UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ETSI Aeroespacial y Diseño Industrial

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Aeroespacial y Diseño Industrial

Diseño e implementación de un prototipo de robot monociclo con rueda de reacción

Trabajo Fin de Máster

Máster Universitario en Ingeniería Mecatrónica

AUTOR/A: Martinez Padilla, Oscar

Tutor/a: Casanova Calvo, Vicente Fermín

CURSO ACADÉMICO: 2023/2024

ABSTRACT

In the world today, process automation is used in different types of industries, which is why it is important to be able to implement projects that improve existing processes. In practically any industrial project, a series of components are used, such as: actuators and sensors. In this project, the complete design process of a robot will be carried out from its mechanical design to its actual implementation in order to apply the knowledge learned in the Mechatronics Engineering master's degree.

This TFM was based on making a unicycle robot that uses a reaction wheel to maintain balance. The robot will have two different motors that must be controlled so that it can move correctly and stay balanced. The first motor is going to control the wheel of the unicycle and the second motor controls the reaction wheel that controls the balance of the robot. The robot must be able to be read the position from the motor wheel in its base by reading the encoder in order to adjust the speed of the second motor so that the robot can remain standing adequately despite suffering small disturbances.

First, the design of the robot was carried out in 3D with the help of Solidworks CAD software to design its parts and then proceed to print them on the university's 3D printer. In addition to this, simulations of the designed model were carried out using the Matlab tool, Simscape to check its functionality. With the help of this tool, simulations of the robot system will be carried out and the closed-loop controller that will be used by the microprocessor to move the robot and maintain balance before different disturbances.

After having the simulations, the robot was physically made with the 3D printed parts and the purchased electronic components. Finally, the programming and construction of the robot was carried out so that it is able to move forward and reverse and be able to maintain balance with the help of the implemented reaction wheel.

ÍNDICE DE CONTENIDOS

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE TABLAS

ÍNDICE DE ANEXOS

LISTA DE SIGLAS

CAD: Computer Asisted Design (Diseño Asistido por Computadora)

PID: Proporcional Integral Derivativo

FOC: Field Oriented Control

TFM: Trabajo Final Master

LQR: Linear Quadratic Regulador (Regulador Lineal cuadrático)

IMUS: Inertial Measurement Units (Unidades de medición inerciales

CC: Corriente Continua

CA: Corriente Alterna

I. INTRODUCCIÓN

En el mundo actual la automatización de procesos se utiliza en distintas industrias, es por este motivo que es importante poder implementar proyectos que mejoren procesos existentes o que los mejoren. En prácticamente cualquier proyecto industrial se utilizan una serie de componentes tales como pueden ser: actuadores y sensores. En este proyecto se realizara el proceso completo de diseño de un robot desde sus diseño mecánico hasta su implementación real para poder aplicar los conocimientos aprendidos en el master de Ingeniería Mecatrónica.

Este TFM se baso en hacer un robot uniciclo que utiliza una rueda de reacción para poder mantener el equilibrio. El robot tendrá dos motores diferentes que se deberán controlar para que pueda moverse correctamente y mantenerse en equilibrio. El primer motor va a controlar la rueda del uniciclo y el segundo motor controla la rueda de reacción que controla el equilibrio del robot. Se deberá poder leer la posición del robot de la rueda del motor por medio de su respectivo encoder para poder ajustar la velocidad del segundo motor para que el robot consiga mantenerse de pie de manera adecuada a pesar de sufrir pequeñas perturbaciones.

En primer lugar se realizaron el diseño del robot en 3D con ayuda del software CAD Solidworks para realizar el diseño de sus partes para luego proceder a imprimirlas en las impresora 3D de la universidad. Además de esto se realizaron las simulaciones del modelo diseñado utilizando la herramienta de Matlab, Simscape para comprobar su funcionalidad. Con ayuda de esta herramienta se van a realizar la simulaciones del sistema del robot y además se va a diseñar el controlador en bucle cerrado que va a utilizar el microprocesador para mover el robot y mantener el balance antes distintas perturbaciones.

Luego de tener las simulaciones se realizo el robot de manera física con las partes impresas en 3D y los componentes electrónicos comprados. Finalmente, se realizo la programación y construcción del robot para que sea capaz de moverse para adelante y de reversa y poder mantener el equilibrio con ayuda la rueda de reacción implementada.

El presente informe consta de las siguientes secciones:

- **Objetivos:** se enumeran los objetivos de la realización de este proyecto.
- **Antecedentes**: se describen diferentes proyectos parecidos realizados en el pasado.
- **Marco Teórico**: se detallan y se presenta información teórica acerca del control automático y de los componentes que se utilizaran para el diseño del robot
- **Metodología**: se describe la metodología que se siguió para el diseño y posterior construcción del prototipo del robot uniciclo.
- **Desarrollo del trabajo y Resultados:** se describen el trabajo y los resultados obtenidos en el desarrollo del prototipo del robot uniciclo. Se describe el diseño 3D en solidworks, la simulación del modelo físico junto a su control con Matlab Simscape y finalmente la implementación real.
- **Pliego de Condiciones**: se describen las condiciones de ejecución del proyecto.
- **Presupuesto**: se brinda el costo aproximado del prototipo real del robot uniciclo
- **Conclusiones**: en esta sección se brindan las conclusiones alcanzadas al realizar este proyecto.

II. OBJETIVOS

2.1 OBJETIVO GENERAL

Realizar el diseño de un robot uniciclo totalmente inalámbrico que sea capaz de equilibrarse por si mismo con ayuda de una rueda de reacción ante distintas perturbaciones.

2.2 OBJETIVOS ESPECÍFICOS

- Analizar los diseños previos de dispositivos con rueda de reacción para adquirir conocimientos para la realización de este trabajo.
- Realizar los diseños en 3D con ayuda del software Solidworks para la implementación real del robot uniciclo con rueda de reacción, el diseño deberá ser ligero y funcional de modo que se puedan montar los componentes eléctricos y electrónicos para la realización robot.
- Realizar los planos para la construcción robot uniciclo con rueda de reacción.
- Realizar la simulación del robot uniciclo en el software Matlab utilizando la herramienta Simscape para comprobar que el robot uniciclo diseñado es capaz de mantenerse de pie correctamente y diseñar el controlador para el microcontrolador que controlara el robot.
- Investigar los componentes eléctricos y electrónicos para realización del robot uniciclo, se deberán usar materiales accesibles y que sean los mas adecuados para la construcción del robot.
- Realizar la fabricación del prototipo del robot uniciclo con rueda de reacción con el diseño 3D elaborado y con los componentes comprados.
- Realizar la programación para el microcontrolador encargado de mover los motores y controlar la posición y velocidad con ayuda de los encoders.

III. ANTECEDENTES

3.1 PRECEDENTES DEL PROBLEMA

Una vez que fuese escogido el modelo de robot a implementar, lo primero que se hizo fue analizar distintos modelos robot elaborados previamente. El robot uniciclo es un robot muy común en problemas de control automático. Se sigue el funcionamiento de un péndulo invertido.

3.1.1 ROBOT UNICICLO DE LEE JAE EN COREA DEL SUR

El primer dispositivo por analizar fue hecho por un Corea del Sur, ellos elaboraron el control de un robot uniciclo diseñado por ellos mismos con un Controlador Difuso (Lee Jae Oh, 2012).

Ilustración 1-Robot Uniciclo de una rueda

El diseño que ellos diseñaron se basaba en el modelo de un péndulo invertido. En este caso particular se tiene una rueda en la base del péndulo invertido. Con este modelo se pueden controlar dos ejes distintos del robot, en este caso el eje de inclinación (pitch) que lo controla la rueda de reacción y el eje de balanceo(roll) que lo controla la rueda del uniciclo. Ellos elaboraron modelo simplificado donde desacoplaron ambos ejes para diseñar el controlador. Asumieron el desacoplamiento como una perturbación y esto les facilito mucho los cálculos.

Utilizaron dos motores de corriente continua para controlar el robot, además de esto utilizaron un acelerómetro de dos ejes y un giroscopio de tres ejes. De procesador utilizaron un ARM Cortex M3 y lo programaron utilizando C. Obtuvieron los resultados de los ángulos de la balanceo e inclinación del robot y lograron mantener en equilibrio el robot uniciclo de manera satisfactoria.

3.1.2 ROBOT UNICICLO WHEELBOT

El robot WheelBot es un robot fabricado por la universidad de Aachen y el instituto de Max Planck de Sistemas Inteligentes (Atwell, 2022). Este robot utiliza una rueda de reacción para poder generar el par necesario para mantener el robot en equilibrio ante perturbaciones.

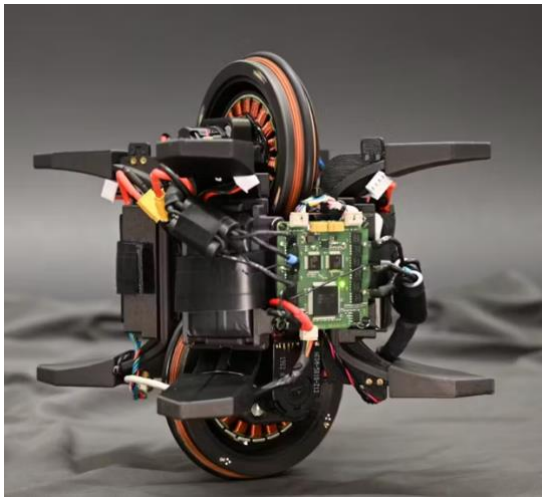


Ilustración 2-Robot WheelBot

Este robot en comparación a otros robot utilizados tiene la habilidad de mantenerse en posición desde cualquier posición inicial y además es capaz de recuperarse ante un caída y levantarse por cuenta propia desde una posición totalmente horizontal. Además de esto es capaz de saltar desde cualquier posición inicial a la posición en equilibrio por cuenta propia. Este robot se elaboro con motores brushless y utiliza un encoder en la rueda. También utiliza cuatro IMUS (Unidad de medición inercial) para conocer su posición en todo momento. Para el control de los motores utilizaron un procesador ARMS M2 con un chip Atmega de 16MHZ junto a los controladores udriver v2. Para el diseño del controlador utilizador Regulador Lineal Cuadrático(LQR).

IV. MARCO TEÓRICO

4.1 PRINCIPIO FÍSICO DEL PÉNDULO INVERTIDO

El péndulo invertido es un problema clásico el mundo del control con lineal. El péndulo invertido es básicamente un modelo que consiste en una varilla de una longitud L que puede estar anclada o montada en un carro donde se le proporciona una fuerza, mediante la fuerza aplicada permite que el carro de varilla pueda llegar a su posición de equilibrio como si se tratara de un péndulo invertido. Una vez que la varilla alcanza su posición debe ser capaz de mantenerse en esa posición por medio del control en bucle cerrado (Gordillo, 2010).

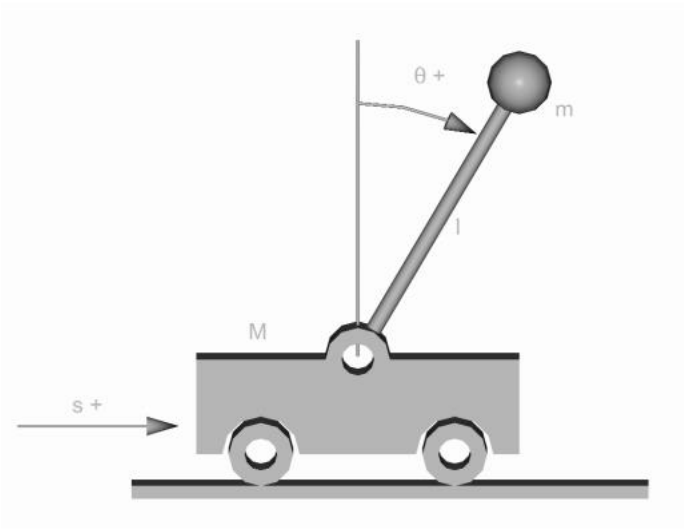


Ilustración 3-Pendulo Invertido con coche

Hay dos etapas básicas en cualquier problema de péndulo invertido:

- Swing Up: Esta es la parte desde donde se le aplica una fuerza al péndulo para que alcance la posición invertida deseada. La fuerza debe de ser la necesaria para que el péndulo únicamente llegue a la posición de equilibrio de los contrario si se le aplica una fuerza demasiado grande se dificulta demasiado el control del péndulo en la posición de equilibrio.
- Estabilización: En esta etapa se llega cuando luego del swing up se alcanza la posición del péndulo invertido y se debe ser capaz de mantener el péndulo en la posición de equilibrio mediante el control el bucle cerrado del sistema.

El péndulo invertido es un problema debido a que es un sistema inestable por naturaleza ya que cualquier pequeña perturbación como puede ser el aire o cualquier leve movimiento ya provoca su caída debido a esto el diseño del controlador debe ser capaz de actuar con la suficiente velocidad y aplicar la acción de control mas adecuada para que sea capaz de mantenerse de pie. Una de la estrategias mas comunes para el control de sistemas de péndulos invertidos es la de control PID en bucle cerrado.

4.2 RUEDA DE REACCIÓN

Una rueda de reacción es un tipo de actuador que se utiliza en satélites. Una rueda de reacción es capaz de generar torque sin propulsores. Utilizan el principio de conservación de momento angular, una rueda de reacción es un masa redonda que al girar genera una inercia que permite el giro de los satélites. Las ruedas de reacción incluyen motores eléctricos que son los encargados de controlar la velocidad en que giran las ruedas de reacción. Las ruedas de reacción son muy útiles para poder reducir el consumo de combustible en los satélites y son muy precisas para realizar pequeños movimientos y giros de los satélites (MSCI, 2013).

Ilustración 4-Rueda de Reacción de un Satélite

4.3 DIBUJO ASISTIDO POR COMPUTADORA (CAD)

El diseño asistido por computadora es el uso de computadoras para así poder desempeñar tareas tales como el diseño de algún tipo de elemento mediante el dibujo técnico. Una herramienta asistida por computadora (CAD) es un software que ayuda a automatizar el proceso de diseño de algún modelo que se quiera realizar por el usuario. El diseño asistido por computadora ha facilitado el proceso de diseño a ingenieros, arquitectos y a otros profesionales en sus labores (Ecured, 2017).

El diseño asistido por computadora facilita la fabricación, el modelado y el diseño de productos. Mediante este proceso es mucho mas rápido y precisa la fabricación de modelos en comparación a cuando el proceso se realizaba únicamente por el hombre. En el diseño asistido por computadora guarda todas las características tales como: las dimensiones, los contornos y la apariencia del modelo que se esta diseñando en la computadora para luego permitir mejorar los diseños de una manera sencilla. Mediante el diseño asistido por computadora se eliminan los errores humanos en el dibujo y permite realizar simulaciones con los modelos que se estén diseñando. El diseño asistido por computadora puede ser realizad en dos o en tres dimensiones (La Revista Informatica, 2019).

Ilustración 5- Lista de Software CAD

Fuente: (Lachaize, 2017)

4.3.1 TIPOS DE CAD

Existen distintos tipos de diseño asistido por computadora basado en las necesidades que tiene el usuario. Los tipos mas comunes son los siguientes:

- CAD Analítico

  Este tipo de diseño utiliza procesos analíticos para definir los limites o acciones. En este tipo de diseño asistido por computadora el dibujo permanece en la memoria de la computadora mediante un sistema de puntos – coordenadas y de vectores de dirección (Ecured, 2017).
- CAD Paramétrico

  Este tipo de diseño asistido por computadora utiliza parámetros para definir sus limites o acciones. En un software paramétrico se guarda la información como si fuera un objeto en la memoria de la computadora (Ecured, 2017).

4.3.2 VENTAJAS DEL CAD

El diseño asistido por computadora ayuda a mejorar la fabricación, desarrollo y diseño de nuevos productos con la ayuda de la computadora. De igual manera es mucho mas rápido y mas preciso el modelado en este tipo de software (Ecured, 2017).

Algunas de las principales ventajas de los softwares CAD son las siguientes:

- Existen librerías comunes de modelos previos.
- Permite realizar modificaciones mas rápidas y con mas facilidad.
- Ya no existe diferencia entre un plano original y una copia.
- Los planos son uniformes y las copias no presentan cambios.
- Aumenta la calidad de los planos.
- Se pueden realizar los diseños con un menor tiempo.
- Se pueden realizar operaciones repetitivas con mucha mas facilidad y en menor tiempo.
- Se pueden obtener modelos tres dimensiones para mejorar la visualización del modelo.
- Se pueden hacer simulaciones o animaciones.
- Se pueden importar los dibujos a software CAE Y CAM.

- Facilita el almacenamiento de los planos.

(Ecured, 2017).

4.3.3 Etapas del Diseño Asistido por Computadora

El proceso CAD consiste en las siguientes etapas:

- **El modelado Geométrico**

  En esta etapa el diseñador describe de forma matemática o analítica a un objeto, se crea un modelo geométrico del objeto mediante línea, superficies, cuerpos o dimensiones. En esta etapa se crea el modelo exacto y completo ya sea en dos o en tres dimensiones (Ecured, 2017).

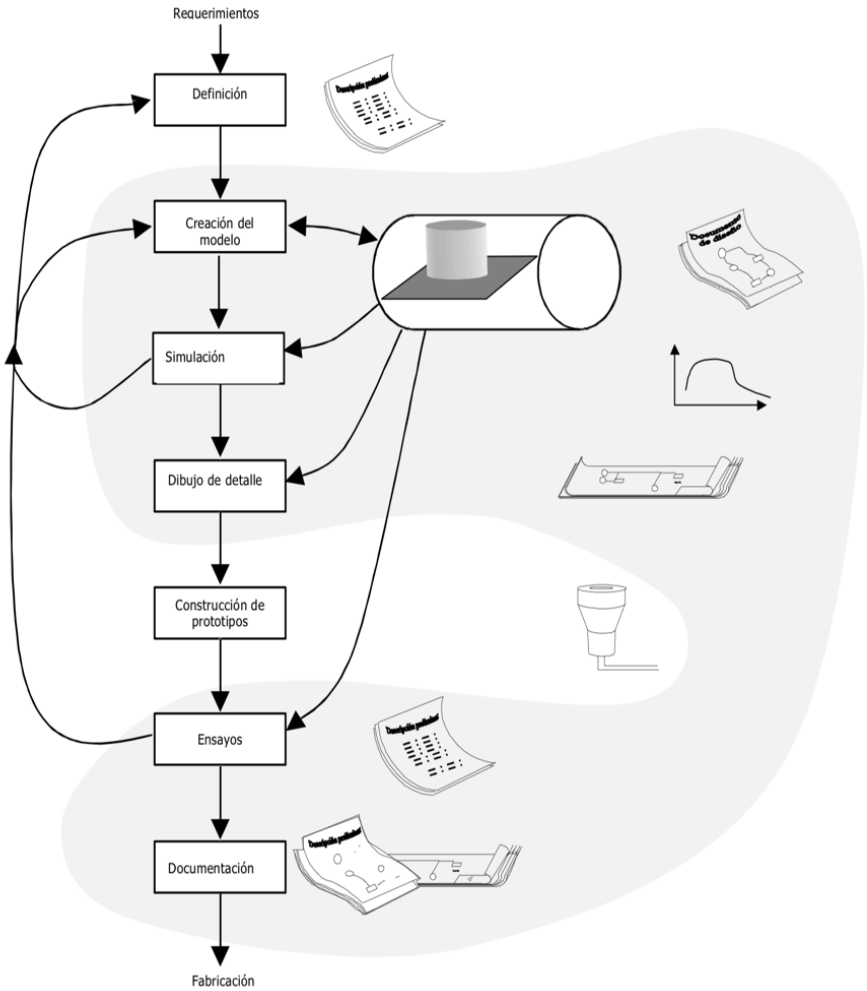


Ilustración 6-Proceso de diseño usando CAD

Fuente: (Torres, 2019)

- **Análisis del Diseño**

  Después que se obtienen las propiedades geométricas del objeto que se esta diseñando, luego se procede a hacer análisis del propiedades físicas del objeto como (deformaciones, esfuerzos o flexiones) (Ecured, 2017).

- **Revisión del diseño**

  En esta etapa del proceso de diseño se revisa que el modelo no tenga ningún tipo de interferencia entre alguno de los componentes. En esta etapa se realiza un ensamblaje para verificar que no haya problemas con el diseño propuesto y para observar el uso del diseño (Ecured, 2017).

- **Documentación y dibujo**

  En esta etapa del proceso de diseño se realizan los planos del modelo que fue diseñado. En el plano se incluyen las cotas correspondientes, diferentes vistas de la pieza y diferentes escalas y observaciones de la pieza (Ecured, 2017).

4.4 COMPONENTES ELÉCTRICOS Y ELECTRÓNICOS NECESARIOS PARA LA ELABORACIÓN DEL ROBOT

4.4.1 MOTORES ELÉCTRICOS

Un motor eléctrico es un elemento que permite transformar la energía eléctrica en energía mecánica para ser utilizado en distintas aplicaciones. Los motores se pueden dividir en distintas categorías, de manera general se dividen en dos grandes ramas basada en el tipo de electricidad que utilizan: motores de corriente continua y motores de corriente alterna. A continuación se brinda una pequeña división de los distintos tipos de motores eléctricos.

- Motores de Corriente Continua (DC)
  - Motor de corriente continua con escobillas
  - Motor de corriente continua sin escobillas (Brushless)
  - Motor de corriente continua de imanes permanentes
  - Motores Paso a Paso (Stepper)
- Motores de Corriente Alterna (CA)
  - Motor Síncrono

- Motor de Inducción (Asíncrono)
- Motor de Reluctancia variable

Para la aplicación del robot uniciclo al ser un robot pequeño y utilizan dispositivos electrónicos se opta por utilizar los motores de corriente continua para la facilidad de su alimentación eléctrica, a continuación se muestra un pequeño resumen y comparativa de los motores de corriente continua.

4.4.1.1 Motor de corriente continua con Escobillas

El motor de corriente continua con escobillas se ha utilizado durante mucho tiempo como la opción mas común de motor para aplicaciones de electrónica. Este tipo de motor cuenta con escobillas que mediante contacto con un conmutador eléctrico generan el flujo de corriente y en consecuencia se produce el movimiento del eje del motor (CochesRc, 2023).

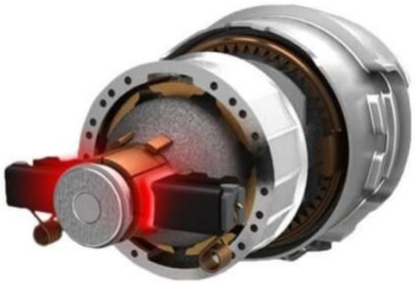


Ilustración 7-Motor de Corriente Continua con Escobillas

Las características de este tipo de motor son las siguientes:

- Este motor usualmente tiene una vida útil mas corta que los motores brushless o Stepper debido al desgaste de los elementos mecánicos y ocupan un mantenimiento mayor que el de los motores brushless.
- El costo de este tipo de motor es menor en comparación a motores brushless y stepper por lo cual son una mejor opción en caso de pequeños presupuestos.
- Generalmente su mantenimiento es mas fácil que otros tipos de motores ya que se hace mas fácil cambiar componentes tales como las escobillas en comparación con los motores brushless que ocupan mas destreza para repáralos.

4.4.1.2 Motor de corriente continua sin Escobillas (Brushless)

Este tipo de motor eléctrico están siendo muy utilizado en la actualidad en proyectos de electrónica así como en la industria. Este tipo de motor funciona sin necesidad de escobillas como lo dice su nombre funcionan por medio de un conmutador electrónico y permite eliminar las fricciones en el motor al no necesitar un conmutador físico y mejoran la eficiencia energética.

Las características de los motores brushless son las siguientes:

- Son mas eficientes que los motores de corriente continua con escobillas y por lo tanto la batería puede durar mas tiempo.
- Al no tener escobillas se elimina la fricción de tener un conmutador físico y eso hace que se desgasten menos y su vida útil sea mayor (CochesRc, 2023).
- Se puede tener mayor velocidad que con los motores con escobillas al ser controlados de manera electrónica.
- Su costo es generalmente mayor que el de los motores de corriente continua con escobillas.

Ilustración 8-Motor de Corriente Continua (Brushless)

4.4.1.3 Motor Paso a paso (Stepper)

Los motores paso a paso son motores parecidos a los motores brushless en el sentido que tampoco llevan escobillas. Este tipo de motor es un dispositivo que convierte impulsos eléctricos en desplazamiento lineales o angulares. Este tipo de motor también ocupa de un conmutador electrónico para poder funcionar y son motores que se utilizan en aplicaciones de máxima precisión (Zhang, S.F.).

Las características mas importantes de este tipo de motor son las siguientes:

- Tienen un gran par en el arranque para bajas velocidades.
- Son motores de baja velocidad y alto par en arranque.
- La posición y velocidad se controlan en bucle abierto por lo cual hay un ahorro en la compra de encoders pero a veces sueles haber problemas de desfase.
- La posición y velocidad del motor dependen de los pulsos que se le proporciones al motor de manera electrónica.

Ilustración 9-Motor Paso a Paso

4.4.2 ENCODERS

Un encoder es un elemento que se utiliza para el control y monitoreo de motores eléctricos. Un encoder es un dispositivo que transforma el movimiento del motor eléctrico en una señal que luego es leída por algún dispositivo como puede ser un microprocesador para utilizarse para determinar la posición, velocidad o dirección el que esta esta girando el motor eléctrico. Los encoder funcionan mediante la detección de señales ópticas o magnéticas. Un encoder con tecnología óptica funciona emitiendo un haz de luz que pasa por un disco de determinada resolución dependiendo del encoder que luego es interrumpida para luego pasar al dispositivo foto detector para determinar la posición (EncoderProduct, S.F.).

