



Universidad Michoacana
de
San Nicolás de Hidalgo.

Facultad de Ingeniería Eléctrica
TESIS

*Control de Posición de un
Cuadricóptero en Dos Ejes*

Para obtener el grado en
Ingeniero en electrónica

Presenta
Benjamín Arcos Alcantar

Asesor
M.C. Salvador Ramírez Zavala

Morelia Michoacán
Julio del 2019



Agradecimientos

Gracias a dios y a mis padres Jaime Arcos Miranda y Graciela Alcantar Martinez, por el apoyo en cada uno de los momentos, que eran difíciles para mí como aun así para ellos. Gracias a mis hermanos por su motivación, a mis familiares y amigos por creer en mí por su apoyo, gracias a mis maestros y asesor por brindarme las herramientas para lograr concluir mi trabajo de tesis así como lograr cumplir con mis proyectos durante mi trayectoria dentro de la Facultad de Ingeniería Eléctrica (FIE). Así también agradecer a esta universidad (Universidad Michoacana de San Nicolás de Hidalgo), por haberme permitido formarme en ella. Gracias a mis compañeros por brindarme el apoyo en esos momentos en los que me surgían dudas y ellos lograban aclararlas.

Muchas gracias a todos por creer en mí, por su motivación y su apoyo durante mi formación como estudiante y persona.

Dedicatorias

A mis padres Jaime Arcos Miranda y Graciela Alcantar Martinez por haberme dado la educación necesaria para elegir el buen camino, muchos de mis logros se los debo a ellos por su apoyo que siempre he tenido, por sus consejo y motivación que me han brindado durante toda mi trayectoria.

Así como toda mi familia, amigos que me has motivado a concluir cada una de mis metas. También la Facultad de ingeniería Eléctrica (FIE) y mis maestros que me enseñaron a superar mis obstáculos durante mi formación dentro de la facultad.

A todos gracias por su apoyo.

Índice

Agradecimientos	1
Dedicatorias	1
Índice	2
Resumen	4
Abstract	5
Palabras Clave	5
Keywords	5
Lista de Figuras	6
Glosario de términos	9
Capítulo 1 Introducción.	11
1.1 Antecedentes	11
1.2 Objetivo General	12
1.2.1 Objetivos particulares	12
1.3 Justificación	13
1.4 Metodología	13
1.5 Descripción de los capítulos	14
Capítulo 2 Estado del arte	15
2.1 Tipos de vehículos aéreos	15
2.2 Estructura general de los vehículos aéreos	17
2.3 Principales movimientos realizados por los vehículos aéreos	20
2.3.1 Movimiento de posición	20
2.3.2 Movimiento de elevación	21
2.3.3 Movimiento de traslación	21
2.4 Giros para lograr las posiciones deseadas	22
2.5 Componentes electrónicos	23
2.5.1 Motores	24
2.5.2 Tipos de Hélices	25
2.5.3 Drivers	27
2.5.4 Sensor de Posición	29
2.5.5 Microcontroladores	30

2.6 Comunicación inalámbrica -----	32
Capítulo 3 Hardware y software -----	33
3.1 Estructura -----	33
3.2 Motor de CD -----	34
3.3 Hélices Híbridas -----	35
3.4 Etapa de potencia.-----	36
3.5 Sensor de posición (MPU6050) -----	37
3.6 Microcontrolador-----	39
3.6.1 Dispositivos de comunicación Bluetooth -----	41
3.6.2 Control con Arduino -----	44
3.7 Controlador PID -----	47
3.8 Interface Gráfica.-----	51
Capítulo 4 Pruebas y Resultados-----	57
4.1 Introducción -----	57
4.2 Filtro digital. -----	57
4.3 Referencia en cero grados -----	60
4.4 Referencia en $x=-10$, $y=10$ -----	64
4.5 Límites de control positivos ($x=30,y=30$) -----	68
4.6 Límite de control negativo en $x=-20$, $y=-40$ -----	72
4.7 Perturbación Baja -----	76
4.8 Perturbación alta -----	80
Capítulo 5 Conclusiones -----	85
5.1 Conclusión -----	85
5.2 Trabajos futuros -----	86
Bibliografía -----	87
Anexo A -----	88
Anexo B -----	98
Anexo C -----	107

Resumen

El trabajo consiste en diseñar un controlador clásico del tipo proporcional integral derivativo (PID), capaz de mantener la estabilidad de un Cuadricóptero en una posición referenciada con coordenadas X, Y; además de poder cambiar la posición deseada del Cuadricóptero con ayuda de una comunicación inalámbrica entre el Cuadricóptero y la PC, así como observar su comportamiento en una computadora personal. Para lograr esto primeramente, se construyó el Cuadricóptero seleccionando los motores, las hélices, y la estructura utilizada (base en la cual van montados todos los elementos electrónicos), así como la electrónica necesaria para lograr el movimiento de los motores y así también la elección del microcontrolador para llevar a cabo el control deseado. Para ello se realizó el programa necesario para controlar el Cuadricóptero en un lenguaje de programación en C. También se realiza una interfaz gráfica en Java para visualizar el comportamiento del Cuadricóptero en forma gráfica con lo cual programo el microcontrolador. Al finalizar se presentan las pruebas realizadas, las cuales resultaron satisfactorias ya que el Cuadricóptero se coloca en la posición deseada y responde a perturbaciones. Y finalmente se presentan algunas conclusiones y trabajos futuros.

Abstract

The job is to design a classic proportional integral derivative (PID) type controller, capable of maintaining the stability of a Quadcopter in a referential position with X, Y strings: in addition to being able to change the desired position of the Quadcopter with the help of a wireless communication between the Quadcopter and the PC, as well as observe their behavior on a personal computer. To achieve this first, the Quadcopter was built by selecting the motors, the propellers, and the height used (base on which all the electronic elements are mounted), as well as the electronics necessary to achieve the movement of the motors and thus also the choice of the microcontroller to carry out the desired control. For this purpose, the program necessary to control the Quadcopter was carried out in a program C language. A Java interface is also made to visualize the behavior of the Quadcopter in graphic form, with which I program the microcontroller. At the end, the tests performed are presented, which were satisfactory since the Quadcopter is placed in the desired position and responds to disturbances. Finally, some conclusions and future works are presented.

Palabras Clave

UAV, RPV, GPS, SRA, Drones, Controlador, Sistema Cuadricóptero, Wifi, PID, PC, Aerostatos ,Aerodinos BLL, Tándem, ESC,DSP,GMS.

Keywords

UAV, RPV, GPS, SRA, Drones, Cuadricóptero, Wifi, PID, PC, *aerostats*, aerodyns, BLL, Tándem, ESC,DSP,GMS

Lista de Figuras

Figura 1. Línea del tiempo de los términos que recibieron los vehículos aéreos desde 1810 hasta el 2010 de acuerdo a la referencia [1].	11
Figura 2. Imagen de algunos aerodinos.	15
Figura 3. Cuadricóptero de uso comercial	16
Figura 4 Vehículo aéreo tipo monorotor es decir, de un solo rotor	17
Figura 5. Birotos tipo Coaxial	17
Figura 6 Aeronave con birotor tipo BLL engranado	18
Figura 7 Aeronave con birotor tipo BLL no engranado	18
Figura 8 Birotor tipo tándem	19
Figura 9. Primer Tricoptero.	19
Figura 10. Imagen de un Cuadricóptero.	19
Figura 11 Base para simulación de vuelo y control del Cuadricóptero diseñado	20
Figura 12 Característica de cómo es necesario variar la velocidad para, lograr el movimiento deseado. En la configuración en cruz como en X, respectivamente.	23
Figura 13. Componentes de un Cuadricóptero.	24
Figura 14. Se muestra el diagrama de un motor de CD y un motor tipo Brushless, respectivamente	24
Figura 15. Hélices acabadas en punta	26
Figura 16. Hélices tipo Bullnose	26
Figura 17. Hélices tipo híbridas	27
Figura 18. Diagrama de un puente H	27
Figura 19 Señales PWM según su ciclo de trabajo	28
Figura 20. Controlador de Velocidad ESC	29
Figura 21. Posición de giro de un vehículo aéreo	30
Figura 22. Arquitectura de un Microcontrolador	31
Figura 23. Microcontrolador ATmega 328 utilizado por la placa Arduino nano	31
Figura 24 Estructura y sus partes del Cuadricóptero diseñado	33
Figura 25 Estructura del Cuadricóptero para el montaje de los elementos electrónicos	34
Figura 26. Motor de CD symaX5C	35
Figura 27 Motor con sus respectiva hélice y engranaje para aumentar el torque	36
Figura 28 Transistor Tip 41c Utilizado como Driver por el Cuadricóptero	36
Figura 29 Diagrama de transistor tipo NPN	37
Figura 30. Sensor MPU 6050	37
Figura 31. Efecto Coriolis.	38
Figura 32 Sensor interno en el MPU 6050 (Giroscopio)	38
Figura 33 Sensor interno en el MPU 6050 (Acelerómetro)	39
Figura 34. Placa Arduino nano	39
Figura 35. Imagen de software ID de Arduino	40
Figura 36. Monitor Serial del ID de Arduino	42
Figura 37 Icono al instalar el Bluetooth y asistente Toshiba (Configuración)	43
Figura 38. Asistente Toshiba	43

Figura 39. Diagrama de diseño del Cuadricóptero -----	44
Figura 40 Esquema del procedimiento del filtro promediador -----	47
Figura 41. Diagrama de un controlado PID para un motor -----	50
Figura 42 Diagrama del software utilizado en la interfaz gráfica diseñada -----	52
Figura 43. Interfaz gráfica de control y monitoreo del Cuadricóptero. -----	53
Figura 44 Área en donde se grafica la respuesta u comportamiento del Cuadricóptero -----	53
Figura 45. Indicadores Analógicos – Digital (Meters). -----	54
Figura 46 Área del manejo del Cuadricóptero -----	54
Figura 47 Archivo de texto creado por la interface grafica -----	56
Figura 48. Gráfica del filtro digital promediador aplicado al eje X -----	58
Figura 49 Zoom del filtro promediador aplicado al eje X -----	58
Figura 50. Gráfica del filtro digital promediador aplicado al eje Y -----	59
Figura 51, Zoom del filtro promediador aplicado al eje Y -----	59
Figura 52 Imagen que muestra al Cuadricóptero en su posición inicial en la cual se toman los datos para su respuesta del controlador implementado con referencia de cero grados en ambos ejes. -----	60
Figura 53. Gráfica de la respuesta del controlador con una referencia de $X=0$, $Y=0$ -----	61
Figura 54. Gráfica de la acción de control para el motor 1 con giro horario. -----	61
Figura 55. Gráfica de la acción de control para el motor 2 con giro horario. -----	62
Figura 56. Gráfica de la acción de control para el motor 1 con giro anti-horario. -----	62
Figura 57. Gráfica de la acción de control para el motor 2 con giro anti-horario. -----	63
Figura 58. Gráfica de la acción de control aplicada a los cuatro motores. -----	63
Figura 59. Gráfica del error para una referencia de cero en ambos ejes. -----	64
Figura 60 Imagen del Cuadricóptero en su posición inicial para iniciar con las pruebas de cambio de referencia -----	64
Figura 61. Gráfica con referencia distinta de cero, para mostrar el cambio de referencia X y Y. -----	65
Figura 62. Gráfica de la acción de control para el motor 1 con giro horario. -----	65
Figura 63. Gráfica de la acción de control para el motor 2 con giro horario. -----	66
Figura 64 Gráfica de la acción control para el motor 1 con giro anti-horario -----	66
Figura 65 Gráfica de la acción de control para el motor 2 con giro anti –horario. -----	67
Figura 66 Gráfica de la acción de control aplicada a los cuatro motores -----	67
Figura 67. Gráfica del error para ambos ejes, en la cual se muestra un pequeño erro marcado por falta de alcance a su referencia marcada. -----	68
Figura 68 Imagen que muestra al Cuadricóptero en su posición inicial en la cual se toman los datos para la respuesta del controlador con los limites positivos -----	68
Figura 69. Gráfica en la cual se muestran, los resultados para los límites positivos en donde el control continúa siendo funcional. -----	69
Figura 70. Gráfica de la acción de control aplicada al motor 1 con giro horario. -----	69
Figura 71. Gráfica de la acción de control para el motor 2 con giro horario. -----	70
Figura 72. Gráfica de la acción de control para el motor 2 con giro anti-horario. -----	70
Figura 73. Gráfica de la acción de control para el motor 2 con giro anti-horario. -----	71
Figura 74. Gráfica de la acción de control, para los cuatro motores. -----	71

Figura 75. Gráfica del error obtenido para los límites positivos, en el eje X y eje Y. -----	72
Figura 76 Imagen que muestra al Cuadricóptero en su posición inicial en cual se toman los datos para la respuesta del controlador de los límites negativos. -----	72
Figura 77. Gráfica de los límites negativos para el eje “x” como para el eje “y” con una referencia inicial de cero grados y con un pequeño sobrepaso en el eje “y” causado por el cambio brusco de 0 a -40 grados. -----	73
Figura 78. Gráfica de la acción de control para el motor 1 con giro horario. -----	73
Figura 79. Gráfica de la acción de control para el motor 2 con giro horario. -----	74
Figura 80. Gráfica de la acción de control para el motor 1 con giro anti-horario. -----	74
Figura 81. Gráfica de la acción de control para el motor 2 con giro anti-horario. -----	75
Figura 82. Gráfica de la acción de control, para los cuatro motores y sus límites negativos. -----	75
Figura 83. Gráfica del error para el eje x y eje y, en los límites negativos. -----	76
Figura 84 Imagen del Cuadricóptero en su posición inicial, para la prueba de una perturbación baja con referencia en cero grado -----	76
Figura 85 Gráfica de una respuesta a una perturbación baja con una posición inicial en $X=20$, $Y=39$ y una referencia de cero grados ($X=0, Y=0$) -----	77
Figura 86. Gráfica de la acción de control para el motor 1 con giro horario. -----	77
Figura 87. Gráfica de la acción de control para el motor 2 con giro horario. -----	78
Figura 88. Gráfica de la acción de control para el motor 1 con giro anti- horario. -----	78
Figura 89. Gráfica de la acción de control para el motor 2 con giro anti-horario. -----	79
Figura 90. Gráfica de la acción de control para los cuatro motores. -----	79
Figura 91 Gráfica del error ante una perturbación baja. -----	80
Figura 92 Imagen del Cuadricóptero en su posición inicial, para la prueba de una perturbación alta con referencia en cero grado -----	80
Figura 93. Gráfica de la respuesta ante una perturbación alta y referencia de $X=0$, $Y=0$. ----	81
Figura 94. Gráfica de la acción de control para el motor 1 con giro horario. -----	81
Figura 95. Gráfica de la acción de control para el motor 2 con giro-anti horario. -----	82
Figura 96. Gráfica de la acción de control para el motor 1 con giro anti - horario -----	82
Figura 97. Gráfica de la acción de control para el motor 2 con giro anti- horario -----	83
Figura 98. Gráfica de la acción de control para los cuatro motores ante una perturbación alta. -----	83
Figura 99. Gráfica del error en ambos ejes, ante una perturbación alta. -----	84

Glosario de términos

Cuadricóptero:

Vehículo aéreo no tripulado impulsado por cuatro motores con hélices, sitiados en forma o configuración de cruz o en x.

Manteniendo el control o posicionándose al variar la velocidad de los motores necesarios de manera que este consiga la posición deseada.

Dron:

Es un vehículo aéreo que vuela sin tripulación. Su nombre se deriva del inglés drone, que en español significa “abeja macho”.

Control remoto:

Un control remoto o mando a distancia es un dispositivo electrónico usado para realizar una operación remota sobre un equipo (Cuadricóptero).

Monoplano:

Es un vehículo aéreo que solo consta de un rotor para lograr su sustentación en el aire.

Autonomía:

Es la capacidad para darse reglas a uno mismo o tomar decisiones sin intervención ni influencia externa.

Arduino:

Es una plataforma de prototipos electrónica de código abierto basada en Hardware y Software.

I²C:

Bus que permite la comunicación entre circuitos integrados. Esto por medio de dos hilos y uno de tierra.

Bluetooth:

Es una especificación industrial para Redes Inalámbricas de Área Personal (WPAN) creado por Bluetooth Special Interest Group, Inc. que posibilita la transmisión de voz y datos entre diferentes dispositivos mediante un enlace por radio frecuencia en la banda ISM (Industrial Scientific and Medical) de los 2.4 Ghz

Interface gráfica:

Es conocida también como GUI (del inglés graphical user interface), es un programa informático que actúa de interfaz de usuario, utilizando un conjunto de imágenes y objetos gráficos para representar la información y acciones disponibles en la interfaz.

Palas:

Alas móviles que ayudan a suspender un vehículo aéreo, gracias a la repulsión de aire ocasionada por su movimiento.

CD:

Corriente directa.

ESC (Electronic Speed Controller):

(Electronic Speed Controller) es un dispositivo electrónico que sirve para controlar la velocidad del motor brushless.

GSM (Global System for Miobile Communications):

Estándar europeo abierto para que una red digital de teléfono móvil soporta voz, datos, mensajes de texto y roaming en varios países. El GSM (Global System for Miobile Communications) es ahora uno de los estándares digitales inalámbricos 2G.

Capítulo 1

Introducción.

1.1 Antecedentes

A lo largo de la primera mitad del siglo XIX Cayley, Stringfellow, DuTemple entre otros pioneros de la aviación, iniciaron con el desarrollo de la aviación no tripulada.

Esto con la necesidad de realizar pruebas de aviones que, posteriormente pudieran ser contruidos y ser piloteados con pilotos a bordo.

A partir del año 1890, a los vehículos aéreos no tripulados, se les reconoció como UAV (Unmannend Aerial Vehicle), el cual fue uno de los términos reconocidos para ser utilizado en la aviación no tripulada. Aunque este es uno de los términos que actualmente está en desuso.

En la actualidad existen diferentes términos que son utilizados y popularizados logrando una costumbre. Esto ocasiona que los términos sean mal interpretados.

Uno de los ejemplos es el termino dron, el cual es un término que proviene del ámbito militar, que se utilizó para vehículos aéreos no tripulados, controlados a distancia mediante mando o autónomos los cuales se pueden posicionar o dirigir hacia a una ruta por medio de GPS (Global Positioning System) esto con fines militares en su inicio.

Este término fue el primero ya que este tipo de estudio y desarrollo de la aviación no tripulada tomo más de un término, desde su inicio de los vehículos aéreos no tripulados hasta la actualidad. Otro de los términos muy reconocidos es el término Sistema Aéreo pilotado Remotamente RPAS (Remotely Piloted Aircraft System).

Estos dos términos (UAV) son solo algunos de los principales que han dado a los vehículos aéreos no tripulados.

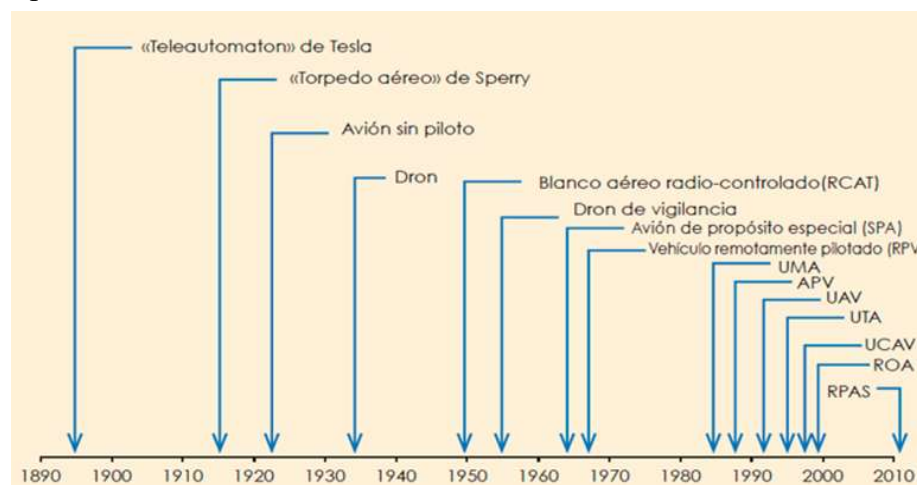


Figura 1. Línea del tiempo de los términos que recibieron los vehículos aéreos desde 1810 hasta el 2010 de acuerdo a la referencia [1].

En la Figura 1 se muestran los nombres que recibieron los vehículos aéreos no tripulados hasta el 2010 [1].

Durante la primera guerra mundial, fue una época en donde se inició con el desarrollo de la aviación, mientras la aviación no tripulada se mantenía un poco ausente por falta de desarrollo tecnológico, en control de estabilidad, control autónomo y control remoto a larga distancia. Sin embargo, en la década de 1900 emergió el desarrollo de los vehículos aéreos no tripulados, con el uso de giroscopios

Durante la década de 1920, Gran Bretaña mostró interés por los vehículos aéreos no tripulados, desarrollando un monoplano no tripulado conocido como LARNYX (Long range gun with Lynx engine).

Una de las décadas más trascendentes fue la década de los 60 durante la guerra fría, gracias a los aviones militares impulsados con sistemas de propulsión a reacción, provocando un mayor interés en el control así como el monitoreo a distancia.

Después de los 60, fueron evolucionando en gran medida, logrando en esta misma década, contar con cámaras, capaces de reconocer territorios enemigos, además de mejoras en los sistemas de control, monitoreo y control a distancia.

Posteriormente en las siguientes décadas hasta el siglo XXI, los vehículos aéreos no tripulados ya contaban con GPS, cámaras infrarrojas, sensores más eficaces, mayor autonomía, radares de apertura sintética (SRA) entre otras mejoras.

En la actualidad uno de los problemas de interés en el área de vehículo no tripulado es el de poder controlar tanto la altitud como su posición del mismo; ya que esto influye en su desempeño de trabajo. Existen infinitud de controladores que realizan esta actividad como pueden ser, controladores PD (Proporcional, derivativo), PID (Proporcional, Integral, Derivativo), FUZZY (Difuso) etc.

1.2 Objetivo General

Diseñar un controlador clásico que sea capaz de posicionar un Cuadricóptero, en dos ejes (Eje X y eje Y) y además que permita seleccionar la posición deseada y ver su desempeño en una computadora personal (PC), por medio de una interface gráfica de manera inalámbrica, así como adquirir experiencia en el área de control aplicada a equipos aéreos no tripulados, principalmente en Cuadricópteros.

1.2.1 Objetivos particulares

- Diseñar una base o estructura que permita el libre movimiento en dos de sus ejes (Eje X y eje Y), permitiendo parcialmente la simulación de un Cuadricóptero en vuelo.

- Seleccionar el material necesario con las características adecuadas para poder llevar a cabo la construcción y control del Cuadricóptero (por ejemplo el microcontrolador, sensor, motores, hélices etc.).
- Realizar una comunicación inalámbrica entre la PC y el Cuadricóptero por medio del puerto serie utilizando Bluetooth.
- Diseñar una interface gráfica en la PC para la comunicación con el Cuadricóptero, que permita seleccionar la referencia de posicionamiento y al mismo tiempo poder visualizar el comportamiento del Cuadricóptero diseñado.
- Diseñar un controlador clásico PID para mantener la posición deseada en los ejes X – Y; además de comprobar su desempeño aplicado a un Cuadricóptero.
- Mencionar algunos de los principales problemas que se presentan en el desarrollo del Cuadricóptero. Así como sus posibles soluciones o recomendaciones para evitar problemas posteriores en la construcción y diseño del controlador.

1.3 Justificación

La necesidad de obtener la experiencia en el diseño de controladores PID aplicado a Cuadricópteros, además de conocer las partes principales por la cual está conformado un Cuadricóptero así como su funcionamiento, es una parte importante que motiva a la realización de este trabajo. Así como fomentar una nueva área de estudio en control donde esta pueda ser integrada en laboratorios para su estudio y ejemplificación de la aplicación del control y diseño de controladores.

1.4 Metodología

Para llevar a cabo este trabajo primeramente se realiza una investigación del funcionamiento de un Cuadricóptero. Posteriormente se identifican cada una de las partes que conforman el sistema. Para este trabajo se debe de diseñar una base para poder fijar el Cuadricóptero y que este tenga movimientos, en los ejes que se pretende aplicar el control de posición (X-Y). Posteriormente se seleccionan los componentes básicos que se utilizan un Cuadricóptero como por ejemplo, los motores, hélices, sensores, microcontroladores etc. Sus características pueden depender uno de otro, es decir por el peso puede cambiar el motor o mecanismo, el tipo de hélices y hasta el microcontrolador o sensores. Por otra parte, se diseña la interface gráfica que se desea utilizar, la cual puede depender de la compatibilidad entre el microcontrolador y el programa utilizado en la PC para realizar la interface.

Cuando se logra el objetivo general, así como los objetivos particulares, se comprueba el funcionamiento del sistema de posición aplicado al Cuadricóptero, sometiénolo a diferentes pruebas y obteniendo su respuesta para cada prueba realizada.

1.5 Descripción de los capítulos

En el Capítulo 1, se presenta la introducción del tema de tesis, el motivo de la investigación, un poco de los antecedentes. También se muestra el inicio de la aviación no tripulada, así como las razones por las cuales es importante el control de posición de un Cuadricóptero.

En el capítulo 2, se presentan los principales tipos de vehículos aéreos que existen y de manera general se describe que es un Cuadricóptero, cuáles son las partes que lo conforman, los movimientos que puede realizar un Cuadricóptero, permitiendo facilitar el manejo y control de los motores (sentido de giro de los motores y lógica de vuelo), con el fin de lograr el posicionamiento.

En el capítulo 3, se presenta el diseño de la estructura del Cuadricóptero en el cual se montan todos los elementos que lo conforman. Se describe el manejo de cada uno de los componentes, como por ejemplo el microcontrolador, sensor de posición, motores, hélices etc. Además del diseño del controlador he implementación con el uso del microcontrolador (Arduino) incluyendo la comunicación vía Bluetooth, diseño y manejo de la interface gráfica.

En el capítulo 4, se presenta los resultados del controlador diseñado, se realizan distintas pruebas para comprobar el buen funcionamiento del control y posicionamiento del Cuadricóptero, algunas de la pruebas son, comparación para diferentes ángulos en condiciones distintas, es decir con o sin perturbaciones.

En el capítulo 5, se presenta las conclusiones sobre el tema tratado. Además de algunas sugerencias para trabajos futuros que se pueden tomar en cuenta, en base a los problemas que se presentaron en el desarrollo del trabajo y así evitar problemas similares.

Capítulo 2

Estado del arte

2.1 Tipos de vehículos aéreos

Los vehículos aéreos pueden calificarse, en función del principio físico que produce su sustentación, es decir estos pueden clasificar en dos tipos los cuales son los Aerostatos y Aerodinos.

Los aerostatos son naves más ligeras que el aire, caracterizadas por contener un fluido de menor densidad que el aire, como hidrogeno o helio, como se muestran en la Figura 2.

Los primeros vehículos de este tipo que fueron desarrollados, en esta categoría se engloban los dirigibles y los globos aerostáticos.



Figura 2. Imagen de algunos aerostatos.

Los aerodinos son vehículos aéreos que son mucho más pesados que el aire, en donde estos para lograr su elevación producen fuerzas aerodinámicas a través de las alas.

Además de ello, este tipo de vehículo aéreo se clasifica según su tipo de alas que poseen para producir su sustentación.

Estos se clasifican en vehículos aéreos de alas fijas y alas giratorias. Los vehículos de alas fijas son aquellos aerodinos, en los cuales las alas se encuentran fijas, unidas con el resto del vehículo.

Entre los aerodinos de ala fija se encuentran los aviones, planeadores, ala deltas, parapentes, paramotores y ultraligeros.

Mientras que los vehículos aéreos de alas giratorias. Son aquellos aerodinos en los cuales las alas no están fijas al cuerpo del vehículo aéreo. A este tipo de alas también se les llaman palas. En donde más de una pala constituye una hélice, su característica de estas es que giran alrededor de un eje montado sobre el cuerpo del vehículo aéreo, consiguiendo de este modo obtener su sustentación.

Entre los aerodinos de alas giratorias se encuentran los autogiros, girondinos, helicópteros, combinados, convertibles y los Cuadricópteros.

Unos de los vehículos aéreos más trascendente en los últimos años son los no tripulados, por el uso que se les ha dado en aplicaciones civiles, comerciales, de logística etc. Un vehículo aéreo no tripulado también conocido como UAV (Unmanned Aerial Vehicle), es un vehículo autónomo capaz de volar sin la supervisión de un piloto humano a bordo. En el ámbito militar se han experimentado grandes avances, en aplicaciones civiles no es la excepción como detección de incendios entre otras aplicaciones, aun un poco deficientes

Este tipo de vehículos no tripulados se clasifican en seis principales categorías:



Figura 3. Cuadricóptero de uso comercial

1. **De blanco:** Su función consiste en simular aviones o ataques enemigos para los sistemas de defensa de tierra o aire.
2. **Reconocimiento:** Recolecta información que permita el reconocimiento de un área.
3. **Combate:** Se encarga de realizar misiones peligrosas.
4. **Logística:** Transporta cargas (Herramientas u material etc.).
5. **Investigación y desarrollo:** Para facilitar la investigación y experimentación de nuevos sistemas y tecnologías.
6. **UAV comerciales y civiles:** UAV diseñados para servir a propósitos civiles.

2.2 Estructura general de los vehículos aéreos

El Vehículo de alas giratorias, tienen un gran ventaja en comparación de los vehículos aéreos de alas fijas, ya que tienen la capacidad de mantener el vuelo en un punto fijo y tiene unas prestaciones de vuelo aceptable es decir, tiene la posibilidad de elevarse desde un punto fijo como de girar en ese mismo punto, así como la capacidad de trasladarse en diferentes direcciones con facilidad.

Los primeros vehículos aéreos de este tipo fueron los helicópteros y con el tiempo se logró la invención de los Cuadricópteros. Uno de los primeros diseños fueron los helicópteros tipo monorotores, los cuales solo disponen de un solo rotor como se muestra en la siguiente figura.(Figura 4).



Figura 4 Vehículo aéreo tipo monorotor es decir, de un solo rotor

Posteriormente se continúa con el desarrollo de helicópteros birotores. Un ejemplo son los helicópteros con dos rotores que tienen un eje propio, a este tipo se le conoce como rotores coaxiales, como el que se muestra en la Figura 5, donde ambos rotores están montados sobre un mismo eje de giro.



Figura 5. Birotores tipo Coaxial

Por otro lado también existen helicópteros birotos conocidos como helicópteros tipo BLL y los Tándem.

Los tipo BLL o también conocidos como birotos lado a lado, constan de un rotor en cada extremo lateral estos pueden ser tipo engranado (sincronizados) es decir; ambos rotores laterales tienen una distancia muy cercana uno del otro, causando que estos tengan que ser sincronizados para que las palas (hélices) no choquen entre ellas como se muestra en la Figura 6.



Figura 6 Aeronave con birotor tipo BLL engranado

Mientras que eso no sucede con los helicópteros birotos no engranados gracias a que se mantiene una distancia entre palas (hélices) evitando que estas colisionen entre ellas, por esta razón no es necesario la sincronización de las palas como se muestra en la Figura 7.



Figura 7 Aeronave con birotor tipo BLL no engranado

Los helicópteros tipo Tándem a diferencia de lo BLL tienen un rotor ubicado en la parte delantera y otro en la parte trasera como se muestra en la Figura 8.



Figura 8 Birotor tipo tándem

Unos de los helicópteros menos conocidos son los helicópteros trirotores que constan de tres rotores como se muestra en la Figura 9. Este tipo de helicópteros solo han sido construidos como prototipos, por la complejidad que se presenta en el control de su estabilidad y manejo.



Figura 9. Primer Tricoptero.

Llegando así a la construcción de vehículos aéreos de cuatro rotores, también conocidos como Cuadricópteros (Figura 10.).



Figura 10. Imagen de un Cuadricóptero.

Estos constan de cuatro rotores y una estructura en forma de cruz o forma de X [2] comúnmente. Este tipo de vehículos aéreos, son los más utilizados de forma que no se necesite llevar consigo un piloto a bordo y es aquí cuando se presenta una amplia aplicación en el área de control.

Además la utilidad en un Cuadricóptero de poseer un control de su posición, es facilitar el uso y su aplicación en distintas área o necesidades como por ejemplo en el uso agrícola, para el monitoreo de grandes hectáreas, aplicación de químicos tóxicos, otro ejemplo es el uso en vigilancia, también es utilizado para realizar tomas de video o fotografías panorámicas y muchas más aplicaciones según las necesidades y posibles aplicaciones.

