Diseño de un Algoritmo Auto-Estabilizador

Juan Pablo Coronel Aguilar

Facultad de Ingeniería Mecánica, Universidad Internacional SEK, Quito

Nota de Autor

Juan Pablo Coronel Aguilar, Facultad de Ingeniería Mecánica, Universidad Internacional SEK;

Director Ing. Gustavo Moreno M.S.C.

Cualquier correspondencia concerniente a este trabajo puede dirigirse a:

juan.coronel.aguilar@gmail.com.

Declaración Juramentada

Yo, JUAN PABLO CORONEL AGUILAR, con cédula de identidad 171883626-3, declaro bajo juramento que el trabajo aquí desarrollado es de mi autoría, que no ha sido previamente presentado para ningún grado a calificación profesional; y, que se ha consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración, cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la UNIVERSIDAD INTERNACIONAL SEK, según lo establecido por la Ley de Propiedad Intelectual, por su reglamento y por la normativa institucional vigente.

JUAN PABLO CORONEL AGUILAR

C.I.: 171883626-3

Dedicatoria y/o agradecimientos

Agradezco a Dios por haberme dado la vida y poder haber llegado hasta este punto de mi vida. Dedico mi proyecto de titulación principalmente a mis padres por haberme dado la educación y apoyarme en cada paso de mi vida. Y finalmente a todos los que me han apoyado en todo este tiempo.

Índice de Contenidos

Índice de Tablas y Figuras

Tablas

Figuras

Resumen

Existe gran variedad de avances tecnológicos en mecanismos y sistemas capaces de auto estabilizarse, como: robots capaces de mantenerse de pie, automóviles que logran conducirse automáticamente, drones, estabilizadores de imagen en las cámaras digitales, etc. En la Universidad Internacional SEK no se han realizado estudios sobre el tema hasta la fecha, por ello, se diseñó un algoritmo auto-estabilizador para un eje, que ayude a los estudiantes a visualizar y comprender los algoritmos de estabilización proporcional, integral y derivativo (PID).

Para la realización del eje auto estabilizador se realizó un diseño mecánico, electrónico y de software usando la plataforma Arduino para el algoritmo de control, y para la visualización la plataforma de Processing. La estructura es de acrílico y los ejes son de acero inoxidable 304. Consta Principalmente con un sensor MPU 6050 que combina un giroscopio y un acelerómetro para medir los grados de inclinación. Los actuadores son dos motores brushless controlados por ESC (Electronic Speed Control), una placa de Arduino UNO.

El algoritmo de control es un PID, que es el encargado de hacer las correcciones pertinentes para mantener el eje estable a cero grados de inclinación. El eje auto estabilizador es capaz de compensar un peso de hasta 170 gramos sin problemas. Es capaz de realizar aproximadamente de 43 correcciones por segundo.

Palabras clave: algoritmo, auto-estabilización, PID, control, Arduino, Processing.

Abstract

There is an extensive variety of technological advances in mechanisms and systems capable of self-steady, as: robots able to keep standing, cars that manage to drive automatically, drones, image stabilizers in digital cameras, etc. At Universidad Internacional SEK studies have not been conducted on the subject so far, therefore, a self-stabilizing algorithm for a shaft was designed to help students visualize and understand proportional, integral and derivative (PID) stabilization algorithms.

For the realization of the auto-stabilizer shaft; a mechanical, electronic and software design was performed using the Arduino platform for the control algorithm and for visualization the Processing platform. The structure is made of acrylic and the shafts are made of 304 stainless steel. It mainly consists with a MPU 6050 sensor that combines a gyroscope and an accelerometer to measure the degrees of inclination. The actuators are two brushless motors controlled by ESC (Electronic Speed Control), a plate of Arduino UNO.

The control algorithm is a PID, which is in charge of making the pertinent corrections to keep stable the axis at zero degrees of inclination. The auto-stabilizer shaft is able of compensating for a weight of 170 grams without problems. It is capable of performing approximately 43 corrections per second.

Keywords: algorithm, auto-stabilization, PID, control, Arduino, Processing.

Introducción

En la actualidad existe gran variedad de avances tecnológicos en mecanismos o sistemas que sean capaces de auto estabilizarse, esto se puede ver en diversas aplicaciones como: robots capaces de mantenerse de pie, automóviles que logran conducirse automáticamente, drones para poder mantenerse estables en el aire, estabilizadores de imagen en las cámaras digitales, etc. Todos estos sistemas requieren subsistemas mecánicos, electrónicos y algoritmos de control para mantenerse en equilibrio y/o realizar otras operaciones específicas.

En la Universidad Internacional SEK no se han realizado estudios sobre el tema hasta la fecha, por ello, se diseñará un algoritmo auto-estabilizador para un eje. La importancia de este trabajo radica en crear un prototipo auto-estabilizador que ayude a los estudiantes a visualizar y comprender los algoritmos de estabilización proporcional, integral y derivativo (PID), sensores y mecanismos utilizados en este este tipo de tecnologías.

Objetivo General

Construir un prototipo auto estabilizador de un eje incluyendo el diseño mecánico, electrónico y de software para facilitar la comprensión de los algoritmos PID de control usados industrialmente; mediante sistemas de control usando la plataforma Arduino UNO, un sensor MPU 6050 que combina un giroscopio y un acelerómetro y motores brushless.

Marco Teórico

A continuación se describen trabajos de mecanismos auto estabilizadores que existen actualmente, los robots auto equilibristas son un gran ejemplo de un mecanismo auto estabilizador, estos cuentan generalmente con dos ruedas, éstas pueden estar situadas en un mismo eje, o en dos ejes como una bicicleta, sin embargo existen también robots con únicamente una rueda. Una característica importante de estos robots de auto-balance es que requieren un giroscopio o un acelerómetro que permite medir las distancias de giro o velocidad de desplazamiento y con ello lograr mantener el equilibrio. Dentro de los robots de auto-balance de dos ruedas en un mismo eje más conocidos encontramos a: (Linke, Two-Wheel Self Balancing Robot, 2011)

Nbot: este robot es capaz de mantenerse en dos ruedas y no requiere ser manejado por

control remoto, por lo cual, este robot es autónomo y capaz de tomar decisiones al caminar, fue creado por David P. Anderson, para mantener el equilibrio usa un giroscopio y un acelerómetro. Este robot fue presentado por la NASA como "Cool Robot of the Week" (Anderson, 2013)

Joe le pendule: este robot utiliza un acelerómetro junto con un giroscopio para medir la inclinación del robot y que éste mantenga el equilibrio, este robot fue desarrollado por Felix Grasser en 2002. (Linke, Two-Wheel Self Balancing Robot, 2011)

LegWay: este robot está construido con LEGO para su funcionamiento usa sensores ópticos de proximidad, fue creado por Steve Hassenplug y su función principal es que esté balanceado en sus dos ruedas y siga una línea negra. (Hassenplug, 2002)

Equibot: a diferencia de muchos de los robots de auto-balanceo este utiliza un sensor infrarrojo para medir la distancia del suelo y mantener el equilibrio fue desarrollado por Dan Piponi en el año 2006. (Linke, Two-Wheel Self Balancing Robot, 2011)

El Segway Personal Transporter fue creado por Dean Kamen en el año 2002 (Kamen, 2011), el Segway consta de cinco giroscopios y dos sensores de inclinación para su óptimo funcionamiento son muy conocidos hoy en día y son usados comúnmente por guardias de seguridad. (Linke, Two-Wheel Self Balancing Robot, 2011)

Entre otros mecanismos que hoy en día se están desarrollando a gran escala son los drones o cuadricópteros, de los cuales se han desarrollado algunos métodos para su control; sin embargo, los algoritmos de control PID son uno de los métodos más utilizados actualmente gracias a su fácil implementación y desempeño. (Guerrero Tavares, 2014)

Los cuadricópteros son vehículos no tripulados, UAV por sus siglas en inglés (Unmanned Aerial Vehicles). Austria fue el primer país en desarrollar uno de estos vehículos no tripulados en el año 1849 con fines militares. (Quadcopter Arena, 2016) Uno de los primeros experimentos relevantes de un cuadricóptero se realizó alrededor de 1907 por los hermanos Jacques y Louis Breguet, éste fue muy inestable. Posteriormente, Étienne Oehmichen desarrolló un cuadricóptero el cual voló a trescientos sesenta metros del suelo y una distancia de un kilómetro. Conjuntamente a éste George de Bothezat creó su propio cuadricóptero para la armada de Estados Unidos. (Krossblade Aerospace Systems LLC, 2016)

En la actualidad los drones o cuadricópteros han creado interés por el estudio para mejorar sus algoritmos de control. Para ello el uso de drones como plataformas aéreas de estudio son indispensables; de los cuales los cuadricópteros más utilizados en los laboratorios son:

AscTec Hummingbird, Draganflyer X4, DIY Drones Arducopter. (Chamorro Hernández & Medina Mora, 2013)

Hoy en día existen trabajos que utilizan algoritmos para auto estabilizar aplicados en robots, por ejemplo, existe una tesis desarrollada por Jesús Norberto Guerrero Tavares de la Universidad Autónoma de Quetaro, la que consiste en emplear un sistema PD para un helicóptero de cuatro rotores (cuadricóptero) el cual cuenta con tres sensores infrarrojos los cuales permiten medir las posiciones del plano cartesiano x, y, z. Y un sensor IMU 3DM-GX1 para medir los ángulos de inclinación. Estos sensores van conectados a un procesador central DSC (Digital Signal Controller) el cual capta las señales de los sensores y reenvía la información de respuesta. (Guerrero Tavares, 2014)

Otro estudio realizado por Verónica Gabriela Ortiz Padilla y Pablo Ramiro Pulla Arévalo de la Escuela Politécnica del Ejército en la carrera de Mecatrónica, es la construcción de un prototipo de cuadricóptero el cual puede volar a una altura de cinco metros y a una distancia de cincuenta metros. Este cuadricóptero utiliza un sistema de control PID y un IMU (Inertial Measurement Unit) que cuenta con un giroscopio y un acelerómetro para poder medir la orientación de la inclinación del cuadricóptero. (Ortiz Padilla & Pulla Arévalo, 2012)

Algoritmos

La definición de algoritmo, según Mataix Aracil es: “Algoritmo es toda combinación de operaciones fundamentales en virtud de la cual de números cualesquiera a,b,c,.., se obtiene un número k.” (Mataix Aracil, 1965) En otras palabras, un algoritmo es un conjunto de establecido de reglas o instrucciones con el fin de resolver un problema. (López, et al., 1999)

Cada día se están creando más innovaciones dentro de la robótica intentando crear robots cada vez más parecidos a los seres humanos, robots capaces de recorrer superficies de otros planetas o robots capaces de transportar distintas cargas, por esto surge la necesidad de crear algoritmos de control que permitan a un robot o una máquina auto equilibrarse. Entre algunos algoritmos de control se tiene: On Off, PID, Redes Neuronales, Backstepping, entre otros. (Guerrero Tavares, 2014) Cabe recalcar que estos algoritmos no solo son utilizados para sistemas de control para auto-balanceo o auto-equilibrio, sino también para control de temperatura en el caso de los algoritmos On Off y PID, robots inteligentes o vehículos inteligentes paro los sistemas de control por redes neuronales.

Sistemas de Control

La definición de sistema de control es: "Sistemas de control es el proceso a seguir para determinar las modificaciones básicas que deben introducirse en el sistema para que se cumplan estas especificaciones." (Ñeco, Reinoso, García, & Aracil, 2003)

En otras palabras, sistemas de control es lograr que, un conjunto de elementos realicen acciones específicas.

El primer sistema de control automático se da con la creación del regulador de velocidad centrífugo por James Watt. (Rodríguez Celi, s.f.) Desde entonces se han realizado diversos cambios en el control automático, utilizando tanto la mecánica como la electrónica, y en tiempos modernos los algoritmos computacionales. (Gonzalez-Longat, s.f.)

La importancia de estos controladores hoy en día es muy extensa ya que son utilizados en diversas aplicaciones de automatización. En un principio fueron empleados para sistemas de navegación en barcos. Actualmente se los encuentra en una infinidad de lugares desde algoritmos simples hasta redes neuronales artificiales (RNA). El presente estudio se enfoca en los sistemas de control auto-estabilizadores, como es el caso de los robots de dos ruedas auto-equilibristas, que dieron paso a los transportes eléctricos de dos ruedas de auto-balanceo como los "Segway". (Linke, 2011)

Así, en la Tabla 1 se puede observar 4 sistemas de control que permiten mantener la estabilidad de un sistema, de ellos el PID fue seleccionado porque a diferencia de los sistemas de control por Redes Neuronales y Backstepping es relativamente sencillo de implementar, y brinda una estabilidad suficiente para el eje auto estabilizador que se va a construir. El controlador tipo On Off no se selecciona ya que es un sistema de control discreto básico y en el diseño se requiere aplicar un control continuo. Solo tiene la capacidad de cambiar entre dos estados; lo cual, no permite la estabilidad necesaria y el eje se mantendrá oscilando sin llegar a estabilizarse. El Backstepping no fue seleccionado ya que no se ha encontrado mucha información de su implementación, en adición, es un algoritmo de una complejidad alta y se requiere de un nivel computacional alto. Finalmente, un sistema de control por redes neuronales, brindaría a la planta una alta estabilidad ya que posee la capacidad de adaptarse; sin embargo, se requiere de un nivel computacional alto.

Tabla 1.

Comparación de sistemas de control.

Controlador		**Estructura**	**Estabilidad**	**Complejidad**
On off	Control lineal	Muy simple. Solo varía entre dos estados: "encendido" y "apagado"	Baja	Baja
PID	Control lineal	Simple, combina tres parámetros o acciones que son: Proporcional, Integral y Derivativo.	Media	Baja
Backstepping	Control no lineal	Procedimiento recursivo que combina la elección de una función de Lyapunov con el diseño de un control por realimentación. Este procedimiento descompone el problema original en una secuencia de problemas de diseño para sistemas de orden reducido.	Alta	Media
Redes Neuronales	Control no lineal	Imitan la estructura de las redes neuronales biológicas, procesa la información de forma paralela, distribuida y es adaptativo, es redundante y tolerante a fallas.	Alta	Alta

Control Proporcional Integral Derivativo (PID)

El controlador PID es uno de los controladores más utilizados actualmente en mecanismos de mesas auto estables o en cuadricópteros, dada a su sencillez comparada con otros. Este sistema de control comenzó a utilizarse en controladores en el siglo IXX. (Barro, 2009) El primer controlador fue implementado por Taylor Instrument Company en el año 1936, en este caso fue implementada la parte derivativa en un sistema de control PI que ya era conocido en 1930. (Minorsky, n.d.)

Este controlador es un sistema de control de lazo cerrado. Un sistema de lazo cerrado realiza una medición de la salida, la cual es comparada y realimentada al principio del proceso. Con este procedimiento se puede corregir errores ya que se está recolectando constantemente datos de la salida y corrigiendo en la entrada (Kuo, 1996). En la Figura 1, se puede observar su funcionamiento.

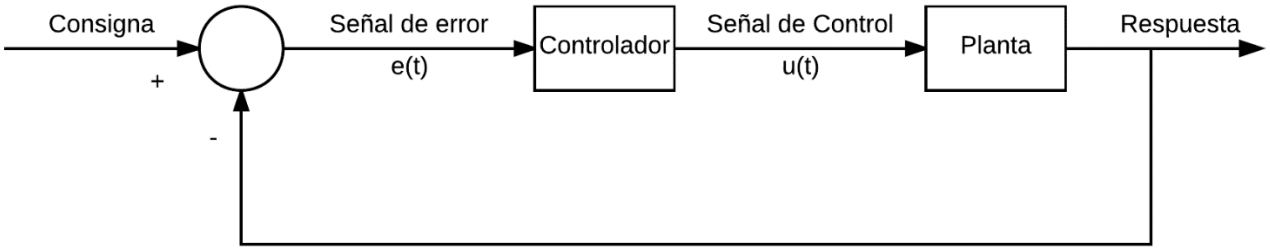


Figura 1. Figura de control de lazo cerrado. Tomado de: Angulo Bahón & Raya Giner, 2004

El PID será el encargado de realizar las correcciones necesarias para que el eje se mantenga en equilibrio. La ecuación que describe el funcionamiento del PID es la siguiente (Moyano):

$$u(t) = K_p e(t) + \frac{K_p}{T_i}\int_0^t e(t)dt + K_p T_d \frac{de(t)}{dt} \quad [1]$$

donde:

- u(t) es la salida del controlador y entrada de control al proceso,
- Kp es la ganancia proporcional,
- e(t) es el error de la señal,
- Ti constante de tiempo integral,
- Td constante de tiempo derivativa.

Esta fórmula general puede dividirse en tres:

La parte proporcional es la encargada de calcular el error con respecto al set point asignado. Su función es similar a un controlador tipo on off, con la ventaja que es multiplicado por una constante kp, la cual, ayuda a contrarrestar el error más rápido, sin embargo si la constante kp asignada es alta, la planta empezaría a oscilar. Un controlador proporcional por sí solo no logra estabilizar completamente la planta, ya que su funcionamiento es como la de un controlador on off. (Visioli, 2006)

$$up(t) = K_p e(t) \quad [2]$$

La parte integral es encargada de llevar el registro de los errores pasados, es decir, que calcula el error acumulado. Para determinar el valor de este error acumulado es necesario determinar el área bajo la curva del error pasado, mientras más grande sea el área bajo la curva mayor será la señal entregada a los actuadores. Un controlador proporcional e integral es capaz de estabilizarse, a diferencia de un controlador proporcional. (Visioli, 2006)

$$ui(t) = K_i \int_0^t e(t)dt \qquad [3]$$

La parte derivativa es la encargada de tomar decisiones para evitar errores futuros, para poder determinar estos errores, calcula la pendiente que tiene la curva de error y la multiplica por la constante kd. Esta parte derivativa ayuda a sumar más energía a la señal entregada a los actuadores, dependiendo de la inclinación de la pendiente; por ello acelera el proceso de estabilización. Un controlador derivativo le da velocidad para estabilizarse a un sistema proporcional e integral. (Visioli, 2006)

$$ud(t) = K_d \frac{de(t)}{dt} \qquad [4]$$

Sintonización de un sistema PID.

Para un buen funcionamiento de la planta es necesario asignar valores a las constantes kp, ki y kd. Para encontrar los valores existen algunos métodos como: el método de oscilación de Ziegler Nichols; realizar un modelo matemático de la planta; o la posibilidad de encontrar los valores kp, ki, y kd experimentalmente. Se seleccionó el método experimental para este trabajo debido a su facilidad de implementación.

Componentes Electrónicos Usados en Ejes Auto Estabilizadores

Sensores giroscopio y acelerómetro.

Un giroscopio es un dispositivo que está formado por tres ejes perpendiculares como se observa en la Figura 2, este debe ser capaz de girar con libertad en sus tres ejes (Weber, White, Mannig, & Febrer, 1954), según (Sears & Zemansky, 1970), un giroscopio es un trompo; el cual tiene simetría de revolución, y gira alrededor de un eje, en el cual uno de sus puntos está fijo y es

el centro de gravedad.

Figura 2. Giroscopio. Tomado de: (Kane & Sternheim, 2007)

Los giroscopios son usados con gran frecuencia en los dispositivos de navegación, principalmente en aviones, barcos y misiles; sin embargo en la actualidad estos no son únicamente mecánicos, sino, que son combinados mecánicos y electrónicos o solamente electrónicos. Los giroscopios electrónicos vienen en un circuito integrado y muchas veces contienen un acelerómetro. (Burnie, et al.)

Al igual que muchos de los sensores electrónicos los giroscopios electrónicos están formados internamente por MEMS (Microelectromechanical Systems) mecanismos extremadamente pequeños aproximadamente de 1 a 100 micrómetros. (Rebeiz, 2003)

En el momento que el giroscopio empieza a moverse, una pequeña masa en el interior se desplaza con los cambios angulares como se muestra en la Figura 3 y emite señales eléctricas pequeñas las cuales son amplificadas. (5hertz, 2014)

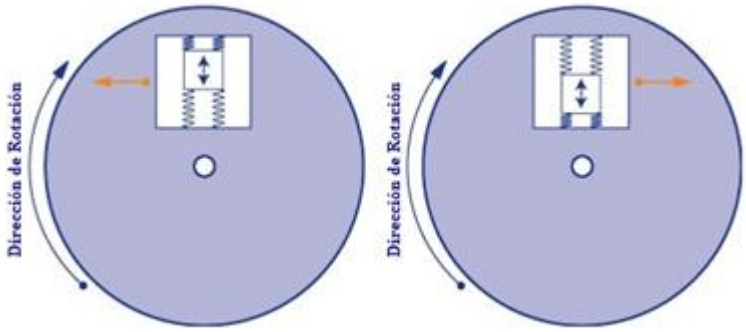


Figura 3. Representación gráfica del funcionamiento de los MEMS internos de un giroscopio eléctrico. Tomado de: (5hertz, 2014)

En el mercado existe gran variedad de giroscopios, en la Tabla 2 se muestra la comparación de 4 giroscopios.

Tabla 2.

Comparación de sensores giroscopios.

Sensor	**Ejes**	**Interfaz**	**Costo**
MPU 6050	3	I2C	Bajo
L3G4200D	3	I2C	Alto
ITG3200	3	I2C	Alto
IDG1215	2	Analógica	Medio

De estos Sensores se seleccionó al sensor MPU 6050 principalmente por su bajo costo, compatibilidad con el sistema de control y la existencia de bibliotecas de Arduino creadas específicamente para este sensor. Los sensores restantes también son compatibles; sin embargo, no fueron seleccionados ya que al no tener bibliotecas específicas dedicadas para cada uno de estos sensores, la programación sería más laboriosa. Adicionalmente, el costo de los sensores restantes es mucho más alto. El MPU 6050 es común y fácil de encontrar en el mercado local, siendo esto un factor más que influye en la selección de este sensor. (Arduino, 2016)

El MPU 6050 es un IMU (Inertial Measurement Unit) que traducido al español es un sistema de medida inercial, los cuales son chips integrados que contienen como mínimo un giroscopio y un acelerómetro. (Prometec.net, n.d.) Específicamente, el MPU 6050 lleva un giroscopio y un acelerómetro.

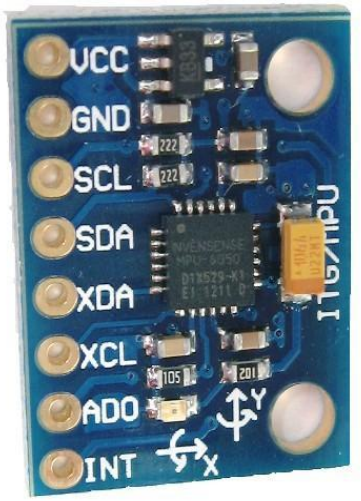


Figura 4. MPU-6050. Tomado de (Arduino, 2016)

Los acelerómetros son dispositivos capaces de medir la rapidez de cambio de la velocidad en función del tiempo, es decir, la aceleración. Son utilizados para saber la posición y orientación de algún elemento. (García, Hidalgo, Loza, & Muñoz, 2013)

Existen varios tipos de acelerómetros de los cuales se destacan principalmente los siguientes: acelerómetros capacitivos y piezo-resistivos. Todos ellos cumplen la misma función de medir la aceleración. (Acelerómetros y sensores de vibración, n.d.)

