulo de inclinación obtenido de los filtros, en condiciones estáticas

4.3.2 Pruebas en condiciones con perturbaciones

Estas pruebas se realizaron con los motores activados del Quadrotor y con hélices instaladas, con el objetivo de analizar el efecto de las perturbaciones generadas en el sistema de determinación de orientación o actitud.

En las tablas 4.11 y 4.12 se muestran los resultados del error RMS, media y varianza de los datos obtenidos en los ángulos de giro e inclinación, respectivamente. Se resaltaron en negrita los mejores resultados. En las figuras 4.4 y 4.5 se muestran la comparación gráfica de las mismas pruebas.

De los resultados mostrados en la tabla 4.11 se observa que la respuesta del Filtro de Kalman Extendido y Matriz de Rotación, presenta un menor error RMS en las estimaciones del ángulo de giro con un valor de 1.8447° a diferencia del filtro de Kalman Lineal (FK) y del primer Filtro de Kalman Extendido (EKF) con un error de 2.16° y 6.6966° respectivamente. Analizando la media de las estimaciones el mejor resultado se obtiene del mismo Filtro aunque la diferencia con la respuesta del filtro de Kalman lineal (FK) no es significativa. Con respecto a la varianza, el EKF y Matriz de Rotación presenta nuevamente una mejor precisión y el peor performance lo presenta la respuesta del EKF.

De los resultados mostrados en la tabla 4.12, se observa que el segundo EKF (EKF y Matriz de Rotación) tiene un menor error RMS con un valor de 4.5874°. Con respecto a la media y varianza también presenta una mayor precisión y exactitud con respecto a los otros dos filtros.

En las figuras 4.4 y 4.5, se pueden observar las estimaciones de ángulos de los tres algoritmos de manera gráfica.

Tabla 4.11: Índices de rendimiento de los algoritmos de fusión de sensores frente a perturbaciones para el ángulo de giro

CONDICIONES CON PERTURBACIONES	Algoritmos de Determinación de actitud ÁNGULO DE GIRO		
	FK	EKF	EKF y Matriz de Rotación
Error RMS $[grados]$	2.1600	6.6966	**1.8447**
Media del error $[grados]$	-0.9990	4.5567	**-0.2562**
Varianza del error $[grados^2]$	3.6675	24.0806	**3.3371**

Tabla 4.12: Índices de rendimiento de los algoritmos de fusión de sensores frente a perturbaciones para el ángulo de inclinación

CONDICIONES CON PERTURBACIONES	Algoritmos de Determinación de actitud ÁNGULO DE INCLINACIÓN		
	FK	EKF	EKF y Matriz de Rotación
Error RMS $[grados]$	5.9968	8.4533	**4.5874**
Media del error $[grados]$	4.4813	1.0679	**-0.2562**
Varianza del error $[grados^2]$	15.8794	70.3173	**3.3371**

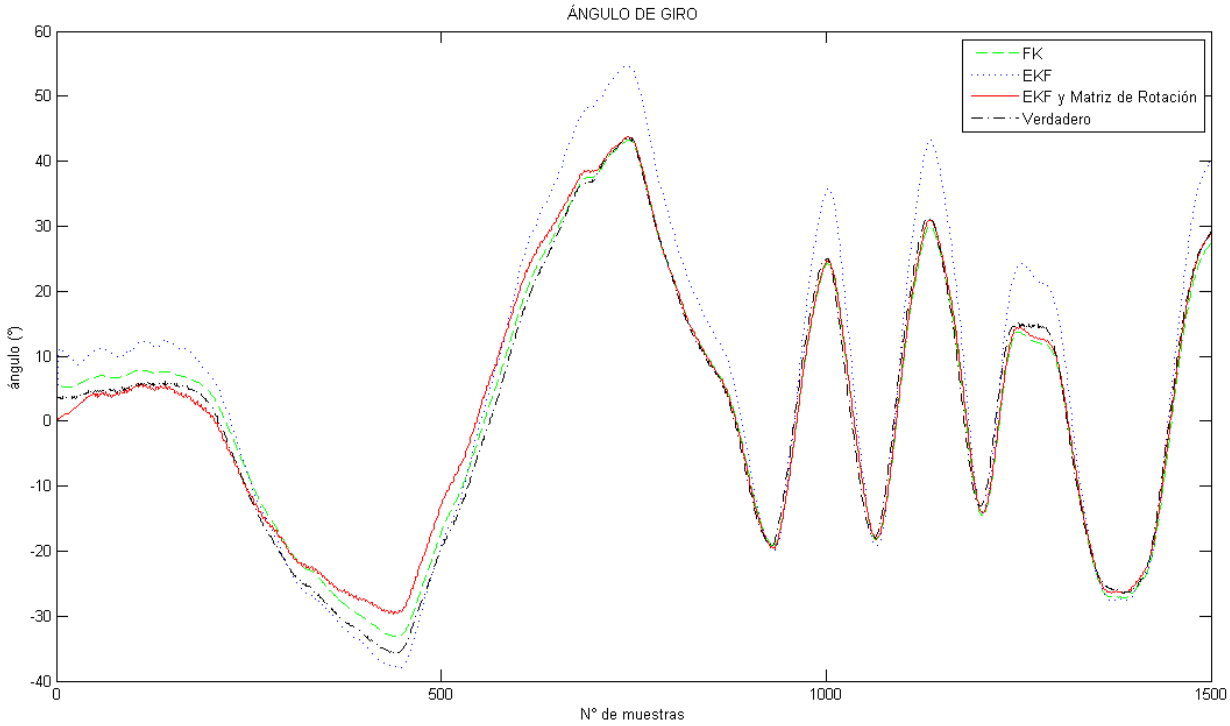

Figura 4.4: Comparación entre el FK (color verde), EKF (color azul), EKF y Matriz de Rotación (color rojo) además de la referencia (color negro) para el ángulo de giro, frente a perturbaciones

Analizando estos resultados se observa que en la figura 4.4 la respuesta del EKF es la señal de peor seguimiento con respecto a la considerada verdadera, en contraparte, tanto el segundo EKF (EKF y Matriz de Rotación) y el filtro de Kalman lineal (FK) realizan un buen seguimiento, aunque difícil de diferenciarlos visualmente, es por ello que se representa mediante valores numéricos que se visualizan en la tabla 4.11. En la figura 4.5 las respuestas del FK y EKF con matriz de Rotación proporcionan los mejores resultados en precisión y exactitud.

Se concluye entonces que el sistema de determinación de orientación utilizando el Filtro de Kalman Extendido y una Matriz de Rotación es el que presenta los mejores resultados, por lo que se obtiene el mejor rendimiento en condiciones con perturbaciones como vibraciones e interferencias magnéticas.