2.3 Principales movimientos realizados por los vehículos aéreos

Los vehículos aéreos tipo aerodinos, como se mencionó anteriormente, necesitan ser impulsados para suspenderse en el aire.

Es necesario tener un poco de conocimiento de los movimientos o giros que un Cuadricóptero, deben realizar para poder elevarse mantenerse en un punto fijo o simplemente realizar un desplazamiento por los aires.

Para poder llevar a cabo los diferentes movimientos en el aire se deben realizar tres principales modos o tipos de movimientos. Estos son de posición, elevación y traslación.

2.3.1 Movimiento de posición

El movimiento de posición tiene la función de mantener la estabilidad del Cuadricóptero.

Es decir el movimiento de posición permite mantener a un Cuadricóptero estable en una posición deseada, por ejemplo este es muy utilizado cuando se desea mantener al Cuadricóptero en el aire estable en punto fijo. Esto es posible gracias al control utilizado en cualquier tipo de Cuadricóptero, comúnmente se utiliza un controlador PID, aunque también puede utilizar cualquier otro tipo de controlador. En el caso de la investigación se realiza el control de estabilidad de un Cuadricóptero que se encuentra montado sobre una base, permitiendo simular su vuelo, cómo la que se muestra en la Figura 11.



Figura 11 Base para simulación de vuelo y control del Cuadricóptero diseñado

Este movimiento además de que es el encargado de mantener la estabilidad en una posición deseada, es el principal y más importante porque de este movimiento depende el movimiento de elevación así como el movimiento de traslación.

Sin el control para el movimiento de posición es complicado mantener la estabilidad al tratar de elevarse o desplazarse, por esa razón es importante el control para mantener la posición.

2.3.2 Movimiento de elevación

El movimiento de elevación así como su nombre lo describe, es el encargado de elevar el Cuadricóptero es decir; controlar la elevación así como el descenso del mismo en una posición ya sea de manera vertical o transversal (en diagonal).

Por ejemplo si se desea elevar de manera vertical, es necesario realizar el aumento de la velocidad en los cuatro motores. En cambio para el descenso simplemente se disminuye la velocidad de los motores.

Mientras que para poder elevarlo de manera transversal es necesario aumentar la velocidad en los cuatro motores, tomando en cuenta que un par de ellos deben de tener mayor velocidad que el otro par y al mismo tiempo ir aumentando la velocidad de los cuatro motores. Esto de manera ideal porque siempre existirá una pequeña diferencia por la dinámica de los motores así como también esa diferencia se puede ver afectada por la distribución de los componentes es decir; un motor o más de uno necesiten mayor velocidad por el peso, este obliga al control a aumentar la velocidad y es ahí cuando puede existir una diferencia.

Lo importante es mantener la lógica de vuelo, que giros deben realizar sus hélices, cuales motores deben actuar para cierto movimiento etc.

Esto para cuando se utiliza la configuración en X que es lo más común. Por ejemplo en esta parte influye el control de estabilidad, para mantener la inclinación ante perturbaciones en el momento de su elevación, porque los motores pueden diferir por su dinámica o simplemente porque la distribución de los componentes no está equilibrada.

2.3.3 Movimiento de traslación

Por ultimo tenemos el movimiento de traslación. El cuál es el encargado de realizar el control de inclinación para poder realizar el desplazamiento del Cuadricóptero de manera horizontal en cualquier dirección.

Esta parte es muy similar a la anterior la diferencia que existe es cuando el Cuadricóptero se encuentra suspendido en el aire este simplemente, disminuye la velocidad de dos motores y aumenta la velocidad del otro par de motores sin aumentar gradualmente la velocidad de ello. Es decir solo se inclina poco para lograr su desplazamiento, sin variar demasiado la velocidad de cada par de motores.

A diferencia de cuando se desea elevar de manera transversal, se inclina el Cuadricóptero, pero al mismo tiempo aumenta de manera proporcional la velocidad de los cuatro motores y el cambio de velocidad es muy notorio.

Al igual que en los movimientos de posición y elevación. Para poder llevar a cabo el desplazamiento se debe de controlar la aceleración de los cuatro motores de manera individual.

2.4 Giros para lograr las posiciones deseadas

Para el control de posiciones del Cuadricóptero es necesario controlar la velocidad de los motores en pares o de manera individual, eso dependerá del tipo de configuración utilizada, si la configuración es en X o en cruz, aunque lo más recomendable es controlar los motores de manera individual.

Para entender la lógica de vuelo o posicionamiento de un Cuadricóptero se debe de tomar en cuenta características de vuelo que son importantes para poder implementar el control.

En primer lugar es necesario contar con la fuerza suficiente para mover el Cuadricóptero, esto dependerá del tipo de motores utilizados, hélices y el peso de la estructura.

Después solo es cuestión de variar la velocidad de los motores por ejemplo, si supone que el Cuadricóptero se encuentra inclinado con un ángulo en el eje $X = 45^\circ$ y uno en $Y = 45^\circ$.

Ahora para llegar a $X=0$ y $Y=0$ es necesario aumentar la velocidad de dos motores y al mismo tiempo disminuir la velocidad de los otros dos motores, esto se lleva a cabo gracias al controlador.

La señal de control para cada motor dependerá del grado de inclinación, es decir la señal dependerá de la lectura del sensor que nos proporciona la posición angular del Cuadricóptero.

Así de una manera similar funciona cuando se encuentra en una posición distinta y se desea llevar a otra, con solo disminuir o mantener la velocidad de los motores. También va depender la configuración seleccionada del Cuadricóptero ya sea la configuración en X o en la configuración en cruz. [3]

Otro punto muy importante que se debe de tomar en cuenta para poder llevar a cabo el posicionamiento y control del Cuadricóptero es el sentido de giro de los motores, es decir un par de ellos debe de tener un sentido de giro horario y el otro par de motores restantes un sentido de giro anti-horario.

Esto se debe a que se debe tener un equilibrio, por ejemplo si todos los motores giraran en un solo sentido, las fuerzas generadas por las hélices causarían que este girara en un sentido circular, mientras que si las fuerzas generadas por las hélices se distribuyen de manera equitativa las fuerzas se contrarrestan logrando mantenerlos sin giros.

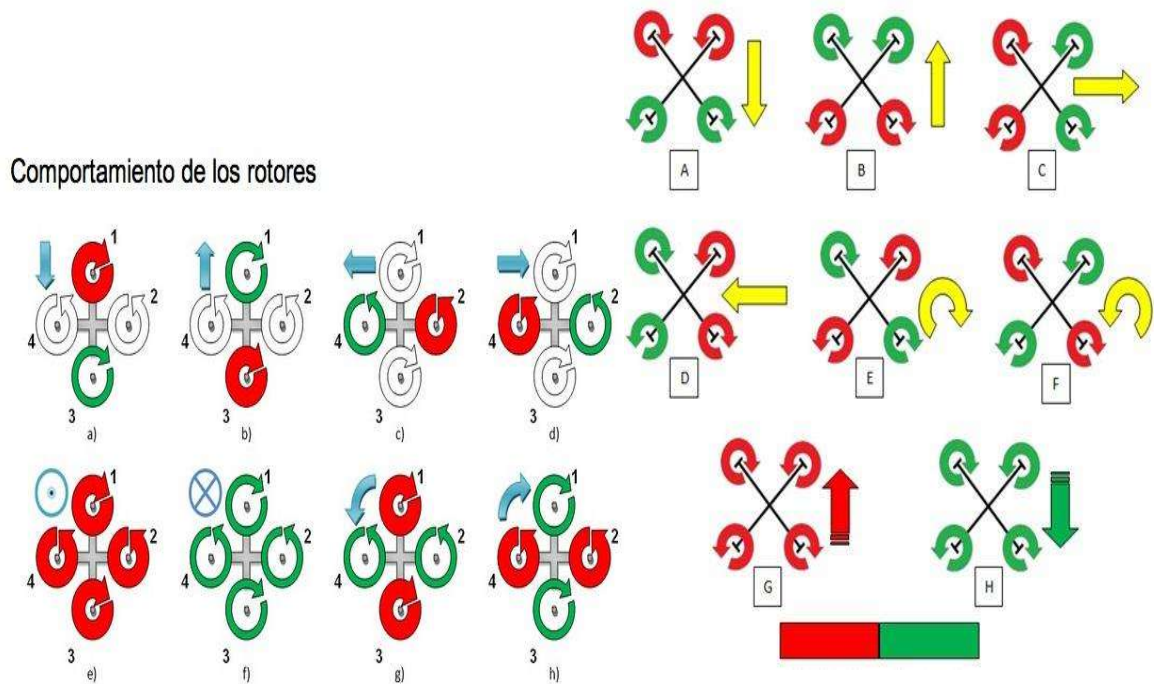


Figura 12 Característica de cómo es necesario variar la velocidad para, lograr el movimiento deseado. En la configuración en cruz como en X, respectivamente.

Además de seleccionar las hélices adecuadas, esto significa que se seleccionan dos hélices para los motores que giran en sentido horario y dos para los motores que giran en sentido anti – horario, cada par de hélices tienen un diseño que permiten impulsar a el Cuadricóptero para lograr el posicionamiento y desplazamiento del mismo.

Por ejemplo en la Figura 12 se observa que para lograr un movimiento siempre, es necesario realizar cambios de velocidad y esto depende del ángulo en donde desea posicionar el Cuadricóptero.

2.5 Componentes electrónicos

Los componentes electrónicos pueden variar según sea el diseño o aplicación de un Cuadricóptero, ya que este puede ser de uso militar, de blanco, de logística, de combate, de carreras o de uso civil etc. Y para cada tipo puede estar conformado con más o menos equipo electrónico.

Por ejemplo, el Cuadricóptero de la Figura 13 puede entrar en la categoría de uso civil, por solo contar con lo básico como son motores, drivers, sensores de posicionamiento y Microcontroladores etc, donde todos estos componentes electrónicos logran conjuntamente la dinámica y control de un Cuadricóptero.

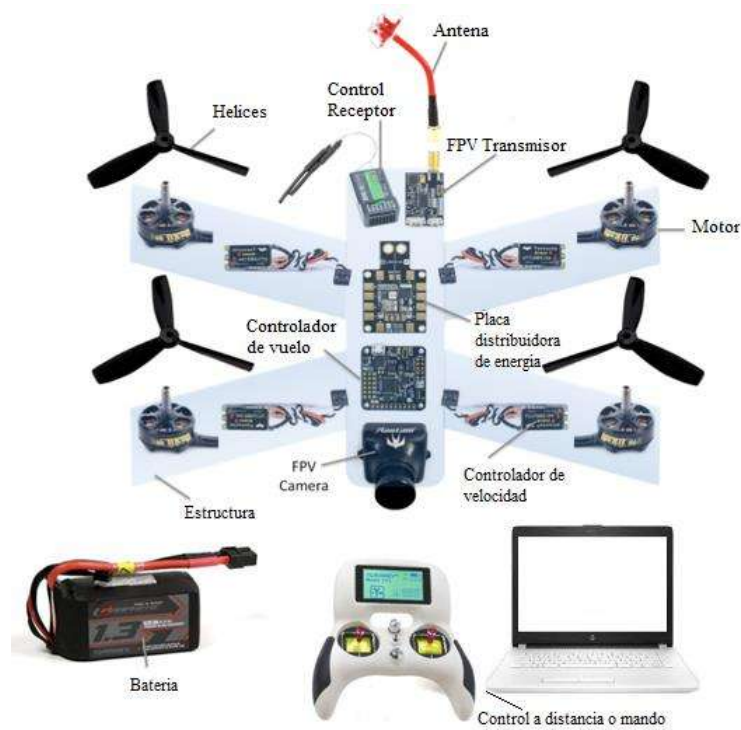


Figura 13. Componentes de un Cuadricóptero.

2.5.1 Motores

Los motores son una de las partes necesarias e indispensables para el funcionamiento de los Cuadricópteros.

Para la construcción de un Cuadricóptero puede utilizarse distintos tipos de motores de los cuales existen principalmente dos tipos, como los que se muestran en la Figura 14. Los motores de CD (Corriente directa) o motores Brushless.

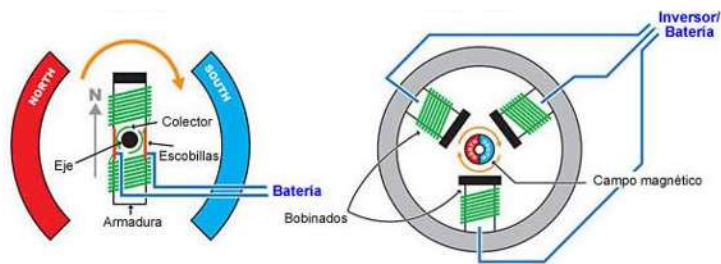


Figura 14. Se muestra el diagrama de un motor de CD y un motor tipo Brushless, respectivamente

En el caso de los motores de CD para el desarrollo y construcción de un Cuadricóptero son los más utilizados tanto por sus características de fácil manejo, bajo consumo, economía entre

otras características que lo hacen útil. Por ejemplo los motores de CD son fáciles de adquirir, existen variedad de modelos, tamaños etc.

Por otro lado existen motores trifásicos más complejos, con mayor consumo de corriente, y como resultado de ello más difíciles de manejar o controlar su velocidad, estos son los motores Brushle.

Este tipo de motores son menos económicos además que es necesario un circuito extra que es utilizado para su regulación y manejo de corriente, así como para su control, esto causa que sea mucho más complejo el algoritmo de control. Por lo que es recomendable utilizar para este trabajo motores de corriente directa.

Algunos de los modelos que son utilizados para pequeños Cuadricópteros son los motores tipo syma X5C, X55, X55C y X5SW los cuales son muy similares. Este tipo de motores son ideales ya que cuentan con un tamaño muy pequeño que lo hacen útiles para reducir el peso del Cuadricóptero, además cuentan con una alta velocidad otra de las ventajas de este tipo de motores es que utilizan voltajes muy pequeños, comúnmente se utilizan voltaje de 3.7V en adelante según el tamaño del motor y sus características [4]

2.5.2 Tipos de Hélices

La hélice son las encargadas de generar un impulso mediante su giro moviendo el aire. Esta mientras mayor aire consiga mover mayor será la fuerza de empuje que logran.

En un Cuadricóptero es necesario tener un par de hélices horarias y otro par de hélices anti-horarias, ya dos de sus motores giran en un sentido horario mientras otros dos de ellos giran en sentido contrario (anti-horario), por esta razón es necesario contar con dos pares de hélices diferentes. Además de que esto permite una distribución de las fuerzas generadas por las hélices. En caso contrario si las cuatro hélices son del mismo tipo al girar en sentido horario, las cuatro fuerzas causarían que el Cuadricóptero intente girar en su propio eje Z (YAW) o realizar movimientos no deseados.

Para poder identificar algunas de las principales características de una hélice, como tamaño, grado de inclinación y número de palas, se pueden identificar con un par de números y uno extra que nos indica el número de palas que la hélice contiene.

Por ejemplo 5x3x3 o 50x30x3, en donde el primer número nos indica el diámetro de la circunferencia que genera cuando gira, mientras que el segundo número nos indica el grado de inclinación de las palas y el último número nos indica la cantidad de palas que la hélice posee.

También debe de tomarse en cuenta que un grado de inclinación muy grande, puede proporcionar un impulso muy bueno, pero en consecuencia este causara un consumo excesivo

de corriente y en el caso de ser un motor muy pequeño este podría dañarse por el exceso de corriente consumida.

Otras de las características de hélices, además del grado de inclinación es la forma de la hélice, existen tres principales tipos de hélices: Las acabadas en punta, las Bullnose y las híbridas.

Las acabadas en punta Figura 15 como su nombre lo dice, son las que su terminación en los extremos de las hélices son en forma de punta. Esta forma nos asegura una buena velocidad y poco consumo de corriente.



Figura 15. Hélices acabadas en punta

Mientras que las hélices tipo Bullnose se caracterizan por que su forma a diferencia de las acabadas en punta, sus extremos terminan en un corte plano como las que se muestran en la Figura 16, estas nos aseguran mayor impulso pero en consecuencia mayor consumo de corriente.



Figura 16. Hélices tipo Bullnose

Por ultimo las híbridas su forma se encuentra entre las dos mencionadas anteriormente, no son tan finas pero tampoco son tan planas, como las que se muestran en la Figura 17 por lo que estas pueden ser una buena opción para probar inicialmente en un Cuadricóptero. [10]



Figura 17. Hélices tipo híbridas

2.5.3 Drivers

Los drivers permiten al microcontrolador manejar motores de mayores amperajes y tensión comparados con los utilizados por el microcontrolador.

En el caso de los motores de corriente directa se utilizan comúnmente los puentes H o simplemente transistores.

Un Puente H es un circuito electrónico que se utiliza para manejar el sentido de giro así, como para regular la velocidad de un motor de CD.

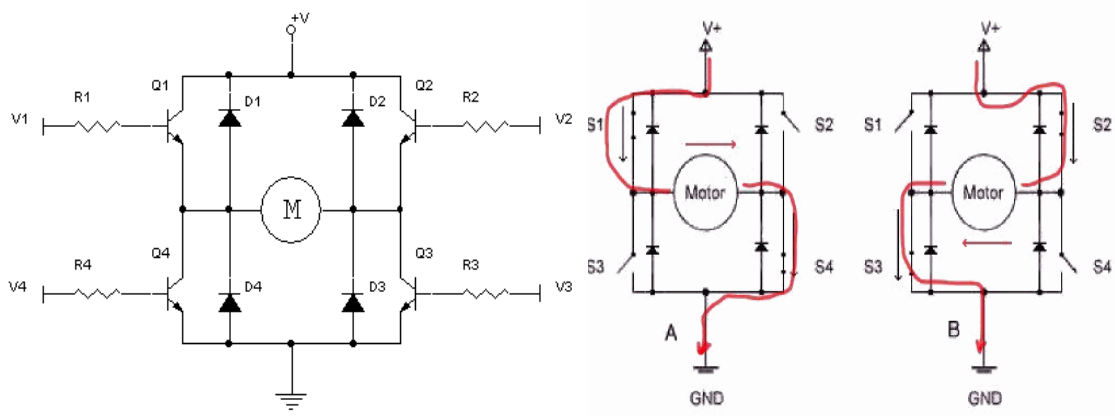


Figura 18. Diagrama de un puente H

Para controlar el sentido de giro basta con invertir la polaridad del motor, para ello es posible gracias al arreglo del circuito de la Figura 18. Por ejemplo para poder realizar el giro en un sentido basta con cerrar el interruptor s1 y s4 mientras que s2 y s3 se mantienen abiertos, en caso que el giro sea sentido contrario los interruptores s2 y s3 se cierran, mientras que s1 y s4 se mantienen abiertos. Para realizarlo de manera electrónica se puede utilizar un arreglo de transistores.

Además se tienen que realizar combinaciones coordinadas en pares, para aplicar una tensión en la base del transistor y así poder cambiar el sentido de giro de un motor de CD. [5]. En la Figura 18 se muestra diagramas básicos de un puente H.

Los elementos para utilizar como interruptores (switches) pueden ser relevadores, transistores o utilizar circuitos integrados. Esto dependerá de las necesidades u aplicación.

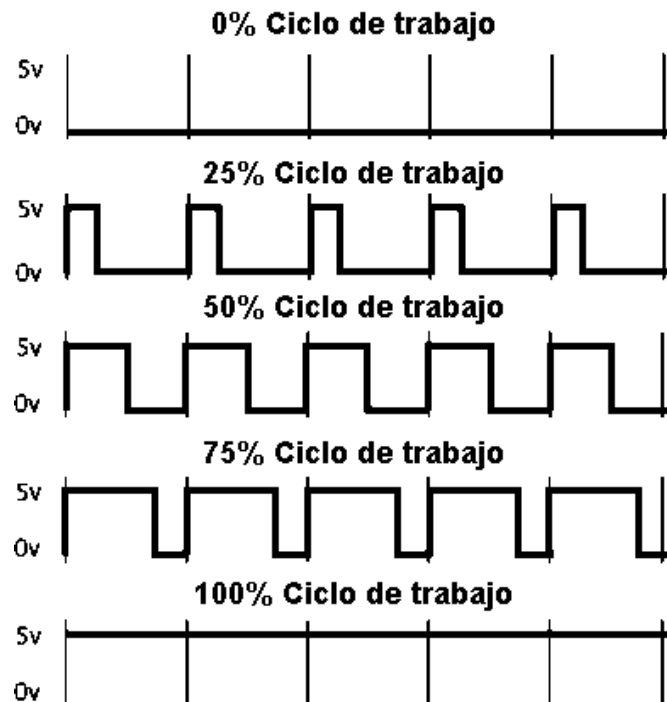


Figura 19 Señales PWM según su ciclo de trabajo

Para control de velocidad del motor, se utilizan transistores y una señal PWM (pulse width modulation). En donde se varía el ciclo de trabajo, es decir se controla el tiempo en alto y tiempo en bajo para lograr el control de la tensión aplicada los transistores como se muestra en la siguiente Figura 19.

En cambio para los motores Brushless son utilizados los controladores electrónicos de velocidad ESC (Electronic Speed Controller), como el que se muestra en la Figura 20.

El ESC consta comúnmente con tres terminales en un extremo, mientras que por el otro comúnmente son cinco terminales.

Por ejemplo en la imagen en lado izquierda:

- Cable rojo y negro: es utilizado para la alimentación del motor Brushless. En donde se conecta la batería o fuente de CD.
- En el conector que es de tres pines:
 - Amarillo: señal PWM.
 - Rojo y negro: Cable rojo positivo (CD: 5.5v-22.2v) y cable negro negativo (gnd).

En el extremo derecho:

- Los tres cables son, para alimentar el motor trifásico Brushless.



Figura 20. Controlador de Velocidad ESC

Para el control de velocidad, el ESC recibirá una señal PWM de 50 Hz y dependiendo de la longitud del ancho de pulso entregara más o menos potencia al motor. [6]

2.5.4 Sensor de Posición

Los sensores de posición o sensores dinámicos constan de un acelerómetro y un giroscopio que está juntos en un mismo circuito integrado aunque también existen por separado.

La diferencia en un circuito que cuenta con un acelerómetro y un giroscopio es, que mientras el acelerómetro detecta la aceleración en los tres ejes Roll, Pitch y Yaw (Alabeo, Cabeceo y Guiñada) correspondientes al eje X, Y y Z respectivamente (Figura 21.). El giroscopio se encarga de detectar la derivada de los giros y así obtener la velocidad de los giros en los tres mismos ejes.

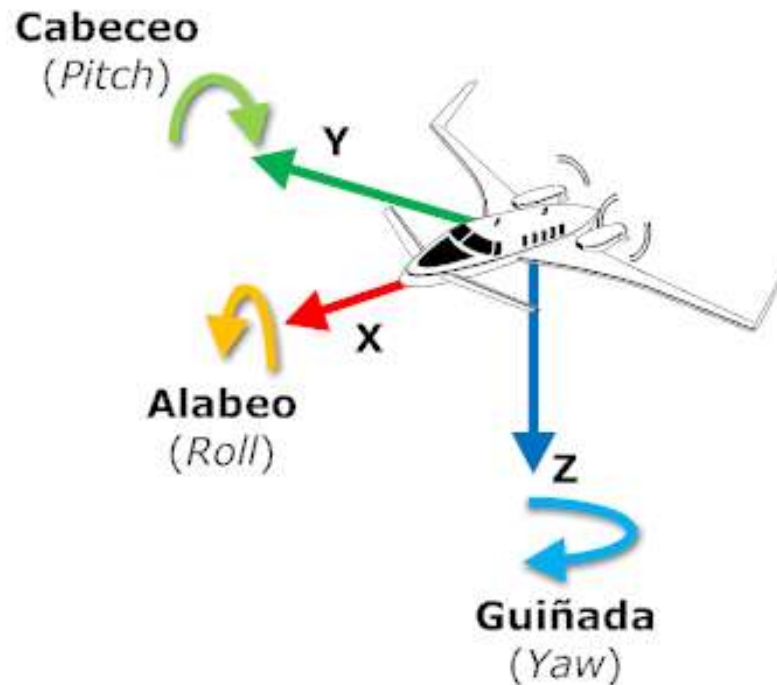


Figura 21. Posición de giro de un vehículo aéreo

El acelerómetro es el encargado de detectar las aceleraciones en los tres ejes los valores son devueltos en variaciones de tensión por el bus I²C. El funcionamiento consiste en utilizar pequeños sensores piezoeléctricos para así poder obtener las variaciones de aceleración en cada eje.

El giroscopio es capaz de medir la velocidad de rotación en los tres ejes, gracias al principio de conservación del momento angular.

De la misma manera que el acelerómetro, el giroscopio presenta cambios de tensión proporcionales a la velocidad de rotación, los cuales dependen de la inclinación los cuales son detectados por sensores capacitivos que finalmente son enviados por el bus I²C. [7]

Por lo tanto es necesario contar con un sensor capaz de brindar ambas lecturas, logrando tener mayor precisión y por lo tanto tener en nuestro caso un mejor control.

2.5.5 Microcontroladores

Un microcontrolador es un circuito integrado, en su interior posee la arquitectura de una computadora, es decir cuenta con un CPU, Memoria RAM, EEPROM y circuitos de entrada y salida entre otros, componentes que se integran y dependen del tipo de fabricante u modelo de microcontrolador. Como los que se muestran en la Figura 22 [8].

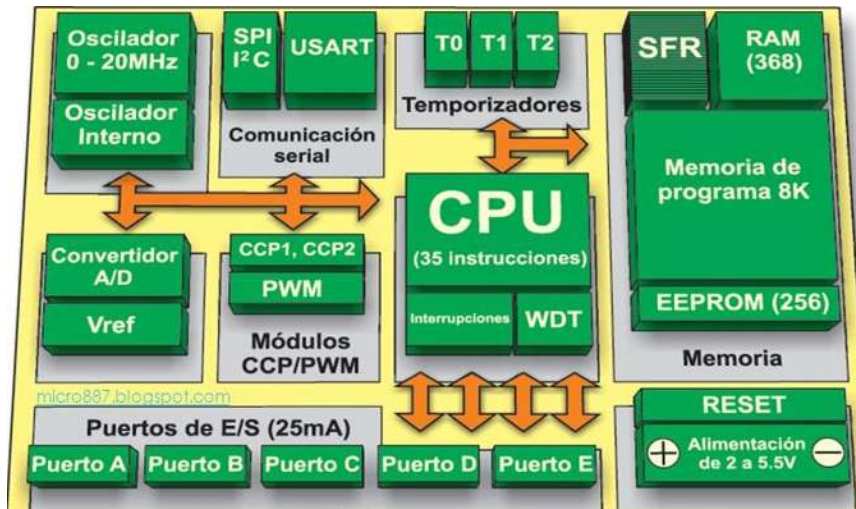


Figura 22. Arquitectura de un Microcontrolador

La elección de un microcontrolador es muy importante a la hora de implementar un controlador para un Cuadricóptero, por ejemplo la velocidad en la que se procesan los datos, los puertos de entrada y salida, consumo de energía, memoria, diseño de la placa (tamaño) entre otras características según el tipo de microcontrolador.

Los controladores más comunes son los Atmel, Microchip, Texas instruments, Intel, Hitachi entre otros. Entre los microcontroladores más utilizado para el control en un Cuadricóptero están los microcontroladores ATmegaxxx de Atmel así como lo de Microchip de gama alta de la familia 18Fxxxx o los procesadores de señales DSP30Fxxxx.



Figura 23. Microcontrolador ATmega 328 utilizado por la placa Arduino nano

Por ejemplo el microcontrolador de Atmel el ATmega 328 que se muestra en la Figura 23, es utilizado en diferentes placas por Arduino entre ellas el Arduino nano, Arduino uno, Arduino Mini y ArduPilot. La ventaja de este tipo de placas es que cuentan con una programación en C, así como los requisitos mínimos para poder implementar el controlador en un Cuadricóptero y llevar a cabo una comunicación entre el microcontrolador y algún dispositivo de monitoreo y mando como una PC o similar.

2.6 Comunicación inalámbrica

Al hablar de comunicaciones inalámbricas se entiende de manera general que son aquellas comunicaciones entre dispositivos (móviles o no móviles).

Dentro de las comunicaciones inalámbricas se encuentran la tecnología RF, Wifi, Bluetooth principalmente. Este tipo de tecnología inalámbrica son las comunes para la transmisión de datos en Cuadricópteros, cada una de ellas puede ser de utilidad, pero para ello se debe tener en cuenta cuales son las necesidades o requisitos para seleccionar la más adecuada. Es decir si no es necesario llevar a cabo una comunicación a larga distancia basta con utilizar una comunicación Bluetooth o en caso contrario puede utilizarse la tecnología RF o Wifi. Esto dependerá de las necesidades al llevar a cabo la construcción del Cuadricóptero. En este trabajo basta con la comunicación vía Bluetooth, ya que el Cuadricóptero se encuentra montado en una base para simular su vuelo y poder realizar el control en dos ejes, por lo que no es necesario una comunicación a larga distancia.

El Bluetooth es una especificación regulada por el grupo de trabajo IEEE 802.15.1, que permite la transmisión de voz y de datos entre diferentes dispositivos mediante un alcance de radiofrecuencia en la banda ISM de 2.4 GHz.

El Bluetooth permite conectar diferentes dispositivos electrónicos, como asistentes personales (PDA) como lo son celulares, ordenadores portátiles etc. Lo que facilita la interactividad entre diferentes dispositivos sin importar el fabricante de cada uno de esos equipos.

El Bluetooth puede ser de corto alcance de aproximadamente 10m y de hasta medio alcance con alcance mayor a 10m. En una red Bluetooth, cualquier dispositivo puede actuar como maestro o como esclavo es decir: el dispositivo maestro se encarga de definir como se establece la comunicación físicamente, mientras que el esclavo coordina sus transmisiones según las especificaciones del maestro. Normalmente el primero que solicita el servicio actúa como maestro, excepto cuando ya se ha establecido la conexión.

Para la elección del tipo de red inalámbrica, esto dependerá del tipo de aplicación así como las facilidades de manipular información según la necesidad del usuario.

En este capítulo se lleva a cabo la introducción de forma general para la construcción de un Cuadricóptero así como la misma base en la cual se monta el Cuadricóptero, para poder realizar la simulación del vuelo del mismo esto con la finalidad de poder aplicar el control a un Cuadricóptero.

Además de conocer algunos de los movimientos y giros necesarios que deben tomarse en cuenta para el diseño del control, Otra de las partes importantes que se presentan en el capítulo son algunos de los componentes básicos que conforman un Cuadricóptero así como el funcionamiento de cada uno de ellos, como por ejemplo los tipos de motores que pueden ser utilizados, hélices, drivers, sensores, microcontroladores etc. Así como la necesidad de llevar a cabo una comunicación inalámbrica, todo esto permite iniciar con el diseño y construcción del Cuadricóptero para llevar a cabo su control.

Capítulo 3

Hardware y software

En este capítulo se muestra la estructura del hardware y software implementado para el Cuadricóptero diseñado. Dentro del hardware esta la estructura seleccionada, los circuitos electrónicos y componentes necesarios como lo son motores, hélices, etapa de potencia sensor de posición, microcontrolador y dispositivos de comunicación de Bluetooth.

Mientras que el software este lo integran dos programas uno para la PC y otro para el microcontrolador. El programa de la PC esta implementado en lenguaje de java en un ambiente de programación de PROCESSING, mientras que el software del microcontrolador está programado en lenguaje C utilizando el ambiente de programación ARDUINO. En la Figura 24 se muestra la estructura del Cuadricóptero diseñado en donde se indican las partes que lo integran.