Los acelerómetros capacitivos están formados por varias prolongaciones en una placa móvil; de un lado de la placa se encuentra conectado un grupo de capacitores, mientras que en el otro lado están conectados el resto, como se muestra en la Figura 5 (a) La fuerza de inercia de la placa móvil genera movimiento de la placa, este movimiento genera que las capacitancias cambien; es decir, uno de ellos incremente, mientras que el otro se reduce. La aceleración es medida a partir de la diferencia de las capacitancias. (Bao, 2000)

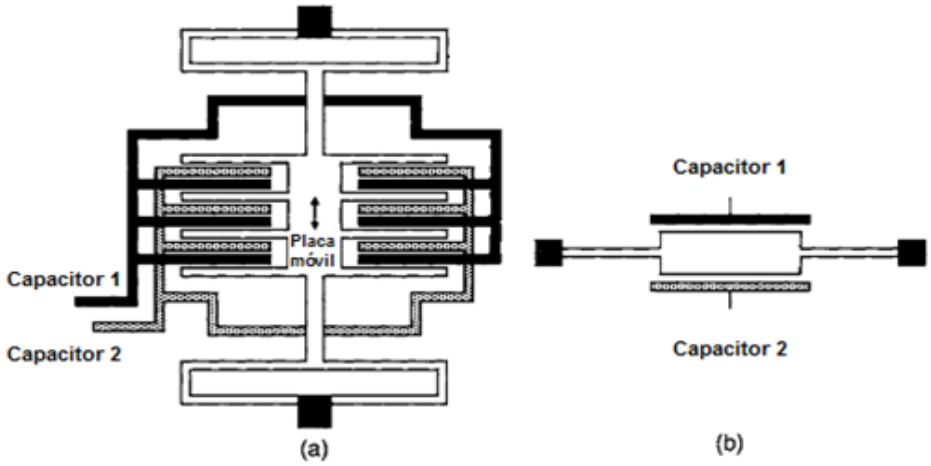


Figura 5. Representación gráfica de la estructura de un acelerómetro capacitivo: (a) vista superior, (b) modelo simplificado. Tomado de: (Bao, 2000)

En los acelerómetros piezo-resistivos se mide la variación de la resistencia causada por la elongación del material que es piezo-resistivo. Su estructura se consta de un marco o carcasa el cual tiene un puente que está conectado a una masa la cual se mueve por acción de la aceleración, este movimiento causa que el puente se doble. En la Figura 6 se puede observar la estructura del acelerómetro. (Bao, 2000)

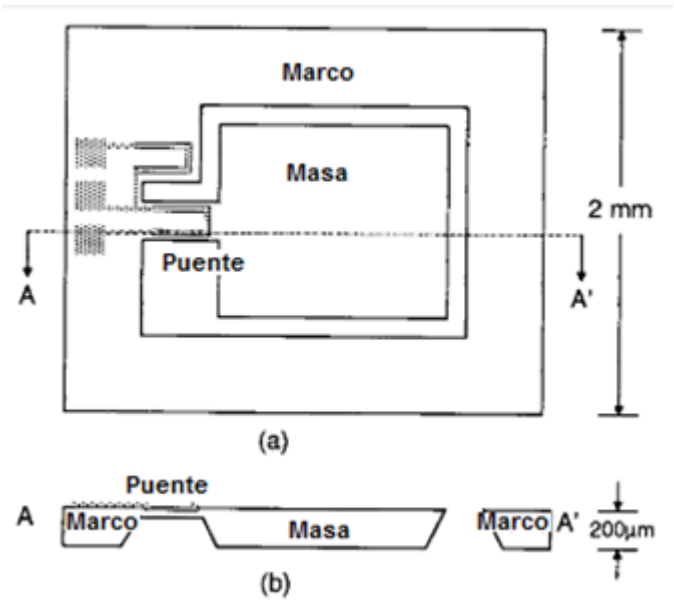


Figura 6. Representación gráfica de la estructura de un acelerómetro: (a) vista superior, (b) vista lateral. Tomado de: (Bao, 2000)

Actuadores.

Motores.

Existe gran variedad de motores eléctricos en el mercado; sin embargo, en este trabajo se toma en cuenta solo los motores con escobillas o motores brushed y los motores de corriente continua sin escobillas o motores brushless (BLDC) por su aplicación tanto en aeromodelismo como en cuadricópteros. En la Tabla 3 se muestra una comparación entre estos motores.

Tabla 3.

Comparación entre motores brushed y motores brushless. Obtenido de: (Yedamale, 2003)

	Motores Brushless	**Motores Brushed**
Mantenimiento	No indispensable	Periódico
Vida útil	Larga	Corta
Eficiencia	Alta	Moderada
Velocidad	Alta	Baja
Control	Complejo	Simple

Los motores que se utilizarán son sin escobillas. Los motores brushless comúnmente son trifásicos, el estator está dividido en la mayoría de los casos en tres devanados conectados en forma de estrella y cada uno de los devanados está formado de varias bobinas. El rotor está formado por imanes permanentes que están ubicados alternando los polos norte y sur. Estos motores logran alcanzar mayores velocidades que los motores comunes con escobillas. (Titus, 2012) no requieren de mayor mantenimiento como los motores con escobillas y tienen una vida larga; sin embargo, estos motores son más costosos, producen menos torque y requieren de un ESC (Electronic Speed Control) para su control. (Cook, 2015)

Controladores de Motores.

Los ESC varían la velocidad del motor dependiendo de la señal que es entregada a este. El ESC consta de tres cables en un extremo que son la alimentación trifásica hacia los motores brushless y en el otro extremo con dos cables de alimentación positivo y negativo que van desde la batería o fuente de voltaje, y finalmente, 3 cables para el sistema de control PWM (Pulse Width Modulation). La velocidad del motor depende de la señal que es entregada al ESC, dependiendo de la longitud de ancho de pulso. Comúnmente existen en el mercado ESC que funcionan con señales de ancho de pulso entre 1 ms a 1.5 ms, 1 ms a 2 ms y de 1 ms a 2.5 ms. Los ESC vienen también de distintos amperajes como: 15 A, 25 A, 30 A, etc. Este amperaje debe ser mayor al amperaje que requiere el motor brushless que se va a usar caso contrario se corre el riesgo de quemar el ESC. (Baraza, 2010)

Tanto el motor brushless como el ESC y las hélices fueron adquiridas en un paquete que incluye los tres artefactos mencionados. Las Figuras 7 y 8 muestran los motores y los ESCs respectivamente, seleccionados para el proyecto.

Figura 7. A2212 1000KV Brushless. Tomado de (HeTPro, n.d.)

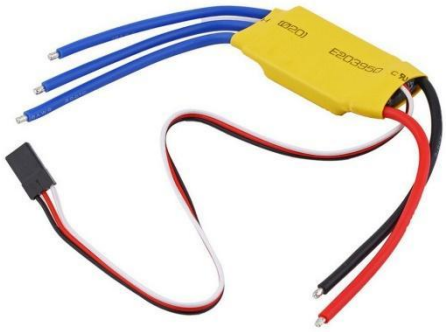


Figura 8. 30A brushless ESC. Tomado de (Amazon, 2016)

Sistema de Control

El sistema de control está basado en la plataforma de prototipos electrónicos que consta con un código y hardware abierto, Arduino, tiene un microcontrolador ATmega328P programable de cuenta con una SRAM (RAM estática) de 2KB, EEPROM (memoria reprogramable) de 1KB y Memoria Flash de 32KB. En la placa de Arduino UNO viene preprogramado con un "bootloader"; el cual permite cargar los programas al microcontrolador más fácilmente y sin necesidad de un programador externo.

Se seleccionó la plataforma Arduino ya que tiene software y hardware libre, cuenta con una gran variedad de librerías para implementar en el software, el costo es bajo, y el entorno de programación es sencillo. Es muy utilizado actualmente en distintos proyectos. (ARDUINO, 2016)

Método

Este trabajo se basa en la construcción de un eje auto estabilizador con dos motores, al cual se le implementó un algoritmo de control para su auto estabilización. En la primera parte, se analizará la estructura mecánica, que incluye los cálculos estructurales y materiales utilizados en el mecanismo; posteriormente se revisará el diseño eléctrico y electrónico en donde se encuentra las conexiones y cálculos eléctricos realizados; luego se analizará diseño de control; y finalmente se revisará el diseño del algoritmo de control separado en el diseño de software en Arduino para el control del sistema y el diseño de software en Processing para la visualización del funcionamiento.

Estructura Mecánica

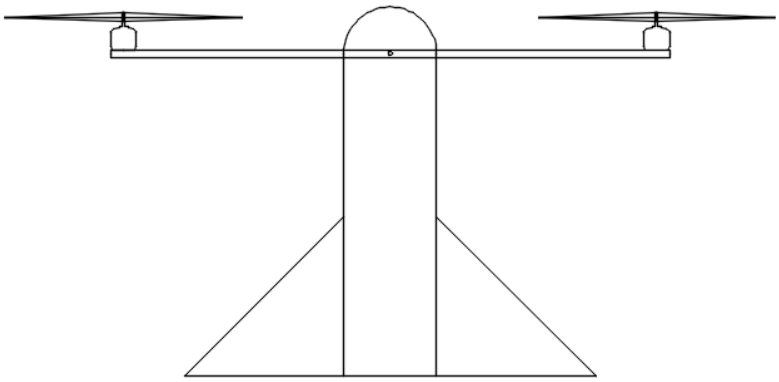


Figura 9. Bosquejo de eje auto-estabilizador con dos motores.

Calculo de reacciones y momento máximo del mecanismo:

Para hacer el cálculo del eje auto-estabilizador se debe tomar en cuenta el diagrama de cuerpo libre, en donde se observan las fuerzas a las que se encuentra sometido el mecanismo. Para la primera parte de los cálculos se tomó en cuenta dos casos. El primer caso es cuando los dos motores del eje auto estabilizador se encuentran funcionando a su máxima potencia y el eje se mantiene de forma horizontal. El segundo caso es cuando uno de los dos motores deja de funcionar mientras que el otro sigue funcionando a su máxima potencia; por lo cual el eje se

encuentra inclinado. En la Figura 10 y 11 se observa los diagramas de cuerpo libre del caso uno y dos respectivamente.

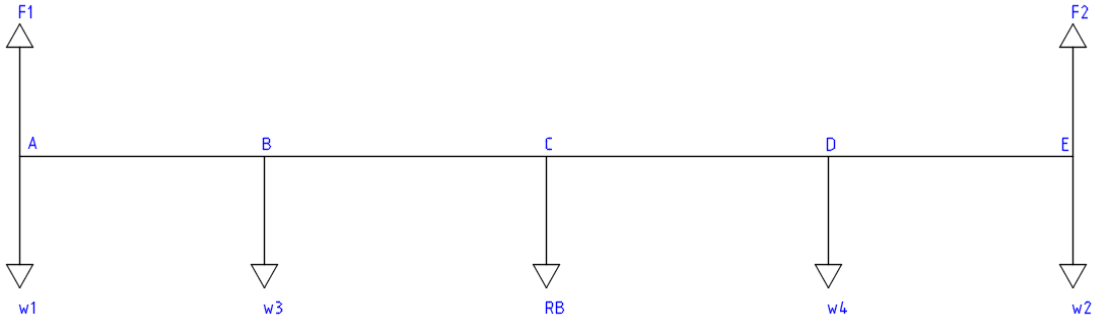


Figura 10. Diagrama de cuerpo libre. Caso 1.

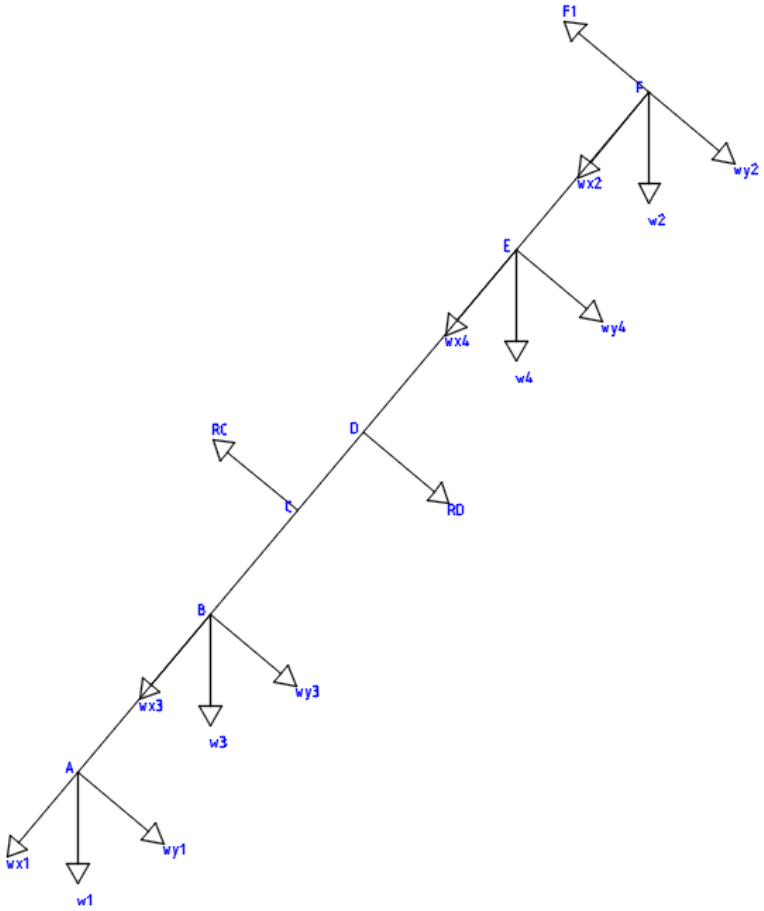


Figura 11. Diagrama de cuerpo libre. Caso 2.

Para el caso número uno se tienen los siguientes datos:

- Masa que levanta el motor uno (m1) y el motor dos (m2) es de 0.733 kg.
- Masa del motor 1 (p1) y masa del motor dos (p2) es de 0.073 kg.
- Masa del ESC (Electronic Speed Control) uno y dos es de 0.027 kg.
- Distancia AB 0.13 m
- Distancia BC 0.15 m
- Distancia CD 0.15 m
- Distancia DE 0.13 m

La masa que levantan lo motores se multiplica por la gravedad para obtener la fuerza 1 (F1) y fuerza 2 (F2). Así mismo la masa de los motores y masa de los ESC se multiplican por la gravedad para obtener el peso. (Peso de los motores w1 y w2, peso de los ESC w3 y w4)

F1 = m1 * 9.8 m/s^2	F2 = m1 * 9.8 m/s^2
F1 = 0.733 kg * 9.8 m/s^2	F2 = 0.733 kg * 9.8 m/s^2
F1 = 7.183 N	F2 = 7.183 N

w1 = m1 * 9.8 m/s^2	w2 = m1 * 9.8 m/s^2
w1 = 0.027 kg * 9.8 m/s^2	w2 = 0.027 kg * 9.8 m/s^2
w1 = 0.715 N	w2 = 0.715 N

w3 = m1 * 9.8 m/s^2	w4 = m1 * 9.8 m/s^2
w3 = 0.733 kg * 9.8 m/s^2	w4 = 0.733 kg * 9.8 m/s^2
w3 = 0.265 N	w4 = 0.265 N

Ya que F1 y w1 al igual que F2 y w2 se encuentran en el mismo punto se saca una resultante R1 y R2.

R1 = F1 –w1	R2 = F2 –w2
R1 = 7.183 N – 0.715N	R2 = 7.183 N – 0.715N
R1 = 6.468 N	R2 = 6.468 N

En este punto la única incógnita que se tiene es la reacción en “C” (RC). Para encontrar la RC se realiza una sumatoria de fuerzas en “y”

$\sum Fy = 0$

$RC = R1 + R2 - w3 - w4$

$RC = 6.468\ N + 6.468\ N - 0.265\ N - 0.265\ N$

$RC = 12.407\ N$

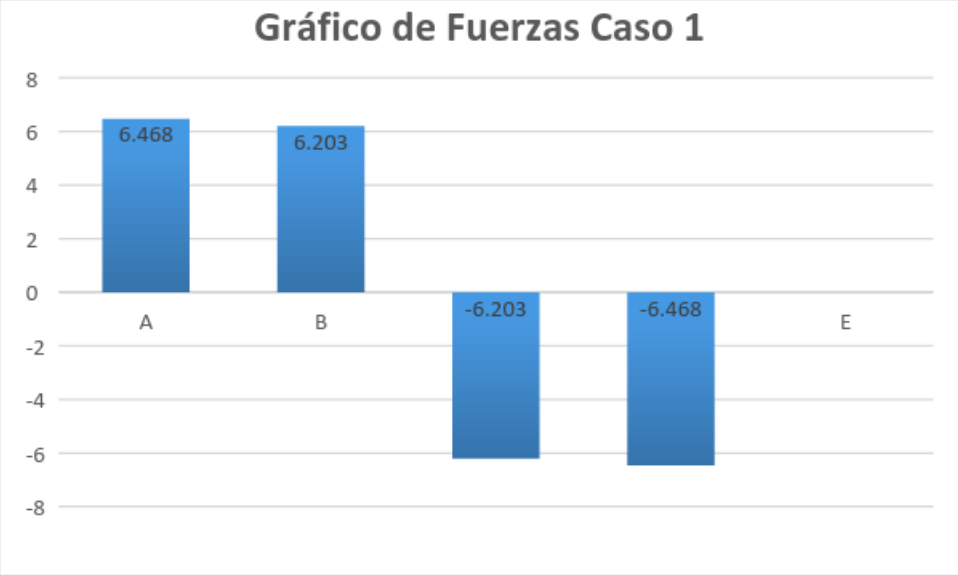


Figura 12. Gráfico de fuerzas caso 1.

Se realiza una sumatoria de momentos para identificar el momento máximo

$\sum Ma = 0$

$\sum Mb = Ma + (6.468\ N * 0.13\ m) = 0.841\ N*m$

$\sum Mc = Ma + (6.203\ N * 0.15\ m) = 1.771\ N*m$

$\sum Md = Ma + (-6.203\ N * 0.15\ m) = 0.841\ N*m$

$\sum Me = Ma + (-6.468\ N * 0.13\ m) = 0\ N*m$

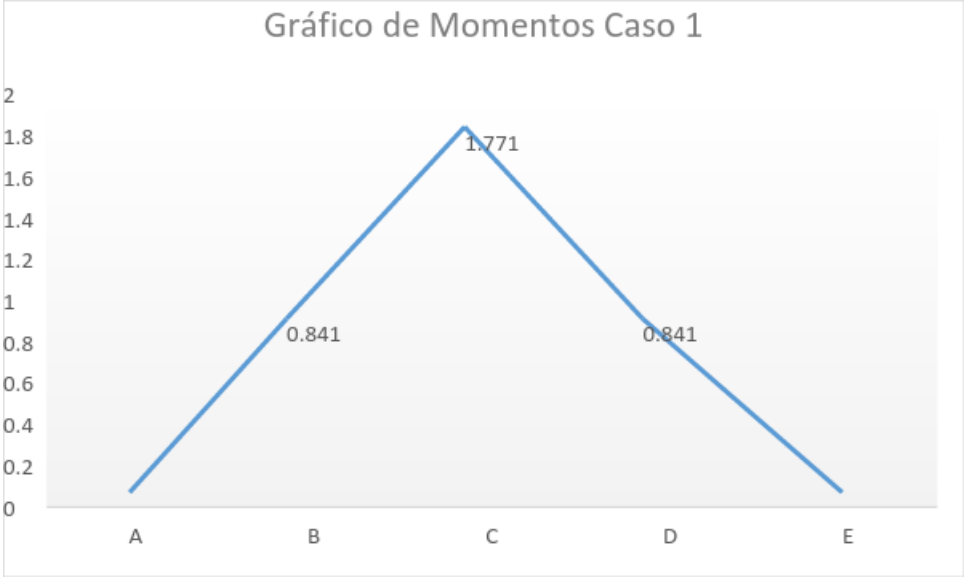


Figura 13. Gráfico de momentos caso1.

Para el caso número dos se tienen los siguientes datos:

- Masa que levanta el motor uno (m1) 0.733 kg.
- Masa del motor 1 (p1) y masa del motor dos (p2) 0.073 kg.
- Masa del ESC (Electronic Speed Control) uno y dos 0.027 kg.
- Distancia AB 0.13 m
- Distancia BC 0.085 m
- Distancia CD 0.065 m
- Distancia DE 0.15 m
- Distancia EF 0.13 m
- Distancia AC 0.215 m
- Distancia CE 0.215 m
- Distancia CF 0.345 m

La masa que levanta el motor se multiplica por la gravedad para obtener la fuerza 1 (F1). De la misma manera la masa de los motores y masa de los ESC se multiplican por la gravedad para obtener el peso. (Peso de los motores w1 y w2, peso de los ESC w3 y w4)

F1 = m1 * 9.8 m/s^2
F1 = 0.733 kg * 9.8 m/s^2
F1 = 7.183 N

w1 = m1 * 9.8 m/s^2
w1 = 0.027 kg * 9.8 m/s^2
w1 = 0.715 N

w2 = m1 * 9.8 m/s^2
w2 = 0.027 kg * 9.8 m/s^2
w2 = 0.715 N

w3 = m1 * 9.8 m/s^2
w3 = 0.733 kg * 9.8 m/s^2
w3 = 0.265 N

w4 = m1 * 9.8 m/s^2
w4 = 0.733 kg * 9.8 m/s^2
w4 = 0.265 N

La inclinación del eje es de 50 grados y para facilitar los cálculos se coloca como si el eje estuviera de forma horizontal; por ello, es necesario descomponer los distintos pesos en sus componentes “x”, “y”.

wx1 = (sin (50)) * 0.715 N = 0.548 N
wy1 = (cos (50)) * 0.715 N = 0.46 N

wx2 = (sin (50)) * 0.715 N = 0.548 N
wy2 = (cos (50)) * 0.715 N = 0.46 N

wx3 = (sin (50)) * 0.265 N = 0.203 N
wy3 = (cos (50)) * 0.265 N = 0.17 N

wx4 = (sin (50)) * 0.265 N = 0.203 N
wy4 = (cos (50)) * 0.265 N = 0.17 N

Ya que F1 y wy2 se encuentran en el mismo punto se saca una resultante R1.

R1 = F1 –wy2
R1 = 7.183 N – 0.46 N
R1 = 6.724 N

Como se tienen dos incógnitas que son la reacción en “c” (RC) y la reacción en “d” (RD) se realiza una sumatoria de momentos en el punto “c”.

∑Mc =0

RD= ((wy1*AC)+(Wy3*BC)-(wy4*CE)+(R1*CF))/CD

RD= ((0.46N*0.215m)+(0.17N*0.085m)-(0.17N*0215m)+(6.724 N*0.345m))/0.065m

RD= 37.04 N

En este punto la única incógnita que existe es la reacción en "c" (RC). Para encontrar la RC se realiza una sumatoria de fuerzas en "y".

∑Fy =0

RC = wy1+wy3+RD+wy4-R1

RC = 0.46N + 0.17N + 6.724 N+ 0.17 N-6.724

RC = 31.116 N

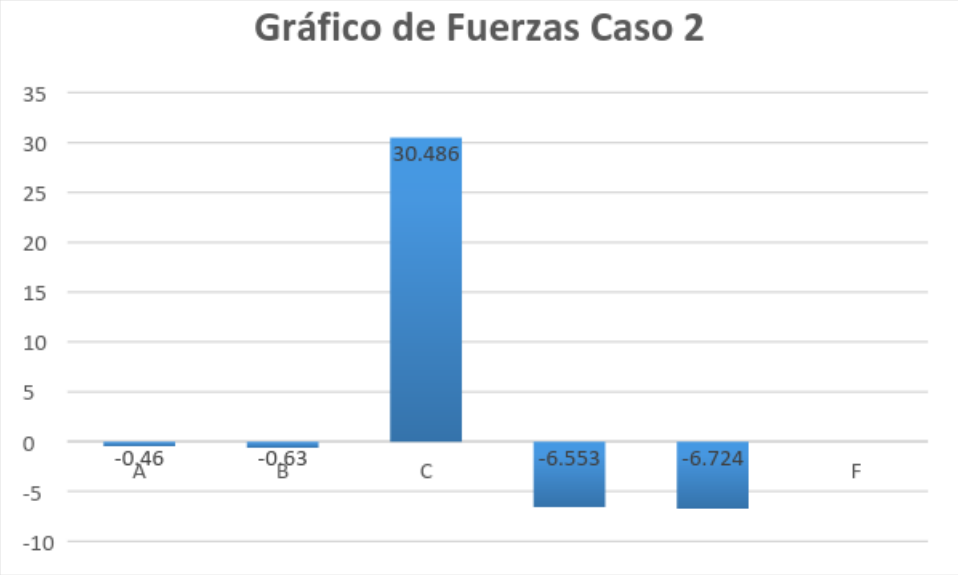


Figura 14. Gráfico de fuerzas caso 2.

