Universidad de Concepción
Facultad de Ingeniería
Departamento de Ingeniería Mecánica

# Construcción y validación de un simulador solar para verificación de algoritmo de actitud en Cubesats en órbitas LEO

POR

**Alvaro Fabian Maricahuin Monsalves**

**Memoria de Titulo presentada a la Facultad de Ingeniería de la Universidad de Concepción para optar al título de Ingeniero Civil Aeroespacial**

Profesores Guía:

PhD Bernardo Hernandez Vicente

PhD (C) Elías Obreque Sepúlveda

Febrero 2024

Concepción (Chile)

**Resumen**

El crecimiento sostenido de las misiones espaciales ha posicionado al Cubesat como una interesante opción para la entrada al mercado espacial. Un Cubesat pertenece a la categoría de nanosatélites, específicamente, es un cubo de lado 10 [cm] que por su tamaño es una alternativa económica para alcanzar el espacio.

Lamentablemente, los Cubesat son equipos con alta tasa de falla, particularmente en los subsistemas de energía (EPS o "Electrical Power Systems") y telecomunicación (TT&C o "Telemmetry, Tracking and Command"), donde la evidencia concluye que aumentar los ensayos es mejor que dotar al satélite de subsistemas redundantes. Dentro de los ensayos necesarios para la validación del satélite se encuentran los ensayos de actitud que tienen estricta relación con los subsistemas de energía y telecomunicación. Un banco de ensayos comercial para la verificación del subsistema de actitud incluye un simulador solar, un simulador de campos electromagnéticos y rodamientos de aire y su costo total excede los USD $20.000.

Con el fin de mantener los Cubesat como una alternativa viable económicamente, en la presente Memoria de Título se trabaja en el diseño, construcción y validación de un simulador solar para verificación del algoritmo de actitud, considerando presupuesto acotado y una implementación en un tiempo máximo de un semestre académico.

Utilizando la metodología de cascada para la concepción de producto se proponen diferentes bancos de ensayo y bajo reuniones con stakeholders se define el banco de ensayos a implementar. El simulador es montado en el Laboratorio de Técnicas Aeroespaciales y su funcionamiento se basa en la interacción de engranajes accionados por motores paso a paso. El sol es simulado por luces LED de 10 [W] de espectro completo controladas a través de relays con la plataforma Arduino.

Con el fin de verificar el banco de ensayos se realizan dos pruebas básicas de funcionamiento adquiriendo datos con un dispositivo que emula un satélite. Las pruebas confirman la funcionalidad del banco de ensayos como un sistema total, contando con motores que realizan los movimientos suavemente y una luz de espectro completo funcional a las aplicaciones requeridas, además de una estructura acorde a las exigencias. Posteriormente, se discuten dos alternativas para realizar pruebas de órbita para finalmente implementar un código basado en la librería AccelStepper sacando el provecho de las funcionalidades de posición y referencia de los motores.

**Palabras clave: Cubesat, ADCS, Simulador Solar, New Space.**

**Abstract**

The continuous growth of space missions has positioned Cubesats as an intriguing option for entry into the space market. Cubesats belong to the Nanosatellites category, characterized by their compact size of a 10 cm side, making them an economical choice for space exploration. Unfortunately, Cubesats have shown a significantly high failure rate, with the "Energy Power Subsystems (EPS)" and "Telemetry, Tracking and Control Subsystem (TT&C)" being critical subsystems prone to failure.

Studies indicate that improving testing procedures is more preferable than incorporating subsystem redundancy to enhance satellite reliability. Among the essential tests for satellite validation,"Attitude Determination and Control Subsystem (ADCS)" testing is crucial and directly related to the performance of EPS and TT&C subsystems. Traditional ADCS testbeds, equipped with a sun simulator, electromagnetic field simulator, and air bearing, often incur costs exceeding $20,000 USD.

In an effort to maintain Cubesats as an economically viable option, this Memoria de Título focuses on the design, construction, and validation of a sun simulator for ADCS algorithm verification. The objective is to achieve comparable functionality within a reduced budget and a maximum implementation time of one academic semester.

By implementing Waterfall Methodology different testbed designs are conceived to be discussed with stakeholders members. Finally the definitive testbed is designed under the approbation of the members. The testbed is built at Laboratorio de Técnicas Aeroespaciales and its functioning is based on the interaction of gears actuated by stepper motors. The Sun is simulated by LED lights of full spectrum controlled through relay with Arduino platform.

To verify the functioning of the testbed two basic tests were executed. In a way to verify the testbed as a costumer, a satellite equiped with light sensors and an Arduino was constructed and used to obtain results of the light emission. These tests effectively verified the functionality of the testbed as a whole system including smooth operation motors and a full spectrum LED chosen to meet specific requirements. Subsequently, a discussion of two alternatives to excute orbit test defined the option to develop. A code based on AccelStepper library is implemented. This choice was driven by the desire to maximize the potential of the library, particularly in terms of achieving accurate position referencing and optimal motor speed control.

**Keywords: Cubesat, Sun Simulator, ADCS, New Space**

# Índice general

# Índice de tablas

# Índice de figuras

# Glosario

| | |
|---|---|
| AIT | Assembly, Integration and Testing |
| ESA | European Space Agency |
| NASA | National Aeronautics and Space Administration |
| GECT | Ground Electrical Comprehensive Test |
| GPS | Global Positioning System |
| EGSE | Electrical Ground Support Equipment |
| TT&C | Telemetry, Tracking and Communication |
| EPS | Electrical Power System |
| EEI | Estación Espacial Internacional |
| ADCS | Attitude & Determination Control System |
| ALOS | Advanced Land Observing Satellite |
| SSOT | Sistema Satelital de Observación Terrestre |
| LEO | Low Earth Orbit |
| GOES | Geostationary Operational Enviromental Satellite |
| ECI | Earth Centered Inertial |
| UT | Universal Time |
| UTC | Universal Time Coordinate |
| GNSS | Global Navigation Satellite System |
| DLR | Deutsches Zentrum für Luft- und Raumfahrt e.V |
| FACE | Facility for Attitude Control Experiments |
| NTNU | Norwegian University of Science and Technology |
| GNC | Guidance, Navigation & Control |
| SGP4 | Simplified General Perturbations Model |
| VSOP | Variations Séculaires des Orbites Planétaires |
| LED | Light Emiting Diode |
| COB | Chip On Board |

**Símbolos**

| | |
|---|---|
| I | Corriente [A] |

# CAPÍTULO 1: Introducción

## 1.1 Contexto

Durante las últimas 2 decadas, se ha visto un aumento en la popularidad de las misiones espaciales, en especial, misiones de nanosatélites Cubesat [1]. Un Cubesat es un tipo de nanosatélite en que su tamaño y forma están estandarizados; un Cubesat de 1U es un cubo de 10x10x10 [cm], estos nanosatélites se pueden acoplar entre ellos en formatos de 2U, 3U, 6U y 12U [2]. El diseño y procedimiento de una misión espacial está normalizado, entre los entes reguladores se encuentra la NASA, en la Figura 1.1 se observa el ciclo de vida de un proyecto según la NASA [3].

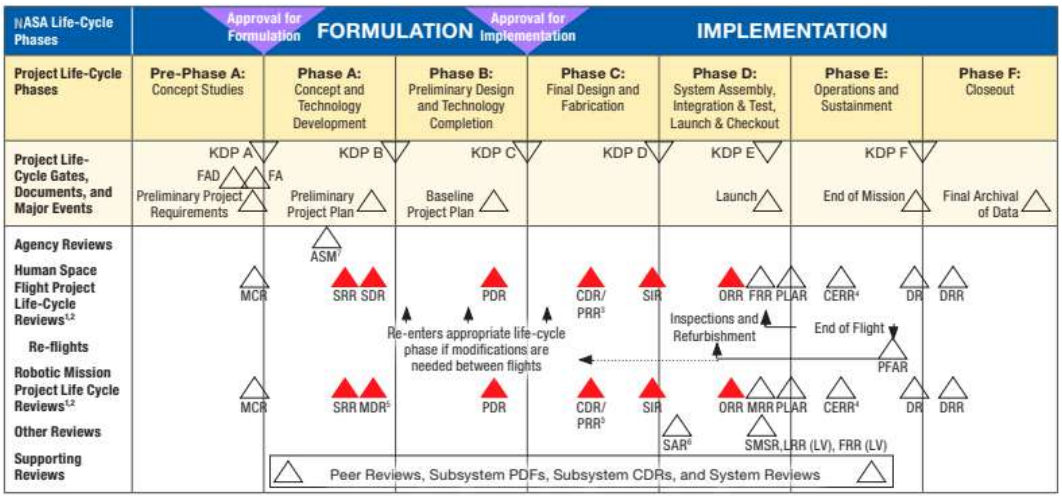



Figura 1.1: Ciclo de vida de un proyecto según NASA [3].

La Fase D corresponde a la fase de ensamblaje, integración y pruebas o fase AIT del inglés “Assmebly, Integration & Test”, dentro de esta fase se encuentra la etapa de ”Ground Electrical Comprehensive Test” [4], en adelante GECT. Esta etapa de ensayos es un paso indispensable en el ciclo del proyecto en donde se pone a prueba el equipo a través de ensayos de performance, de funcionalidad general de los subsistemas y de compatibilidad de las interfaces. Durante la etapa GECT, los ensayos se llevan a cabo en equipos ”EGSE” [4] por sus siglas del inglés ”Electrical Ground Support Equipment”, los equipos EGSE son apoyo fundamental en el cumplimiento de objetivos generales y específicos de los ensayos; simuladores de satélite, instrumentación de señales, analizadores de espectro C y Simulador solar son ejemplos de equipos EGSE. Finalmente, según

NASA [3], la fase D concluye con un sistema capaz de alcanzar todos los propósitos para el cual se ha creado.

A pesar de los rigurosos procesos que se deben realizar durante el proyecto, los cubesat son satélites con baja taza de éxito, como se puede observar en la Figura 1.2, solo un 15 % de los satélites han sido exitosos cumpliendo todos sus objetivos (Mission success), mientras que un gran porcentaje se distribuye entre misiones que logran parcialmente sus objetivos (Primary op.) y aquellas que pudieron conectar con el satélite sólo una vez (Commissioned). Dentro de las fallas, un 21 % se atribuye directamente a fallas del satélite como se observa en la Figura 1.2. Por otra parte, los subsistemas cuyas fallas son fatales al sistema total son los subsistemas TT&C (Telemetry, Tracking and Communication) y EPS (Electrical Power System), además de ser fallas consideradas falla infante, es decir, en los primeros días de operación [5]. Allí nace la interrogante si para contrarrestar las fallas prematuras es más conveniente equipar al sistema con subsistemas redundantes o aumentar los ensayos de los subsistemas; estudios del mantenimiento y la confiabilidad de los cubesat demuestran que es mejor aumentar las pruebas a las que son sometidos [6].

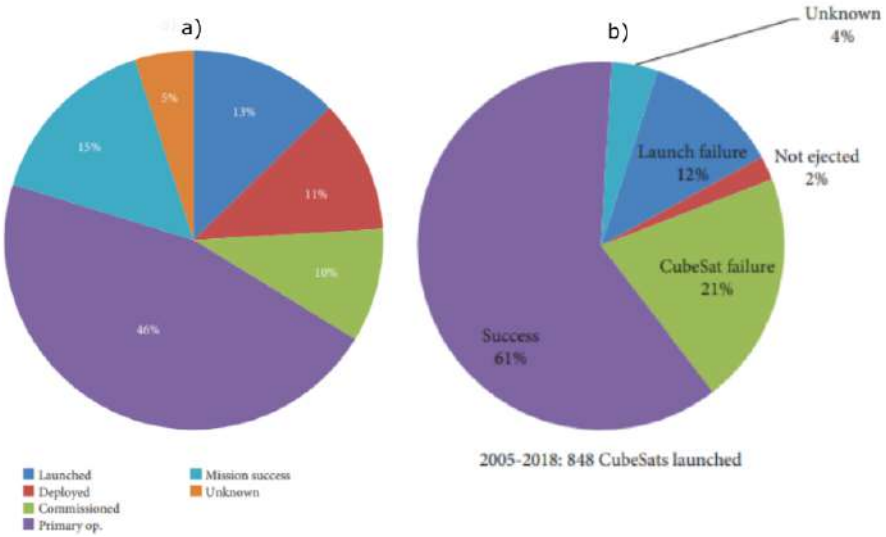



Figura 1.2: Estadísticas de éxito y tipo de fallas en Cubesats [7].

El subsistema "Attitude Determination & Control Systems" o ADCS, es el subsistema del satélite que le permite conocer su actitud y orientarse haciendo uso de los actuadores que posee, algunas de las funciones que dependen del subsistema ADCS son:

1. Orientación del satélite para su energización a través de paneles solares.
2. Apuntamiento de antenas para comunicación con estaciones terrestres.
3. Apuntamiento de cámaras o telescopios para cumplimiento de misiones astronómicas y/o observación terrestre.
4. Orientación para Rendezvous para envío de carga útil como misiones a la Estación Espacial Internacional o EEI.

En definitiva, las funciones del subsistema ADCS se relacionan directamente con las funciones de los subsistemas TT&C y EPS, apoyándose de las conclusiones de los estudios mencionados [5–7], se podría deducir que un aumento de las pruebas del subsistema ADCS podría ayudar a mejorar el rendimiento de los subsistemas TT&C y EPS.