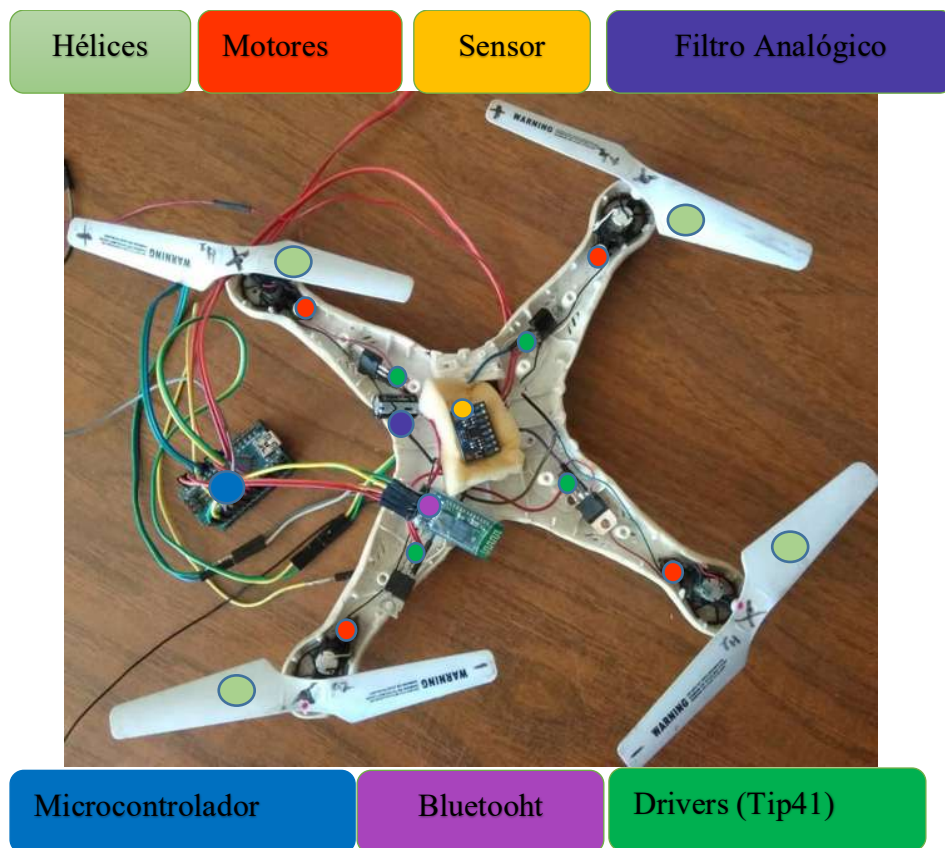


Figura 24 Estructura y sus partes del Cuadricóptero diseñado

3.1 Estructura

El estructura es la parte en donde van montados todos elementos de control electrónico y actuadores.

Para el diseño de la estructura debe de tomarse en cuenta el tamaño de los elementos que se montaran sobre la estructura (control electrónico, sensores, etc). Y también el peso de todos los elementos y sus componentes electrónicos. Esta es una de las partes que son importantes para lograr el objetivo.

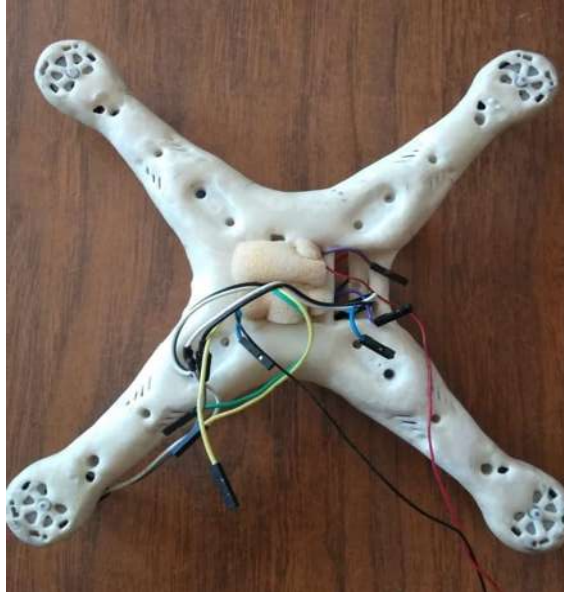


Figura 25 Estructura del Cuadricóptero para el montaje de los elementos electrónicos

Posteriormente se selecciona el tipo de material para la estructura, es recomendable utilizar una estructura, rígido que no trasmita o que sea tan sensible a vibraciones que afecten a los componentes, especialmente a los sensores que nos proporcionan lecturas de posición.

Una mala lectura ocasionada por las vibraciones y ruidos eléctricos-electromagnéticos muy perjudicial y pueden entorpecer el control del Cuadricóptero. Los materiales más comunes son la fibra de carbono, nylon poliamida, fibra de vidrio, plástico, aluminio, resina acrílica etc. En este caso es utilizado el plástico combinado con una capa de resina acrílica para endurecer la estructura. Además se utiliza la configuración en X. En la Figura 25 se muestra una fotografía de la estructura seleccionada con longitud de 19.3 X 19.3 cm la cual fue utilizada, para el desarrollo de este trabajo.

3.2 Motor de CD

El motor de CD se caracteriza por contar con escobillas, las cuales permiten realizar la conmutación (cambio de polaridad), y gracias a ello poder realizar el movimiento del rotor.

Su funcionamiento se puede explicar de manera simple: Una corriente eléctrica circula por un embobinado de hilos de cobre, es la parte móvil del motor (rotor), al circular la corriente atreves de este embobinado produce un campo magnético que se opone al campo generado por imanes permanentes que se encuentran dentro del motor sobre su carcasa. Estos campos

(el campo generado por la corriente que circula por el embobinado y el campo generado por los imanes permanentes) se rechazan entre sí, mientras sus polos se encuentren iguales. En cuanto se aproximan al punto en donde las fuerzas son de atracción en ese momento las escobillas realizan el cambio de polaridad y en consecuencia se repelen manteniendo el rotor girando mientras el conmutador realiza el cambio de polaridad y el embobinado es alimentado por una corriente eléctrica.

Este funcionamiento facilita controlar la velocidad del motor, solo con aumentar o disminuir la tensión, aplicando una señal PWM. Esta característica que facilita el control de velocidad en un motor de este tipo, es lo que contribuye a facilitar el algoritmo de control utilizado por el Cuadricóptero ignorando los ruidos eléctricos (armónicos).

Un problema a la hora de utilizar motores de CD, es la mala eficiencia a causa del rozamiento de las escobillas, por esta razón, para el uso en un Cuadricóptero es necesario utilizar filtros para la parte eléctrica, en cambio para la parte mecánica a veces es necesario utilizar un engranaje que permite aumentar el torque y en consecuencia también poder utilizar hélices de mayor tamaño. Algunas opciones para la elección de un motor es el elegir un motor *syma X5C*, *X55*, *X55C* y *X5SW*. Para este trabajo se utilizaron motores *syma X5C*, en donde sus dimensiones son de 8 x 20 mm, el diámetro del eje de 1 mm y son a 3.7 V normalmente además. Posterior a realizar la elección de los motores se continúa con la elección de las hélices.



Figura 26. Motor de CD *symaX5C*

3.3 Hélices Híbridas

La elección de las hélices como se mencionó en el capítulo 2 dependerá, del tipo de aplicación, en este caso se busca un consumo mínimo, por el tipo de motores que se utiliza para la construcción del Cuadricóptero y para ello el empuje debe ser el adecuado para el movimiento de la estructura y sus componentes, además de tomar en cuenta la capacidad de los motores.

Regularmente este tipo de Cuadricópteros utilizan hélices de tipo híbridas. Por qué se adaptan a la capacidad de los motores como a la necesidad del empuje necesario para mantener el Cuadricóptero en un punto deseado, pero no necesariamente el suficiente empuje. Esto depende mucho del tipo de motores utilizados. Como los que muestran en la Figura 27



Figura 27 Motor con sus respectiva hélice y engranaje para aumentar el torque

3.4 Etapa de potencia.

El objetivo de esta etapa es proporcionar energía necesaria para el manejo de los motores de CD, debido a que el microcontrolador no puede manejar corrientes altas para controlar la velocidad de cada motor. Como en este caso los motores de CD son motores de bajo consumo, se selecciona un transistor Tip41c de unión NPN. Este transistor de media potencia, viene en un encapsulado TO-220 como el que se muestra en la Figura 28.

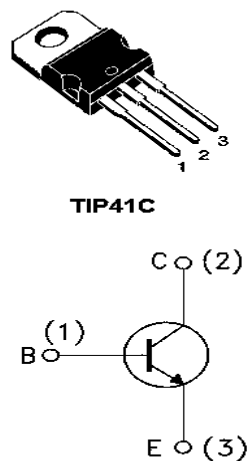


Figura 28 Transistor Tip 41c Utilizado como Driver por el Cuadricóptero

Este transistor soporta hasta 6A de corriente y picos de 10 A, potencia de 65W y temperaturas de 150°. En la Figura 29 se muestra un diagrama de cómo se utiliza el transistor para el manejo de un solo motor.

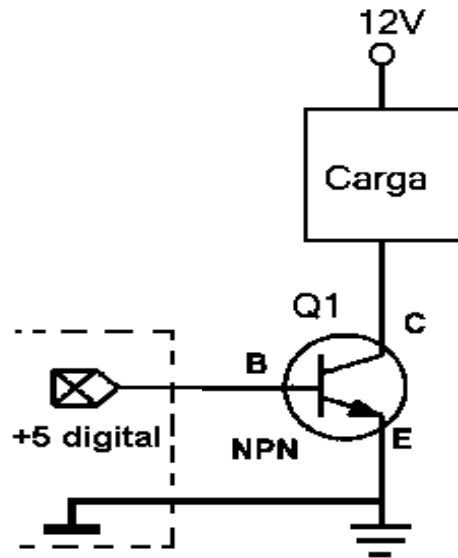


Figura 29 Diagrama de transistor tipo NPN

3.5 Sensor de posición (MPU6050)

Para el sensor de posición se utiliza un sensor MPU6050 que es un circuito integrado (Figura 30), que contiene un acelerómetro y un giroscopio en un mismo empaquetado, el cual tiene una resolución de hasta 16 bits lo cual puede ser dividido en 65536 fracciones, esto aplica para cada uno de los tres ejes X,Y y Z.

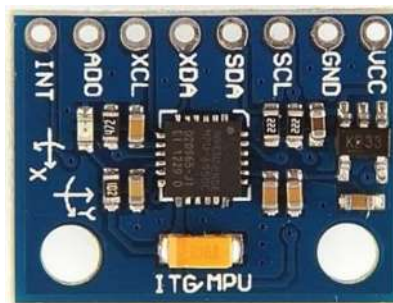


Figura 30. Sensor MPU 6050

El Giroscopio se encarga de medir el movimiento de rotación, es decir se encarga de medir la velocidad angular.

El funcionamiento del giroscopio se basa en el efecto Coriolis el cual consiste en la desviación de una masa desde un punto A hasta un punto B' sin llegar al punto B a causa de que la masa se encuentra en un movimiento circular como se muestra en Figura 31.

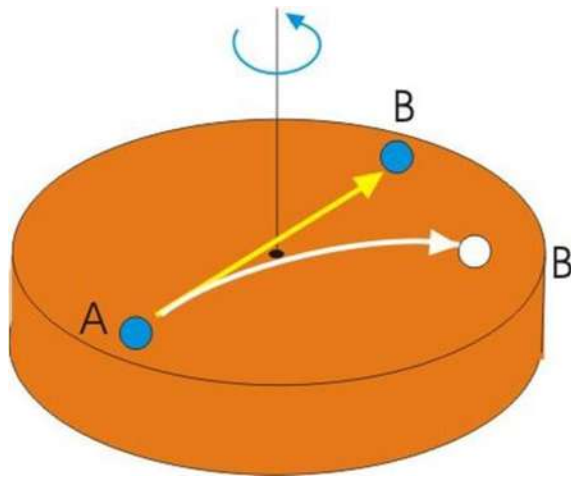


Figura 31. Efecto Coriolis.

De esta manera se utiliza un pequeño sensor para cada eje en el MPU6050.

El sensor utiliza una pequeña masa resonante, como se muestra en la Figura 32, que genera una pequeña tensión proporcional al desplazamiento angular.

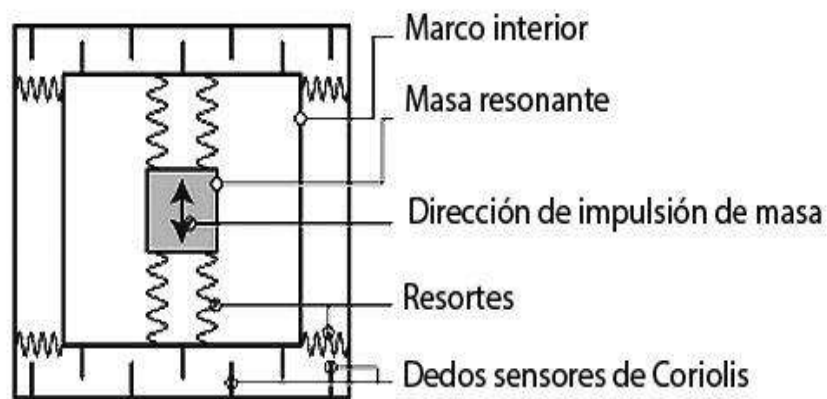


Figura 32 Sensor interno en el MPU 6050 (Giroscopio)

En el caso del Acelerómetro se encarga de medir la aceleración, inclinación vibraciones.

El sensor es capaz de detectar estos pequeños cambios gracias a su construcción, se basa en utilizar una pequeña lamina móvil la cual es sostenida por pequeños resortes, mientras que por otra parte se utilizan electrodos fijos, obteniendo una diferencia de potencial a causa de la separación entre el electrodo y la lamina móvil, como el que se muestra en la Figura 33.

Como resultado se obtiene una tensión proporcional a el desplazamiento del sensor. [11]

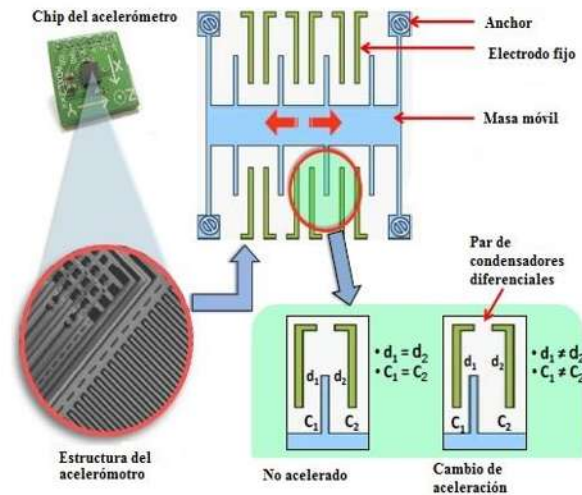


Figura 33 Sensor interno en el MPU 6050 (Acelerómetro)

3.6 Microcontrolador

Para el desarrollo de este trabajo se utiliza un microcontrolador de la familia ATMEL AVR implementado en una placa de desarrollo Arduino. La placa de desarrollo es la Arduino Nano que tiene el microcontrolador Atmega 328 la cual se muestra en la Figura 34. Esta plataforma de Arduino es libre de código abierto basada en el hardware y software, es programable en lenguaje C, lo cual permite procesos lógicos y matemáticos dentro de la placa, además de controlar y gestionar recursos de componentes externos conectados a la placa con la facilidad de programar y cargar el programa al microcontrolador de manera directa sin utilizar componentes extras, es decir con tan solo conectar un cable USB del microcontrolador Arduino hacia la PC.

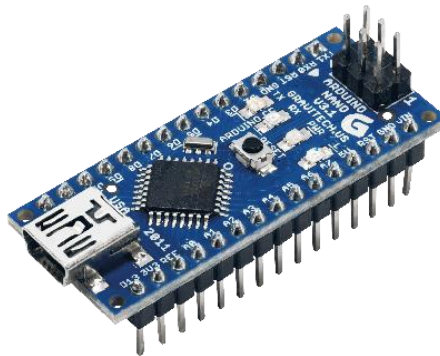


Figura 34. Placa Arduino nano

Algunas de sus principales características del microcontrolador Atmega328 son las siguientes:

- 32 Kbytes de memoria Flash
- 1 byte de memoria RAM
- Reloj de 16 MHz
- 13 pines de entradas y salidas digitales
- 6 pines con salida PWM
- 5 pines para entradas analógicas
- 6 pines para salidas analógicas
- Fácil de conectarse a una PC
- Corriente de CD en pines digitales de entrada y salida de hasta 40mA.
- Pin de 3.3V con corriente de hasta 50mA
- EEPROM de 512 Bytes

El fabricante Arduino no solo se encarga de proporcionar las placas, también proporciona un software de Entorno de Desarrollo Integrado IDE (integrated Development Environment) la principal característica del software IDE es que su entorno de programación ha sido empaquetado como un programa de aplicación, el cual consiste en un editor de código, un compilador, un depurador y un constructor de interface gráfica (GUI) como el que se muestra en la Figura 35. Además de contar con herramientas para cargar el programa en la memoria flash del microcontrolador y librerías

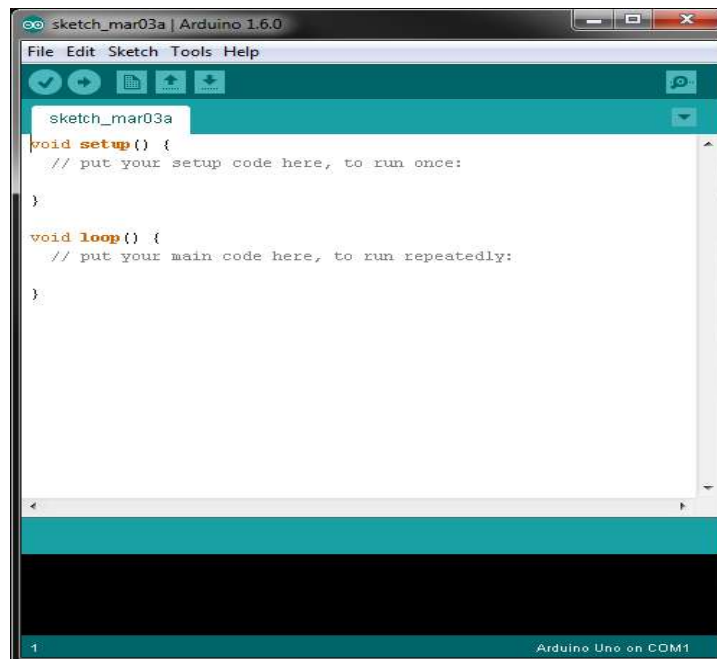


Figura 35. Imagen de software ID de Arduino

3.6.1 Dispositivos de comunicación Bluetooth

Para poder llevar a cabo la transmisión de datos entre el microcontrolador y la PC se utiliza una comunicación, para esto el microcontrolador se conecta a un módulo Bluetooth HC-05 que es caracterizado porque se puede utilizar como maestro o esclavo, es decir que además de recibir conexiones de desde una PC, Tablet, celular etc. También puede generar conexiones hacia otros dispositivos. Esto permite conectar dos módulos de Bluetooth y formar una conexión punto a punto entre dos microcontroladores o dispositivos. Así como el bajo consumo de corriente y la conexión a distancia.

El módulo HC-05 viene preconfigurado de la siguiente forma:

- Modo o rol: Esclavo
- Nombre por defecto: HC-05
- Código de emparejamiento: 1234
- Velocidad por defecto: 9600

El módulo HC-05 tiene 4 estados los cuales es importante conocer

- Estado desconectado
- Estado conectado o de comunicación
- Modo AT1
- Modo AT2

Estado Desconectado: Entra al estado Desconectado tan pronto es alimentado el módulo, y cuando no se ha establecido una conexión Bluetooth con ningún otro dispositivo. Además el led del módulo parpadea rápidamente.

Estado Conectado o de Comunicación: Entra a este estado cuando se establece una conexión con otro dispositivo Bluetooth y el led se encuentra en un doble parpadeo.

Todos los datos que se ingresan al módulo HC-05 por el pin RX se transmiten vía Bluetooth al dispositivo conectado y los datos recibidos se devuelven por el pin TX.

Modo AT1: Para entrar a este estado después de conectar y alimentar el modulo es necesario presionar el botón del HC-05. En este estado podemos enviar comandos AT tomando en cuenta que debemos tener la misma velocidad es decir 9600. En este estado el led parpadea rápidamente igual que en el estado desconectado.

Modo AT2: Para entrar a este estado es necesario tener presionado el botón al momento de alimentar el módulo, es decir el módulo debe de encender con el botón presionado, ya después de que este enciende se suelta el botón y queda en el estado de modo AT2. Para este estado es necesario enviar los comandos AT a una velocidad de 38400 baudios, esto es muy útil cuando nos olvidamos de la velocidad con la que hemos dejado el modulo.

El led en este estado parpadea rápidamente.

En el caso de la configuración para utilizarlo en el Cuadricóptero, basta con tan solo configurar lo más básico con el modo AT2, por ejemplo:

- Cambiar el Nombre.
- Utilizarlo como esclavo pero en este caso está por defecto así que no se cambia esa opción.
- Es necesario cambiar la velocidad de transmisión a 115200, por la cantidad de información que se maneja además de la necesidad de una velocidad alta.

Con los cambios mencionados es suficiente para que el Cuadricóptero sea capaz de comunicarse con la PC. Considerando que en los procesos realizados por el microcontrolador son rápidos.

Para hacer el cambio de la velocidad y el nombre, primeramente se carga el programa que se encuentra en el Anexo C en la placa de Aduino y posteriormente se selecciona la opción “Ambos NL & CL” (pestaña inferior derecha) y la comunicación de 38400 baudios (pestaña inferior derecha) como se muestra en la Figura 36. Después solo se envían los comandos AT por el monitor serial

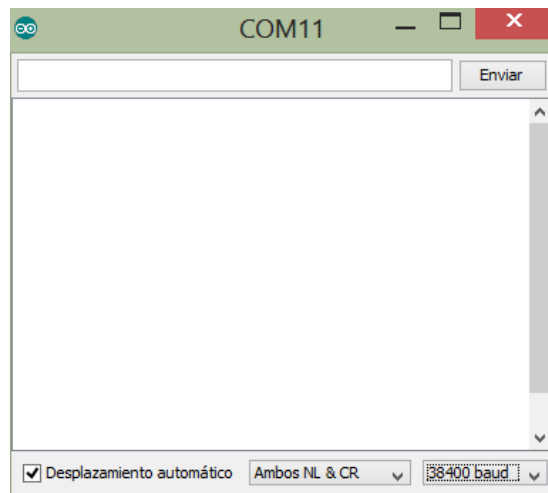


Figura 36. Monitor Serial del ID de Arduino

Para verificar que se establecer la conexión del Bluetooth con el Arduino se envía “AT” y el modulo debe de responder con un “OK” Si este responde de manera correcta, solo es cuestión de enviar dos comandos más para cambiar la velocidad y el nombre, de la siguiente manera.

- AT + NAME = “Nombre que se le quiere asignar al módulo”
- AT + UART = 115200,0,0 “velocidad, Bit de stop, Bit de paridad”

Se selecciona la velocidad de 115200 porque es la más rápida y esta permite procesar mayor cantidad de datos, que son necesarios para la visualización y control del Cuadricóptero. [12]

Al terminar con la configuración del módulo, ya solo se sustituye el cable USB por el módulo HC-05, se conecta la alimentación del módulo y el RX del módulo al TX del Arduino y TX del módulo al Rx del Arduino (el Rx y TX en el Arduino están etiquetados así como en el módulo)

Por último para realizar la conexión con la PC. Si la PC no cuenta con Bluetooth como fue en este caso, se puede comprar un adaptador Bluetooth. Después es necesario instalar un programa para que la PC reconozca al Bluetooth como un puerto COM. En este caso se instaló un asistente TOSHIBA Bluetooth Stack 8.000.03. Al instalar el asistente se genera un icono de Bluetooth en donde se puede configurar el Bluetooth, como el que se muestra en la Figura 37. Para posteriormente realizar la conexión, solo es cuestión de encender el módulo y usar la PC para buscar la conexión con el módulo como se muestra en la Figura 38

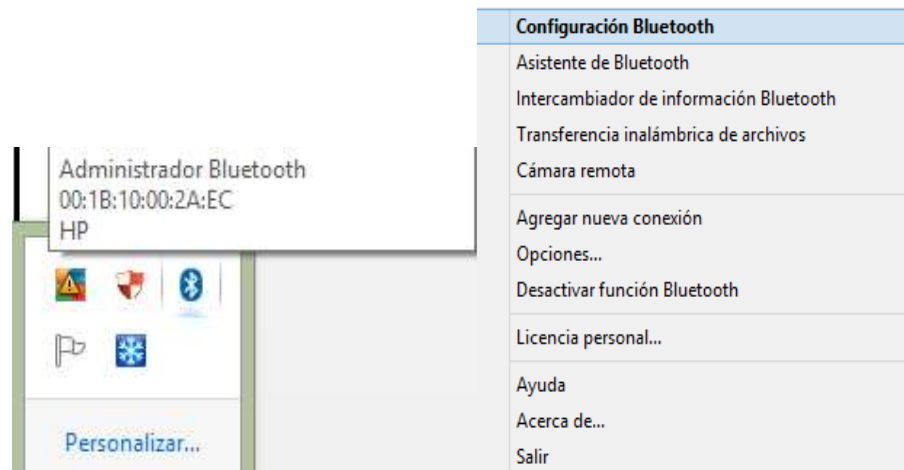


Figura 37 Icono al instalar el Bluetooth y asistente Toshiba (Configuración)



Figura 38. Asistente Toshiba

3.6.2 Control con Arduino

El software del microcontrolador esta implementado en lenguaje C en la plataforma de Arduino. El diagrama de la Figura 39 muestra de manera general, como se diseñó el programa para controlar la posición del Cuadricóptero en los ejes X-Y.

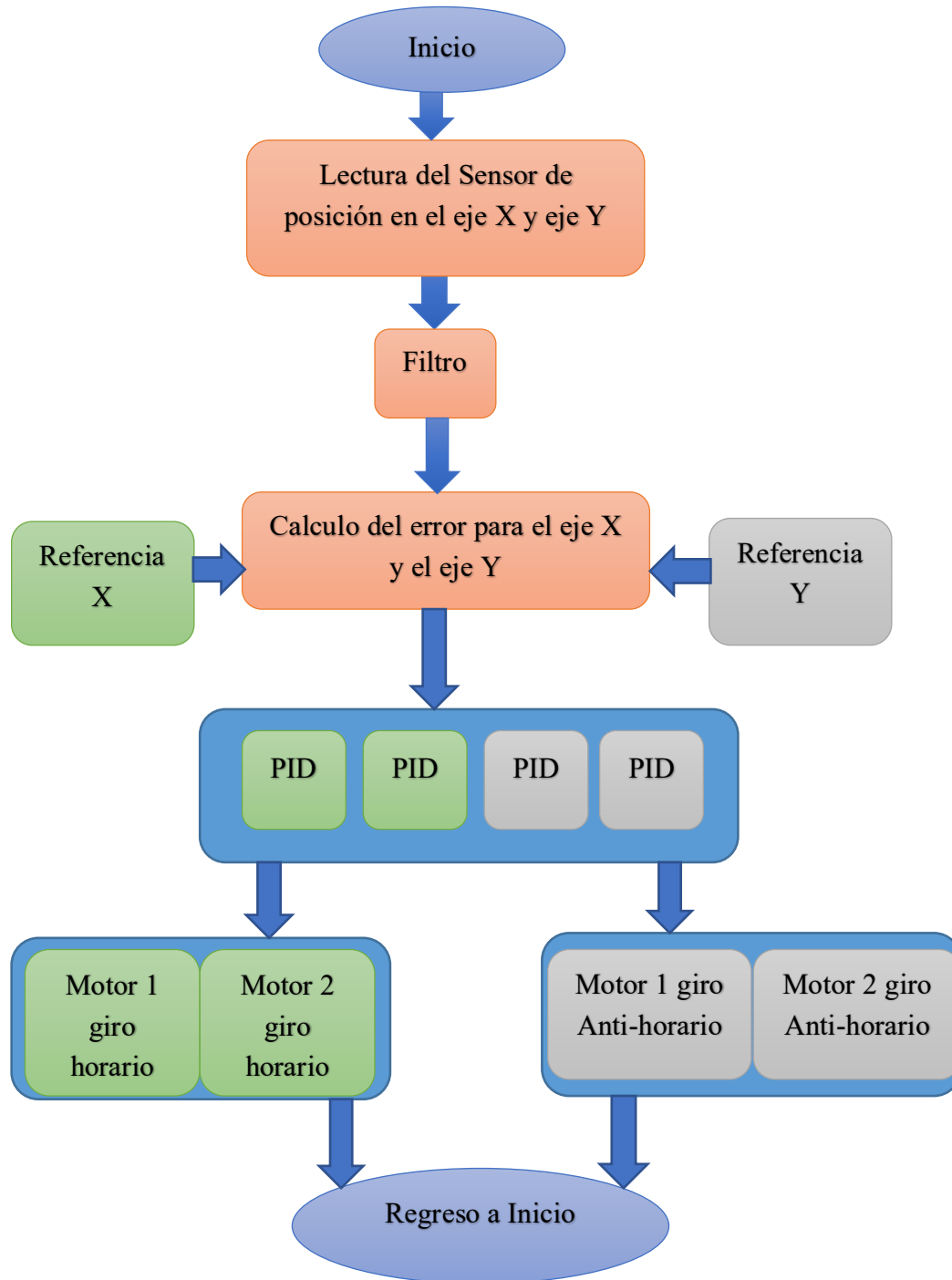


Figura 39. Diagrama de diseño del Cuadricóptero

Inicialmente se toma la lectura del sensor de posición (MPU6050) en los ejes X-Y. Es indispensable conocer el ángulo en el cual se encuentra el Cuadricóptero en cada momento, para poder aplicar el control, es decir poder aplicar la señal de control correcta en cada instante a los motores, basándose en la posición del Cuadricóptero.

Es importante recordar que existe más de una causa que provoca una medición errónea, errores que entorpecen el control y complican el diseño del mismo, como por ejemplo; el sensor es demasiado sensible a perturbaciones como lo puede ser el ruido eléctrico, vibraciones mecánicas causadas por el trabajo de los motores y el tipo de la estructura el cual puede ser un material que facilita la transmisión de vibraciones a través de su cuerpo.

Por esta razón es importante agregar un filtro para reducir el error en cada medición.

En este caso se agregaron dos filtros digitales y un filtro analógicos para mejorar el control y estabilidad del controlador.

Para el filtro analógico solo se utilizó un Capacitor de $2200\mu f$ a 16v conectado entre las terminales de alimentación y tierra. En cambio para el caso de los filtros digitales se implementaron por medio de software, uno de los filtros llamado filtro complementario el cual es utilizado para reducir el ruido y obtener una buena lectura del sensor de posición, mientras que el otro se le llama filtro promediador y este es utilizado para obtener el promedio de los valores del sensor, capturados por el microcontrolador y en consecuencia el resultado de la mejora del controlador.

En la lectura del acelerómetro como la del giroscopio, es decir hace uso de ambos sensores (complemento), para obtener una lectura con el mínimo de interferencia y error.

En el caso del acelerómetro este es muy sensible a aceleraciones en los dos ejes por lo que no se puede obtener una lectura exacta, mientras que el giroscopios acumula su error con el tiempo por esta razón se complementan los dos sensores (acelerómetro y giroscopio que contiene el MPU6050).

La medición del acelerómetro para el eje X-Y se calcula mediante las ecuaciones trigonométricas.(1) y (2)

$$AnguloX = \tan^{-1} \left(\frac{y}{\sqrt{x^2+z^2}} \right) \quad (1)$$

$$AnguloY = \tan^{-1} \left(\frac{x}{\sqrt{y^2+z^2}} \right) \quad (2)$$

Estas ecuaciones son derivadas de la relación de aceleración gravitacional y la posición en la cual se encuentre el sensor.

Ahora para realizar el cálculo del ángulo pero con el uso del giroscopio, este consta de mediciones cada instante de tiempo Δt es decir suponiendo que se conoce la posición inicial y posterior a ello se realiza un giro, gracias al monitoreo en cada instante de tiempo Δt se logra

obtener el ángulo actual, de esta manera y con el uso de las ecuaciones (3) y (4) es posible el cálculo del ángulo.

$$AnguloX = AnguloXAnt + GiroX \Delta t \quad (3)$$

$$AnguloY = AnguloYAnt + GiroY \Delta t \quad (4)$$

Como se mencionó este filtro es llamado complementario ya que consta de la combinación de ambas ecuaciones es decir, se complementan entre sí para lograr el buen funcionamiento del filtro. De esta manera se obtiene una nueva ecuación (ecuación (5)) que describe el filtro complementario, la cual es la siguiente:

$$Angulo = 0.98(Angulo + AnguloGiroscopio \Delta t) + 0.02 * AnguloAcelerometro \quad (5)$$

Con la diferencia que se multiplica por un porcentaje, el cual permite el ajuste del cálculo del ángulo en cada una de las ecuaciones anteriores es decir la suma de ambos porcentajes nos da como resultado la unidad ($0.98+0.02=1$).

Por lo tanto este filtro complementario es utilizado para obtener la medición del sensor de posición y que el resultado sea lo más exacto posible.

Pero como el resultado es afectado al utilizar el sensor en el Cuadricóptero este continua con variaciones, es la razón por la cual se aplica el segundo filtro digital (filtro promediador).