Se realiza una sumatoria de momentos para identificar el momento máximo

∑Ma = 0

∑Mb = Ma + (-0.46 N * 0.13 m) = -0.06 N*m

∑Mc = Mb + (-0.63 N * 0.085 m) =-0.114 N*m

∑Md = Mc + (30.486 N * 0.065 m) = 1.857 N*m

∑Me = Md + (-6.553 N * 0.15 m) = 0.874 N*m

∑Mf = Me + (-6.724 N * 0.13 m) = 0 N*m

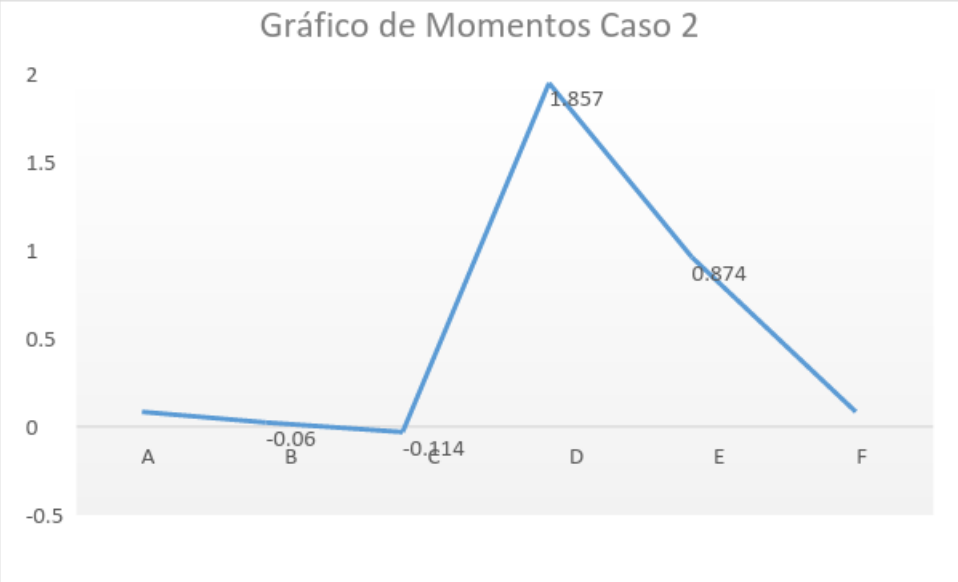


Figura 15. Gráfico de momentos caso 2.

Al hacer el análisis de momentos máximos tanto en el caso uno como en el caso dos; se observa que en el caso dos, se tiene un momento máximo de 1.857 N*m mientras que en el caso uno, el momento máximo es de 1.771 N*m, por lo cual se trabaja con el cálculo del caso número dos.

Con el resultado del momento máximo, se aplica la fórmula del esfuerzo máximo para calcular posteriormente el factor de seguridad del balancín de acrílico donde van a ir sujetos los Motores.

$$\sigma x = \frac{Mmax \times C}{I} \quad [5]$$

$$C = \frac{h}{2} \quad [6]$$

$$I = \frac{b \times h^3}{12} \quad [7]$$

donde:

Mmax: Es el momento máximo,

C: El centro instantáneo,

I: El momento de Inercia,

h: Es la altura del rectángulo de la sección transversal,

b: La base del rectángulo de la sección transversal.

Como dato se tiene, que la sección transversal del acrílico, tiene de base 0.04m y 0.004m de altura.

$$\sigma x = \frac{1.857N * m * 0.002\ m}{2.133 * 10^{-10}}$$

$$\sigma x = \ 1.741 * 10^{7} Pa$$

Para determinar los esfuerzos que actúan sobre el balancín se tiene:

$$\sigma 1,3 = \frac{\sigma x+\sigma y}{2} \pm \sqrt{(\frac{\sigma x+\sigma y}{2})^2 + \tau xy^2} \qquad [8]$$

En este caso se tiene que: tanto el esfuerzo en "y" como la cortante "xy" son cero, por lo cual el esfuerzo uno es igual al esfuerzo máximo, y el esfuerzo tres es igual a cero.

$$\sigma 1 = \sigma x \qquad [9]$$

Aplicando la teoría de Mohr Coulomb se tiene la siguiente fórmula:

$$\frac{\sigma 1}{St} - \frac{\sigma 3}{Sc} = \frac{1}{F.S} \qquad [10]$$

donde:

St es la resistencia a la tracción,

Sc es la resistencia a la compresión,

F.S es el factor de seguridad.

Ya que el esfuerzo tres es igual a cero la ecuación queda:

$$\frac{\sigma 1}{St} = \frac{1}{F.S} \quad [11]$$

En este punto se despeja el factor de seguridad

$$F.S = \frac{St}{\sigma 1} \quad [12]$$

La resistencia a la tracción es de 7.2*10^7Pa (Egox, n.d.)

$$F.S = \frac{7.2 * 10^7 Pa}{1.741 * 10^7 Pa}$$

$$F.S = 4.136$$

Como el valor del factor de seguridad es alto, se deduce que el material que se escogió resiste los esfuerzos a los que está sometido.

El eje de giro del balancín y el eje que detendrá el giro del balancín también requieren de un análisis de fuerzas y momento máximo para posteriormente determinar el diámetro que se deberá utilizar. Para el análisis nuevamente se tiene dos casos. El primero es el eje donde va a girar el balancín (caso 3) y el segundo el eje que va a detener el balancín (caso 4). En ambos casos el estudio se realiza en la vista lateral del mecanismo. A continuación en la Figura 16 y 17 se observa los diagramas de cuerpo libre del primer caso y el segundo caso respectivamente.

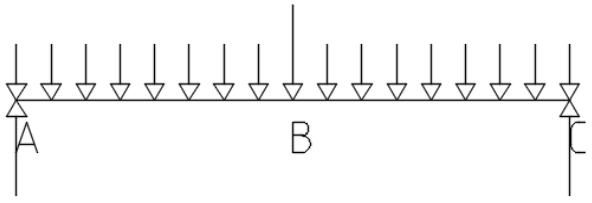


Figura 16. Diagrama de cuerpo libre. Caso 3.

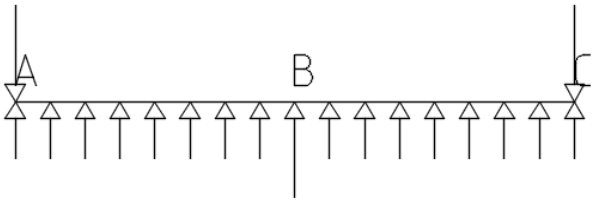


Figura 17. Diagrama de cuerpo libre. Caso 4.

Como se observa en este caso se tiene una carga distribuida en los dos casos. Tanto en el primer caso como en el segundo caso se utiliza las fuerzas ya calculadas previamente en el análisis de fuerzas en el caso dos. Por lo cual se tiene que la fuerza en el punto "b" en el caso 3 es de 37.04 N mientras que en el caso 4 es de 31.116 N.

Para el caso 3 se tiene los siguientes datos:

- B= 37.04 N
- AB= 0.02m
- BC= 0.02m
- AC= 0.04m

Como se tiene que la reacción en "A" y la reacción en "C" son desconocidas, se realiza una sumatoria de momentos en el punto "A".

∑Ma=0

$$C = \frac{B * AB}{AC}$$

$$C = \frac{37.04N * 0.02m}{0.04m}$$

$$C = 18.52\ N$$

Como en este punto la única incógnita que queda es la reacción en el punto “A”, se realiza una sumatoria de fuerzas en “y”.

∑Fy=0

$A = B - C$

$C = 37.04N - 18.52N$

$A = 18.52\ N$

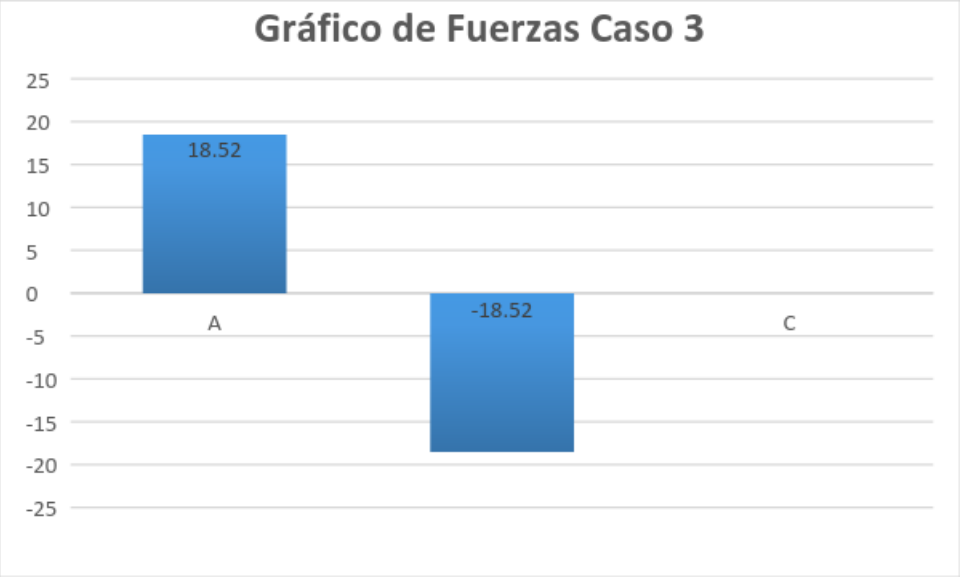


Figura 18. Gráfico de fuerzas caso 3.

Se realiza una sumatoria de momentos para identificar el momento máximo

∑Ma = 0

∑Mb = Ma + (18.52 N * 0.002 m) /2= 0.185 N*m

∑Mc = Mb + (-18.52 N * 0.002 m)/2 = 0 N*m

Figura 19. Gráfico de momentos caso 3.

Para el caso 4 se tiene los siguientes datos:

- B= 31.116 N
- AB= 0.02m
- BC= 0.02m
- AC= 0.04m

Como se tiene que la reacción en "A" y la reacción en "C" son desconocidas, se realiza una sumatoria de momentos en el punto "A".

∑Ma=0

$$C = \frac{B * AB}{AC}$$

$$C = \frac{31.116N * 0.02m}{0.04m}$$

$$C = 15.558\ N$$

Como en este punto la única incógnita que queda es la reacción en el punto "A", se realiza una sumatoria de fuerzas en "y".

∑Fy=0

$A = B - C$

$C = 31.116N - 15.558N$

$A = 15.558\ N$

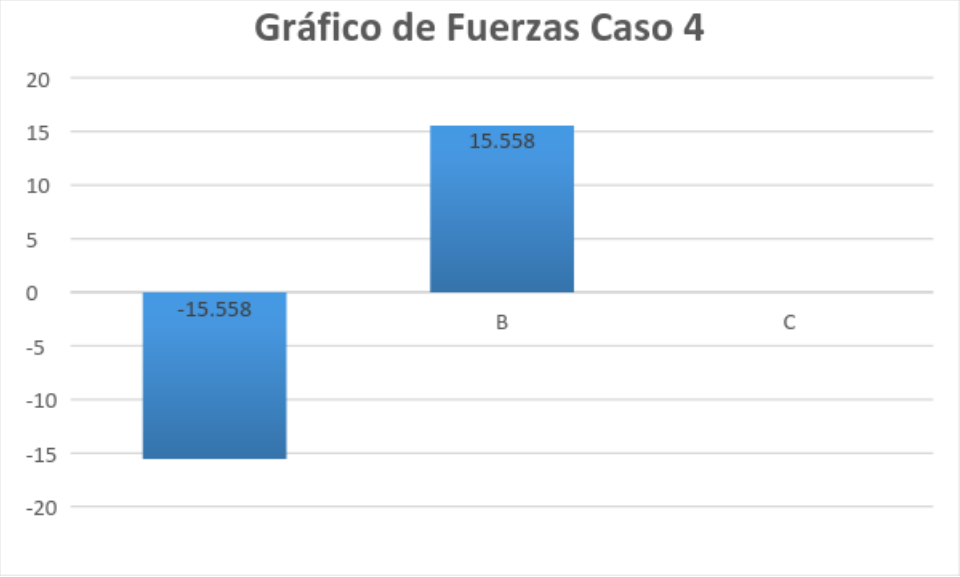


Figura 20. Gráfico de fuerzas caso 4.

Se realiza una sumatoria de momentos para identificar el momento máximo

∑Ma = 0

∑Mb = Ma + (-15.558 * 0.002 m) /2= -0.156 N*m

∑Mc = Mb + (15.558* 0.002 m)/s = 0 N*m

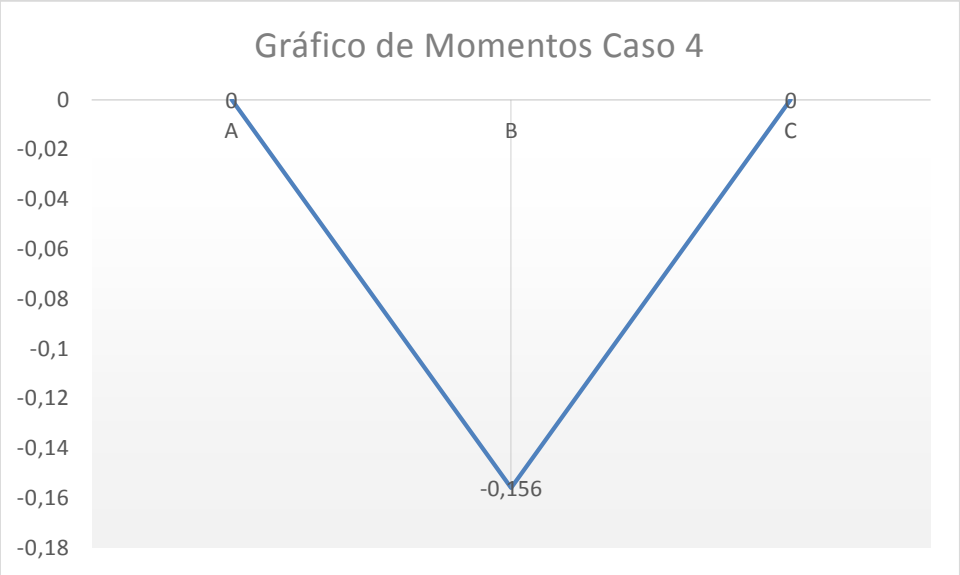


Figura 21. Gráfico de momentos caso 4.

Partiendo del análisis de momento máximo se realiza el cálculo del diámetro que debe utilizarse en estos ejes. Ya que el momento máximo en el caso 3 resulta ser mayor que en el caso 4, los cálculos son en base al caso 3.

Con el resultado del momento máximo, se aplica la fórmula del esfuerzo máximo para calcular posteriormente el factor de seguridad del balancín de acrílico donde van a ir sujetos los Motores.

$$\sigma x = \frac{Mmax \times C}{I} \qquad [5]$$

$$C = \frac{d}{2} \qquad [13]$$

$$I = \pi \frac{d^4}{64} \qquad [14]$$

donde:

Mmax: Es el momento máximo,

C: El centro instantáneo,

I: El momento de Inercia,

d: Es el diámetro del eje.

Como se requiere ver el diámetro que se desea utilizar se deja el diámetro (d) como incógnita.

$$\sigma x = \frac{0.185N * m * \frac{d}{2}\, m}{\pi \frac{d^4}{64}}$$

$$\sigma x = \frac{1.886 * 10^{-6} * MN * m}{d^3}$$

Se determina los esfuerzos que actúan sobre el eje:

$$\sigma 1,3 = \frac{\sigma x + \sigma y}{2} \pm \sqrt{(\frac{\sigma x + \sigma y}{2})^2 + \tau xy^2} \qquad [8]$$

En este caso se tiene que tanto el esfuerzo en "y" como la cortante "xy" son cero, por lo cual, el esfuerzo uno es igual al esfuerzo máximo, y el esfuerzo tres es igual a cero.

$$\sigma 1 = \sigma x$$

Aplicando la teoría de Mohr Coulomb para carga estática tenemos la siguiente fórmula:

$$\frac{\sigma 1}{St} - \frac{\sigma 3}{Sc} = \frac{1}{F.S} \qquad [10]$$

donde:

St es la resistencia a la tracción,

Sc es la resistencia a la compresión,

F.S es el factor de seguridad.

Ya que el esfuerzo tres es igual a cero la ecuación queda:

$$\frac{\sigma 1}{St} = \frac{1}{F.S} \quad [11]$$

Se despeja el esfuerzo 1 y se tiene de dato St y F.S.

St Tabla A-22 Para Acero Inoxidable 304 es 568MPa

F.S es de 5.

$$\sigma 1 = \frac{St}{F.S} \quad [15]$$

$$\sigma 1 = \frac{568Mpa}{5}$$

$$\sigma 1 = 113.6Mpa$$

Como el esfuerzo uno y el esfuerzo en "x" son los mismos igualamos las ecuaciones

$$113.6MPa = \frac{1.886 * 10^{-6} * MN * m}{d^3}$$

Se despeja d

$$d = \sqrt[3]{\frac{1.886*10^{-6}MNm}{113.6MPa}} \quad [16]$$

$$d = 0.003m$$

El diámetro del eje debe ser por lo tanto de 3 milímetros de un acero inoxidable 304, este análisis también se puede realizar para una carga cíclica ya que el eje no siempre va estar estático. Por ello se tiene la siguiente fórmula:

$$\sigma 1 = \frac{Se}{F.S} \quad [17]$$

$$Se = ka * kb * kc * kd * ke * kf * S'e \quad [18]$$

donde:

Se es el límite de resistencia a la fatiga,

S'e es el límite de resistencia a la fatiga en viga rotatoria,

ka Factor de superficie,

kb Factor de forma,

kc Factor de carga,

kd Factor de temperatura,

ke Factor de confiabilidad,

kf Factor de efectos varios.

Primero se calcula el factor de superficie ka

$$ka = a * (Sut)^b \quad [19]$$

donde:

a y b son datos obtenidos de la Tabla 6-2 shigley,

Sut resistencia última a la tracción,

a= 1.58,

b= -0.085,

Sut= 568Mpa.

$$ka = 1.58 * (568)^{-0.085} \quad [20]$$

$$ka = 0.922$$

Factor de tamaño kb

$$kb = (\frac{de}{7.62})^{-0.107} \quad [21]$$

$$de = 0.37 * d \quad [22]$$

donde:

- de es el diámetro efectivo,
- d el diámetro del eje.

Se reemplaza el diámetro efectivo en la fórmula de kb para dejar como incógnita el diámetro.

$$kb = (\frac{0.37 * d}{7.62})^{-0.107}$$

$$kb = 1.382 * d^{-0.107}$$

Factor de carga kc. Ya que el elemento se encuentra en flexión.

kc=1

Factor de temperatura kd. No se tiene especificada una temperatura de trabajo por lo cual se asume que:

kd=1

Factor de confiabilidad ke. Como no se sabe el valor de la confiabilidad se asume que esta es la menor por lo tanto tenemos que:

ke=1

Factores varios kf. Como el eje no muestra ninguna muesca se tiene que:
kf=1

Calculamos S'e el cual está dado por:

$$S'e = 0.5 * Sut \quad [23]$$

$$S'e = 0.5 * 568$$

$$S'e = 284Mpa$$

Con estos datos se calcula el valor de Se

$$Se = ka * kb * kc * kd * ke * kf * S'e$$

$$Se = 0.922 * 1.382 * d^{-0.107} * 1 * 1 * 1 * 1 * 284Mpa$$

$$Se = 361.766 * d^{-0.107}Mpa$$

Reemplazamos el valor de Se y con un factor de seguridad de 5 tenemos que el esfuerzo es:

$$\sigma 1 = \frac{361.766 * d^{-0.107}Mpa}{5}$$

$$\sigma 1 = 72.353 * d^{-0.107}Mpa$$

Finalmente se reemplaza el esfuerzo uno en el esfuerzo "x"

$$72.353 * d^{-0.107}Mpa = \frac{1.886 * 10^{-6}}{d^3}$$

Se despeja el diámetro:

$$d = \sqrt[2.83]{\frac{1.886 * 10^{-6}MNm}{72.353MPa}}$$

$$d = 0.0021m$$

Dado a que el diámetro que se obtuvo en el análisis de carga estática es mayor, es el que

se utilizara en el mecanismo. Por lo tanto, es un eje de acero inoxidable 304 de 3 mm de diámetro.

El mecanismo del eje auto-estabilizador sería como se muestra en la Figura 22.

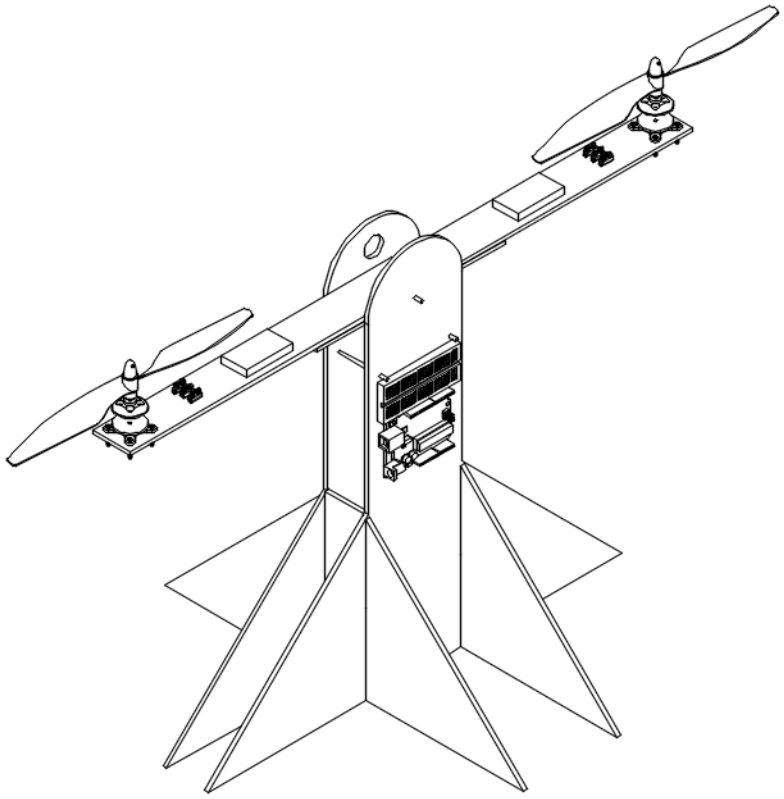


Figura 22. Mecanismo de eje auto estabilizador con dos motores en tres dimensiones.

Diseño Eléctrico y Electrónico

Las conexiones eléctricas se muestran en la Figura 23. Los ESC y los motores se alimentan con un voltaje de 11.5. Cada motor requiere de 4 amperios cuando está funcionando a toda su capacidad, por lo cual se uso una fuente por cado motor. El sensor MPU-6050 necesita de una alimentación de 3.3 voltios que es suministrada por el Arduino.

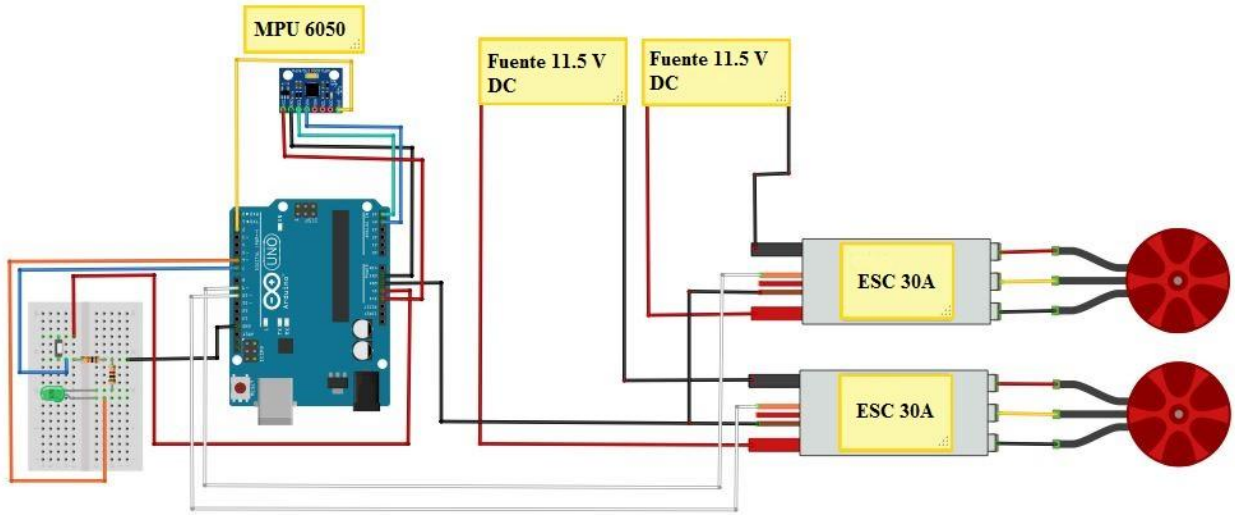


Figura 23. Diagrama eléctrico de eje auto estabilizador con dos motores.

El circuito de encendido y apagado de los motores consta de un pulsador, dos resistencias y un LED, requirió de un cálculo simple para encontrar el valor de las resistencias a utilizar.

Se sabe que la corriente máxima que puede circular en lo pines de entrada y salida del Arduino es de 40mA. por lo tanto; para calcular la primera resistencia que va conectada en la salida del pulsador, se tiene 5 voltios que salen del Arduino y 40 mA. por lo cual aplicando la ley de ohm se tiene:

$$R=V/I \quad [24]$$

$R=5V/0.004A$

$R=1250\Omega$

$R=1.25k\Omega$

Por lo cual se instaló una resistencia de 10kΩ.