### 1.2 Simulador Solar

El subsistema ADCS está compuesto de actuadores, sensores y algoritmos diseñados para el control de actitud del satélite. La combinación típica de sensores consiste en Startracker y GPS de alta resolución en conjunto a los respectivos algoritmos de estimación. Un Startracker es una cámara que identifica estrellas, determina su posición y a través de una base de datos previamente cargada en el dispositivo es capaz de determinar la orientación del satélite. Algunos satélites equipados con esta configuración de sensores son las plataformas GOES-1 [8], ALOS [9] y el satélite nacional SSOT [10]. Otro sensor utilizado en el subsistema ADCS es el sensor solar, este captura la luz solar a través de una película fotosensible y en función de la magnitud recibida calcula el ángulo de incidencia. El sensor solar soluciona una de las debilidades del startracker, este último se ve muy afectado por la potente luz solar obligando a aplicar filtros físicos y de algoritmo para su funcionamiento [8]. Nuevas misiones satélitales prefieren los sensores solares por sobre los startracker debido a su menor tamaño y masa, un ejemplo es el sensor solar en el nanosatélite español NANOSAT1-B [11] de tamaño 3x3 [cm] y 24 [g] de masa.

La implementación exitosa de un sensor solar en una plataforma satélital necesita obligatoriamente de pruebas y verificaciones. La evaluación de performance de un sensor solar se realiza en un simulador solar, este simulador solar posiciona una fuente de luz de características similares al sol en trayectorias conocidas para evaluar el subsistema y sus componentes. Las mediciones del satélite se comparan con la trayectoria conocida y los resultados esperados. De estos analisis derivan la calibración del sensor, la determinación del offset y diferentes parámetros asociados al algoritmo. En función de los resultados, las conclusiones pueden implicar la modificación del modelo del satélite como por ejemplo cambios en la posición del sensor o incluso la sustitución del sensor por alguno de propiedades que mejor se adapten a la misión.

Además de los ensayos lumínicos, un simulador solar desempeña un papel crucial en las pruebas técnicas simulando campos electromagnéticos y reducción de fricción a través de rodamientos de aire. En la Figura 1.3, se observa un simulador solar con todas las características mencionadas anteriormente.

El costo asociado a la construcción del equipo en la Figura 1.3 asciende a alrededor de 20000 €. Dentro de este costo, la sección encargada de simular la posición de la luz solar representa un 17.5 % del total, es decir un costo de 3500 € [12]. En esta versión acotada únicamente a la simulación de la posición de la luz solar, se vuelven prescindibles las cajas de Helmholtz, que son utilizadas para simular los campos electromagnéticos, así como el sistema de rodamiento de aire destinado a minimizar la fricción.

Esta adaptación que limita las funcionalidades del equipo únicamente a la simulación de la posición de la luz solar aporta una mayor versatilidad al banco de pruebas. Esto posibilita la evaluación de distintos sensores y algoritmos de orientación para el satélite, con el objetivo de encontrar la

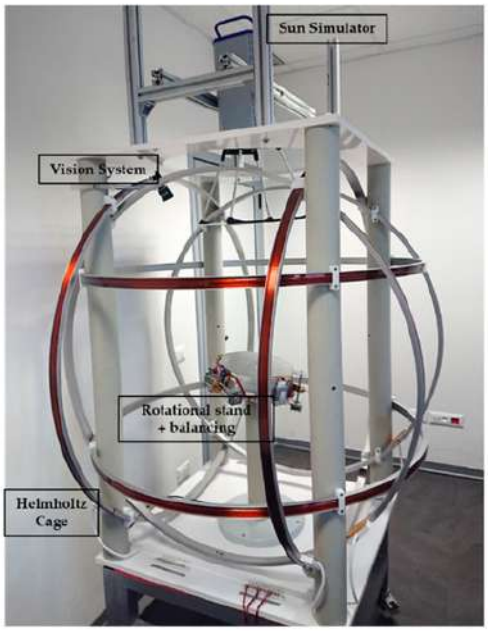



Figura 1.3: Simulador Solar [12].

combinación óptima que garantice el cumplimiento de los requisitos y objetivos establecidos para la misión.

## 1.3 Objetivos

El objetivo principal de esta memoria consiste en: Construir y validar la operación de un simulador solar para verificación de algoritmo de actitud en Cubesats en órbita LEO.

Los objetivos específicos del proyecto son:

1. Validación del diseño preliminar con stakeholders para asegurar una comprensión de las capacidades y limitaciones técnicas del banco de ensayos.
2. Elaboración del presupuesto del banco de ensayos en función de las características y alcanzable por el DIM UdeC.
3. Construcción del banco de ensayos en el espacio físico y tiempo previamente definidos.
4. Ejecución de pruebas en el banco de ensayos de al menos una órbita LEO para evaluar el rendimiento y la funcionalidad del proyecto aplicando correcciones y mejoras en caso de ser necesario.
5. Redacción del informe final del proyecto, incorporando documentación de la construcción y los resultados de las pruebas ejecutadas que validan el funcionamiento del simulador.

## 1.4 Consideraciones de diseño

Los requerimientos mínimos extraídos a partir de las entrevistas con los patrocinadores:

1. Proporcionar espacio suficiente para ensayos de satélites Cubesats de 12U.
2. Simular el movimiento relativo al sol en órbitas LEO.
3. Proveer la fuente de luz necesaria para activar los sensores del satélite.
4. Priorizar la seguridad del satélite evitando cualquier tipo de movimiento en los ensayos.
5. Presentar resultados a tiempo real, procurando continuidad en los datos.

## 1.5 Metodología

Para la metodología del proyecto se utilizará el modelo *Waterfall* o cascada, se elige este método dado que este proyecto reúne suficientes características que recomiendan la aplicación del método mencionado, cómo:

1. Requisitos estables: Los requisitos del proyectos están definidos en el planteamiento del problema y cambiarán muy poco en el desarrollo del proyecto.
2. Documentación completa: El proyecto requiere una documentación detallada que será presentada a través de avances a lo largo del semestre y en el informe de presentación al final del semestre.
3. Restricciones externas: El proyecto tiene fecha límite máxima y requisitos esenciales para su éxito.

El proceso del método cascada consiste en 6 fases, donde para pasar a la siguiente se necesita de haber completado la etapa anterior en su totalidad [13]. Las etapas del método cascada se muestran en la Figura 1.4.

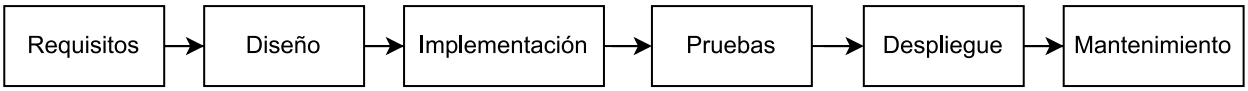



Figura 1.4: Etapas del método cascada [13].

A continuación se detallan los procesos para el cumplimiento de cada una de las etapas:

- Requisitos: Los requisitos fueron abordados en el Proyecto de Ingeniería [14] y serán revisados y reformulados en caso de ser necesario con reuniones con los stakeholders para finalizar con una sesión conjunta en donde se revisen cada uno de los objetivos y requisitos para la aprobación de la totalidad del grupo.
- Diseño: El diseño del banco de ensayos será reformulado en base a los requisitos y los comentarios de los stakeholders. Posteriormente con el diseño aprobado, se elaborará el presupuesto de todos los componentes necesarios su adquisición a través de los fondos del DIM UdeC.

- Implementación: La implementación del banco de ensayos comienza con la elección y acondicionamiento del espacio físico en el que se instalará para posteriormente revisar cada uno los componentes adquiridos comprobando su buen funcionamiento. Finalmente se construye el banco de ensayos dejando documentación de cada una de las acciones realizadas. Además la implementación también considera la elaboración del software para la ejecución de las pruebas.
- Pruebas: Las pruebas en el banco de ensayos se diseñan utilizando como base el código resultado del Proyecto de Ingeniería [14], a través del código desarrollado en la etapa anterior se detallarán todos los resultados que se desea obtener en las pruebas, además se debe indicar el procedimiento de ejecución de cada prueba considerando el encendido de los componentes electrónicos, las órdenes necesarias para la ejecución de las pruebas y el apagado correspondiente.
- Despliegue: Una vez se ha terminado la fase de pruebas y se ha verificado el funcionamiento del banco de ensayos, entonces se pueden realizar diferentes ensayos que servirán para la documentación del informe de presentación.

Finalmente la etapa de Mantenimiento no se abordará en este proyecto pues la finalidad no es un producto comercial sino la construcción de un banco de ensayos en el marco del proyecto de graduación.

### 1.6 Carta Gantt

La Carta Gantt correspondiente a la organización del proyecto se encuentra en el Anexo.

# CAPÍTULO 2: Marco teórico

Para simular la posición del Sol relativa al satélite, se deben considerar una serie de cálculos y de transformaciones entre diferentes sistemas de referencia que permitan conocer la posición del Sol en el sistema deseado. Utilizando como base el trabajo en [14], estos cálculos se resumen en:

1. Conocer la posición del Sol en el sistema Inercial.
2. Conocer la posición del satélite en el sistema Inercial.
3. Obtener el vector Sol desde el satélite para cada instante en sistema inercial.
4. Simular la rotación del satélite con cuaterniones.
5. Obtener el vector Sol desde el satélite en sistema de referencia centrado en el cuerpo.

A continuación se resume la teoría utilizada para obtener los resultados expuestos anteriormente.

## 2.1 Sistemas de referencia.

La amplia variedad de disciplinas que se benefician de la astro-navegación y la mecánica orbital conlleva la interpretación de resultados desde diferentes sistemas de referencia, donde se prefieren unos sobre otros según la información de la que se dispone y los resultados esperados. Al comienzo de este capítulo, se mencionan dos sistemas de referencia, el Sistema de Referencia Inercial o ECI por sus siglas en inglés (Earth Centered Inertial) y el Sistema de Referencia Centrado en el cuerpo o BodyFrame, a continuación se describen estos sistemas de referencia y algunas de sus aplicaciones.

### 2.1.1 Sistema de referencia inercial

Este sistema se origina en el centro de masas de la Tierra. Su eje X está orientado en dirección al equinoccio de primavera, el plano XY coincide con el plano ecuatorial celeste y el eje Z apunta hacia el polo norte celestial, que coincide con el eje de rotación de la Tierra. La Figura 2.1 ilustra este marco de referencia y muestra una representación gráfica de sus ejes. Este sistema también es conocido como sistema geocéntrico inercial o sistema geocéntrico ecuatorial.

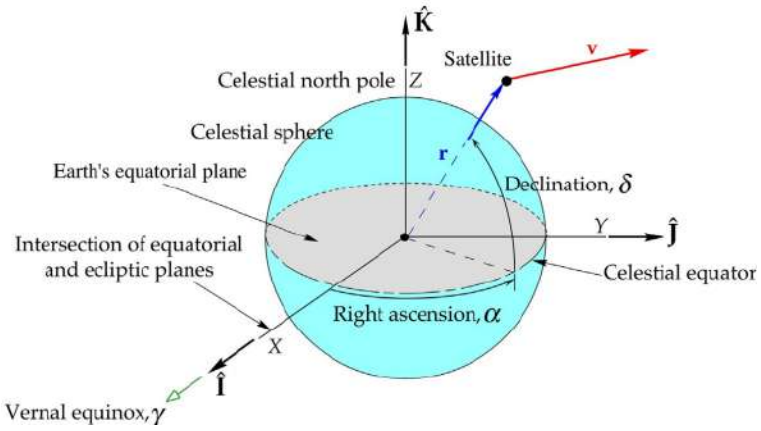



Figura 2.1: Sistema de referencias geocéntrico inercial [15].

### 2.1.2 Sistema de referencia Bodyframe

el sistema de referencia Bodyframe, centrado en el cuerpo, es el sistema de referencia asociado al satélite. En este sistema, el origen también se encuentra en el centro de masa del satélite, y sus ejes están definidos por la geometría del propio satélite, es decir, se orientan a lo largo de las direcciones principales de inercia, tal como se representa en la Figura 2.2.

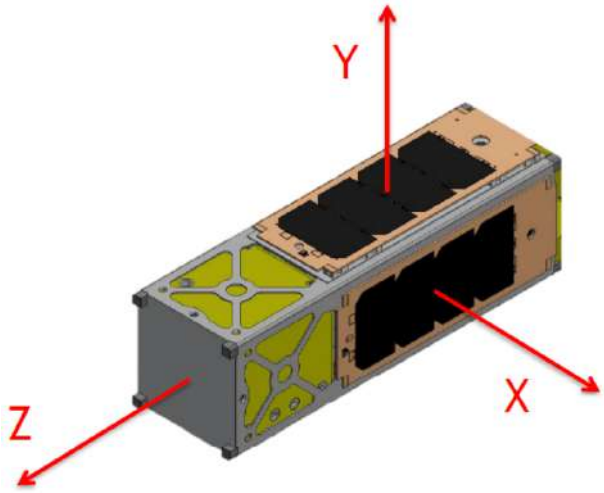



Figura 2.2: Sistema de referencia centrado en el cuerpo.

## 2.2 Posición de un cuerpo en el cielo

La interacción entre dos cuerpos másicos considerados esféricos y sin ninguna perturbación mas que la interacción misma entre ellos está modelada por la Ecuación de Kepler

$$\ddot{\vec{r}} = \frac{-G \cdot m_1 \cdot m_2}{r^3}\vec{r}. \tag{2.1}$$

Donde $\vec{r}$ es el vector posición de un cuerpo respecto del otro, $G$ es la constante gravitacional y $m$ las masas de los cuerpos en interacción. La solución de la ecuación diferencial en 2.1 permite obtener la posición y velocidad de los cuerpos en análisis. La interacción Tierra-Satélite está definida a partir de la ecuación anterior, añadiendo perturbaciones como la interacción solar, el arrastre atmosférico y la forma de la tierra es posible definir la órbita del satélite y los parámetros orbitales descritos en [15]. De la misma forma y con alta complejidad, la interacción Tierra-Sol está definida con la ecuación de Kepler y la solución de la ecuación deriva en las efemérides solares.