El filtro promediador tiene la función obtener el promedio de una cantidad de valores limitados por el microcontrolador y la respuesta del filtro es decir el valor de muestras seleccionado para obtener el promedio debe ser seleccionado según la necesidad del filtrado así como la capacidad del microcontrolador.

El filtro consiste inicialmente en seleccionar una muestra y dividirla entre el número de muestras (en el primer caso es 1), al continuar con la segunda muestra se suma con el resultado del promedio de la muestra anterior y se divide entre el número de muestras (muestra número 2) y se continua de la misma manera para la tercera muestra hasta alcanzar el valor del número máximo de muestras seleccionado para el filtro.

Al llegar al número máximo de muestras se continúa con el mismo procedimiento tomando como divisor al número de muestras máximo para el filtro.

Para el filtro promediador se realiza un esquema del procedimiento como se muestra en la Figura 40:

Al solucionar el problema de la medición errónea del sensor que proporciona la posición del Cuadricóptero, se continúa con el diseño del controlador PID aplicado al Cuadricóptero, para este caso se implementa un controlador PID por cada uno de los motores.

Es decir se hace uso de cuatro controladores tipo PID.

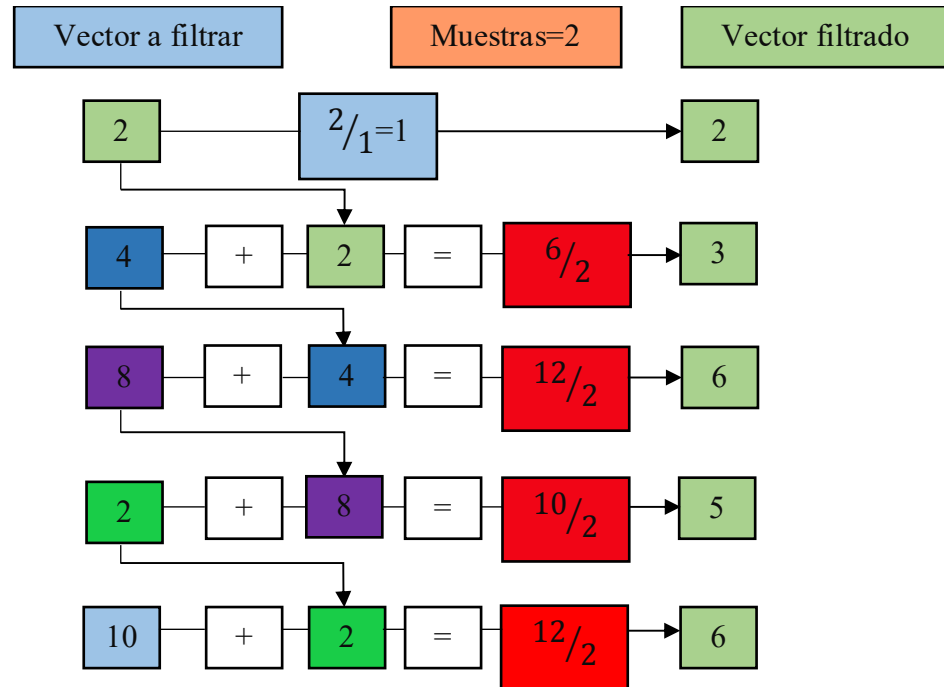


Figura 40 Esquema del procedimiento del filtro promediador

3.7 Controlador PID

El controlador PID permite llevar a cabo el control del Cuadricóptero es decir; proporciona el control de estabilidad para posicionar el Cuadricóptero en una posición deseada, sobre los ejes X-Y, eliminando las perturbaciones que puedan afectar tanto a los elementos del Cuadricóptero o actividades externas.

El controlador funciona de acuerdo al error obtenido respecto a la posición de referencia y el valor de la posición actual.

El error esta expresado como $e(t)$ en donde el error está presente en cada instante de tiempo pero como no es posible manejar valores en tiempo real por el microcontrolador, por la infinidad de valores que existen, es necesario utilizar un tiempo de muestreo t , para discretizar la lectura y así poder realizar su cálculo de acuerdo a la ecuación (6).

$$error = Ref - Ang \quad (6)$$

A partir del error obtenido o calculado es posible obtener los parámetros por el cual está constituido el controlador (PID) es decir; se puede estimar la acción proporcional, integral y diferencial. Para los tres casos como se menciona es necesario conocer el error en cada instante de tiempo. En el caso de la acción proporcional está dada de manera proporcional al error de ahí el termino acción proporcional, la cual está dada por la ecuación (7) en el tiempo.

$$U_p = K_p e(t) \quad (7)$$

En donde K_p es una constante de proporcionalidad, es decir esta constante permite adecuar la acción proporcional permitiendo aumentar la acción proporcional o disminuirla. Es decir; esta constante es utilizada para acondicionar la acción de control proporcional.

Para discretizar la ecuación de acción de control proporcional, basta con realizar la multiplicación de la constante proporcional por el error calculado en cada tiempo de muestreo es decir cada $64\mu s$ en este caso por la interrupción que es utilizada por el microcontrolador para realizar la estimación del control, de esta manera se obtiene la ecuación (8) para realizar el cálculo.

$$U_p = K_p(error) \quad (8)$$

En el caso de la acción integral se obtiene mediante la integral del error multiplicado por una constante de proporcionalidad correspondiente a la parte integrar de la misma manera que la parte proporcional. Para lo cual la parte integral está representada mediante la ecuación (9).

$$U_i = K_i \int edt \quad (9)$$

En donde K_i es la constante de integración mencionada anteriormente, la cual tiene la misma función, que la constante de proporcionalidad es decir permite ajustar la parte integral. La ventaja de la acción integral es poder alcanzar la referencia en un tiempo más corto es decir permitir una respuesta más rápida, gracias a que la integral del error permite acumular el error demandando una respuesta cada vez más rápida si el error está presente.

Para poder utilizar aplicar la parte integral con el microcontrolador se tiene que discretizar la integral del error, para ello es posible representarla a partir de la sumatoria de todos los errores ocurridos durante un tiempo de muestreo ya definido, además la multiplicación por su constante de proporcionalidad K_i , de esta manera se puede representar mediante la ecuación (10)

$$\begin{aligned} errorPasado &= error_{pasado} + error_i \\ U_i &= K_i errorPasado \end{aligned} \quad (10)$$

Ahora para la acción de control derivativa esta depende de la misma manera del error pero en este caso, el control derivativo interviene cuando existe un cambio en el error, por lo tanto la acción de control es expresada como la derivada del error multiplica por una constante de proporcionalidad derivativa K_d , como se muestra en la ecuación (11)

$$U_d = K_d \frac{de}{dt} \quad (11)$$

Al que en los casos anteriores para poder manejar o representar de manera discreta por el microcontrolador, se tiene que utilizar el tiempo de muestreo que el microcontrolador puede manejar. Para el caso de la derivada se expresa como el cambio del error o como la pendiente del error, para ello se calcula el error derivativo como la diferencia del error calculado menos

el error anterior, de esta manera se puede expresar mediante la ecuación (12) el error derivativo.

$$errorDerivativo = error - errorAnterior \quad (12)$$

De la ecuación anterior se multiplica la constante de proporcionalidad para así poder obtener la acción de control derivativa, como se muestra en la ecuación (13).

$$U_d = K_d errorDerivativo \quad (13)$$

Con el cálculo de acción de control derivativo, permite mejorar bastante el control de posición del Cuadricóptero gracias a que permite eliminar los cambios bruscos u oscilaciones permitiendo mejorar el control.

Finalmente para poder aplicar el control PID se realiza la suma de cada una de las acciones de control que son necesarias para la constitución del controlador PID, como se muestra en la ecuación (14)

$$PID = U_p + U_i + U_d \quad (14)$$

Como el controlador funciona según el error respecto a la referencia y el valor de la posición actual, este se puede ejemplificar con un caso por ejemplo; si se supone que se encuentra inclinado con un ángulo de 45° en ambos ejes “X” y “Y” y posteriormente se desea llevar a una nueva posición por ejemplo a la posición de 0° para el eje “X” como para el eje “Y”.

En este caso el error es de 45° por lo que es necesario aplicar una señal adecuada a los motores, para llevar a la posición deseada. Tomando en cuenta que solo un par de ellos necesitan una señal con mayor intensidad mientras que el otro par debe disminuir su velocidad, para permitir que los motores que actuaran para levantar el Cuadricóptero puedan moverlo con mayor facilidad.

Posteriormente al aproximarse a la posición deseada los motores que disminuyeron su velocidad tienden a tratar de mantener esa velocidad para mantener el Cuadricóptero mientras que los motores que levantaron el Cuadricóptero de la misma manera tiene que tratar de mantener la velocidad, para emparejarse con el otro par de motores y lograr mantener el Cuadricóptero suponiendo que aún no existe ninguna perturbación.

Algo importante para trabajar con el diseño del controlador, es recordar que el sensor proporciona el valor del ángulo, puede ser positivo como negativo según sea la posición en la que se encuentre, para ello se tiene que tomar en cuenta el valor y acondicionar para que el microcontrolador pueda interpretar el valor del ángulo en cada cuadrante.

Por eso es necesario, considerar las diferentes posiciones en donde el Cuadricóptero puede posicionarse. También la acción de los motores para lograr llegar a cada una de la referencia que se desea.

En la siguiente Figura 41 se puede apreciar un diagrama de un controlador PID para uno de los motores utilizados en el Cuadricóptero.

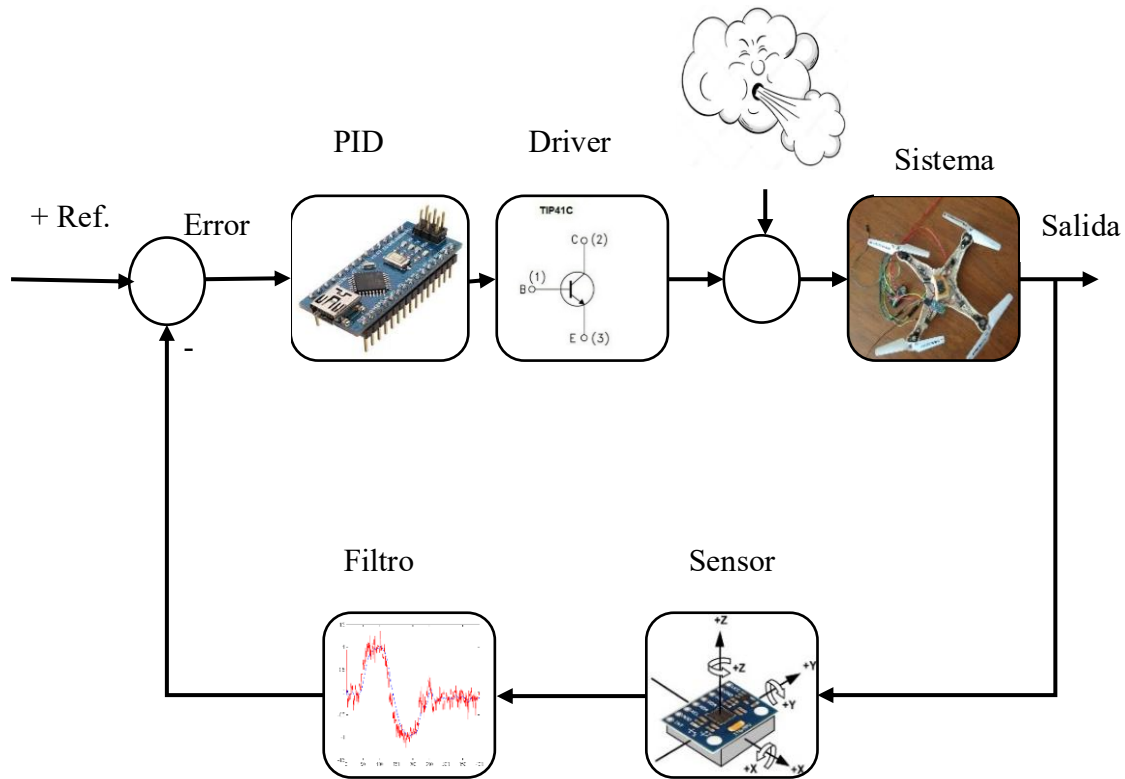


Figura 41. Diagrama de un controlado PID para un motor

En el caso de que no exista comunicación entre la PC y el microcontrolador esté toma como posición de referencia de cero grados en ambos ejes. Es importante hacer mención que el Cuadricóptero puede recibir perturbaciones en cualquier instante, tanto internas debido a ruido de los circuitos o externas como es el movimiento y el viento del medio ambiente; así que una de las tareas principales del controlador es tratar de eliminar este tipo de perturbaciones en el sistema.

El sensor (MPU6050) es el dispositivo que se encarga de medir el valor actual de la posición tanto del eje “X” y “Y” y recordando que este sensor proporciona los datos de posición a través del giroscopio y acelerómetro por lo cual se utiliza un filtro complementario digital implementado en el microcontrolador poder calcular la posición actual del sistema y así poder calcular el error de posición del sistema. El software del controlador PID funcional para el Cuadricóptero se puede apreciar en el Anexo A.

3.8 Interface Gráfica.

En la actualidad el contacto con la computadora se ha vuelto una actividad muy común y necesaria, es por lo cual es necesario crear ambientes amigables que faciliten el uso de todas las herramientas y control desde las computadoras o celulares específicamente.

Es así por lo cual se han creado una Interfaces Graficas de Usuario (GUI). Por ejemplo gracias a las interfaces graficas de usuario los sistemas operativos se convirtieron en soluciones mucho más visuales y accesibles fáciles de manejar por las personas en general.

Para mostrar la gran importancia de contar con una interface gráfica, podemos poner como ejemplo la simple visualización de datos de una manera más fácil de entender comparada con una visualización de instrucciones paso a paso.

Es decir los datos que pueden llegar a la PC desde el Cuadricóptero son demasiado rápidos como para interpretarlos de manera visual, en cambio en una gráfica los datos o señales se pueden interpretar con una mayor claridad.

El comportamiento de los componentes como lo son sensores, motores y hasta del mismo controlador diseñado.

Y no basta solo con la visualización de los datos procesados en el Cuadricóptero, también para el control o mando del mismo es necesario manejarlo de manera directa es decir; para asignarle la posición u referencia paso a paso con las coordenadas mediante números o instrucciones numéricas es complicado, en cambio sí se tiene un mando por ejemplo un perilla (Dial) o un Slider, que solo con cambiarla de posición esta nos envíe los datos correspondientes al giro o desplazamiento facilitando el manejo.

Para ello existen herramientas para el diseño de interfaces gráficas, que pueden ser diseñadas con solo la configuración de bloques o por medio del cogido.

En este caso se utiliza processing, el cual es un entorno de programación de código abierto y además una herramienta de desarrollo basada en java, orientada para crear imágenes, animaciones e interacciones, para realizar adquisición de datos en sistemas visualmente presentables.

La ventaja de processing es la facilidad de programación. Este software permite crear formas y figuras visuales con respecto a la sintaxis escrita sobre el IDE de processing.

Para mostrar de manera general cual es la estructura del código para la interfaz diseñada, se utiliza el diagrama de la Figura 42.

Por ejemplo para describir de manera general el código se tiene como ejemplo el siguiente diagrama que representa de una manera general como se estructuro el código de la interfaz diseñada además este código, se encuentra en el Anexo B.

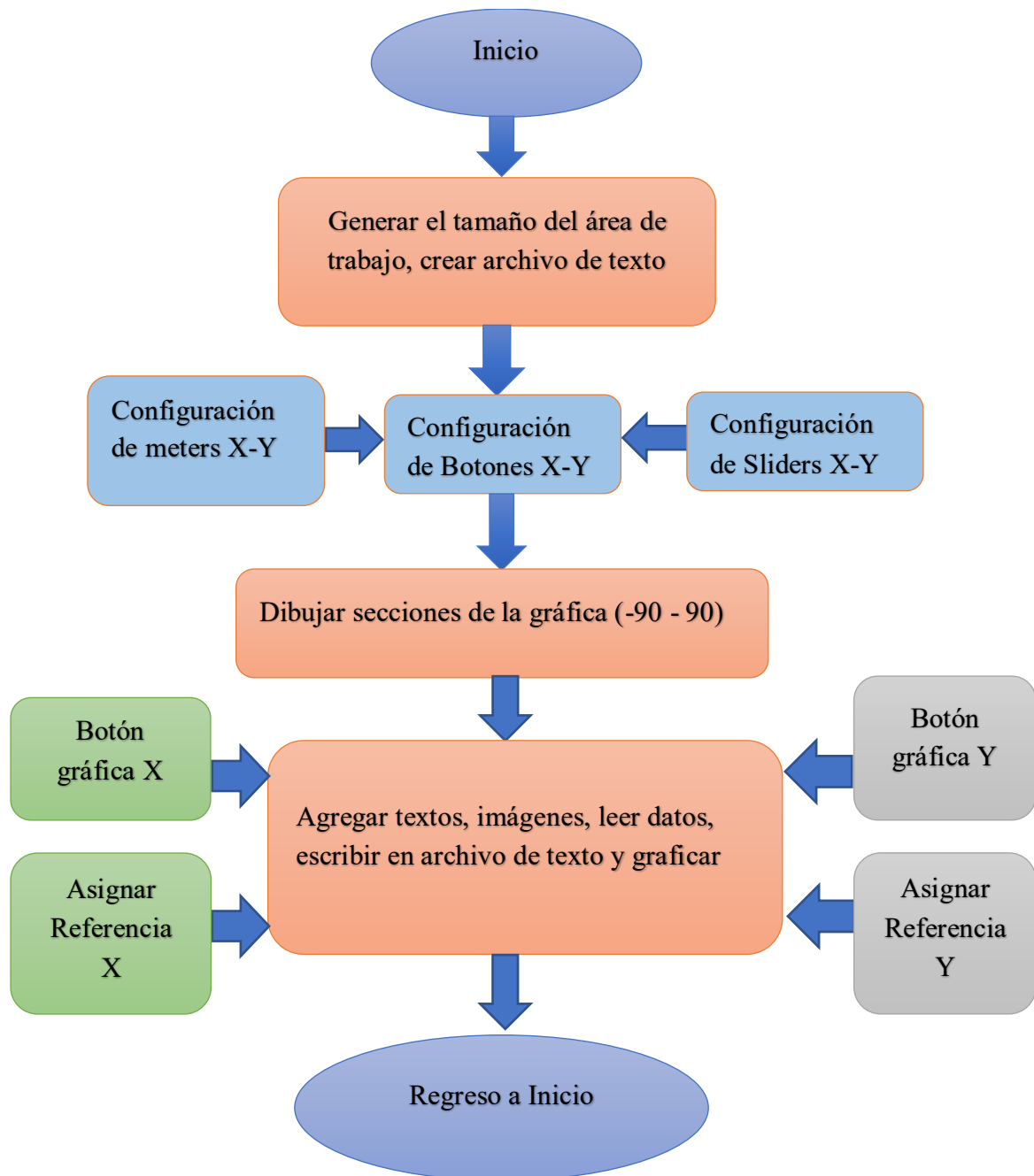


Figura 42 Diagrama del software utilizado en la interfaz gráfica diseñada

En la siguiente Figura 43 se puede apreciar la imagen de la interfaz diseñada la cual permite el monitoreo y control del Cuadricóptero, con una interfaz que busca que el fácil manejo e interpretación, es decir que una persona sin experiencia pueda manipular y entender gráficamente el funcionamiento de la misma.

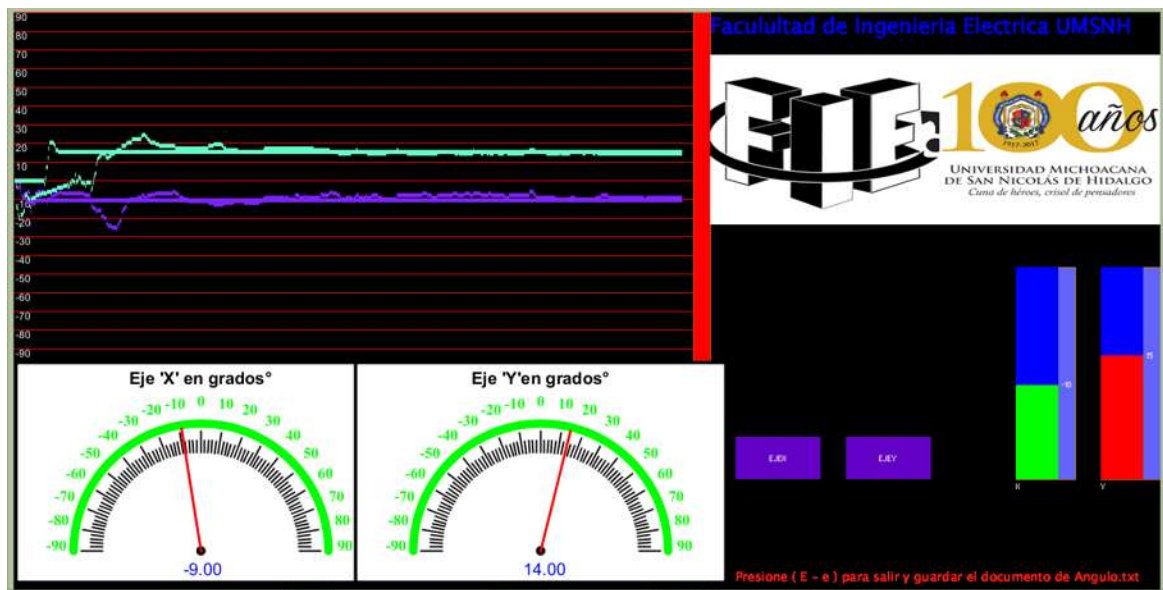


Figura 43. Interfaz gráfica de control y monitoreo del Cuadricóptero.

La interfaz gráfica de manera resumida cuenta con, una área en la cual se grafica la señal de referencia y el comportamiento del Cuadricóptero (Cambio de los ángulos) como se muestra en la Figura 44

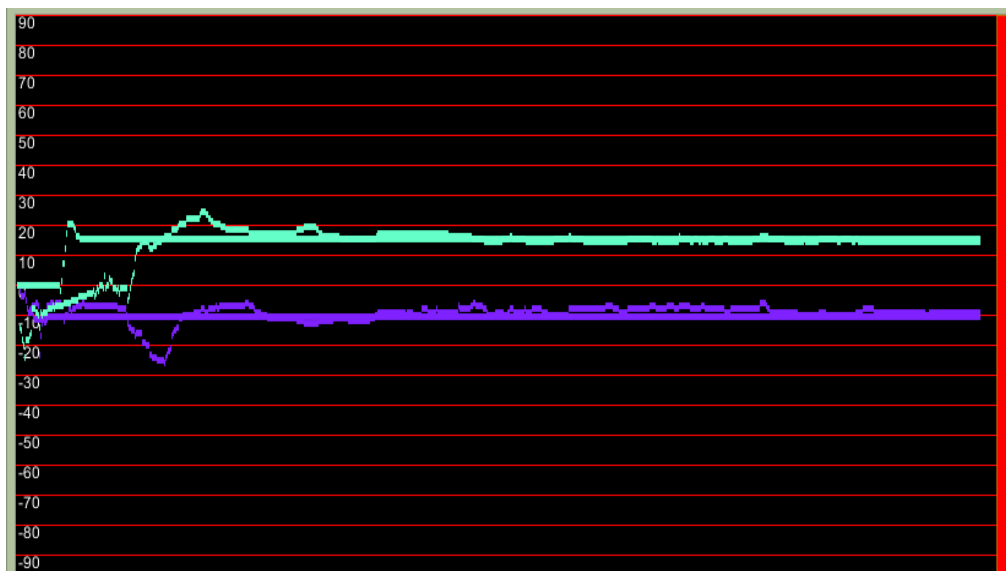


Figura 44 Área en donde se grafica la respuesta u comportamiento del Cuadricóptero

Al igual cuenta con Meters o indicadores tipo analógicos – digital que interpretan el ángulo en donde se encuentra posicionado el Cuadricóptero, este comportamiento es el mismo que se

presenta en la parte gráfica. La finalidad de contar con este tipo de indicadores, es la de poder tener una doble interpretación que permita facilitar la visualización del Comportamiento del Cuadricóptero ya sea gráficamente o por medio de indicadores tipo analógico – digital, como los que se muestran en la Figura 45.

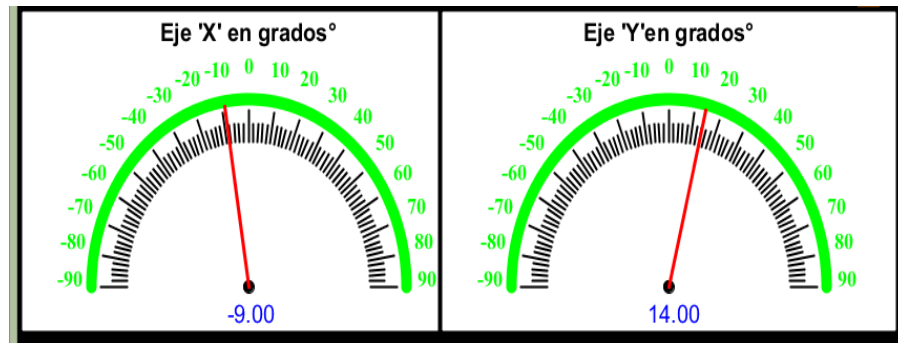


Figura 45. Indicadores Analógicos – Digital (Meters).



Figura 46 Área del manejo del Cuadricóptero

Otros de los elementos que son importantes en una interfaz gráfica son los mandos o controles, por ejemplo para posicionar el Cuadricóptero es necesario mandar las coordenadas o posición que se desea, para ello se asigna una referencia para el eje X, como para el eje Y por medio de dos Sliders (uno por eje).

También cuenta con dos botones que permiten mostrar o limpiar los datos que son graficados ya sea para el eje X o para el eje Y, es decir con estos dos botones se puede limpiar el área en donde se realiza la gráfica, mostrar un solo eje junto con su referencia o ambos ejes con su referencia según se desee. Esto se puede observar en la Figura 46.

Existe una opción en la misma interfaz, que permite guardar los datos del comportamiento ocurrido desde el inicio de la ejecución del programa. La información es guardada en un archivo de texto (txt), al presionar la letra “E” o “e” en la interface. La información guardada en el archivo de texto puede ser limitada, dependerá de la capacidad de transmisión del Cuadricóptero hacia la PC, es decir cuando mayor es la transmisión de información el control y la misma velocidad de la interface, tiende a hacerse más lento.

La información que se desea guardar en el archivo de texto, se agrega en el código (`output.print((int)(value[posición del arreglo recibido]) + "\t");`), es decir se agrega una variable o línea de código correspondiente a los datos que se quieren guardar, así la información que se encuentra ordenada de la siguiente manera.

En la primera columna se encuentra ángulo correspondiente al eje X, en la segunda columna se encuentra el ángulo correspondiente Y, mientras que en la columna 3 y 4 se agregan el mismo ángulo X-Y pero con separación por comas, en la columna 5 y 6 se agrega la acción de control correspondiente al motor 1 y motor 2 del eje X, en la columna 7 y 8 se encuentra la acción de control para el motor 1 y 2 del eje Y. En la columna 9 y 10 se encuentra el error para el eje X y eje Y, por último se agrega la hora actual en la cual se está ejecutando el programa.

Esta información es necesaria para comprobar el funcionamiento del control diseñado, aunque se puede reducir la información guardada o aumentar si es necesario, en la Figura 47 se puede observar un ejemplo del archivo de texto que es generado por la interface.

Además algo importante que debe tomarse en cuenta es, que en este capítulo se logró elegir cada uno de los elementos necesarios para el control y construcción del Cuadricóptero, es decir; todo el hardware y software que constituyen el objetivo de este trabajo.

Por ejemplo se logró la elección de la estructura, así como el material para necesario en este caso para lograr el objetivo. También el diseño de su base, la cual permite el movimiento para poder aplicar el control.

En ambos casos control e interface se logra el diseño del software enfocado al objetivo del trabajo. Para mostrar los resultados obtenidos, estos se presentaran en el siguiente capítulo.

Angulo: Bloc de notas										
Archivo	Edición	Formato	Ver	Ayuda						
-9	14	-9,	14,	194,	211,	200,	205,	-1,	0,	17:17:12
-9	14	-9,	14,	194,	211,	200,	205,	-1,	0,	17:17:12
-9	14	-9,	14,	194,	211,	200,	205,	-1,	0,	17:17:12
-10	14	-10,	14,	203,	202,	200,	205,	0,	0,	17:17:13
-10	14	-10,	14,	203,	202,	200,	205,	0,	0,	17:17:13
-10	14	-10,	14,	203,	202,	200,	205,	0,	0,	17:17:13
-10	14	-10,	14,	203,	202,	200,	205,	0,	0,	17:17:13
-10	14	-10,	14,	203,	202,	200,	205,	0,	0,	17:17:13
-10	15	-10,	15,	203,	202,	192,	214,	0,	-1,	17:17:13
-10	14	-10,	14,	203,	202,	200,	205,	0,	0,	17:17:14
-10	14	-10,	14,	203,	202,	200,	205,	0,	0,	17:17:14
-10	14	-10,	14,	203,	202,	200,	205,	0,	0,	17:17:14
-10	14	-10,	14,	203,	202,	200,	205,	0,	0,	17:17:14
-10	14	-10,	14,	203,	202,	200,	205,	0,	0,	17:17:14
-10	14	-10,	14,	203,	202,	200,	205,	0,	0,	17:17:14
-10	14	-10,	14,	203,	202,	200,	205,	0,	0,	17:17:15
-10	14	-10,	14,	203,	202,	200,	205,	0,	0,	17:17:15
-10	14	-10,	14,	203,	202,	200,	205,	0,	0,	17:17:15
-10	14	-10,	14,	203,	202,	200,	205,	0,	0,	17:17:15
-10	14	-10,	14,	203,	202,	200,	205,	0,	0,	17:17:15
-9	15	-9,	15,	194,	211,	192,	214,	-1,	-1,	17:17:16
-9	15	-9,	15,	194,	211,	192,	214,	-1,	-1,	17:17:16
-9	15	-9,	15,	194,	211,	192,	214,	-1,	-1,	17:17:16

Figura 47 Archivo de texto creado por la interface grafica

Capítulo 4

Pruebas y Resultados

4.1 Introducción

En el este capítulo se presentan algunas de las pruebas que son necesarias para comprobar el funcionamiento del Cuadricóptero. Se inicia con los resultados del filtro promediador que es utilizado para reducir ruido en el sensor de posición. Posteriormente se continúa con pruebas al controlador, para ello se aplican distintas pruebas, para mantener una referencia deseada. Se inicia con una referencia de cero grados en ambos ejes ($x=0, y=0$), después se realiza una prueba con una referencia distinta a la de cero grado, en este caso se cambia la referencia a una en $x=-10$ y $y=10$ para observar el comportamiento del controlador.

Además se obtienen los límites para ambos ejes es decir; se realizan pruebas para el alcance del control en ángulos positivos como para ángulos negativos. El primer caso en donde ambos ejes son positivos y para el segundo caso cuando los dos ángulos son negativos.

Por último se realizan pruebas en donde se aplican perturbaciones cuando este se encuentra en funcionamiento, una perturbación mínima (golpe pequeño) que provoque la inestabilidad. Para la siguiente prueba se aplica una perturbación pero más alta que sea más notorio como este pierde su referencia por un momento y vuelve a regresar la referencia fijada.

4.2 Filtro digital.

Al iniciar las pruebas al Cuadricóptero, se inicia con las pruebas al filtro digital. La razón por la cual es necesario realizar pruebas al filtro digital, son los problemas en la medición del sensor de posición, estos problemas son causados principalmente por vibraciones y ruido eléctrico.

Para ello se muestran los resultados al medir la posición, sin utilizar el filtro digital y con el filtro digital, como se muestra en la Figura 48, 49, 50 y 51

Con el uso del filtro se logra reducir la mala inestabilidad además de mejorar el control. Este filtro se encarga de obtener un promedio de la señal original permitiendo reducir picos que afectan la estabilidad y pulsaciones en motores.

El filtrado aplicado (filtro promediador) es muy necesario para mejorar la respuesta del controlador, así como para lograr el objetivo de controlar el Cuadricóptero en los dos ejes, esto a causa de los problemas de vibraciones y ruido eléctrico causado por los motores.

Además en la mayoría de las aplicaciones de control es necesario el uso de filtros analógicos y digitales por razones similares de perturbaciones eléctricas o perturbaciones externas como vibraciones las cuales siempre dependerá de la aplicación.