Para la segunda resistencia que va en la salida del LED, se tiene que el LED verde consume 2.4 voltios, el suministro de energía es de 5 voltios y 40mA.

$Vtotal= 5V-2.4V$

$Vtotal= 2.6V$

$R=V/I$

$R=2.6V/0.004A$

R=650Ω

Por lo cual, se procedió a utilizar una resistencia de 1kΩ.

Diseño de Control

En la figura 24 se muestra el diagrama de control del eje auto estabilizador con dos motores. La planta consta de un sensor MPU-6050, éste sensor mide los grados de inclinación del eje, y dos actuadores que son los motores brushless con sus respectivos ESC. Uno de los motores funciona como contrapeso; es decir, con una velocidad constante, lo que permite que el eje se incline para el lado opuesto de donde se encuentra éste motor. Mientras que el segundo motor puede variar su velocidad para contrarrestar al motor de velocidad constante y perturbaciones externas, para lograr mantener el eje estabilizado en el set point. El set point se encuentra en cero grados de inclinación.

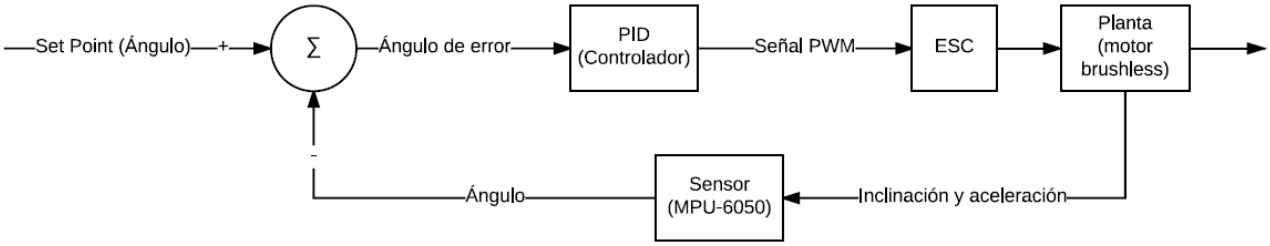


Figura 24. Diagrama de control del eje auto estabilizador con dos motores.

Experimentalmente se obtuvieron los valores de las constantes kp, ki y kd adecuados para que la planta se mantuviera estable, quedando la constante kp con un valor de 2.5, la constante ki con un valor de 0.009 y la constante kd con un valor de 500.

El objetivo del sistema de control es mantener el eje estable en el set point (cero grados de inclinación), es decir el eje debe permanecer horizontal, por lo cual, el ángulo medido por el sensor es comparado respecto al set point; como resultado se obtiene un ángulo de error, el cuál entra al controlador PID, el que transforma el ángulo de error en una señal PWM que es entregada al ESC, el cual varía las rpm (revoluciones por minuto) del motor. El motor inclina el eje dependiendo de las revoluciones por minuto que son entregadas, de esta manera compensa el ángulo de error.

Cuando el ángulo medido por el sensor es negativo, el PID envía una señal a los ESC para que los motores aumenten las revoluciones y de esta manera el motor levanta el eje, compensando el error hasta que el eje se mantenga horizontal a cero grados. De la misma manera

si el ángulo es mayor que el set point, la señal enviada del PID a los ESC hará que los motores disminuyan la velocidad hasta estabilizarse en el set point.

Diseño del Algoritmo de Control

Diseño del algoritmo del sistema de control.

Toda la programación del sistema de control del eje auto estabilizador fue compilada en Arduino, mientras que la visualización gráfica fue realizada en Processing. Para la programación de Arduino se diseñó el siguiente diagrama de flujo que se muestra en la Figura 25.

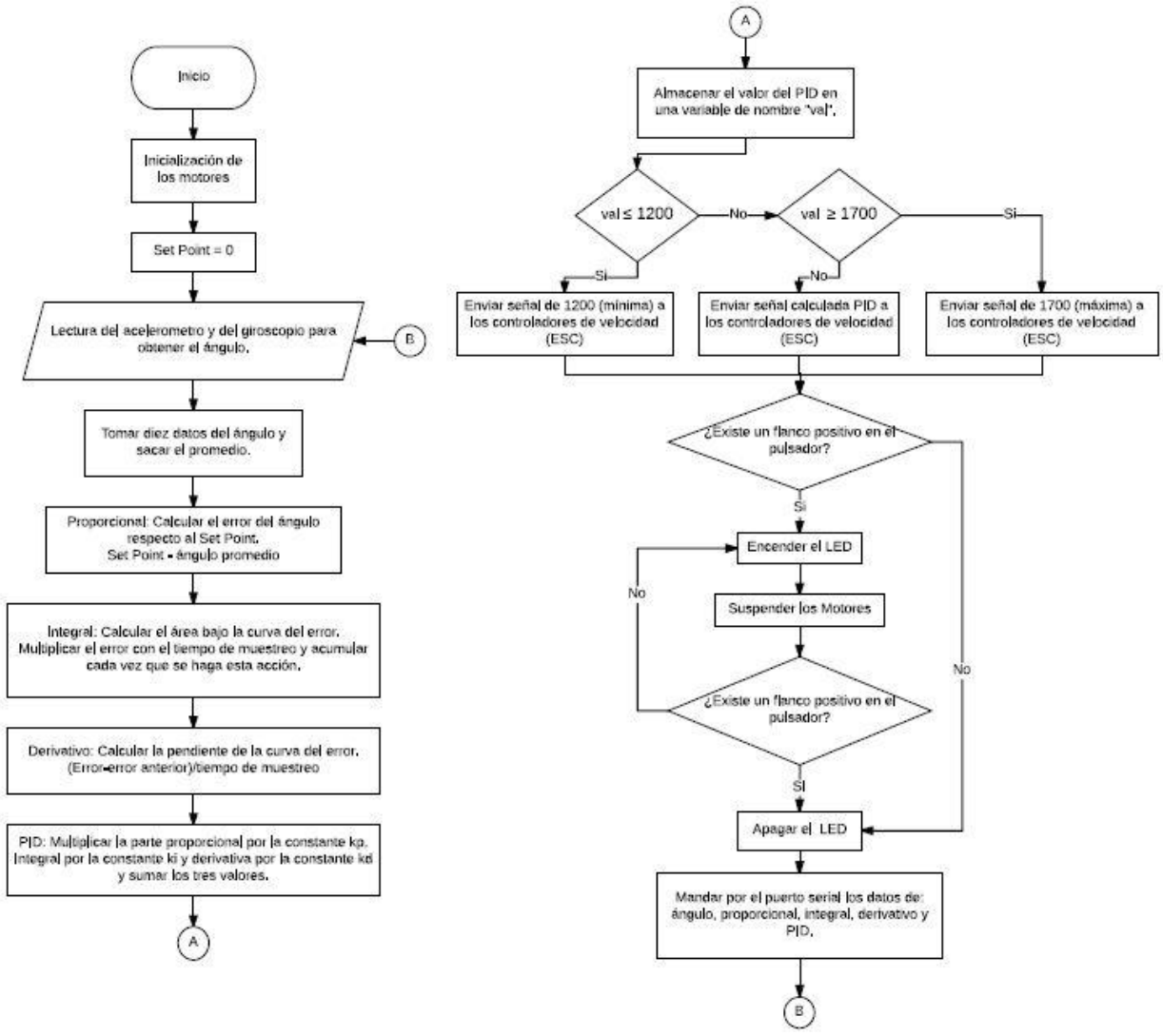


Figura 25. Diagrama de flujo de la programación en Arduino.

Diseño del software en Arduino.

Para conseguir el algoritmo final del eje auto estabilizador con dos motores se requirió seguir una serie de pasos previo a su obtención como: inicialización de los ESC; prueba de velocidad y control de los motores; control del giroscopio; y finalmente incorporación de un algoritmo PID en la planta. Adicionalmente se instaló un pulsador con un LED para encender y apagar los motores. Para ello en la programación, fue necesario la creación de un detector de flancos. Los programas completos tanto de Arduino como Processing se encuentran en los anexos C y D.

Los ESC trabajan con señales PWM al igual que los servos; por ello, en la programación de Arduino es necesario implementar la librería "Servo.h". Para lograr que un ESC se inicialice se debe realizar el siguiente procedimiento: enviar la señal PWM más baja al ESC y después de cinco segundos enviar la señal PWM más alta. Estas señales pueden variar dependiendo del ESC que se utilice, por ello es necesario revisar la hoja técnica de los mismos. De esta manera se determino que la señal PWM más baja de estos ESC es de 1000 y la más alta es de 2000. Con este procedimiento los ESC quedan listos para su funcionamiento.

Los ESC varían las rpm de los motores dependiendo de la señal PWM entregada, por ello, se realizó pruebas y se determinó los rangos de trabajo de los motores. Uno de los motores funciona con una velocidad constante de 3294 rpm, es decir una señal PWM de 1350, mientras que el otro motor, el que varía su velocidad, trabaja en un rango de 2403 a 5373 rpm, es decir una señal PWM de 1200 a 1700. Se trabaja con el rango de 1200 a 1700en vez de un rango de 1000 a 2000 como se encuentra en la hoja técnica por las siguientes razones: de acuerdo a experimentaciones en el laboratorio se determinó que con un ancho de pulso de 1200 el motor se encuentra en el límite de encendido, para valores menores a este el motor se apagaba, y el rango máximo esta en 1700 por precaución, ya que para valores mayores, el mecanismo completo tendía a elevarse.

Para de medir los grados de inclinación del eje se utilizó un sensor MPU-6050, este sensor es un dispositivo I2C por lo cual se requiere el uso de la librería "Wire.h". El sensor recoge los datos de inclinación y aceleración, a partir de estos datos se obtiene el ángulo en el que se encuentra el eje. Ya que el ángulo que se obtiene del sensor, se ve afectado por las vibraciones de los motores y el ruido eléctrico; se realiza un promedio por cada diez datos del

ángulo.

Utilizando la fórmula 2 este ángulo es comparado con un set point y con ello se obtiene el error, que será multiplicado por una constante denominada kp (constante proporcional), obteniendo así la parte proporcional.

El error integral se obtiene utilizando la fórmula 3, se calcula el área bajo la curva del error, para facilitar el cálculo de esta área; se divide el área total en áreas rectangulares más pequeñas y se suman todas las áreas rectangulares, como se muestra en la Figura 26. La base de cada área rectangular corresponde al tiempo que el programa se demora en ejecutarse cada vez; esto ocurre cada 23 milisegundos (± un milisegundo), mientras que el altura de cada rectángulo corresponde al valor leído del error en ese mismo instante. Por lo tanto, el error integral, es acumulativo. El error integral se multiplica por su constante ki (constante de integración).

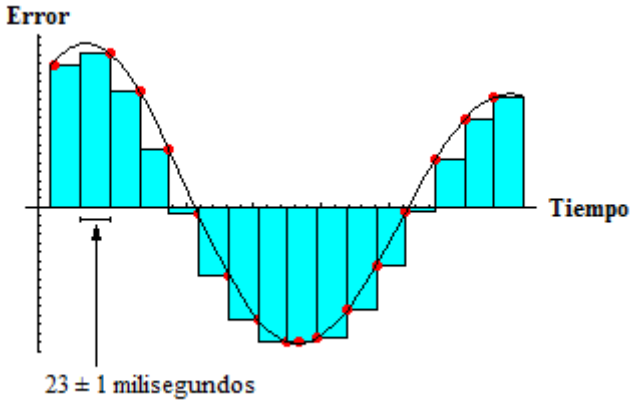


Figura 26. Cálculo del área bajo la curva del error. Tomado de: (Universidad de Granada, 2001).

Para obtener la parte derivativa, es necesario medir la pendiente, para lo cual se utiliza la ecuación de la pendiente:

$$m = \frac{(y_2 - y_1)}{(x_2 - x_1)} \qquad [25]$$

En donde "m" es la pendiente, $(y_2 - y_1)$ es la diferencia entre el error y el error anterior, y $(x_2 - x_1)$ es el tiempo de 23 milisegundos (± 1 milisegundo). Utilizando la formula 4, la pendiente se multiplica por la constante kd (constante derivativa).

Finalmente, se suma la parte proporcional, integral y derivativa, obteniendo la señal de control PWM, sin embargo; esta señal no siempre es adecuada para los ESC, por ello se crea límites para que esta señal no sobrepase los limites previamente establecidos de 1200 a 1700. De esta manera si la señal del PID es mayor a 1700 esta es transformada a 1700 y si la señal es menor a 1200 esta es transformada a 1200.

Diseño para la visualización de las señales de control.

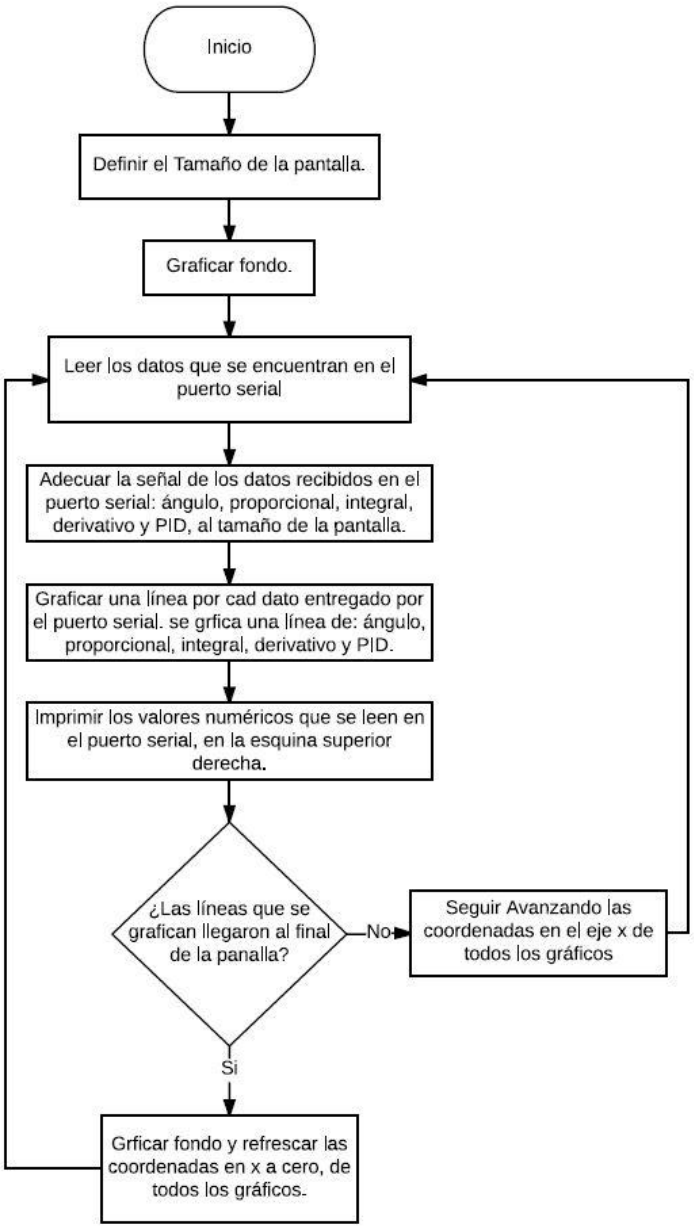


Figura 27. Diagrama de flujo de la programación en Processing.

Para visualizar el funcionamiento del sistema de control se utilizó la plataforma de software Processing, dado que es un software libre diseñado primordialmente para aplicaciones gráficas. El código sigue el mismo funcionamiento que un osciloscopio tal como se representa por medio del diagrama de flujo de la Figura 27. El gráfico, permite la visualización del funcionamiento del PID a tiempo real, se puede comparar el ángulo de inclinación y las correcciones realizadas por el sistema de control. Este gráfico facilita el entendimiento del funcionamiento del algoritmo de control.

Para graficar los datos más importantes, tales como: el ángulo, el error proporcional, integral y derivativo, y el valor total del PID, es necesario trabajar conjuntamente entre Arduino y Processing. Por lo cual en la programación de Arduino se debe enviar por el puerto serial los datos del ángulo, el error proporcional, integral y derivativo, y PID separados por comas entre ellos y en el último dato un salto de línea.

Se debe asignar el tamaño de la pantalla en la que se va a trabajar. La pantalla es de 1350 píxeles horizontalmente y 720 píxeles verticalmente. Se graficó el fondo dividiendo la pantalla en dos partes iguales, la parte superior es donde se grafica la señal del error; mientras que, la parte inferior, es la que grafica la parte proporcional, integral, derivativa y PID. Tanto la pantalla de arriba como la de abajo están divididas en 18 secciones, que indican, en la superior los ángulos de inclinación de 10 en 10, mientras que en la inferior los valores de 1215 a 7155 rpm en la parte izquierda y los valores de la señal PWM de 1000 a 2000 en la parte derecha. La pantalla tiene adicionalmente líneas verticales que indican el tiempo, estas líneas representan un segundo. En la Figura 28 se puede observar como esta distribuida la pantalla del programa en Processing.

Para graficar se da las instrucciones de recopilar los datos del puerto serial, se indica que están separados por una coma y que se lean cada vez que existe un salto de línea. Se adecua los datos obtenidos desde el puerto serial, para que avancen en la pantalla asignada sin salirse de los márgenes. Para que la línea del gráfico avance conforme los datos recibidos del puerto serial, se utilizó dos variables una que indica la posición actual y otra la posición anterior, se debe indicar que la variable de posición anterior almacene el valor que se encuentra en posición actual; de este modo cada vez que se ejecute esta orden el valor que se encuentre la posición actual va a pasar a la posición anterior. Con los dos puntos que se obtiene, punto de posición anterior y punto de posición actual, se grafica una línea. Se sigue el mismo procedimiento para todos los datos obtenidos del puerto serial. Cuando la posición de los gráficos se encuentran al final de la

pantalla en el eje horizontal, se imprime el fondo encima de todo, de esta manera no se grafican las líneas sobre de las anteriores.

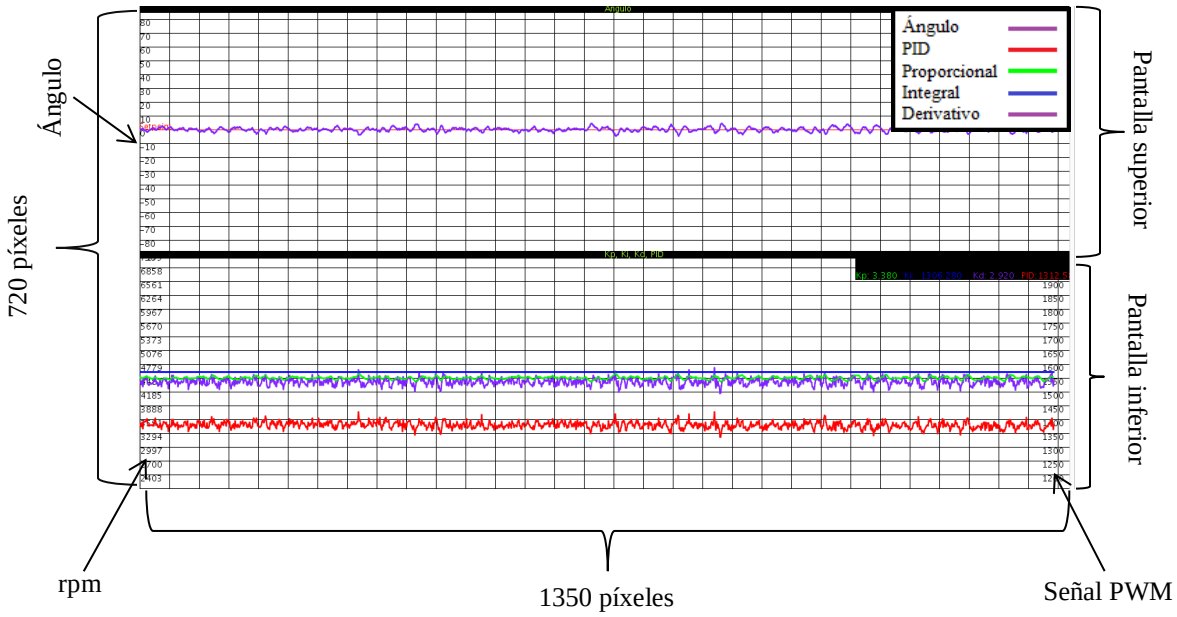


Figura 28. Distribución de la pantalla de Processing.

Para que se grafiquen los valores numéricos que tiene cada dato enviado por el puerto serial, se recoge estos datos sin modificarlos y se les asigna una posición en la pantalla, se dibujó un recuadro en la misma posición de los datos y cada que se recibe un dato se imprime el mismo recuadro, de esta manera los valores no se escriben sobre los anteriores.

Resultados

Discusión

Al eje auto estabilizador implementado con el algoritmo de control PID, se le realizó 12 pruebas, de las cuales, la primera fue con el motor de contra peso y sin ningún peso adicional, mientras que en las 11 pruebas restantes se realizaron con el motor de contrapeso y adicionalmente un peso en el extremo del eje, donde se encuentra el motor controlado por el PID, como se muestra en la Figura 41. Se empezó con un peso de 26 gramos y se fue incrementando el peso, 16 gramos más cada prueba hasta los 186 gramos.

En la Figura 29 se observa el funcionamiento del eje auto estabilizador, sin peso y sin alteraciones. La línea morada superior indica el ángulo en el que se encuentra; por lo cual se ve que el mecanismo esta estable en un ángulo cercano a cero grados, las variaciones son mínimas. El motor de velocidad variable tiene una velocidad alrededor de 3294 rpm.

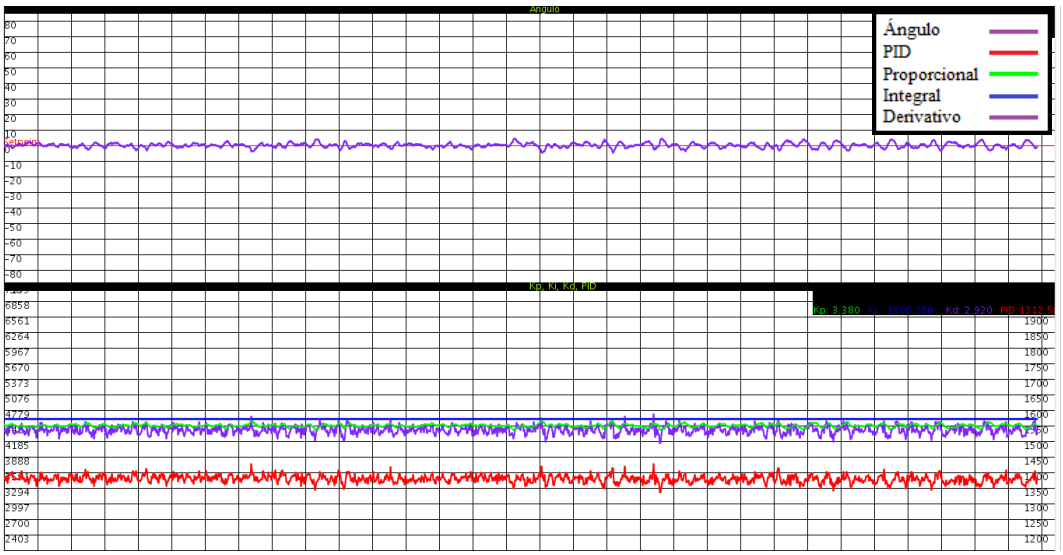


Figura 29. Gráfico del programa realizado en Processing. Eje auto estabilizador sin peso adicional.

En la segunda prueba que se realizó, se adicionó el primer peso de 26 gramos, en la Figura 30 se puede apreciar tanto en el círculo rojo superior como en el inferior, el pico en donde fue aplicado el peso. Este peso causo una mínima desestabilización. Se puede apreciar que la señal del PID (línea roja del gráfico inferior) subió aproximadamente a 1400, lo que indica que el motor de velocidad variable aumentó su velocidad a 3591 rpm para compensar el peso adicional.

Adicionalmente se puede observar que el mecanismo tardó menos de un segundo para estabilizarse.