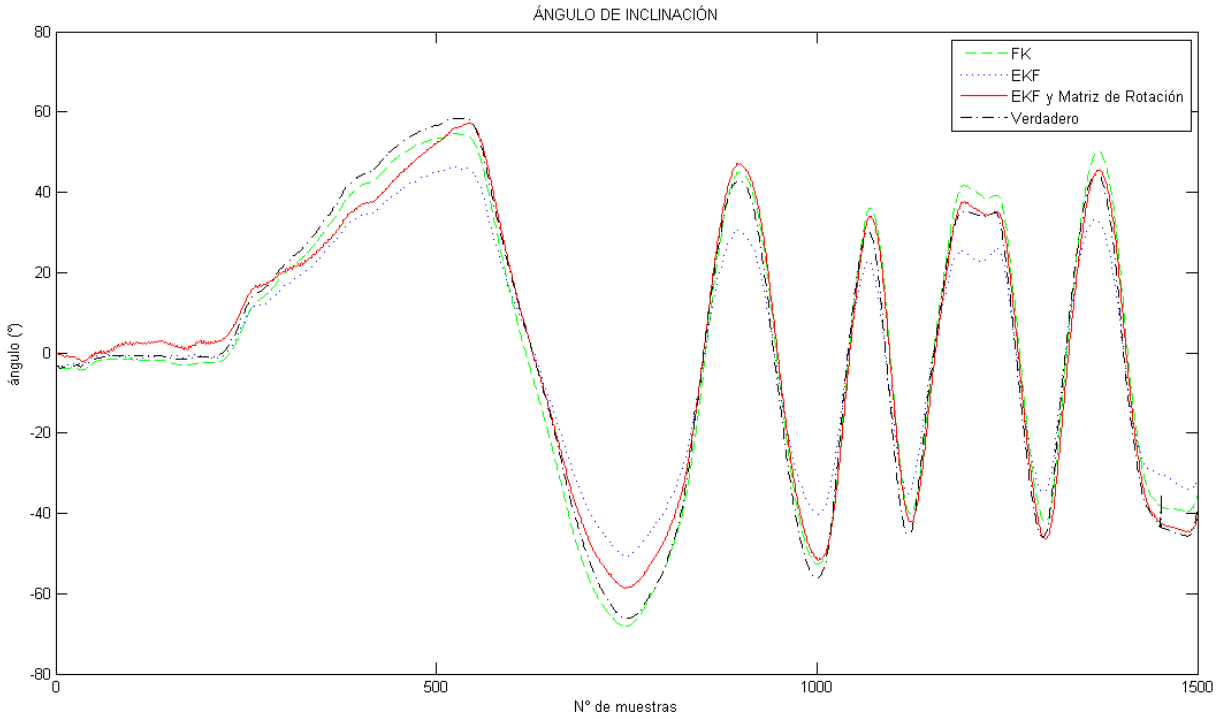

Figura 4.5: Comparación entre el FK (color verde), EKF (color azul), EKF y Matriz de Rotación (color rojo) además de la referencia (color negro) para el ángulo de inclinación, frente a perturbaciones

Se observa también que el EKF y Matriz de Rotación tiene una mayor precisión en el ángulo de giro con un error RMS de 1.8447° a diferencia del ángulo de inclinación con un error de 4.5874°, pero se consideran ambos resultados favorables para un óptimo control del Quadrotor.

Analizando la razón del mejor rendimiento del Filtro de Kalman Extendido y Matriz de Rotación. Se determinó que a pesar de los buenos resultados del Filtro de Kalman lineal (FK), como se muestran en las figuras 4.4 y 4.5, este filtro presenta deficiencias al realizar rotaciones demasiado alejadas de la horizontal, esto se debe a las estimaciones obtenidas del acelerómetro y del giroscopio ya no se encuentran sobre el mismo plano. Sin embargo, este problema no afecta a los filtros de Kalman Extendido desarrollados, ya

que para éstos se utilizó una representación de la orientación basada en cuaterniones, además del uso de sistemas no lineales en su modelado.

Comparando la respuesta del EKF con la respuesta del EKF y Matriz de Rotación, este último presentó un mejor rendimiento debido a que en condiciones dinámicas y con vibraciones se producen interferencias magnéticas ya sea por los motores y también por objetos o materiales metálicos cercanos, los cuales generan un campo magnético que afectan a las mediciones obtenidas del magnetómetro, invalidando en primer lugar la calibración realizada a este sensor y por consiguiente disminuyendo la precisión de las estimaciones. En estas condiciones el rendimiento del EKF no sólo se afecta en el eje de desviación, sinó tambien en los demás ángulos. De manera diferente las estimaciones del EKF con una Matriz de Rotación se afectan poco por las interferencias magnéticas al no realizar las mediciones del magnetómetro en su modelado (ver figura 3.15), a cambio de esto se utiliza una Matriz de Rotación mediante los ángulos de Euler para calcular el ángulo de desviación, es decir si hubiese una interferencia magnética solamente afectaría al ángulo de desviación y no al ángulo de giro ni de inclinación. Si bien el algoritmo del EKF y Matriz de Rotación aumentar los cálculos computacionales, este no es considerable ya que el tiempos de ejecución tiene una diferencia poco significativa de 0.2 ms como se muestra en tabla 3.6.

4.4 Análisis del Sistema de Determinación de orientación final

En esta sección se analiza con mayor detalle el rendimiento del sistema utilizando el Filtro de Kalman Extendido con una Matriz de Rotación, ya que fue el filtro que presentó una mayor precisión y exactitud en su respuesta ante las diferentes pruebas en condiciones estáticas y dinámicas.

En estas nuevas pruebas se incluyó al ángulo de desviación para su análisis, aunque sólo de manera visual ya que no se incluye un valor referencial.

Los resultados de las pruebas realizadas se muestran en las figuras 4.6, 4.7 y 4.8, donde se comparan las estimaciones del sistema con los ángulos obtenidos del acelerómetro, giroscopio y magnetómetro, además del valor real o referencial pero éste último sólo para el ángulo de giro e inclinación. En la figura 4.9 se muestra la gráfica la estimación del sesgo (bias) del giroscopio que está incluido en el vector de estados del EKF.

En las figuras 4.6 y 4.7 se aprecia que los valores de los ángulos obtenidos con el giroscopio no se afectan tanto por las vibraciones pero tienen un error el cual se incrementa de manera progresiva en el tiempo. También se muestra que las vibraciones afectan considerablemente a los ángulos obtenidos del acelerómetro. Finalmente se observa que la señal obtenida con las estimaciones del sistema realiza un buen seguimiento a las señal real.

En la figura 4.8 se observa que las mediciones del ángulo de desviación mediante el acelerómetro y magnetómetro son ruidosas, a diferencia de las estimaciones realizadas por el sistema de determinación de orientación. También se puede notar que el ángulo de desviación obtenido utilizando el giroscopio tiene un error que se incrementa en el tiempo.

En la figura 4.9 se aprecian las estimaciones de los sesgos del giroscopio, que presentan un error RMS de velocidad angular (°/s) igual a 0.6330, 0.2368 y 0.8183 en el eje x, eje y, eje z respectivamente.

Se concluye entonces que los resultados del sistema de determinación de orientación implementado en un dispositivo de lógica reconfigurable (FPGA) tiene un buen rendimiento, con estimaciones de gran exactitud y precisión y robusto frente a perturbaciones como vibraciones e interferencias magnéticas.