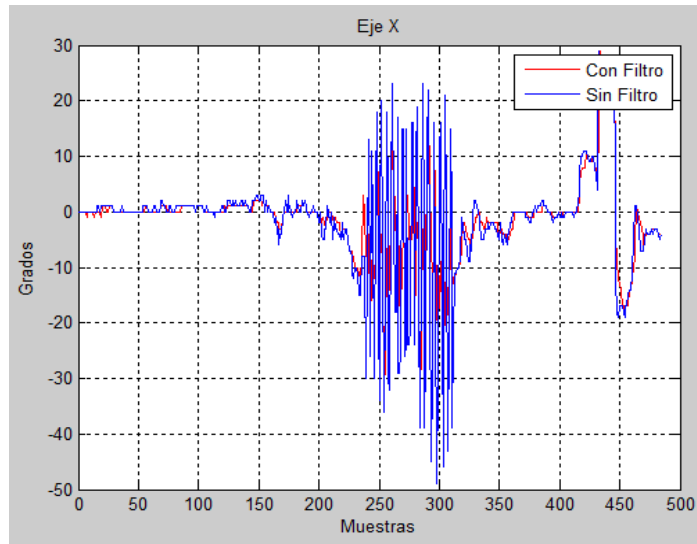


Figura 48. Gráfica del filtro digital promediador aplicado al eje X

Para notar que se logra reducir el ruido se, realiza un zoom a una parte de la señal en la Figura 48. Como resultado se logra apreciar el resultado en la Figura 49.

Los resultados de la aplicación del filtro es una señal con menores variaciones comparada con la señal sin filtro. Esto se puede apreciar en la Figura 49 en donde la señal filtrada tiene menos cambios comparada con la que no es filtrada. Las variaciones de posición que se aplicaron al Cuadricóptero se toman en ambos ejes y los resultados son presentados por eje con un zoom para apreciar los resultados.

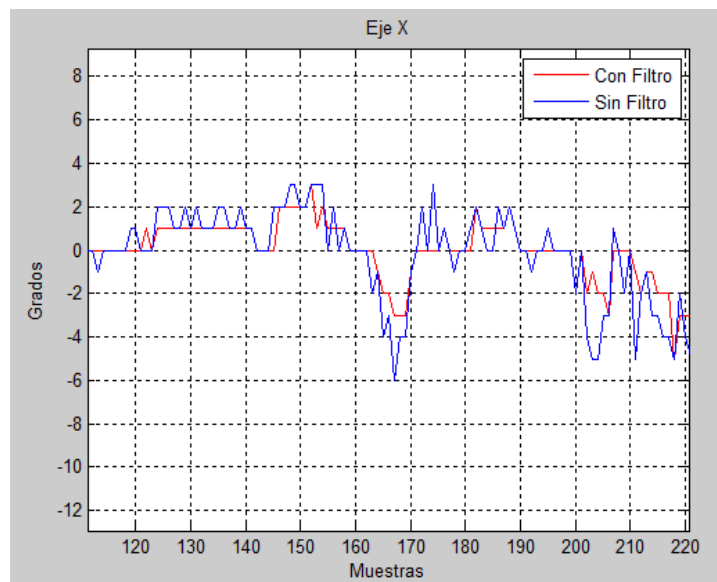


Figura 49 Zoom del filtro promediador aplicado al eje X

Además de obtener aparentemente una gran cantidad de picos, una parte de estos se debe a que los datos capturados no son los datos reales, es decir la transferencia de datos desde el microcontrolador hacia la PC son pocos por la necesidad de mejorar la velocidad en el control. Por esta razón las muestras capturadas causan que gráficamente se perciba una gran cantidad de ruido, aunque realmente es menor y al igual el uso del filtro permite eliminar parte de ese ruido.

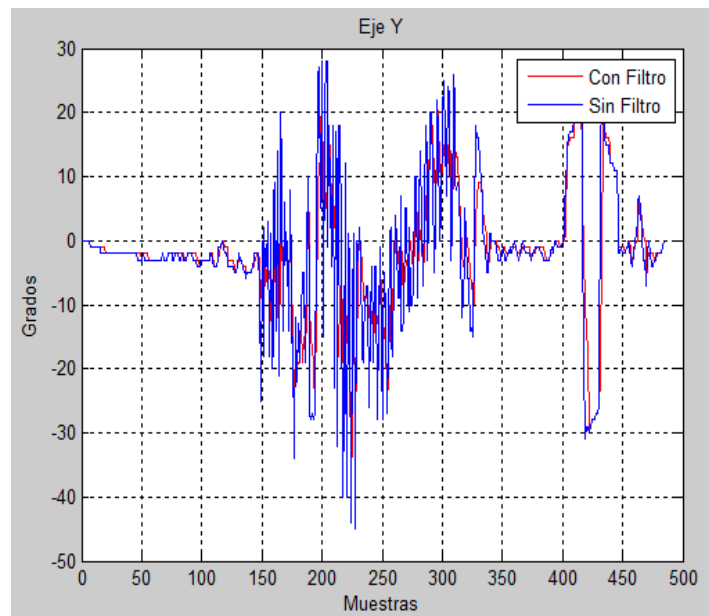


Figura 50. Gráfica del filtro digital promediador aplicado al eje Y

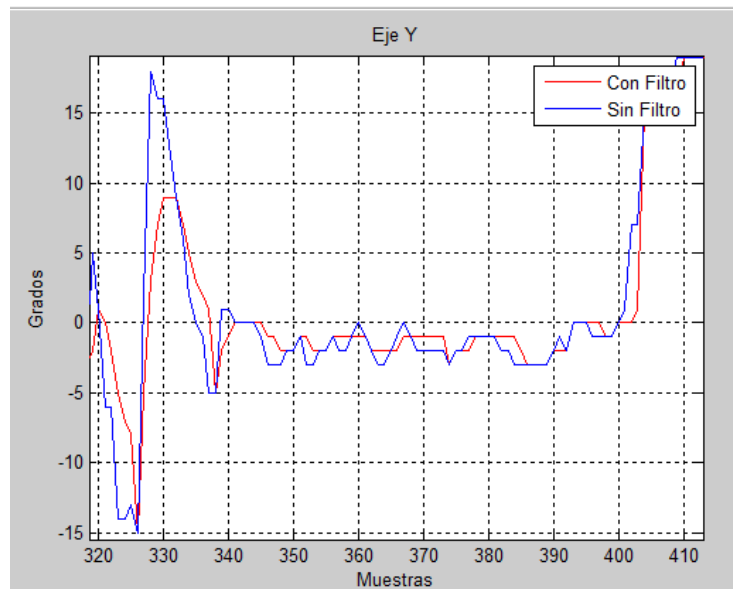


Figura 51, Zoom del filtro promediador aplicado al eje Y

4.3 Referencia en cero grados

Una de las principales pruebas que se deben llevar a cabo es la respuesta del controlador, acción de control y error.

Para ello se realizan diferentes pruebas, una de las pruebas es: a partir de una posición inicial X, Y ; llevarlo a una posición referenciada, el procedimiento consiste en realizar un cambio de referencia y observar la respuesta del controlador.

En el primer caso se inicia desde una posición $x=11, y=27$ con un cambio de referencia de $x=0, y=0$. El resultado es una respuesta con algunas oscilaciones antes de llegar a su referencia marcada, estas oscilaciones al inicio son causadas por el cambio de referencia. Es decir el error es grande y por consiguiente este ocasiona que la acción de control sea de una gran magnitud inicialmente, provocando oscilaciones al intentar llegar rápidamente a la referencia marcada.



Figura 52 Imagen que muestra al Cuadricóptero en su posición inicial en la cual se toman los datos para su respuesta del controlador implementado con referencia de cero grados en ambos ejes.

Mientras que los pequeños pulsos que se aprecian cuando este llega a su referencia, son causados por la cantidad de datos manejados, es decir la cantidad está limitada a graficar o capturar solo un porcentaje de los datos reales que el Cuadricóptero utiliza.

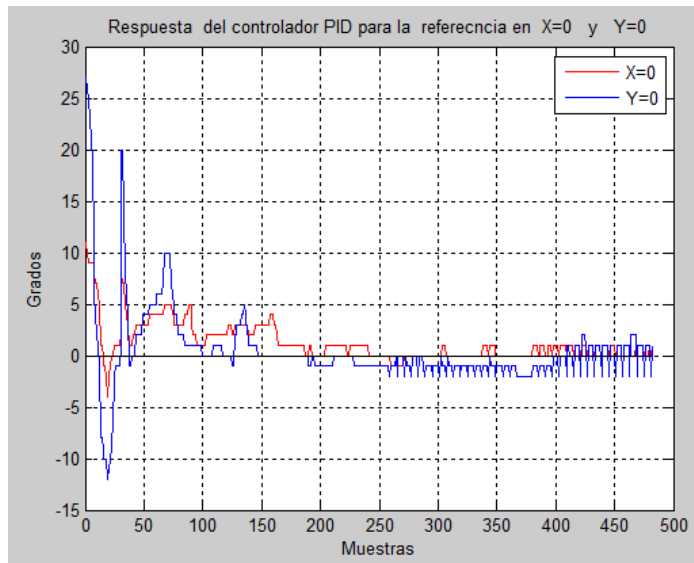


Figura 53. Gráfica de la respuesta del controlador con una referencia de $X=0$, $Y=0$

En el caso de la acción de control, los datos son representados por datos del 0-255 que son utilizados por la señal PWM, estos datos no son fáciles de interpretar gráficamente por lo cual se tiene que aplicar un filtro.

El filtro se aplica a los datos capturados, esto para cada una de las cuatro señales aplicadas a los motores. Así se puede apreciar como el controlador u acción de control es muy distinto para cada uno de los motores. Por ejemplo como se ha mencionado anteriormente, un par de motores en este caso actúan con una intensidad alta mientras que los restantes no actúan, inicialmente y conforme se llega a la referencia los motores tienden a mantener su velocidad.

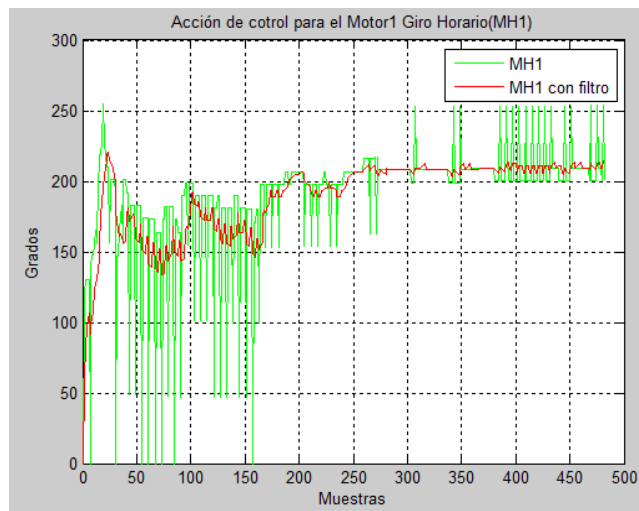


Figura 54. Gráfica de la acción de control para el motor 1 con giro horario.

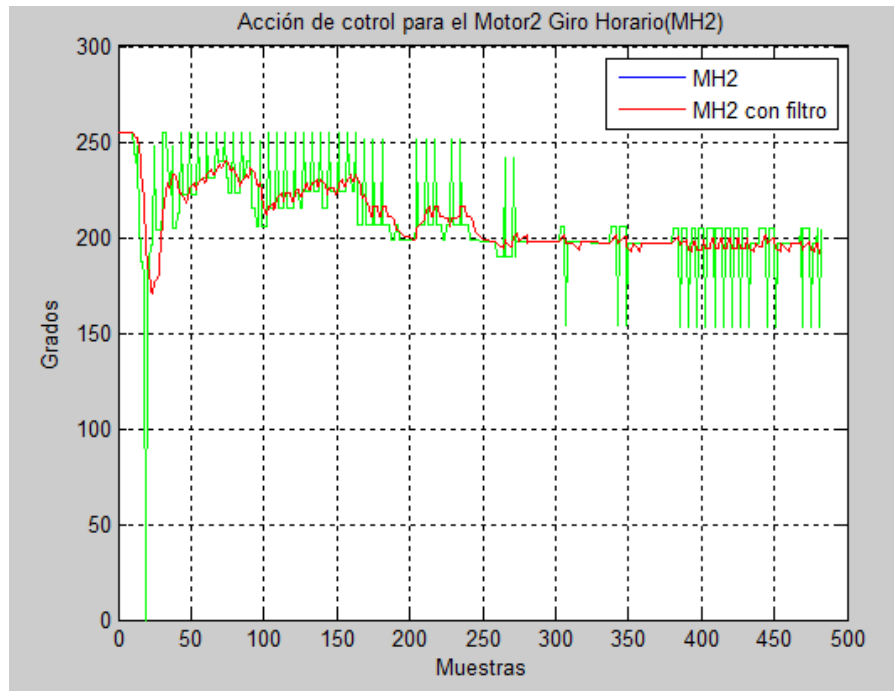


Figura 55. Gráfica de la acción de control para el motor 2 con giro horario.

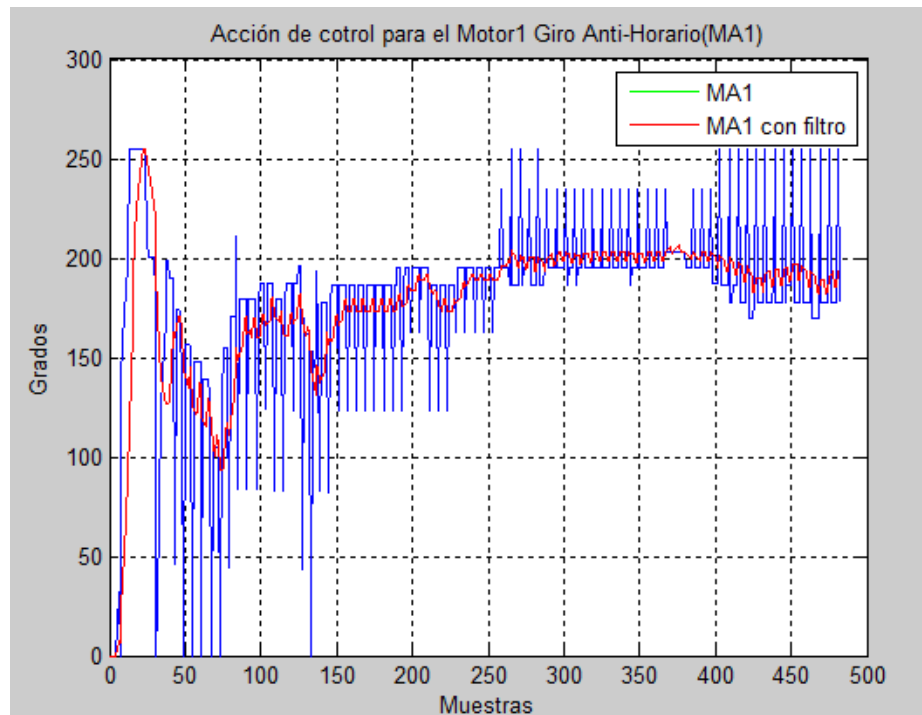


Figura 56. Gráfica de la acción de control para el motor 1 con giro anti-horario.

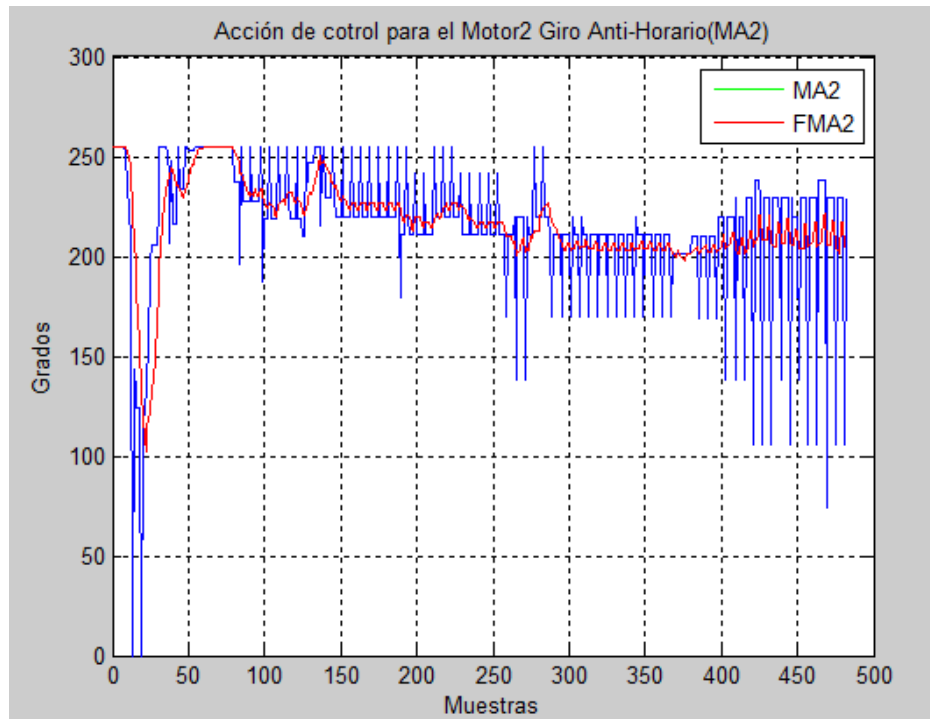


Figura 57. Gráfica de la acción de control para el motor 2 con giro anti-horario.

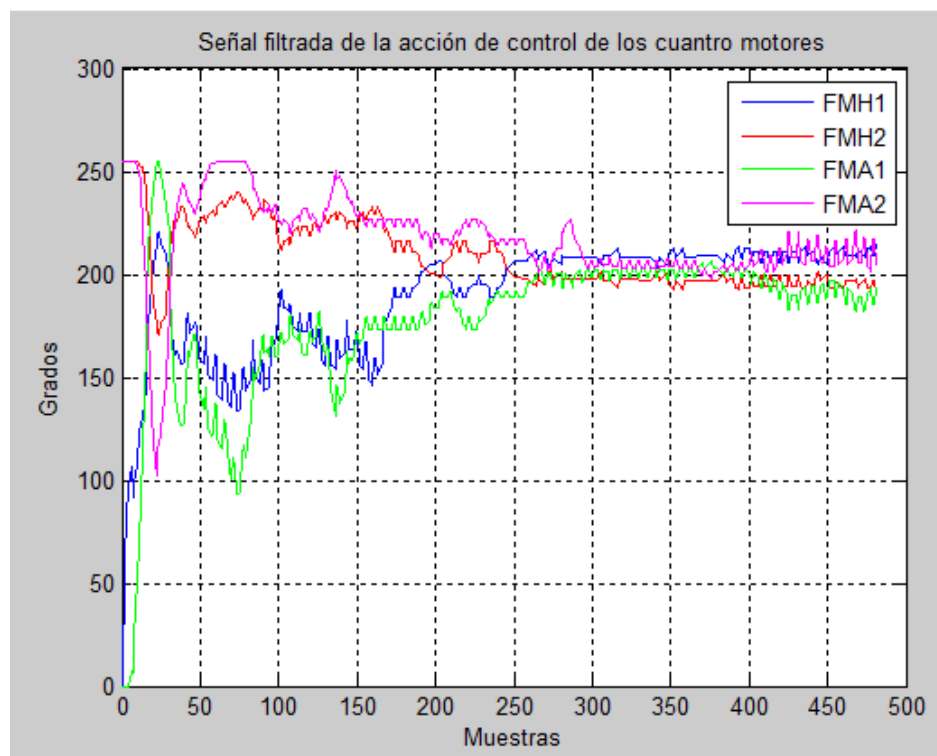


Figura 58. Gráfica de la acción de control aplicada a los cuatro motores.

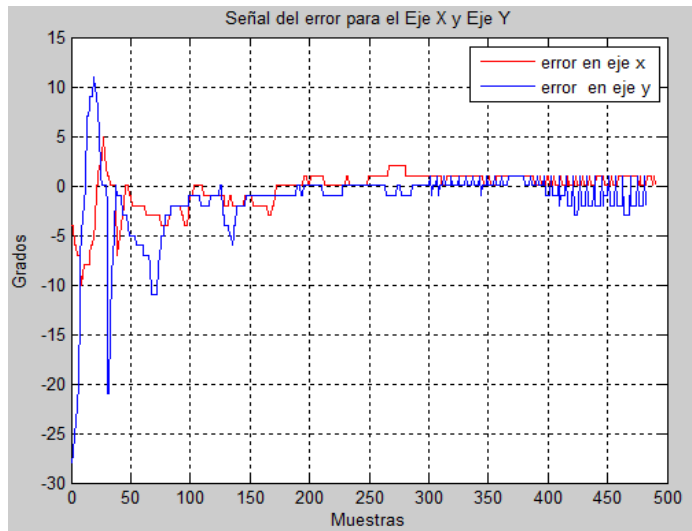


Figura 59. Gráfica del error para una referencia de cero en ambos ejes.

4.4 Referencia en $x=-10$, $y=10$

Otra de las pruebas es cambiar la referencia, es decir que el control se pueda llevar en ambos ejes. Para ello se realiza un cambio de referencia de $x=0, y=0$ a $x=-10, y=10$, así en la Figura 61 se puede observar como las oscilaciones son menores al inicio, en este caso es así porque el control no necesita de un cambio tan brusco por que la velocidad de los motores al inicio es alta y para el cambio solo es cuestión de disminuir poco la velocidad, esto para un par de motores mientras que para el otro par de ellos es lo contrario, por la posición en la que se encuentra antes del cambio ($x=0, y=0$). Con esta prueba se observa que es posible llevar el control a distintas posiciones, limitándose la inclinación provocada por la base.

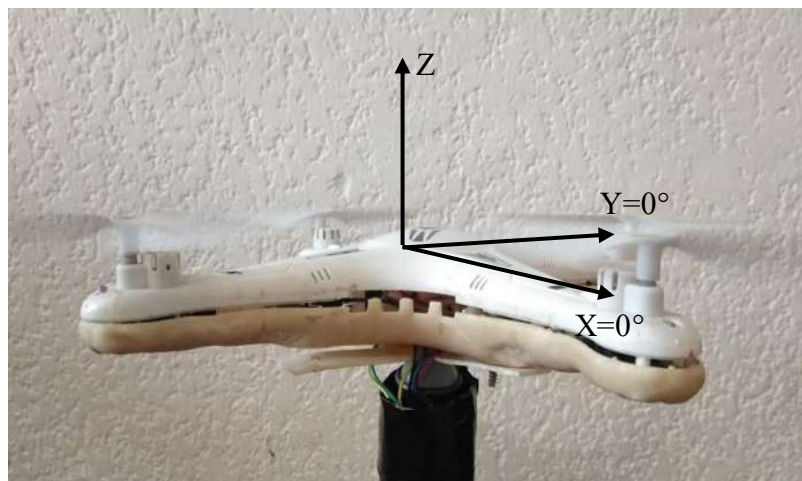


Figura 60 Imagen del Cuadricóptero en su posición inicial para iniciar con las pruebas de cambio de referencia

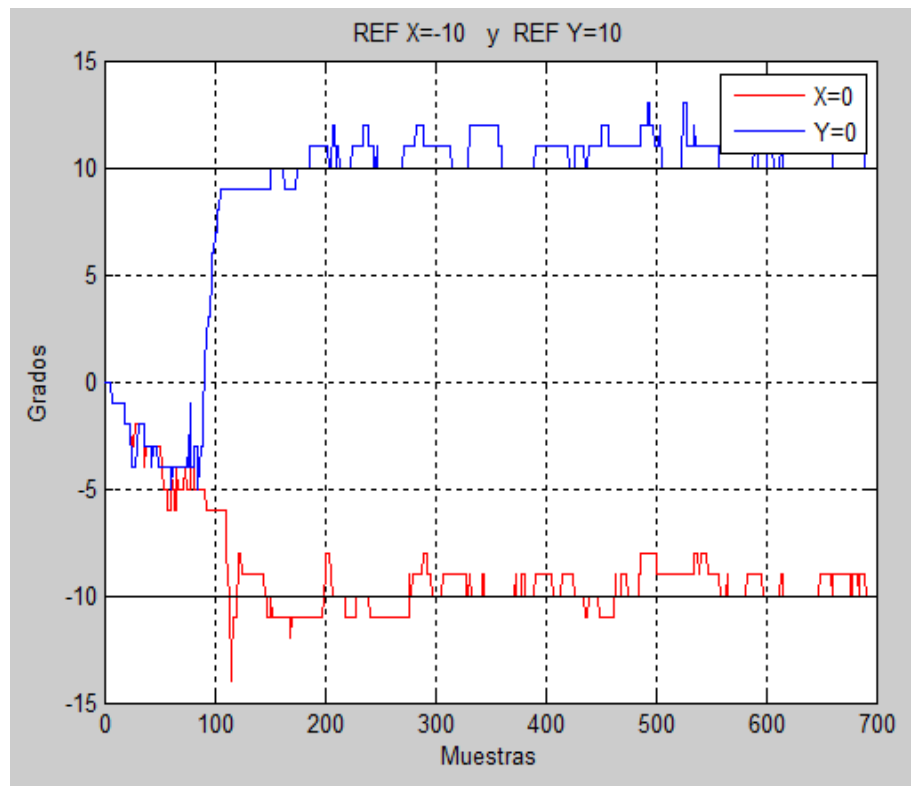


Figura 61. Gráfica con referencia distinta de cero, para mostrar el cambio de referencia X y Y.

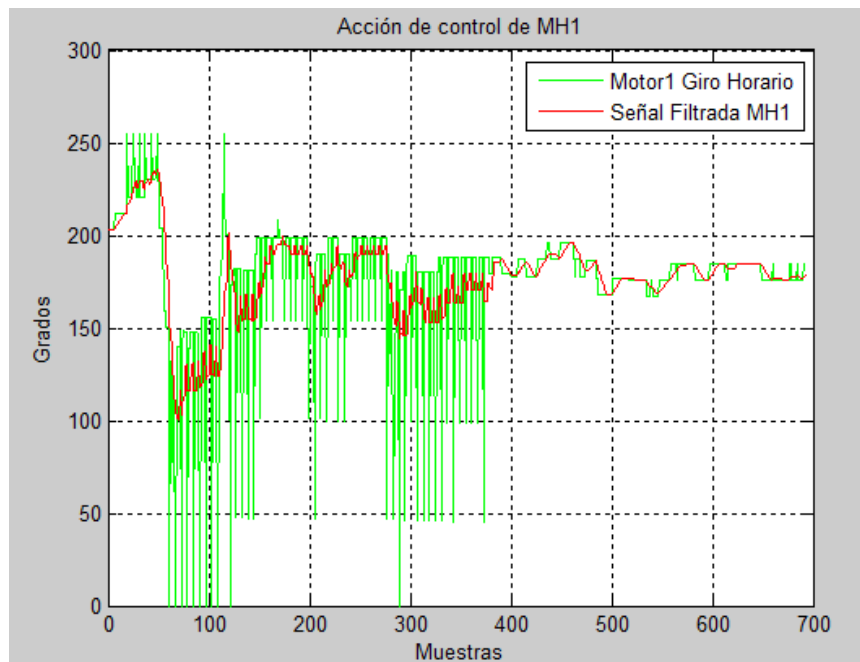


Figura 62. Gráfica de la acción de control para el motor 1 con giro horario.



Figura 63. Gráfica de la acción de control para el motor 2 con giro horario.

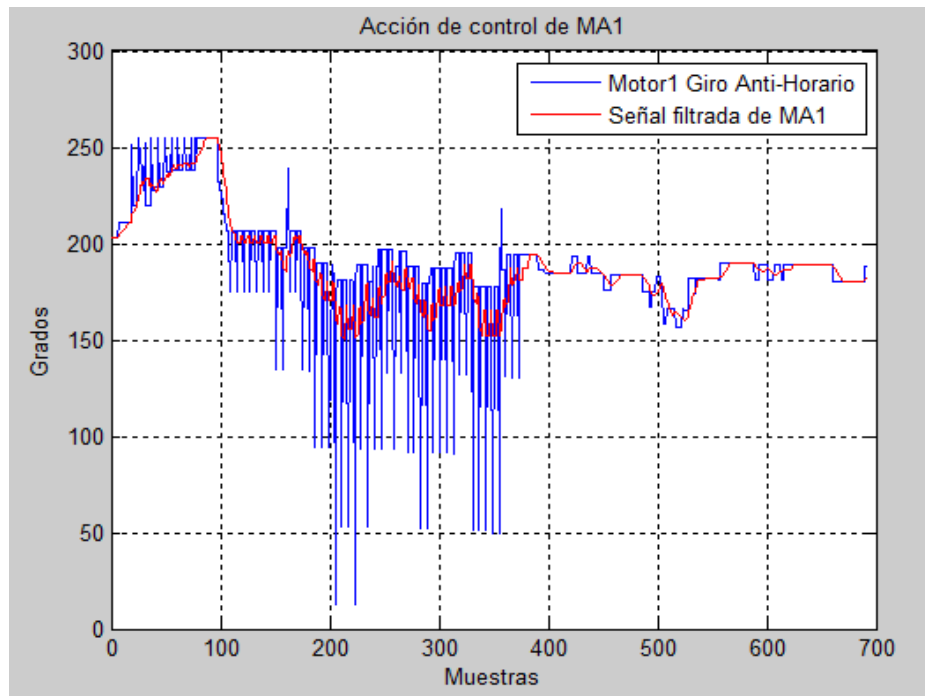


Figura 64 Gráfica de la acción control para el motor 1 con giro anti-horario

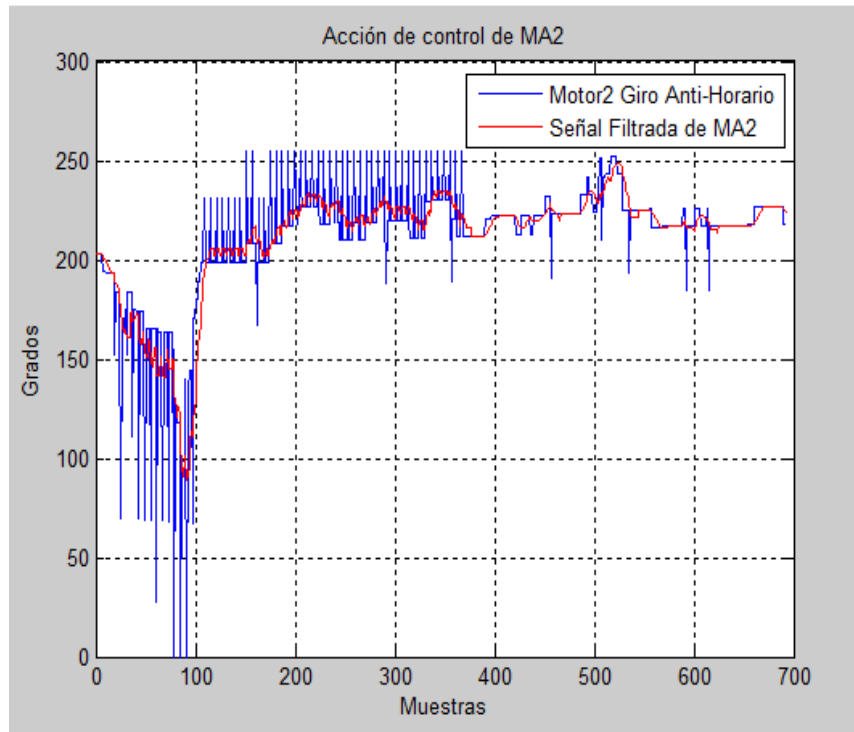


Figura 65 Gráfica de la acción de control para el motor 2 con giro anti –horario.