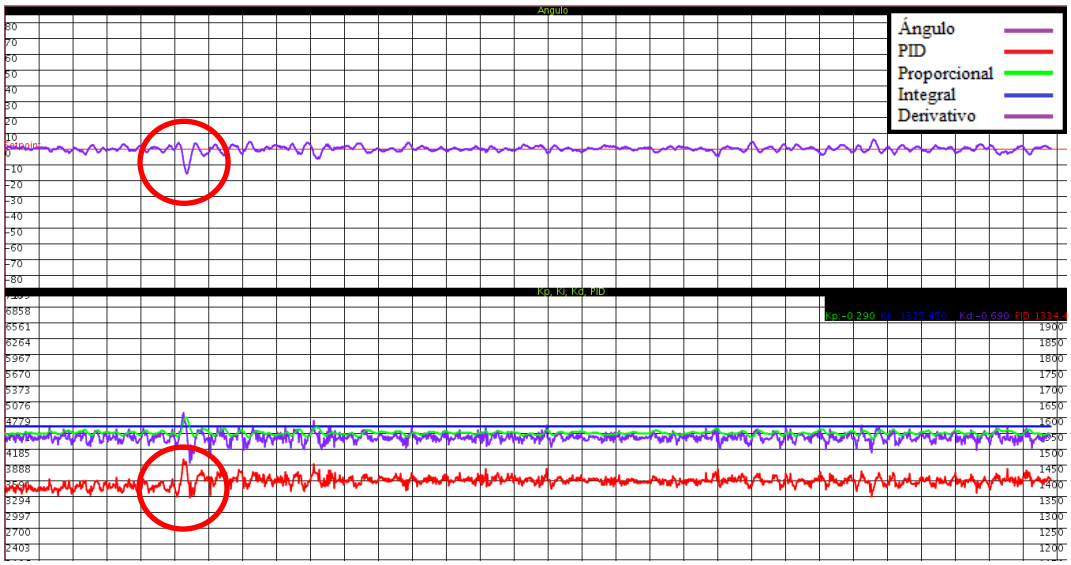


Figura 30. Gráfico del programa realizado en Processing. Eje auto estabilizador con un peso adicional de 26 gramos.

La prueba número tres muestra datos similares que la prueba dos, sin embargo el pico que se formó al instante de adicionar el peso es mayor al anterior, como se puede apreciar en la Figura 31.

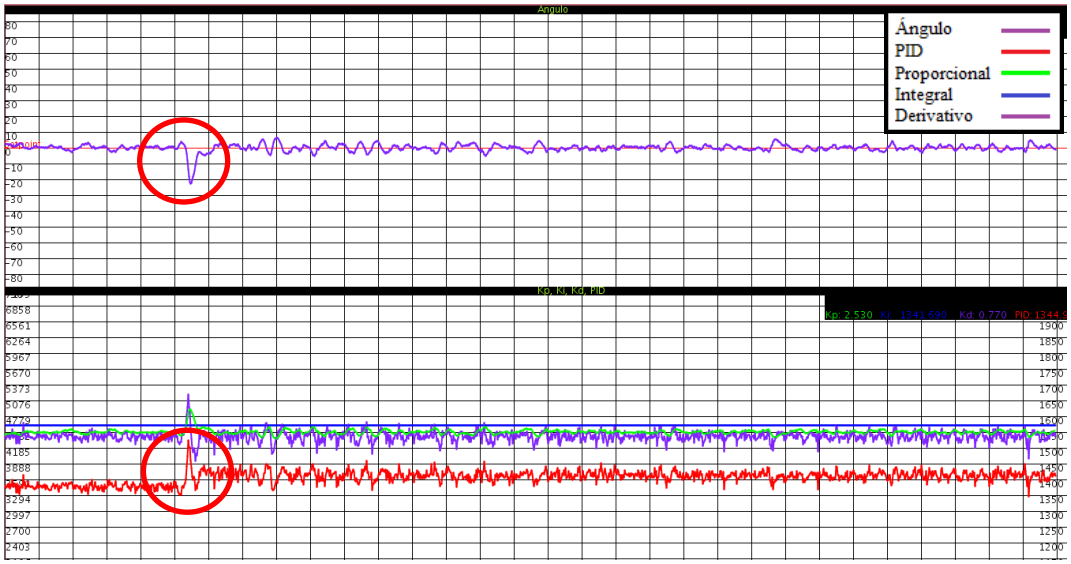


Figura 31. Gráfico del programa realizado en Processing. Eje auto estabilizador con un peso adicional de 42 gramos.

En la figura 32, Se puede observar la prueba número cuatro, en la cual al incrementar el contra peso el valor de la señal del PID aumenta para compensar el peso adicional y llegar a la estabilización. Las pruebas cinco y seis muestran resultados similares tal como se aprecian en las Figuras 33 y 34 respectivamente.

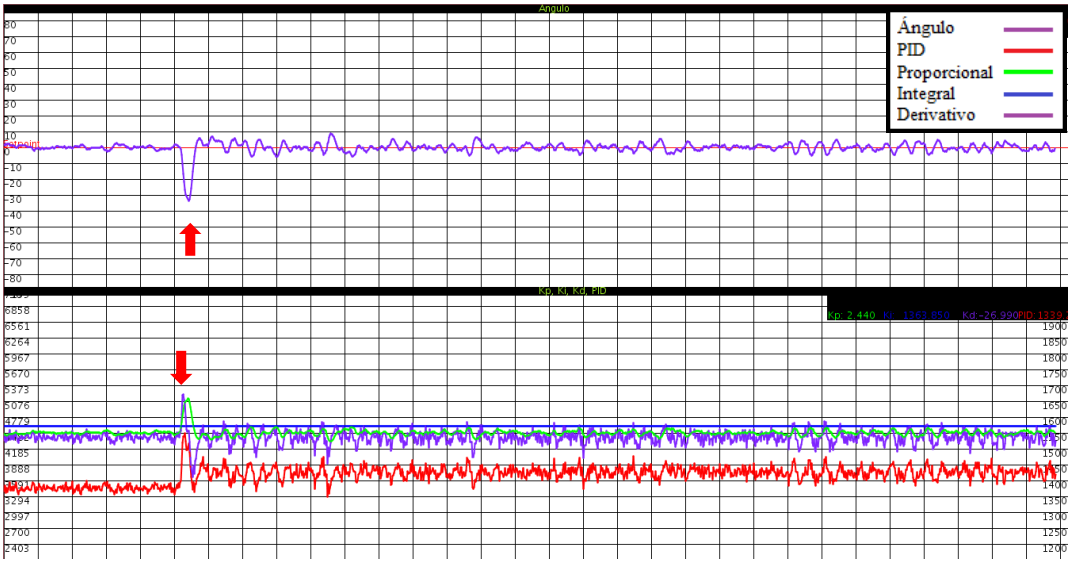


Figura 32. Gráfico del programa realizado en Processing. Eje auto estabilizador con un peso adicional de 58 gramos.

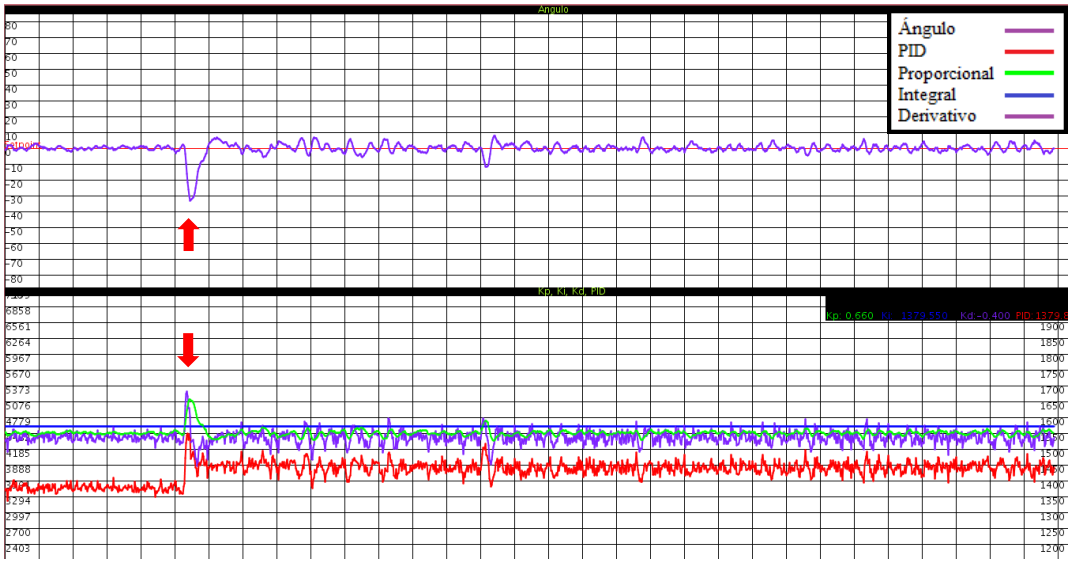


Figura 33. Gráfico del programa realizado en Processing. Eje auto estabilizador con un peso adicional de 74 gramos.

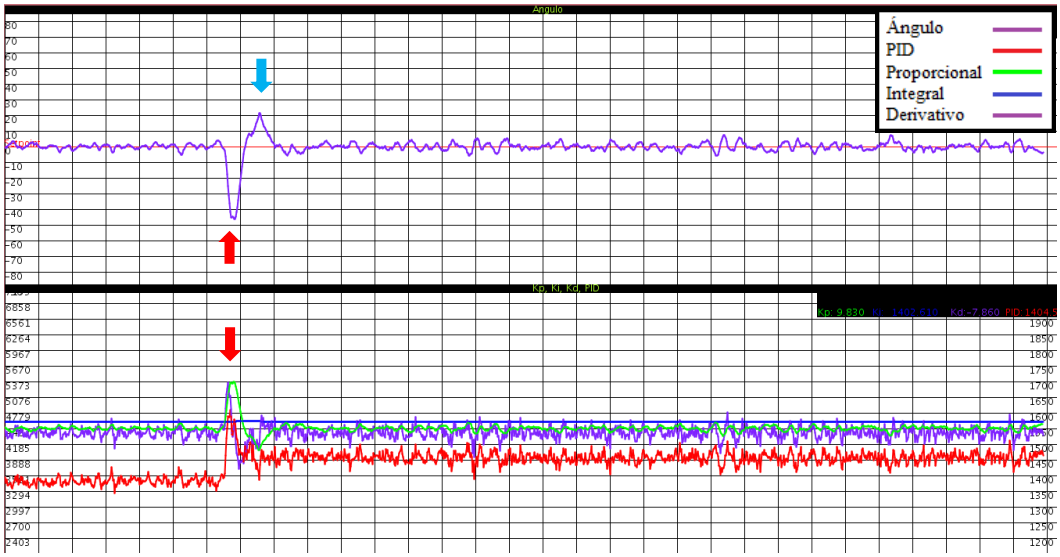


Figura 34. Gráfico del programa realizado en Processing. Eje auto estabilizador con un peso adicional de 90 gramos.

En la Figura 34 se observa la prueba seis, donde la línea morada tiene dos picos, uno negativo (flecha roja) y otro positivo (flecha azul). El pico positivo es causado debido a que el algoritmo intenta compensar el error causado por el peso aplicado, dando una señal PID sobre dimensionada al motor, con el fin de poder compensar con más rapidez el error. Por lo cual se puede observar que el pico en la señal de PID (línea roja del gráfico) es más grande al instante de aplicar el peso, y posteriormente se aprecia que se reduce hasta estabilizarse y mantener el mecanismo cercano al set point. En los casos siete, ocho, nueve, diez y once se tiene resultados similares donde están dos picos, tanto el positivo como el negativo, tal como se puede apreciar en las Figuras 35, 36, 37, 38 y 39 respectivamente.

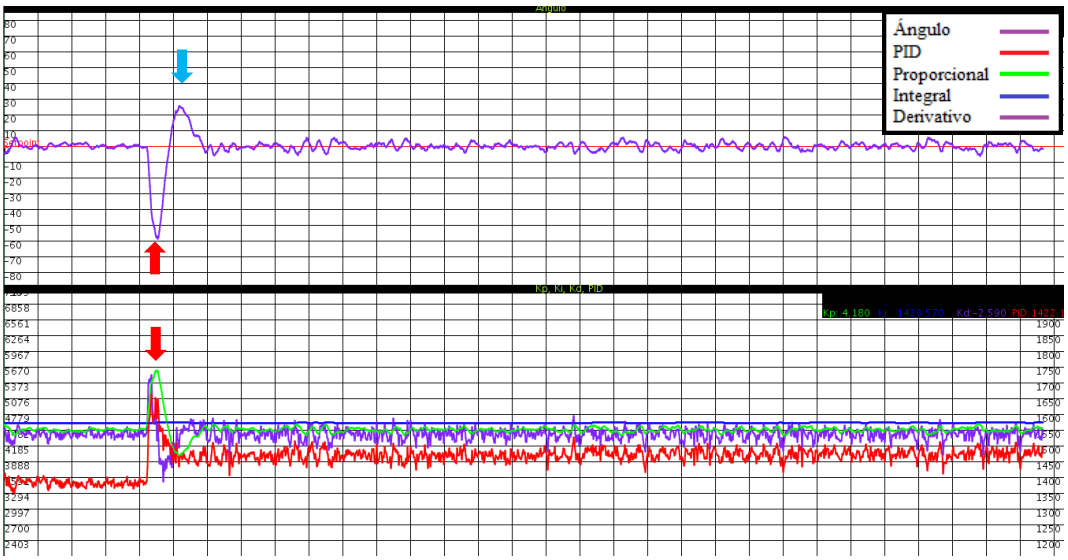


Figura 35. Gráfico del programa realizado en Processing. Eje auto estabilizador con un peso adicional de 106 gramos.

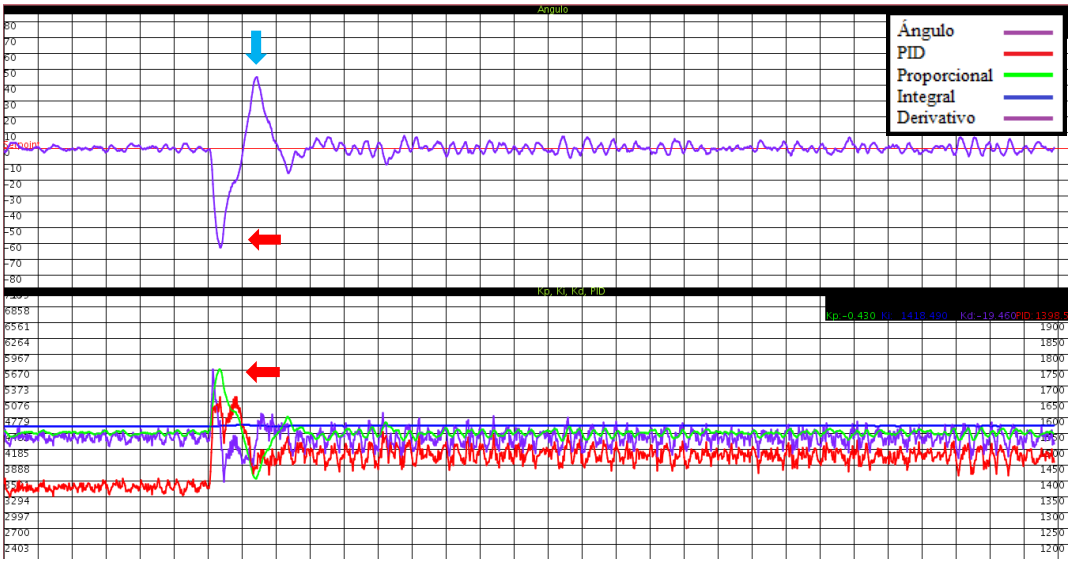


Figura 36. Gráfico del programa realizado en Processing. Eje auto estabilizador con un peso adicional de 122 gramos.

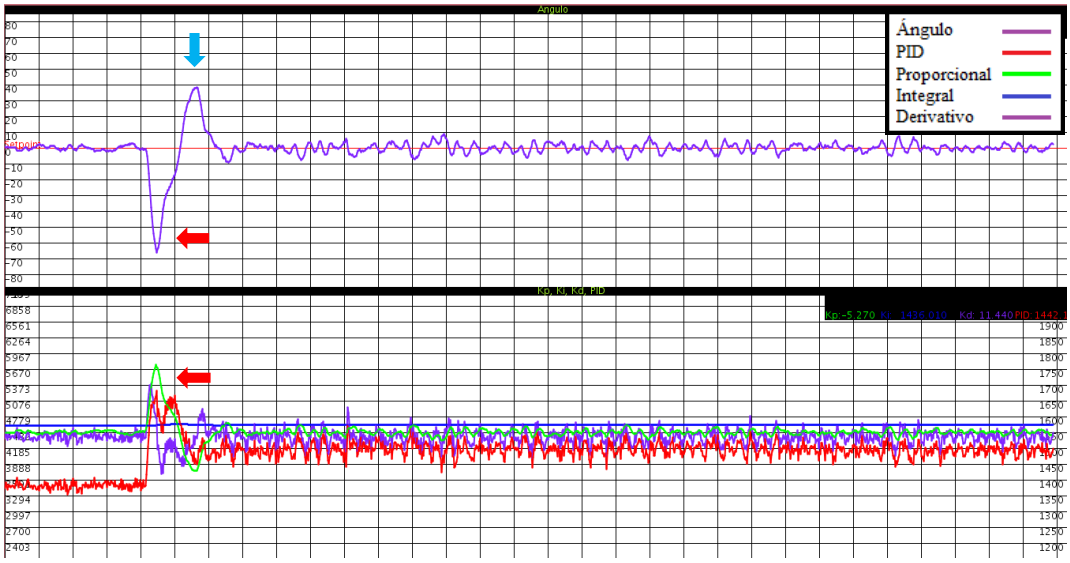


Figura 37. Gráfico del programa realizado en Processing. Eje auto estabilizador con un peso adicional de 138 gramos.

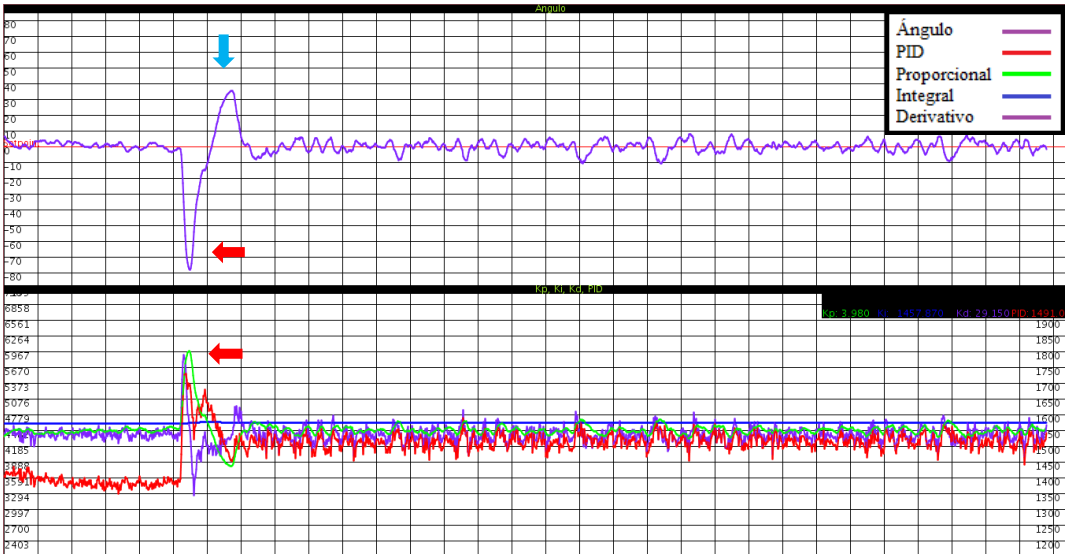


Figura 38. Gráfico del programa realizado en Processing. Eje auto estabilizador con un peso adicional de 154 gramos.

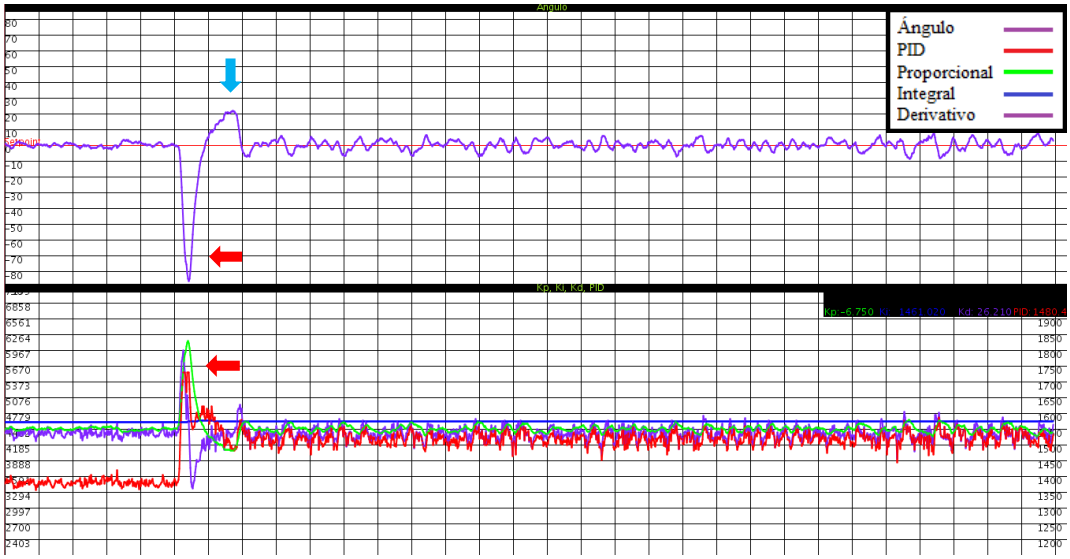


Figura 39. Gráfico del programa realizado en Processing. Eje auto estabilizador con un peso adicional de 170 gramos.

La última prueba realizada con peso se puede observar en la Figura 40, donde se aprecia que el peso aplicado es superior al peso que el eje auto estabilizador puede compensar; por lo cual, se observa que la línea morada que indica el ángulo de inclinación, no se encuentra en el set point; sino, por debajo del mismo. Y a su vez se puede ver que la señal enviada del PID es la mayor posible; es decir, 1700 lo que indica que el motor se encuentra a 5375 rpm, sin embargo no se llega a estabilizar.

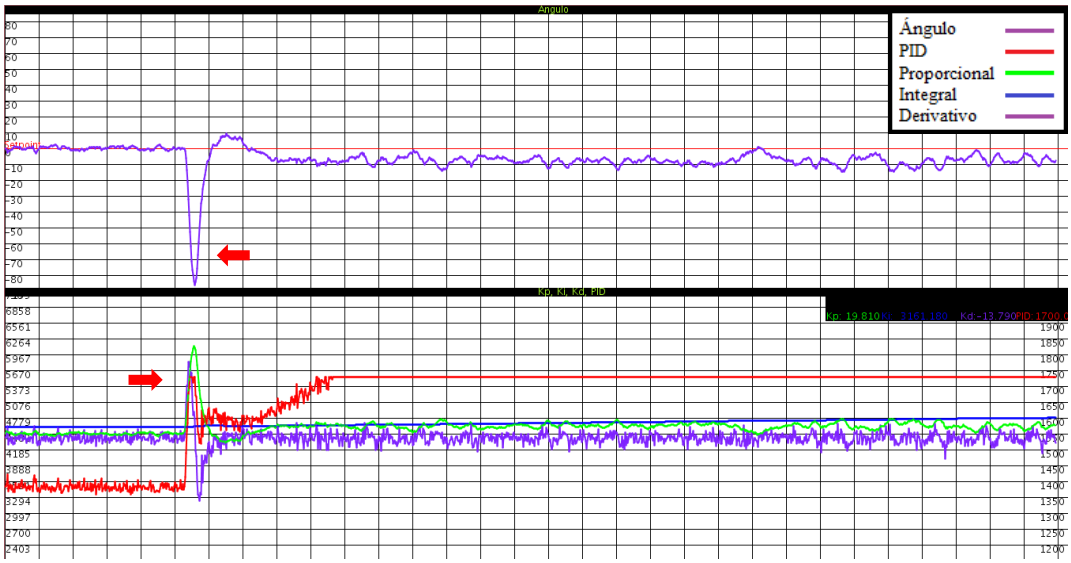


Figura 40. Gráfico del programa realizado en Processing. Eje auto estabilizador con un peso adicional de 186 gramos.