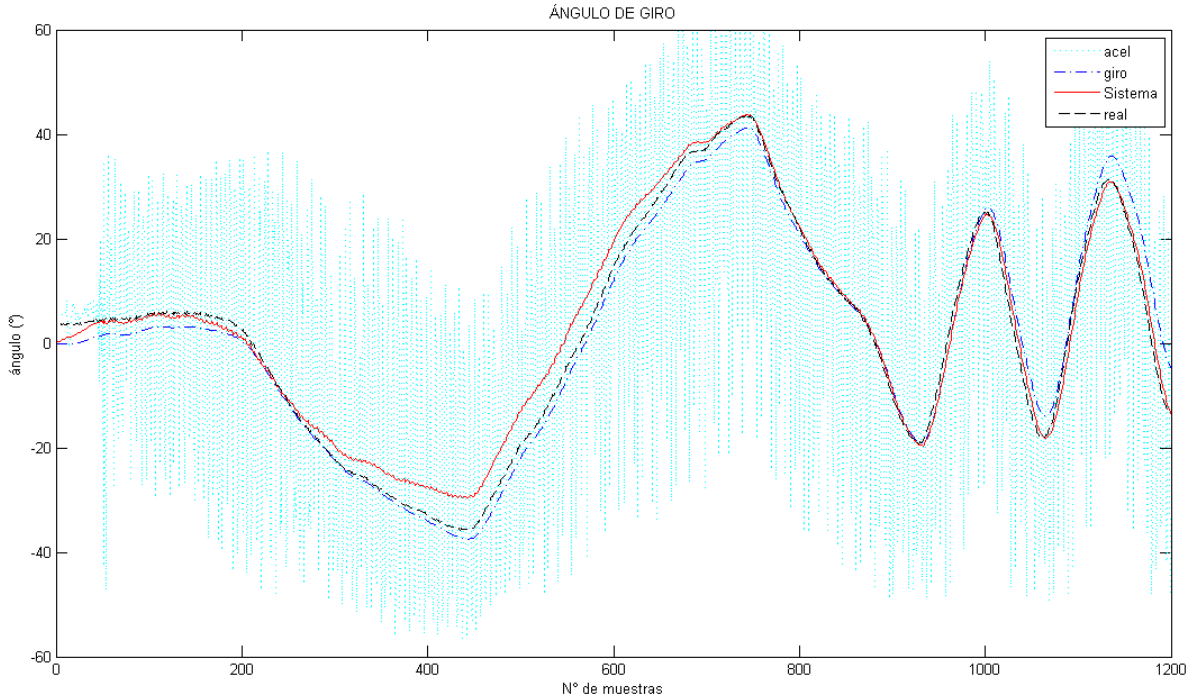

Figura 4.6: Comparación entre el ángulo de giro obtenido del acelerómetro, giroscopio, real y del sistema implementado, en condiciones con perturbaciones

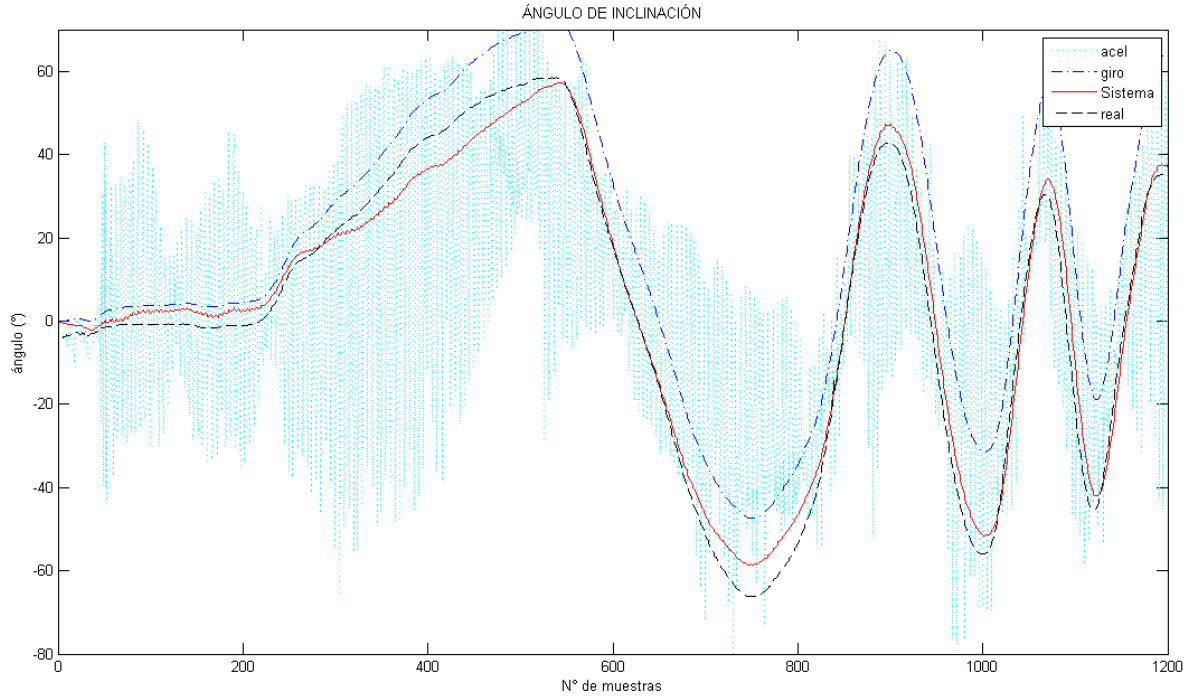

Figura 4.7: Comparación entre el ángulo de inclinación obtenido del acelerómetro, giroscopio, real y del sistema implementado, en condiciones con perturbaciones

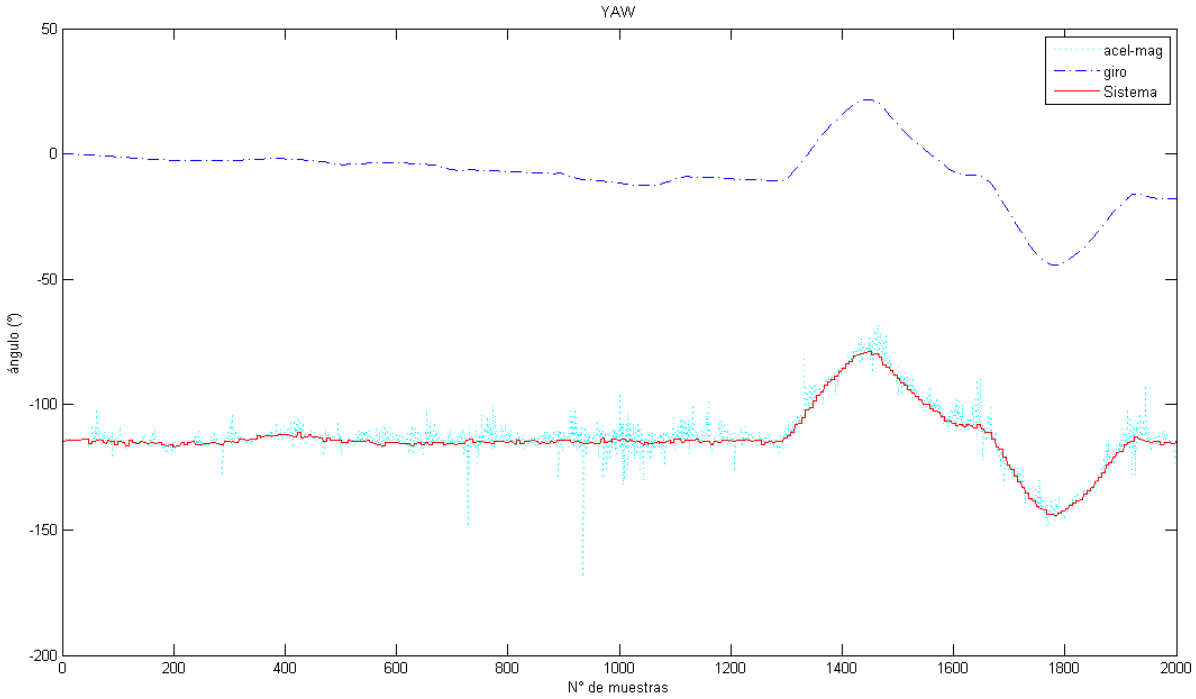

Figura 4.8: Comparación entre el Yaw obtenido del acelerómetro-magnetómetro, giroscopio, y del sistema, en condiciones con perturbaciones