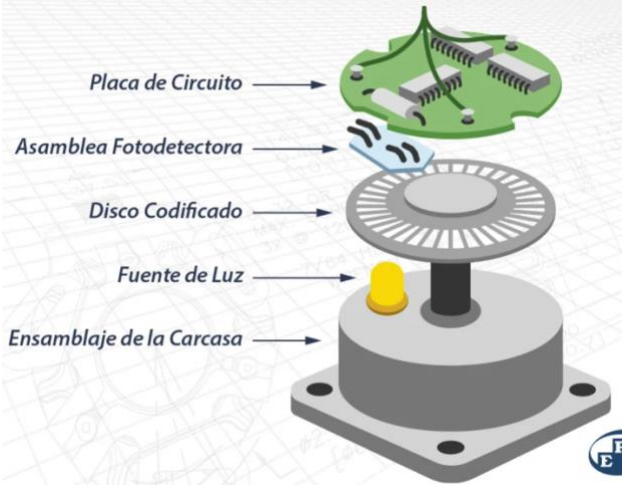


Ilustración 10-Estructura Encoder Óptico

Los encoder pueden ser de dos tipos distintos:

- Incremental: En este tipo de encoder solo se muestra las posición que ha cambiado sin tener una referencia en todo momento.
- Absolutos: En este tipo de encoder se tiene una referencia absoluta de la posición por lo tanto se pude indicar la posición que ha cambiado y como ha cambiado en referencia a la posición absoluta del encoder, por lo tanto se obtiene mas información y nunca se pierda la posición para aplicación que requieran de mas precisión.

4.4.3 IMUS (INERTIAL MEASUREMENT UNITS)

Las unidades de medición inerciales (IMUS) son dispositivos que miden distintos cambios en velocidades angulares, fuerza gravitacionales y orientaciones por medio de sensores que incorporan tales como acelerómetros, giroscopios y magnetómetros. Este tipo de dispositivos se utilizan en distintas aplicaciones como pueden ser: robótica, sistemas de navegación, celulares y sistemas de control. Para conocer la orientación de un cuerpo en un espacio tridimensional es importantes conocer su ángulos para conocer su posición, velocidad u orientación en todo momento para luego ser utilizado por microprocesador para poder controlar al objeto en todo momento (Nuri, 2013).

Con el giroscopio se puede saber la orientación y el giro del dispositivo mediante la posición angular. Mediante el acelerómetro se puede saber la aceleración lineal del dispositivos en distintas orientaciones y también la inclinación del dispositivo para conocer su orientación en todo momento.

4.5 CONTROL AUTOMÁTICO

El control automático es una de las ramas de la ingeniería que tiene como objetivo mantener distintas variables de un proceso en unos ciertos márgenes. El control automático es utilizado en distinto procesos industriales para la automatización de distintos procesos existentes. El control automático se divide en su mayoría de casos en unos de estos dos sistemas:

- **Sistema de lazo abierto:**

  En este tipo de sistema no se dispone de una señal de realimentación, por lo cual no se compara la señal de referencia a salida por lo tanto no se sabe el estado del proceso con exactitud en tiempo real.

- **Sistemas de lazo cerrado:**

  En este tipo de sistema se cuenta con realimentación a la salida del proceso. Se compara la señal de salida del proceso con la referencia y se ajusta hasta eliminar el error. En este tipo de control se ocupan tener sensores en la salida de los procesos para conocer su estado en todo momento y poder realizar los ajustes necesarios para alcanzar el valor de referencia de proceso. Aumenta el costo del sistema pero a cambio se obtiene un proceso mucho mas exacto en la salida. Los sistemas de lazo cerrado usualmente tiene controladores PD o PID.

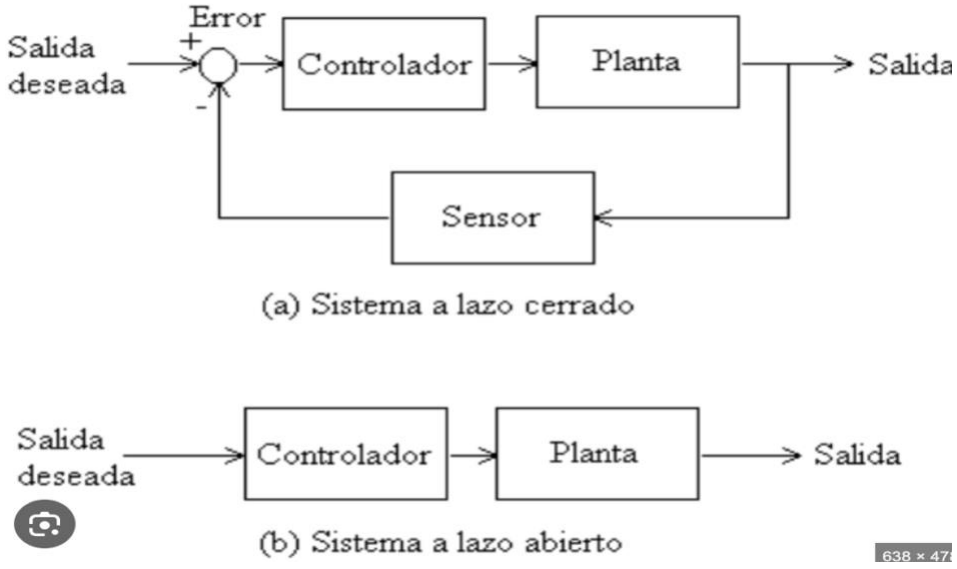


Ilustración 11-Sistemas de lazo abierto y lazo cerrado

4.5.1 CONTROLADOR PID

El controlador PID se utiliza en sistema de lazo cerrado para que el sistema alcance en la salida el valor deseado, en este sistema se le aplica una acción proporcional, una acción derivativa y una acción integral al sistema. Un controlador PID ocupa de una retroalimentación que usualmente es proporcionado por un sensor en la salida del sistema. Es una de las mejores formas de diseñar un controlador sin embargo en sistemas complejas se hace algo difícil encontrar o cálculos los parámetros adecuados para el sistema. Cada una de estas acciones tiene una finalidad diferente en el diseño del controlador (PICUINO, 2024).

- **Acción Proporcional**

  Se multiplica el valor de esta acción al error del sistema a una constante KP que determina la acción proporcional que va a tener el controlador PID. Aumentar la acción proporcional aumenta la velocidad del sistema, disminuye el error y aumenta la oscilación del sistema (PICUINO, 2024).

- **Acción Derivativa**

  Esta acción es proporcional a la derivada de la señal de error multiplicada por una constante KD. Aumentar KD tiene los siguientes efectos: reduce las oscilaciones del sistema, disminuye la velocidad del sistema y el error en régimen permanente se mantiene igual (PICUINO, 2024).

- **Acción Integral**

  Esta acción calcula la integral del error y la multiplica por una constante Ki. Esta constante va sumando el error y en consecuencia se aumenta la acción integral. Sus características son: Disminuye el error en régimen permanente, aumenta las oscilaciones del sistema y aumenta la velocidad del sistema (PICUINO, 2024).

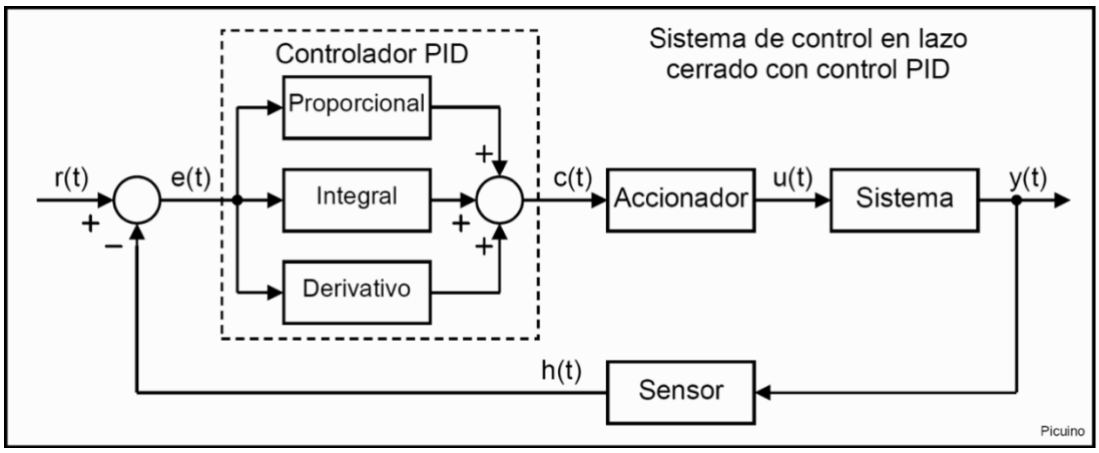


Ilustración 12-Esquema Controlador PID

4.5.2 TÉCNICAS CONTROL DE MOTORES ELÉCTRICOS BRUSHLESS

Para el control de motores eléctricos existen diferentes técnicas que cumplen con el objetivo de controlar la velocidad de un motor brushless, entre estas se encuentran las siguientes técnicas:

- **Control basado en conmutación sinusoidal**

  En este tipo de control se generan 3 corrientes sinusoidales desfasadas en 120 grados al devanado del motor para generar las señal de referencia del motor. En este tipo de control se necesita de un encoder para conocer con exactitud la posición del motor en todo momento y se emplea un controlador PD o PID dependiendo del procesos. El problema de este tipo de control es que a mayor velocidad se produce un mayor error al desalinearse el vector de corriente con el vector del motor lo que provoca que el control disminuya gradualmente a altas velocidades. Para solucionar este error se le ocupa proporcionar mayor corriente al motor lo que provoca una perdida de eficiencia y de energía (Juanpere, 2015).

- **Control Vectorial (Field Oriented Control (FOC))**

  Este tipo de control funciona controlando los vectores de corriente de manera directa es un sistema. Se controla el par y el flujo del motor de manera independiente lo que permite un mejor control del par. En este tipo de control se aplican modelos matemáticos para convertir el sistema del motor en un sistema lineal que luego permite simplificar los cálculos siempre que se conozca la posición del motor con un encoder para asegurarse que los campos magnéticos del rotor del motor se mantengan ortogonales entre si. En la salida se comparan las corrientes del modelo con

las corrientes calculadas que pasan por controladores PI del par y del flujo para alcanzar una respuesta adecuada. Este tipo de control permite un par alto en el arranque y un control preciso con alta eficiencia (Juanpere, 2015).

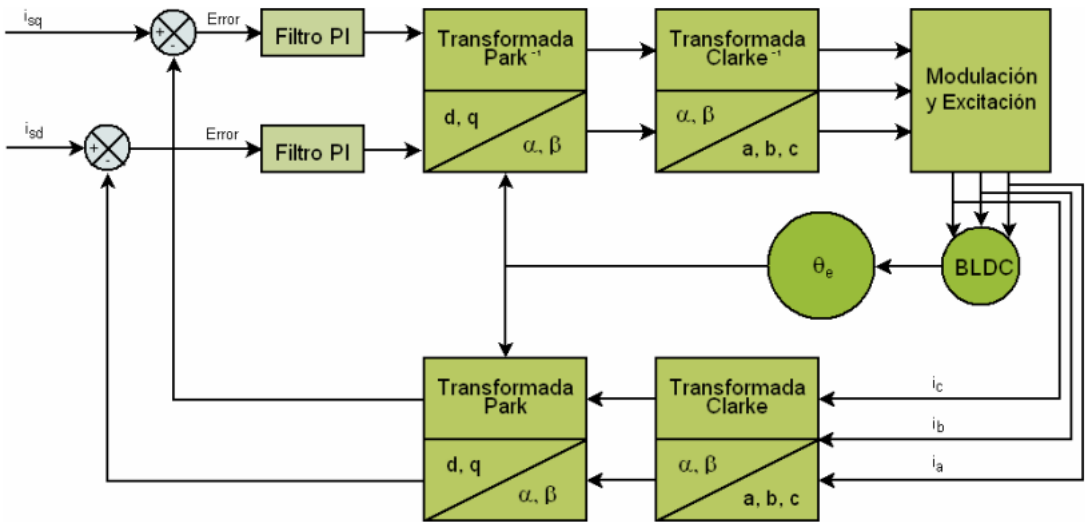


Ilustración 13-Esquema Control Vectorial (FOC)

4.6 RESISTENCIA DE MATERIALES Y FACTOR DE SEGURIDAD

La resistencia de materiales es una de las ramas de la ingeniería que encarga de realizar el análisis comportamiento de los sólidos que son sometidos a cargas exteriores.

Resistencia y Rigidez

Se dice que un cuerpo puede resistir unas determinadas cargas cuando dicho solido no sufre una rotura o deformación. Sin embargo, estas cargas pueden producir deformaciones en el cuerpo que hacen que su trabajo dentro de una estructura no sea el adecuado. Es aquí donde entra el concepto de rigidez. Un cuerpo será más rígido frente a determinadas fuerzas cuanto menos deformaciones sufra. Para que un cuerpo se considere rígido este no debe deformarse ante las fuerzas externas o que su deformación sea tan pequeña que el cuerpo no sufra afectación. Por ejemplo, una viga de un puente se puede deformar, pero su máxima deformación no debe impedir que la gente pase por el puente. Si es así la viga es rígida (Industrial, 2019).

4.6.1 ESFUERZOS

Esfuerzo se le conoce a la fuerza interna que un elemento de una estructura experimenta cuando es sometido a fuerzas externas. Estos elementos deben soportar muy bien estos esfuerzos sin deformarse o romperse.

Dependiendo de la dirección y sentido de las fuerzas actuantes y de la posición del cuerpo sobre el cual actúan, se consideran los siguientes esfuerzos:

- **Tracción**

Este esfuerzo es perpendicular a la sección transversal del cuerpo. Este tipo de esfuerzo tiende a alargar el cuerpo (Industrial, 2019).

- **Compresión**

Igual que en el caso anterior, es perpendicular a la sección transversal del cuerpo, solo que este tiende a acortar el cuerpo (Industrial, 2019).

- **Cizalladura o Cortadura**

Este es cuando actúan fuerzas contrarias sobre el cuerpo, que están situadas en dos planos contiguos, tienden a deslizarse entre sí las secciones en que actúan (Industrial, 2019).

- **Flexión**

Este es cuando actúan fuerzas que tienden a doblar el cuerpo (Industrial, 2019).

4.6.2 DEFORMACIONES

Según Ramírez, López, Manzanillo & Portillo (2013): “Son consecuencia de procesos mecánicos, a partir de fuerzas externas o internas que afectan a las características mecánicas de los elementos constructivos. En el caso de las deformaciones, son una primera reacción del elemento a una fuerza externa, al tratar de adaptarse a ella”.

Existen las siguientes clasificaciones para el comportamiento de las deformaciones de los materiales:

- **Comportamiento Elástico:** se da cuando un sólido se deforma adquiriendo mayor energía potencial elástica y, por tanto, aumentando su energía interna sin que se produzcan transformaciones

termodinámicas irreversibles. La característica más importante del comportamiento elástico es que es reversible: si se suprimen las fuerzas que provocan la deformación el sólido vuelve al estado inicial de antes de aplicación de las cargas (Ramírez, López, Manzanillo, & Portillo, 2013).

- **Comportamiento Plástico:** aquí existe irreversibilidad; aunque se retiren las fuerzas bajo las cuales se produjeron deformaciones elásticas, el sólido no vuelve exactamente al estado termodinámico y de deformación que tenía antes de la aplicación de estas (Ramírez, López, Manzanillo, & Portillo, 2013).

4.7 COEFICIENTE DE SEGURIDAD

Según INFAIMON (2018): "El coeficiente de seguridad, también conocido como factor de seguridad, es el cociente resultante entre el valor calculado de la capacidad máxima de un sistema y el valor del requerimiento esperado real al que se verá sometido. Por este motivo, se trata de un número mayor que uno, que indica la capacidad de exceso que tiene el sistema sobre sus requerimientos."

Las formas más usuales de definir un cociente de seguridad de un diseño son las siguientes:

- Como cociente entre la resistencia del material y la tensión realmente existente.
- Como cociente entre la fuerza última o máxima para un funcionamiento correcto y la fuerza realmente existente (INFAIMON, 2018).

Por otra parte, en un proyecto que contiene elementos mecánicos, existen dos alternativas para incluir un coeficiente de seguridad en el mismo:

- Mayorar las fuerzas realmente esperadas, multiplicándolas por el coeficiente de seguridad (coeficiente de seguridad de mayoración de carga).
- Minorar la resistencia realmente esperada del material, dividiéndola por el coeficiente de seguridad (coeficiente de seguridad de minoración de la resistencia) (INFAIMON, 2018).

Para concluir un valor del coeficiente de seguridad superior a la unidad indica seguridad ante el fallo, tanto mayor, cuanto más elevado sea su valor, mientras que un valor inferior al valor de 1 de coeficiente de seguridad demuestra inseguridad o probabilidad de fallo. En función de la variabilidad de las cargas aplicadas y las propiedades del material, cada valor del coeficiente de seguridad se puede asociar a una probabilidad de fallo o de supervivencia de la pieza analizada (INFAIMON, 2018).

V. METODOLOGÍA

5.1 ENFOQUE

El enfoque de este proyecto es cuantitativo debido a que en esta investigación se van a recolectar diferentes datos empíricos y luego se procederá a analizar esto datos para así poder solucionar el problema planteado.

El procedimiento que se sigue para la metodología cuantitativa es hipotético-deductivo. Se inicia formulando hipótesis de la teoría, luego de esto se definen las variables y sigue con la recolección de datos para luego analizarlos y posteriormente interpretarlos. Los datos son la base para la prueba de la hipótesis y los modelos formulados por el investigador (Monje, 2011).

5.2 PREGUNTAS DE INVESTIGACIÓN

En la elaboración de este trabajo se presentan incógnitas que se resolverán en el desarrollo de este proyecto. Las preguntas son las siguientes:

- ¿Qué diseño se debe usar para el robot uniciclo?
- ¿Qué tipo de motores son convenientes?
- ¿Cómo reducir los costos y obtener un prototipo funcional?
- ¿Cómo escoger los componentes electrónicos a utilizar?
- ¿Qué estrategia de control es la mas optima para el diseño del controlador de robot uniciclo?

5.3 TÉCNICAS E INSTRUMENTOS APLICADOS

Para poder llevar a cabo este proyecto fue necesario el conocimiento de Diseño Asistido por computadora, en este caso se utilizo el software SolidWorks 2022 y MatlabR2022b. Esta herramienta fue utilizada para realizar las modificaciones en el diseño del modelo anterior y así mismo para realizar las diferentes simulaciones de tensión y flexión para observar comportamiento del robot uniciclo ante diferentes cargas.

Tabla 1- Técnicas e Instrumentos Utilizados en el Proyecto

Técnicas	Instrumentos	Actividades
Diseño Asistido por Computadora (CAD)	SolidWorks 2022	Diseño, Simulaciones de cargas y elaboración de planos.
Programación en Matlab Simscape	Matlab R2022B	Diseño y simulación del prototipo y diseño del Control
Conocimiento en Electrónica	Máquina de Soldar, Cinta Adesiva, pegamento, Destornilladores, Juego de llaves Allen,	Construcción del Prototipo del Robot Uniciclo

5.4 METODOLOGÍA DE VALIDACIÓN

Para comprobar la validez del proyecto se hizo uso del software SolidWorks 2022 y Matlab R2022. Con la ayuda de este software se realizan las simulaciones estáticas del dispositivo. En este software se procedió a seleccionar el material de cada componente del ensamblaje del dispositivo de rehabilitación y con esto se procedió a realizar las simulaciones de tensión y flexión del robot uniciclo. Luego se obtuvo el factor de seguridad. Con esto se pudo observar que el diseño propuesto está bien y no tiene grandes deformaciones que puedan poner en peligro el diseño propuesto.

Además de esto también se valido el diseño con el uso de software Matlab Simscape para verificar que el diseño elaborado en solidworks fuera funcional para la aplicación propuesta. Así mismo se pudo comprobar que era posible mantener el balance del robot por medio del control automático. Se simulo el sistema en bucle cerrado simulando los motores y sensores que serían utilizados en el diseño del prototipo físico.

VI. DESARROLLO DEL TRABAJO Y RESULTADOS

6.1 DISEÑO 3D DEL ROBOT UNICICLO

Los diseños que se estudiaron para la realización de este proyecto fueron la base para el desarrollo del prototipo, ya que se tomaron en cuenta varios aspectos fundamentales que permitieron que hubieran otros prototipos funcionales. Una vez se selección el diseño mas idóneo para la realización del prototipo se realizo un diseño preliminar del robot uniciclo.

Una vez de que ya se tuvieran las piezas diseñadas y se realizara el ensamblaje de todos los componentes con el uso de Solidworks se investigo acerca de cuales componentes electrónicos se utilizarían para el prototipo. Una vez ya se tenia totalmente definido los componentes electrónicos se realizaron modificaciones a las distintas partes y se colocaron los agujeros para la tornillería a medida de los distintos componentes electrónicas que se utilizarían. Se realizaron las distintas piezas tomando en cuenta de que fuesen ideales para la impresión 3D y que su montaje fuese relativamente sencillo.

6.1.1 DESCRIPCIÓN DE LAS PIEZAS DEL ROBOT UNICICLO

A continuación en esta sección se describirán las piezas diseñadas del Robot uniciclo y la explicación de que realiza que cada pieza y explicación de su diseño:

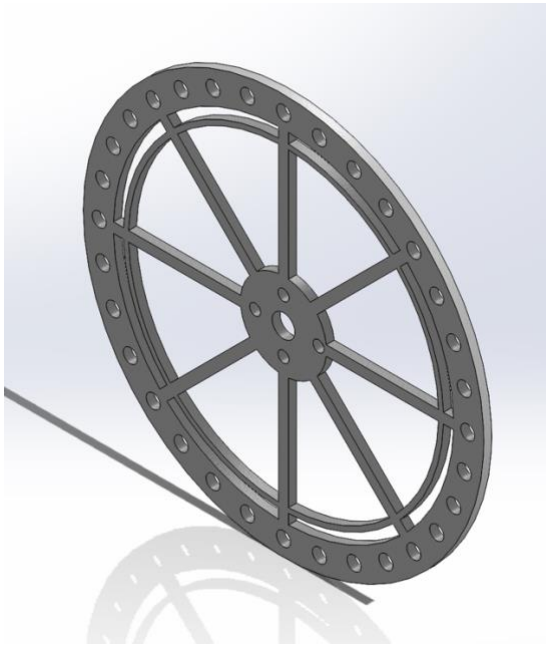


Ilustración 14-Rueda de reacción

En la ilustración 14 se muestra una de las partes mas importantes de la realización del prototipo del robot uniciclo. Esta pieza es la rueda de reacción que se encarga de proporcionar el par para mantener el robot en equilibrio. La pieza cuenta con los agujeros para poder fijar el motor gimball 2804 con su tornillería. Además cuenta con los agujeros para colocar tornillos en toda la circunferencia de la rueda de reacción, con estos tornillo se puede aumentar la masa de la rueda de reacción y permite realizar distintas pruebas con distintas masas de la manera que sea mas conveniente.

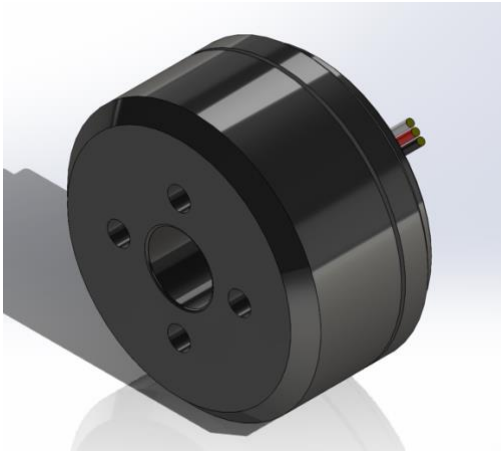


Ilustración 15-Motor Brushless Gimball 2804

En la ilustración 15 se muestra el motor gimball que se utilizaría para el diseño del robot uniciclo. Es un motor brushless Gimball que se utiliza en distintas aplicación de cámaras y drones. En este caso se utilizo el diseño de Abdullah Osdurmazz que esta disponible en grabcad. Es importante tener el diseño para comprobar que la tornillería y las dimensiones del motor funcionan con el uso del robot y no hay ninguna interferencia.

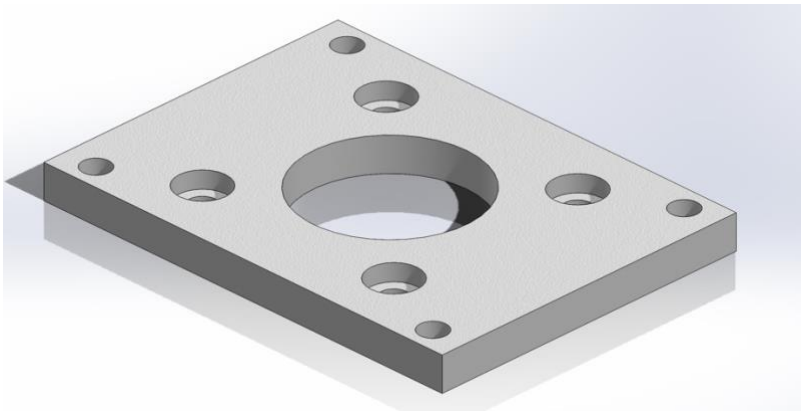


Ilustración 16-Pieza Placa Base

En la ilustración 16 se muestra la pieza placa base. Esta pieza es la que se encarga de conectar la parte superior e inferior del robot. La pieza lleva los agujeros para colocar la tornillería y fijar la pieza que va a soportar toda la parte superior del robot, además de esto las pieza cuenta con un agujero central que va a permitir que pasen los cables que se utilizaran para conectar el motor de la rueda inferior y del encoder hacia el controlador.

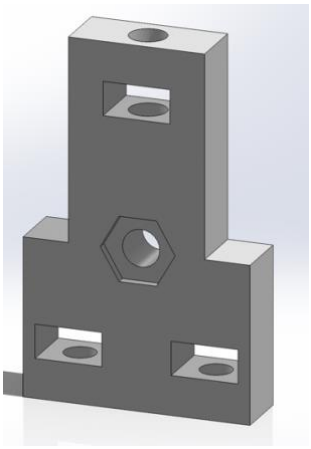


Ilustración 17-Pieza Unión base superior

En la ilustración 17 se muestra la pieza unión de pase superior. Esta pieza sirve para conectar la base en la que va montada la rueda de reacción y la base de la batería. Cuenta con la tornillería para fijar la pieza con los otros componentes.