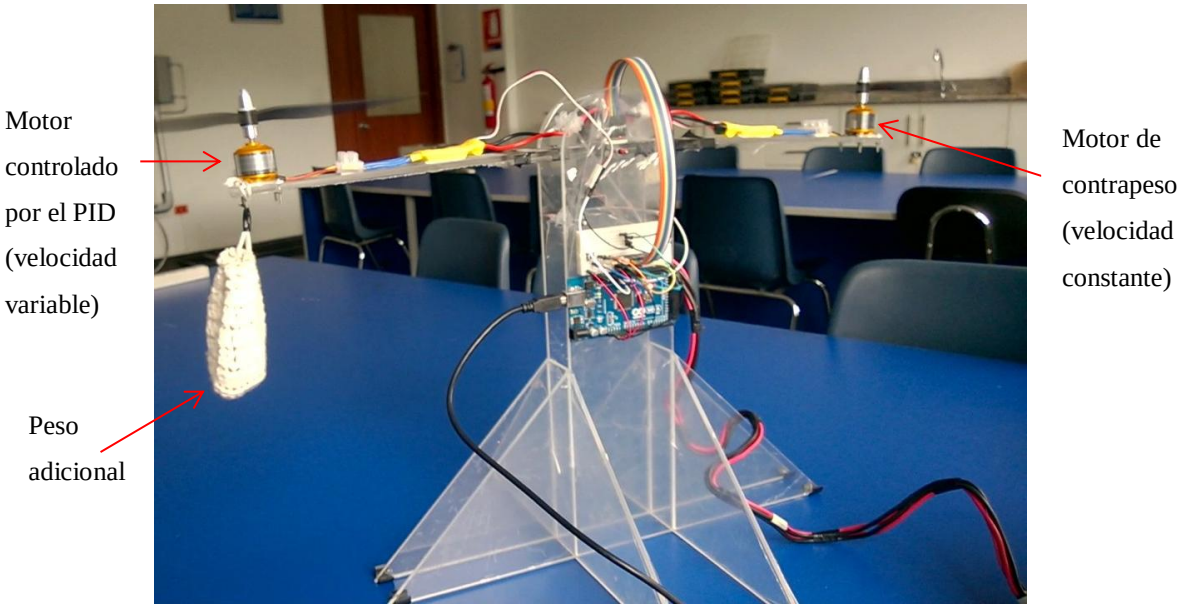


Figura 41. Gráfico del eje auto-estabilizador funcionando y con peso adicional.

Conclusiones

Es necesario utilizar una fuente de poder para cada uno de los motores, ya que las fuentes utilizadas solo proveen hasta 3 amperios, mientras que los motores cuando están trabajando en rangos altos de velocidad requieren más de 3 amperios para su buen funcionamiento. Por lo cual una sola fuente para los dos motores no abastecería el amperaje.

El acrílico transmite las vibraciones causadas por los motores, debido a esto, el valor del ángulo medido se ve afectado, ya que dichas vibraciones son captadas por el acelerómetro confundiendo así la medida del ángulo.

En el proceso de implementación del algoritmo PID se descubrió que no es necesario adecuar la señal del PID por medio de la función “map”; sino que, debe ser entregada directamente a los ESC, ya que las constantes del algoritmo (kp, ki, kd) ya están adecuadas para los ESC.

Debido a las vibraciones y al ruido eléctrico, es indispensable sacar un ángulo promedio, para lograr tener un dato más certero. Se realizaron pruebas de 100, 50, 20, 10 y 3 datos para promediar. Por lo cual se llegó a la siguiente conclusión: este procedimiento puede ser contraproducente, en el sentido que, si se escoge que se promedie más de 10 datos del ángulo el programa pierde velocidad y por lo tanto, la velocidad de reacción se ve afectada, mientras que si

se escoge tres datos para promediar, no son suficientes datos para tener un promedio certero.

La velocidad de ejecución del programa es de 23 milisegundos ± un milisegundo de diferencia; por lo tanto el programa realiza aproximadamente 43 correcciones por segundo.

Los ESC producen ruido electromagnético que afecta notablemente al sensor MPU 6050, por lo cuál la posición del sensor debe ser lo más alejada posible de los ESC.

El eje auto estabilizador es capaz de estabilizarse gracias al algoritmo de control PID, el cual logra compensar los errores en un periodo de tiempo corto. Para pesos adicionales entre 26 a 74 gramos, le toma un segundo hasta estabilizarse, mientras que para pesos entre 90 y 170 gramos le toma dos segundos para estabilizarse.

El programa de visualización ayuda a comprender el funcionamiento del algoritmo de control PID, ya que muestra a tiempo real las perturbaciones y correcciones realizadas.

Este algoritmo de auto estabilización puede ser utilizado para futuros proyectos, como cuadricópteros.

Recomendaciones

Se recomienda eliminar todos los posibles cambios adicionales, tales como el balance en los motores y el juego en el eje de del balancín para ganar estabilidad.

Se recomienda realizar un análisis de vibraciones de la estructura para evitar el ruido mecánico.

Se recomienda seleccionar un material que no transmita tantas vibraciones como el acrílico. De esta manera se reducirá el ruido mecánico y se obtendrán datos más certeros del ángulo.

Se recomienda alejar el sensor de los ESC ya que estos emiten ruido electromagnético y causa errores en la medición del ángulo.

Se recomienda utilizar una fuente de 12 voltios y 8 amperios por motor.

Referencias

5hertz. (2014). *http://5hertz.com*. Retrieved from http://5hertz.com/tutoriales/?p=431

Acelerometros y sensores de vibracion. (n.d.). Retrieved from http://projecte-hermes.upc.edu/Enginyeria_Aeroespacial/3A/Mec%C3%A0nica%20II/Laboratori/Acelerometros%20y%20sensores%20de%20vibracion.pdf

Amazon. (2016). *Amazon.com*. Retrieved from http://www.amazon.in/Brushless-Motor-Speed-Controller-REES52-COM/dp/B0153E00WU?ie=UTF8&SubscriptionId=AKIAIFFSIGLKIIRCM2BA&camp=2025&creative=165953&creativeASIN=B0153E00WU&linkCode=sp1&tag=applinx-21

Anderson, D. P. (2013). *nBot Balancing Robot*. Retrieved from http://www.geology.smu.edu/~dpa-www/robo/nbot/

Angulo Bahón, C., & Raya Giner, C. (2004). Tecnología de sistemas de control. In C. Angulo Bahón, & C. Raya Giner, *Tecnología de sistemas de control* (p. 34). Cataluña: Universidad Politécnica de Cataluña.

Angulo Bahón, C., & Raya Giner, C. (2004). Tecnología de sistemas de control. In C. Angulo Bahón, & C. Raya Giner, *Tecnología de sistemas de control* (p. 35). Cataluña: Universidad Politécnica de Cataluña.

ARDUINO. (2016). *ARDUINO*. Retrieved from https://www.arduino.cc/

Arduino. (2016). *http://playground.arduino.cc*. Retrieved from http://playground.arduino.cc/Main/MPU-6050

Bao, M. H. (2000). In M. H. Bao, *Micro Mechanical Transducers: Pressure Sensors, Accelerometers and Gyroscopes* (Vol. VIII). Elsevier B.V.

Baraza, C. (2010, Noviembre 21). *quadruino.com*. Retrieved from http://www.quadruino.com/guia-2/materiales-necesarios-1/esc

Barrett, S. F. (2010). *Arduino Microcontroller Processing for Everyone!* Morgan & Claypool Publishers.

Barro, J. (2009, Noviembre 02). *Polo Estable*. Retrieved from https://poloestable.wordpress.com/2009/11/02/origenes-del-pid/

Budynas, R. G., & Nisbett, K. J. (2008). *Diseño en ingeniería mecánica de Shigley* (Octava edición ed.). McGraw-Hill Interamericana, S.A. DE C.V.

Burnie, D., Davies, K., Dixon, D., Glover, D., Gribbin, J., Gribbin, M., et al. (n.d.). *Gran Enciclopedia Escolar.* Barcelona: Planeta-De Agostini.

Chamorro Hernández, W. O., & Medina Mora, J. L. (2013). *Ensamblaje y Control de un Cuadricóptero.* Quito: Escuela Politécnica Nacional Proyecto previo a la obtención del título de ingeniero en electrónica y control.

Cook, D. (2015). *Robot Building for Beginners.* Apress.

Egox. (n.d.). *www.egox.com.ar*. Retrieved from http://www.acrilicosegox.com.ar/detalle.php?a=manual-tecnico-del-acrilico&t=8&d=8

García, P. E., Hidalgo, M., Loza, J. L., & Muñoz, J. (2013). *PRÁCTICAS CON ARDUINO. EDUBÁSICA. PRÁCTICAS PARA 4º ESO.* Albacete.

Gonzalez-Longat, F. M. (n.d.). *Introduccón a la Teoría de Control*. Retrieved from www.giaelec.org: http://fglongatt.org/OLD/Archivos/Archivos/Control_I/Capitulo1TC-2007.pdf

Goussault, R. (n.d.). *Quadcopter Project.*

Guerrero Tavares, J. N. (2014). *Imprementación de un algoritmo de control para el lebantamiento de un helicóptero de cuatro rotores.* Santiago de Querétaro: (Tesis de Maestría, Universidad Autónoma de Querétaro).

Hassenplug. (2002). *Steve's LegWay*. Retrieved from http://www.teamhassenplug.org/robots/legway/

HeTPro. (n.d.). *HeTPro*. Retrieved from hetpro-store.com: https://hetpro-store.com/motor-brushless-a2212-13t-1000-kv/

Kamen, D. (2011). *segway.com*. Retrieved from http://www.segway.com/

Kane, J. W., & Sternheim, M. M. (2007). Física. In J. W. Kane, & M. M. Sternheim, *Física* (p. 180). Barcelona: Editorial Reverté.

Krossblade Aerospace Systems LLC. (2016). *Krossblade Aerospace*. Retrieved from http://www.krossblade.com/history-of-quadcopters-and-multirotors/

Kuo, B. C. (1996). *Sistemas de control automático.* México: Prentice Hall.

Linke, D. (2011). *Two-Wheel Self Balancing Robot.* Massachusetts: (A Major Qualifying Project submitted to the faculty of Worcester Polytechnic Institute in partial fulfillment of the requirements for the Degree of Bachelor of Science in Electrical and Computer Engineering).

Linke, D. (2011). *Two-Wheel Self Balancing Robot.* Worcester: Major Qualifying Project submitted to the faculty of Worcester Polytechnic Institute.

López, A., Dorado, A., Manso, J., Méndez, M. L., Calvo, A., Saavedra, M., et al. (1999). *Enciclopedia ESTUDIANTIL.* Madrid: CULTURAL, S.A.

Mataix Aracil, C. (1965). *ALGEBRA PRACTICA.* Madrid: DOSSAT, S.A.

Mazzone, V. (2002). *Controladores PID.* Quilmes: Universidad Nacional de Quilmes.

Minorsky, N. (n.d.). *ieeecss.org*. Retrieved from Nicolas Minorsky and the Automatic Steering of Ships: http://ieeecss.org/CSM/library/1984/nov1984/w10-15.pdf

Moyano, J. E. (n.d.). *Arduino PID - Guía de uso de la librería.* Retrieved from brettbeauregard.com: http://brettbeauregard.com/blog/wp-content/uploads/2012/07/Gu%C3%ADa-de-uso-PID-para-Arduino.pdf

Nussey, J. (2013). *Arduino For Dummies.* Chichester: John Wiley & Sons.

Ñeco, R., Reinoso, O., García, N., & Aracil, R. (2003). *Apuntes de sistemas de control.* Elche: Editorial Club Universitario.

Ogata, K. (2003). *Ingeniería de control moderna.* Madrid: Pearson Educación.

Ortiz Padilla, V. G., & Pulla Arévalo, P. R. (2012). *Diseño y Construcción de un Cuadricóptero a Control Remoto.* Sangolqí: Escuela Politécnica del Ejercito (Proyecto de Grado).

Prometec.net. (n.d.). *Prometec.net*. Retrieved from http://www.prometec.net/imu-mpu6050/

Quadcopter Arena. (2016). *QUADCOPTER ArenA*. Retrieved from http://quadcopterarena.com/

Rebeiz, G. M. (2003). In *RF MEMS Theory, Design, and Technology.* New Jersey: John Wiley & Sons.

Rodríguez Celi, M. (n.d.). *Universidad Simón Bolívar*. Retrieved from http://prof.usb.ve/mirodriguez/control/Introduccion_sistemas_control/un_poco_de_historia.html

Sears, F. W., & Zemansky, M. W. (1970). *Física General.* Madrid: Aguilar S.A.

Titus, J. (2012, Agosto 20). *ECN*. Retrieved from www.ecnmag.com: https://www.ecnmag.com/article/2012/08/careful-designers-get-most-brushless-dc-motors

Universidad de Granada. (2001, Noviembre 2001). *www.ugr.es*. Retrieved from http://www.ugr.es/~dpto_am/docencia/cie_mat_calculo/Integral/Links/Integral%20de%20Riemann_lnk_2.html

Visioli, A. (2006). *Practical PID Control.* Springer Science & Buisness Media.

Weber, R. L., White, M. W., Mannig, K. V., & Febrer, J. (1954). *Física General Moderna.* Brcelona: Editorial Reverté S.A.

Yedamale, P. (2003). Brushless DC (BLDC) motor fundamentals. microchip Technology Inc.

Anexos

Anexo A.

Tabla A-22

Resultados de ensayos a la tensión de algunos metales* *Fuente:* J. Datsko, "Solid Materials", capítulo 32, en Joseph E. Shigley, Charles R. Mischke y Thomas H. Brown, Jr. (editores en jefe). *Standard Handbook of Machine Design*, 3a. ed., McGraw-Hill, Nueva York, 2004, pp. 32.49-32.52.

			Resistencia (a la tensión)					
Número	Material	Condición	Fluencia S_y, MPa (kpsi)	Última S_u, MPa (kpsi)	A la fractura, σ_f, MPa (kpsi)	Coeficiente σ_0, MPa (kpsi)	Resistencia a la deformación, exponente m	Resistencia a la fractura ϵ_f
1018	Acero	Recocido	220 (32.0)	341 (49.5)	628 (91.1)†	620 (90.0)	0.25	1.05
1144	Acero	Recocido	358 (52.0)	646 (93.7)	898 (130)†	992 (144)	0.14	0.49
1212	Acero	HR	193 (28.0)	424 (61.5)	729 (106)†	758 (110)	0.24	0.85
1045	Acero	TyR 600°F	1 520 (220)	1 580 (230)	2 380 (345)	1 880 (273)†	0.041	0.81
4142	Acero	TyR 600°F	1 720 (250)	1 930 (210)	2 340 (340)	1 760 (255)†	0.048	0.43
303	Acero inoxidable	Recocido	241 (35.0)	601 (87.3)	1 520 (221)†	1 410 (205)	0.51	1.16
304	Acero inoxidable	Recocido	276 (40.0)	568 (82.4)	1 600 (233)†	1 270 (185)	0.45	1.67
2011	Aleación de aluminio	T6	169 (24.5)	324 (47.0)	325 (47.2)†	620 (90)	0.28	0.10
2024	Aleación de aluminio	T4	296 (43.0)	446 (64.8)	533 (77.3)†	689 (100)	0.15	0.18
7075	Aleación de aluminio	T6	542 (78.6)	593 (86.0)	706 (102)†	882 (128)	0.13	0.18

*Los valores se tomaron de una o dos coladas y se considera que pueden obtenerse usando especificaciones de compra. La deformación por fractura puede variar hasta en 100%.
†Valor derivado.

Figura 42. Tabla A-22 obtenida de: (Budynas & Nisbett, 2008)

Anexo B.

Acabado superficial	Factor a S_{ut}, kpsi	Factor a S_{ut}, MPa	Exponente b
Esmerilado	1.34	1.58	−0.085
Maquinado o laminado en frío	2.70	4.51	−0.265
Laminado en caliente	14.4	57.7	−0.718
Como sale de la forja	39.9	272.	−0.995

De C. J. Noll y C. Lipson, "Allowable Working Stresses", en *Society for Experimental Stress Analysis*, vol. 3. núm. 2, 1946, p. 29. Reproducido por O. J. Horger (ed.), *Metals Engineering Design ASME Handbook*, McGraw-Hill, Nueva York. Copyright © 1953 por The McGraw-Hill Companies, Inc. Reproducido con autorización.

Figura 43. Tabla 6-2. Parámetros en el factor de la condición superficial, obtenido de: (Budynas & Nisbett, 2008)

Anexo C.

Programa de Arduino:

```
#include <Wire.h>

#include <Servo.h>


#define MPU 0x68 //Direccion I2C de la IMU

#define A_R 16384.0 //Ratios de conversion

#define G_R 131.0 //Ratios de conversion

#define RAD_A_DEG = 57.295779 //Conversion de radianes a grados 180/PI


//MPU-6050 da los valores en enteros de 16 bits

//Valores sin refinar

int16_t AcX, AcY, AcZ, GyX, GyY, GyZ;

 //Angulos

float Acc[2];

float Gy[2];

float Angulo;

//Variables par el PID

unsigned long Tiempo_de_medicion = 1;

float Temp;

unsigned long Tiempo_pasado=0;
```

```
float Setpoint=0;

float Angulo_promedio=0;

float error;

float error_integrador=6000;

float Pendiente;

float error_anterior=0;

float P_I_D;

int acum=10;


// constantes del PID kp, ki y kd

float kp=2.5;//kp=1;//200;//1800;//5000;//100;//5;//1300;

float ki=0.009;//ki=0.002;//2.7095;//2;//800;//5;//0;//8;

float kd=500;//kd=500;//5.875;//0.927375;//0;//300;

float val;

int val_esc;

int pulsador=0;

int pulsador2=0;

int x=0;


Servo esc; // servo clas con nombre esc

Servo esc2; // servo clas con nombre esc

void setup()
```

```
{
pinMode(7, INPUT);
pinMode(6,OUTPUT);
Wire.begin();
Wire.beginTransmission(MPU);
Wire.write(0x6B);
Wire.write(0);
Wire.endTransmission(true);
Serial.begin(115200);

  esc.attach(9); //Se asigna la varable al pin 9
  esc2.attach(10); //Se asigna la varable al pin 10
  Inicializar_motores();
Serial.println("Inicio");

}
void loop()
{
    for(int i=0; i<acum;i++){
  lectura_angulo();
  Angulo_promedio=Angulo_promedio+Angulo;
  }
```

```
Angulo_promedio=Angulo_promedio/acum;
  calcular_PID();
   val= P_I_D;


 if (val <= 1200)
   {
     val = 1200;
   }
 else if (val >= 1700) // se pone limite en 1700 por precaución
   {
    val = 1700;
   }
   Graficar();
esc2.writeMicroseconds(val);
////////////parar motores/////
pulsador=digitalRead(7);
if ((pulsador == HIGH) && (pulsador2==LOW)){
 x=1-x;
 delay(100);
}
pulsador2=pulsador;
 if(x == 1){
```

```
   while(x==1){
   digitalWrite(6, HIGH);
   Serial.println("Apagando...");
   suspender_motores();
  if(x==0){}
  break;
   }
 }
   if(x==0){
   digitalWrite(6,LOW);
   esc.writeMicroseconds(1350);
   loop();
 }
  ////////////////////////////////
}


void lectura_angulo(){
  //Leer los valores del Acelerometro de la IMU
  Wire.beginTransmission(MPU);
  Wire.write(0x3B); //Pedir el registro 0x3B - corresponde al AcX
  Wire.endTransmission(false);
  Wire.requestFrom(MPU,6,true); //A partir del 0x3B, se piden 6 registros
```

```
  AcX=Wire.read()<<8|Wire.read(); //Cada valor ocupa 2 registros
  AcY=Wire.read()<<8|Wire.read();
  AcZ=Wire.read()<<8|Wire.read();

   //Se calculan los angulos Y, X respectivamente.
  Acc[0] = atan((AcY/A_R)/sqrt(pow((AcX/A_R),2) + pow((AcZ/A_R),2)))*RAD_TO_DEG;
  //Leer los valores del Giroscopio
  Wire.beginTransmission(MPU);
  Wire.write(0x43);
  Wire.endTransmission(false);
  Wire.requestFrom(MPU,4,true); //A diferencia del Acelerometro, solo se piden 4 registros
  GyX=Wire.read()<<8|Wire.read();
  GyY=Wire.read()<<8|Wire.read();
  //Calculo del angulo del Giroscopio
  Gy[0] = GyX/G_R;
  //Aplicar el Filtro Complementario
  Angulo = 0.98 *(Angulo+Gy[0]*0.010) + 0.02*Acc[0];
}
void calcular_PID(){
  Temp = millis() - Tiempo_pasado;
  Tiempo_pasado = Tiempo_pasado+Temp;
```

```
    error = (Setpoint - Angulo_promedio); // se calcula el error de acuerdo con el set point.
    error_integrador = error * Temp + error_integrador; // se calcula la parte integral (área bajo la curva)
    Pendiente = (error - error_anterior)/Temp; // se calcula la parte derivativa (la pendiente)
    P_I_D = (error * kp) + (error_integrador * ki) + (Pendiente * kd);
    error_anterior = error;
}


 void Inicializar_motores(){
  esc.writeMicroseconds(1000);
  esc2.writeMicroseconds(1000);
  delay (5000);
  esc.writeMicroseconds(2000);
  esc2.writeMicroseconds(2000);
  delay (1000);
  esc.writeMicroseconds(1350);
 }
 void Graficar(){
  Serial.print(Angulo_promedio);
  Serial.print(',');
  Serial.print(error*kp);
  Serial.print(',');
  Serial.print(error_integrador*ki);
```

```
 Serial.print(',');
 Serial.print(Pendiente*kd);
 Serial.print(',');
 Serial.print(val);
 Serial.println(',');
}
void suspender_motores(){
  esc.writeMicroseconds(1000);
  esc2.writeMicroseconds(1000);
}
```

Anexo D.