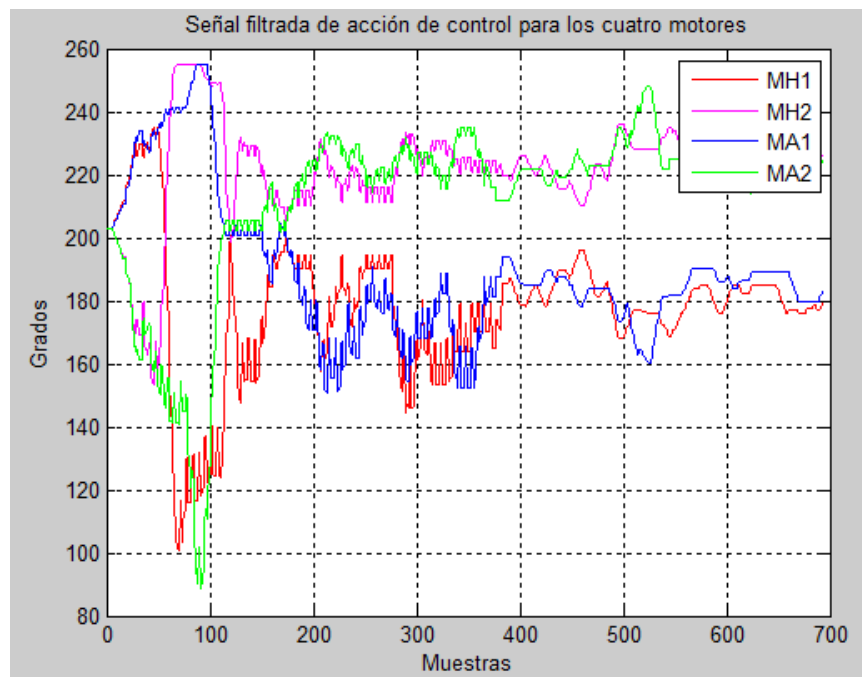


Figura 66 Gráfica de la acción de control aplicada a los cuatro motores

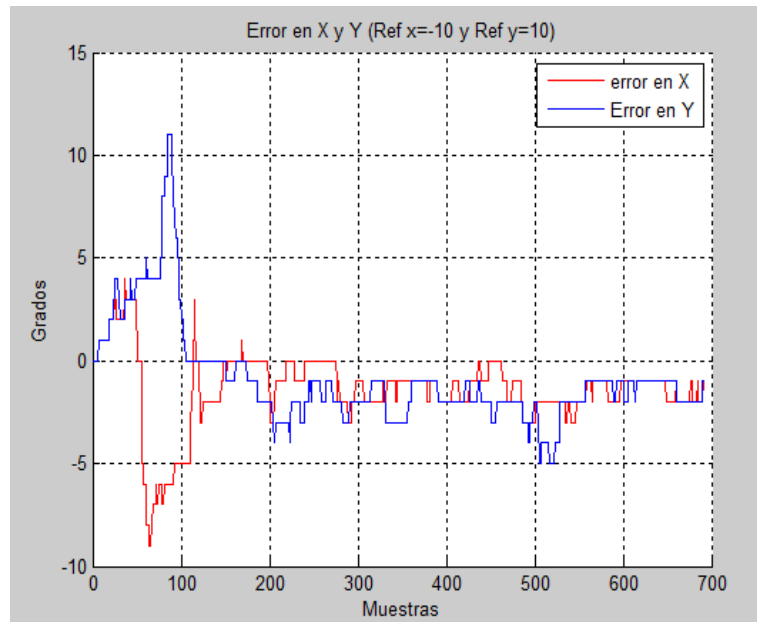


Figura 67. Gráfica del error para ambos ejes, en la cual se muestra un pequeño error marcado por falta de alcance a su referencia marcada.

4.5 Límites de control positivos ($x=30, y=30$)

Los límites son otra de las pruebas que se llevan a cabo, para saber cuál es el alcance del control es decir; hasta que posición se puede llevar un buen control del Cuadricóptero, para ello se realiza una prueba en los ejes positivos, además de esto se toma en cuenta la limitaciones que provoca la base en la cual está montada la estructura del Cuadricóptero. Para realizar las pruebas a los límites, primero se realizan para la parte positiva de cada eje en este caso en $X=30$, $Y=30$. Esta coordenada fue la ideal para que el control obtuviera una respuesta adecuada, en la Figura 69 se puede observar los resultados.

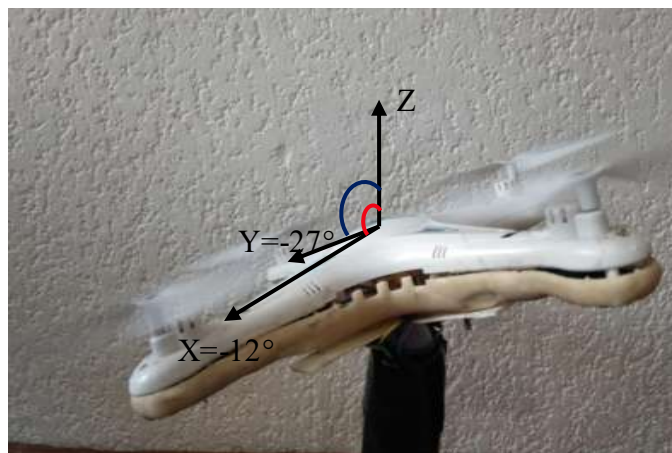


Figura 68 Imagen que muestra al Cuadricóptero en su posición inicial en la cual se toman los datos para la respuesta del controlador con los límites positivos

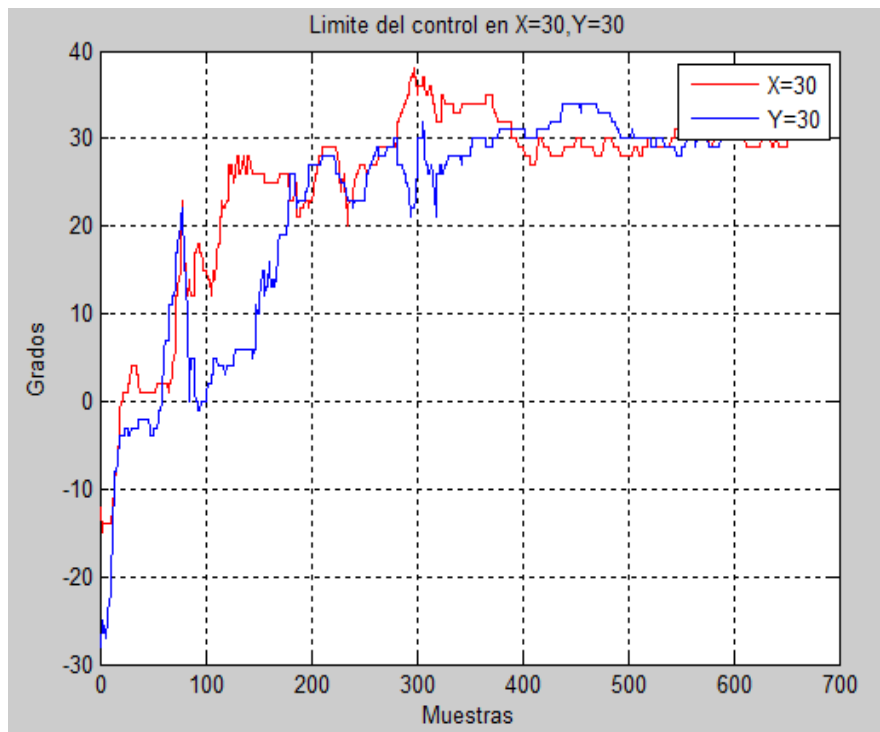


Figura 69. Gráfica en la cual se muestran, los resultados para los límites positivos en donde el control continúa siendo funcional.

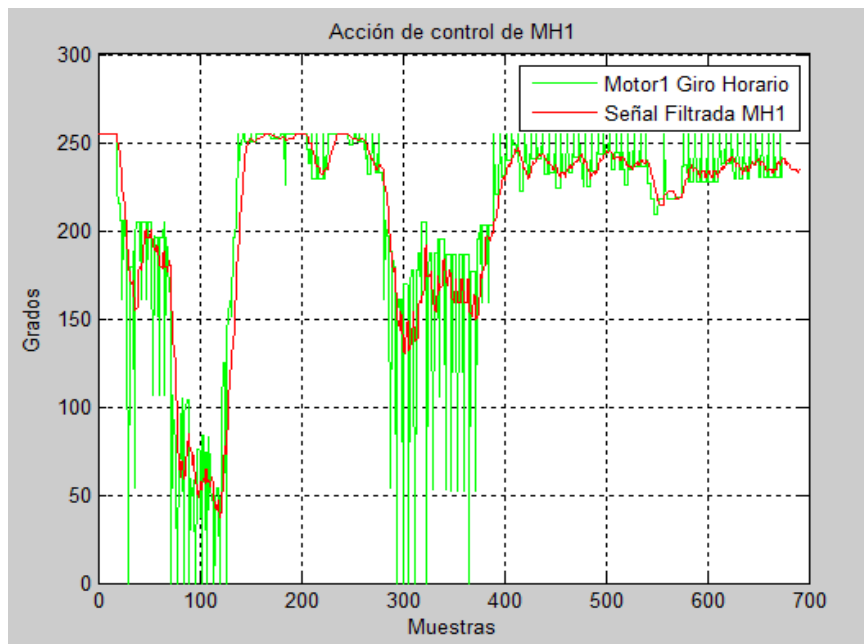


Figura 70. Gráfica de la acción de control aplicada al motor 1 con giro horario.

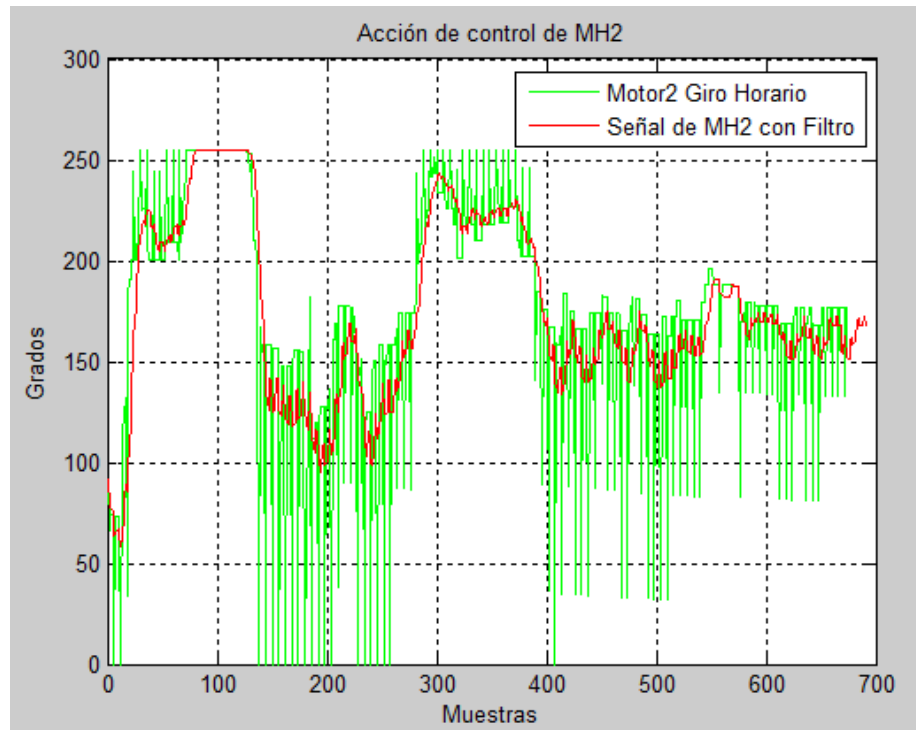


Figura 71. Gráfica de la acción de control para el motor 2 con giro horario.

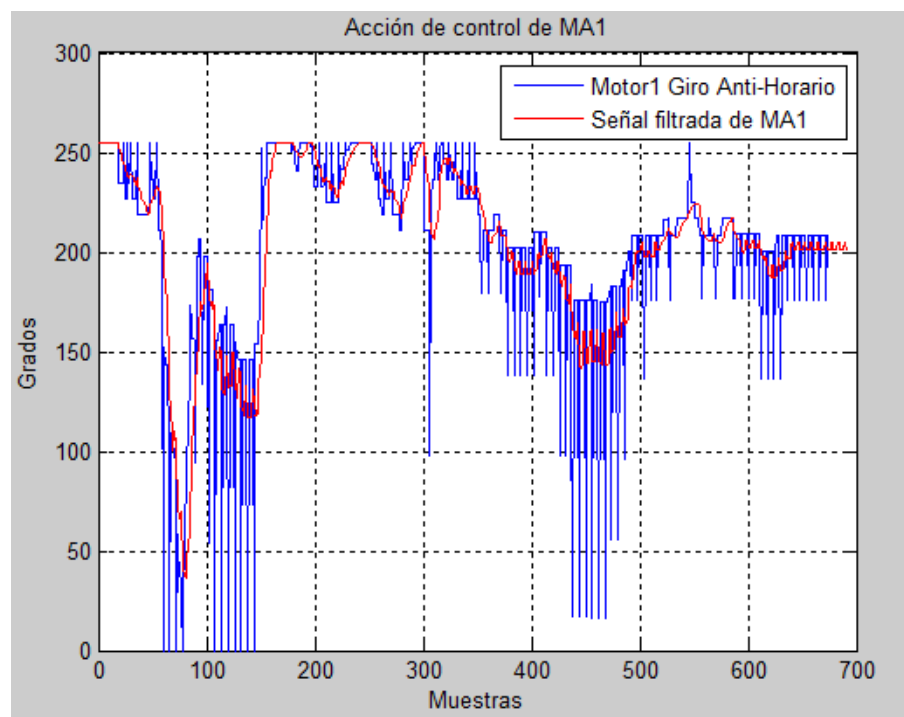


Figura 72. Gráfica de la acción de control para el motor 2 con giro anti-horario.

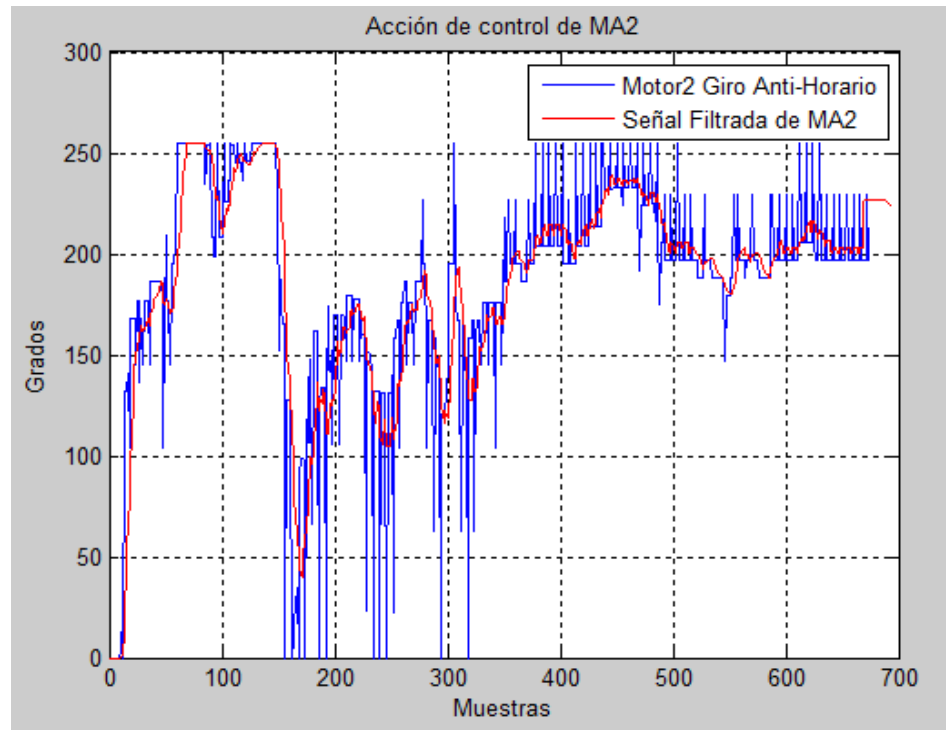


Figura 73. Gráfica de la acción de control para el motor 2 con giro anti-horario.

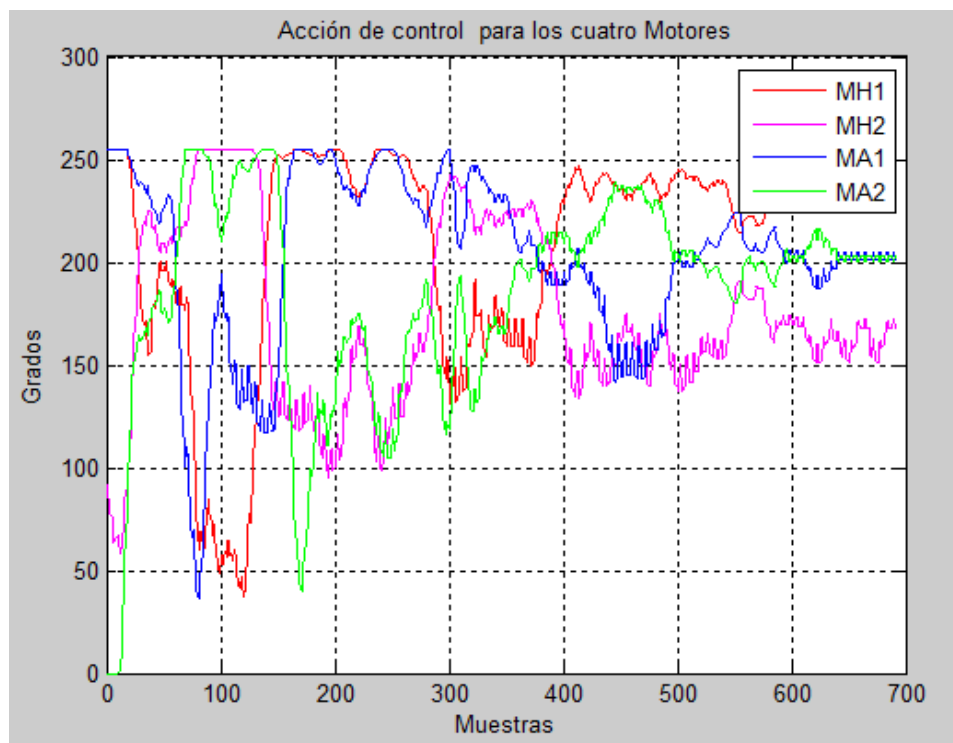


Figura 74. Gráfica de la acción de control, para los cuatro motores.

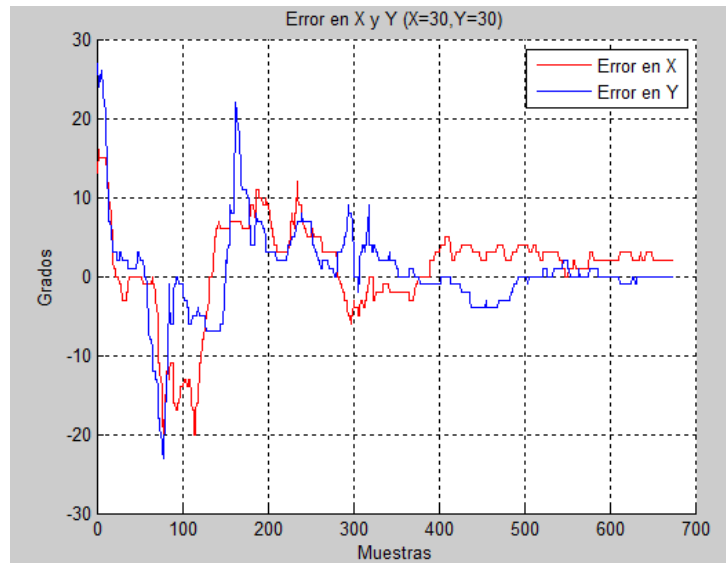


Figura 75. Gráfica del error obtenido para los límites positivos, en el eje X y eje Y.

4.6 Límite de control negativo en $x=-20$, $y=-40$

En el caso del control negativo se puede ver una diferencia mayor entre ambos ejes, esto se debe a que la base no permite una mayor inclinación como por ejemplo en el eje x negativo, que solo permite llegar a una referencia de -20 grados de inclinación, en cambio para el eje y es posible llegar hasta una referencia de -40 grados, como se muestra en la Figura 77

Al intentar llevar el control del Cuadricóptero a valores superiores a los ya mencionados, los motores llegan a sus límites por causa de la base y la estructura que no permiten una mayor inclinación además, el control se ve afectado es decir; el control funciona incorrectamente, porque se intenta llegar a la referencia sin poder hacerlo, por esta razón se marcan los límites en donde el control es funcional.

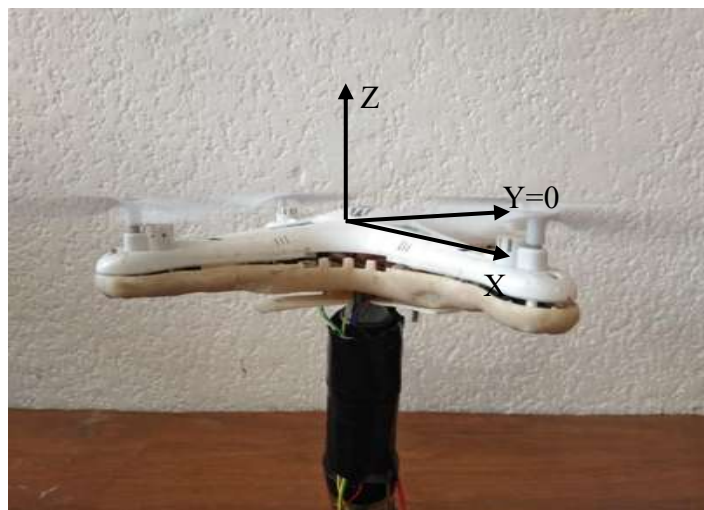


Figura 76 Imagen que muestra al Cuadricóptero en su posición inicial en cual se toman los datos para la respuesta del controlador de los límites negativos.

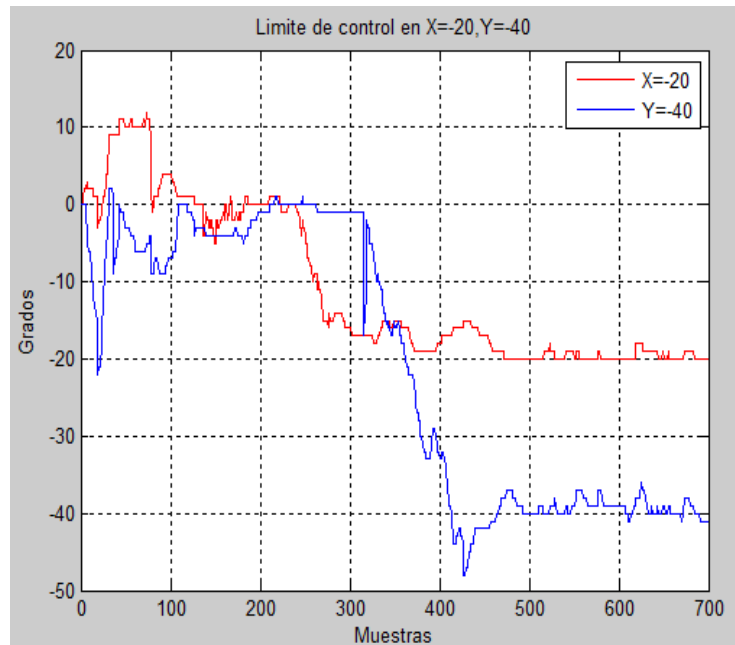


Figura 77. Gráfica de los límites negativos para el eje “x” como para el eje “y” con una referencia inicial de cero grados y con un pequeño sobrepaso en el eje “y” causado por el cambio brusco de 0 a -40 grados.

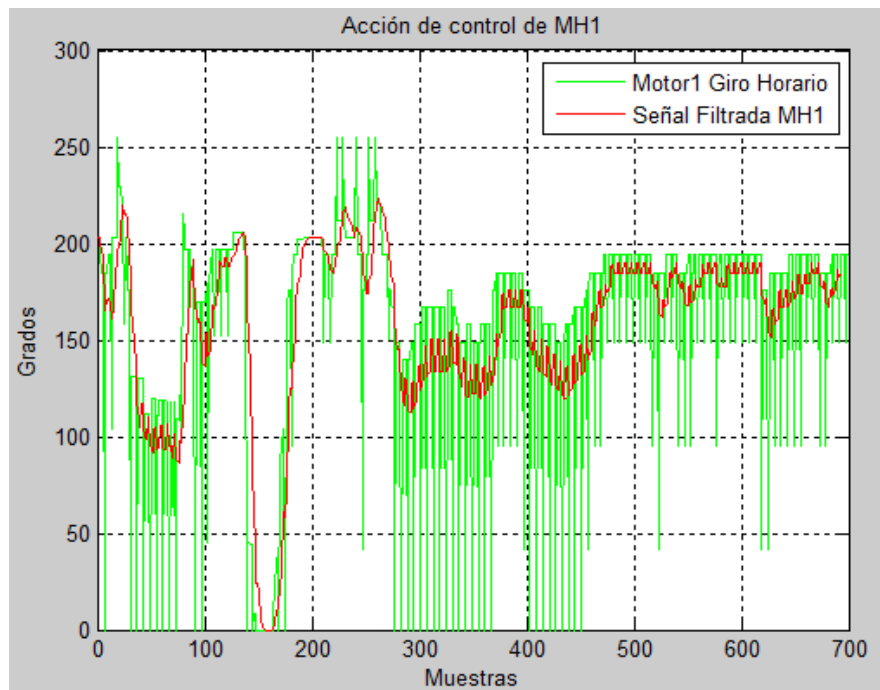


Figura 78. Gráfica de la acción de control para el motor 1 con giro horario.

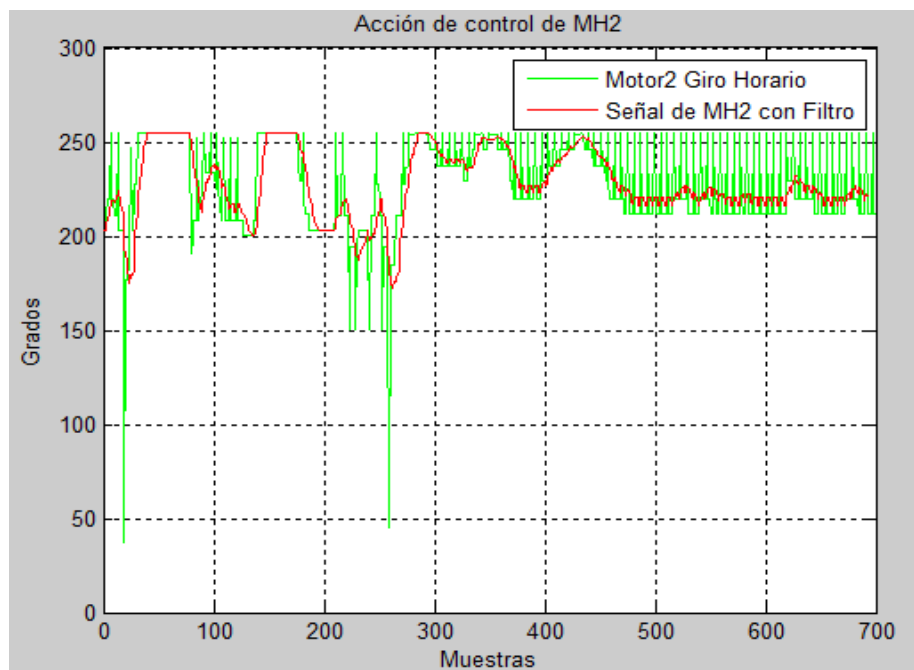


Figura 79. Gráfica de la acción de control para el motor 2 con giro horario.

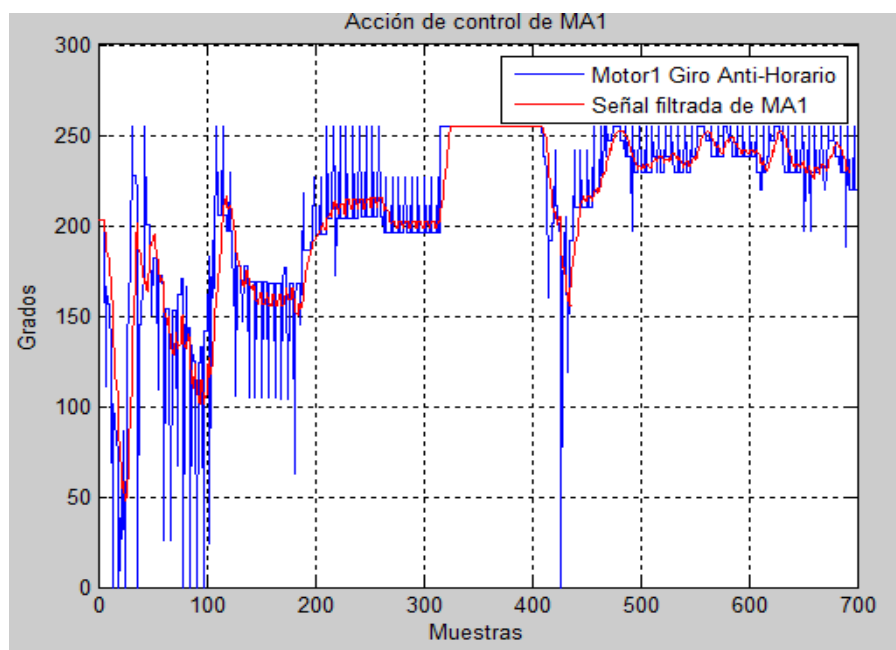


Figura 80. Gráfica de la acción de control para el motor 1 con giro anti-horario.

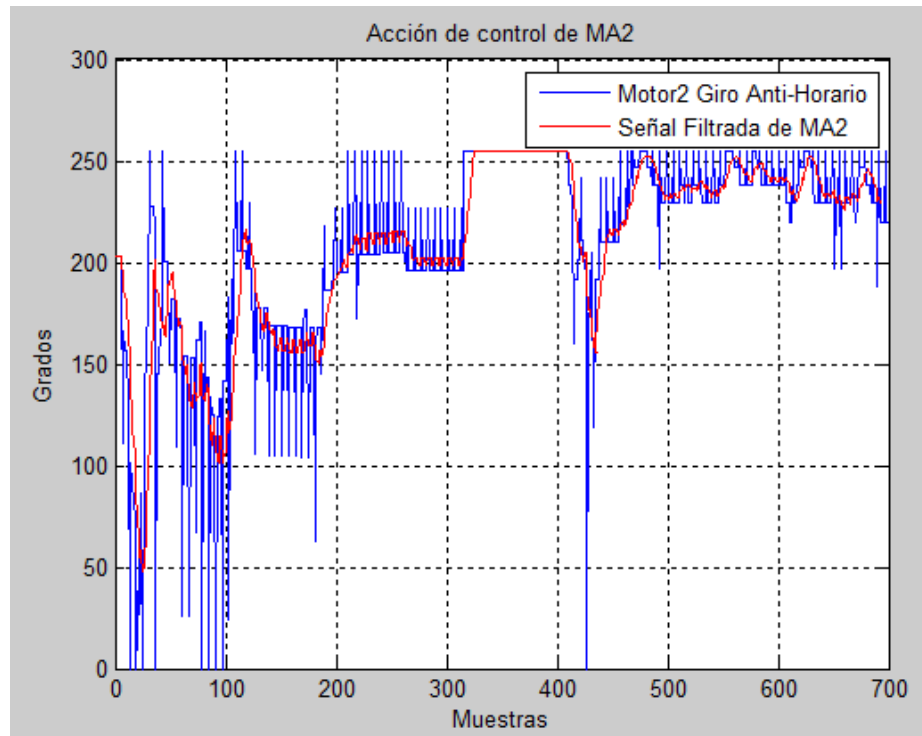


Figura 81. Gráfica de la acción de control para el motor 2 con giro anti-horario.

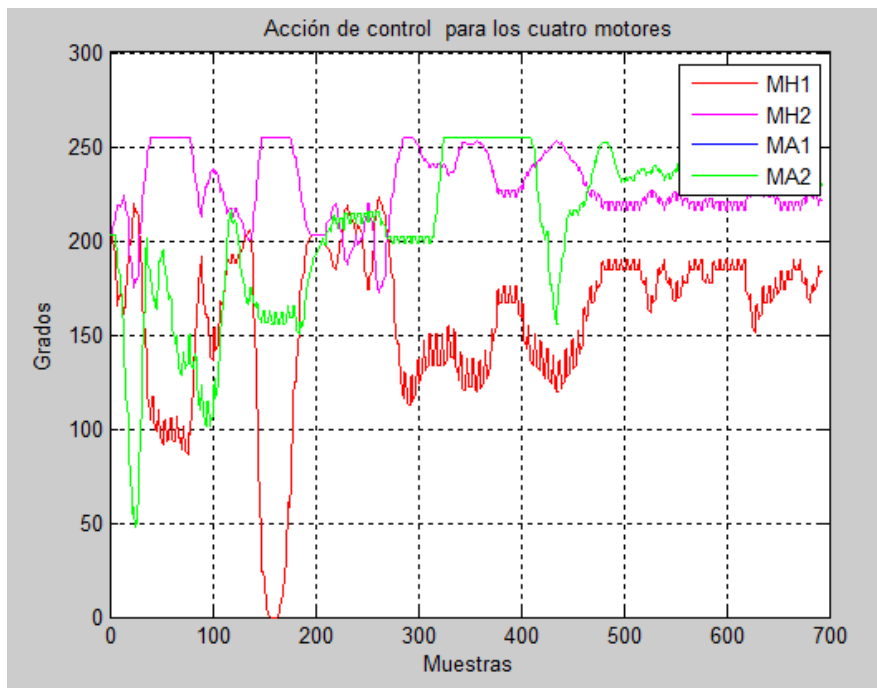


Figura 82. Gráfica de la acción de control, para los cuatro motores y sus límites negativos.