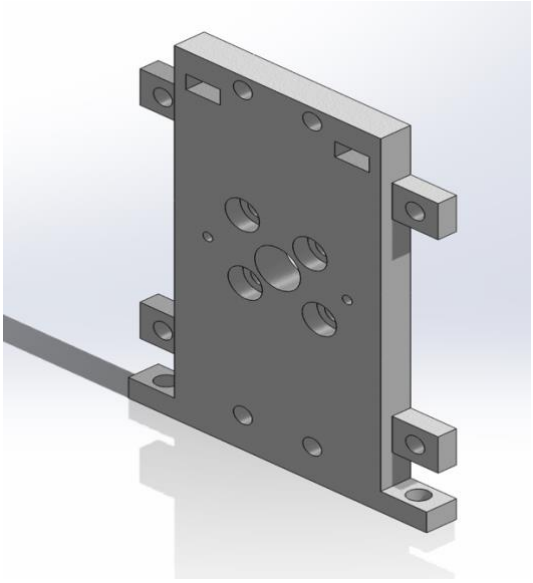


Ilustración 18-Pieza Base Rueda de Reacción

En la ilustración 18 se puede observar la pieza base de la rueda de reacción. En esta pieza se tienen los agujeros para poder fijar el motor brushless 2804 que luego llevara la rueda de reacción atornillado. Cuenta con un agujero central para la colocación del encoder AS5600 junto con su cableado. Así mismo también esta pieza cuenta con los tornillos para poder fijar por la parte posterior el controlador gimball 3.1 que es el encargado del controlar el robot.

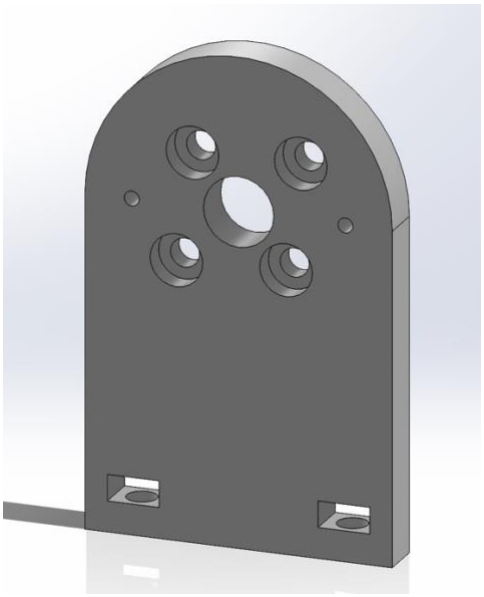


Ilustración 19-Pieza base Rueda

En la ilustración 19 se muestra la pieza que se utiliza para fijar el motor de la rueda inferior y que luego se une con la base intermedia del robot uniciclo. Cuenta con el agujero donde se va a colocar el encoder AS5600.

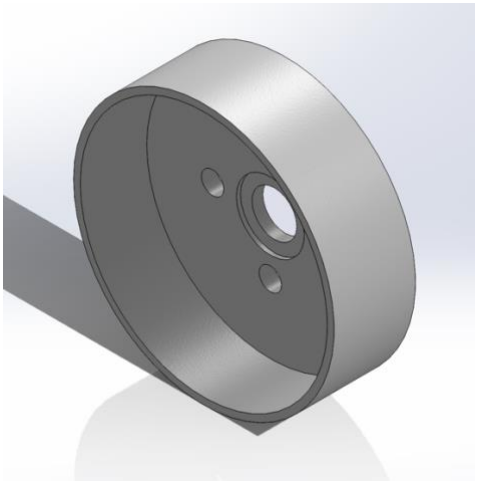


Ilustración 20-Pieza Carcasa Motor

En la ilustración 20 se muestra la pieza en la que se va colocar el motor de la rueda inferior. La pieza tiene las medidas del motor brushless 2804 y los agujeros para colocar su tornillería y poder fijar el motor junto con el encoder.

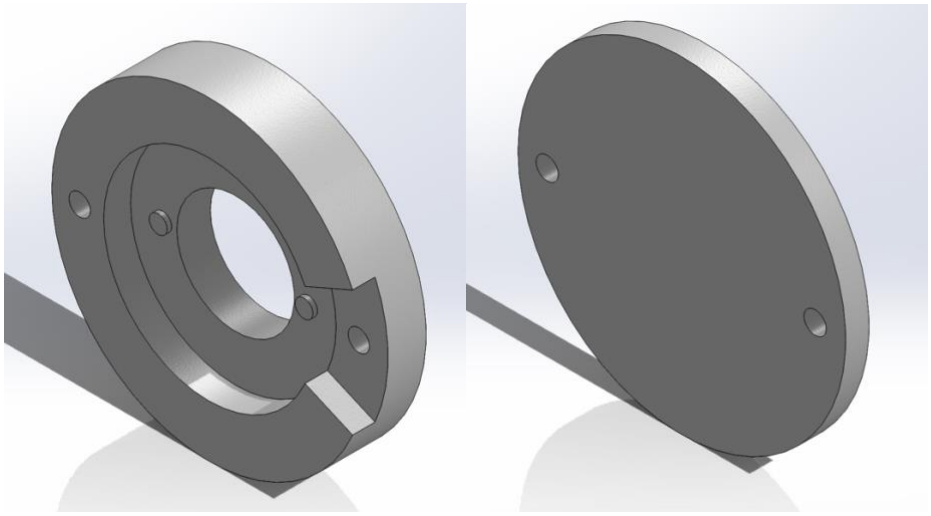


Ilustración 21-Pieza Encoder AS5600 y su tapadera

En la ilustración 21 se puede observar la pieza que es la encargado de llevar el encoder AS5600 en su interior. Se realizo esta pieza para poder montar el encoder y que pueda ir fijado directamente en el eje

del motor y que encaje de una manera firme y no se mueva. La pieza tiene las dimensiones para ser utilizada únicamente con este encoder y para atornillarlo directamente a la pieza que contiene el motor. De igual manera se realizo una tapadera para cubrir la pieza y poder cambiar el encoder en cualquier momento en caso de algún fallo o avería.

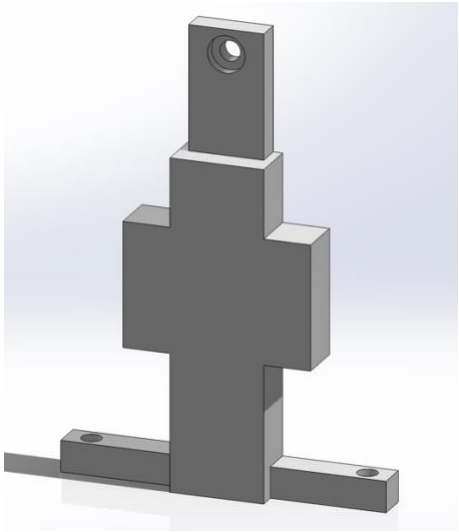


Ilustración 22-Pieza Base Batería

En la ilustración 22 se muestra la pieza base de la batería. Esta el pieza en la que se colocar la batería que se encargara de suministrarle la energía al robot uniciclo y también se pude colocar un modulo bluetooth para modificar el robot de manera remota.

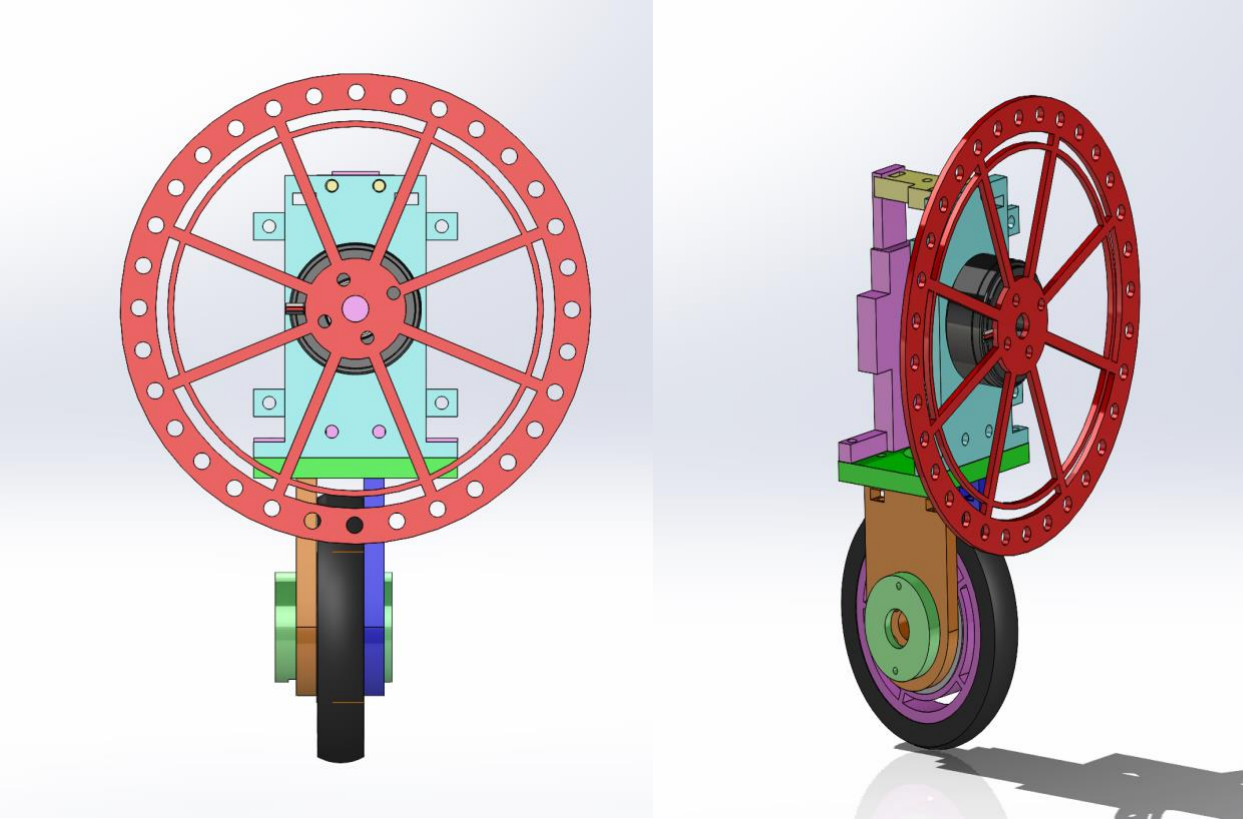


Ilustración 23-Ensablaje final de Prototipo del robot Uniciclo

Finalmente en la ilustración 23 se muestra el diseño final del robot uniciclo. De esta manera queda el robot al proceder a ensamblar todas las piezas que serán impresas en 3D. Así mismo se comprueba que no hayan interferencias entre los componentes y se realizaran las simulaciones estáticas del robot uniciclo.

6.1.2 SIMULACIÓN DISEÑO 3D UTILIZANDO SOLIDWORKS

Una vez que ya se tuvo el diseño final que se utilizaría, se procedió a realizar las simulaciones de esfuerzos resultantes para observar la viabilidad del diseño propuesto ante las posible cargas que podría soportar el robot uniciclo y para verificar que el diseño fuese seguro y no sufriera ninguna deformación. Además de esto se realizo el calculo del factor de seguridad para comprobar la seguridad del diseño propuesto.

El proceso para realizar el análisis de elementos finitos utilizando solidworks es un proceso relativamente sencillo debido a las herramientas con las que cuenta el software. A continuación se describirán los pasos mas importantes para realizar un análisis de elementos finitos utilizando el software solidworks:

1. Verificar que se dispone del complemento de simulación estática de solidworks.
2. Previamente a hacer el análisis se le debe asignar el material a cada componente diferente según sus propiedades. En este caso se aplico las propiedades mecánicas del material PLA a las piezas que serian impresas en 3D y se le asigno el material a los motores.
3. Se selecciona realizar un análisis estático en la pieza que contenía el ensamblaje final del robot uniciclo.
4. Se establecen las sujeciones y los puntos de contacto que tiene el robot, en este caso el punto de contacto seria en la rueda inferior.
5. Además de esto en las sujeciones se puede simular como si las piezas fuesen sujetas por tornillería o soldadura dependiendo de las construcción del dispositivo, en este caso se opto por que este unido por tornillería.
6. Se definen las cargas externas a las que estará sometido el robot, en este caso se calcularon como el peso de los componentes electrónicos que se encargan de la parte de control del equilibrio del robot. En este caso la fuerza que se aplico fue una fuerza distribuida en las caras del robot que soportarían el peso de los componentes. La fuerza aplicada fue de 10N en estas caras del robot uniciclo.
7. Se genera el mallado del robot, en este caso se opto por utilizar el mallado automático que genera solidworks. Dependiendo del tamaño del mallado se incrementa el tiempo de simulación y se puede necesitar de ordenadores mas potentes. El mallado que se utilizo cuenta con 124942 nodos y se realiza basado en la curvatura del robot.
8. Se generan los resultados del análisis estático.

9. Se crean los resultados y se ajustan los factores de deformación y las distintas escalas de las fuerzas resultantes.
10. En este estudio se hizo el Análisis de tensiones de Von Mises, el análisis de deformaciones y se calculo el factor de seguridad.

6.1.3 RESULTADOS DEL DISEÑO 3D

A continuación se adjuntan los resultados mas importantes que se obtuvieron del análisis estático del robot uniciclo junto con un breve análisis.

6.1.3.1 Análisis de tensiones de Von Mises

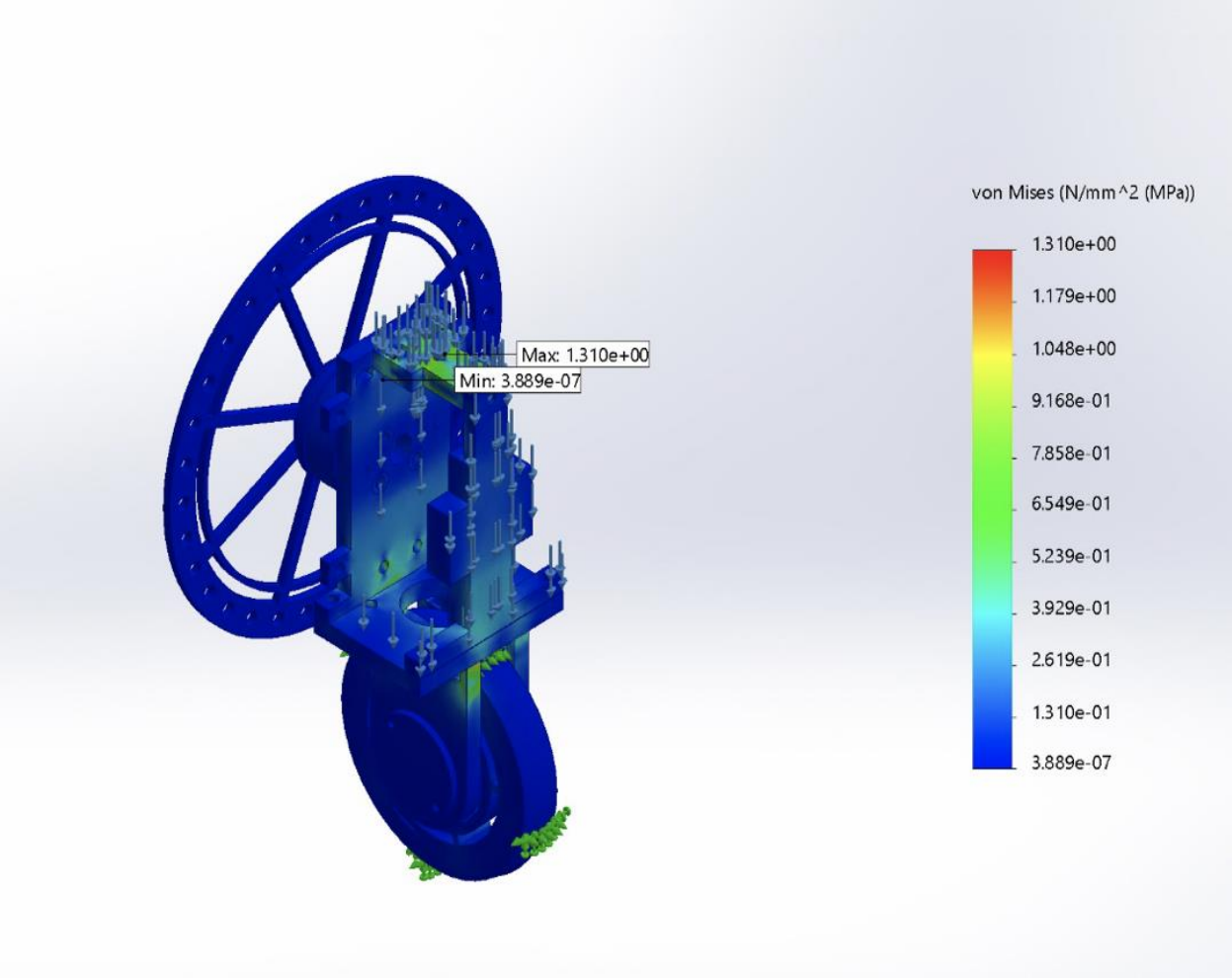


Ilustración 24-Resultados de tensiones de Von Mises

Se realizo el análisis de tensiones de Von Mises, este análisis permite ver las tensiones máximas a los que es sometido el robot uniciclo. El análisis de tensiones de von mises es uno de los análisis estáticos mas básicos e importantes para poder observar a que tensiones máximas es sometido un objeto y para poder determinar si es posible tener una fractura o deformación debido a las cargas externas. En el caso del análisis de Von Mises elaborado se puede observar que se obtiene una tensión máximas de 1.31Mpa el cual es un valor muy bajo, se puede observar que toda la pieza esta sometido a cargas muy parecidas y no sufre ninguna tensión excesiva. Luego comparando las tensión máxima del material PLA que es el material con el cual se imprimirán las piezas su limite de resistencia la flexión se sitúa en 55Mpa por lo tanto el valor al que será sometido el robot es muy bajo y no hay prácticamente ninguna posibilidad de deformación o de rotura elástica.

6.1.3.2 Análisis de Desplazamientos Máximos

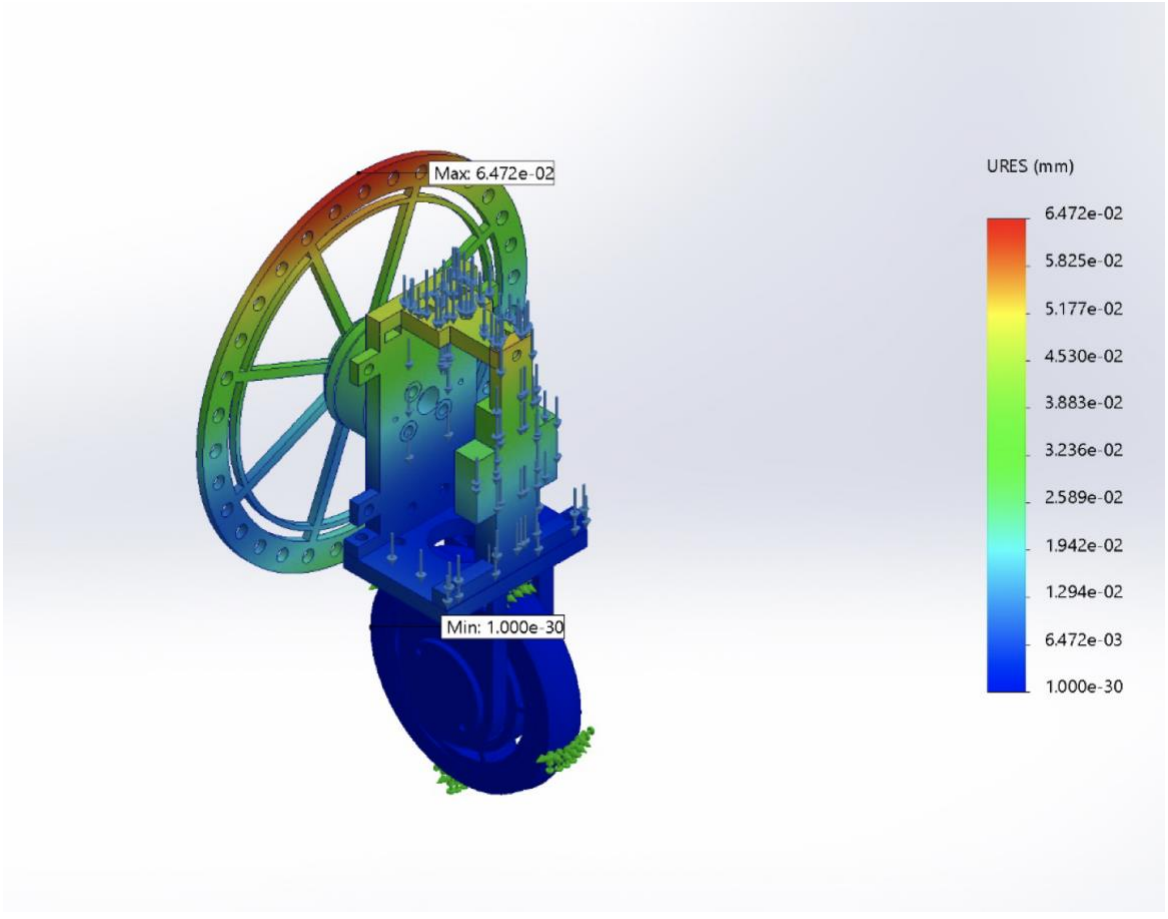


Ilustración 25-Analisis de Desplazamiento Máximos

Luego de obtener el análisis de tensiones de Von Mises se obtuvo el análisis de deformaciones máximas, este análisis es útil para observar las deformación máximas según la escala escogida a la es sometido el robot uniciclo según las fuerzas provocadas por los pesos de sus componentes electrónicos. Se puede observar en la imagen que se expresan las deformaciones máximas en milímetros, el valor máximo que alcanzan las deformación serán de 0.0647mm, lo que lo hace prácticamente un valor despreciable y que no tendrá ninguna afectación para el funcionamiento del robot.

6.1.3.3 Análisis de Factor de Seguridad

El ultimo análisis que se hizo mediante solidworks es el análisis del factor de seguridad. El factor de seguridad es uno de los análisis mas importantes que se realiza en materiales o estructuras. El factor de seguridad determina dividiendo la carga máxima que puede soportar entre la carga esperada o real. El valor del factor de seguridad debe estar siempre en una valor mayor a 1, debido a que bajo de este valor significa que el componente mecánica sufrirá un fallo mecánico y no es seguro. Un valor de 3 de factor de seguridad por ejemplo significa que el componente puede soportar hasta 3 veces la carga a la que es sometida. En el caso del estudio que se realizo se obtuvo un factor de seguridad mínimo de 49.6, con este valor se verifica que el diseño seleccionado es seguro y que es apto para la aplicación del robot uniciclo. Se podrían soportar cargas mucho mayores y no habría ningún inconveniente. El valor esta muy por encima de 1 por lo tanto, el diseño seleccionado no necesita de modificaciones adicionales.

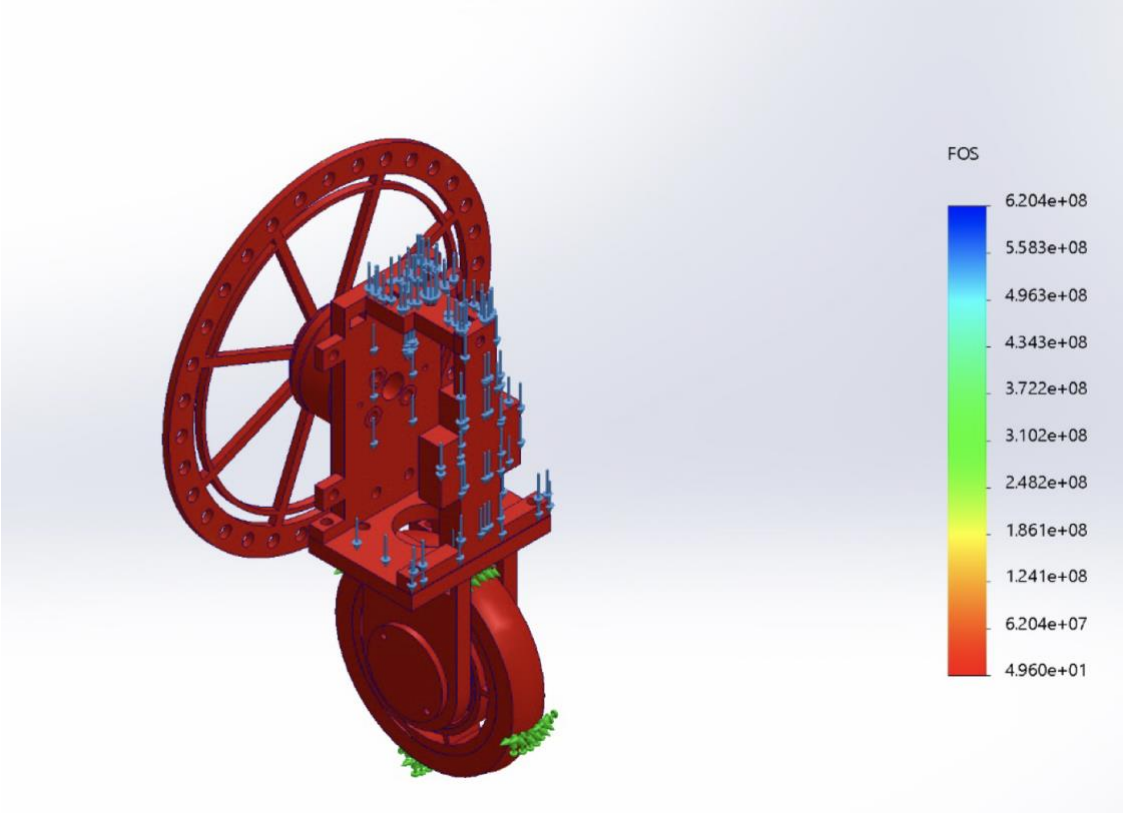


Ilustración 26-Analisis de Factor de Seguridad

6.1.3.4 Análisis de Masa del Robot Uniciclo

Finalmente se hizo el análisis de masa del robot uniciclo. Este análisis se utiliza para ver como se distribuyen las inercias del ensamblaje final y para lograr observar en que posición de encuentra el centro de masa del dispositivo. También se dan las propiedades y peso aproximado del robot uniciclo lo que será muy útil para las simulaciones con Matlab. El robot cuenta con una masa aproximada de 0.23Kg y su centro de masa se encuentra a una altura de 6.5cm.