Programa en Processing:

```
import processing.serial.*;
Serial myPort;        // puerto serial
float xPos1=1;        // horizontal position of the graph
float xPos2=0;
float xPos3=1;
float xPos4=1;
float xPos5=1;
/////
float yPos1,yPos2,yPos3,yPos4,yPos5;
float lastxPos1,lastxPos2,lastxPos3,lastxPos4,lastxPos5;
float lastyPos1,lastyPos2,lastyPos3,lastyPos4,lastyPos5;
```

```
/////
 int H=1350;       // tamaño horizontal del grafico
 int V=720;         // tamaño vertical del grafico
 void setup () {
 // set the window size:
 size(H,V);
  println(Serial.list());
 myPort = new Serial(this, Serial.list()[0], 115200);
  myPort.bufferUntil('\n');
graficar_fondo();
}

 void draw () {
 // everything happens in the serialEvent()
 }

 void serialEvent (Serial myPort) {
 String inString = myPort.readStringUntil('\n');
 int a=V/36;
 if (inString != null) {
fill(0);
stroke(0);
```

```
rect (1300,10,50,30);

fill(0);

stroke(0);

rect(1040,366,310,30);


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


float inByte1 = map(value[0],-90, 90, 360, height);//angulo, violeta

float inByte2 = map(value[1],-350, 350, 0, 360);//kp,verde

float inByte3 = map(value[2],-28300, 28300, 0, 360);//ki,azul

float inByte4 = map(value[3],-421, 448, 0, 360);//kd,violeta

float inByte5 = map(value[4],1000, 2000, 0, 360);//PID,rojo


/////////Angulo/////

stroke(127,34,255);//violeta

strokeWeight(2);

yPos1= int(height-inByte1);

line(lastxPos1, lastyPos1 , xPos1, yPos1);

lastxPos1=xPos1;

lastyPos1=yPos1;

/////////kp/////////

stroke(0,255,0);//verde
```

```
strokeWeight(2);
yPos2= int(height-inByte2);
line(lastxPos2, lastyPos2 , xPos2, yPos2);
lastxPos2=xPos2;
lastyPos2=yPos2;
//line(xPos, -1*(inByte2-(height)) , xPos, -1*(inByte2+1-(height)));


/////////ki/////////
stroke(0,0,255);// azul
strokeWeight(2);
yPos3= int(height-inByte3);
line(lastxPos3, lastyPos3 , xPos3, yPos3);
lastxPos3=xPos3;
lastyPos3=yPos3;
//line(xPos, -1*(inByte3-(height)) , xPos, -1*(inByte3+1-(height)));


/////////kd/////////
stroke(127,34,255);//violeta
strokeWeight(2);
yPos4= int(height-inByte4);
line(lastxPos4, lastyPos4 , xPos4, yPos4);
lastxPos4=xPos4;
```

```
lastyPos4=yPos4;
//line(xPos, -1*(inByte4-(height)) , xPos, -1*(inByte4+1-(height)));

/////////PID/////////
stroke(255,0,0);//rojo
strokeWeight(2);
yPos5= int(height-inByte5);
line(lastxPos5, lastyPos5 , xPos5, yPos5);
lastxPos5=xPos5;
lastyPos5=yPos5;
//line(xPos, -1*(inByte5-(height)) , xPos, -1*(inByte5+1-(height)));

  ////////////////////////
////////Angulo////////
textSize(12);
fill(255,0,0);
text(value[0],1300,25);
/////////kp/////////
textSize(12);
fill(0,255,0);//verde
text("Kp: ",1040,396);
fill(0,255,0);//verde
```

```
text(value[1],1060,396);

////////ki////////

textSize(12);

fill(0,0,255);//azul

text("Ki: ",1110,396);

fill(0,0,255);//azul

text(value[2],1130,396);

////////kd////////

textSize(12);

fill(127,34,255);//violeta

text("Kd: ",1210,396);

fill(127,34,255);//violeta

text(value[3],1230,396);

////////PD////////

textSize(12);

fill(255,0,0);//rojo

text("PID: ",1280,396);

fill(255,0,0);//rojo

text(value[4],1300,396);



 // at the edge of the screen, go back to the beginning:
```

```
 if (xPos1 >= width) {
 xPos1 = 0;
graficar_fondo();
lastxPos1=xPos1;
lastyPos1=yPos1;

 }
 else {
 // increment the horizontal position:
 xPos1++;
 }

 if (xPos2 >= width) {
 xPos2 = 0;
graficar_fondo();
lastxPos2=xPos2;
lastyPos2=yPos2;

 }
 else {
 // increment the horizontal position:
 xPos2++;
```

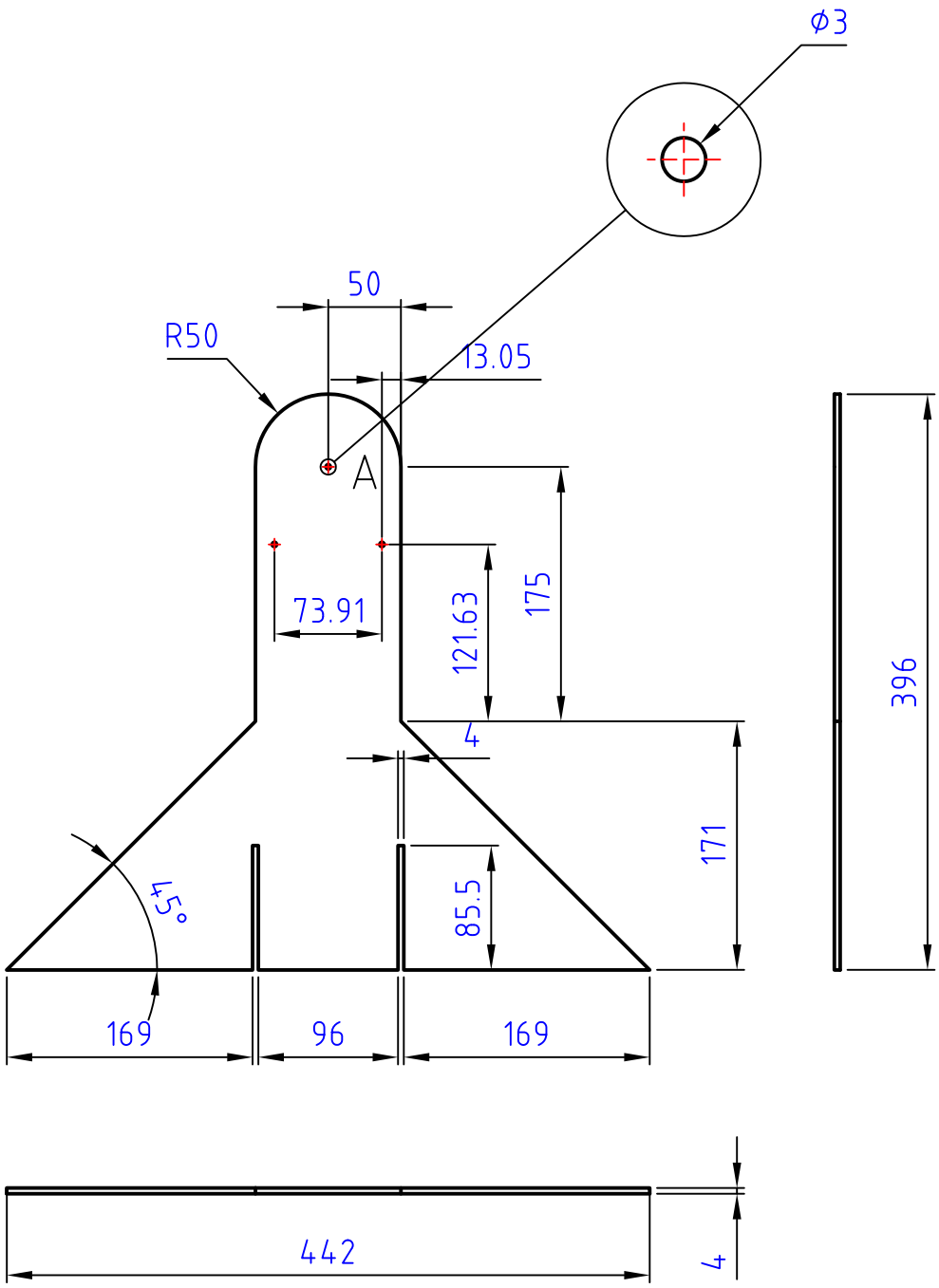
```
 }


 if (xPos3 >= width) {
 xPos3 = 0;
graficar_fondo();
lastxPos3=xPos3;
lastyPos3=yPos3;


 }
 else {
 // increment the horizontal position:
 xPos3++;
 }


  if (xPos4 >= width) {
 xPos4 = 0;
graficar_fondo();
lastxPos4=xPos4;
lastyPos4=yPos4;
 }
 else {
 // increment the horizontal position:
```

```
xPos4++;
}

 if (xPos5 >= width) {
xPos5 = 0;
graficar_fondo();
lastxPos5=xPos5;
lastyPos5=yPos5;

}
else {
// increment the horizontal position:
xPos5++;
}

}
}

void graficar_fondo(){
background(255,255,255);
int a=(V/2)/18;
int b =(V/2)/18;
```

```
 int c=(H/31);


 for (int d = 0; d <= 45; d=d+1)
{
  stroke(0);
  strokeWeight(1);
  line(c*d, 0, c*d, V);
}
for (int i = 0; i <= 18; i = i+1)
{
  stroke(0);
  strokeWeight(1);
  line(0, a*i, H, a*i);
  fill(0);
  text(90-10*i,0,a*i+10);
}
for (int u = 0; u <= 18; u = u+1)
{
  stroke(0);
  strokeWeight(1);
  line (0, (b*u)+360, H ,(b*u)+360);
  fill(0);
```

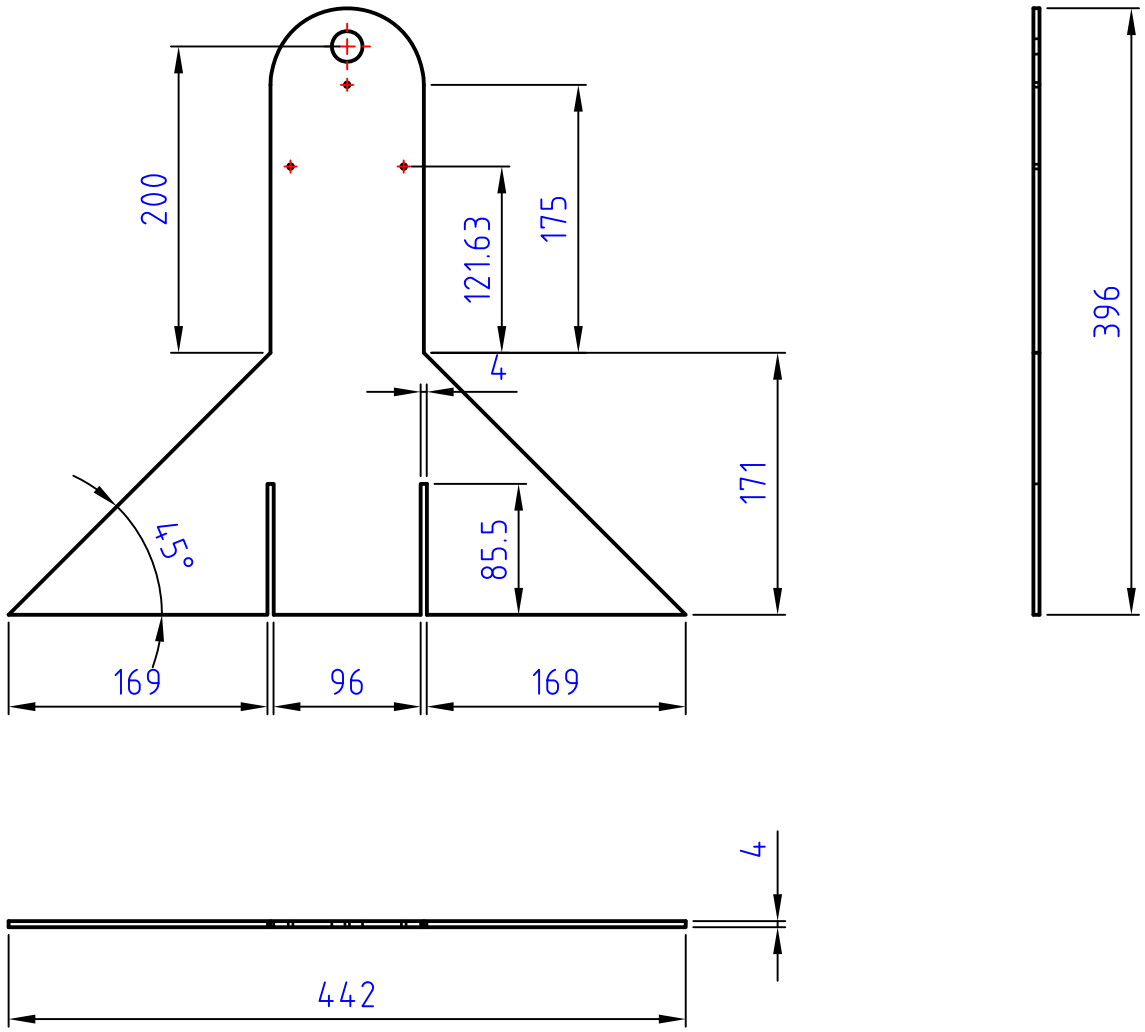
```
  text(2360-((50*u)+360),1310,((b*u)+360)+10);
  text(7515-((297*u)+360),1,((b*u)+360)+10);
  text("RPM",41,((b*u)+360)+10);
}


  fill(0);
  stroke(0);
  rect (0,0,H,10);
    fill(0);
  stroke(0);
  rect (0,356,H,10);
//////linea de setpoint////
  stroke (255,0,0);
  line(0, 180,H, 180);
textSize(12);
fill(173,255,47);
text("Angulo", 675, 9);
fill(255,0,0);
text("Setpoint",0,180);
fill(173,255,47);
text("Kp, Ki, Kd, PID",675,365);
 }
```

A (2:1)

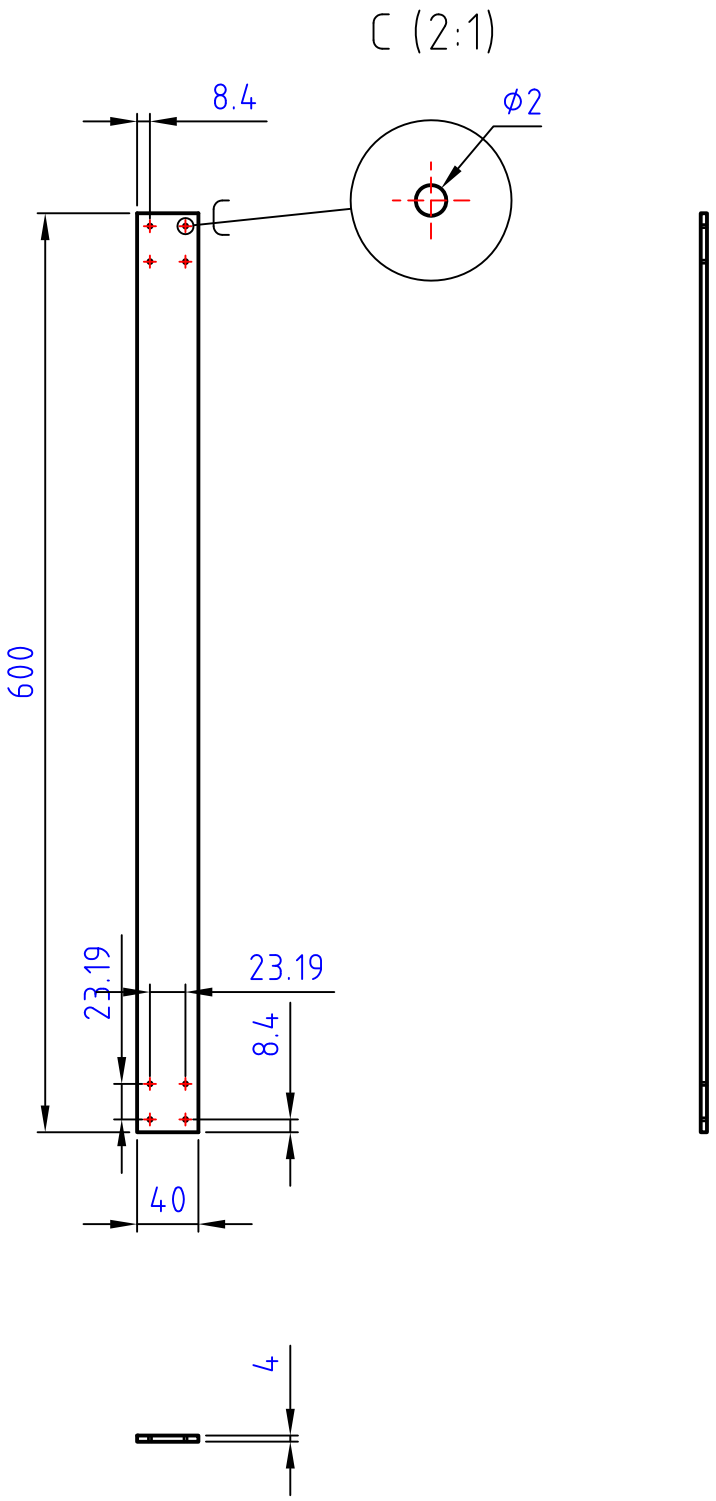


Trat. Termico	Ninguno	UISEK	CARRERA DE INGENIERIA MECANICA		
Recubrimiento	Ninguno				
MATERIAL: Acrilico	Tol. Gral. +-0.01	Escala: 1:5	DIB:	Juan Pablo Coronel Aguilar	
			DIS:	Juan Pablo Coronel Aguilar	
			REV:	Ing. Gustavo Moreno	
Soporte frontal	A-001	Fecha 04/05/2017			

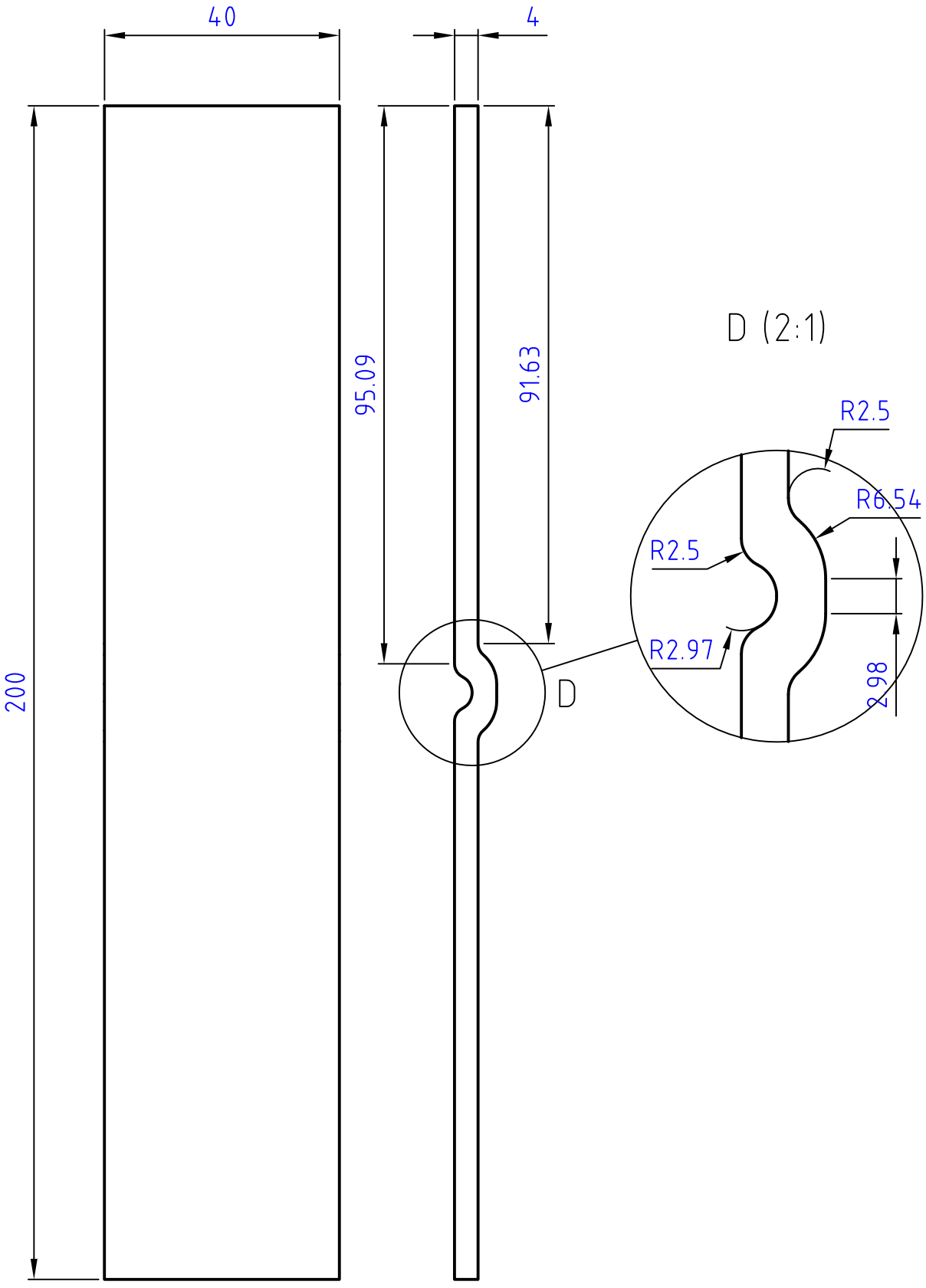
B (2:1)



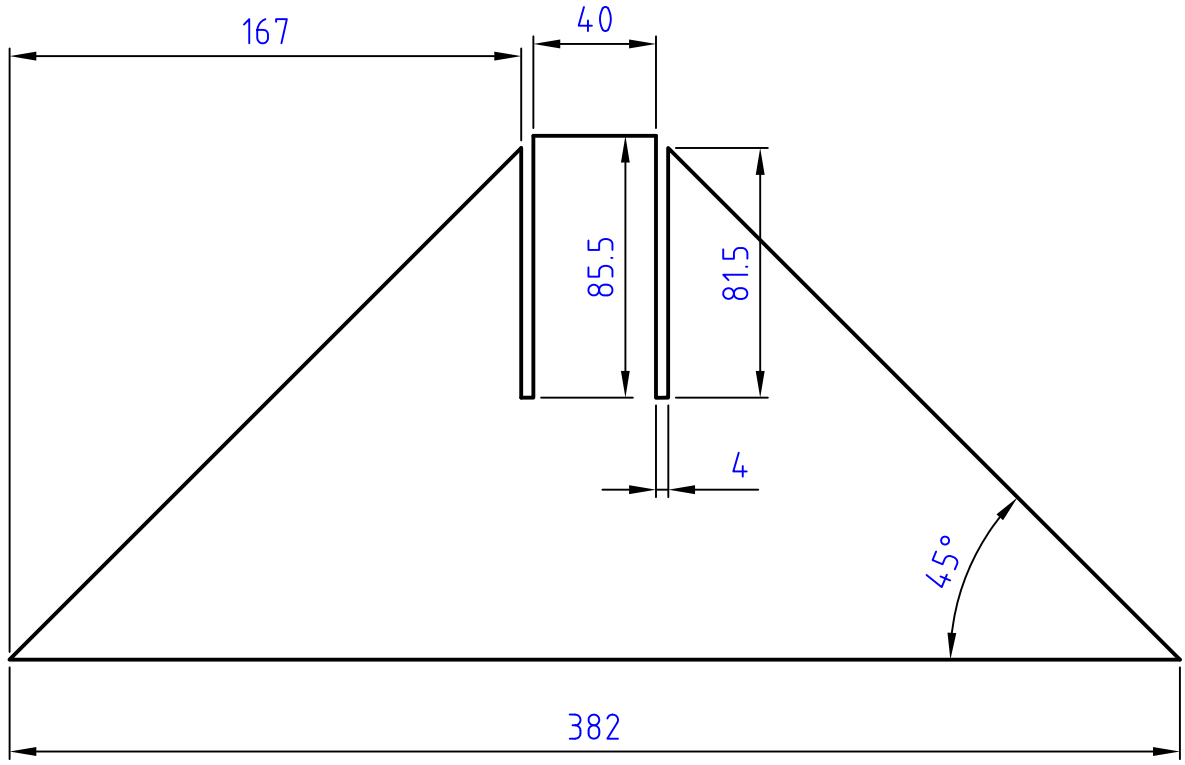
Trat. Termico	Ninguno		UISEK	CARRERA DE INGENIERIA MECANICA	
Recubrimiento	Ninguno				
MATERIAL: Acrílico		Tol. Gral. +-0.01	Escala: 1:5	DIB:	Juan Pablo Coronel Aguilar
				DIS:	Juan Pablo Coronel Aguilar
				REV:	Ing. Gustavo Moreno
Soporte Posterior			A-002	Fecha 04/05/2017	

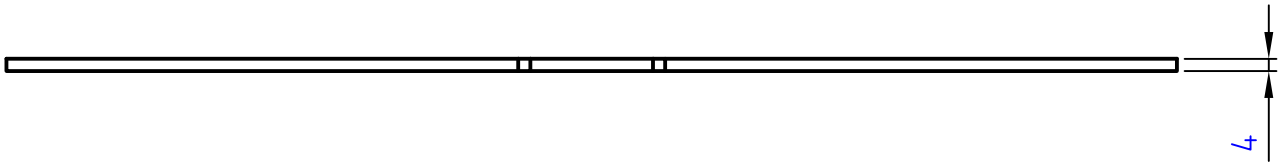


Trat. Termico	Ninguno		UISEK	CARRERA DE INGENIERIA MECANICA	
Recubrimiento	Ninguno				
MATERIAL: Acrílico		Tol. Gral. +-0.01	Escala: 1:5	DIB:	Juan Pablo Coronel Aguilar
				DIS:	Juan Pablo Coronel Aguilar
				REV:	Ing. Gustavo Moreno
Eje Principal			A-003	Fecha 04/05/2017	

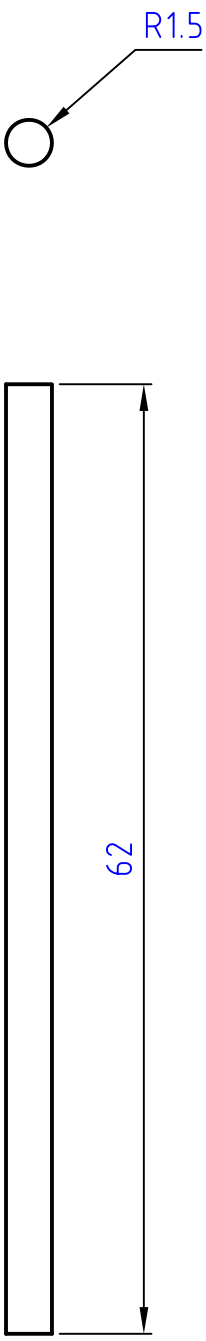


Trat. Termico	Ninguno		UISEK	CARRERA DE INGENIERIA MECANICA	
Recubrimiento	Ninguno				
MATERIAL: Acrílico		Tol. Gral. +-0.01	Escala: 1:1	DIB:	Juan Pablo Coronel Aguilar
				DIS:	Juan Pablo Coronel Aguilar
				REV:	Ing. Gustavo Moreno
Soporte Inferior del Eje Principal			A-004	Fecha	04/05/2017