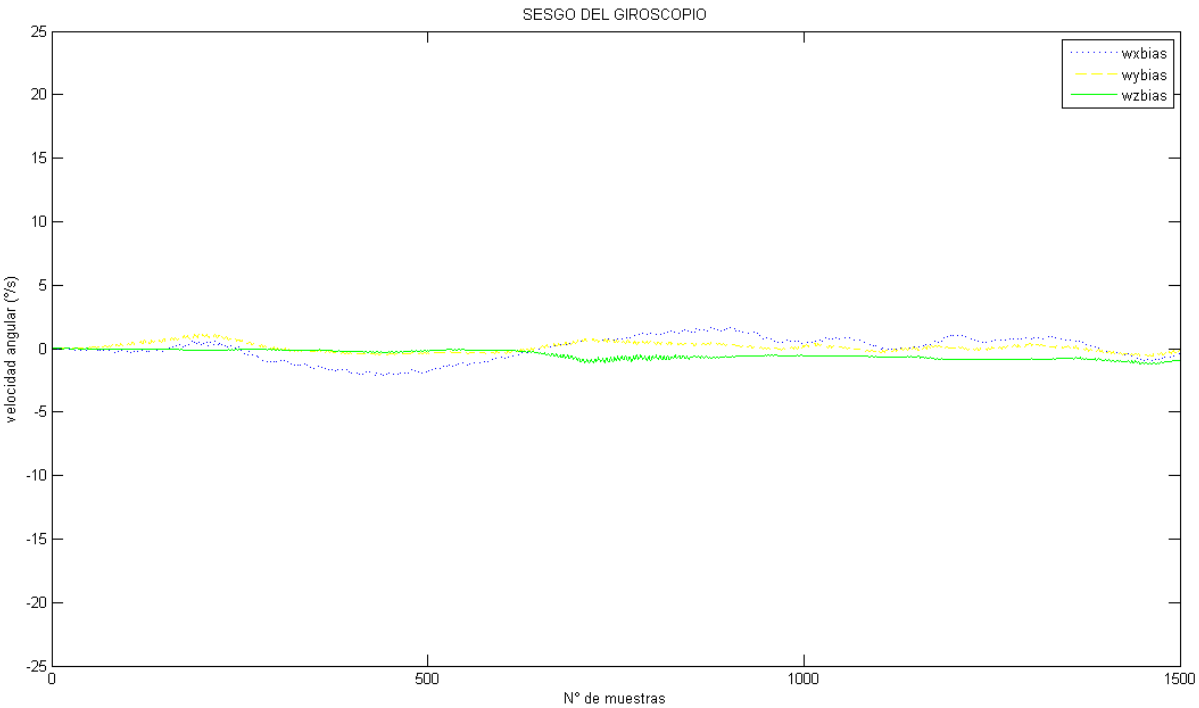

Figura 4.9: Sesgo del giroscopio en el eje x (azul), eje y (amarillo), eje z (verde)

4.5 Análisis de resultados

De los resultados obtenidos del error RMS, media y varianza con los filtros implementados, se concluye que el Filtro de Kalman Extendido evaluado con cuaterniones y una Matriz de Rotación mediante ángulos de Euler resultó ser el mejor estimador al presentar un mejor rendimiento tanto en condiciones estáticas como ante perturbaciones.

Por el contrario, el EKF mostró una menor exactitud mediante las estimaciones por ser vulnerable a las interferencias magnéticas.

Si bien la implementación del EKF con la Matriz de Rotación incrementa el tiempo computacional, las pruebas demostraron que este incremento es poco significativo con una diferencia de 0.2 ms en sus tiempos de ejecución, aplicando el método práctico para los cálculos.

CAPÍTULO 5

CONCLUSIONES Y TRABAJOS FUTUROS

En este proyecto se diseñó, simuló e implementó un sistema de determinación de orientación sobre un dispositivo de lógica reconfigurable (FPGA) utilizando sensores del tipo MEMS de bajo costo para un vehículo aéreo no tripulado del tipo Quadrotor.

También se desarrollaron y compararon tres métodos de fusión de sensores: un filtro de Kalman lineal (FK), filtro de Kalman Extendido (EKF) y Filtro de Kalman Extendido con una Matriz de Rotación.

Se utilizó una plataforma mecánica de bajo costo y de diseño propio para determinar los parámetros de calibración

5.1 Conclusiones

- Las estimaciones de los ángulo de giro, inclinación y desviación del Quadrotor obtenidas a través del Filtro de Kalman Extendido que utiliza las mediciones de la fuerza magnética, velocidad angular y gravedad (MARG) tuvieron un error RMS reducido, concluyendo que el sistema de determinación implementado captura el

movimiento del UAV con precisión y exactitud, convirtiéndose en una plataforma ideal para la implementación de un controlador como puede ser de: orientación, posición o velocidad del Quadrotor.

- El sistema de determinación de orientación desarrollado puede ser implementado en cualquier dispositivo de lógica reconfigurable de la marca Altera.
- Además, este sistema de determinación puede ser implementado en otros UAV como: hexacopter, octocopter o UAV de ala fija (aviones), simplemente volviendo a determinar la varianza de ruido de los sensores en el filtro de Kalman Extendido.
- La implementación del sistema SBA redujo el tiempo computacional del sistema al ejecutar el protocolo de comunicaciones I^2C en un lenguaje de descripción de hardware y permitir que el procesador SOPC NIOS II procese únicamente los algoritmos de calibración y fusión de sensores.
- Se redujo el costo total del proyecto utilizando la plataforma mecánica de calibración de diseño propio, ya que los sensores comerciales calibrados son de costo elevado; mediante esta calibración se mejoró significativamente la precisión de las mediciones de los sensores.
- Finalmente la implementación física del sistema de determinación de orientación propuesto en esta investigación no conlleva a gastos elevados en comparación con otros productos comerciales de determinación de orientación usados.

5.2 Trabajos futuros

Algunas investigaciones futuras para este proyecto, podrían ser:

- La adición de un módulo GPS al sistema permitiría no sólo realizar estimaciones de orientación sino también de trayectoria (posición y velocidad) del Quadrotor.

- Construir una plataforma de banco de pruebas más sofisticada con la capacidad de simular disturbios y mediciones en el eje de desviación, además de la posibilidad de realizar giros simultáneos en los tres ejes del Quadrotor.
- Implementar el algoritmo de estimación de orientación (EKF y Matriz de Rotación) en un lenguaje de descripción de hardware (VHDL), para de esta manera disminuir considerablemente la carga computacional en el procesador SOPC NIOS II.
- Implementación de un algoritmo de control de actitud para el Quadrotor.

BIBLIOGRAFÍA

Aberga, P. (2012). Procesadores Soft-Core: Aplicaciones con FPGA's.

Altera Corp. (2014). *Introduction to the Quartus II Software*. Obtenido de http://www.altera.com/literature/manual/archives/intro_to_quartus2.pdf

Altera Corp. (2015). *High-Performance FPGA Architecture*. Recuperado el 11 de Mayo de 2015, de https://www.altera.com/products/general/devices/stratix-fpgas/about/fpga-architecture.html

Altera Corp. (23 de Enero de 2015). *Stratix V Device Handbook*. Recuperado el 11 de Mayo de 2015, de https://www.altera.com/en_US/pdfs/literature/hb/stratix-v/stx5_core.pdf

Altera Corp. (s.f.). *Nios II Processor Reference*. Recuperado el 12 de 04 de 2014, de http://www.altera.com/literature/hb/nios2/n2cpu_nii5v1.pdf

Altera Corp. (s.f.). *Nios II Software Developer Handbook* . Recuperado el 14 de Mayo de 2015, de http://people.ece.cornell.edu/land/courses/ece5760/NiosII_doc/Nios_SW_handbook.pdf

Altera Corp. (s.f.). *SOPC Builder User Guide*. Recuperado el 13 de 06 de 2014, de http://www.altera.com/literature/ug/ug_sopc_builder.pdf

Altug, E., Ostrowsky, J., & Taylor, C. (Junio de 2004). Vision Based Control of Model Helicopters. *IEEE International Conference on Mechatronics*, 316-321.