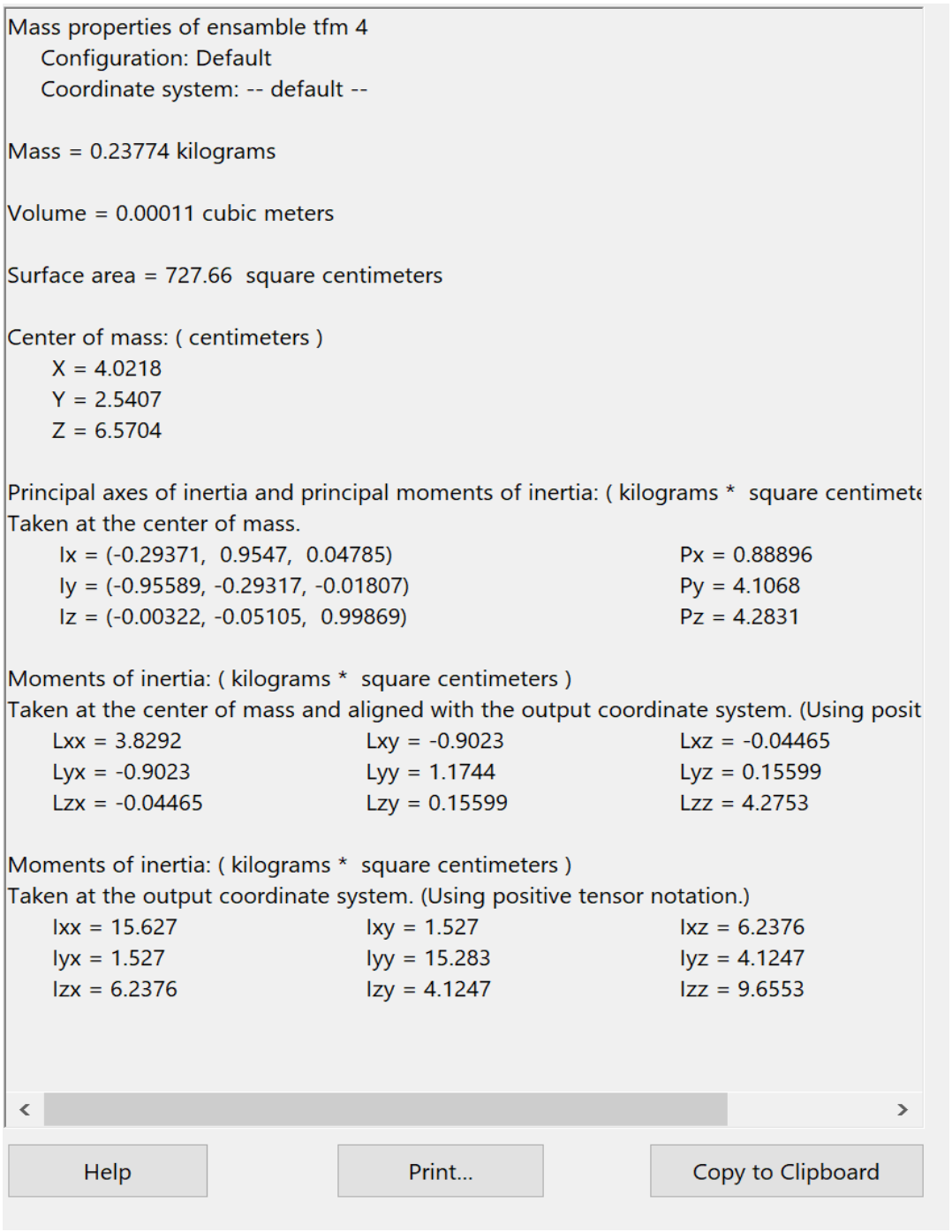


Ilustración 27-Propiedades de Masa del Robot Uniciclo

De estos análisis estáticos se obtuvo la siguiente tabla de resumen de los resultados.

Tabla 2-Resultados Obtenidos en Análisis Estático

Análisis Realizado	Conclusión de Análisis
Análisis de Tensiones Máximas y Mínimas (escala Von Mises)	La fuerza que se aplica al robot uniciclo de distribuye de manera uniforme, pudiendo tolerarla sin que afecte su posición o que alguna de las piezas se dañe. Las tensiones máximas se dan en la parte superior y las mínimas en la inferior. La tensión máxima es de 1.31 Mpa valor muy por debajo del limite de flexión del material utilizado para la impresión 3D tal como es el PLA.
Análisis de Deformaciones Máximas y Mínimas	Según resultados obtenidos se observó que, al aplicar la fuerza el dispositivo no sufrió ninguna deformación significativa; por lo que se concluye que el ensamble es seguro. Cabe destacar que las deformaciones máximas se dan en la rueda de reacción y el resto en la base del robot uniciclo.
Análisis del Factor de Seguridad del Dispositivo	Se obtuvo un Factor de Seguridad de 49.6, lo cual indica que el andador puede resistir prácticamente 50 veces el peso que tendrá que soportar, en este caso se diseñó tomando en cuenta el peso de los componentes electrónicos. Este factor nos da la seguridad de que el diseño el seguro y que no necesita modificaciones adicionales.

6.1.4 Planos de Construcción del Dispositivo

Una vez que el diseño fuese concluido y verificado mediante las simulaciones de elementos finitos se procedió a la ultima parte del diseño 3D del dispositivo. Se procedió a realizar los planos de las piezas del robot uniciclo en una manera clara y precisa para cualquier persona que en un futuro este interesada en la realización de este proyecto. Los planos se adjuntan en la parte de Anexos, del anexo 1 al anexo 13 en la parte final del presente informe.

6.2 REALIZACIÓN DE SIMULACIÓN CON MATLAB SIMSCAPE

Una vez que ya se tuvo el diseño del prototipo del robot uniciclo en Solidworks junto sus respectivas simulaciones se procedió a hacer una simulación del comportamiento físico del robot asignándole sus respectivos grados de libertad por medio del software Matlab con la herramienta Simscape. Matlab dispone de distintas herramientas para el simular sistemas físicos, con esta herramienta es posible introducir diseños previamente realizados y es posible observar como se comportan los actuadores tales como los motores por medio de sensores. De igual manera es posible realizar el diseño del controlador del sistema físico y realizar simulaciones para poder analizar su comportamiento.

6.2.1 ELABORACIÓN DEL ROBOT UNICICLO EN MATLAB SIMSCAPE

Para poder iniciar con la simulación en Matlab Simscape era necesario tener la piezas en formato Step. Este se hizo con la ayuda de solidworks para exportar las piezas ya en el formato Step. Para facilitar el proceso de unir las distintas piezas en Simscape se unieron varias piezas de la base del robot en solidworks y ya luego se exportaron de manera conjunta en formato Step. En total se exportaron 4 piezas para realizar el modelo: la rueda de reacción, el motor, la base del robot y el ensamblaje de la rueda inferior. Una vez que ya se tenían las piezas listas se procedió a colocarlas en Simscape con el bloque File Solid, este bloque permite importar geometrías diseñadas previamente desde distintos software de CAD hacia Matlab.

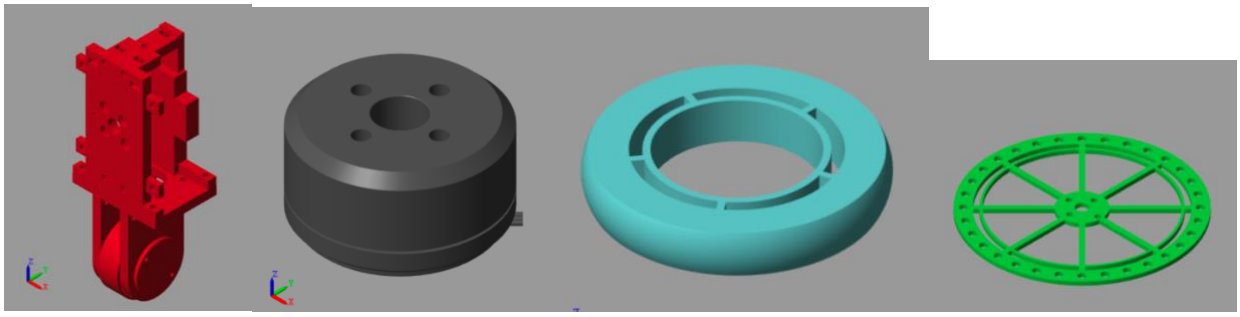


Ilustración 28-Componentes Exportados en Step en Simscape

Una vez que se colocaron las piezas en necesario realizar su ensamblaje, esto se realiza por medio de transformación geométricas donde es posible cambiar su orientación por medio de matrices de rotación o de traslación. Además de esto es necesario asignarle sus grados de libertad al robot, para esto se utilizan en nuestro caso dos juntas cinemáticas: pares prismático y pares de revolución. Los pares prismáticos permiten que el robot solo sea capaz de moverse a través de una eje y los pares de revolución permite

que los motores puedan girar y con esto hacer que las piezas del robot a las que irán sujetados giren al mismo tiempo.

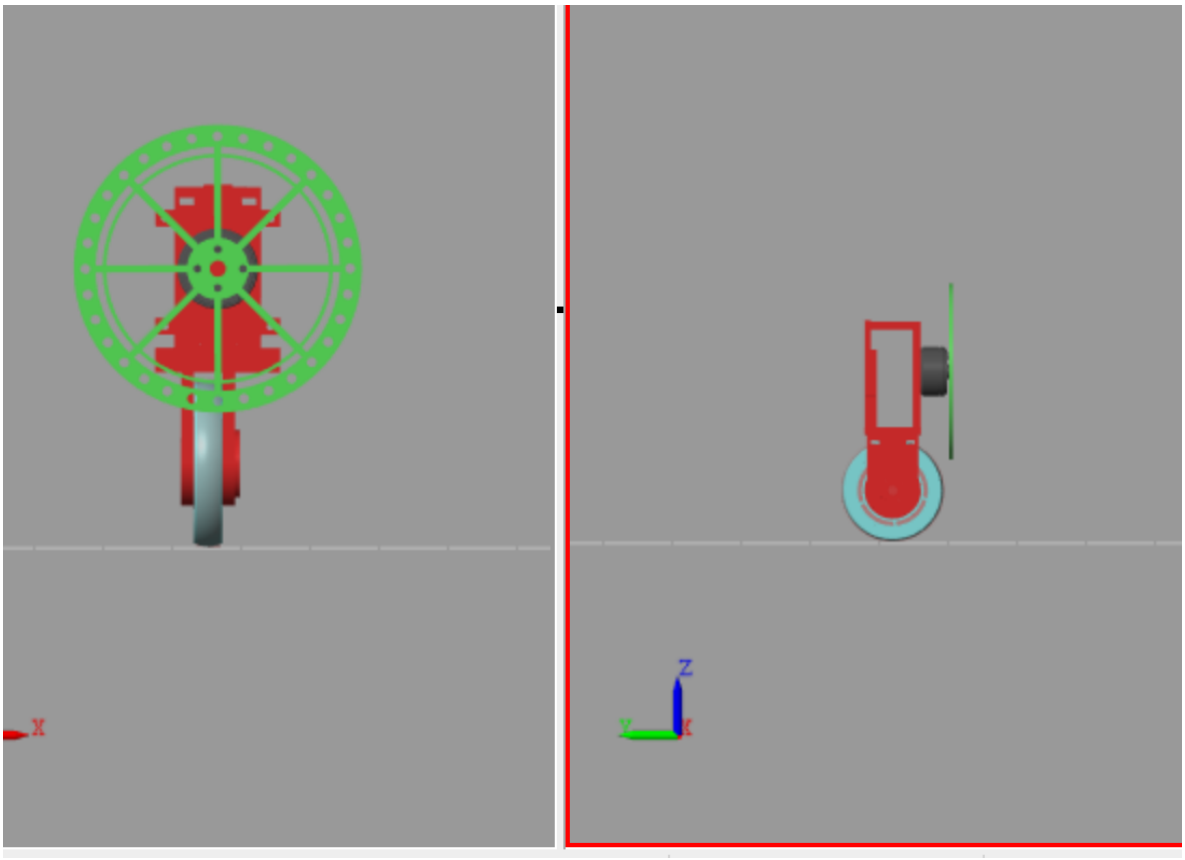


Ilustración 29-Ensamblaje final del Robot uniciclo en Simscape

El proceso para realizar el modelo en Simscape se puede resumir de la siguiente manera:

1. Exportar piezas en formato Step desde Solidworks.
2. Asignarle sus propiedades inerciales basadas en su densidad o masa.
3. Conectar la primera pieza al mundo con respecto a su eje de referencia.
4. Realizar las transformaciones geométricas ya sean se traslación o de rotación para orientar de manera correcta cada pieza.
5. Insertar las uniones de las piezas para asignarle sus grados de libertas correspondientes. En este caso se utilizaron uniones de revolución y prismáticas.
6. Activar en las opciones de los sensores de las uniones que contendrán los motores para simular los sensores y también activar la opciones de entradas por par. Mediante la entradas de par se

pueden introducir distintas señales al motor para que se mueva de acuerdo a las señales proporcionadas.

7. Colocar un plano infinito que es donde se apoyara el robot y se le asignan sus propiedades de amortiguamiento para simular de la manera mas cercana el contacto del robot con el suelo.
8. Se une el plano infinito con la rueda inferior.
9. Se colocan un par de revolución y una junta prismática al inicio del modelo para poder simular la caída del robot para simular la caída y poder mantener el equilibrio mediante el par de entrada que proporciona el motor con la rueda de reacción.
10. Se procede a realizar el diseño del controlador mediante la señal de entrada de la unión de revolución que une el motor y la rueda de reacción.
11. Se ajusta el controlador modificando los parámetros del controlador en bucle cerrado. En este caso se opto por un controlador PID.
12. Se comprueba que el robot es capaz de mantenerse de pie con el controlador diseñado y es capaz de mantener su equilibrio de manera precisa.
13. Se obtienen las graficas correspondientes a los resultados mas relevantes. En este caso las graficas de la posición y la velocidad de la rueda de reacción.

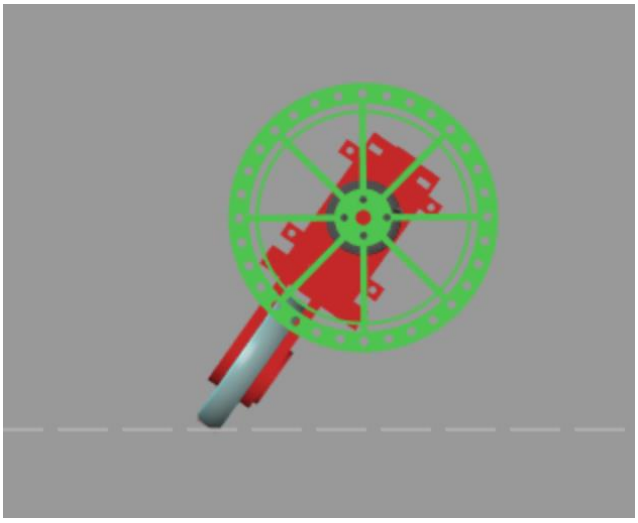


Ilustración 30-Simulacion de la caída del robot uniciclo para realizar la pruebas

6.2.2 RESULTADOS DE LAS SIMULACIONES EN SIMSCAPE

Se obtuvieron resultados satisfactorios de las simulaciones del robot uniciclo utilizando Simscape. En primer lugar se logro diseñar un Controlador que logro mantener el equilibrio del robot uniciclo proporcionándole una señal de entrada simulando el motor. Para que el robot lograra mantener su equilibrio en primer lugar se tuvo que hacer que el robot se cayera esto se hizo asignándole los grados de libertad correspondientes al robot y asignando todas las orientaciones de los distintos componentes del robot. A continuación se puede observar el diagrama de bloques final del ensamblaje elaborado en Simscape.

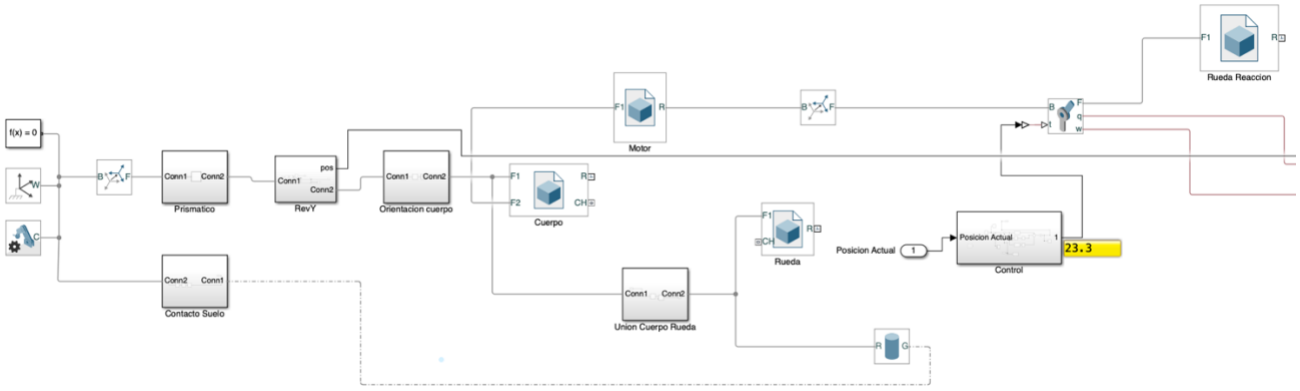


Ilustración 30-Modelo final del Robot Uniciclo en Simscape

Se diseño el controlador del robot uniciclo implementando un controlador PID. Mediante las herramientas de los sensores de posición de las juntas de revolución se simulo como si se tratase de un encoder conectado al motor. Se le asigno una señal de entrada de tipo sinodal al par de revolución y se intento un seguimiento de la posición basada en esta entrada. La parte mas difícil de este proceso en Simscape fue lograr la implementación del controlador. Se realizaron distintas pruebas con controladores en principio PD pero no se obtuvieron resultados satisfactorios, no se lograba corregir la posición del robot de manera correcta antes de que se cayera. Finalmente se opto por el diseño un controlador PID, con este controlador si fue posible mantener el equilibrio del robot de manera satisfactoria.

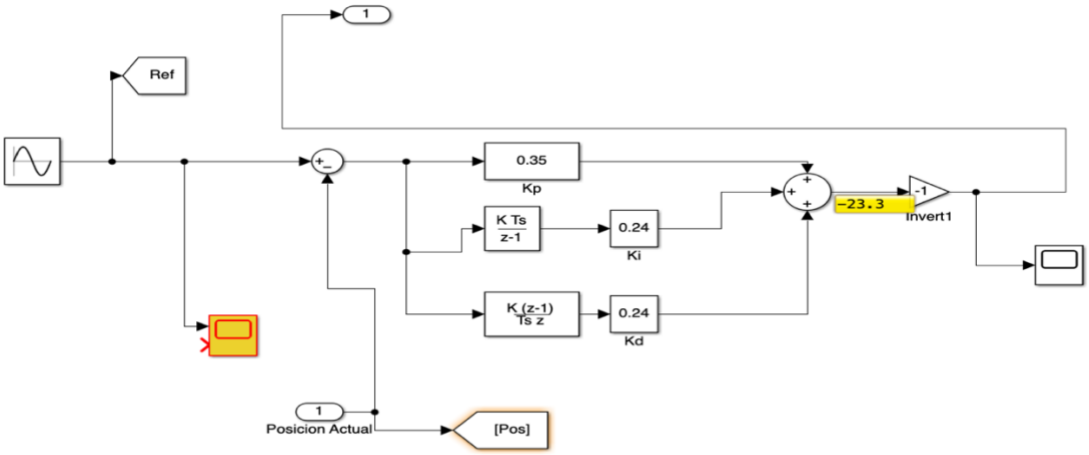


Ilustración 31-Controlador PID diseñado

En la entrada del controlador se le introdujo la señal sinodal de 16Hz, luego de intentar distintos métodos sin mucho éxito para ajustar el controlador finalmente se hizo a prueba y error. En primer lugar se intento ajustar el parámetros KP hasta que el motor fuera capaz de reacción con suficiente velocidad, luego se ajustaron los parámetros KI y KD hasta alcanzar un resultado satisfactorio. Al final los parámetros del controlador fue el siguientes: Kp de 0.35, KI de 0.24 y KD de 0.24.

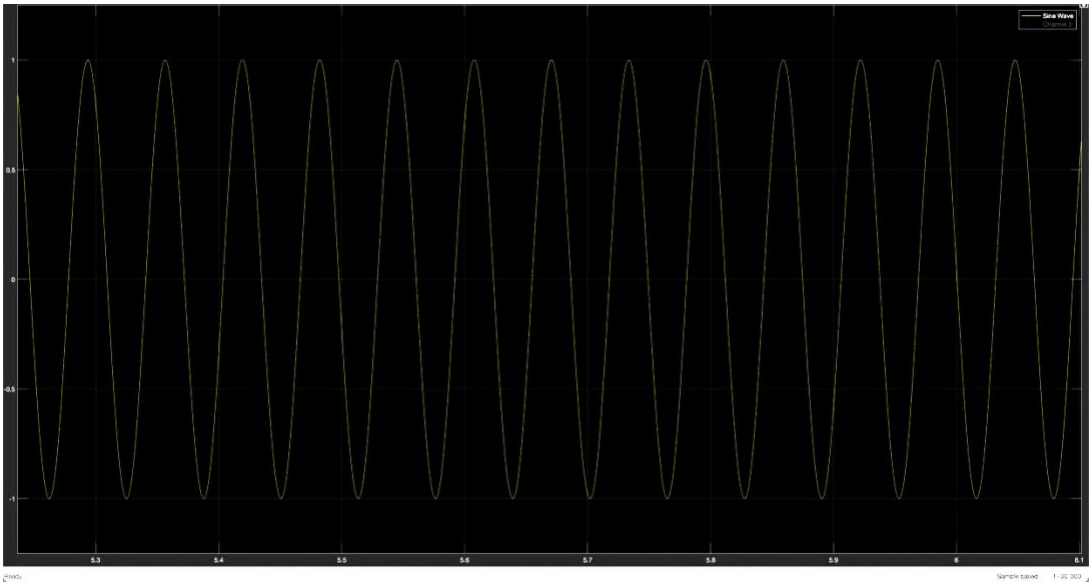


Ilustración 32-Señal de Referencia Sinodal

Esta fue la señal de entrada que se le proporciono al par de revolución en su entrada. Es una señal sinodal de 16 HZ y amplitud de 1. El controlador debía de ser capaz de seguir esta referencia de manera exitosa y de controlar la velocidad de la rueda de reacción para evitar que el robot uniciclo se cayera.

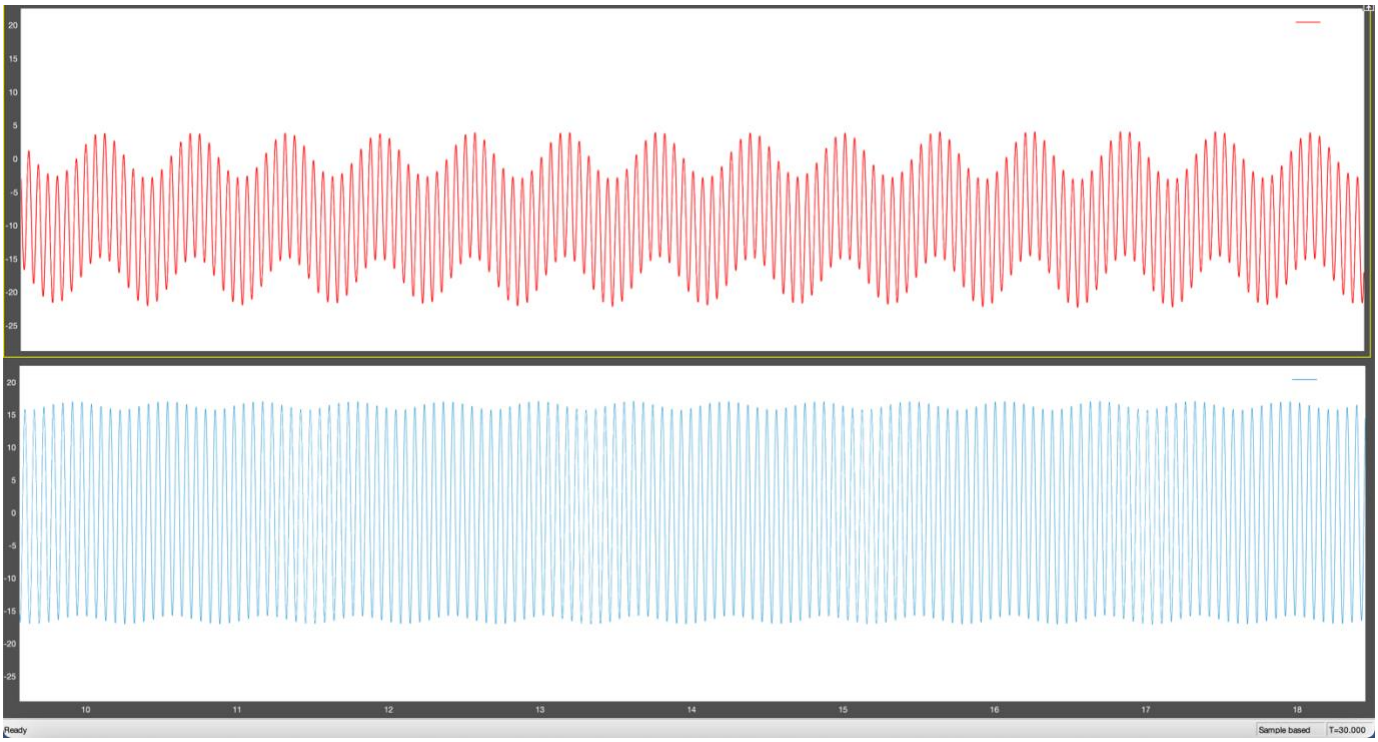


Ilustración 33-Grafica de Posición y velocidad

En la ilustración 33 se puede observar la grafica de la posición del robot uniciclo. Se puede observar como el robot uniciclo a pesar de sufrir oscilación siempre es capaz de volver a su posición original y es capaz de mantenerse de pie de manera satisfactoria. En la grafica inferior se pude observar la velocidad en rad/s, se pude observar como cambia la velocidad en ambos sentidos para generar el par en la dirección contraria a la dirección a que esta cayendo el motor y así evitar su caída y lograr mantener el robot en posición vertical.