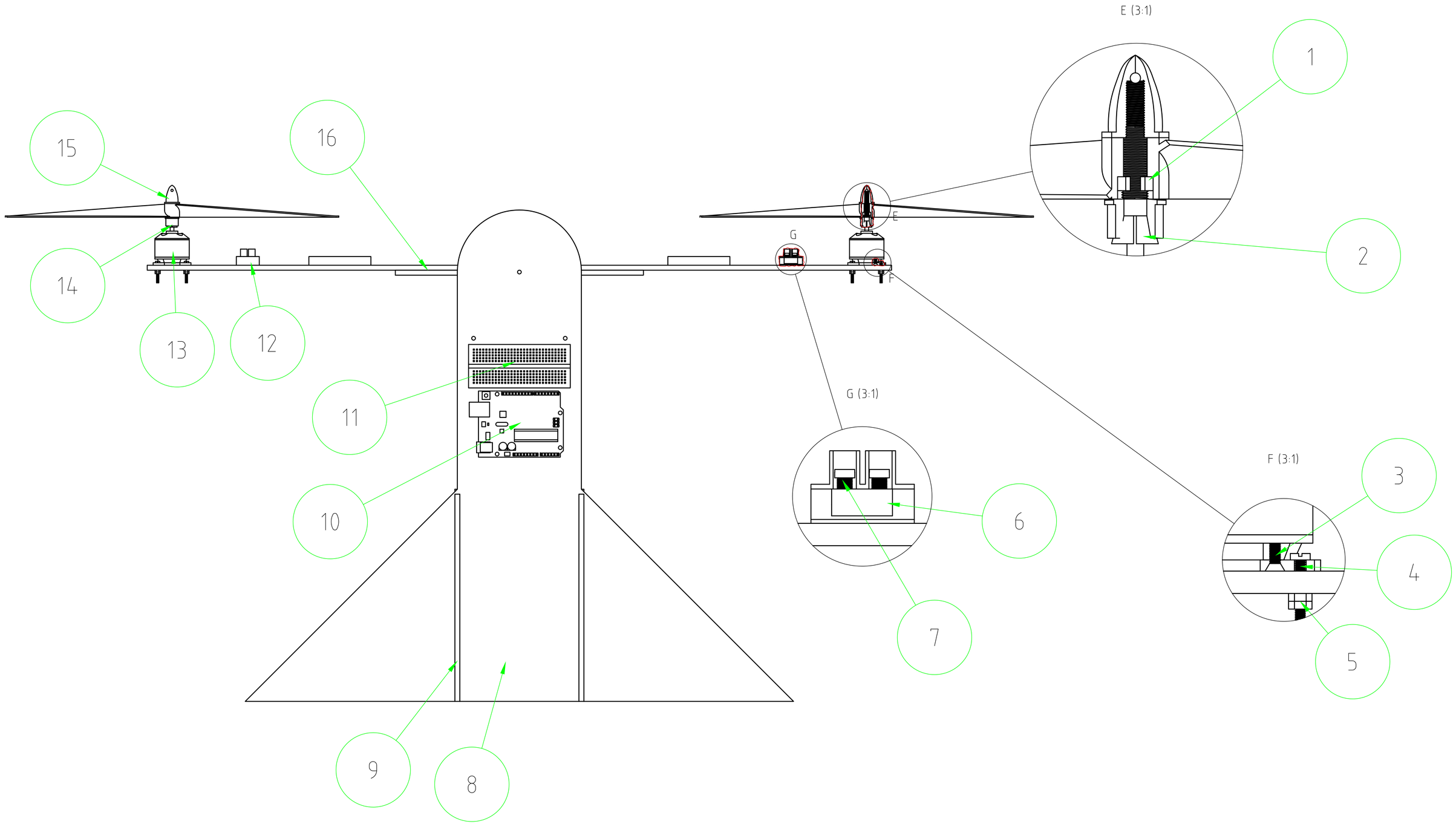


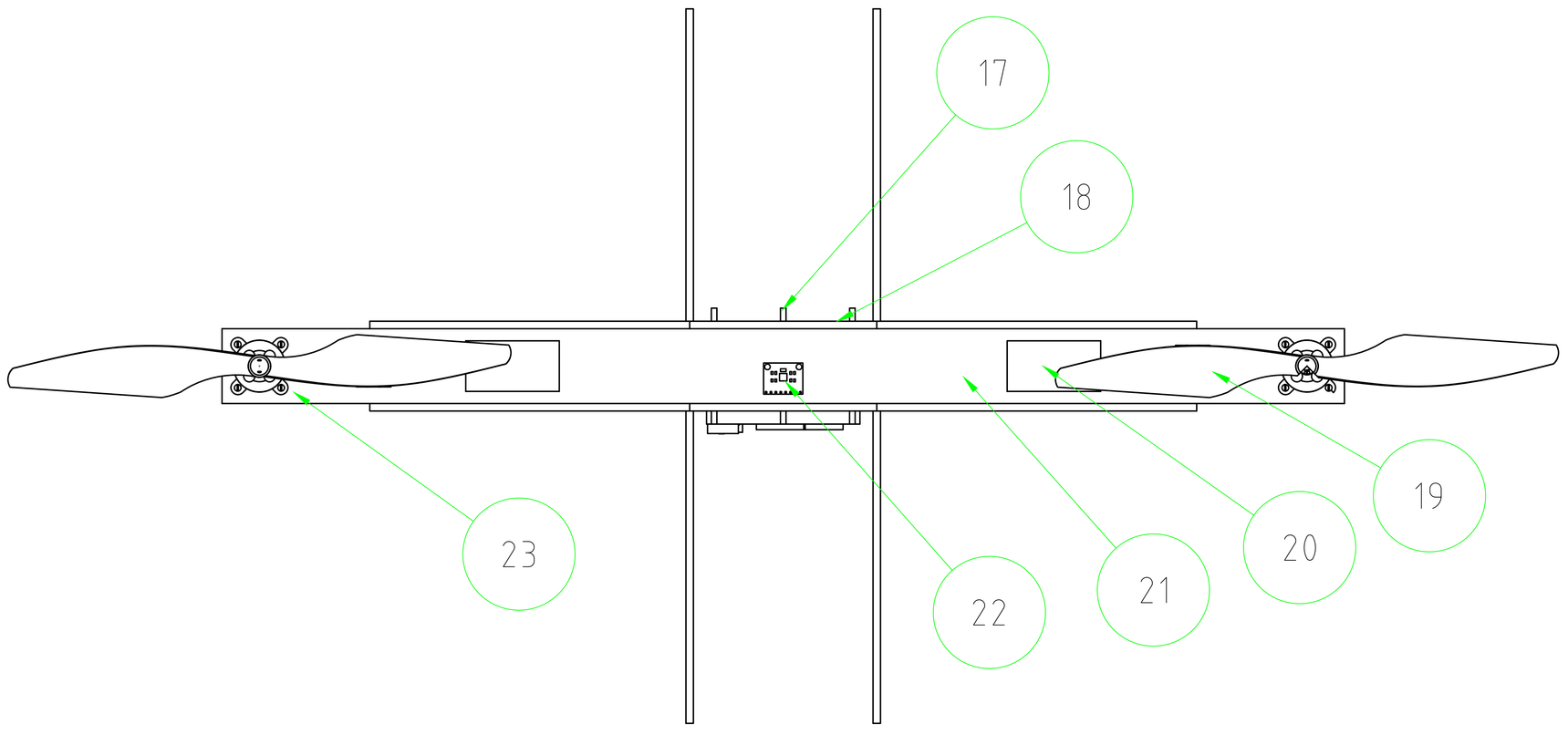


Trat. Termico	Ninguno		UISEK	CARRERA DE INGENIERIA MECANICA	
Recubrimiento	Ninguno				
MATERIAL: Acrílico		Tol. Gral. +-0.01	Escala: 1:2.5	DIB:	Juan Pablo Coronel Aguilar
				DIS:	Juan Pablo Coronel Aguilar
				REV:	Ing. Gustavo Moreno
Soporte Lateral			A-005	Fecha 04/05/2017	



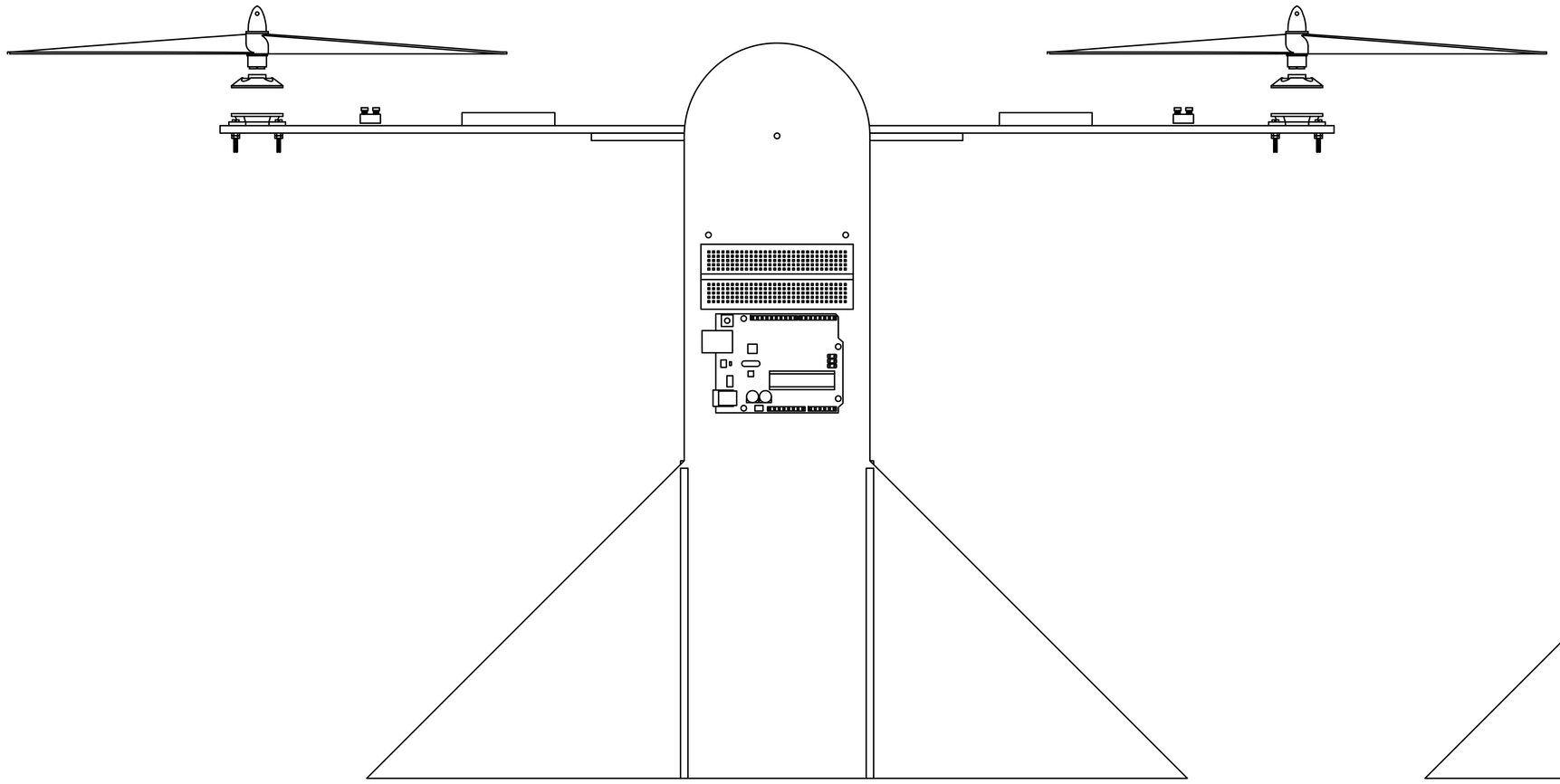
Trat. Termico	Ninguno		UISEK	CARRERA DE INGENIERIA MECANICA	
Recubrimiento	Ninguno				
MATERIAL:	Acero Inoxidable 304	Tol. Gral. +-0.01	Escala: 2:1	DIB:	Juan Pablo Coronel Aguilar
				DIS:	Juan Pablo Coronel Aguilar
				REV:	Ing. Gustavo Moreno
Eje			A-006	Fecha	04/05/2017

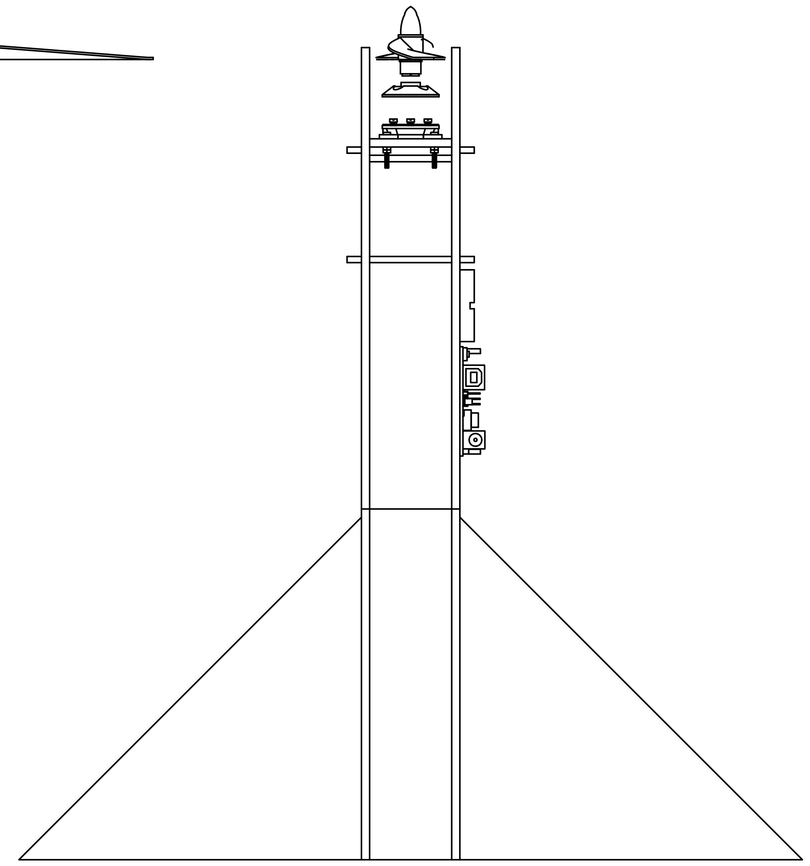


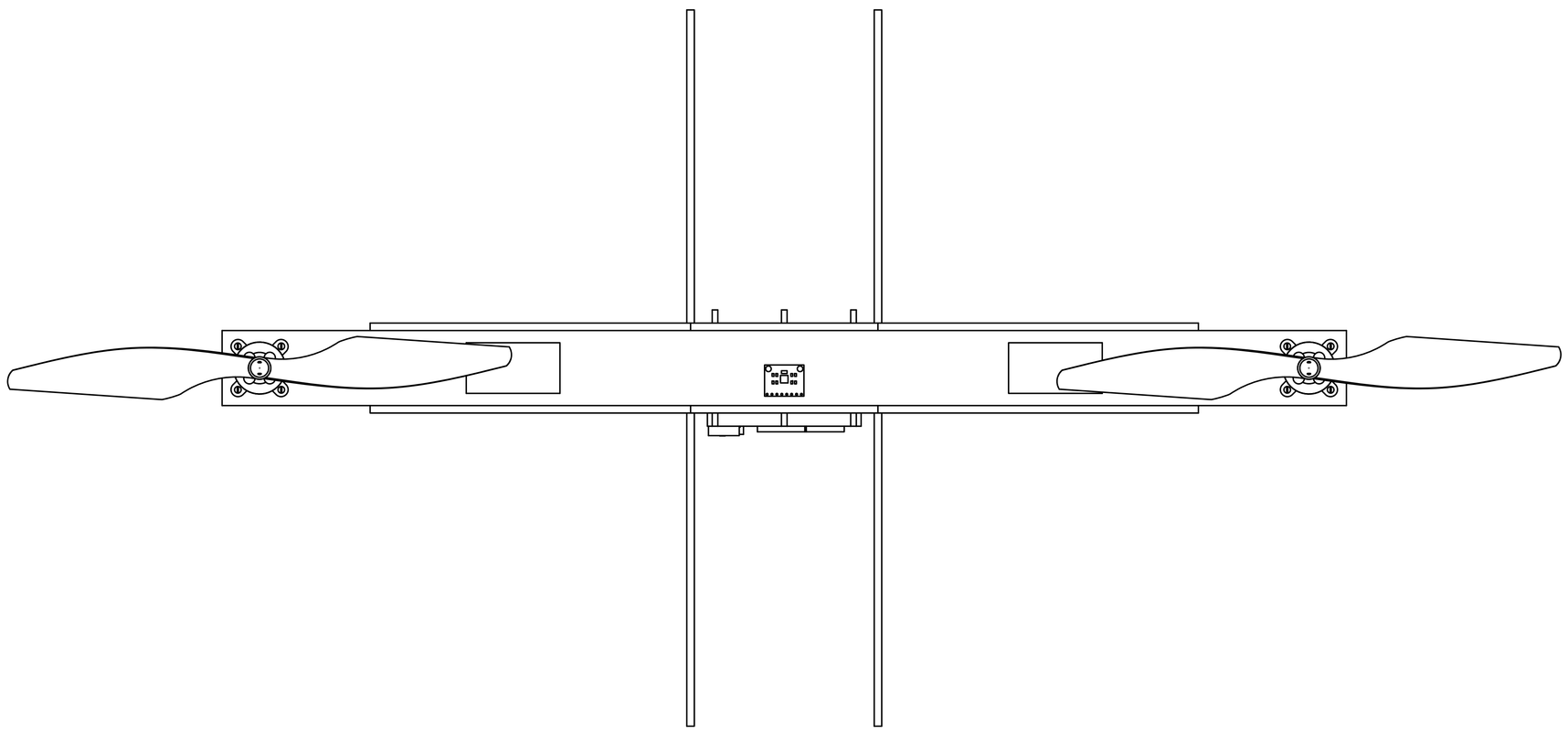


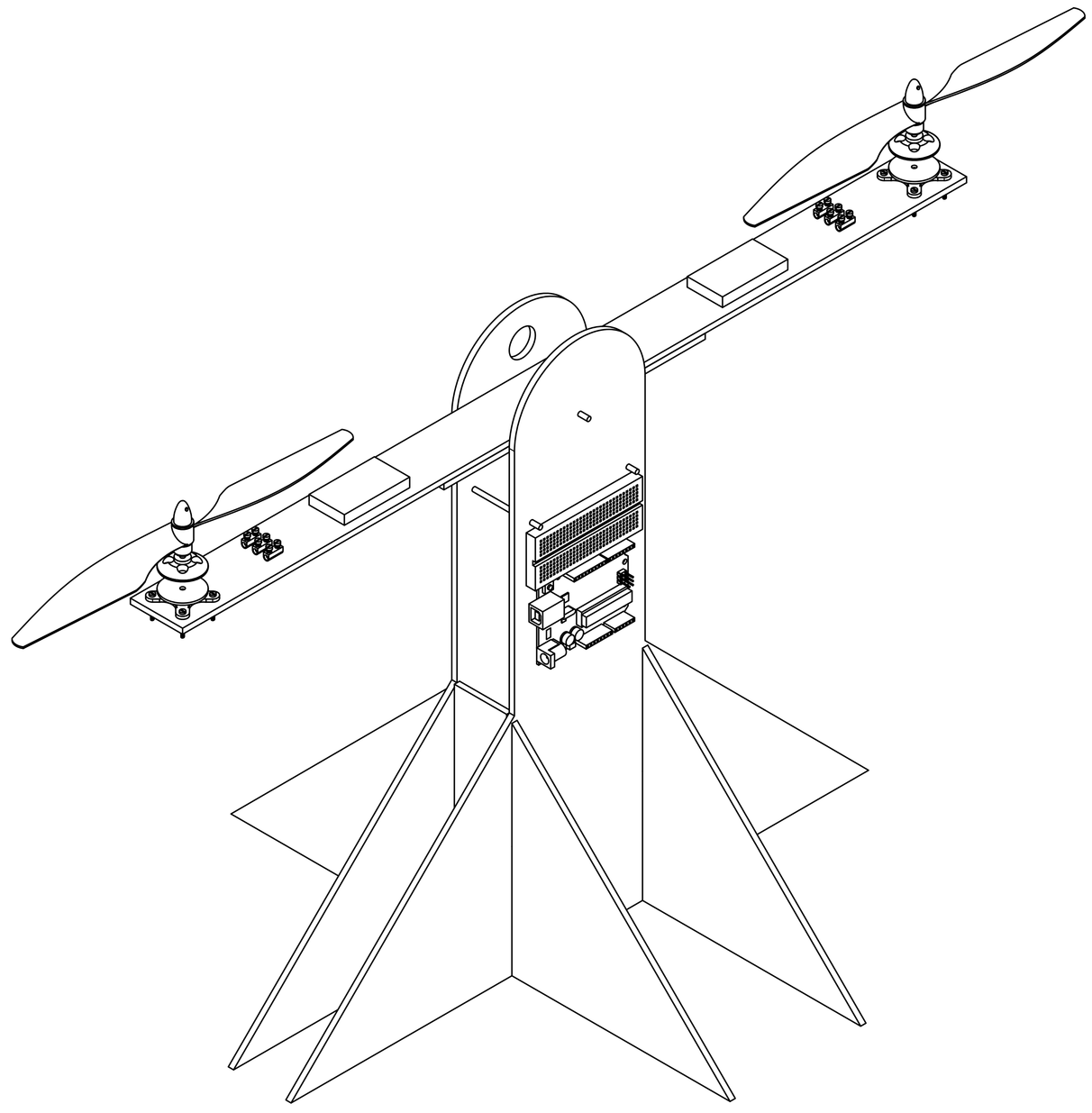
N#	Cantidad	Nombre	Ubicación	Plano	Estandar	Material	Unidades
23	2	Bancada del motor	J 4.15		ALLOY 2024	Aluminio	mm
22	1	MPU6050	J 6.13		N/A	N/A	mm
21	1	Eje principal	J 7.5	A-003	ASTM D638	Acrílico	mm
20	2	Electronic Speed Control	J 7.9		N/A	N/A	mm
19	2	Hélice	J 8.75		ASTM D790	Plástico	mm
18	1	Soporte Posterior	J 6.9	A-002	ASTM D638	Acrílico	mm
17	3	Eje	J 6.15	A-006	ASTM 304	Acero Inox 304	mm
16	1	Sujetador infeior del eje primcipal	C 5.95	A-004	ASTM D638	Acrílico	mm
15	2	Sujetador superior de la hélice	B 4.05		ALLOY 2024	Aluminio	mm
14	2	Sujetador inferior de la hélice	B 4.05		ALLOY 2024	Aluminio	mm
13	2	Motor	C 4.05		N/A	Aluminio	mm
12	2	Terminal Eléctrico	C 4.65		N/A	Plástico	mm
11	1	Proto barra	D 6.65		N/A	N/A	mm
10	1	Arduino	D 6.65		N/A	N/A	mm
9	2	Soporte lateral	F 6.15	A-005	ASTM D638	Acrílico	mm
8	1	Soporte frontal	F 6.5	A-001	ASTM D638	Acrílico	mm
7	12	Tornillos de los terminales eléctricos	D 9.05		DIN 8.8	Acero Galvanisado	mm
6	6	Cobre interno del terminal	E 9.35		ASTM 110	Cobre	mm
5	16	Tuercas de los pernos de la bancada del motor	E 12.4		DIN 8.8	Acero Galvanisado	mm
4	8	Tornillos de la bancada del motor	E 12.45		DIN 84	Acero Galvanisado	mm
3	8	Tornillos de la base del motor	E 12.2		DIN 8.8	Acero Galvanisado	mm
2	2	Sujetador del eje del motor	C 11.3		ALLOY 2024	Aluminio	mm
1	2	Bocín	B 11.3		ASTM D790	Plástico	mm

UISEK	CARRERA DE INGENIERÍA MECÁNICA	DIB.: Juan Pablo Coronel Aguilar DIS.: Juan Pablo Coronel Aguilar REV.: Ing. Gustavo Moreno	Escala: 1:2.5
Nombre	A-001		Fecha: 18/08/2016









UISEK	CARRERA DE INGENIERÍA MECÁNICA	DIB:	Juan Pabla Coronel Aguilar	Escala: 1:15
		DIS:	Juan Pablo Coronel Aguilar	
		REV:	Ing. Gustavo Moreno	
Eje Auto Estabilizador		A-002		Fecha: 04/05/2011