Analog Devices. (2012). *Digital Accelerometer ADXL345*. Obtenido de http://www.analog.com/en/mems-sensors/mems-inertial-sensors/adxl345/products/product.html

Berbra, C., Gentil, S., & Lesecq, S. (2009). Identification of Multiple faults in an Inertial Measurement Unit. *7th Workshop Advanced Control and Diagnostic, ACD*.

Bouabdallah, S., & Siegwart, R. (2007). Full control of a Quadrotor. *Intelligent Robots and Systems*, 153-158.

Bouabdallah, S., Murrieri, P., & Siegwart, R. (2004). Design and Control of an Indoor Micro Quadrotor. *IEEE International Conference on Robotics and Automation* , 4393-4398.

Brian, E. (2004). *Design of a Four Rotor Hovering Vehicle.*

Brown, S., & Rose, J. (2002). Architecture of FPGA and CPLDs: A Tutorial. *Design & Test of Computers, IEEE, 13*, 42-57.

Das, A., Subbarao, K., & Lewis, F. (2009). Dynamic inversion with zero-dynamics stabilization for Quadrotor control. *Control Theory & Aplications, IET*, 303-314.

Diebel, J. (2006). *Representing Attitude Euler Angles, Unit Quaternions, and Rotation Vectors.* California.

G.Hoffmann, D.Dostal, S.Waslander, J.Jang, & C.Tomlin. (2004). Stanford Testbed of Autonomous Rotorcraft for Multi-Agent Control(STARMAC). *Digital Avionics Systems Conference.DASC, 2*.

Honeywell International Inc. (2011). *3-Axis Digital Compass IC HMC5883L*. Obtenido de http://www.adafruit.com/datasheets/HMC5883L_3-Axis_Digital_Compass_IC.pdf

IEEE. (2015). *IEEE Standards Association*. Recuperado el 12 de Mayo de 2015, de https://standards.ieee.org/findstds/standard/1076-2000.html

InvenSense, Inc. (2012). *ITG-3200 Product Specification*. Obtenido de https://www.sparkfun.com/datasheets/Sensors/Gyro/PS-ITG-3200-00-01.4.pdf

Jiong, Y., Lei, Z., Jiangping, D., Rong, S., & Jianyu, W. (2010). GPS/SINS/BARO Integrated Navigation System for UAV. *International Foroum on Information Technology and Applications* , 19-25.

Julier, S., & Uhlmann, J. (Abril de 1997). A new Extension of a the Kalman Filter to Nonlinear Systems. *SPIE AeroSense Symposium*.

Kumar, A., & Pinjare, S. (2012). Implementation of Neural Network Back Propagation Training Algorithm on FPGA. *International Journal of Computer Applications*, 6-13.

La Viola, J. (2003). A Comparison of Unscented and Extended Kalman Filtering for Estimating Quaternion Motion. *American Control Conference*.

Madgwick, S. (30 de Abril de 2010). An efficient orientacion filter for inertial and inertial/magnetic sensor arrays. *Technical report, Department of Mechanical Engineering, University of Bristol*.

MathWorks, Inc. (s.f.). *PID Controller Tuning*. Recuperado el 10 de 01 de 2014, de http://www.mathworks.com/discovery/pid-tuning.html

MathWorks, Inc. (2009). *Ellipsoid fit*. Recuperado el 2014, de http://www.mathworks.com/matlabcentral/fileexchange/24693-ellipsoid-fit

MathWorks, Inc. (s.f.). *System Identification ToolBox*. Recuperado el 03 de 01 de 2014, de http://www.mathworks.com/products/sysid/description2.html

Maxifield, C. (2004). *The Design Warrior's Guide to FPGAs.*

Maybeck, P. (1979). *Stochastic models, estimation, and control.* Ohio: Academic Press.

Mckerrow, P. (2004). Modelling the Draganflyer four-rotor helicopter. *IEEE international Conference on Robotics and Automation*, 3596-3601.

National Instruments (NI). (21 de Diciembre de 2011). *National Instruments*. Recuperado el 10 de Mayo de 2015, de National Instruments.

NOAA. (25 de 10 de 2007). *NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION*. Recuperado el 22 de mayo de 2014, de National Geophysical Data Center: http://www.ngdc.noaa.gov/geomag/

Nonami, K., Kendoul, F., Suzuki, S., Wang, W., & Nakazawua, D. (2010). *Autonomous Flying Robots Unmanned Aerial Vehicles and Micro Aerial Vehicles.* Tokyo: Springer Japan.

Ogata, K. (2002). *Ingeniería de Control Moderna.* Prentice Hall.

Pereyra, M., Gallia, N., Alasia, M., & Micolini, O. (2013). Heterogeneous Multi-Core System,Synchronized by a Petri Processor on FPGA. *Latin America Transactions*, 224-229.

Pflimlin, J., Hamel, T., Soueres, P., & Metni, N. (2005). NonLinear Attitude and Gyroscope's bias Estimation for a VTOL UAV. *38*(3).

Pierre, M., & Denis, G. (2004). Comparison between the Unscented Kalman Filter and the Extended Kalman Filter for the Position Estimation Module of an Integrated Navigation Information System. *IEEE Intellligence Vehicles Symposium*.

Raffo, G. (2007). Modelado y control de un helicoptero quad-rotor. *Master's thesis*. Universidad de Sevilla, España.

Risco, M. (2014). *Arquitectura de Bus Simple, un Conjunto de Herramientas para el Desarrollo Portable de Sistemas en Chip*. Recuperado el 17 de Mayo de 2015, de https://docs.google.com/file/d/0B4ddOwd1tkyZYmthVUd4RnpHWmc/edit?pli=1

S. Bouabdallah, P. M. (2004). Design and Control of an Indoor Micro Quadrotor. *IEEE International Conference on Robotics and Atutomation, 5*, 4393-4398.

Sánchez, J. (1 de Octubre de 2011). Historia de Matemáticas Hamilton y el Descubrimiento de los Cuaterniones. *G.I.E Pensamiento Matemático*.

Shaojie, S., Yash, M., Nathan, M., & Vijay, K. (Junio de 2013). Vision Based State Estimation and Trayectory Control Towads High Speed Fligh with a Quadrotor. *Robotics: Science and Systems*.

Skog, I., & Handel, P. (2006). Calibration of a MEMS Inertial Mesaurement Unit. *XVII IMEKO World Congress on Metrodology for Sustainable Development*.

Sparkfun Electronics. (2012). *Sparkfun*. Recuperado el noviembre de 2011, de http://www.sparkfun.com/

Terasic Technologies Inc. (2011). *DE0-Nano user manual*. Obtenido de http://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&No=593&PartNo=4

Van der Merwe, R., & Wan, E. (2004). Sigma Point Kalman Filters for Nonlinear Estimation and Sensor Fusion Applications to Integrate Navigation. *AIAA Guiadance, Navigationm and Control Conference* .

Volnei A., P. (2004). *Circuit Design with VHDL.* England: Massachusetts Institute of Technology.

Wan, E., & Van der Merwe, R. (2000). The Unscented Kalman Filter for Nonlinear Estimation. *IEEE Adaptative Systems for Signal Processing, Comunications, and Control Symposium*, 153-158.

Welch, G., & Gary, B. (24 de Julio de 2006). An Introduction to the Kalman Filter. *Technical report* .

Zhang, T., Kang, Y., & Achtelik, M. (2009). Autonomous Hovering of a Vision/IMU Guided Quadrotor. *International Conference on Mechatronics and Automation.ICMA*.

ANEXOS

A. Diagrama UML de las configuraciones de los sensores

En la figura A.1 se muestra el diagrama UML (Lenguaje Unificado de Modelado) de las configuraciones para el acelerómetro, magnetómetro y giroscopio. Las configuraciones se realizaron utilizando el SBAController.