## 2.3 Cuaterniones

Comúnmente una rotación de vectores o de sistemas de referencia se realiza a través de la definición de ángulos de euler o su homólogo los ángulos de yaw, pitch y roll. Estos ángulos dan origen a las matrices de rotación donde cada componente es una función seno o coseno de alguno de los ángulos mencionados.

Los cuaterniones son una alternativa a las matrices de rotación y son la representación algebraica de una rotación de un vector en un ángulo $\theta$ al rededor de un eje $\hat{u}$ como muestra la Figura 2.3.

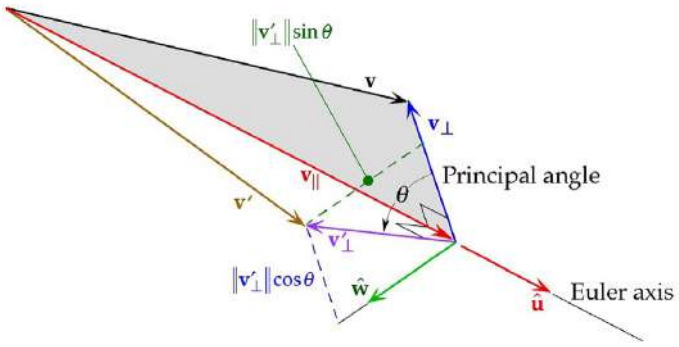



Figura 2.3: Rotación de un vector a través de un ángulo $\theta$ a lo largo del eje $\hat{u}$ [15].

Un cuaternión se escribe de la siguiente manera:

$$q = \begin{bmatrix} \sin\left(\frac{\theta}{2}\right) \cdot \hat{u} \\ \cdots \\ \cos\left(\frac{\theta}{2}\right) \end{bmatrix}.$$

Aplicado a la mecánica orbital, $\hat{u}$ es el vector unitario en el eje respecto al cual el sistema inercial se rota al sistema bodyframe y $\theta$ es el ángulo de rotación.

## 2.4 El tiempo

A lo largo de la historia han existido diferentes formas de medir el tiempo, cada civilización ha creado sus calendarios para su orden y organización como por ejemplo los calendarios hebreo,

gregoriano o romano. La mayoría de los calendarios y las formas de organizar el tiempo se basan en fenómenos astronómicos como el calendario lunar, las estaciones del año o el paso del Sol por determinados puntos [16].

Como una forma de unificar la concepción del tiempo con el fin de comprender de manera única y exacta un suceso, se define la escala de tiempo universal o UT por sus siglas en inglés "Universal Time". Un UT es un día solar medido en el Meridiano de Greenwhich. Una mejora del UT a través de relojes atómicos y segundos solapados da origen al UTC o "Universal Time Coordinate" con el que se definen las zonas horarias en todo el mundo, es decir, actualmente Chile está en la zona UTC-3 y Francia UTC+1, siendo UTC+0 la ubicación geográfica del Meridiano de Greenwhich.

Con la escala de tiempo definida e igual en todo el mundo, es necesario definir una forma de expresar una fecha y para ello se utilizan escalas de tiempo. La escala de tiempo utilizada en la mecánica orbital son los días Julianos. Los días julianos o número de días julianos son los días transcurridos desde el mediodía del 1 de Enero de 4713 AC. Para disminuir la magnitud de las mediciones, se define la época J2000, que son los días julianos a partir del mediodía del 1 de Enero del 2000, de esta forma:

$$J2000 = JD - 2451545,$$

por ejemplo, hoy 31 de Enero de 2024 son 8766 días julianos en J2000.

## 2.5 SGP4

El modelo de Perturbación General Simplificado, conocido como SGP4, es un método de propagación orbital desarrollado en las décadas de 1960 y 1970. Sin embargo, no fue hasta 2006 que Vallado mejoró el rendimiento del código, proponiendo la utilización de los TLE como condiciones iniciales en la solución [17]. Un TLE o Two Line Element es un archivo de texto que contiene información descriptiva del satélite y sus elementos orbitales en un tiempo determinado, tal como muestra la Figura 2.4, la información contenida en un TLE es útil para propagar un satélite en su órbita utilizando por ejemplo, el SGP4.

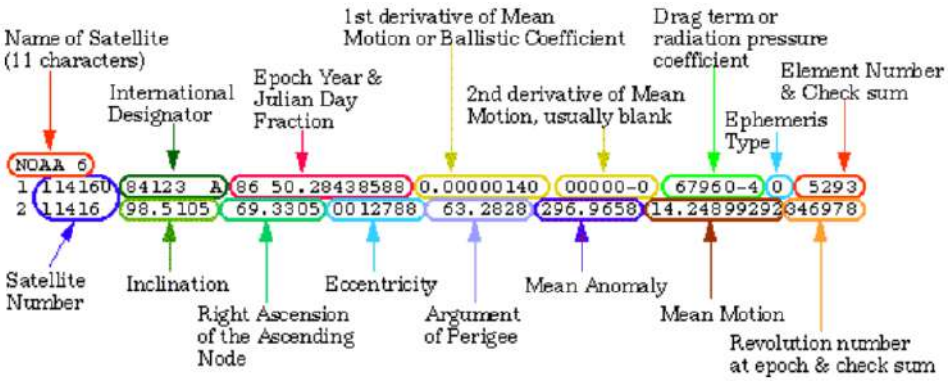



Figura 2.4: Información contenida en un TLE [18]

El SGP4 emplea métodos numéricos como el método de mínimos cuadrados y el método de eliminación de Gauss-Jordan para llevar a cabo la propagación. Este propagador tiene en cuenta perturbaciones como el achatamiento de la Tierra, el arrastre atmosférico, la radiación y los efectos gravitacionales derivados de la interacción con otros cuerpos celestes, como el Sol o la Luna.

# CAPÍTULO 3: Estado del arte

Como se mencionó en la sección 1.2, un simulador solar incluye al menos tres grandes prestaciones:

1. Simulador Solar: Una fuente de luz capaz de simular la condición lumínica bajo la cual opera el satélite.
2. Caja de Helmotz: Una configuración de diodos y resistencias para simular los campos electromagnéticos que percibe el satélite en operación.
3. Rodamiento de aire: Un rodamiento de aire conformado por cilindros presurizados con el fin de eliminar efectos de roce.

A continuación, se presentan algunos simuladores en la industria, se revisan sus características, funcionalidad y el hardware utilizado.

## 3.1 Astro-und Feinwerktechnik Adlershof GmbH

ASTROFEIN es una firma alemana con una amplia experiencia de casi 30 años en el campo de la industria espacial. Ha participado en destacadas misiones, como ROSETTA y BIRD. Entre su amplia gama de productos se incluyen ruedas de reacción, giroscopios, magnetómetros y una cama de pruebas destinada a satélites [19]. La cama de pruebas satelitales, representada en la Figura 3.1, incorpora los tres atributos mencionados al comienzo de este capítulo.

Respecto a la funcionalidad, el rodamiento de aire no solo le permite una libre rotación respecto al eje vertical de 360 [°] sino que también una rotación respecto al eje horizontal de al rededor de 30 [°], esto lo consigue a través de cilindros presurizados con calibración de centro de gravedad y una fuente de poder independiente, además de un sistema de seguridad en caso de superar alguna condición crítica de operación.

Este banco de ensayos cuenta con un simulador de GNSS o "Global Navigation Satellite System", esto consiste en la simulación de luz solar, estrellas, planetas, luz difusa y protones gracias a la potente lampara de halogenuros metálicos como las de la Figura 3.2, esta lámpara está ubicada a 3 [m] del sistema y es capaz de moverse a través de toda la semiesfera mayor del banco de ensayos, además la luz puede ser atenuada hasta un 30 % a través de una interfaz de DMX [20]. La lámpara de halogenuros y el control por software de este banco de ensayos le permite simular una mayor cantidad de escenarios de la operación del satélite.

Figura 3.1: Astro-und Feinwerktechnik Adlershof GmbH [19]

Figura 3.2: Lámpara de Halogenuro metálico.

### 3.2 DLR FACE: Facility for Attitude Control Experiments

El centro aeroespacial alemán o DLR, en su departamento de Guidance, Navigation and Control o GNC cuenta también con una cama de pruebas que incluye un simulador solar. Orientada a "European Small Satellites" de entre 100-200 [kg], el simulador FACE o Facility for Attitude Control Experiments del DLR cuenta con caja de Helmotz, rodamiento de aire y la fuente de luz, además un computador principal y sensores integrados como muestra la Figura 3.3. Esta plataforma sirve como banco de pruebas únicamente para satélites que sean autosustentables energéticamente [21].

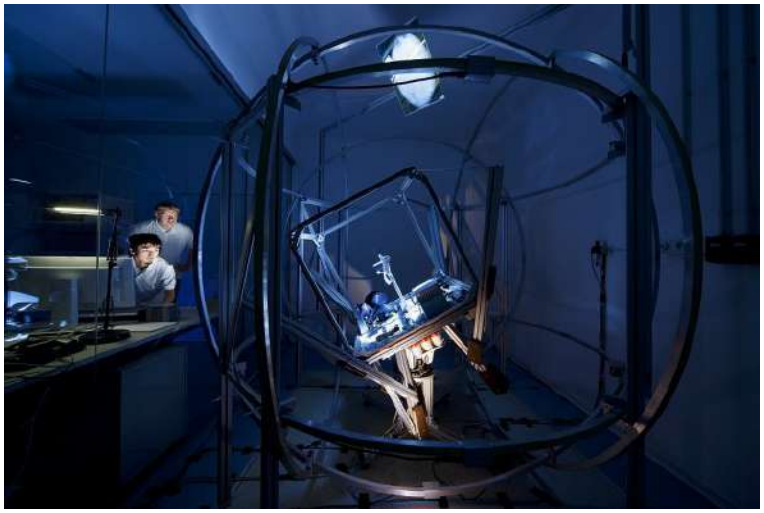


Figura 3.3: Cama de pruebas FACE de DLR [21]

El rodamiento de aire y las cajas de Helmotz funcionan de igual manera que el simulador Astrofein de la Figura 3.1. Respecto al simulador solar, este consiste en una luz con color de temperatura de 3200 [K], la apertura, foco y haz de luz son ajustables. La dirección de la luz respecto al punto de pivote es fija, de todas formas, la luz puede ser reflectada a través de un espejo para simular un mayor rango de misiones.

Una extensión del simulador FACE es el "Star Recognition Hardware" en la Figura 3.4, la extensión está conformada por 24 patrones con 7 LEDs instalados en la circunferencia del simulador que simulan posiciones de estrellas, estos patrones permiten ensayos de actitud cuasi-inerciales útiles para misiones de alta precisión de apuntamiento como telescopios espaciales.


Figura 3.4: Star Simulator en FACE de DLR [21].

### 3.3 Norwegian University of Science and Technology

La Universidad de Ciencia y Tecnología de Noruega, conocida como NTNU, también ha desarrollado una plataforma de pruebas dedicada a los Small Satellites. Esta plataforma surgió como

resultado de un proyecto de tesis en el programa de robótica, en el cual se diseñó e implementó una plataforma de pruebas similar a las dos mencionadas anteriormente. Esta plataforma cuenta con una caja de Helmotz, cojinetes de aire y un simulador solar. En este caso, el simulador solar se presenta como una fuente de luz suspendida a una distancia ajustable, lo que permite controlar la intensidad de la luz que incide sobre los sensores. En la Figura 3.5 se ilustra la plataforma de pruebas de NTNU.

Figura 3.5: Cama de pruebas NTNU [22].

Específicamente, la fuente de luz utilizada para simular el Sol es la linterna igloo de 11.600 [lumen] que se observa en la Figura 3.6, en este caso no hay espejo móvil para reflectar la luz por lo tanto la dinámica de la interacción Sol-satélite se hace únicamente a través del rodamiento de aire y los algoritmos de control del satélite. A pesar de que la fuente de luz es de menor complejidad que otros simuladores, también pudo verificar sensores y poner a prueba los algoritmos del satélite Orbit NTNU, el autor menciona que el único problema al que se enfrentó fue al reflejo de la luz en las ventanas y estructura de aluminio que fácilmente se puede solucionar con recubrimientos de color oscuro [22].

Figura 3.6: Linterna Iiglo 11600 lumen [23].

### 3.4 Sputnix

La compañía Sputnix, de origen ruso, opera en el sector espacial desde 2011 y ha desempeñado un papel protagónico en diversas misiones satelitales. Su participación más notable incluye el envío de dos Cubesats educativos, SiriuSat1 y SiriuSat2 a la Estación Espacial Internacional. Además de sus contribuciones a misiones, Sputnix ofrece una plataforma de pruebas para el subsistema ADCS, la cual se presenta en la Figura 3.7. La empresa define su banco de ensayos como un sistema para ensayos y verificación del subsistema ADCS de pequeños satélites en condiciones de orbitas bajas.

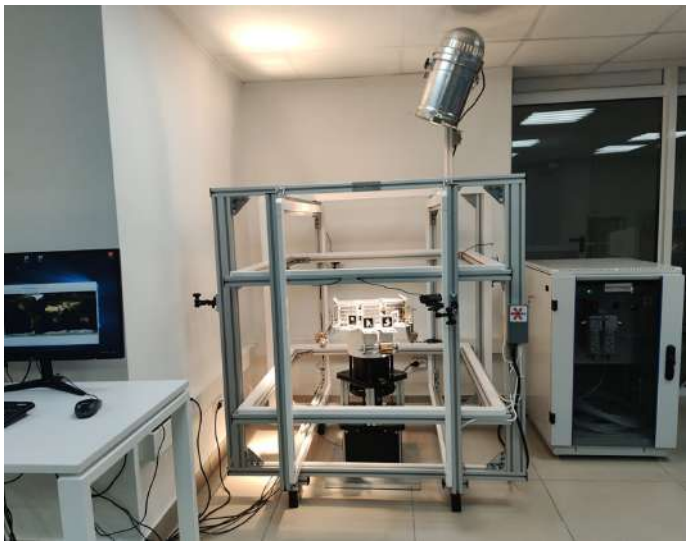


Figura 3.7: Cama de pruebas Sputnix [24].