En la ilustración 34 se puede observar la acción de control que se le proporciona al par de revolución para lograr mantener el equilibrio del robot.

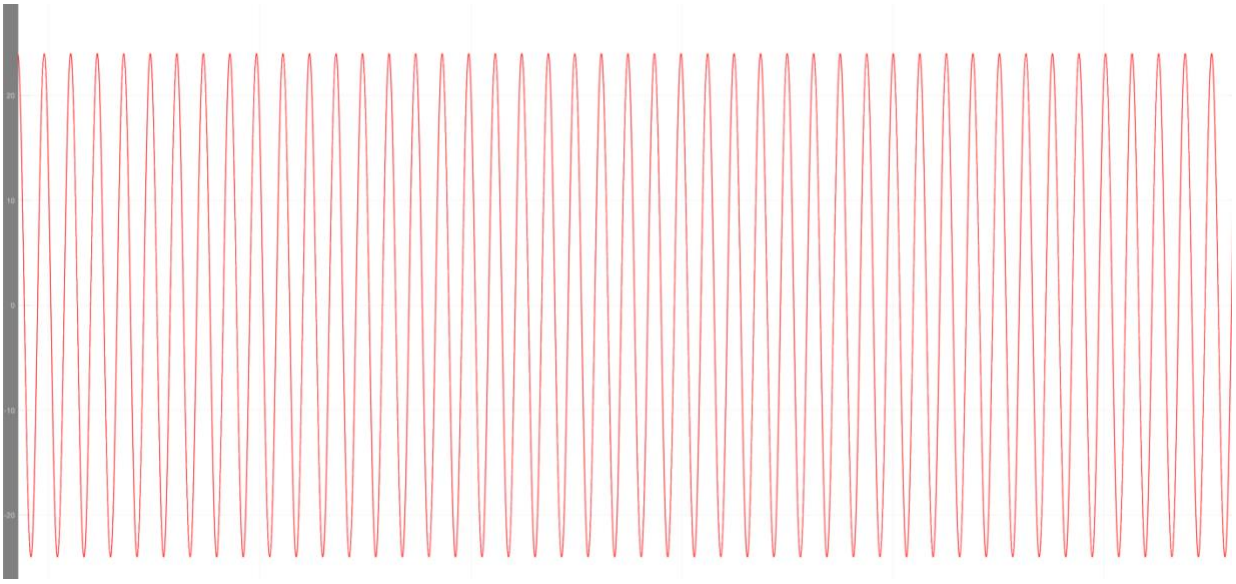


Ilustración 34-Accion de Control

6.3 IMPLEMENTACIÓN REAL

Una vez que se realizo la investigación previa de los precedentes previos del robot uniciclo, se realizo el diseño en 3D de las piezas necesarias para la construcción del prototipo del robot uniciclo. Luego se realizo la simulación por medio de Matlab Simscape del comportamiento del robot Uniciclo y se opto por el diseño del controlador necesario y se verifico que cumplía con la especificaciones de diseño solicitadas se procedió finalmente la realización del prototipo final con los componentes electrónicos comprados.

6.3.1 COMPONENTES ELECTRÓNICOS EMPLEADOS

A continuación se brinda una breve lista de los componentes electrónicos necesarios para la realización del prototipo del robot uniciclo junto con su descripción y especificaciones y una explicación de acerca porque se seleccionaron los siguientes componentes.

6.3.1.1 Acelerómetro y Giroscopio

Para la realización de este proyecto es muy importante conocer la posición y orientación del robot en todo momento, por esto es necesario contar con un IMU (unidad de medición inercial) que cuente con acelerómetro y giroscopio en sus tres ejes. Con esto en cuenta se selecciono el MPU6050.

Ilustración 35-MPU6050

El MPU6050 es una unidad de medición inercial que cuenta con un acelerómetro de tres ejes y un giroscopio de tres ejes. Este sensor tiene las siguientes características:

- Alimentación de 3.3-5V.
- Comunicación a través de I2C
- Librerías ya creadas en arduino para facilitar su uso.
- 6 grados de libertad.
- Conversor análogo digital de 16 bits.

6.3.1.2 Motor Eléctrico

En vista de lo visto en marco teórico finalmente se opto por utilizar motores brushless. Este tipo de motores son muy útiles en proyectos de electrónica y son compatibles con los encoders que se utilizaran en este proyecto.

Motor Brushless Gimball 2804T

Se opto por este motor debido a que es un motor de bajo costo y alto rendimiento. Este tipo de motor es muy utilizado en aplicaciones de video y de drones. Este motor permite alcanzar altas velocidades y proporciona un gran par a pesar de su tamaño (Iflight, 2022).

Este motor tiene las siguientes características:

- Resistencia: 10 Ohmios
- Peso: 42 gramos
- KV: 145KV
- Velocidad sin carga: 1740 RPM
- Voltaje: 12V
- Configuración de polos: 14N12P
- Distancia entre agujeros: 19mm y 12mm
- Diámetro del eje: 6.5mm
- Corriente: 0.8 A
- Diámetro: 35mm
- Cableado: 24 AWG

Ilustración 36-Motor Brushless GM2804

6.3.1.3 Encoder

Para la realización de este proyecto se necesita de un encoder para poder realizar el control de los motores en lazo cerrado. Se necesita conocer la posición de rueda inferior y con este valor ajustar la velocidad de la rueda de reacción. Los encoder irán acoplados a ambos motores para poder leer su posición en todo momento.

Finalmente se opto por hacer uso de un encoder magnético. En este caso se ocupa colocar un imán en el eje del motor que el encoder sea capaz de leer la posición del motor mediante la posición del campo magnético del imán. Un encoder magnético tiene de ventaja que no ocupa estar en contacto directo con el motor, pero es muy importante que la colocación de los imanes sea la correcta.

Encoder AS5600

Este tipo de encoder es un encoder absoluto, es un encoder muy utilizado en proyecto de electrónica con Arduino. Este encoder es de tipo magnético.

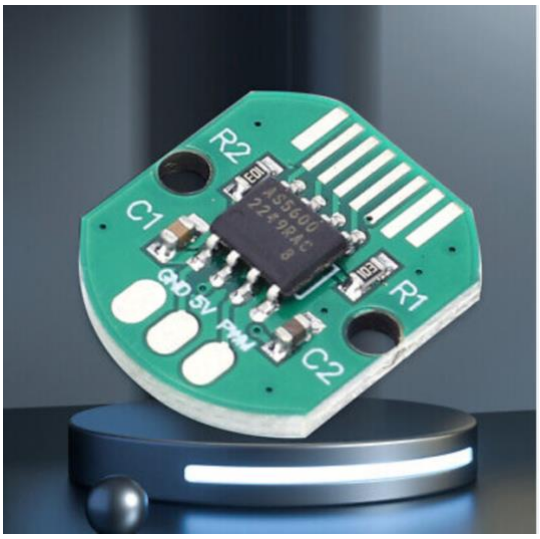


Ilustración 37-Encoder AS5600

Las características mas importantes de este encoder son las siguientes:

- Resolución de 12 Bits
- Encoder sin contacto
- Encoder de tipo absoluto
- Su lectura puede ser a través de un puerto analógico del microprocesador PWM o de la interfaz I2C para extraer la posición.
- Cuenta con una librería propia en Arduino.
- Fácil programación
- Distancia de detección: 0.5-3mm.
- Alimentación: 5V

6.3.1.4 Controlador

El controlador es uno de los componentes mas importantes en la realización de este proyecto. El controlador debe de cumplir con los siguientes requisitos:

- Debe contar con un procesador que permita controlar los motores con suficiente velocidad.
- Se debe ser capaz de programar en C
- Debe contener los drivers para controlar los motores brushless.
- Debe ser capaz de controlar dos motores a la vez.
- Se deben poder conectar el MPU6050 y ambos encoders del motor.
- Debe ser capaz de ser alimentado por una batería de 12V para hacer del robot totalmente inalámbrico.

6.3.1.5 Controlador Gimball 3.1 con Atmega 328P

Con estos criterios en mente y con la previa selección de motores brushless tipo gimbal se opto por seleccionar el controlar Gimball 3.1 con procesador Atmega328P.

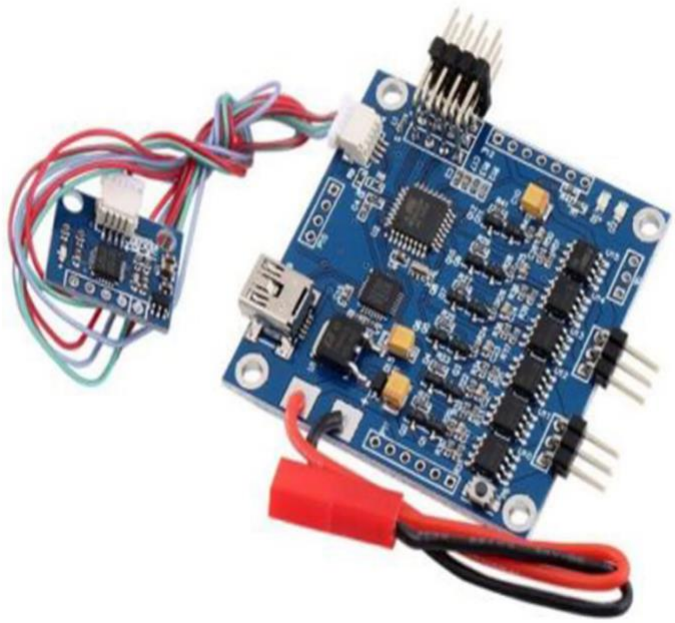


Ilustración 38-Controlador Gimball 3.1 con Atmega 328

Este controlador contiene todos los elementos necesarios para la realización de este proyecto, algunas de las características mas importantes de este controlador son las siguientes:

- Alimentación entre 10-12V.
- Dispone de los drivers para conectar dos motores brushless a la vez.
- Dispone de un procesador Atmega328.
- Cuenta con las salidas analógicas para poder conectar los encoders de los motores.
- Incluye el sensor MPU6050.
- Corriente máxima: 10 A.
- Al tener el procesador Atmega 328 permite la programación a través del Arduino Ide.
- Puerto de conexión mini USB.

Al seleccionar este controlador se hace que el diseño del robot sea mucho mas compacto y se reduce el uso de componentes electrónicos adicionales. Si se hubiera seleccionado un Arduino se hubiera tenido que incluir por aparte un driver de motores brushless y las conexiones hubieran sido un poco mas complicadas al intentar colocar todos los componentes en el robot.

6.3.1.6 Batería

Para que el robot pudiera ser totalmente inalámbrico se opto por incorporar en el diseño final una batería para la alimentación del controlador de los motores brushless. Finalmente se opto por una batería LiPo de 1500mah. Esta batería es capaz de alimentar al robot por un periodo de tiempo mas que suficiente y proporciona la corriente adecuada para que los motores puedan trabajar correctamente. Además de esto es una batería ligera que se puede recargar fácilmente.

Ilustración 39-Bateria de LiPo de 1500mah

6.3.1.7 Modulo Bluetooth

Finalmente se opto por incorporar en el robot de un modulo Bluetooth para ser capaces de cambiar los parámetros de los controladores de los motores del robot de manera remota. El modulo bluetooth que se selección es el modulo bluetooth HC-06.

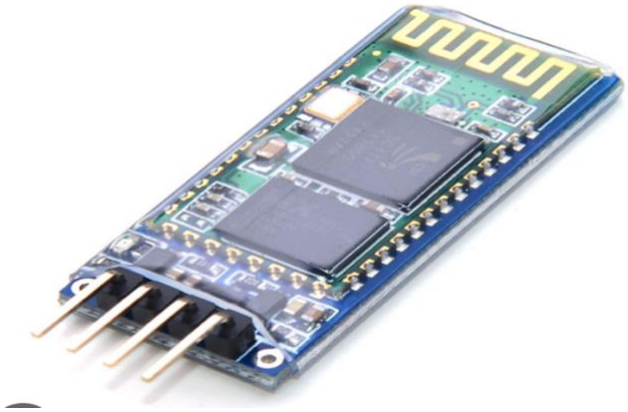


Ilustración 40-Modulo Bluetooth HC06

6.3.2 MONTAJE FINAL DEL PROTOTIPO DEL ROBOT UNICICLO

Una vez que ya se realizo el diseño en 3D y se realizaron las distintas simulaciones se procedió a comprar componentes electrónicos que se utilizaron para la elaboración del prototipo final. Una vez que ya se

definió el diseño final de igual manera se envió a imprimir a la Universidad las piezas en 3D para la elaboración del prototipo final.

Previo a realizar el montaje final del robot uniciclo en primer lugar se hicieron pruebas con los componentes electrónicos para comprobar su funcionamiento y para poder familiarizarse con su funcionamiento. Se comprobó que ambos motores funcionasen de manera correcta y se realizaron pruebas sencillas de cada motor y también de los encoders.

En la siguiente imagen se pueden observar los componentes electrónicos y las piezas impresas en 3D previas al montaje del robot.

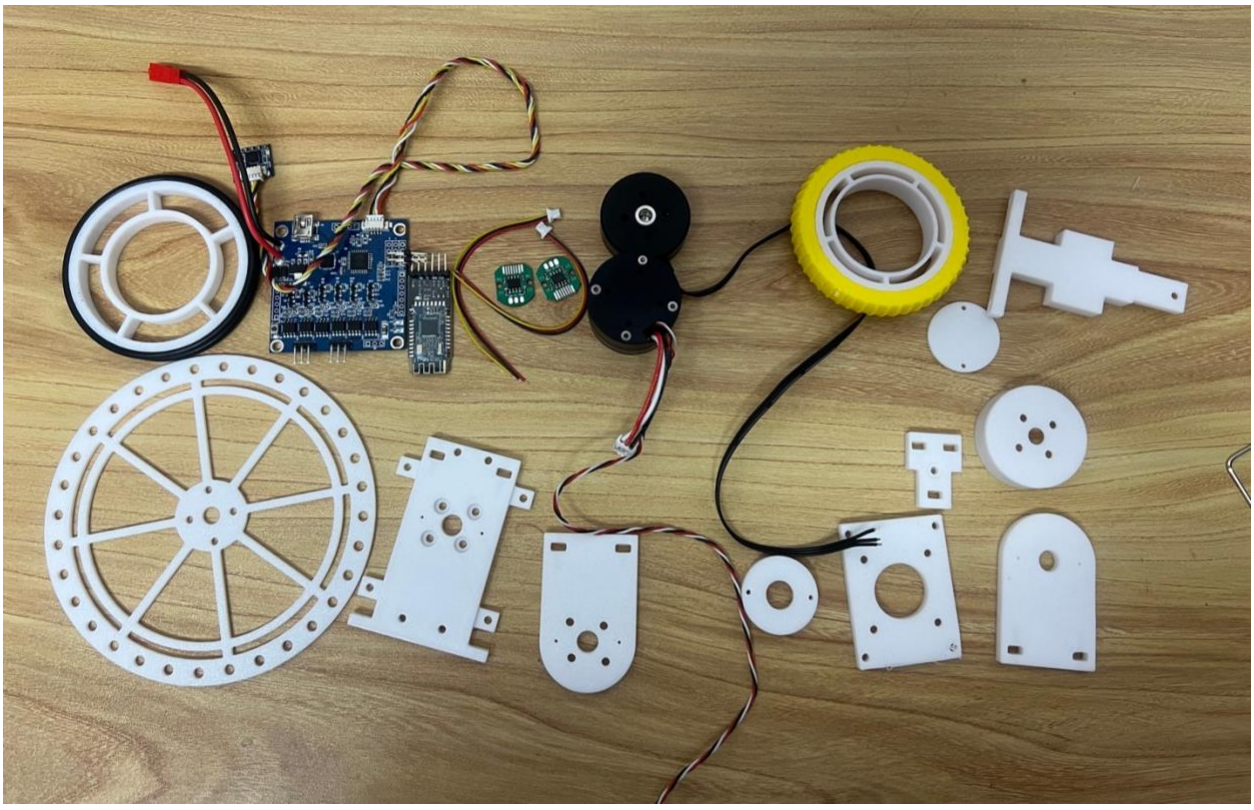


Ilustración 41-Componentes previos al Montaje

Para realizar el montaje final se ocuparon de distintos componentes adicionales tales como: multímetro, crimpadora de cables, soldadura con estaño, terminales, cables Dupont y elementos de tornillería varios.

A continuación se dará un breve resumen de los pasos que se siguieron para la implementación del robot uniciclo:

- Comprobar el correcto funcionamiento de los motores eléctricos.
- Realizar pruebas de la lectura de la posición de los motores por medio de los encoder AS5600.
- Familiarizarse con el MPU5060 para obtener las medidas de los 3 ejes del acelerómetro y del giroscopio.
- Investigar acerca de distintas formas de control de motores brushless.
- Familiarizarse con la librería de arduino SimpleFoc. Es una librería que mediante el control de voltaje de los motores brushless logra implementar algoritmos de control con mucha mas facilidad.
- Proceder a imprimir las piezas en 3D previamente realizadas con Solidworks.
- Proceder a montar todas las piezas en 3D junto con los componentes electrónicos con ayuda de la tornillería y demás herramientas.
- Realizar prueba con el robot para poder calibrar adecuadamente los motores eléctricos y los valores de los offset del MPU6050 ya montado en el robot.
- Implementar el código para el control del equilibrio del robot uniciclo.
- Realizar pruebas para encontrar los parámetros de los controladores de los distintos ejes del robot.
- Realizar pruebas finales y últimos ajustes.

A continuación se muestran imágenes del montaje final del robot uniciclo con las piezas impresas en 3D y con los componentes electrónicos.

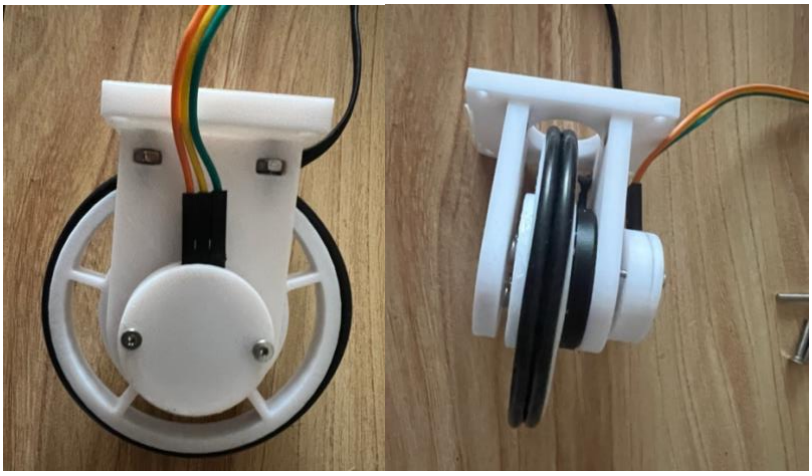


Ilustración 42-Montaje de Rueda Inferior

En la ilustración 43 se puede observar el montaje de la rueda inferior del robot uniciclo. Esta es la parte inferior del robot uniciclo, se coloco el motor Brushless 2804 acoplado directamente a la rueda del robot unido con tornillería. Además lleva en el lateral el encoder AS5600 para poder leer la posición y velocidad del robot en todo momento y para poder ajustarla de acuerdo al control en bucle cerrado.

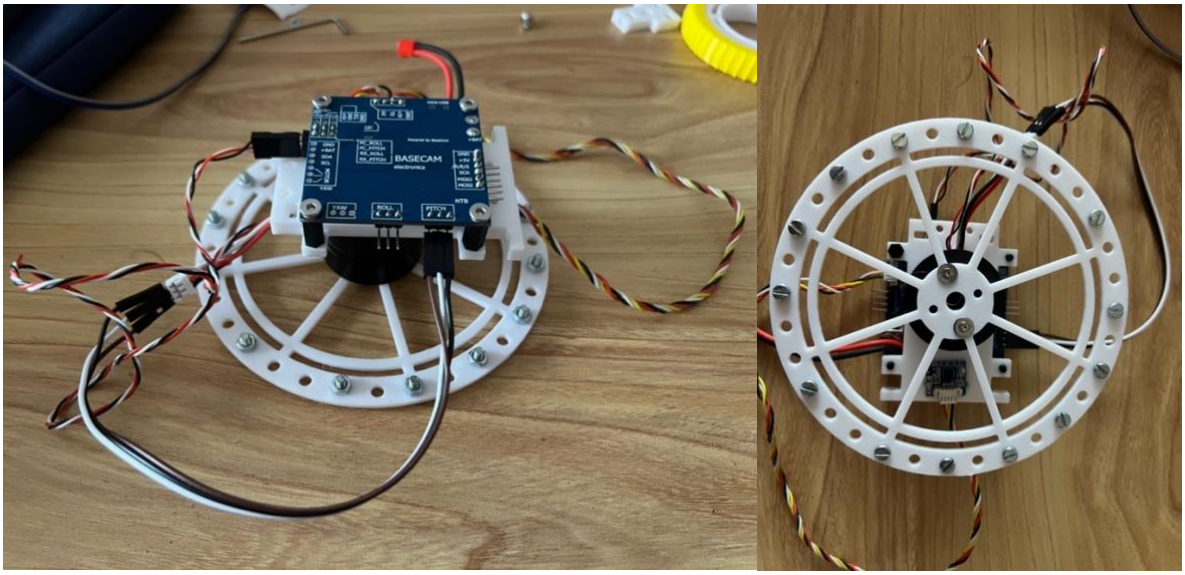


Ilustración 43-Montaje Superior del Robot Uniciclo

En la ilustración 43 se puede observar la parte mas importante del montaje del robot uniciclo. Se observa la rueda de reacción atornillado a uno de los motores brushless 2804. Este motor ya lleva incorporado el encoder en la carcasa del motor. También se observa en la parte frontal el MPU5060 va fijado en esta

parte para conocer la orientación del robot en todo momento. Finalmente en la parte trasera se observa en controlador de motores gimball 3.1 que se coloco con ayuda de unos espaciadores M3 para que los pines del controlador tuvieran espacio y quedaran libre para conectarse con facilidad.

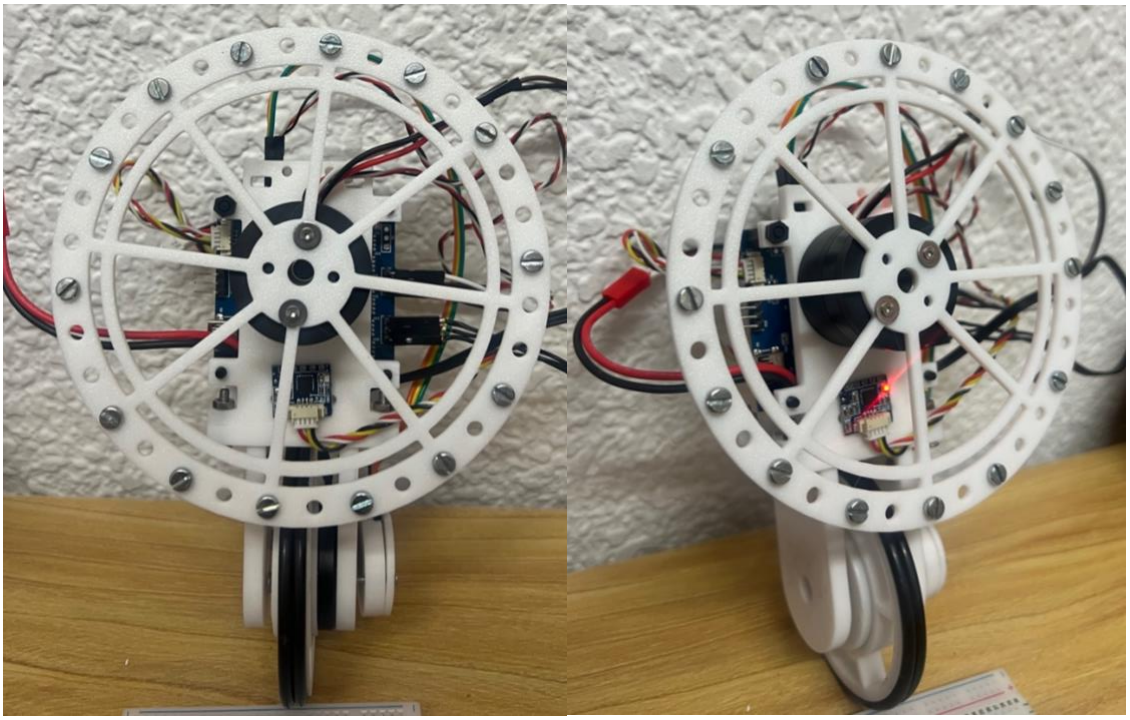


Ilustración 44-Montaje Final de Robot Uniciclo

6.3.3 PROGRAMACIÓN DEL ROBOT UNICICLO

Para la elaboración del código del robot se contaba con la ventaja de que el controlador cuenta con procesador Atmega328P por lo tanto es posible realizar el código con Arduino Ide y es posible utilizar todas las librerías que cuenta arduino. En este caso en particular luego de investigar acerca de distintos métodos de control de motores brushless se descubrió la librería Simple Foc. Este proyecto se hizo principalmente con la ayuda de esta librería para controlar los motores brushless con una mayor facilidad.