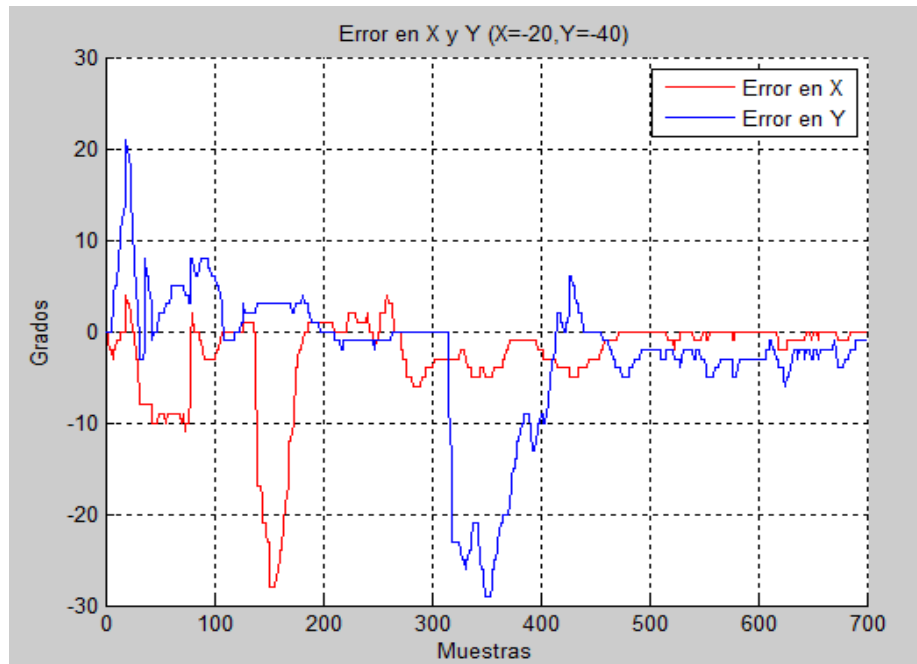


Figura 83. Gráfica del error para el eje x y eje y, en los límites negativos.

4.7 Perturbación Baja

Una de las pruebas que son indispensables en el diseño de controladores, es someterlos a pruebas, en donde se le aplican perturbaciones inesperadas, para esperar que el controlador mantenga su referencia sin ser tan afectado.

Para ello se aplica una pequeña perturbación cuando el Cuadricóptero se encuentra en funcionamiento es decir; en su referencia marcada, esperando como resultado que este oscile lo menos posible intentando mantener su referencia, como se muestra en la Figura 85.

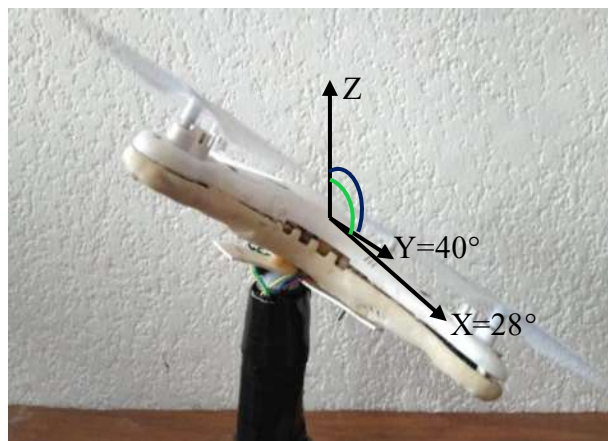


Figura 84 Imagen del Cuadricóptero en su posición inicial, para la prueba de una perturbación baja con referencia en cero grado

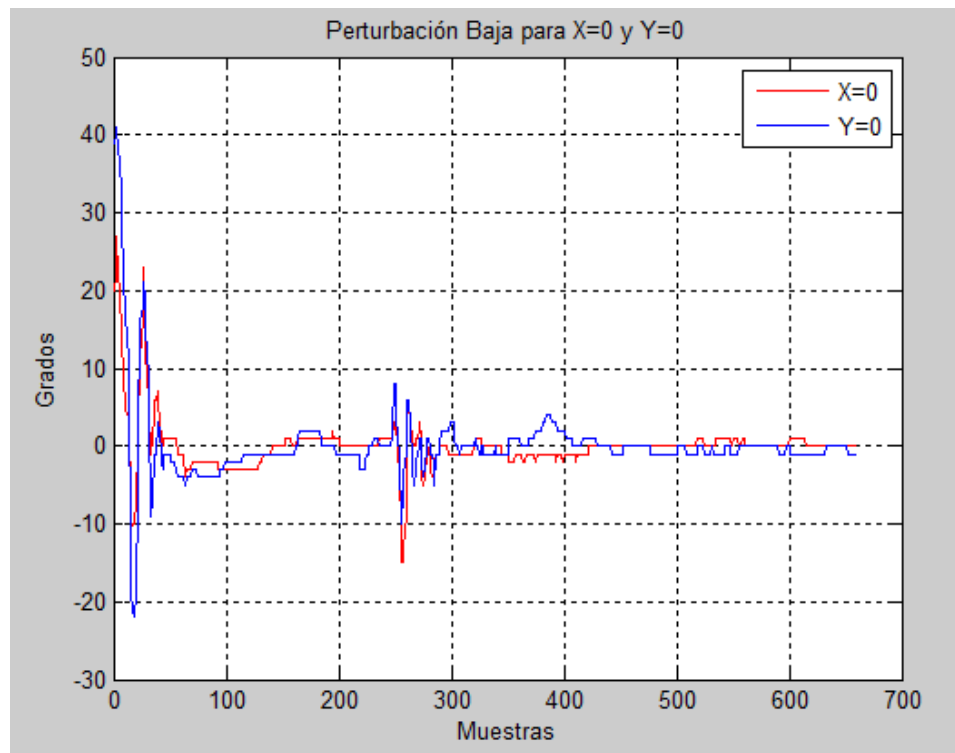


Figura 85 Gráfica de una respuesta a una perturbación baja con una posición inicial en $X=20$, $Y=39$ y una referencia de cero grados ($X=0,Y=0$)

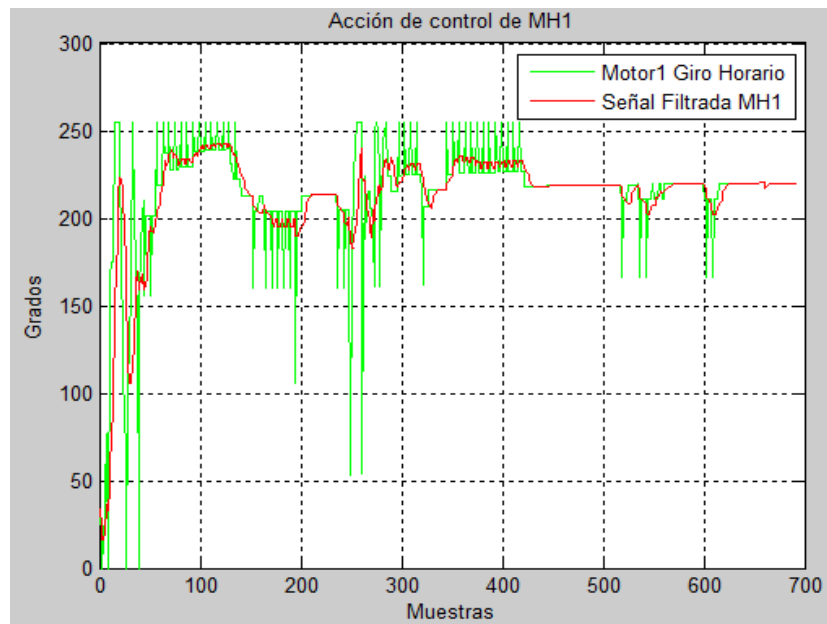


Figura 86. Gráfica de la acción de control para el motor 1 con giro horario.

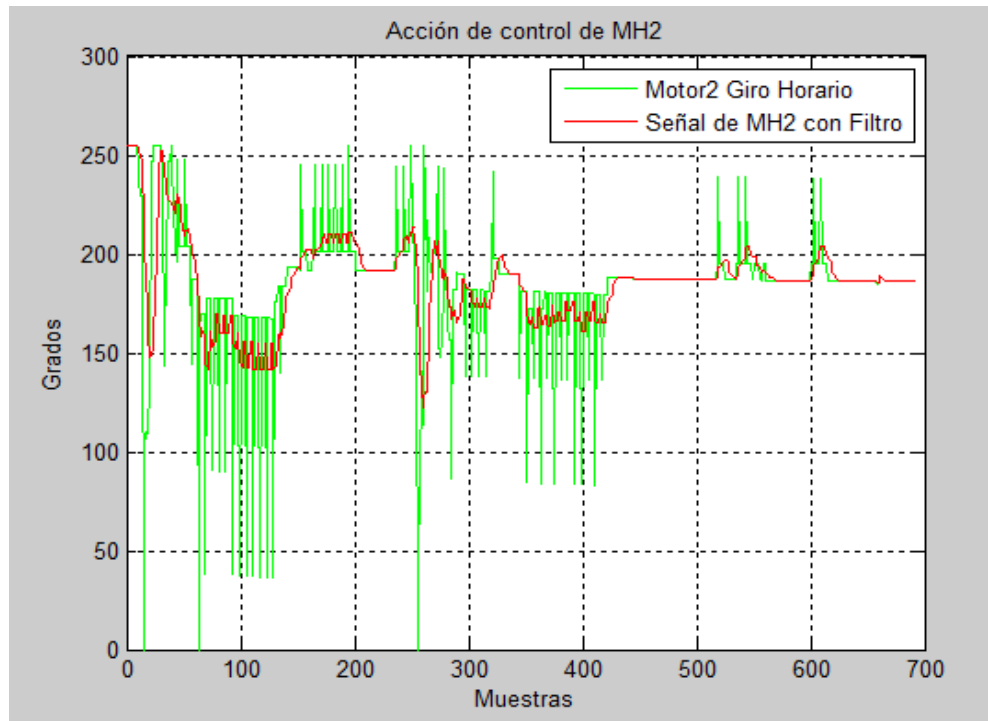


Figura 87. Gráfica de la acción de control para el motor 2 con giro horario.

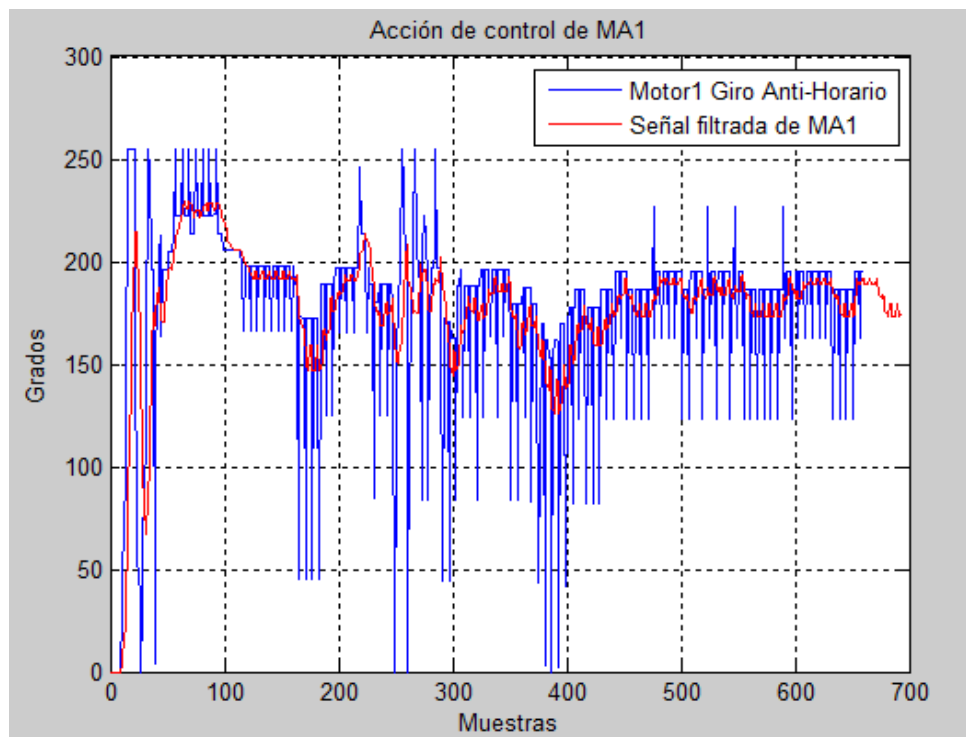


Figura 88. Gráfica de la acción de control para el motor 1 con giro anti- horario.

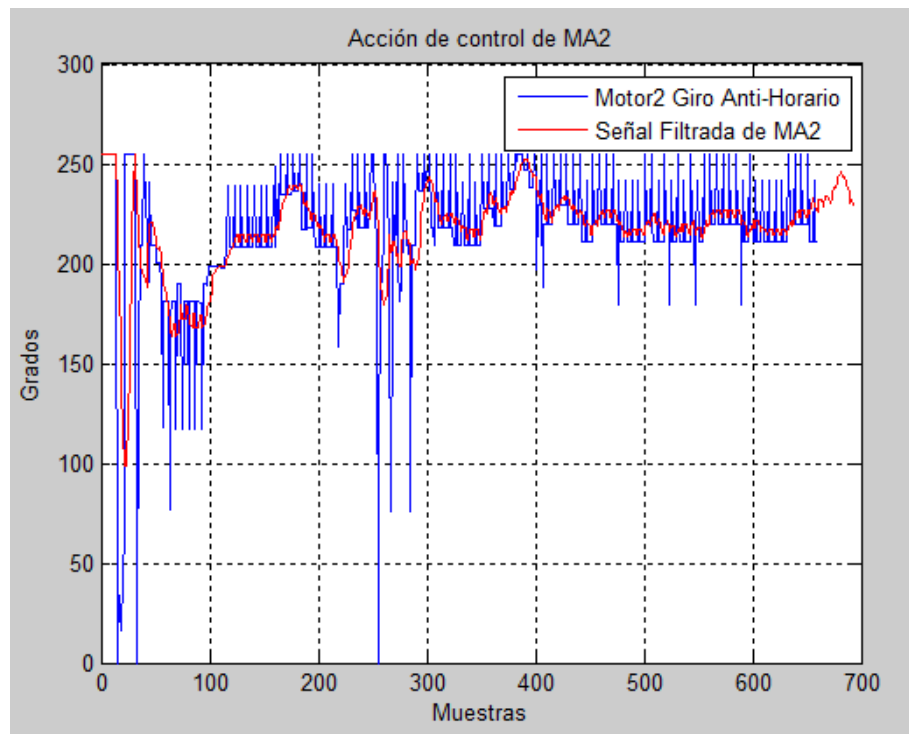


Figura 89. Gráfica de la acción de control para el motor 2 con giro anti-horario.

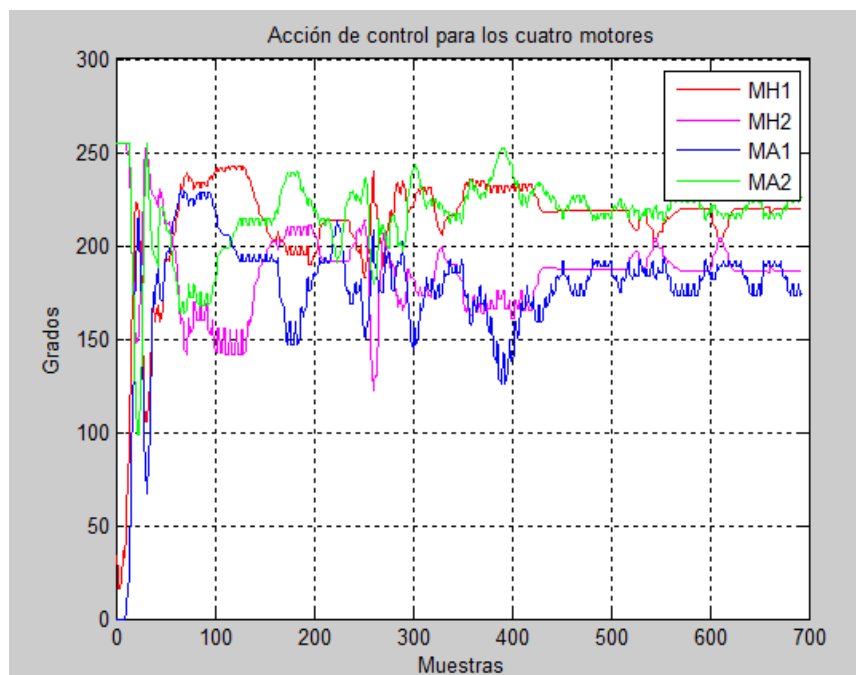


Figura 90. Gráfica de la acción de control para los cuatro motores.

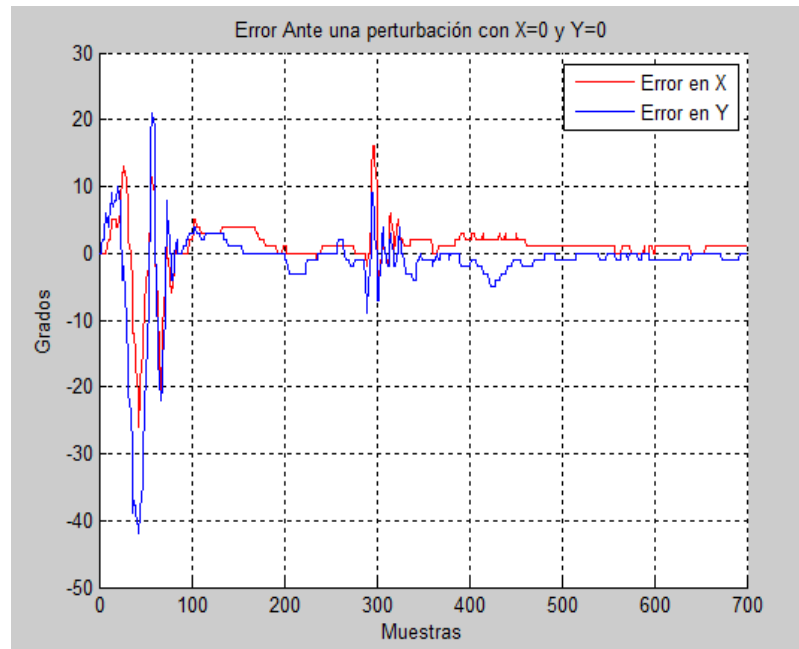


Figura 91 Gráfica del error ante una perturbación baja.

4.8 Perturbación alta

Por último se realiza una prueba aplicando una perturbación más alta a la perturbación anterior (perturbación baja), esto con el fin de mostrar la diferencia de su respuesta al aplicar una perturbación mayor, además de la respuesta del control aplicado en el Cuadricóptero.

Con esta prueba también se puede observar, que es posible retomar su referencia aun afectando su estabilidad, como se puede observar en la Figura 93.

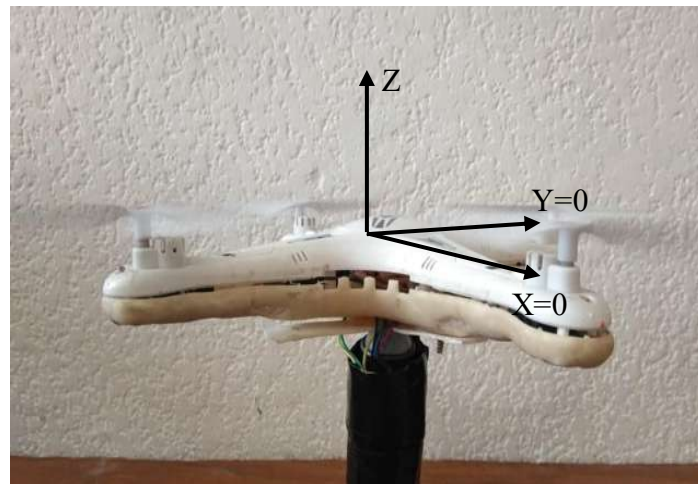


Figura 92 Imagen del Cuadricóptero en su posición inicial, para la prueba de una perturbación alta con referencia en cero grado

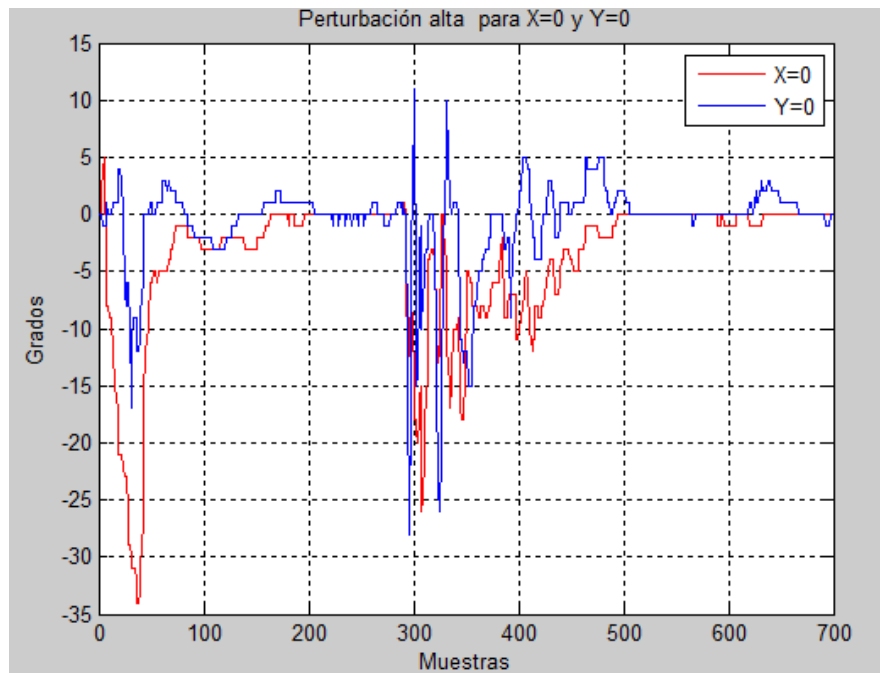


Figura 93. Gráfica de la respuesta ante una perturbación alta y referencia de $X=0$, $Y=0$.

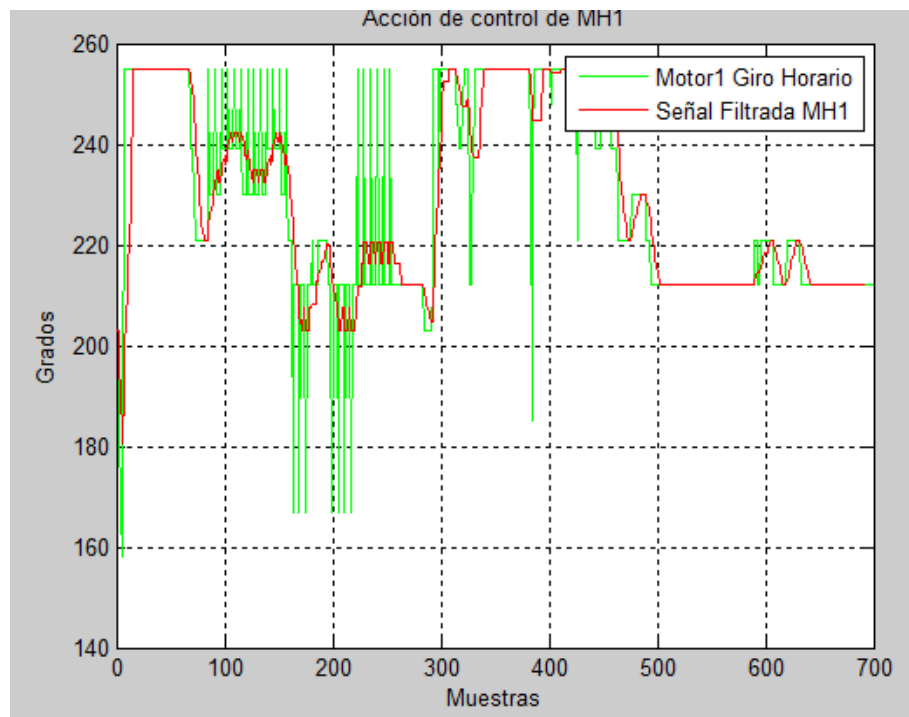


Figura 94. Gráfica de la acción de control para el motor 1 con giro horario.

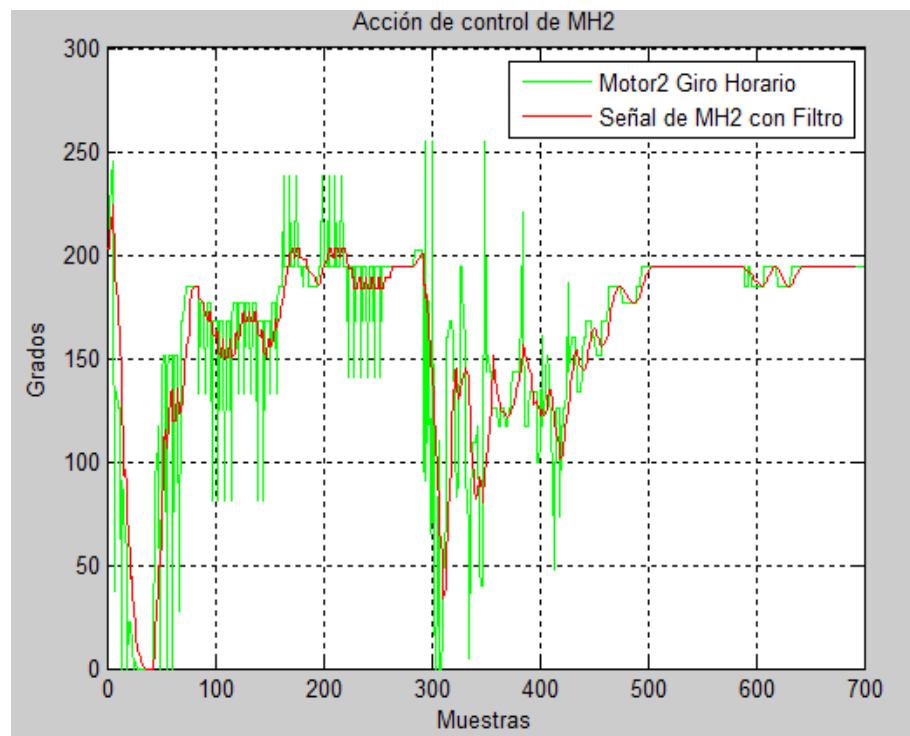


Figura 95. Gráfica de la acción de control para el motor 2 con giro-anti horario.

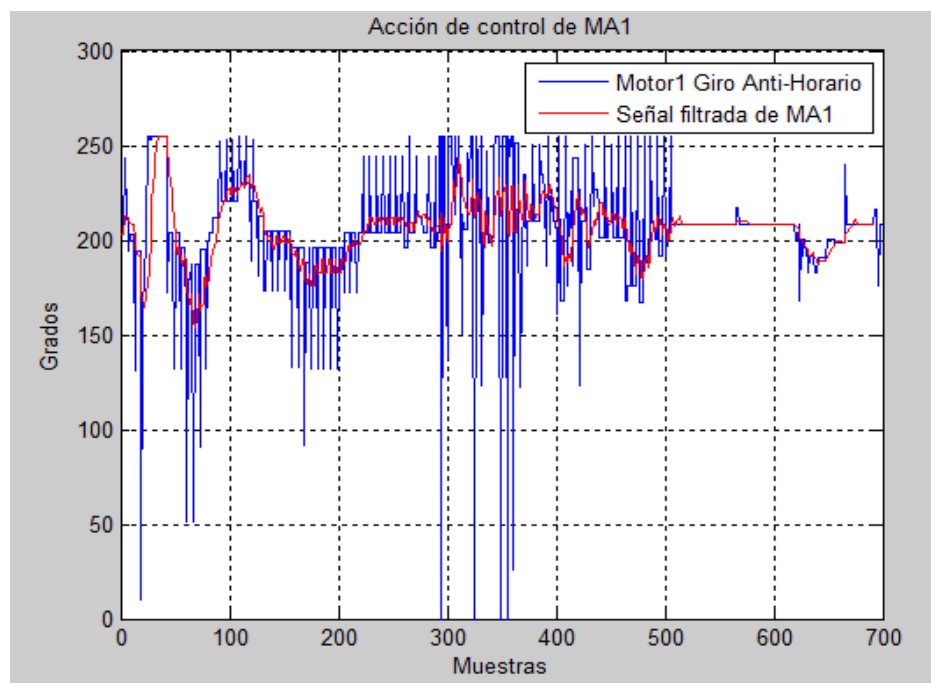


Figura 96. Gráfica de la acción de control para el motor 1 con giro anti - horario

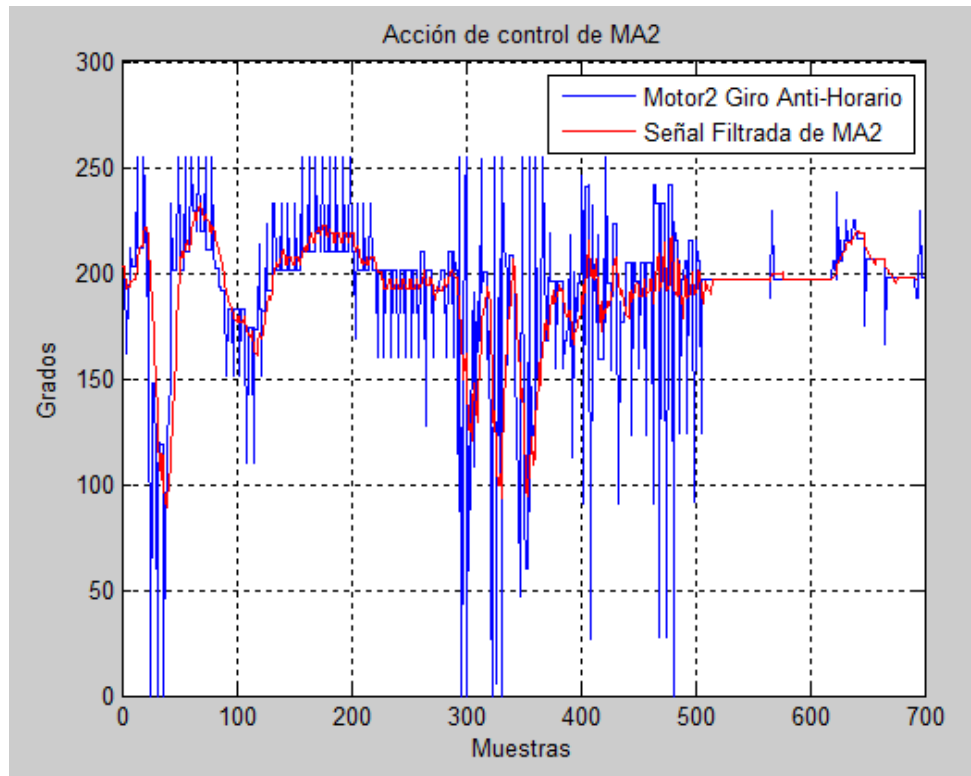


Figura 97. Gráfica de la acción de control para el motor 2 con giro anti- horario

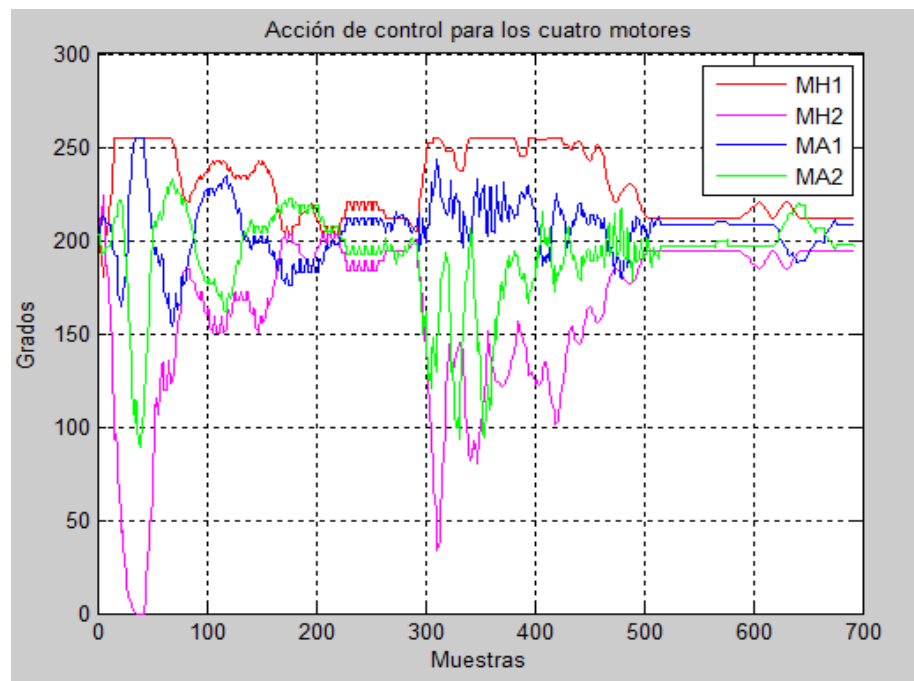


Figura 98. Gráfica de la acción de control para los cuatro motores ante una perturbación alta.