Dentro de la funcionalidad del banco destacan la elaboración de modos de operación en el software del satélite, determinación experimental del tensor de inercia y sus ejes principales, simulación de condiciones de emergencia y respuesta. El software del banco de ensayos calcula las trayectorias a simular en base al modelo SGP4 e información almacenada en TLE, además la presentación de resultados es a través de los cuaterniones en el sistema de coordenadas orbital [24].

Respecto a la fuente de luz, se identifica una lámpara de halogenuros metálicos como la Figura 3.2 fija en su posición. La luz está a 1.5 [m] de distancia resultando en una iluminación de 90.000 [lux] aproximadamente.

# CAPÍTULO 4: Diseño del banco de ensayos

## 4.1 Resultados del Proyecto de Ingeniería

En el trabajo del Proyecto de Ingeniería [14] se obtuvo el vector Sol en el Bodyframe del satélite, este resultado se consiguió a través del procedimiento descrito mediante el diagrama de bloques de la Figura 4.1. El código escrito en lenguaje Python utiliza una fecha para obtener la efemérides del Sol a través del VSOP87 y un TLE asociado a la misma fecha para propagar con SGP4 como el simulador de Sputnix, con transformaciones asociadas a la mecánica rotacional del satélite se obtienen los resultados en las Figuras 4.2 y 4.3 para el satélite SUCHAI-3 el día 4 de Julio de 2023.

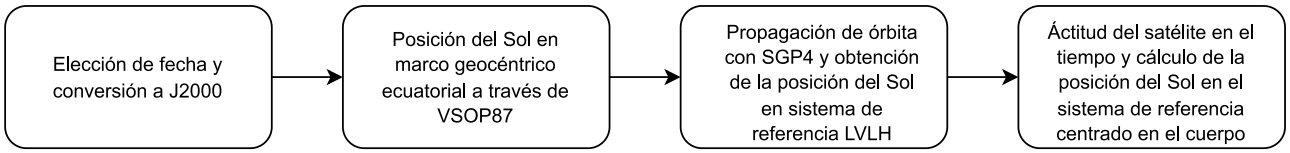



Figura 4.1: Diagrama de bloques para cálculo del vector Sol [14].

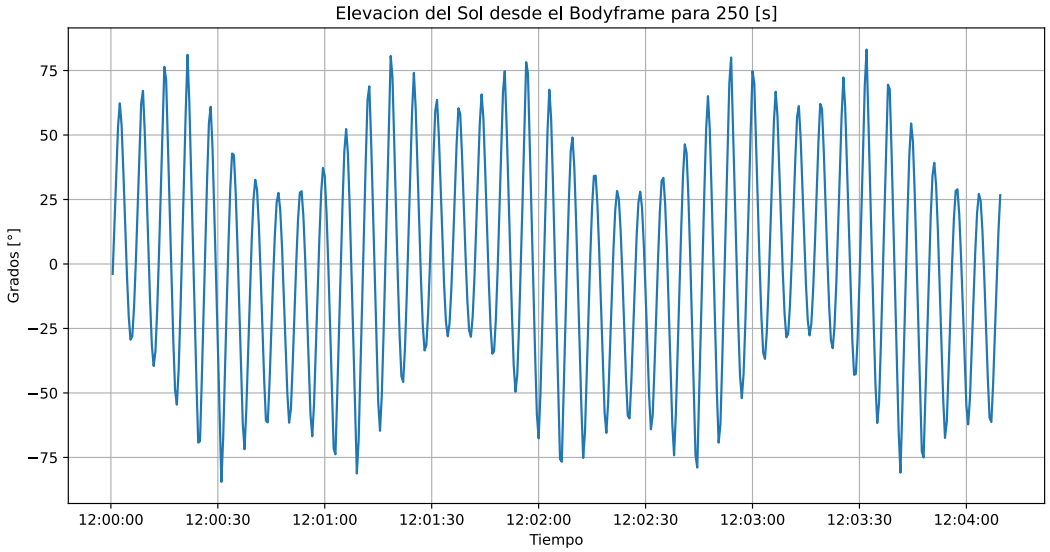



Figura 4.2: Elevación del Sol desde el bodyframe para 250 [s] [14].

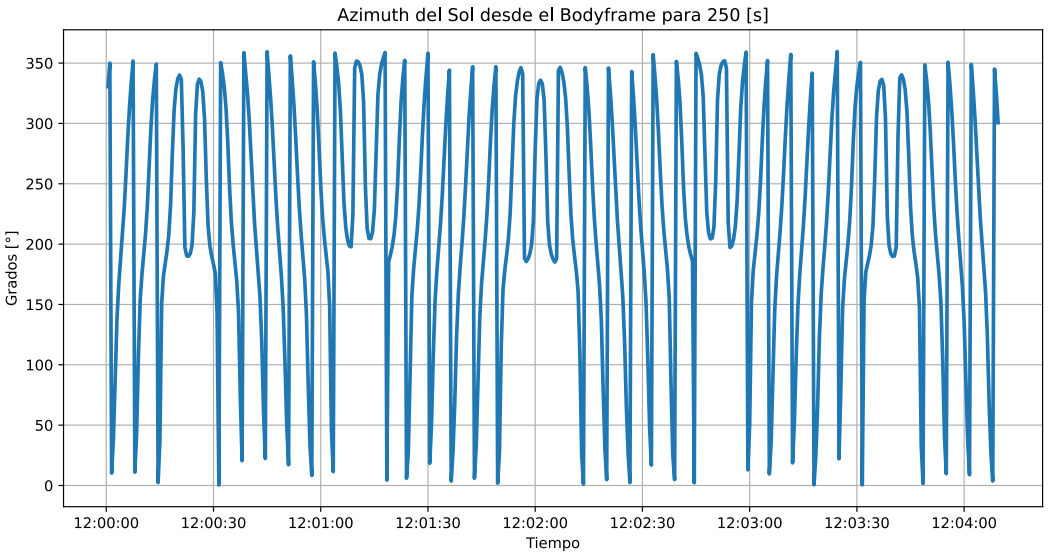



Figura 4.3: Azimuth del Sol desde el bodyframe para 250 [s] [14].

Los resultados de la Elevación oscilan en un rango de $[\frac{-\pi}{2}, \frac{\pi}{2}]$ y para el Azimuth en un rango aproximadamente de [0,2$\pi$]. En coordenadas esféricas esto se traduce en un completo recorrido en las direcciones longitudinales y latitudinales del satélite, suponiendo al satélite como una esfera, en la Figura 4.4 se indica la proyección del vector Sol en los 250 [s].

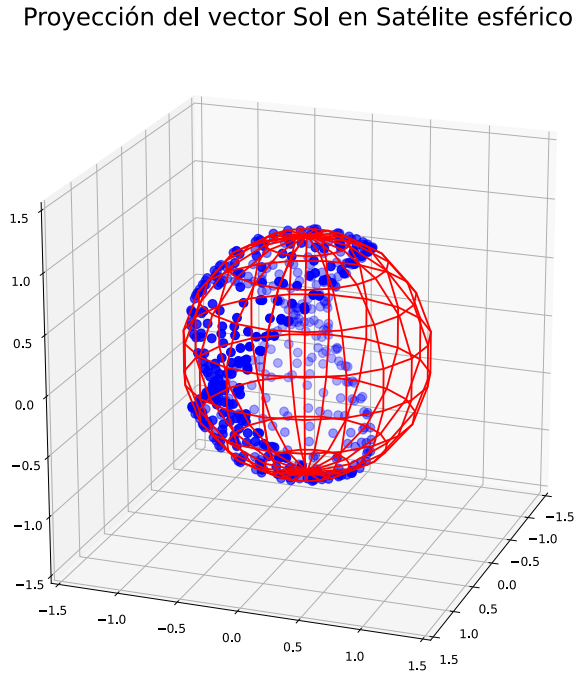



Figura 4.4: Proyección del vector Sol en bodyframe sobre esfera para 250 [s]

Con estos resultados, se requiere que el banco de ensayos pueda proyectar la luz en la totalidad

del dominio de ambas direcciones.

## 4.2 Primeras iteraciones de diseño

En base a los resultados anteriores y los requerimientos de diseño impuestos por los stakeholders, se trabajaron dos propuestas que físicamente eran iguales sin embargo, se diferencian en la fuente de luz que se utiliza. En específico, el banco de ensayos propuesto al final del Proyecto de Ingeniería [14] consiste en la estructura de perfiles de aluminio que sostiene 20 semicircunferencias con múltiples luces LED como se observa en la Figura 4.5, en esta estructura se propone que la fijación del satélite sea a través de piolas de acero.

Las dos alternativas de luces LED consisten en cintas LED de RGB 5050 que se muestran en la Figura 4.6(a) y por otro lado, luces LED COB de 10 [W] como la que se muestra en la Figura 4.6(b). La disposición de las luces al rededor de todo el satélite apuntando al centro, la fijación del satélite y el espacio disponible del banco de ensayos le permiten cumplir con todas las condiciones de diseño mencionadas en la Sección 1.4. A continuación se detalla la disposición de las luces y la resolución en cada dirección para ambas propuestas.

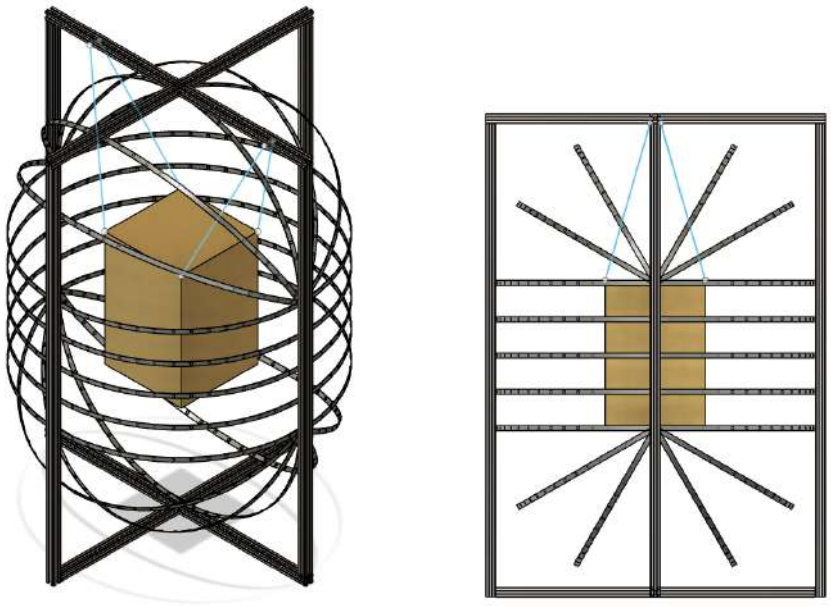


Figura 4.5: Primera iteración de la estructura del banco de ensayos.

(a) WS2815 RGB String


(b) Y32-SY COB LED

Figura 4.6: Alternativas de fuente de luz para el banco de ensayos.

Cada una de las 20 semicircunferencias sostenidas en la estructura tiene un perímetro de 1 [m], de igual forma, cada cinta de luces LED RGB de la Figura 4.6(a) tiene una longitud de 1 [m] y contiene 30 luces LED. Con un total de 600 luces, la propuesta con luces RGB tiene una resolución de 6 [°] en el Azimuth y 18 [°] en la Elevación.

Por otro lado, las luces LED COB son individuales y la cantidad a utilizar está en función del peso que soporte cada semicircunferencia, la potencia para alimentarlas y el costo de adquisición. Para esta propuesta se considera el uso de 10 luces equidistantes en cada semicircunferencia, resultando en una resolución de 18 [°] en el Azimuth y los mismos 18 [°] en la Elevación.

En ambas alternativas, el montaje de las luces que determina la posición y orientación de las mismas, plantea un gran desafío, así como también la programación en el manejo continuo de cientos de luces. Además, en la 2da alternativa debido a la potencia de cada una de las luces, se requiere que cada una de las 200 luces LED sea activada a través de un relay lo que plantea un desafío electrónico de gran escala.

Continuando con la iteración de bancos de ensayo y revisando los resultados y el estado del arte presentado, se plantea una última alternativa. Este diseño utiliza como fuente de luz las luces LED COB de 10 [W] de la propuesta (b) presentada anteriormente. Para simular la trayectoria requerida, se utilizan engranajes que se activan con motores paso a paso, que en conjunto a rodamientos permiten el movimiento continuo e ininterrumpido. Un pedestal sostendrá fijo el satélite al centro del banco de ensayos y los engranajes se encargarán de simular la trayectoria solicitada.

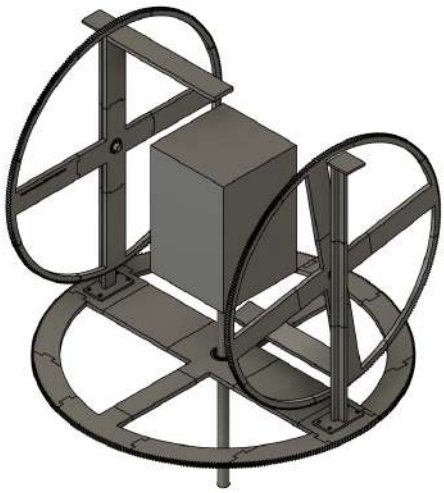


Figura 4.7: Diseño final del banco de ensayos.