La librería simplefoc permite un control mas preciso de motores brushless por medio de algoritmos del control FOC. Esta librería utiliza el control de voltaje de los motores en lugar del control de torque del control FOC tradicional. Para aplicaciones de electrónica es ideal para el control de motores brushless ya que permite un control mas preciso y suave proporcionando un par mayor con un mayor eficiencia energética.

A continuación se describirán las partes mas relevantes del código que se hizo para este proyecto, el resto del código se puede ver en los anexos en el final de este informa.

```
///Declaracion de pines de motores y encoders
BLDCMotor motor_EjeX = BLDCMotor(3, 5, 6, 7);
BLDCMotor motor_EjeY = BLDCMotor(9, 10, 11, 7);
MagneticSensorAnalog sensor_EjeX = MagneticSensorAnalog(A1, 14, 1020);
MagneticSensorAnalog sensor_EjeY = MagneticSensorAnalog(A0, 14, 1020);
```

Se declaran los pines en los que colocaran los motores de cada eje y también los pines de los encoder así mismo se coloca la resolución de los encoder, se puede observar que ya se están utilizando funciones de las librería SimpleFoc.

```
sensor_EjeX.init();                                          // inicializa el encoder
motor_EjeX.linkSensor(&sensor_EjeX);                         // asocia el motor a su respectivo encoder
motor_EjeX.voltage_power_supply = 12;
motor_EjeX.controller = ControlType::voltage;                // Cotrol por voltaje
motor_EjeX.voltage_sensor_align = 4;
motor_EjeX.foc_modulation = FOCModulationType::SpaceVectorPWM;  // Modulacion FOC
motor_EjeX.init();                                           // inicializa el motor
motor_EjeX.initFOC();                                        // alinea el encoder e inicia FOC
```

En esta parte del código se inicializan tanto el encoder como el motor para ser utilizados con la librería simple foc. Se establece el voltaje que se le proporciona al motor y se define el tipo de control que se utilizara que en este caso es por voltaje. También se le dice que se generaran señales PWM para el control del motor y finalmente se alinea el motor con el encoder. Este proceso se realiza cada vez que se energiza el controlador. Esto se repite para el otro motor.

```
if (abs(robot_angleX - angle_offsetX) > 5 || abs(robot_angleY - angle_offsetY) > 5) vertical = false;
if (abs(robot_angleX - angle_offsetX) < 0.2 && abs(robot_angleY - angle_offsetY) < 0.2 && !vertical) vertical = true;

gyroX = GyZ / 131.0; // Convierte a deg/s
gyroY = GyY / 131.0;

filtro_EjeX = alphaX * gyroX + (1 - alphaX) * filtro_EjeX;
filtro_EjeY = alphaY * gyroY + (1 - alphaY) * filtro_EjeY;
```

En esta parte se determina si el robot se encuentra en posición vertical o no. Si la diferencia en la lectura del giroscopio es mayor a 5 grados se determina que el motor no esta vertical y si la diferencia se encuentra menor a 0.2 grados se determina que esta en posición vertical y se procede a aplicar la acción de control que busca que el robot se mantenga en equilibrio. Finalmente se le aplica un filtro pasa baja a la lectura de MPU5060 para disminuir el ruido de la lectura del sensor.

```
int sign(int x) {
  if (x > 0) return 1;
  if (x < 0) return -1;
  if (x = 0) return 0;
}
//Se calculan las acciones de control de ambos ejes

float acc_ControlX(float error_angulo, float filtro, float m_vel) {
  float u =  K1X * error_angulo + K2X * filtro + K3X * m_vel;
  if (abs(u) > motor_EjeX.voltage_power_supply * 0.8)
    u = sign(u) * motor_EjeX.voltage_power_supply * 0.8;
  return u;
}

float acc_ControlY(float error_angulo, float filtro, float m_vel) {
  float u =  K1Y * error_angulo + K2Y * filtro + K3Y * -m_vel;
  if (abs(u) > motor_EjeY.voltage_power_supply * 0.8)
    u = sign(u) * motor_EjeY.voltage_power_supply * 0.8;
  return u;
}
```

En esta parte es donde se calculan las acciones de control de ambos ejes. En este caso se opto por un controlador tipo PD con un filtro. Se calcula la diferencia entre los ángulos del robot y se multiplica por una constante K1, también se multiplica la velocidad del motor por una constante K3 y se le aplica un filtro. Se utiliza la función sign para determinar que la acción de control se encuentra dentro de un margen seguro para proteger el motor. De igual manera se opera con el voltaje del motor al 80% por motivos de seguridad del motor.

```
if (test) {
  motor_EjeX.move(4 * (motor_EjeY.shaft_angle - motor_EjeX.shaft_angle));
  motor_EjeY.move(4 * (motor_EjeX.shaft_angle - motor_EjeY.shaft_angle));
} else if (cuenta_bucle++ > tiempo_bucle) {
  float voltajefinal_EjeX = 0, voltajefinal_EjeY = 0;

  if (vertical) {
    voltajefinal_EjeX = acc_ControlX(robot_angleX - angle_offsetX, filtro_EjeX, sensor_EjeX.getVelocity() + vel_motorX);
    vel_motorX += sensor_EjeX.getVelocity() / 50;
    voltajefinal_EjeY = acc_ControlY(robot_angleY - angle_offsetY, filtro_EjeY, sensor_EjeY.getVelocity() + vel_motorY);
    vel_motorY += sensor_EjeY.getVelocity() / 40;

    motor_EjeX.move(voltajefinal_EjeX);
    motor_EjeY.move(-voltajefinal_EjeY);
  } else {
    motor_EjeX.move(voltajefinal_EjeX);
    motor_EjeY.move(voltajefinal_EjeY);
    vel_motorX = 0;
    vel_motorY = 0;
  }
  cuenta_bucle = 0;
}
```

En esta parte dentro del loop se da la opción de realizar el test, con la función test sirve para determinar que los motores están funcionando de manera correcta. Si el robot se encuentra en posición vertical en esta parte se inicia con el algoritmo de control implementado. Se determina el voltaje final de cada eje empleando la función de control previamente descrita. El primer componente de la función es la diferencia de los ángulos del robot, el segundo termino es el filtro pasa baja de los valores de lectura del giroscopio y el ultimo termino es el de la velocidad del motor en ese momento.

6.3.4 Problemas con la Implementación Real

Finalmente se tuvieron problemas con la implementación real durante las pruebas de los ajustes de los valores de las constantes de los controladores. Al momento de realizar las pruebas finales para intentar mantener el dispositivo el controlador se sobrecalentó y algunos de los drivers de los motores dejaron de funcionar de manera correcta. Al ser un procesador de bajo costo que no esta específicamente diseñado para ser programado mediante el arduino ide probablemente no cuenta con las protección adecuadas y eso provoco un sobrecalentamiento que provoco la avería de los drivers de los motores. Debido a esto no se pudo completar de manera satisfactoria la implementación real. En el caso de realizar la continuación de este proyecto seria mas recomendable buscar hacerlo con un controlador mas robusto debido a las pruebas que se deben hacer para poder mantener el robot en su posición vertical. Además se tiene la dificultad que se tiene para poder subir el código en este controlador debido a que de fabrica no se puede

programar de manera directa vía su puerto usb y es necesario quemar el programa ya sea a través de un usbasp o desde un bootloader.

VII. PLIEGO DE CONDICIONES

7.1 OBJETO

El diseño del prototipo y la implementación del robot uniciclo es un proyecto de tipo académico que se realizo con el objetivo de la realización del TFM del master en Ingeniería Mecatrónica. El proyecto consta de diferentes etapas, entre ellas se encuentra: elaboración y simulación del diseño 3D del robot uniciclo por medio de solidworks, implementación y simulación del sistema físico utilizando Matlab Simscape para simular el funcionamiento físico y finalmente la implementación real del robot uniciclo.

7.2 CONDICIONES DE LOS MATERIALES

7.2.1 DESCRIPCIÓN

Los materiales que se utilizaran para la elaboración de este proyecto deben ser componentes que cumplan con criterios de seguridad, los materiales deben ser capaz de soportar las cargas y de implementar un diseño seguro y funcional. Los componentes electrónicos deben de ser componentes que cumplan con criterios de seguridad y normativas vigentes y deben de ser compatibles entre ellos mismos para implementar el prototipo del robot.

7.3 CONDICIONES DE LA EJECUCIÓN

7.3.1 DESCRIPCIÓN

Para poder realizar la ejecución de este proyecto se debe contar con los siguientes programas en un ordenador:

- Solidworks 2022 o superior
- Matlab R2022b o superior
- Arduino Ide

De igual manera se debe contar con los materiales previamente descritos para proceder a realizar la implementación real del robot.

7.4 PRUEBAS Y AJUSTES FINALES DE SERVICIOS O CONDICIONES

Las pruebas finales se deben realizar en distintas etapas: en primer lugar para realizar las pruebas del diseño 3D se debe realizar el ensamblaje de las piezas y se deben realizar las simulación estáticas correspondientes para verificar la seguridad del diseño propuesto. En la simulación del sistema físico utilizando Matlab Simscape se debe de utilizar esta herramienta para comprobar que el diseño propuesto es capaz de mantenerse en equilibrio de manera satisfactoria al simular el funcionamiento de los motores y sensores. La implementación real debe de realizarse el montaje de todas las piezas y de los componentes electrónicos comprados previamente, se debe realizar la simulación e implementar el funcionamiento del robot.

VIII. PRESUPUESTO

8.1 MATERIALES

Los siguientes materiales se requirieron para la elaboración de este proyecto:

- Tornillos M3
- Tornillos M2
- Tuercas M3
- Controlador Gimball 3.1 con procesador Atmega328P
- 2 Encoder AS5600
- IMU MPU6050
- Bateria LiPo de 1500mah
- 2 imanes radiales de 8mm
- Modulo Bluetooth HC06
- Piezas Impresas en 3D

A continuación, se adjunta la tabla con el presupuesto real para elaboración del proyecto y su costo final aproximado en Euros:

Tabla 3-Presupuesto Para Construcción del Robot Uniciclo

Cantidad	Material	Precio Unitario (Euros)	Coste Total (Euros)
35	Tornillos M3	0.08	2.8
25	Tuercas M3	0.06	1.5
12	Tornillos M2	0.12	1.44
1	Controlador Gimball 3.1	34	34
2	Encoder AS5600	5	10
1	MPU6050	6	6
2	Motor Gimball 2804	20	40
1	Batería LiPo de 1500mah	26	26
2	Imanes radiales de 8mm	0.8	1.6
1	Modulo Bluetooth HC06	11	11
1	Piezas Impresas en 3D	35	35
1	Materiales varios(Cableado, terminales)	10	10
Total en Euros			179.34

IX. CONCLUSIONES

9.1 CONCLUSIÓN GENERAL

- ✓ Se logró implementar el diseño del dispositivo uniciclo mediante el diseño 3D de componentes impresos en 3D junto a los componentes electrónicos comprados para realizar el montaje del robot uniciclo con un diseño inalámbrico.

9.2 CONCLUSIONES ESPECÍFICAS

- Se realizó un análisis previo en los diferentes diseños de robot uniciclo elaborados en el pasado, de estos modelos se obtuvieron las ideas para el diseño del robot uniciclo actual.
- Se realizó un análisis de los resultados de tensiones y deformaciones del robot uniciclo mediante el software SolidWorks observando que los resultados daban una estructura adecuada y segura para la realización de la implementación física.
- Se realizaron los planos de las diferentes piezas que conforman la construcción de este robot uniciclo para realizar su montaje.
- Se logro realizar la simulación del comportamiento físico del robot uniciclo con ayuda de Matlab Simscape, con esta herramienta se pudo comprobar que era posible que el robot de mantuviera en equilibrio simulando los motores y sensores del sistema, finalmente se implemento el algoritmo de control para implementar el balance del dispositivo.
- Se seleccionaron los componentes electrónicos para la implementación del robot uniciclo en base a las especificación del robot y la información teórica recopilada.
- Se logro implementar un prototipo físico del robot uniciclo con las piezas diseñadas previamente junto a los componentes comprados y posteriormente se realizo su programación para que el robot mantuviera su equilibrio.
- Finalmente no se logro obtener un prototipo funcional debido a problemas con el controlador del robot sin embargo se dejo la base para poder finalizar el proyecto en un futuro.

X. BIBLIOGRAFÍA

1. Atwell, C. (2022). *This Unicycle-Like Robot Can Autonomously Right Itself Upright with One Wheel*. Obtenido de Hacksterio.io: https://www.hackster.io/news/this-unicycle-like-robot-can-autonomously-right-itself-upright-with-one-wheel-3b3cbe7e9514
2. CochesRc. (2023). *MOTORES BRUSHLESS VS MOTORES DE ESCOBILLAS. CONOCE LAS DIFERENCIAS*. Obtenido de CochecitosRc: MOTORES BRUSHLESS VS MOTORES DE ESCOBILLAS. CONOCE LAS DIFERENCIAS
3. Cristina Rebollo, P. V. (2011). *Diseño de un Andador para Adultos.* Pamplona.
4. Ecured. (2017). *Diseño asistido por computadora* . Obtenido de https://www.ecured.cu/Diseño_asistido_por_computadora
5. EncoderProduct. (S.F.). *¿Qué es un Encoder?* Obtenido de Encoder Product: https://www.encoder.com/article-que-es-un-encoder
6. Gordillo, F. (2010). *El Péndulo Invertido: un Desafío para el Control No Lineal*. Obtenido de ResearchGate: El Péndulo Invertido: un Desafío para el Control No Lineal
7. Iflight. (2022). *GM2804 Gimbal Motor*. Obtenido de Iflight: https://shop.iflight.com/ipower-gm2804-gimbal-motor-pro1153
8. Industrial, T. (2019). *Tecnología-Industrial.es*. Obtenido de http://www.tecnologia-industrial.es/Resistencia%20de%20los%20materiales.htm
9. INFAIMON. (11 de Mayo de 2018). *INFAIMON*. Obtenido de https://blog.infaimon.com/coeficiente-de-seguridad-filosofia-diseno/
10. José Luis Guamushig Laica, M. A. (2018). *Diseño y construcción de un andador inteligente para el desplazamiento autónomo de los adultos mayores con visión reducida y problemas de movilidad del Hogar de Vida "Luis Maldonado Tamayo" mediante la investigación de técnicas de visión artificial.* Latacunga.

11. Juanpere, R. (2015). *Técnicas de control para motores Brushless*. Obtenido de ReserachGate: https://www.motronic.es/upfiles/taller_img/files/mantenimiento-y-reparacion-de-servomotores-es_2595.pdf

12. La Revista Informatica. (2019). *DISEÑO ASISTIDO POR COMPUTADORA* . Obtenido de http://www.larevistainformatica.com/DISENO-ASISTIDO-COMPUTADORA.HTML

13. Lachaize, A. (2017). *Which CAD software you should start with?* . Obtenido de https://blog.fydiz.com/which-cad-software-you-should-start-with-dfa184bbd616

14. Lee Jae Oh, H. I. (2012). *Fuzzy sliding mode control of unicycle robot*. Obtenido de IEEE Explore: Fuzzy sliding mode control of unicycle robot

15. Madera, R. (2018). *Design and Construction of a Prototype of a Gait Trainer.* Mayaguez, Puerto Rico.

16. Monje, C. (2011). *Metodología de Investigacion Cuantitativa y Cualitativa.* Obtenido de https://www.uv.mx/rmipe/files/2017/02/Guia-didactica-metodologia-de-la-investigacion.pdf

17. MSCI. (2013). *What is a Reaction Wheel?* Obtenido de Microsat System Canada: https://www.reactionwheel.ca/resources/reactionwheel.html

18. Nuri, A. (2013). *Inertial Measurement Units (IMUs): A Brief Summary of Key Concepts* . Obtenido de Nyamet Medium: https://nyahmet.medium.com/inertial-measurement-units-imus-a-brief-summary-of-key-concepts-b86e6dbf6458

19. PICUINO. (2024). *Controlador PID*. Obtenido de Picuino: https://www.picuino.com/es/control-pid.html

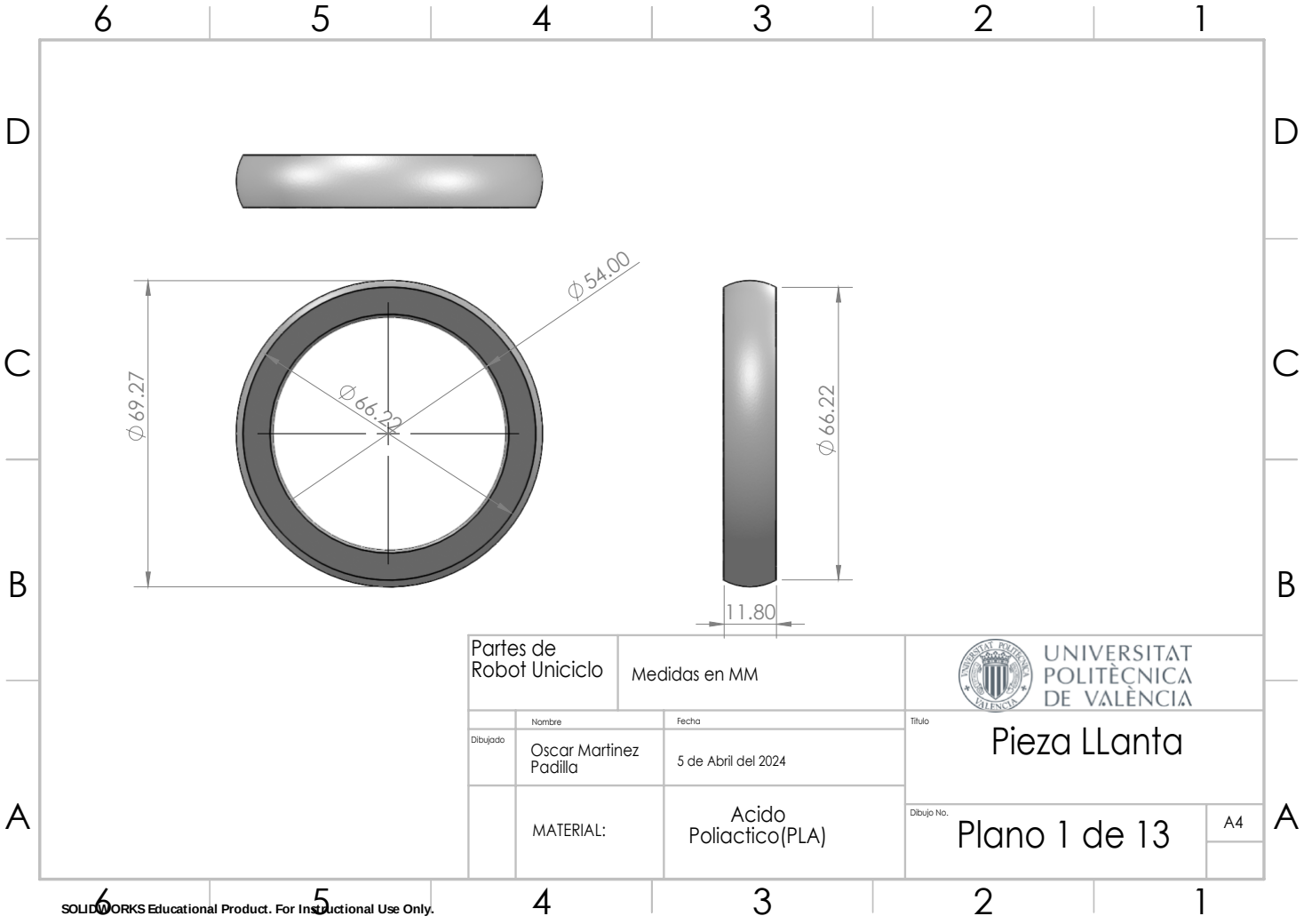
20. Ramírez, A., López, J., Manzanillo, Á., & Portillo, D. (1 de Febrero de 2013). *Monografías.* Obtenido de https://www.monografias.com/trabajos96/deformacion-y-resistencia-materiales/deformacion-y-resistencia-materiales.shtml

21. SolidWorks, 2. (2024). Diseño. Valencia, Valencia, España.

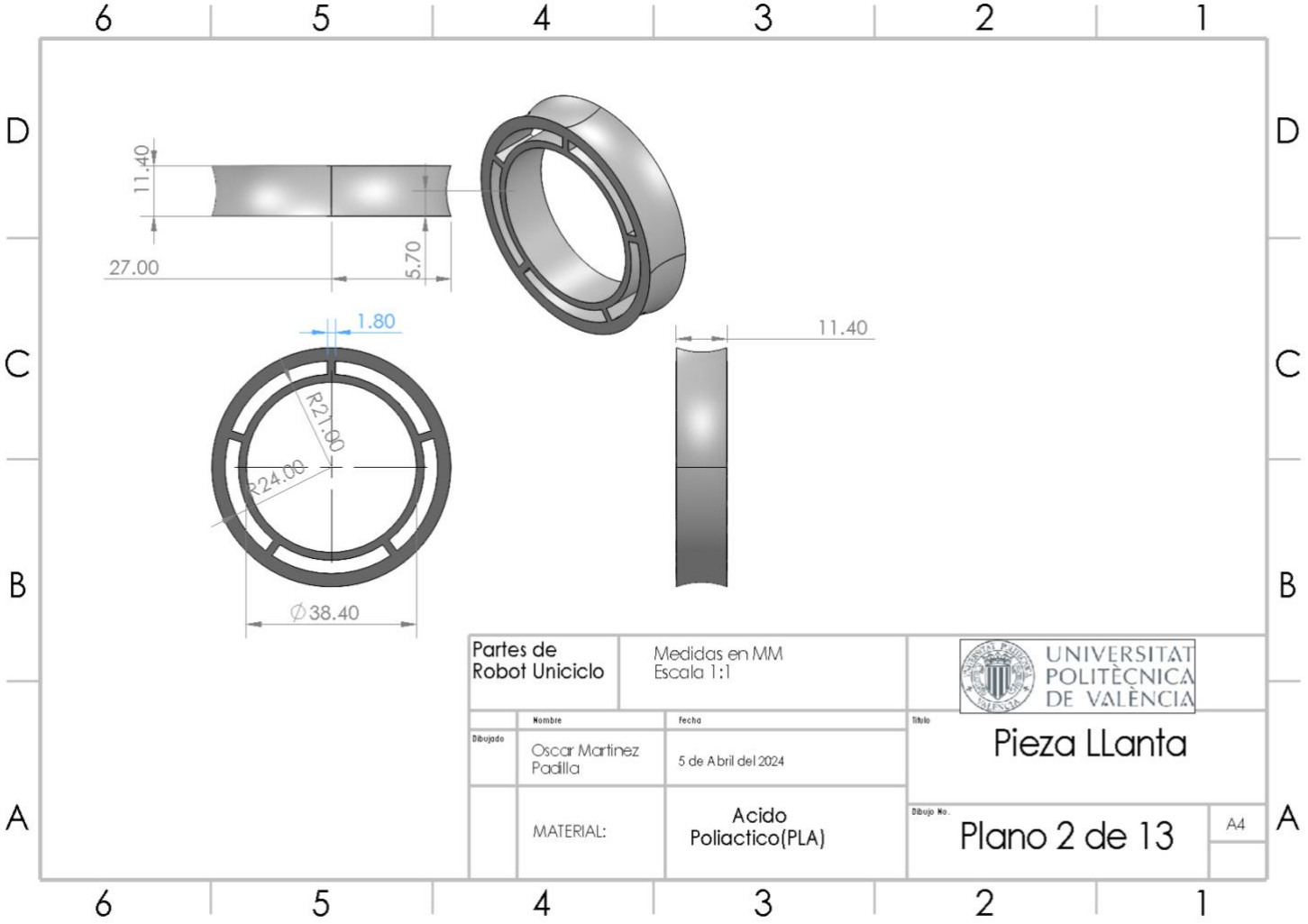
22. Torres, J. (2019). *Diseño asistido por ordenador*. Obtenido de https://lsi.ugr.es/~cad/teoria/Tema1/RESUMENTEMA1.PDF

23. Zhang, D. (S.F.). *Motores de CC sin escobillas frente a motores paso a paso*. Obtenido de https://www.omc-stepperonline.com/es/support/motores-de-cc-sin-escobillas-frente-a-motores-paso-a-paso