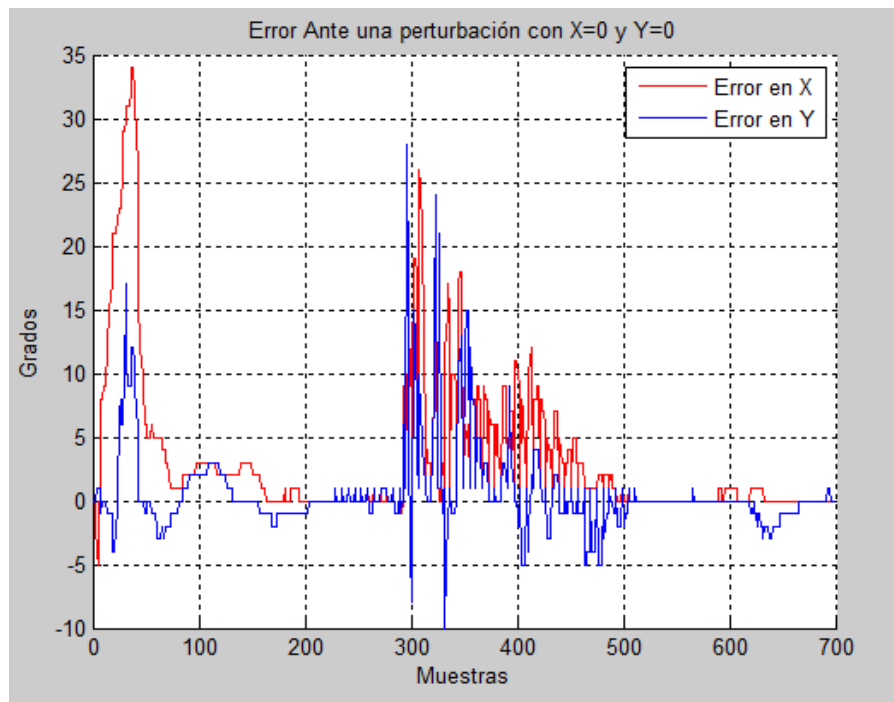


Figura 99. Gráfica del error en ambos ejes, ante una perturbación alta.

Capítulo 5

Conclusiones

5.1 Conclusión

El trabajo y construcción de un Cuadricóptero no es una tarea sencilla, el primer problema es la elección de los componentes, se complica cuando no se tiene un conocimiento básico del funcionamiento de un Cuadricóptero.

Por tal motivos primero se tiene que enfocar en el funcionamiento de un Cuadricóptero, cuando se comprende el funcionamiento es decir; se entiende cuáles son los movimientos que este puede realizar y de qué manera es posible que se lleven a acabo esos movimientos.

Al tener en cuenta lo anterior es posible continuar con la elección de los elementos que constituyen el Cuadricóptero, para ello primero se tiene que elegir el tipo de chasis para poder continuar con la elección de los motores y hélices.

La elección de los motores y hélices se eligen de tal manera que sean capaz de mover el chasis, posterior a ello se toma en cuenta que tipos de driver que es necesario para manejar el motor-hélice. Así como el tipo de microcontrolador y sensor de posición.

Al seleccionar un elemento es necesario, comprobar que este funcione correctamente, para ello se deben realizar las pruebas necesarias, así con cada parte o elemento del Cuadricóptero. Es decir se busca que al realizar y comprobar el resultado sea un buen funcionamiento en el caso de cada uno de sus componentes electrónicos.

Posteriormente se busca distribuir los componentes, es decir colocar cada parte del Cuadricóptero de tal manera que el peso este distribuido es decir; que el peso no se encuentre concentrado solo en un punto del Cuadricóptero al menos que este se encuentre centrado, también buscando que las conexiones se puedan realizar de la manera más sencilla posible.

Un problema que se debe considerar es cuando las pruebas que se realizan a cada uno de los elementos del Cuadricóptero funcionan correctamente por separado y al funcionar en conjunto cambia su funcionamiento. Por ejemplo se presentó problemas con el sensor de posición (MPU6050), cuando se realizaron las primeras mediciones al sensor sin el uso de motores, las mediciones resultaron buenas, pero posteriormente al agregar los motores estos ocasionan vibraciones y ruidos eléctricos que provocaban que el sensor presentara mediciones erróneas, es solo una de las partes que se puede complicar el control, ya que no es fácil saber qué problema causa un mal funcionamiento.

Al concluir con el arreglo de cada elemento del Cuadricóptero se inicia con el diseño del controlador y la comunicación inalámbrica.

El diseño del controlador consiste en aplicar el control a cada uno de los motores, ya que estos tienen un comportamiento distinto ya sea por la posición en la que el Cuadricóptero o por una perturbación o mala distribución de componentes.

Y por último se agrega la comunicación inalámbrica y diseño de la interface gráfica llegando a concluir con las pruebas del control de posición del Cuadricóptero.

5.2 Trabajos futuros

En un trabajo futuro se puede utilizar mejores componentes para la construcción del Cuadricóptero como por ejemplo:

Sensores, motores de mayor potencia, la estructura es una parte importante porque el material puede ser muy propenso a transmitir vibraciones así como su peso puede no ser el adecuado además del tamaño entre otras características.

La conexión inalámbrica es un punto indispensable para el caso del control inalámbrico. Esto porque un cable entorpece el ajuste de las ganancias del controlador. Además la comunicación vía Bluetooth se puede utilizar otro tipo de comunicación como por ejemplo la comunicación WIFI o de Radio Frecuencia.

Tomando en cuenta los detalles mencionados en los capítulos anteriores, como posibles causas de errores en las mediciones, tipos de motores, hélices, capacidad del microcontrolador, maneras de vuelo de un Cuadricóptero, material y diseño del chasis etc.

La elección del chasis se hace según la fuerza de impulso de los motores, es decir el material del Cuadricóptero debe de ser ligero y con una facilidad de movimiento.

También como se ha mencionado al realizar pruebas por separado de los componentes, es indispensable hacerlo nuevamente pero ya en conjunto, para detectar problemas.

En el caso del diseño de la interface y comunicación inalámbrica, lo recomendable es utilizar o diseñar una interface que permita una buena comunicación entre la PC y el microcontrolador, así como el fácil manejo diseño y manejo de la interface gráfica (ambiente amigable).

Bibliografía

- [1] D. d. c. d. l. U. d. V. IRTIC, «Origen y desarrollo de los drones,» 9 7 2015. [En línea]. Available: <http://drones.uv.es/origen-y-desarrollo-de-los-drones/>.
- [2] kimerius.com/app/download/5784118485/Tipos+de+aeronaves.pdf. [En línea]. Available: kimerius.com/app/download/5784118485/Tipos+de+aeronaves.pdf.
- [3] rfmartiezo, «<https://github.com>,» 23 Jun 2015. [En línea]. Available: https://github.com/nicosquare/digital_ii/wiki/3.-Dinámica-del-cuadricóptero.
- [4] S. M. Díaz, «Diseño y construccion de un Quadricopter,» [En línea]. Available: <https://upcommons.upc.edu/bitstream/handle/2099.1/21902/102664.pdf?sequence=1&isAllowed=y>.
- [5] V.Garcia, «Hisipavila,» 27 08 2014. [En línea]. Available: <https://www.hispavila.com/el-puente-h/>.
- [6] M. R. G. y. C. B. David Martinez, «Quadrino.com,» [En línea]. Available: <http://www.quadrino.com/guia-2/materiales-necesarios-1/esc>.
- [7] V. R. Vicedo, «Universidad Politecnica de Valencia,» 2014. [En línea]. Available: https://riunet.upv.es/bitstream/handle/10251/50184/TFG%20Victor%20Ramos%20Vicedo_14043889009892250550389307375467.pdf?sequence=2.
- [8] C. A.Reyes, Microcontroladores programacion en Basic Segunda Edicion, Quito Ecuador: Rispergraf, 2006.
- [9] V. Garcia, «Electronica Practica Aplicada,» 10 Abril 2013. [En línea]. Available: <https://www.diarioelectronicohoy.com/blog/controladores-basicos-drivers>.
- [10] FpvMax, «<http://fpvmax.com>,» 3 Febrero 2017. [En línea]. Available: <https://www.youtube.com/watch?v=a9A23eyBu2s&t=71s>.
- [11] J_RPM, «https://www.youtube.com/channel/UCbHXxa7a84fsFGTNYt8EXtQ?sub_confirmation=1,» 15 Julio 2015. [En línea]. Available: <https://www.youtube.com/watch?v=qAqBTACXFPs&t=1010s>.
- [12] C. d. m. HC-05. [En línea]. Available: http://www.naylampmechatronics.com/blog/24_configuracion-del-modulo-bluetooth-hc-05-usa.html.

Anexo A

```
#include <avr/io.h>

#include <avr/interrupt.h>

#include "I2Cdev.h"

#include "MPU6050.h"

#include "Wire.h"

signed int  filtrox( float nuevox);

signed int  filtroy( float nuevoy);

// La dirección del MPU6050 puede ser 0x68 o 0x69, dependiendo
// del estado de AD0. Si no se especifica, 0x68 estará implícito

MPU6050 sensor;

////////////////Variables para comunicacion serial PC-Arduino////////////////

char cadena[30]; //Creamos un array que almacenará los caracteres que escribiremos en la consola del PC. Le asignamos un tope de
caracteres, en este caso 30

byte posicion=0; //Variable para cambiar la posición de los caracteres del array

int valor; //Variable del valor entero

////////////////////////////////////

// Valores RAW (sin procesar) del acelerómetro y giroscopio en los ejes x,y,z

int ax, ay, az;

int gx, gy, gz;

long tiempo_prev;

float dt;

float ang_x, ang_y;

float ang_x_prev, ang_y_prev;

//int pulsosLED = 13; //Indicador de interrupcion

////////DECLARACION DE SALIDAS PWM////////

const int H1 = 5; //Pin de Señal PWM para motor H1

const int H2 = 6; //Pin de Señal PWM para motor H2

const int Anti1 = 3; //Pin de Señal PWM para motor A1

const int Anti2 = 11; //Pin de Señal PWM para motor A2

////////////////////////////////////

const int MUESTRAS=300;

////////////////////////////////////

////////////////////////////////////

signed int Rx=0;
```

```

signed int Ry=0;

int Resultadox=0;

int Resultadoy=0;

int AnguloNewx=0;

int AnguloNewy=0;

int PH;

int NH;

int PA;

int NA;

//////////REFERENCIA//////////

int RefH=0;

int RefA=0;

//*****

signed int Ref = 0;

float RefAnt=0;

signed int Eje=0;

signed int x=0,y=0;

//*****

////////////////////

int MotorH1=0;

int MotorH2=0;

int MotorA1=0;

int MotorA2=0;

//////////ERROR//////////

int errorH1=0;

int errorH2=0;

int errorA1=0;

int errorA2=0;

int errorpassH1=0;

int errorpassH2=0;

int errorpassA1=0;

int errorpassA2=0;

int errorantH1=0;

int errorantH2=0;

int errorantA1=0;

int errorantA2=0;

```

```

//////////ACCION DE CONTROL//////////

float UH1=0;

float UH2=0;

float UA1=0;

float UA2=0;

int UantH1=0;

int UantH2=0;

int UantA1=0;

int UantA2=0;

int UnH1=0;

int UnH2=0;

int UnA1=0;

int UnA2=0;

////////////////////////////////////

////////// //GANANCIAS//////////

float KpH1=11.2;

float KpH2=10.7;

float KpA1=10.5;

float KpA2=11.5;

float KiH1=0.00036;

float KiH2=0.00035;

float KiA1=0.00031;

float KiA2=0.00031;

float KdH1=56;

float KdH2=55;

float KdA1=40;

float KdA2=40;

////////////////////////////////////

void setup() {

//pinMode (pulsosLED,OUTPUT);

pinMode (H1,OUTPUT);

pinMode (H2,OUTPUT);

pinMode (Anti1,OUTPUT);

pinMode(Anti2,OUTPUT);

Serial.begin(115200); //Iniciando puerto serial

TCCR2B = (TCCR2B & 0b1111000) | 0x01; //Timer22 configuracion

```

```

cli();

TCCR1A=0;

TCCR1B=0;

TIMSK1 = (1<<TOIE1);

TCCR1B|=(1 << CS10);

sei();

Wire.begin();      //Iniciando I2C

sensor.initialize(); //Iniciando el sensor

if (sensor.testConnection())// Serial.println("Sensor iniciado correctamente");

// else Serial.println("Error al iniciar el sensor");

}

////////////////////void LOOP////////////////////////////////////

void loop() {

// Leer las aceleraciones y velocidades angulares

sensor.getAcceleration(&ax, &ay, &az);

sensor.getRotation(&gx, &gy, &gz);

dt = (millis()-tiempo_prev)/1000.0;

tiempo_prev=millis();

//Calcular los ángulos con acelerometro

float accel_ang_x=atan(ay/sqrt(pow(ax,2) + pow(az,2)))*(180.0/3.14);

float accel_ang_y=atan(-ax/sqrt(pow(ay,2) + pow(az,2)))*(180.0/3.14);

//Calcular angulo de rotación con giroscopio y filtro complemento

ang_x = 0.96*(ang_x_prev+(gx/131)*dt) + 0.04*accel_ang_x;

ang_y = 0.96*(ang_y_prev+(gy/131)*dt) + 0.04*accel_ang_y;

ang_x_prev=ang_x;

ang_y_prev=ang_y;

if(Serial.available()){

Ref = Serial.read();

if(Ref >= 3) && (Ref <255)){

RefAnt = map (Ref,3,255,-90,92);

Eje =RefAnt;

}

if (Ref == 1){

x=Eje;

}

}

```

```

        if (Ref == 2){
            y=Eje;
        }
Serial.print(AnguloNewx);

Serial.print(',');

Serial.print(AnguloNewy);

    Serial.print(',');

Serial.print(MotorH1);

Serial.print(',');

Serial.print(MotorH2);

    Serial.print(',');

Serial.print(MotorA1);

    Serial.print(',');

Serial.print(MotorA2);

    Serial.print(',');

Serial.print(errorH1);

    Serial.print(',');

Serial.println(errorA1);


delay(170);

//////////////////////////////////Fin de Void Loop//////////////////////////////////

}

ISR(TIMER1_OVF_vect){

Rx=0;

Ry=0;

Rx=ang_x - x;

Ry=ang_y - y;

Resultadox = filtrox(Rx);

Resultadoy = filtroy(Ry);

AnguloNewx = Resultadox + x;

AnguloNewy = Resultadoy + y;

////PARA CALCULAR EL ERROR Y SU COMPLEMENTO EN "X" y "Y"////

PH=Resultadox;

NH=-Resultadox ;

PA=Resultadoy;

NA=-Resultadoy;

```

```

errorH1 = RefH - PH;

errorH2 = RefH - NH;

errorA1 = RefA - PA;

errorA2 = RefA - NA;

////////////////////////////////////

////////ACUMULADOR DE ERROR PARA LA PARTE INTEGRAL////////

errorpassH1 = errorH1+ errorpassH1;

errorpassH2 = errorH2+ errorpassH2;

errorpassA1 = errorA1+ errorpassA1;

errorpassA2 = errorA2+ errorpassA2;

////////////////////////////////////

////////////////////////////////////

    if (errorpassH1 < -32000){

        errorpassH1 = -32000;

    }

    if (errorpassH2 < -32000){

        errorpassH2 = -32000;

    }

    if (errorpassA1 < -32000){

        errorpassA1 = -32000;

    }

    if (errorpassA2 < -32000){

        errorpassA2 = -32000;

    }

////////////////////////////////////

////////////////////////////////////

    if (errorpassH1 > 32000){

        errorpassH1 = 32000;

    }

    if (errorpassH2 > 32000){

        errorpassH2 = 32000;

    }

    if (errorpassA1 > 32000){

        errorpassA1 = 32000;

    }

    if (errorpassA2 > 32000){

```

```

    errorpassA2 = 32000;

}

////////////////////////////////////
////////////////////////////////////ERROR DIFERENCIAL////////////////////////////////////

double errorDH1 = errorH1 - errorantH1;

double errorDH2 = errorH2 - errorantH2;

double errorDA1 = errorA1 - errorantA1;

double errorDA2 = errorA2 - errorantA2;

////////////////////////////////////
////////////////////////////////////

float PH1 = KpH1*errorH1;

float IH1 = KiH1*errorpassH1;

float DH1 = KdH1*errorDH1;

float PH2 = KpH2*errorH2;

float IH2 = KiH2*errorpassH2;

float DH2 = KdH2*errorDH2;

float PA1 = KpA1*errorA1;

float IA1 = KiA1*errorpassA1;

float DA1 = KdA1*errorDA1;

float PA2 = KpA2*errorA2;

float IA2 = KiA2*errorpassA2;

float DA2 = KdA2*errorDA2;

////////////////////////////////////
////////////////////////////////////

UH1=PH1+IH1+DH1;

UH2=PH2+IH2+DH2;

UA1=PA1+IA1+DA1;

UA2=PA2+IA2+DA2;

////////////////////////////////////
////////////////////////////////////

errorantH1 = errorH1;

errorantH2 = errorH2;

errorantA1 = errorA1;

errorantA2 = errorA2;

////////////////////////////////////
////////////////////////////////////

```

```

if (UH1 < -255){
UH1 = -255;
}
if (UH2 < -255){
UH2 = -255;
}
if (UA1 < -255){
UA1 = -255;
}
if (UA2 < -255){
UA2 = -255;
}

////////////////////////////////
////////////////////////////////

if(UH1 > 255){
UH1 = 255;
}
if(UH2 > 255){
UH2 = 255;
}
if(UA1 > 255){
UA1 = 255;
}
if(UA2 > 255){
UA2 = 255;
}

////////////////////////////////
////////////////////////////////

MotorH1=((((UH1 + 255)/(320)))*255;
MotorH2=((((UH2 + 255)/(320)))*255;
MotorA1=((((UA1 + 255)/(320)))*255;
MotorA2=((((UA2 + 255)/(320)))*255;

////////////////////////////////

if(MotorH1 > 255){
MotorH1 = 255;
}

```



```

    if(MotorH2 > 255){
MotorH2 = 255;
    }

    if(MotorA1 > 255){
MotorA1 = 255;
    }

    if(MotorA2 > 255){
MotorA2 = 255;
    }

////////////////////////////////

analogWrite(H1,MotorH2);
analogWrite(H2,MotorH1);
analogWrite(Anti1,MotorA2);
analogWrite(Anti2,MotorA1);

////////////////////////////////

}

/*****

filtro() Elimina el ruido de las entradas al sistema

*****/

signed int filtrox(float nuevox)
{
    static unsigned char cont_error=0,cont_error1=0;

    static signed int suma[MUESTRAS];

    signed int suma_en;

    float resultx;

    unsigned int i;

    if(cont_error < MUESTRAS)
    {
        suma[cont_error++]=nuevox; /* Almacena el nuevo dato en el arreglo */

        for(i=0,suma_en=0;i<cont_error;i++)

            suma_en=suma_en+suma[i]; /* Suma todos los elementos del arreglo */

        resultx=suma_en/cont_error; /* Obtiene el promedio */
    }

    else

    {

        for(i=0;i<MUESTRAS-1;i++) /* Desplaza todos los elementos del arreglo */

```

```

        suma[i]=suma[i+1];

suma[i]=nuevox; /* Inserta el elemento nuevo */

        for(i=0,suma_en=0;i<MUESTRAS;i++) /* Hace la suma de los elementos */

            suma_en=suma_en+suma[i];

        resultx=suma_en/MUESTRAS; /* obtiene el resultado */

    }

    return resultx; /* Regresa el resultado */

}

/*****

filtro() Elimina el ruido de las entradas al sistema

*****/

signed int filtroy(float nuevoy)

{

    static unsigned char cont_error=0,cont_error1=0;

    static signed int suma[MUESTRAS];

    signed int suma_en;

    float resulty;

    unsigned int i;

    if(cont_error < MUESTRAS)

    {

        suma[cont_error++]=nuevoy; /* Almacena el nuevo dato en el arreglo */

        for(i=0,suma_en=0;i<cont_error;i++)

            suma_en=suma_en+suma[i]; /* Suma todos los elementos del arreglo */

        resulty=suma_en/cont_error; /* Obtiene el promedio */

    }

    else

    {

        for(i=0;i<MUESTRAS-1;i++) /* Desplaza todos los elementos del arreglo */

            suma[i]=suma[i+1];

        suma[i]=nuevoy; /* Inserta el elemento nuevo */

        for(i=0,suma_en=0;i<MUESTRAS;i++) /* Hace la suma de los elementos */

            suma_en=suma_en+suma[i];

        resulty=suma_en/MUESTRAS; /* obtiene el resultado */

    }

    return resulty; /* Regresa el resultado */

}

```

Anexo B

```
import processing.serial.*;
import controlP5.*;
import meter.*;

PrintWriter output;

int xPos=1;

int ux,uy,contadorx=1,contadory=1;

float inByte1;

float inByte2;

int graficax,graficay;

int p;

int a = 396/18;

int value2;

int Push;

int Push2;

float R1;

float R2;

int Ref1;

int Ref2;

StringList recordData;

int contadorxt=0;

Serial myPort;

ControlP5 cp5;

Meter m, m2;

//***** INICIO SETUP *****

//*****

void setup(){

  output = createWriter("Angulo.txt");

  //recordData = new StringList();

  size(1350,700);

  ux=30;

  uy=500;
```

```

println(Serial.list());

// myPort = new Serial(this,"COM40",115200);

myPort = new Serial(this,"COM11",115200);


//***** METER EJE X *****/

//*****

m = new Meter(this,0,410);


m.setTitleFontSize(20);
m.setTitleFontName("Arial bold");
m.setTitle("Eje 'X' en grados°");


String [] scaleLabels={"-90","-80","-70","-60","-50","-40","-30","-20","-10","0","10","20","30","40","50","60","70","80","90"};
m.setScaleLabels(scaleLabels);
m.setScaleFontSize(18);
m.setScaleFontName("Times new roman bold");
m.setScaleFontColor(color(0,255,0));


m.setDisplayDigitalMeterValue(true);
m.setArcColor(color(0,255,0));
m.setArcThickness(10);
m.setMinScaleValue(-90);
m.setMaxScaleValue(90);
m.setMinInputSignal(-90);
m.setMaxInputSignal(90);
m.setNeedleThickness(3);

//***** END METER EJE X *****/

//*****

//***** METER EJE Y *****/

//*****

m2 = new Meter(this,400,410);
m2.setTitleFontSize(20);
m2.setTitleFontName("Arial bold");
m2.setTitle("Eje 'Y' en grados°");


String [] scaleLabels2={"-90","-80","-70","-60","-50","-40","-30","-20","-10","0","10","20","30","40","50","60","70","80","90"};

```

```

m2.setScaleLabels(scaleLabels2);

m2.setScaleFontSize(18);

m2.setScaleFontName("Times new roman bold");

m2.setScaleFontColor(color(0,255,0));


m2.setDisplayDigitalMeterValue(true);

m2.setArcColor(color(0,255,0));

m2.setArcThickness(10);

m2.setMinScaleValue(-90);

m2.setMaxScaleValue(90);

m2.setMinInputSignal(-90);

m2.setMaxInputSignal(90);


m2.setNeedleThickness(3);


//***** END METER EJE X *****

//*****

//***** BOTON-GRAFICA EJE Y *****

//*****

cp5 = new ControlP5(this);

cp5.addButton("EjeY").setValue(0)


        .setPosition(ux+650 + 300,uy)

        .setSize(100,50)

        .setColorForeground(color(255,0,0))

        .setColorBackground(color(100,0,200))

        .setColorActive(color(100,0,0))

        .setColorLabel(color(255,255,255));


//***** END BOTON Y *****

//*****

//***** BOTON-GRAFICA EJE X *****

//*****

cp5 = new ControlP5(this);

cp5.addButton("EjeX").setValue(0)

```

```

        .setPosition(ux+820,uy)

        .setSize(100,50)

        .setColorForeground(color(155,0,0))

        .setColorBackground(color(100,0,200))

        .setColorActive(color(100,0,0))

        .setColorLabel(color(255,255,255));

//***** END BOTON X *****

//*****

        //setColorActive(color(255,255,255));

//***** SLIDER EJE Y *****

//*****

cp5.addSlider("Y")

        .setPosition(1280,300)

        .setSize(50,250)

        .setRange(-90,90)

        .setColorBackground(color(0,0,255))

        .setColorForeground(color(0,255,0))

        .setColorValue(color(255,255,255))

        .setColorActive(color(255,0,0));

//***** END SLIDER Y *****

//*****

//***** SLIDER EJE X *****

//*****

        cp5.addSlider("X")

        .setPosition(1180,300)

        .setSize(50,250)

        .setRange(-90,90)

        .setColorBackground(color(0,0,255))

        .setColorForeground(color(0,255,0))

        .setColorValue(color(255,255,255))

        .setColorActive(color(255,0,0));

//***** END SLIDER X *****

//*****

myPort.bufferUntil('\n');

background(0); //Color de la pantalla

for(int i = 0; i <= 18; i = i + 1){

```

```

stroke(255,0,0);

line(0,(a*i),800,(a*i));

text(90-10*i,2,(a*i)+10);

}

}

//***** END SETUP *****

//*****

//***** INICIO DRAW *****

//*****

void draw(){

    fill(0,0,255);

    textSize(25);

    text("Facultad de Ingenieria Electrica UMSNH",820,25);

    fill(255,30,20);

    textSize(15);

    text("Presione ( E - e ) para salir y guardar el documento de Angulo.txt",850,670);

    //*****CEJA DE LA GRAFICA*****

    fill(255,0,0);// para rellenar el cuadro de un color

    stroke(204, 102, 0);//Margen del cuadro de un color

    rect(800, 0, 20,410);// Genera el cuadro posicion , tamaño

    //*****

    //*****ZONA DE SLIDERS*****

    fill(100,100,250);// para rellenar el cuadro de un color

    stroke(204, 102, 0);//Margen del cuadro de un color

    rect(1180, 300,70,250);// Genera el cuadro posicion , tamaño

    fill(100,100,250);// para rellenar el cuadro de un color

    stroke(204, 102, 0);//Margen del cuadro de un color

    rect(1280, 300,70,250);// Genera el cuadro posicion , tamaño

    //*****

    //*****IMAGEN *****

    //imageMode(CENTER);

    PImage imagen=loadImage("FIE.png");

    image(imagen,820,50,270,200);

    PImage imagen2=loadImage("UMSNH.png");

    image(imagen2,1090,50,260,200);

```

```

value2 = (int)random (0,10);

//*****

String inString = myPort.readStringUntil('\n');

//*****LECTURA DEL PUERTO SERIAL*****

if (inString != null){

    float[] value = float(split(inString,','));

    if (contadortxt == 0){

        output.println( "Si se utiliza MATLAB se puede utilizar las columnas 1,2 y utilizar el comando (A = cell2mat(C) ) para poder graficar \n ");

        output.println( "si la columna 1 y 2, se puede hacer uso de Excel para graficar \n");

        contadortxt=1;

        }

    if(contadortxt >0){

        output.print((int)(value[0]) + "\t");

        output.print((int)(value[1]) + "\t");

        output.print((int)(value[0])+","+ "\t");

        output.print((int)(value[1])+","+ "\t");

        output.print((int)(value[2])+","+ "\t");

        output.print((int)(value[3])+","+ "\t");

        output.print((int)(value[4])+","+ "\t");

        output.print((int)(value[5])+","+ "\t");

        output.print((int)(value[6])+","+ "\t");

        output.print((int)(value[7])+","+ "\t");

        output.print(hour() +":");

        output.print(minute() +":");

        output.println(second());

        }

    m.updateMeter((int)(value[0]));

    println(value[0]);

    m2.updateMeter((int)(value[1]));

    inByte1 = map(value[0],-90,90,0,400);

    inByte2 = map(value[1],-90,90,0,400);

    R1= map((int)Push,-90,90,0,400);

    R2 = map(Push2,-90,90,0,400);

    Ref1 = (int)R1;

    Ref2 = (int)R2;

```



```

    if(graficay == 1){
        stroke(127,34,255);

        line(xPos, -1*(inByte1 - 400) , xPos, -1*(inByte1+4 - 400 ) );//grafica lineas de 2 pixeles

        line(xPos, -1*(Ref1 - 400 ) , xPos, -1*(Ref1 + 4 -400 ) );//grafica lineas de 2 pixeles

        }

    if (graficax == 1){
        stroke(100,255,200);

        line(xPos, -1*(inByte2 - 400) , xPos, -1*(inByte2+4 - 400) );//grafica lineas de 2 pixeles

        line(xPos, -1*(R2 -400) , xPos, -1*(R2+4 -400 ) );//grafica lineas de 2 pixeles

        }

    if(xPos >= 800){
        xPos = 0;

        background(0);//Color de la pantalla

        for (int i = 0; i <= 18;i = i+1){
            stroke(255,0,0);

            fill(255,255,255);

            textSize(11);

            line(0,(a*i),800,(a*i));

            text(90-(10*i),2,(a*i)+10);

            }

        }

    else {
        xPos++;

        }

    }//Cierra "if (inString != null)"

}

// Cierra "void Draw"

//***** END DRAW *****

//*****

//***** INICIO FUNCION PARA EL BOTON EN Y *****

//*****

public void EjeY(){
    if(contadory == 1){

        graficay=1;

        contadory = 2;
    }
}

```

```

return;

}

if (contadory == 2){

contadory = 1;

graficay=0;

return;

}

}

//***** END FUNCION BOTON Y*****

//*****

//***** INICIO FUNCION PARA EL BOTON EN X *****

//*****

public void Ejex(){

if (contadorx == 1){// INICIALMENTE AL PRESIONAR EL BOTON ENTRA DIRECTAMENTE A LA PRIMERA CONDICION POR
EL VALOR INICIAL DE contadorx=1;

// posteriormente a casusa de la funcion RETURN SALE DE LA FUNCION Ejex Y PARA LA SIGUIETE OCACION POR
ENTRA DIRECTAMENTE A LA SEGUNDA CONDICION POR QUE CAMBIA EL VALOR DEL CONTADOR

// Y ASI SICESIVAMENTE.

graficax=1;

contadorx = 2;

return;

}

if (contadorx == 2){

contadorx = 1;

graficax=0;

return;

}

}

//***** END FUNCION BOTON X*****

//*****

//***** INICIO FUNCION PARA EL SLIDER EN X *****

//*****

void X(int led){

Push = led;

float RLedx;

RLedx = map (Push,-90,90,3,255);

```

```

    int RLedxx=(int) (RLedx);

myPort.write(RLedxx) ;

myPort.write(1);

println(Push);

}

//***** END FUNCION  X*****

//*****

//** INICIO FUNCION PARA EL SLIDER EN Y **

//*****

    void Y(int led2){

    Push2 = led2;

    float RLedy;

    RLedy = map (Push2,-90,90,0,255);

    int RLedyy=(int) (RLedy);

myPort.write(RLedyy);

myPort.write(2);

    }

//***** END FUNCION  SLIDER Y*****

//*****

//***** INICIO FUNCION PARA TECLADO *****

//*****

    void keyPressed(){

    if(key == 'e' || key == 'E'){

    output.flush();

    output.close();

    exit();

    }

    }

//***** END TECLADO *****

//*****

```

Anexo C

```
#include <SoftwareSerial.h>

SoftwareSerial BT(10,11);// Pines RX-D10 y TX-D11

void setup()

{

  Serial.begin (9600);

  BT.begin(38400);

  Serial.println("Inicio de comandos AT: ");

}

void loop(){

  if (Serial.available()){

    BT.write(Serial.read());

  }

  if (BT.available()){

    Serial.write(BT.read());

  }

}
```