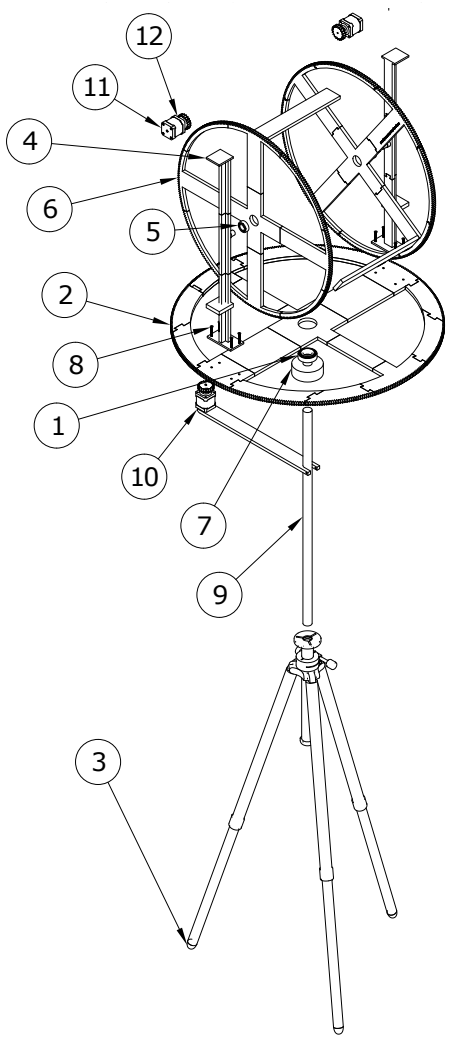
Con esta ultima propuesta se cumplen todas las consideraciones de diseño, proveyendo espacio suficiente para ensayos de Cubesat 12U, una fuente de luz suficiente para la activación de los sensores y de espectro completo como la luz LED COB, además de asegurar la continuidad de los datos utilizando dos engranajes verticales que simulan la totalidad del dominio de la Elevación a pesar del tubo central que tiene como misión priorizar la seguridad del satélite cumpliendo su rol de soporte.

Sin embargo, la propuesta no está exenta de desafíos. El movimiento del sistema se origina con el contacto entre engranajes por lo que asegurar el traspaso de movimiento se convierte en una prioridad. Además, el sistema apoya su operación en los rodamientos, de tal forma que su montaje también es crítico para el banco de ensayos. Cualquier procedimiento de montaje o diseño mal ejecutado resultará en efectos negativos para el sistema, entre ellos, vibraciones, desbalanceamiento o discontinuidad.

A continuación se detalla la geometría del banco de ensayos y como a través de los componentes y su ensamblaje se planea sortear las dificultades propias de la propuesta.

## 4.3 Diseño final y componentes

En la Figura 4.8 se muestra el plano general del banco de ensayos y un listado de sus componentes.




| Lista de piezas | | |
|---|---|---|
| elemento | ctd | número de pieza |
| 1 | 1 | Rodamiento WOKOST 30205 |
| 2 | 1 | Engranaje Horizontal PLA |
| 3 | 1 | Tripode |
| 4 | 2 | Soporte Vertical PLA |
| 5 | 2 | Rodamiento AOF 6002 2RS |
| 6 | 2 | Engranaje Vertical PLA |
| 7 | 1 | SlipRing 6 Canales |
| 8 | 8 | Perno M4 |
| 9 | 1 | Tubo Aluminio 25 mm |
| 10 | 1 | Soporte Horizontal NEMA |
| 11 | 3 | NEMA 17 |
| 12 | 3 | Engranaje NEMA |

Figura 4.8: Vista desmontada del banco de ensayos.

A continuación se detallan los componentes utilizados y los argumentos en su elección.

1. Tubo de Aluminio 25 [mm]: Se elige un tubo de aluminio para soportar el satélite y la estructura al rededor. Se elige este material por su bajo costo económico comparado a otros metales, por su disponibilidad y por su resistencia a temperatura que facilitará el montaje y durabilidad.

2. Engranaje Horizontal: Un engranaje de 370 dientes y diámetro total de 744 [mm]. El engranaje es de PLA y será impreso en las impresoras del Laboratorio de Fabricación del DIM.

3. Engranaje vertical: Dos engranajes de 290 dientes y un diámetro de 584 [mm]. Este engranaje posee el soporte para la luz COB de espectro completo que simulará el Sol. De igual manera el engranaje será fabricado en los laboratorios del departamento.

   El extremo del tubo de aluminio está 18 [cm] por sobre el engranaje horizontal, esta es la distancia máxima para alinear al centro un Cubesat de 12U. Por lo anterior, no es suficiente con un engranaje vertical para cumplir el objetivo de recorrido total en la Elevación, debido

a colisiones del soporte de luz del engranaje con el tubo de aluminio. Para subsanar las colisiones, se utilizan dos engranajes verticales donde cada engranaje vertical será responsable del movimiento en una mitad del dominio.

4. Engranajes NEMA: Los engranajes NEMA son los engranajes que se montan sobre los motores para la generación del movimiento, este engranaje posee 20 dientes dando un diámetro de 44 [mm].

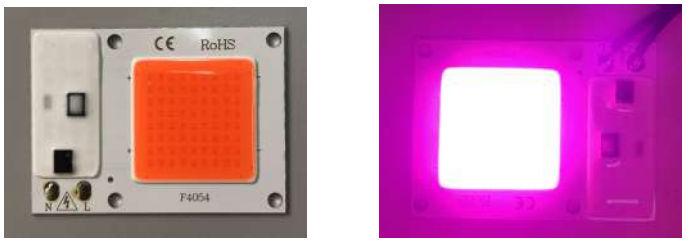
   Todos los engranajes son de módulo 2, tienen un ángulo de presión de 20 [°] y tienen 10 [mm] de espesor, la fabricación de los engranajes se hace en el laboratorio del departamento en las impresoras Ender 3 V2 disponibles, por lo tanto, los engranajes se dividen en las partes necesarias para su impresión y su posterior unión se hace a través de trapecios coincidentes entre las piezas.

5. Motores paso a paso NEMA 17: Un motor paso a paso es un motor que funciona a través de bobinas y pulsos para generar movimiento, a diferencia de un motor servo, los motores paso a paso no tienen límite de recorrido en sus revoluciones. Para el banco de ensayos se eligen los motores NEMA 17 con un torque de 56 [Ncm], que en conjunto a los rodamientos, son los responsables de generar el movimiento. Como se mencionó, estos motores funcionan en base a pasos, en este caso, el motor NEMA 17 realiza 200 pasos en una revolución, sin embargo, con los drivers correspondientes este valor puede aumentar hasta 6400 pasos por revolución [25] dando como resultado un movimiento de mayor resolución.

6. Rodamiento WOKOST 30205: Para soportar las cargas radiales propias del movimiento del banco de ensayos y la carga axial producto del peso de la estructura sobre el engranaje horizontal, se elige un rodamiento de rodillos cónicos. El rodamiento WOKOST 30205 es un rodamiento de rodillos cónicos que soporta cargas dinámicas de 33.500 [N] [26], cuenta con un diámetro interior de 25 [mm] y un diámetro exterior de 52 [mm] como se muestra en el Anexo C. Estas medidas son importantes a considerar en el diseño del engranaje horizontal para el posterior montaje.

7. Rodamiento AOF 6002 2RS: En ausencia de cargas axiales, se opta por un rodamiento rígido de bolas. En este caso, el rodamiento AOF 6002 2RS es un rodamiento rígido de bolas que soporta cargas dinámicas de hasta 5.850 [N], con un diámetro interior de 15 [mm] y un diámetro exterior de 32 [mm] [27], en el Anexo C se encuentra un diagrama del rodamiento. Además el 2RS indica la presencia de sellos en las bolas que prolongan la vida útil del rodamiento manteniendo el lubricante dentro del rodamiento y protegido de basura externa.

8. Luz LED COB de espectro completo: La fuente de luz elegida para simular el Sol es la luz LED COB o Chip On Board de espectro completo de la Figura 4.9. Gracias a su característica de espectro completo, es capaz de simular el espectro que emite el Sol en su totalidad, como se observa en la Figura 4.10. La luz tiene una potencia de 10 [W] y se conecta directo a línea de 220 [V], su peso de 6,6 [g] no representa un gran desafío para la estructura y el soporte en el engranaje vertical.




(a) Luz apagada (b) Luz prendida

Figura 4.9: Luz LED COB de 10 [W] de espectro completo.

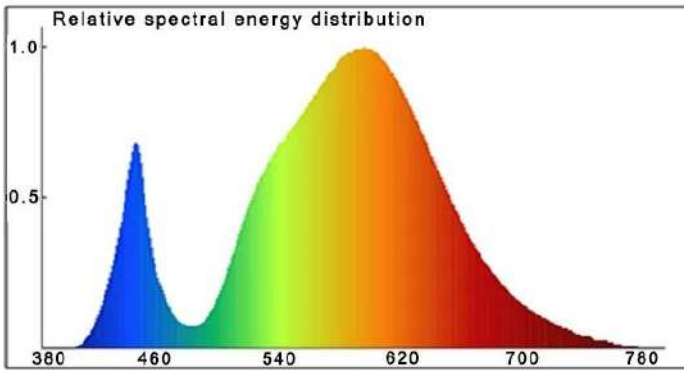



Figura 4.10: Espectro de emisión de la luz LED COB.

9. Slip Ring de 6 canales: Debido a la rotación del engranaje horizontal y la necesidad de alimentar energéticamente los motores se necesita de un slip ring. Un Slip Ring es un anillo deslizante que permite la rotación de cables sin enredarse y/o cortarse, en este caso son 6 cables que se pueden conectar en total, como se muestra en la Figura 4.11.


Figura 4.11: Slipring de 6 canales diámetro interior de 25,4 [mm] [28].

10. Soportes vertical y horizontal: Los soportes vertical y horizontales son los soportes para los motores paso a paso. En el soporte vertical se instala también el rodamiento AOF para permitir la rotación del engranaje vertical, este soporte vertical se une al engranaje horizontal

con 4 pernos en su base. Por su parte, el soporte horizontal solo cumple la función de soportar al motor que acciona el engranaje horizontal, un extremo del soporte se une al tubo de aluminio con un perno pasante que presione el extremo con semicircunferencia.

En los Anexos se encuentran los planos de los engranajes y los soportes tanto vertical como horizontal.

### 4.4 Presupuesto

Con la definición del diseño y los componentes, es posible realizar el presupuesto para ser presentado a las autoridades del Departamento de Ingeniería Mecánica de UdeC. La electrónica necesaria se compra internacionalmente en una gran compra para abaratar los costos de envío, en esta compra se consideran más items de los necesarios en caso de fallas de los equipos, luego, componentes como motores, PLA y rodamientos se adquieren en territorio nacional priorizando la llegada temprana de los materiales antes que abaratar costos.

Tabla 4.1: Presupuesto del Simulador Solar.

| Item | Cantidad | Precio Unitario [CLP] | Envío [CLP] | Precio Total [CLP] |
|---|---|---|---|---|
| Nema 17 56 [N.cm] | 3 | $15.990 | $7.400 | $55.370 |
| Arduino UNO R3 | 1 | | | LTA (*) |
| Rodamiento WOKOST 30205 | 1 | $8.000 | 0 | $8.000 |
| Rodamiento AOF 6002 2RS | 2 | $3.900 | 0 | $7.800 |
| Slip Ring de 6 canales | 1 | | | $101.000 |
| A4988 Driver+Shield | 6 | | | |
| COB LED de espectro completo | 5 | | | |
| PIN Photodiode BPW34 | 10 | | | |
| Cables | | | | LTA (*) |
| Tubo de aluminio 25 [mm] | 1 | | | LTA (*) |
| Tripode | 1 | $11.990 | 0 | $11.990 |
| PLA 1 [Kg] | 4 | $12.390 | 0 | $49.560 |
| Costo total | | | | $233.720 |

(*) Los items donde el costo es LTA, son items que están disponibles en el Laboratorio de Técnicas Espaciales por lo tanto no tienen un costo directo en este proyecto.

# CAPÍTULO 5: Montaje del banco de ensayos

El espacio físico asignado para el montaje del banco de ensayos es en la sala de electrónica del Laboratorio de Técnicas Aeroespaciales, la decisión se toma en función del espacio y el acceso a materiales disponibles. Además se tuvo en cuenta el fácil acceso a línea eléctrica y la baja concurrencia a la sala permitiendo el cuidado del simulador mientras no esté operativo. El montaje del banco de ensayos comienza una vez todas las piezas de PLA han sido impresas.

A continuación se hace un breve repaso al procedimiento de impresión y las configuraciones utilizadas en la impresora para luego describir el procedimiento de montaje del banco de ensayos.

## 5.1 Impresión de engranajes de PLA

Tal como se mencionó en el capítulo anterior, la fabricación de los engranajes y soportes se realizan en el Laboratorio de Fabricación del DIM en las impresoras Ender 3 V2.

Las impresoras Ender 3 V2 poseen una cama de impresión de tamaño 235 x 235 x 250 [mm], dado que los engranajes superan estas dimensiones, es necesario dividir en piezas que quepan en las impresoras Ender 3 V2. Luego, para la unión entre ellas, se utilizan volúmenes trapezoidales coincidentes que encajan las piezas unas con otras para generar el total del engranaje.

Las dimensiones de los trapecios se indican en las Figuras 5.1, 5.2 y 5.3 a continuación.

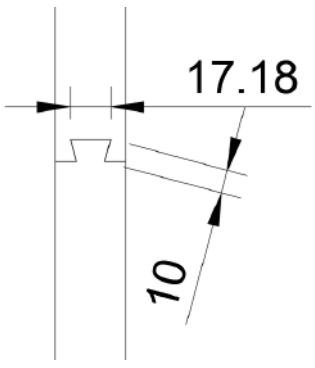



Figura 5.1: Trapecios en el soporte vertical.