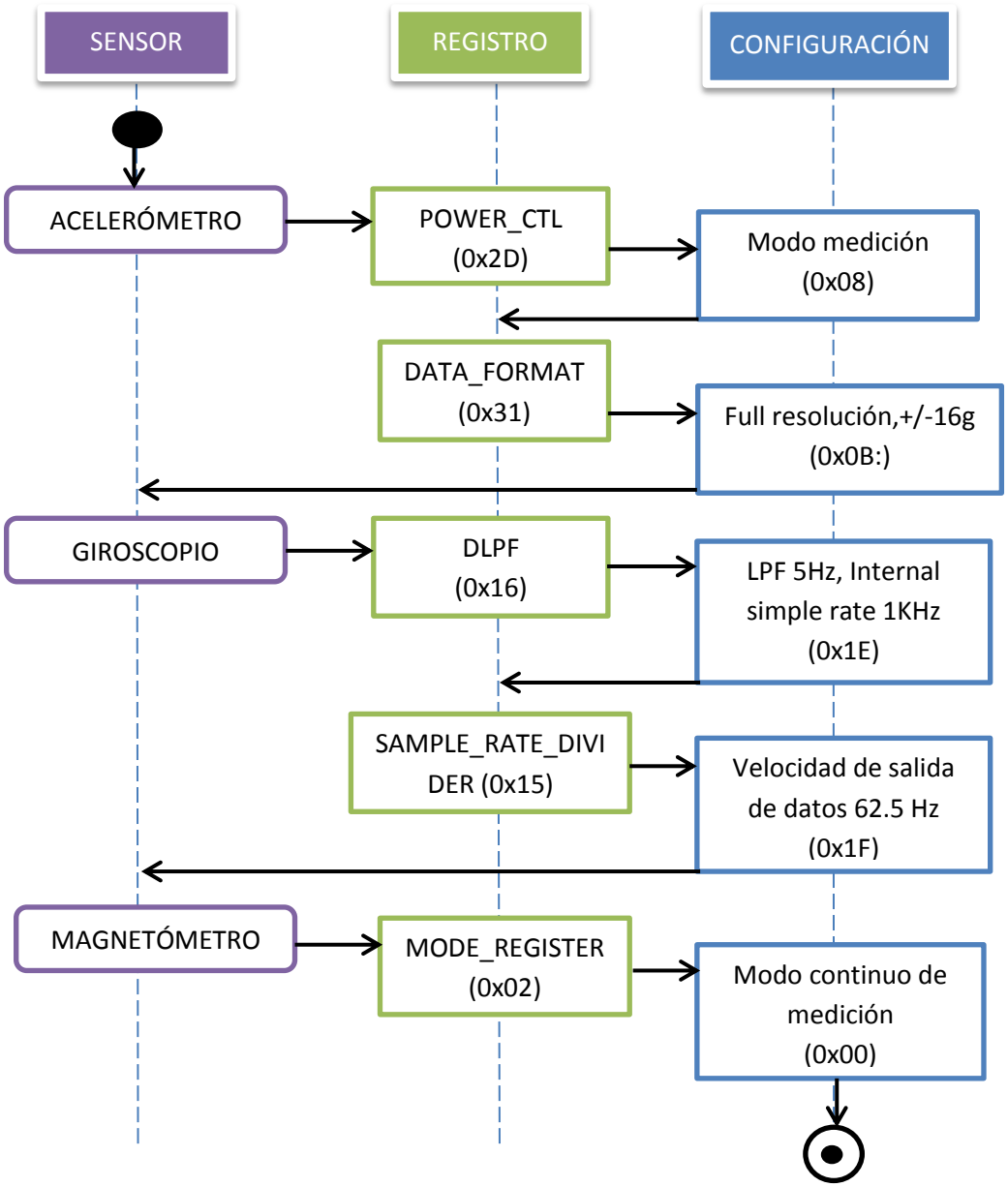

Figura A.1: Diagrama de flujo de las configuraciones de los sensores

B. Estructura mecánica de la plataforma de calibración

En el sistema mecánico de la plataforma de calibración se incorporó en la base un mecanismo conformado por un motor DC con encoder (figura B.1), cuyo giro se transmite a un tornillo sin fin y un sistema de engranajes mediante una polea. Este sistema proporciona una buena reducción de la velocidad, además de generar mayor fuerza de giro.

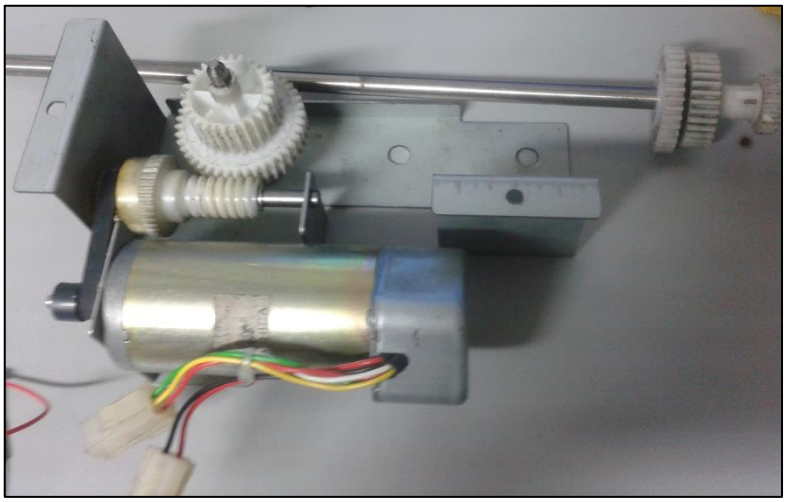

Figura B.1: Sistema mecánico reductor de velocidad

En la figura B.2 se muestra el sistema acoplado a la plataforma superior, mediante un eje ajustado por una chumacera.

Para obtener los parámetros de calibración correctos del magnetómetro, la plataforma se tenía que construir de un material no metálico para no generar distorsiones magnéticas, por lo que se eligió melanina, que es un material ligero, económico y de fácil manipulación, pero principalmente este material no genera ninguna distorsión magnética. Por esta razón, también el sensor tendría que estar lo suficientemente alejado del campo magnético generado por el motor DC, así que, con mediciones realizadas a diferentes distancias del motor, se eligió una distancia adecuada de separación de 1.5 m entre el motor DC y la caja que contiene los sensores. En dicha caja (figura B.3) también se

coloca una DE0-Nano para la obtención de datos de los sensores, un transmisor inalámbrico (Xbee) y una batería como fuente de energía, en este caso se utilizó la batería de un teléfono portátil.

Figura B.2: Sistema reductor de velocidad de la plataforma de calibración

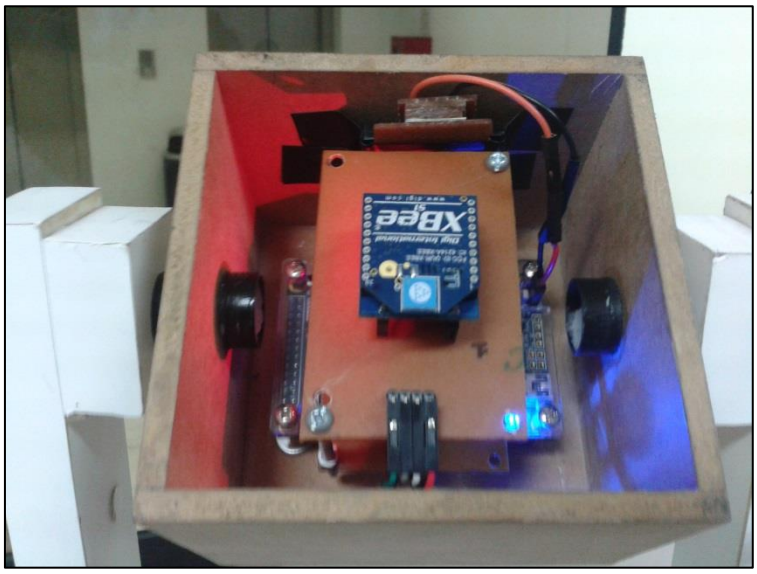

Figura B.3: Caja de la plataforma de calibración con los sensores, DE0-Nano, Xbee y una batería

C. Vibraciones en el Quadrotor

En este anexo se muestran los resultados de las tablas (4.6), (4.7) y (4.8) de manera gráfica. Los datos fueron calibrados antes de ser graficados.