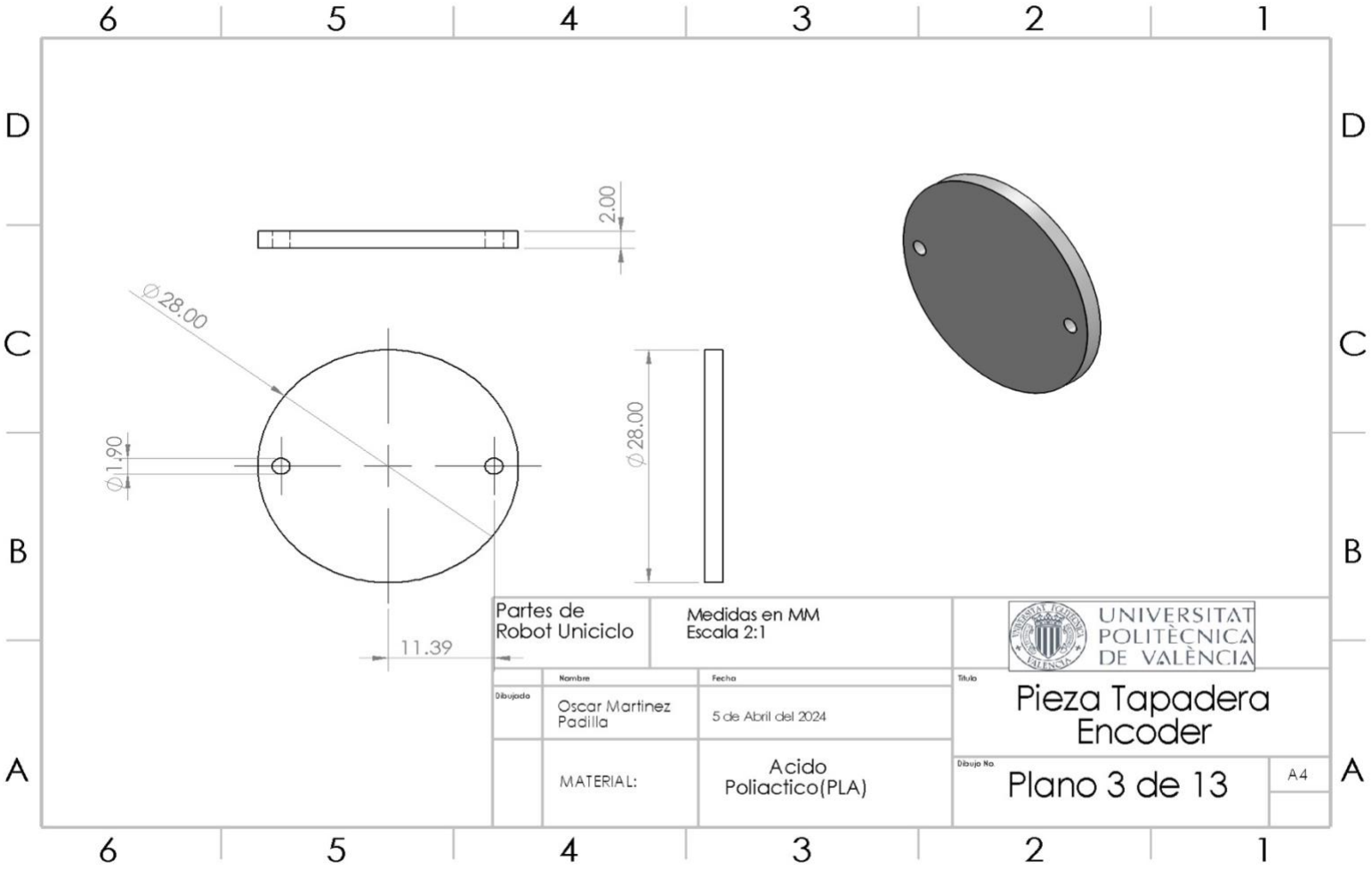
XI. ANEXOS



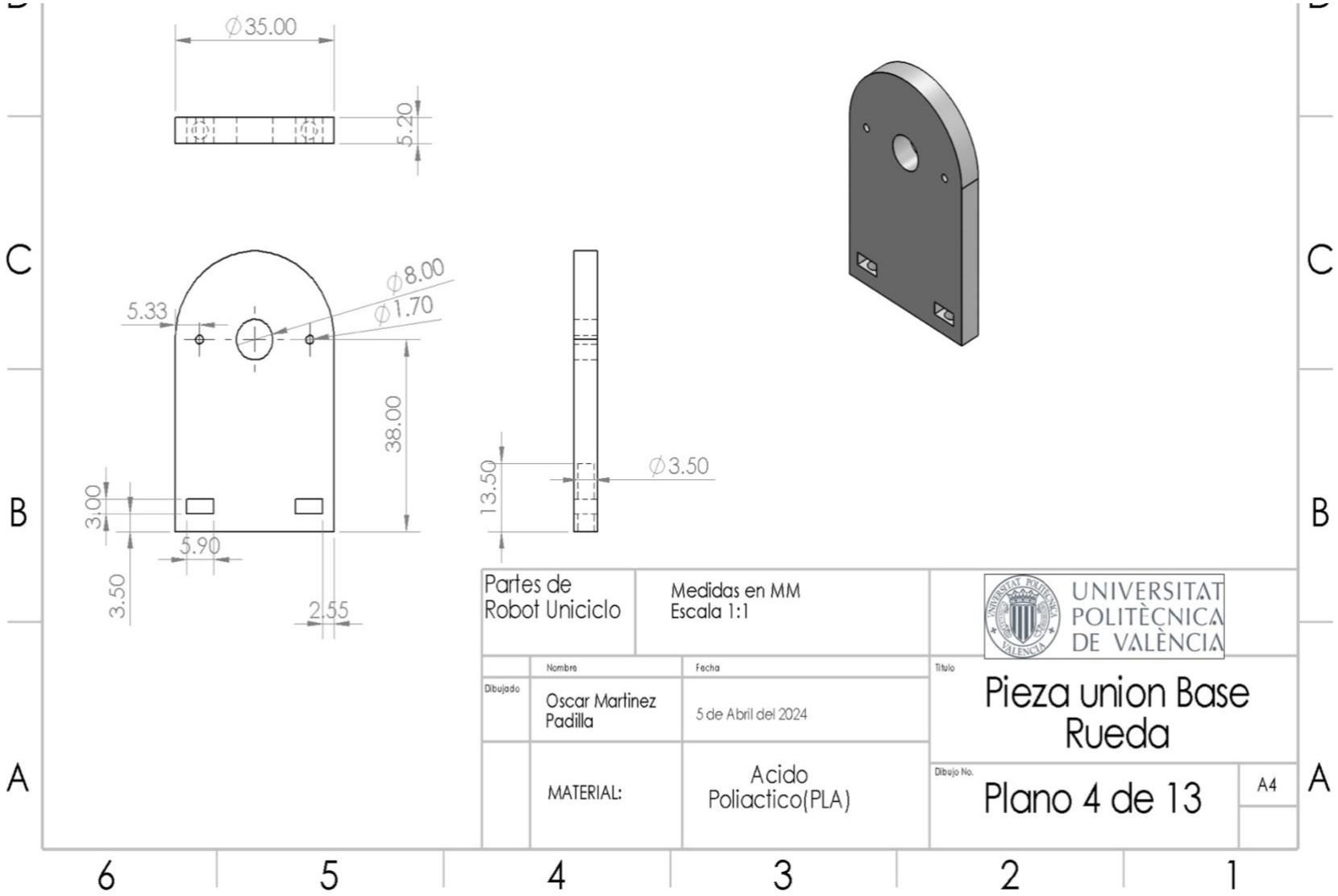
Anexo 1-Plano Pieza Llanta



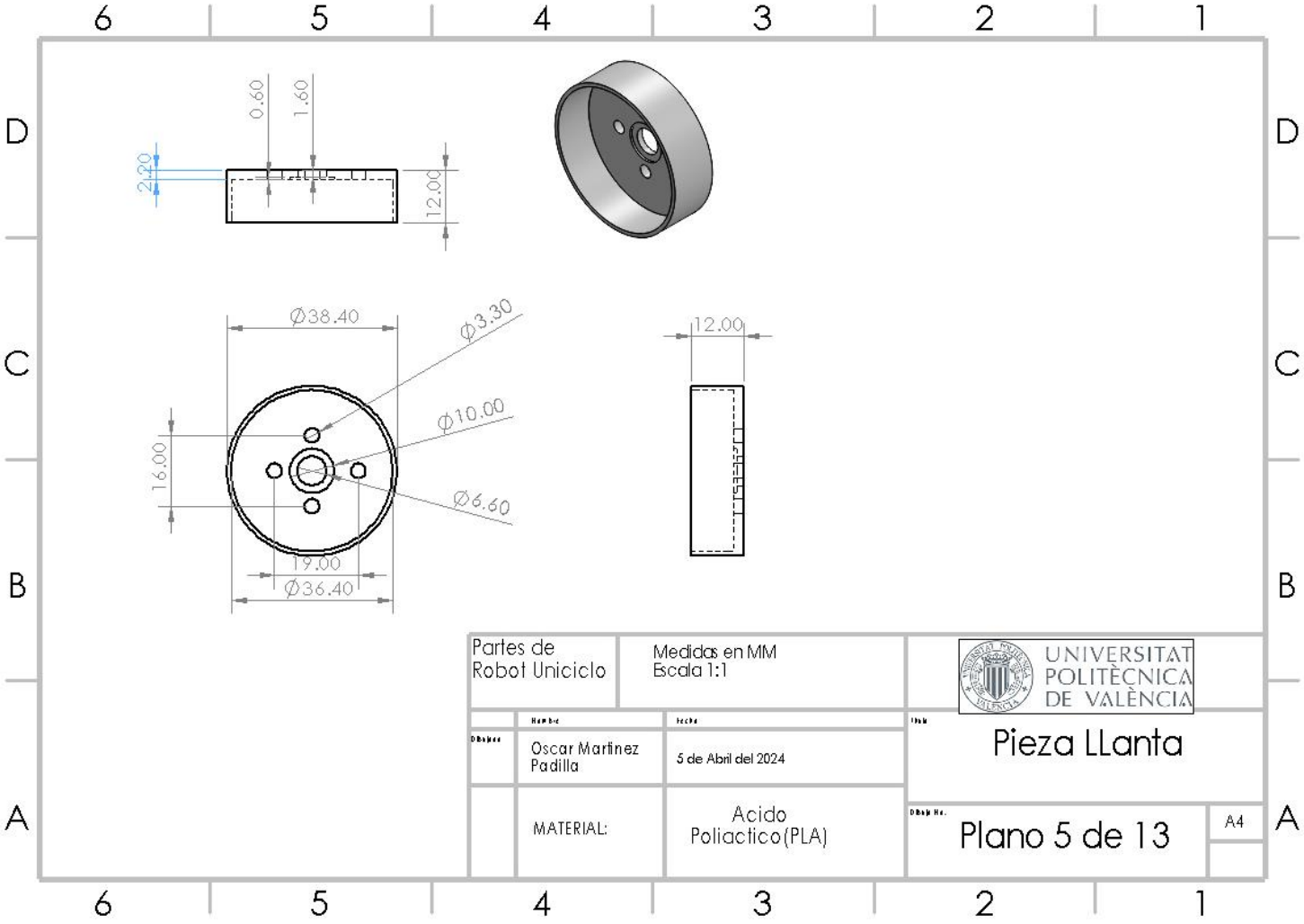
Anexo 2-Plano Pieza Rin Llanta



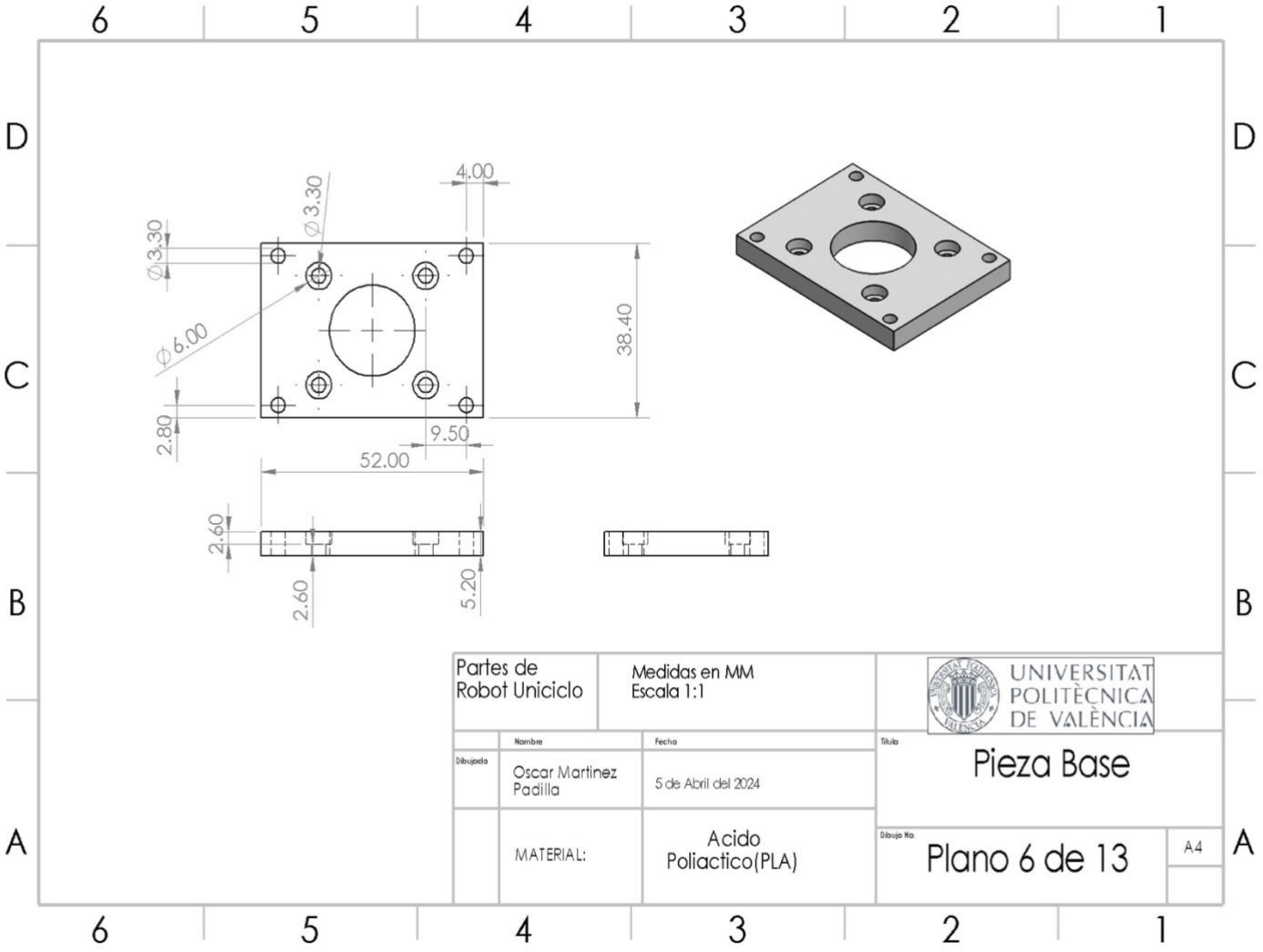
Anexo 3-Plano Pieza Tapadera Encoder



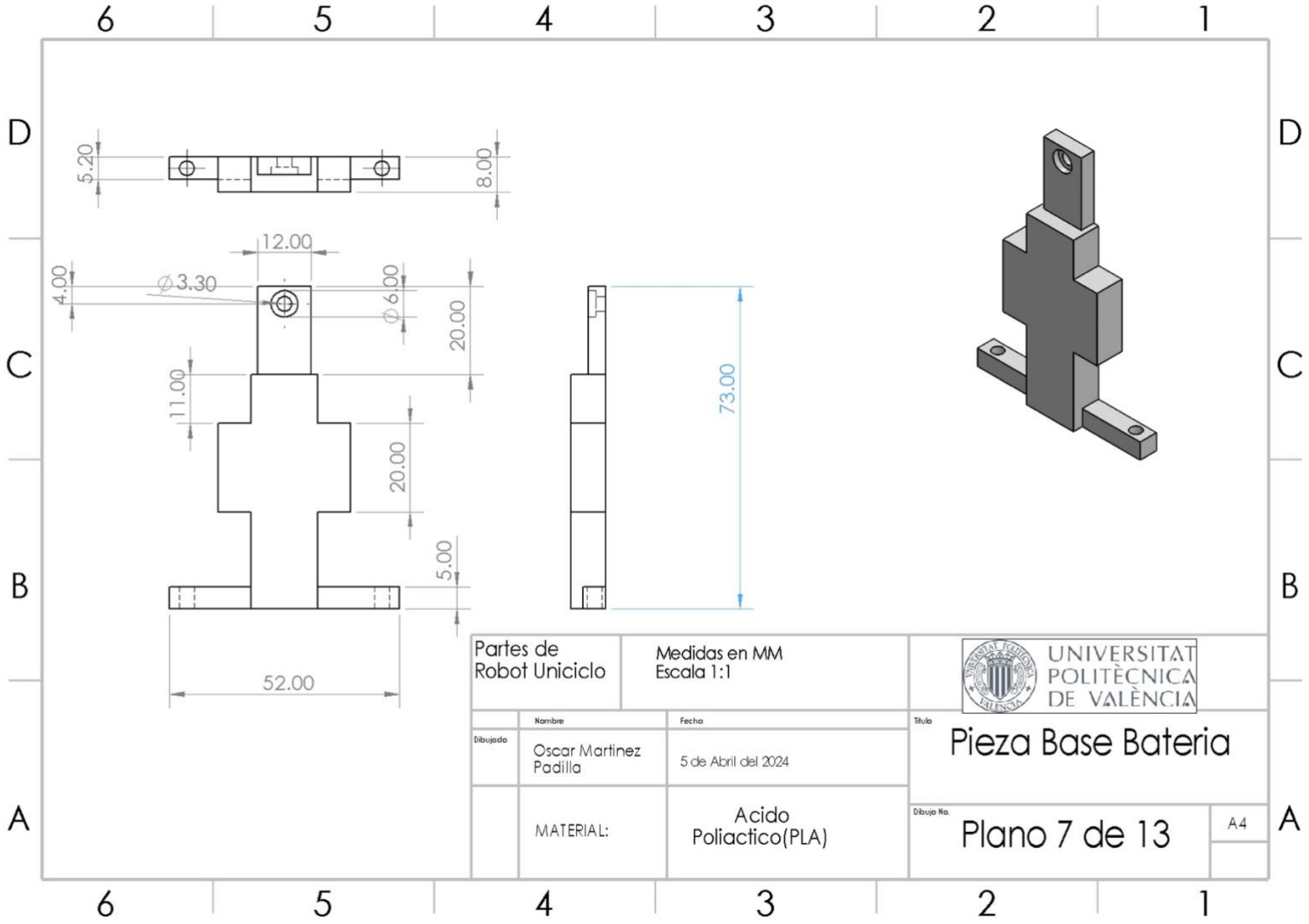
Anexo 4-Plano Pieza Unión Base Rueda



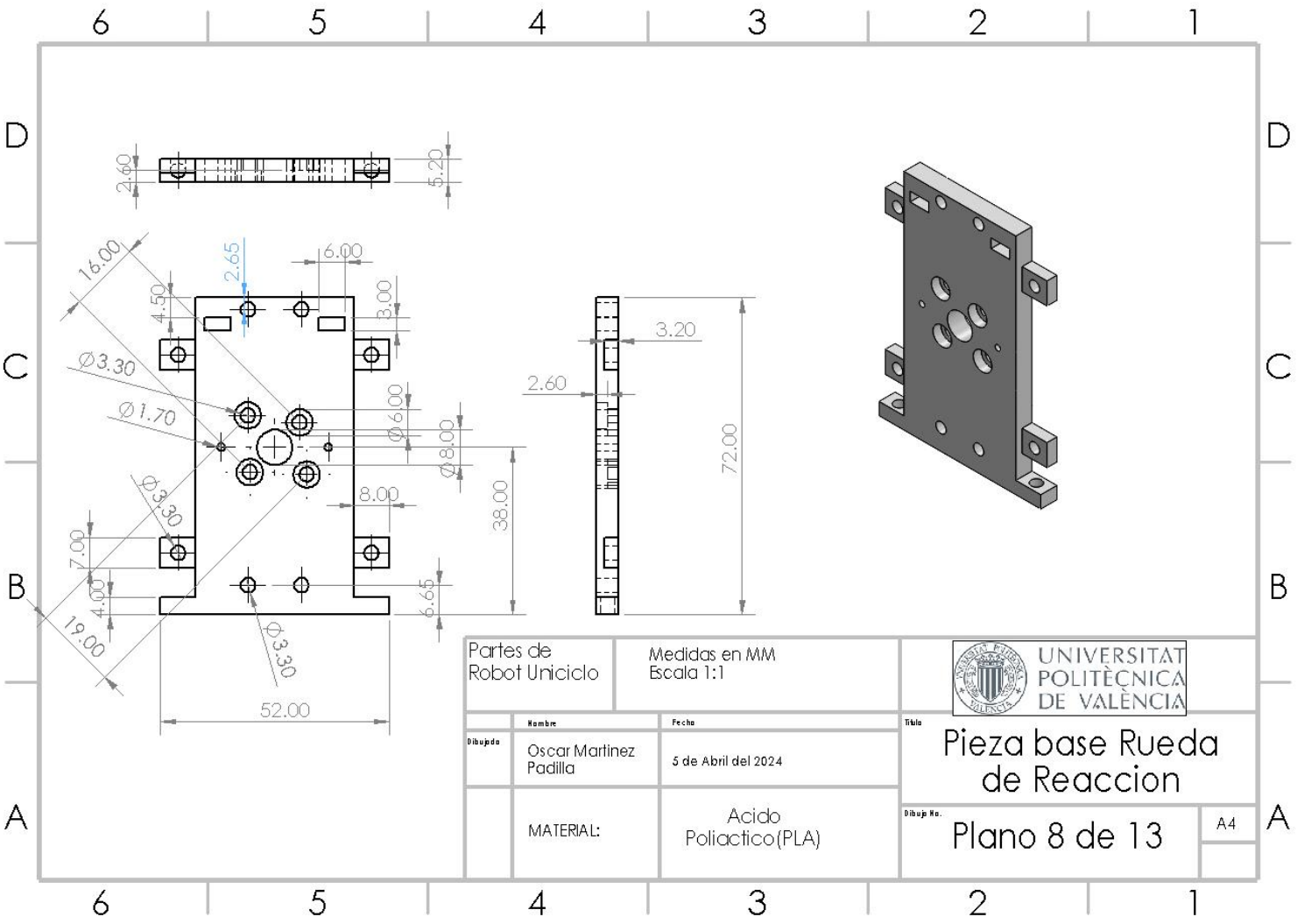
Anexo 5-Plano Pieza Carcasa Motor



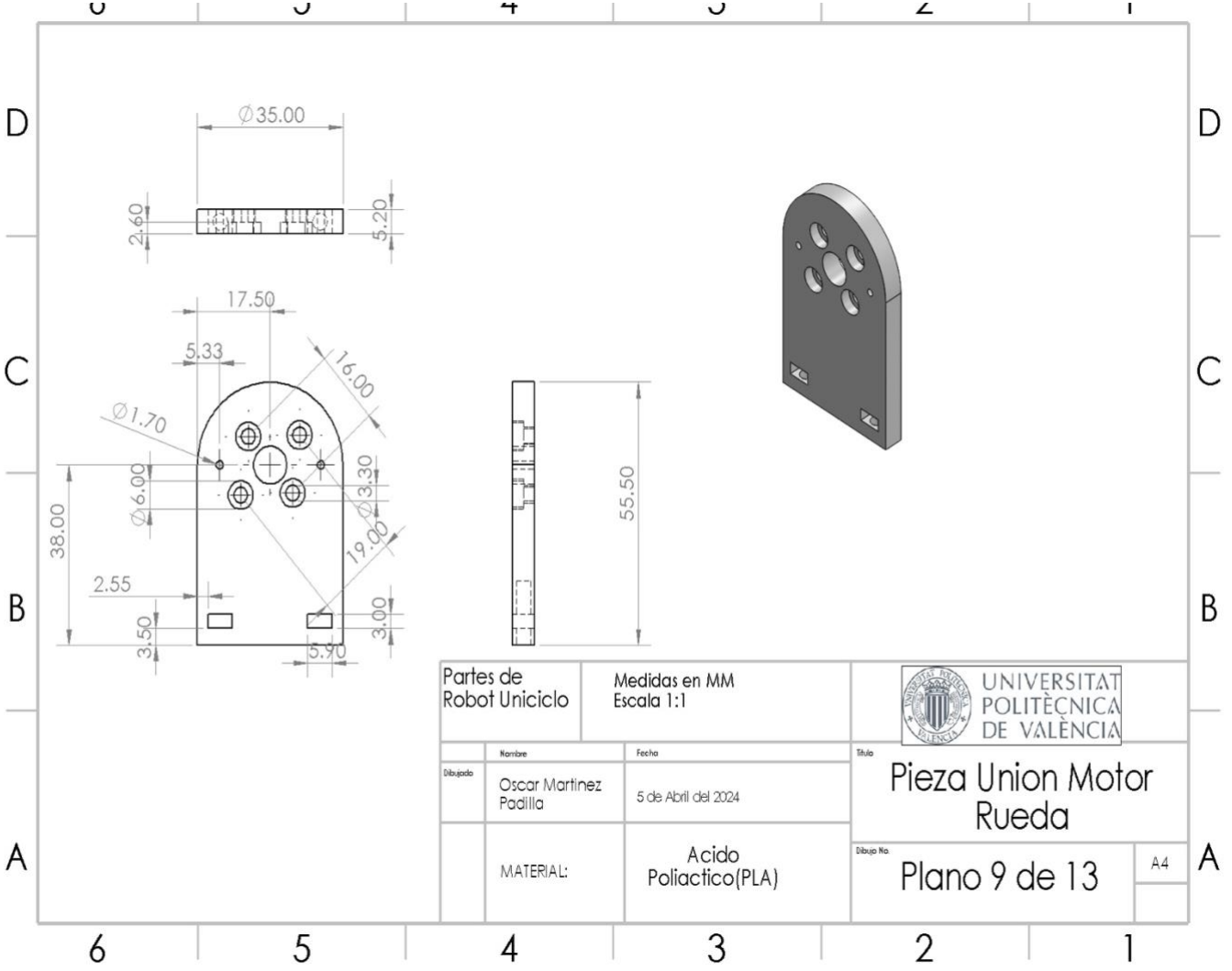
Anexo 6-Plano Pieza Base



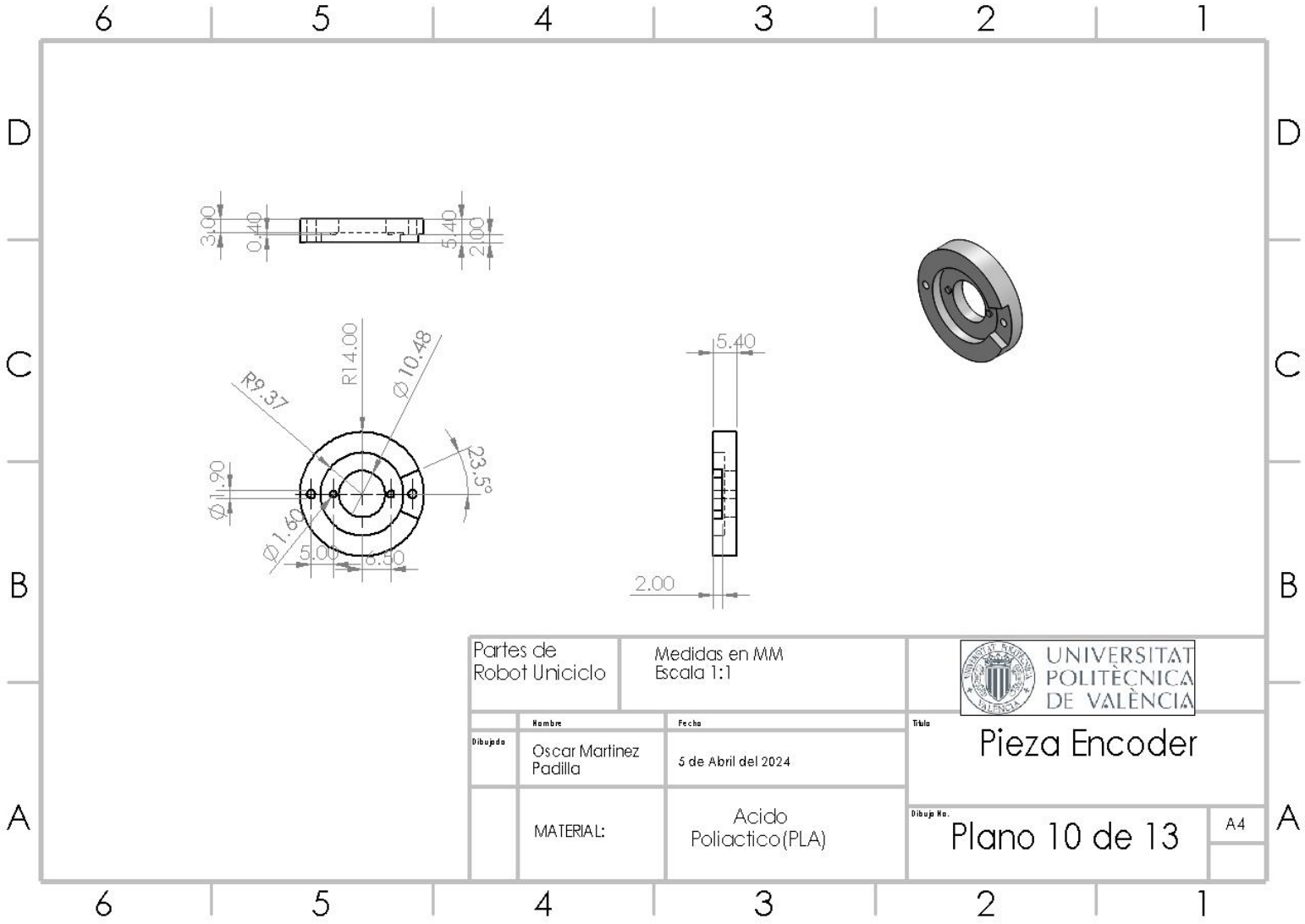
Anexo 7-Plano Pieza Base Batería



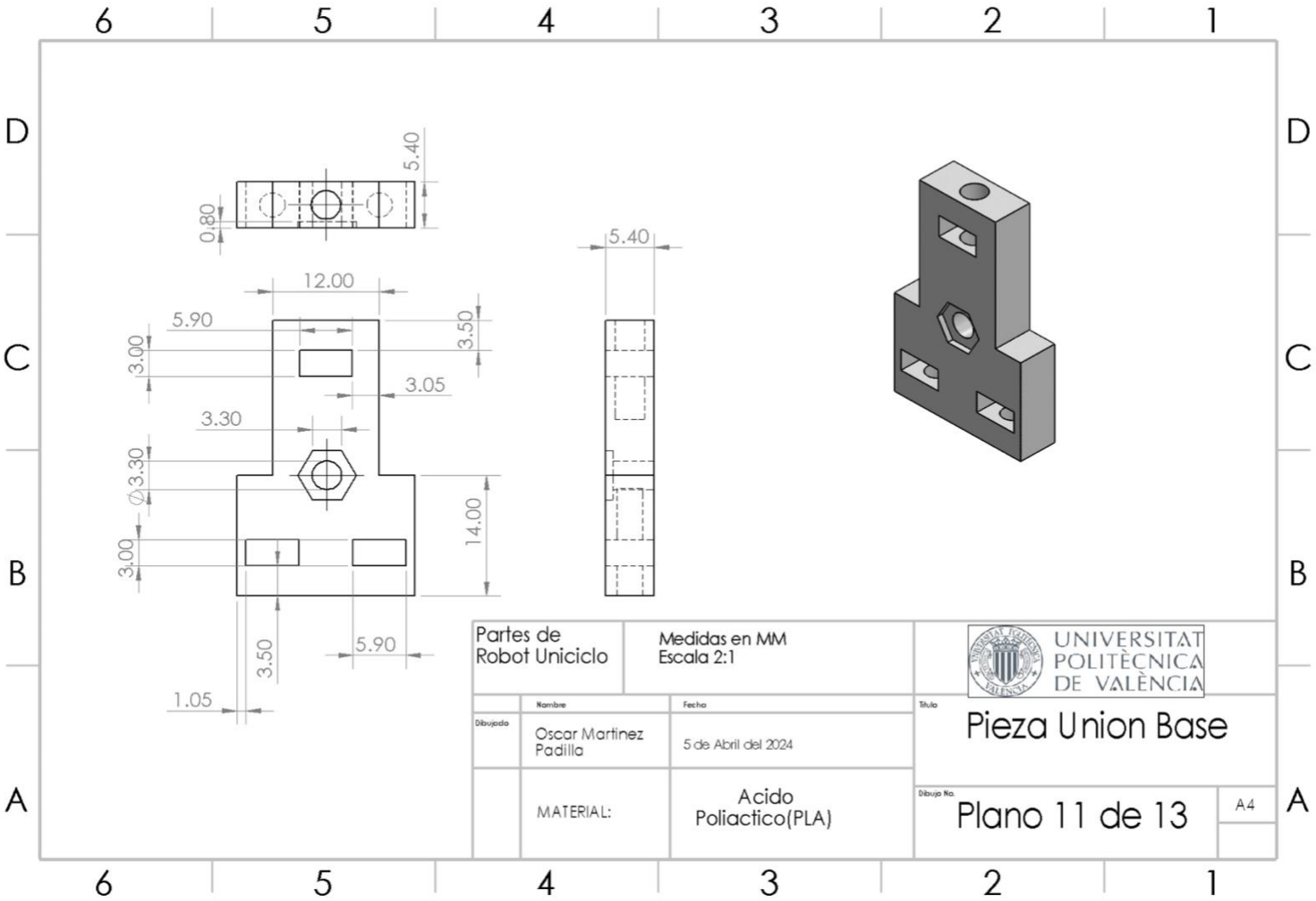
Anexo 8-Plano Pieza Base Rueda Reacción



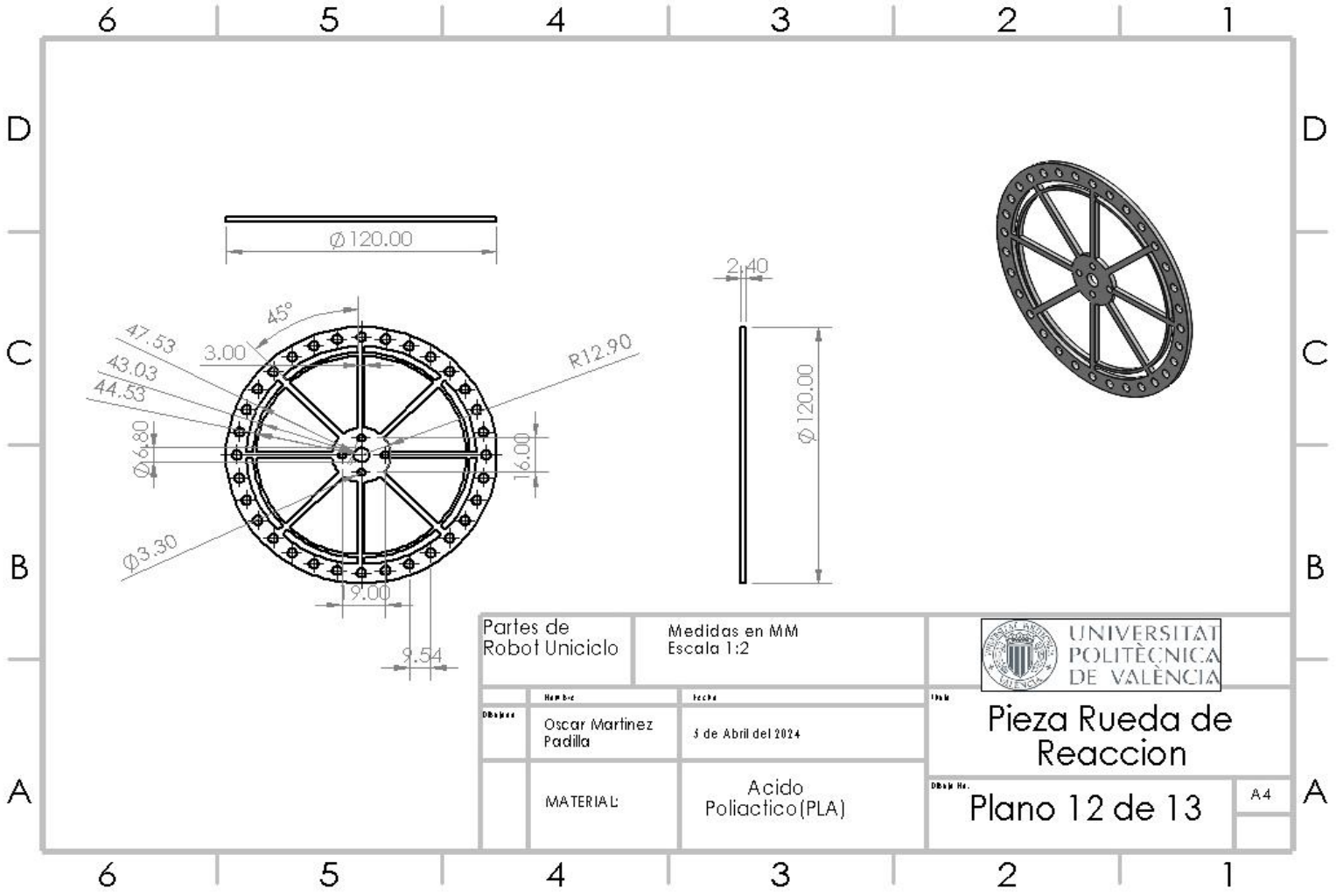
Anexo 9-Plano Pieza Unión Motor Rueda



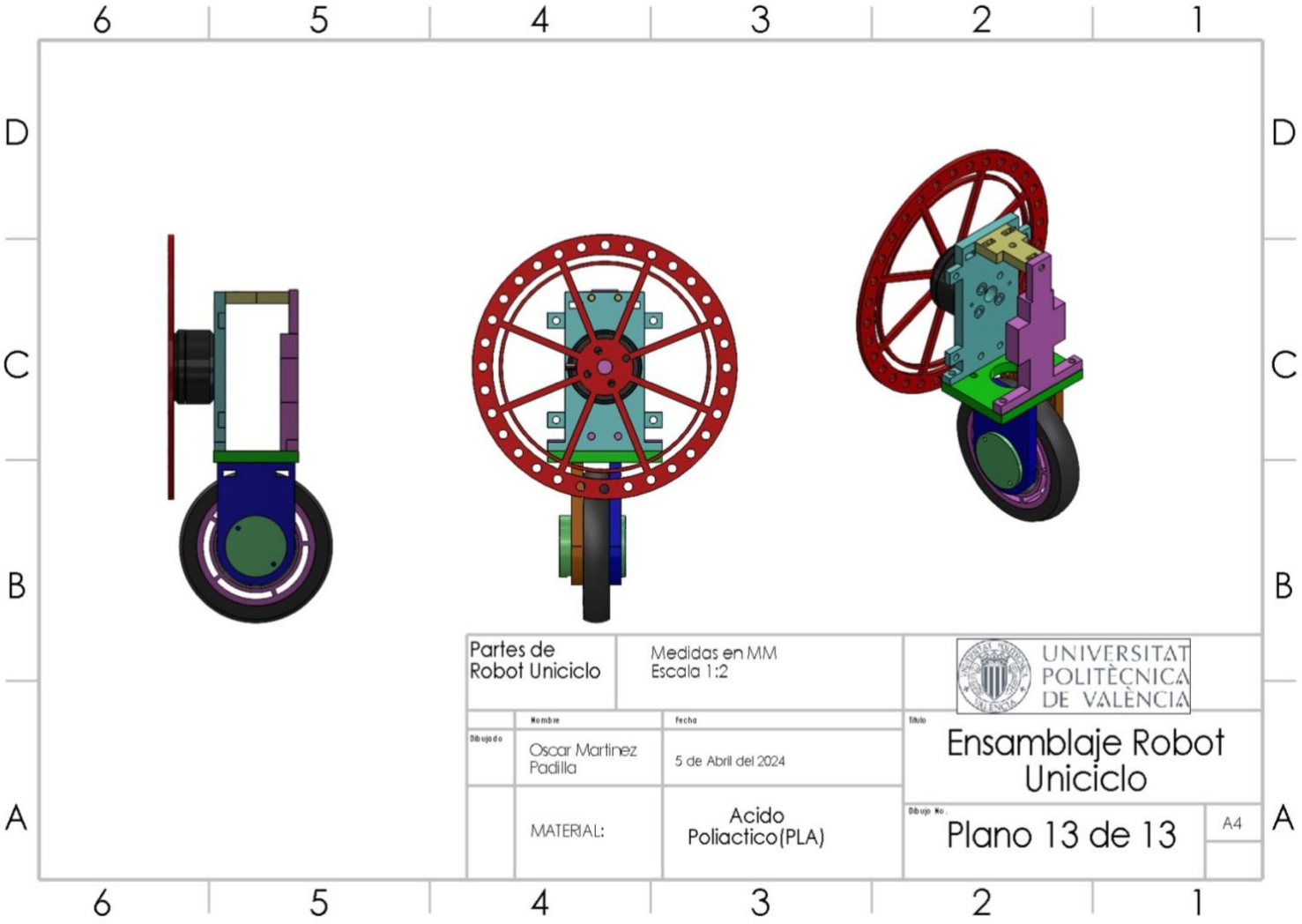
Anexo 10-Plano Pieza Encoder Rueda



Anexo 11-Plano Pieza Unión Base



Anexo 12-Plano Pieza Rueda de Reacción



Anexo 13-Plano Ensamblaje Robot Uniciclo

Anexo 14-Codigo del Robot Uniciclo

```
#include <SimpleFOC.h>

///Declaración de pines de motores y encoders
BLDCMotor motor_EjeX = BLDCMotor(3, 5, 6, 7);
BLDCMotor motor_EjeY = BLDCMotor(9, 10, 11, 7);
MagneticSensorAnalog sensor_EjeX = MagneticSensorAnalog(A1, 14, 1020);
MagneticSensorAnalog sensor_EjeY = MagneticSensorAnalog(A0, 14, 1020);

/// Se definen los registros del MPU6050, se obtienen del datasheet
#define MPU6050 0x68            // Dirección del MPU6050
#define ACCEL_CONFIG 0x1C       // Dirección de lecturas Acelerómetro
#define GYRO_CONFIG 0x1B        // Dirección lecturas del Giroscopio
#define ACCEL_XOUT_H 0x3B
#define ACCEL_XOUT_L 0x3C
#define ACCEL_YOUT_H 0x3D
#define ACCEL_YOUT_L 0x3E
#define ACCEL_ZOUT_H 0x3F
#define ACCEL_ZOUT_L 0x40
#define GYRO_XOUT_H 0x43
#define GYRO_XOUT_L 0x44
#define GYRO_YOUT_H 0x45
#define GYRO_YOUT_L 0x46
#define GYRO_ZOUT_H 0x47
#define GYRO_ZOUT_L 0x48
#define PWR_MGMT_1 0x6B //Registro para resetear MPU


//Variable para almacenar lecturas acelerómetro y giroscopio
int16_t  AcX, AcY, AcZ;
int16_t GyY, GyX, GyZ;
float gyroX, gyroY;

//Rango de salida del MPU6050
#define accSens 0           // 0 = 2g
#define gyroSens 1          // 1 = 500rad/s
#define valor_gyro 0.99
```

```
//Variable para almacenar Offset
int16_t  AcX_offset = -740;
int16_t  AcY_offset = 260;
int16_t  AcZ_offset = 110;
int16_t  GyY_offset = 0, GyZ_offset = 0 ;
int32_t  GyY_offset_sum = 0, GyZ_offset_sum = 0;

float robot_angleX, robot_angleY, filtro_EjeX, filtro_EjeY;
float Acc_angleX,  Acc_angleY;
float angle_offsetX = -5.40, angle_offsetY = 1.98;

float K1X = 1.68;
float K2X = 0.26;
float K3X = 0.17;
float K1Y = 0.84;
float K2Y = 0.1;
float K3Y = 0.18;

int tiempo_bucle = 4;

boolean vertical = false;
boolean test = false;

float alphaX = 0.25; // Filtro pasa bajo
float alphaY = 0.75; // Filtro pasa bajo


void setup() {

// Se inicializan MPU y ambos motores junto a sus respectivos encoders con la librería simplefoc
  Serial.begin(115200);
  delay(1000);

  angle_setup();
```

```
  sensor_EjeX.init();                                  // inicializa el encoder
  motor_EjeX.linkSensor(&sensor_EjeX);                 // asocia el motor a su respectivo encoder
  motor_EjeX.voltage_power_supply = 12;
  motor_EjeX.controller = ControlType::voltage;        // Control por voltaje
  motor_EjeX.voltage_sensor_align = 4;
  motor_EjeX.foc_modulation = FOCModulationType::SpaceVectorPWM;     // Modulacion FOC
  motor_EjeX.init();                                   // inicializa el motor
  motor_EjeX.initFOC();                                // alinea el encoder e inicia FOC

  sensor_EjeY.init();
  motor_EjeY.linkSensor(&sensor_EjeY);
  motor_EjeY.voltage_power_supply = 12;
  motor_EjeY.controller = ControlType::voltage;
  motor_EjeY.voltage_sensor_align = 4;
  motor_EjeY.foc_modulation = FOCModulationType::SpaceVectorPWM;
  motor_EjeY.init();
  motor_EjeY.initFOC();
}


long cuenta_bucle = 0;
float vel_motorX = 0, vel_motorY = 0;

void loop() {
  // ~1ms
  motor_EjeX.loopFOC();
  motor_EjeY.loopFOC();

  Tuning();
  angle_calc();

  if (abs(robot_angleX - angle_offsetX) > 5 || abs(robot_angleY - angle_offsetY) > 5) vertical = false;
  if (abs(robot_angleX - angle_offsetX) < 0.2 && abs(robot_angleY - angle_offsetY) < 0.2 && !vertical) vertical = true;

  gyroX = GyZ / 131.0; // Convierte a deg/s
  gyroY = GyY / 131.0;
```

```
  filtro_EjeX = alphaX * gyroX + (1 - alphaX) * filtro_EjeX;
  filtro_EjeY = alphaY * gyroY + (1 - alphaY) * filtro_EjeY;

  if (test) {
    motor_EjeX.move(4 * (motor_EjeY.shaft_angle - motor_EjeX.shaft_angle));
    motor_EjeY.move(4 * (motor_EjeX.shaft_angle - motor_EjeY.shaft_angle));
  } else if (cuenta_bucle++ > tiempo_bucle) {
    float voltajefinal_EjeX = 0, voltajefinal_EjeY = 0;

    if (vertical) {
      voltajefinal_EjeX = acc_ControlX(robot_angleX - angle_offsetX, filtro_EjeX, sensor_EjeX.getVelocity() + vel_motorX);
      vel_motorX += sensor_EjeX.getVelocity() / 50;