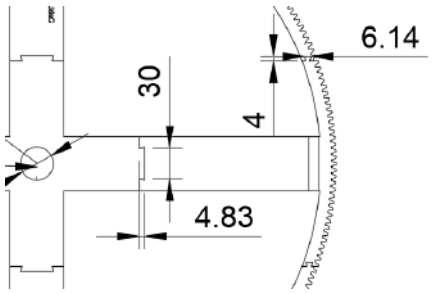



Figura 5.2: Trapecios en el engranaje vertical.

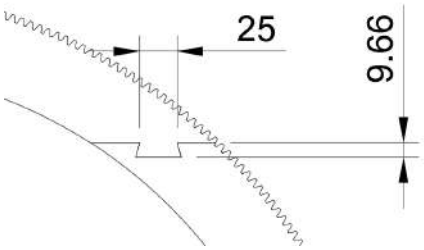



Figura 5.3: Trapecios en el engranaje horizontal.

La Figura 5.2 indica las dimensiones de los trapecios para los engranajes verticales, donde la

preocupación es únicamente el correcto encaje de los dientes. Las Figuras 5.3 y 5.1 indican las dimensiones de los trapecios en los engranajes horizontales y soportes, estos trapecios son más grandes que los trapecios del engranaje vertical debido que los soportes y engranaje horizontal cargan con el peso de la estructura y motores paso a paso por lo que se requiere una mayor área resistente.

Además, considerando las cargas sobre el engranaje horizontal, en la pieza central del engranaje se utilizan rectángulos embutidos como los que muestra la Figura 5.4.

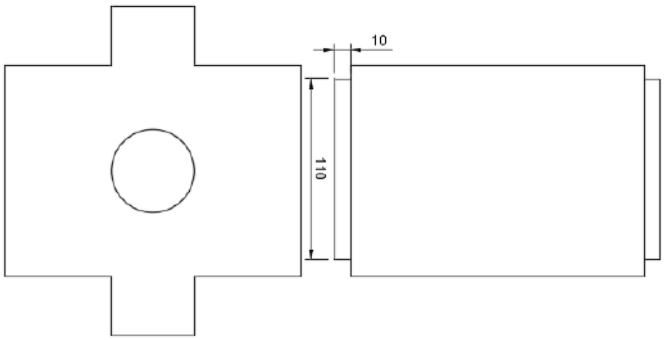



Figura 5.4: Rectángulos embutidos en la pieza central del engranaje horizontal.

Las configuraciones utilizadas en las impresoras se muestran en la Tabla 5.1, se utiliza una densidad del 30 % buscando una combinación entre las horas de impresión y el material utilizado en la pieza que determina la resistencia a través de la forma de relleno. En los casos donde sea necesario se utilizan soportes y en ninguna de las piezas se considera una película de adhesión.

Tabla 5.1: Configuraciones de impresión.

| Configuración | Valor |
|---|---|
| Densidad | 30 % (*) |
| Velocidad | 60 $[\frac{mm}{s}]$ |
| Soportes | Solo cuando se necesite. |
| Adhesión | Nunca |
| Lineas de pared | 3 (**) |
| Calidad | 0.2 [mm] (calidad estándar) |

(*) Para pieza central del engranaje horizontal, se opta por 45 %.
(**) Para pieza central del engranaje horizontal, se usan 5 lineas de pared en vista de la instalación de la carcasa del rodamiento.

## 5.2 Montaje

Con las piezas impresas y el encaje verificado, comienza la preparación del resto de componentes para el montaje total. El tubo de aluminio tiene un diámetro exterior de 1 [in] o 25,4 [mm] y el rodamiento de rodillos cónicos tiene un diámetro de 25 [mm] en su interior, la diferencia de 0,4 [mm] es considerablemente mayor para montar el rodamiento con ayuda de procesos térmicos por

lo que se necesita de un desgaste en la sección de interés del tubo. Con la ayuda del torno y de lijas de metal se desgasta una longitud de 19 [cm] en un extremo del tubo para el montaje del rodamiento, obteniendo el desgaste de la sección superior que se observa en la Figura 5.5 con un diámetro final de 25,1 [mm], de esta forma el montaje del rodamiento se logra con la utilización de lubricante y pistola de calor. En la Figura 5.5 se observa el rodamiento de rodillos montado, junto con el slip ring y el soporte horizontal en posición, estos dos últimos no requieren trabajos adicionales en el tubo para el montaje dado que sus diámetros son mayores a 25,4 [mm].

Figura 5.5: Desgaste en la sección de interés del tubo de aluminio

La carcasa del rodamiento de rodillos y los rodamientos de bolas se montan directamente a las piezas de impresión 3D, donde los orificios han sido considerados en la impresión de tal forma que el montaje se logra únicamente disminuyendo la temperatura de las piezas metálicas para su contracción y posterior montaje en las piezas plásticas. Las piezas centrales de los engranajes con sus rodamientos montados se muestran en la Figura 5.6.

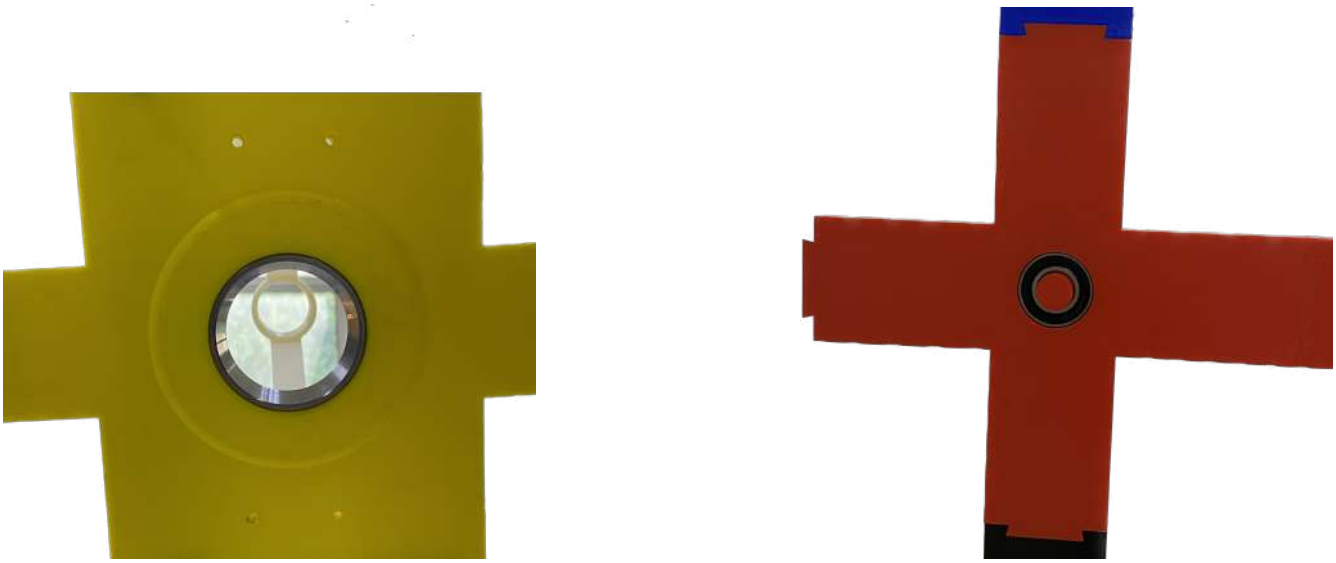


Figura 5.6: Piezas centrales de los engranajes con carcasa y rodamiento instalados.

Una vez montados los rodamientos en los engranajes y en el tubo de aluminio central, se puede montar el banco de ensayos para revisión del estado estructural y eventuales mejoras en esta misma.

## 5.3 Evaluación y mejoras

Con el banco de ensayos montado, se pueden incluir los motores paso a paso en el montaje que representan la carga principal del banco de ensayos. Como resultado, se observa flexión en los extremos del engranaje horizontal que puede derivar en perdida de contacto con el engranaje NEMA o incluso ruptura de la pieza, por lo tanto se debe aumentar la rigidez del engranaje horizontal.

Para aumentar la rigidez, se añade una barra de aluminio de extremo a extremo conectando ambos soportes verticales. Durante revisiones, se descubrió que el mayor problema de la rigidez era en las secciones de unión entre las piezas, es decir en los trapecios y las secciones de encaje, por lo tanto la barra de aluminio se acopla al engranaje horizontal con pares de pernos M3 en cada extremo de las piezas que lo conforman como se muestra en la Figura 5.7. Los pernos se acompañan de sus respectivas tuercas y golillas de presión para el apriete, con el fin de que la rigidez del sistema sea responsabilidad de la barra de aluminio y los trapecios sean únicamente responsables del encaje y unión de las piezas, además de permitir el eventual reemplazo de alguna de las piezas.

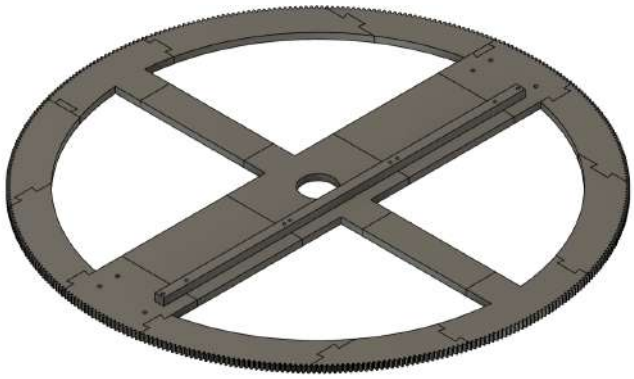


Figura 5.7: Engranaje horizontal con rigidizador.

Por otro lado, el rodamiento de rodillos cónicos utilizado no posee seguros por lo que la estructura completa se desmontaba si una fuerza se aplicaba en un extremo de la estructura. Como solución, se fabrica el seguro de la Figura 5.8, que se instala a través del tubo de aluminio y también con pernos M3, tuercas y golillas de presión al engranaje horizontal.

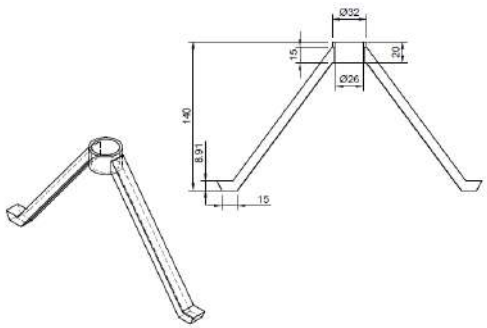



Figura 5.8: Seguro para rodamiento de rodillos.

En la Figura 5.9 se muestra el banco de ensayos con su estructura completamente montado y un acercamiento al engranaje horizontal para mostrar las mejoras descritas anteriormente.


(a) Simulador Solar.

(b) Rigidizador en engranaje horizontal.

Figura 5.9: Banco de ensayos montado y detalle de rigidizador.

Con las mejoras, se instalan nuevamente los motores paso a paso y como resultado se observa un evidente aumento en la rigidez y estabilidad de la estructura.

# CAPÍTULO 6: Circuito electrónico

Una vez finalizada la estructura, se definen los componentes electrónicos para el funcionamiento de los motores paso a paso y la activación de las luces LED. En este capítulo se repasan los componentes y sus conexiones para instalarse en el banco.

## 6.1 Componentes electrónicos

En el Capítulo 4 se mencionaron los componentes electrónicos como Arduino R3, Motores NEMA 17 con sus driver y las luces LED COB, sin embargo, estos componentes requieren de entradas de energía de diferente magnitud. A continuación se agrupan los componentes según la entrada de voltaje que requieren y la fuente a utilizar según sea el caso.

1. Voltaje de 220 [V]

   En el banco de ensayos, los únicos componentes que requieren de voltaje de 220 [V] son las luces LED COB, sin embargo, también se alimenta de 220 [V] el transformador que se describe a continuación.

2. Voltaje de 12 [V]

   Los componentes que requieren de 12 [V] para su funcionamiento son los motores y el Arduino, para alimentarlos se opta por una fuente de poder que utilice como entrada 220 [V] y tenga de salida los 12 [V]. Además, la fuente de poder debe suministrar la corriente necesaria para alimentar los motores de manera simultanea. Con el fin de dimensionar y seleccionar la fuente de poder, se calcula el consumo de corriente por parte de los motores, para ello se busca el consumo de un motor NEMA 17 que según [29] es de 1.7 [A] y por lo tanto el consumo de tres motores es de

   $$I_{total} = 5,1[A].$$

   Finalmente la fuente de poder debe ser capaz de entregar un voltaje de 12 [V] y una corriente de 5.1 [A]. Para ello, se elige la fuente en la Figura 6.1 que entrega una corriente de 6 [A].

Figura 6.1: Transformador 12 [V]-6 [A] AC/DC.

3. Voltaje de 5 [V]

   Por otro lado, los componentes que requieren de 5 [V] son los driver del motor paso a paso y los relay que activan las luces LED. Dado que las fuentes de energía disponibles en el banco de ensayos son de 220 y 12 [V], se necesita una nueva fuente de energía o reducir el voltaje de alguna de ellas, la primera alternativa es poco eficiente dado el reducido espacio en el banco, por lo tanto se opta por la reducción de voltaje desde una de las fuentes. Un Buck es un dispositivo electrónico utilizado para la reducción de voltaje utilizando sus resistencias y potenciómetros integrados. El dispositivo adquirido para el banco de ensayos es el de la Figura 6.2 que permite una reducción de 12 [V] a 5 [V].

Figura 6.2: Buck para reducción de voltaje.

## 6.2 Actualización del presupuesto

A continuación se muestra la Tabla 6.1 con el presupuesto actualizado del banco de ensayos. En ella se añaden los componentes electrónicos descritos anteriormente, pernos y tuercas M3 y M5 adquiridas para la rigidización y finalmente la adquisición de un segundo rodamiento de rodillos cónicos WOKOST debido a la ruptura del rodamiento durante el montaje.