En las figuras C.1, C.2 y C.3 se muestran los resultados del efecto en los sensores de las vibraciones producidos por los motores tanto para el acelerómetro, giroscopio y magnetómetro, respectivamente. Las pruebas se realizaron a diferentes potencias de los motores, controlados por PWM (min450 - max700).

En la figura C.1 se muestra el gran nivel de perturbación que generan las vibraciones al acelerómetro (>1g), a diferencia de las mediciones del giroscopio (<1°/s) como se muestra en la figura C.2. El efecto de las vibraciones para el magnetómetro es también muy bajo, ya que la principal causa de disturbios para este sensor son las interferencias magnéticas.

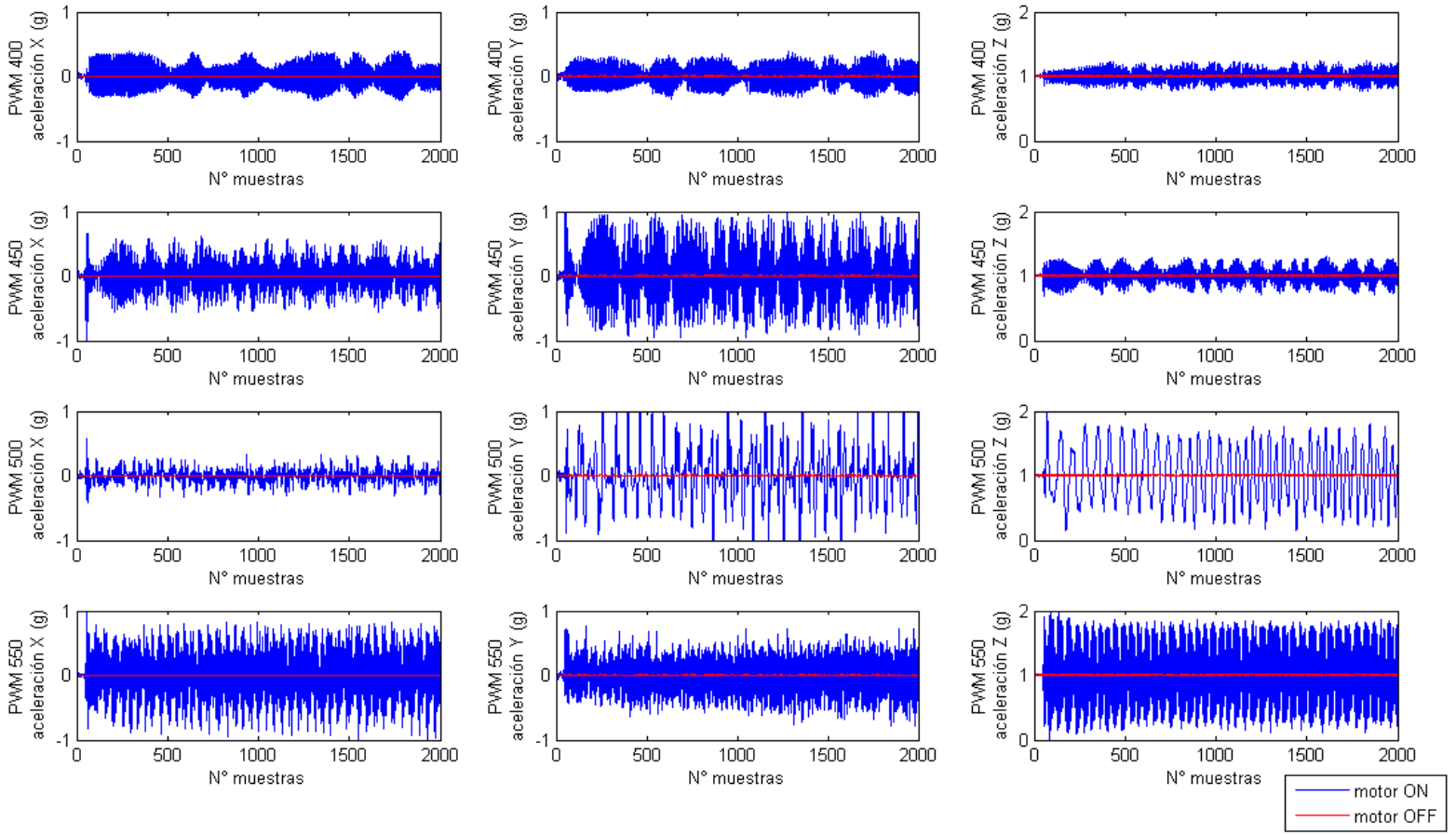

Figura C.1: Vibraciones en el acelerómetro

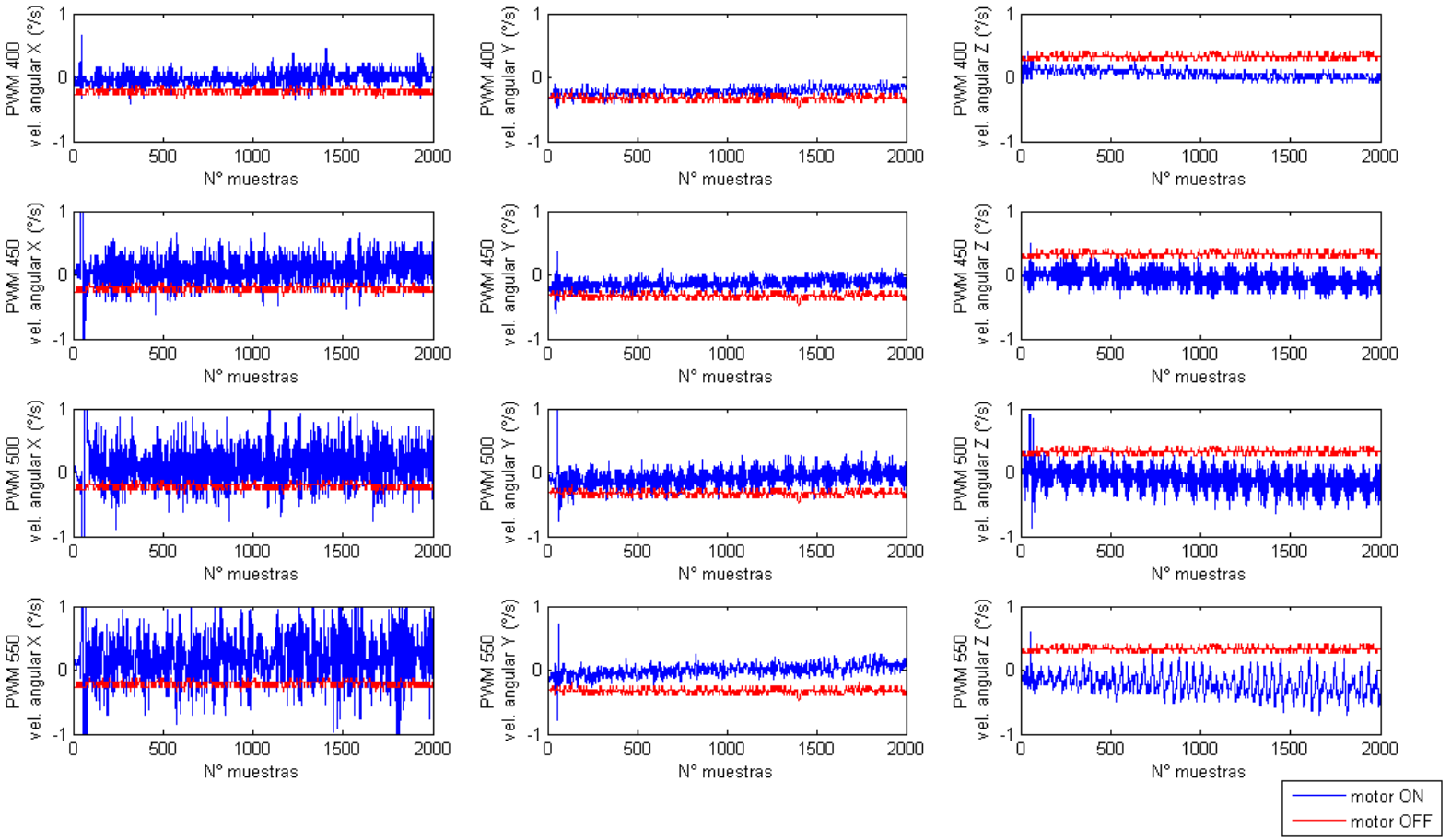

Figura C.2: Vibraciones en el giroscopio

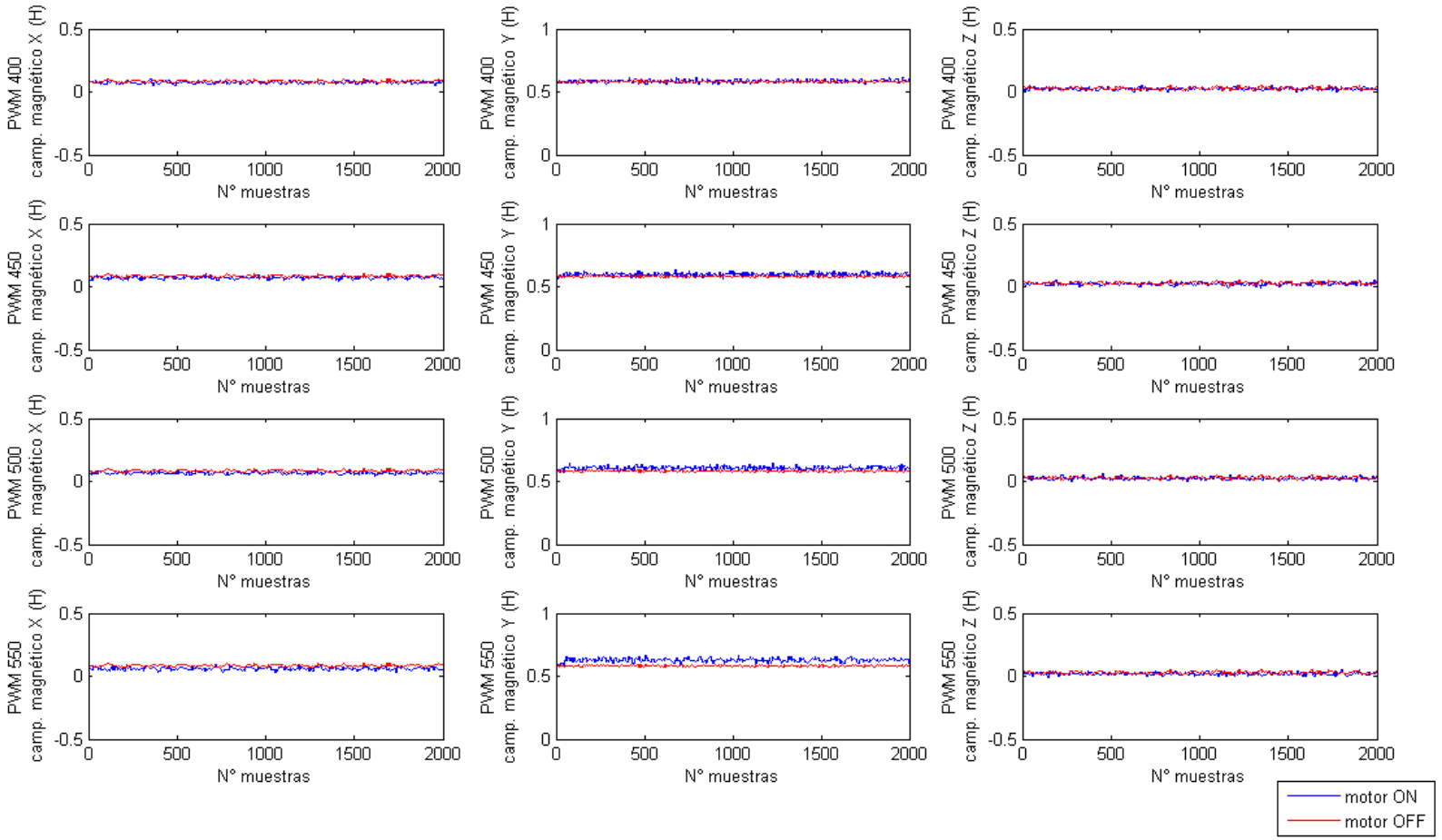

Figura C.3: Vibraciones en el magnetómetro

D. Estructura del Quadrotor