      voltajefinal_EjeY = acc_ControlY(robot_angleY - angle_offsetY, filtro_EjeY, sensor_EjeY.getVelocity() + vel_motorY);
      vel_motorY += sensor_EjeY.getVelocity() / 40;

      motor_EjeX.move(voltajefinal_EjeX);
      motor_EjeY.move(-voltajefinal_EjeY);
    } else {
      motor_EjeX.move(voltajefinal_EjeX);
      motor_EjeY.move(voltajefinal_EjeY);
      vel_motorX = 0;
      vel_motorY = 0;
    }
    cuenta_bucle = 0;
  }
}

int sign(int x) {
  if (x > 0) return 1;
  if (x < 0) return -1;
  if (x = 0) return 0;
}
//Se calculan las acciones de control de ambos ejes
```

```
float acc_ControlX(float error_angulo, float filtro, float m_vel) {
  float u =  K1X * error_angulo + K2X * filtro + K3X * m_vel;
  if (abs(u) > motor_EjeX.voltage_power_supply * 0.8)
    u = sign(u) * motor_EjeX.voltage_power_supply * 0.8;
  return u;
}

float acc_ControlY(float error_angulo, float filtro, float m_vel) {
  float u =  K1Y * error_angulo + K2Y * filtro + K3Y * -m_vel;
  if (abs(u) > motor_EjeY.voltage_power_supply * 0.8)
    u = sign(u) * motor_EjeY.voltage_power_supply * 0.8;
  return u;
}

void writeTo(byte device, byte address, byte value) {
  Wire.beginTransmission(device);
  Wire.write(address);
  Wire.write(value);
  Wire.endTransmission(true);
}

//setup MPU6050
void angle_setup() {
  Wire.begin();
  delay (100);
  writeTo(MPU6050, PWR_MGMT_1, 0);
  writeTo(MPU6050, ACCEL_CONFIG, accSens << 3); // la escala del acelerómetro
  writeTo(MPU6050, GYRO_CONFIG, gyroSens << 3); // la escala del giroscopio
  delay (100);

  Serial.println("Calibrating gyroscope...");

  // Calculo de los offset del giroscopio
  for (int i = 0; i < 1024; i++) {
    angle_calc();
```

```
    GyZ_offset_sum += GyZ;
    GyY_offset_sum += GyY;
    delay(5);
  }
  GyZ_offset = GyZ_offset_sum >> 10;
  Serial.print("GyZ offset value = "); Serial.println(GyZ_offset);
  GyY_offset = GyY_offset_sum >> 10;
  Serial.print("GyY offset value = "); Serial.println(GyY_offset);
}

void angle_calc() {

  // Lectura de datos del MPU6050 usando I2C

  Wire.beginTransmission(MPU6050);
  Wire.write(GYRO_XOUT_H);
  Wire.endTransmission(false);
  Wire.requestFrom(MPU6050, 6, true);
  GyX = Wire.read() << 8 | Wire.read(); // 0x43 (GYRO_XOUT_H) & 0x44 (GYRO_XOUT_L)
  GyY = Wire.read() << 8 | Wire.read(); // 0x45 (GYRO_YOUT_H) & 0x46 (GYRO_YOUT_L)
  GyZ = Wire.read() << 8 | Wire.read(); // 0x47 (GYRO_ZOUT_H) & 0x48 (GYRO_ZOUT_L)

  Wire.beginTransmission(MPU6050);
  Wire.write(ACCEL_XOUT_H);
  Wire.endTransmission(false);
  Wire.requestFrom(MPU6050, 6, true);
  AcX = Wire.read() << 8 | Wire.read(); // 0x3B (ACCEL_XOUT_H) & 0x3C (ACCEL_XOUT_L)
  AcY = Wire.read() << 8 | Wire.read(); // 0x3D (ACCEL_YOUT_H) & 0x3E (ACCEL_YOUT_L)
  AcZ = Wire.read() << 8 | Wire.read(); // 0x3F (ACCEL_ZOUT_H) & 0x40 (ACCEL_ZOUT_L)

  // Se suman los offset a los valores previos para obtener los valores actuales
  AcX += AcX_offset;
  AcY += AcY_offset;
  AcZ += AcZ_offset;
  GyZ -= GyZ_offset;
```

```
  GyY -= GyY_offset;

  robot_angleX += GyZ * 6.07968E-5;
  Acc_angleX = -atan2(AcY, AcX) * 57.2958;       // angulo del robot en Eje X
  robot_angleX = robot_angleX * valor_gyro + Acc_angleX * (1.0 - valor_gyro);

  robot_angleY += GyY * 6.07968E-5;
  Acc_angleY = atan2(AcZ, AcX) * 57.2958;        // angulo del robot en Eje Y
  robot_angleY = robot_angleY * valor_gyro + Acc_angleY * (1.0 - valor_gyro);


}

void Tuning() {   ///función para modificar los parámetros del controlador y realizar el test de los motores
  if (!Serial.available())  return 0;
  delay(2);
  char param = Serial.read();
  if (!Serial.available()) return 0;
  char cmd = Serial.read();
  Serial.flush();
  switch (param) {
    case 'b':
      if (cmd == '+')    K1X += 0.01;
      if (cmd == '-')    K1X -= 0.01;
      printValues();
      break;
    case 'g':
      if (cmd == '+')    K2X += 0.01;
      if (cmd == '-')    K2X -= 0.01;
      printValues();
      break;
    case 'c':
      if (cmd == '+')    K3X += 0.005;
      if (cmd == '-')    K3X -= 0.005;
      printValues();
```

```
      break;

    case 'p':
      if (cmd == '+')    K1Y += 0.01;
      if (cmd == '-')    K1Y -= 0.01;
      printValues();
      break;
    case 'd':
      if (cmd == '+')    K2Y += 0.01;
      if (cmd == '-')    K2Y -= 0.01;
      printValues();
      break;
    case 's':
      if (cmd == '+')    K3Y += 0.005;
      if (cmd == '-')    K3Y -= 0.005;
      printValues();
      break;
   case 't':
      if (cmd == '+')    test = true;
      if (cmd == '-')    test = false;
      if (test) Serial.println("test ON");
      else Serial.println("test OFF");
      break;
   }
}

void printValues() {
  Serial.print("K1X: "); Serial.print(K1X);
  Serial.print(" K2X: "); Serial.print(K2X);
  Serial.print(" K3X: "); Serial.println(K3X,3);
  Serial.print("K1Y: "); Serial.print(K1Y);
  Serial.print(" K2Y: "); Serial.print(K2Y);
  Serial.print(" K3Y: "); Serial.println(K3Y,3);
}
```