Tabla 6.1: Presupuesto del Simulador Solar.

| **Item** | **Cantidad** | **Precio Unitario** [CLP] | **Envío [CLP]** | **Precio Total [CLP]** |
|---|---|---|---|---|
| Nema 17 56 [N.cm] | 3 | $15.990 | $7.400 | $55.370 |
| Arduino UNO R3 | 1 | | | LTA (*) |
| Rodamiento WOKOST 30205 | 1 | $8.000 | $0 | $8.000 |
| Rodamiento WOKOST 30205 | 1 | $7.000 | $0 | $7.000 |
| Rodamiento AOF 6002 2RS | 2 | $3.900 | $0 | $7.800 |
| Slip Ring de 6 canales | 1 | | | $101.000 |
| A4988 Driver+Shield | 6 | | | |
| COB LED de espectro completo | 5 | | | |
| PIN Photodiode BPW34 | 10 | | | |
| Cables | | | | LTA (*) |
| Tubo de aluminio 25 [mm] | 1 | | | LTA (*) |
| Tripode | 1 | $14.190 | $0 | $14.190 |
| PLA 1 [Kg] | 4 | $12.390 | $0 | $49.560 |
| Transformador 220 [V]-12 [V] AC/DC | 1 | $16.990 | 0 | $16.990 |
| Buck 12 [v]-5 [V] | 1 | $9.210 | $0 | $9.210 |
| Pernos y tuercas | 12 | $2.598 | $0 | $2.598 |
| Sueldo Ingeniero Civil Aeroespacial (**) | 3 [meses] | $1.200.000 | 0 | 3.600.000 |
| Costo total | | | | $3.871.718 |

(*) Los items donde el costo es LTA, son items que están disponibles en el Laboratorio de Técnicas Espaciales por lo tanto no tienen un costo directo en este proyecto.
(**) El costo del ingeniero ha sido calculado como el tiempo en que un ingeniero se demoraría con todos los materiales e información a disposición.

## 6.3 Conexiones del circuito

En la Figura 6.3 se indican las conexiones del circuito.

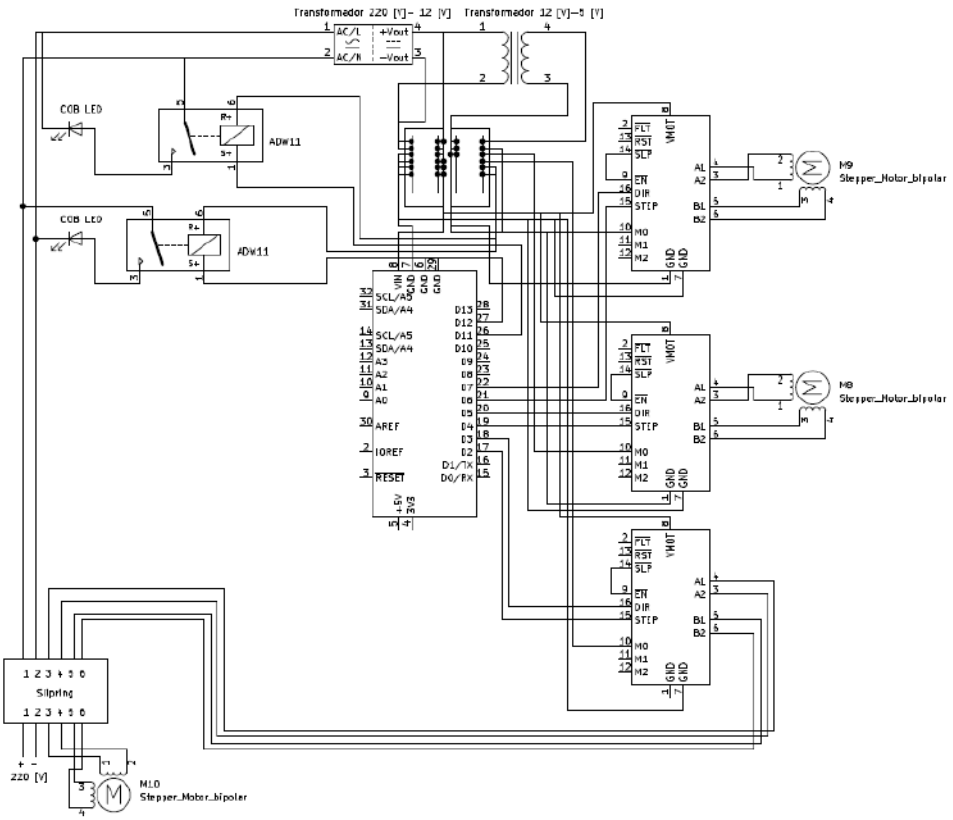



Figura 6.3: Circuito electrónico del banco de ensayos.

A continuación se explican las conexiones y los pines utilizados.

1. Placa de soldadura: Se utiliza una placa de cobre de soldadura para ordenar y distribuir el circuito del banco de ensayos. Una sección de la tarjeta está energizada con 12 [V] y otra sección con 5 [V] para mantener el orden de las conexiones.

2. Arduino UNO R3: Al centro del circuito se observa la placa Arduino energizada con 12 [V] a través del Pin Barrel Jack para evitar caídas de voltaje. Para la conexión con los componentes al rededor, se usan 8 pines digitales elegidos de manera óptima para el montaje.

3. Transformador 220 [V]-12 [V] AC/DC: En la parte superior, el transformador de 220 [V] a 12 [V] con el que se alimentan los motores paso a paso.

4. Buck 12 [V]-5 [V]: Al lado del transformador AC/DC, se encuentra el buck de 12 [V] a 5 [V] con el que se alimentan relays y drivers.

5. Driver DR 8825: A la derecha del circuito, los tres driver y los motores paso a paso. Las conexiones del driver consisten en cuatro pines del motor al driver, luego, en el driver los pines STEP y DIR conectados al Arduino que controlan los pasos y la dirección, respectivamente. En los pines M0 se conecta el cable de voltaje de 5 [V] y la tierra al respectivo GND, finalmente los pines SLP y EN se conectan entre sí. Los pines que controlan a los motores desde el Arduino son los pines D2 a D7.

6. Relay y LED COB: A la izquierda del circuito se encuentran los relays y luces LED COB. Las Luces conectadas en línea directa a 220 [V] pasando por el relay para la correcta activación de las luces. Por su parte, los relays conectados a los pines D11 y D12 del Arduino.

7. Slipring: A la esquina inferior izquierda, el SlipRing, de sus seis canales, dos son utilizados para el cable con 220 [V] de entrada a la fuente de poder y los restantes cuatro se usan para el motor del engranaje horizontal que permanece estático.

Respecto a la consideración de diseño sobre la visualización de datos en tiempo real, no es posible realizar la tarea con el módulo bluetooth adquirido. En específico, se utilizó el módulo bluetooth HC-06, durante las pruebas de su funcionamiento no se pudo asegurar una conexión rápida y estable al ordenador, por lo tanto se decide no utilizarlo y visualizar los datos de otra manera.

Finalmente, para recopilación de resultados y una eventual carga de archivos a las plataformas Arduino, se usan módulos Shield para tarjetas SD como el de la Figura 6.4. Este tipo de módulos se ancla al Arduino y permite mantener todas las conexiones descritas anteriormente, a excepción de los pines D9 y D10 que son los pines utilizados para conectarse a la tarjeta SD.

Figura 6.4: Módulo Shield SD para Arduino UNO.

## 6.4 Montaje en la estructura

Para el montaje, se busca una forma eficiente de instalar los componentes en el banco de ensayos buscando que no afecte al movimiento de los engranajes, no genere sombra hacia el centro del banco de ensayos, cuidando el orden de los cables y el fácil acceso en caso de cualquier inconveniente. Considerando lo anterior se decide que la tarjeta de soldadura, la fuente de poder, el buck y los driver se ubiquen en la cara inferior del engranaje horizontal. El Arduino y los relay se ubicarán en la cara superior del engranaje horizontal, el Arduino al centro de la estructura y los relay en la base de los postes para conectarse a las luces LED.

Para la soldadura e instalación de los componentes se desmonta el banco de ensayos para tener el engranaje horizontal separado del resto de piezas. A la tarjeta se sueldan pines hembra para facilitar las conexiones, una vez soldados todos los pines macho y hembra según correspondan se ubican y pegan definitivamente los componentes en su posición para definir el largo de los cables, por último se rotulan los cables con el fin de identificar su función, voltaje y destino.

Específicamente los componentes en la cara inferior quedan ubicados según como indica el esquema digital en la Figura 6.5.

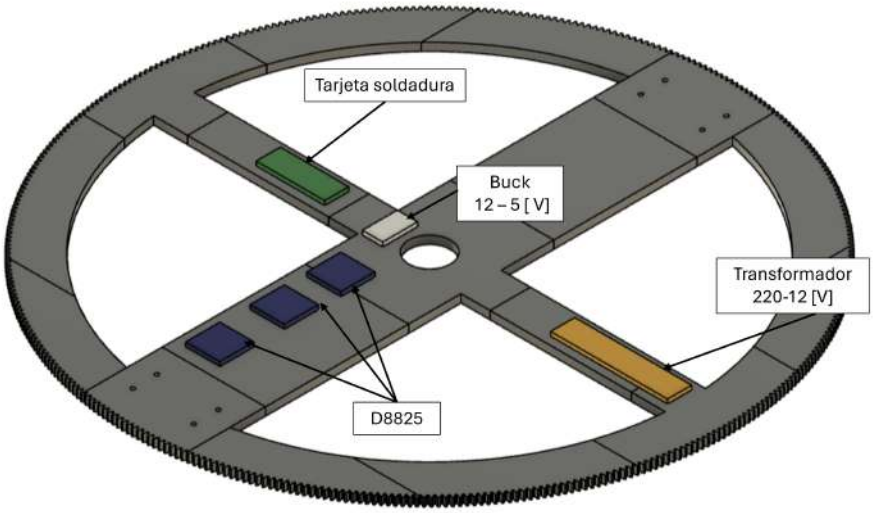



Figura 6.5: Montaje electrónico en la cara inferior del banco de ensayos.

## 6.5 Mejoras estructurales y electrónicas

Luego del montaje se prueban los diferentes componentes para revisar el estado de su conexión, el comportamiento bajo largos periodos de actividad y los resultados de los movimientos propios del sistema. Para ello se compila al microcontrolador Arduino un código tal que cada uno de los motores completa dos revoluciones en una dirección con ambas luces LED encendidas, luego una pausa de un segundo y finalizando con dos revoluciones en sentido contrario con las luces apagadas, de esta forma se prueban todos los componentes y se evalúa la estructura luego de la acción de los motores.

Estructuralmente el banco de ensayos responde de buena manera, sin embargo, la posición del transformador 220-12 [V] genera flexión en la sección del engranaje horizontal provocando que el engranaje horizontal pierda contacto con el respectivo engranaje NEMA del motor. Por lo anterior, se aumenta el espesor de los 3 engranajes NEMA pasando de 10 [mm] a 30 [mm]. Se decidió cambiar los 3 engranajes de inmediato para evitar futuras modificaciones.

Por otro lado, electrónicamente el banco de ensayos no responde de manera correcta a las ordenes en el código del microcontrolador. Uno de los motores verticales realiza el doble de vueltas y en ocasiones los tres motores sufren de pequeñas pausas provocando que los motores no terminen las revoluciones en los lugares esperados. Además, tanto motores como drivers sufren de sobrecalentamiento. Todos los síntomas anteriores son un problema para la correcta operación del banco de ensayos dado que provocarán imprecisión en los movimientos e interrupción en la operación del banco.

Con las fallas identificadas se procedió a la inspección de los motores, los driver y el microcontrolador Arduino. Entendiendo el funcionamiento de un motor paso a paso [25], estos motores no

tienen referencia y solo obedecen a señales y/o pulsos emanados del driver, además, considerando experiencia previa con motores del proveedor CIMECH, se descarta que el motor paso a paso sea el origen de los problemas descritos.

Continuando, se revisan los driver 8825 adquiridos en AliExpress. Los driver constan de dos componentes, el driver y el shield con disipador de calor pasivo. Para los driver se lleva a cabo una inspección de dos etapas, la primera consiste en una inspección física de los componentes revisando su temperatura al tacto, sus reguladores de corriente y conexiones. La segunda etapa consiste en la revisión con un pulsómetro de los pulsos recibidos y emanados por el driver. A continuación se indican los hallazgos de cada una de las etapas de inspección.

- Inspección física: Durante la inspección física se revisan los reguladores de corriente del driver que son solo tornillos que se regulan al tacto, sin referencias ni topes. Las dificultades para regular la corriente generan sobrecalentamiento en los driver, incluso, la regulación de un driver afecta a la regulación y operación de los otros driver generando corto circuitos en algunas ocasiones. Ninguno de los fenómenos descritos debería ocurrir en la operación del banco de ensayos. Respecto a conexiones y configuración de microstepping no se encontró nada fuera de lo normal.
- Inspección electrónica: Con ayuda de un osciloscopio, se revisan las señales que emanan del driver y se dirigen al motor y aquellas que se emanan del microcontrolador y son recibidas por el driver. Para esta inspección se busca el equilibrio en la regulación de corriente de los driver y se opera aún con el motor que realiza el doble de revoluciones. La inspección a la salida del driver demuestra que la señales no son las escritas en el código del microcontrolador, en ocasiones no corresponden en magnitud pero en ningún momento corresponden en el tiempo de acción de la señal, lo que genera el sobrecalentamiento del motor al ser señales imposibles de obedecer. Por su parte, las señales que recibe el driver son las señales correctas, sin embargo aquí se observan las pausas indebidas que se generan en los motores perdiendo su punto final.