La estructura del UAV está conformada por motores brushless, controladores ESC (Electronic Speed Controller), una tarjeta de control de diseño propio que contiene: los sensores, RC radio transmitter/receiver y conversores 5-3.3V. Las características de estos compontes son:

Motores brushless

Son motores sin escobillas de 920Kv o RPM/V (figura B.1), con la capacidad de carga de aproximadamente 600 gramos por motor. Este tipo de motores son de pequeño tamaño (22 x 12mm) ideales para un Quadrotor.

Figura D.1: Motor brushless DJI.

ESC (Electronic Speed Controller)

Los motores brushless funcionan con voltaje trifásico que se controla mediante un ESC (controlador de velocidad). En la figura B.2 se muestra el ESC 30A opto con un rango de frecuencia de 30 – 450Hz. Funciona con baterías de 3 o 4 celdas.

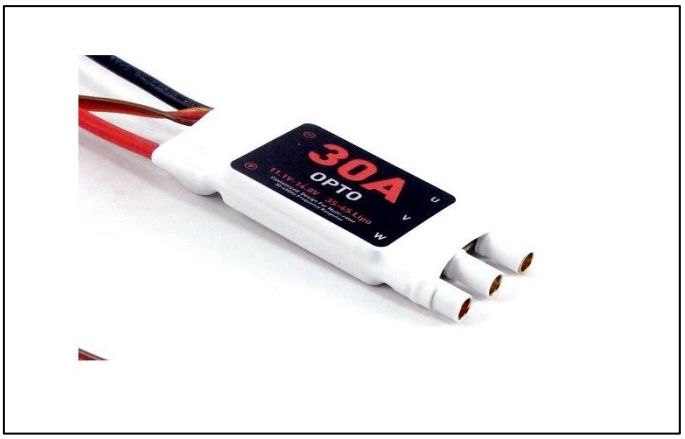

Figura D.2: ESC 30A OPTO.

Batería

La batería utilizada es del tipo LiPo (Lithium Polymer) de 3 celdas con una capacidad máxima de 4500mAh a 11.1 volt. Y un peso de 386g. Se eligió este tipo de baterías debido a su alta capacidad de almacenamiento y poco peso, en la figura B.3 se muestra una imagen de la batería.

Figura D.3: Batería LiPo 3 celdas 11.1v.