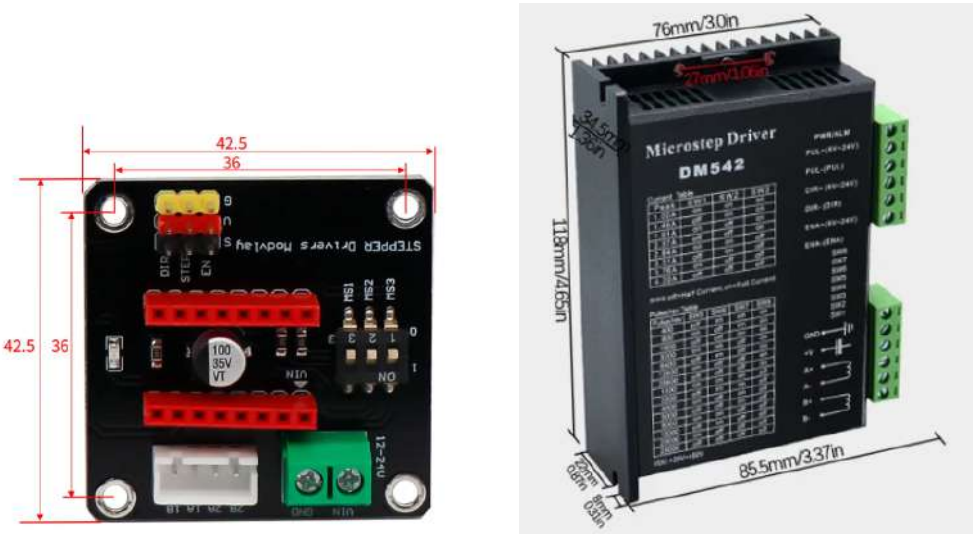
En conclusión, la incorrecta operación de los motores es responsabilidad compartida entre los driver 8825 y el microcontrolador Arduino, ambos componentes serán sustituidos en el banco de ensayos y las conexiones electrónicas se realizarán nuevamente para una instalación limpia.

El microcontrolador Arduino UNO se reemplaza por el mismo modelo pero esta vez por la versión original del componente, por su parte los driver 8825 serán reemplazados por el driver DM542. Estos nuevos componentes estaban disponibles en el LTA por lo tanto no afecta al presupuesto del banco de ensayos.

### 6.6 Nuevo montaje en la estructura

El driver DM542 en la Figura 6.6.b cuenta con algoritmos de control avanzado “DSP” o Digital Signal Processing, además cuenta con sistemas de identificación automática del motor y autoconfiguración de los parámetros. Por otro lado, la corriente peak se configura con switches que permiten 8 diferentes valores de corriente, de la misma forma, los switches de microstepping permiten una resolución de hasta 25600 pasos por revolución [30]. En conclusión, el driver DM542 ofrece mejores

prestaciones, es más estable y más seguro frente a sobrecalentamientos y flujos de corriente, sin embargo, es de considerable mayor tamaño que el driver 8825 como se observa en las cotas de la Figura 6.6 lo que obliga a replantear la posición de los componentes dentro del banco de ensayos.




(a) Driver D8825. (b) Driver DM 542 [30].

Figura 6.6: Drivers utilizados y sus dimensiones.

Las conexiones electrónicas se realizan de la misma forma, con la tarjeta de soldadura y el Arduino al centro, sin embargo ahora los driver solo necesitan una conexión a tierra de 5 [V] y la conexión de 12 [V]. En la Figura 6.7 se observa la conexión electrónica con los nuevos driver.

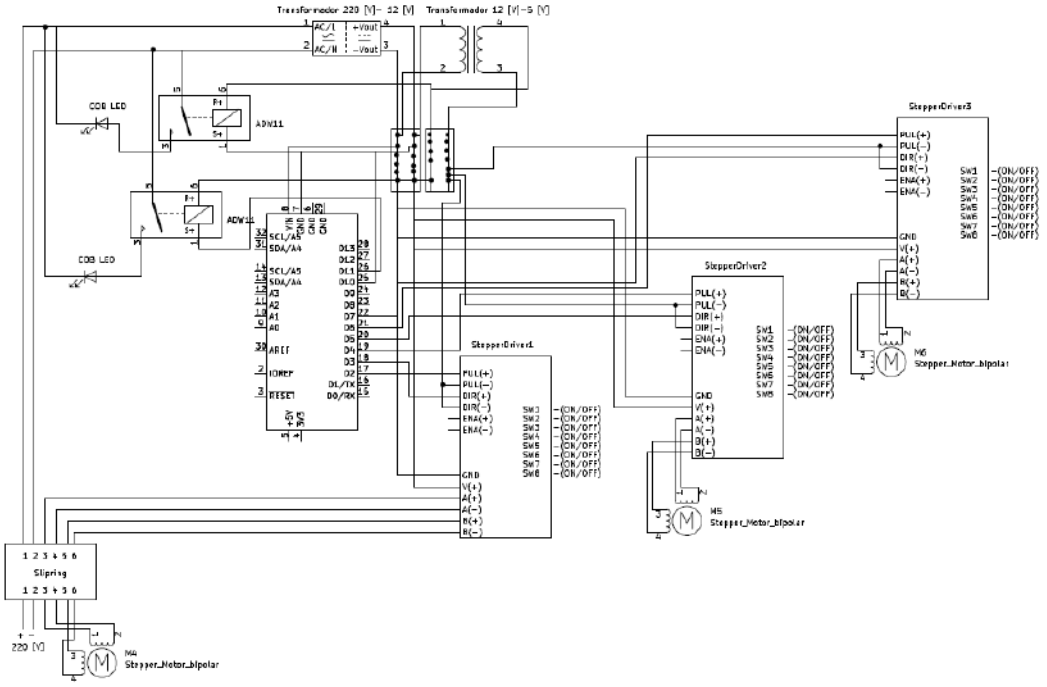


Figura 6.7: Nuevo circuito del banco de ensayos.

Debido al tamaño del driver DM542, la ubicación de los driver cambia con respecto a lo definido en el esquema de Figura 6.5. En esta nueva versión, se ubican 2 driver en una sección del engranaje y el último driver en la sección contraria. El resto de componentes no sufre cambios en su ubicación, en la Figura 6.8 se muestra el nuevo esquema digital del montaje electrónico del banco de ensayos y en el Anexo D se encuentran imágenes del banco de ensayos.

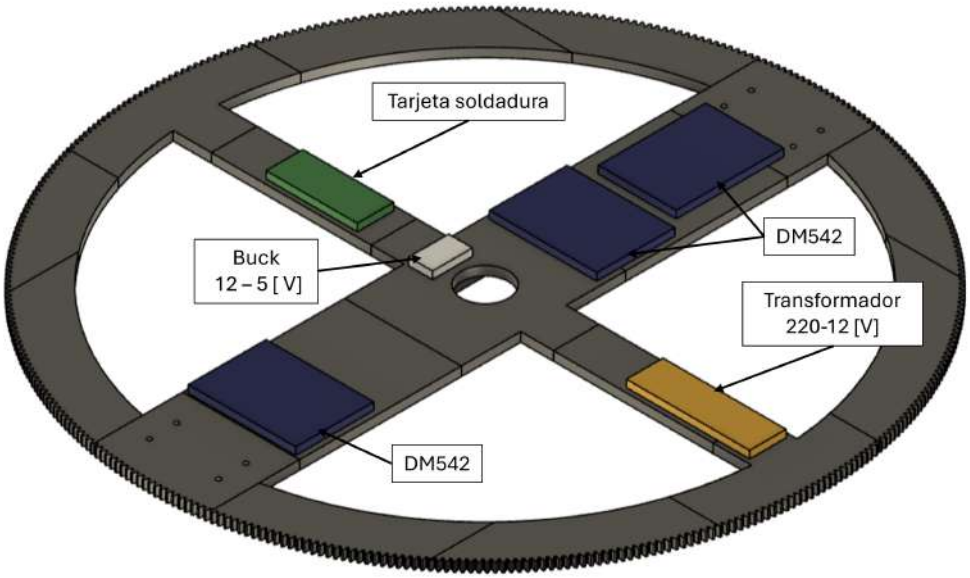



Figura 6.8: Montaje electrónico del banco de ensayos.

# CAPÍTULO 7: Pruebas de funcionamiento.

Una vez montado el circuito electrónico en la estructura, se debe verificar el funcionamiento del banco de ensayos como un sistema completo. Para ello se ejecutan dos pruebas básicas diferentes con tal de revisar los movimientos de los motores, el funcionamiento de las luces y la respuesta de un eventual satélite en el banco.

## 7.1 Dispositivo de adquisición de datos.

Con el fin de evaluar el banco de ensayos desde la perspectiva del satélite a ensayar, se diseña un dispositivo de adquisición de datos que simule la presencia de un satélite. El dispositivo consiste en un cubo de 10x10x10 [cm] simulando un Cubesat de 1U, anclado a una plataforma de 10 [cm] de altura que eleve el satélite al centro del banco de ensayos. La fabricación del dispositivo se lleva a cabo en el Laboratorio de Técnicas Aeroespaciales utilizando la máquina láser para maquinar las diferentes planchas de madera que lo formarán.

El dispositivo cuenta con 3 sensores foto transistores BPW34 como el que se muestra en la Figura 7.1 para medir la luz incidente sobre las caras del satélite, estos se energizan con un Arduino UNO que a su vez almacena la medición de los sensores en una tarjeta Micro SD, además los sensores se conectan a una resistencia de 10 [kΩ] y a los pines digitales del Arduino dando como resultados valores de medición entre 0 y 1024 [-], el circuito para el dispositivo se incluye en el Anexo E. Los sensores se ubican en 3 caras del satélite que representen 3 planos diferentes, en cuestión, los sensores están ubicados en las caras “Top”, “Right” y “Front” como se indica en la Figura 7.2 donde también se muestra el satélite y su anclaje.

Figura 7.1: Sensor BPW34.

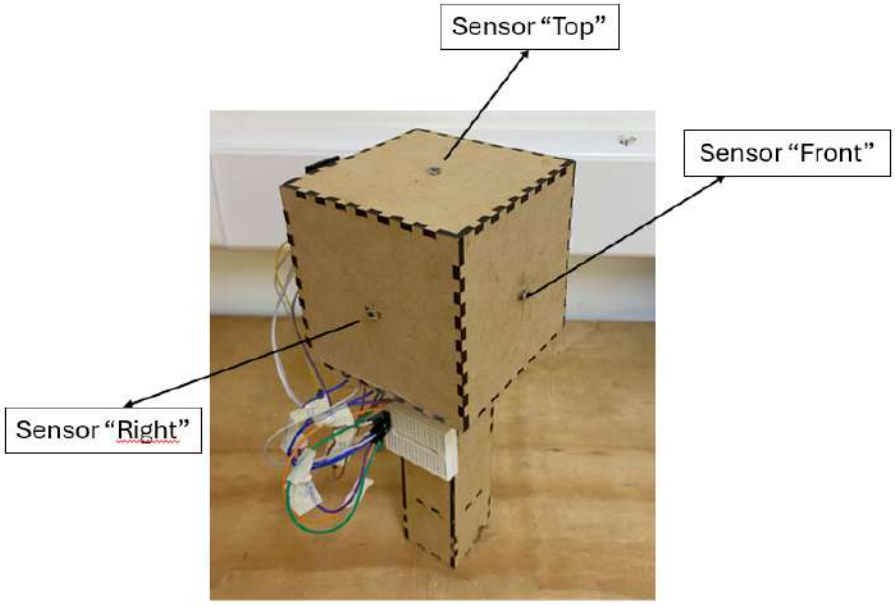



Figura 7.2: Dispositivo y ubicación de sensores [31].

Además, para ambas pruebas que se ejecutarán a continuación, se requiere de una posición inicial, para ello se elige la posición que se muestra en la Figura 7.3 con ambos soportes de luces en contacto con el tubo central de aluminio.

Figura 7.3: Referencia de posición inicial para pruebas.

## 7.2 Prueba 1

La Prueba 1 consiste en mover la fuente de luz dando una vuelta completa en la elevación sin mover la luz en la dirección del azimuth, para ello se moverán ambos engranajes verticales pero solo uno con la luz encendida, al llegar a 180 [°] en la elevación, ambos engranajes cambiarán

el sentido de movimiento al mismo tiempo en que la luz que estaba apagada se encenderá para finalizar el recorrido. Con este prueba se desea probar la exactitud y continuidad en el movimiento de los motores así como las consecuencias de los cambios de dirección en la estructura. Además, se desea evaluar la sensibilidad del sensor BPW34 a la fuente de luz. Para ejecutar este prueba se debe seguir el itinerario que se indica a continuación:

1. Compilar en el Arduino del banco de ensayos el código “Prueba1.ino”.
2. Compilar en el Arduino del satélite el código “LecturaSensores.ino”.
3. Introducir tarjeta MicroSD en Arduino UNO en el satélite.
4. Posicionar ambos LED en contacto con el tubo de aluminio para definir posición inicial del movimiento.
5. Posicionar el satélite con el sensor “Front” frente al engranaje 1.
6. Conectar el banco de ensayos a línea directa con el cable rotulado con “220 [V]”.

### 7.2.1 Resultados

Como indica el código “LecturaSensores.ino” en los anexos, la lectura de los sensores y el guardado de los datos se realiza con un delay de 250 [microseconds]. Debido a que la luz de espectro completo es una combinación de diferentes longitudes de onda, la medición del satélite se ve afectada por fluctuaciones gracias a la sensibilidad diferente a las distintas longitudes de onda como se observa en la Figura 7.4. Por lo anterior, las mediciones se someten a un filtro para eliminar aquellos datos y posteriormente se aplica media móvil en intervalos de 3 datos. En la Figura 7.5 se muestra el resultado con filtros aplicados de la Prueba 1, esta fue ejecutada dos veces, es decir, la luz LED recorre los 360 [°] en la elevación dos veces y además con filtros aplicados.

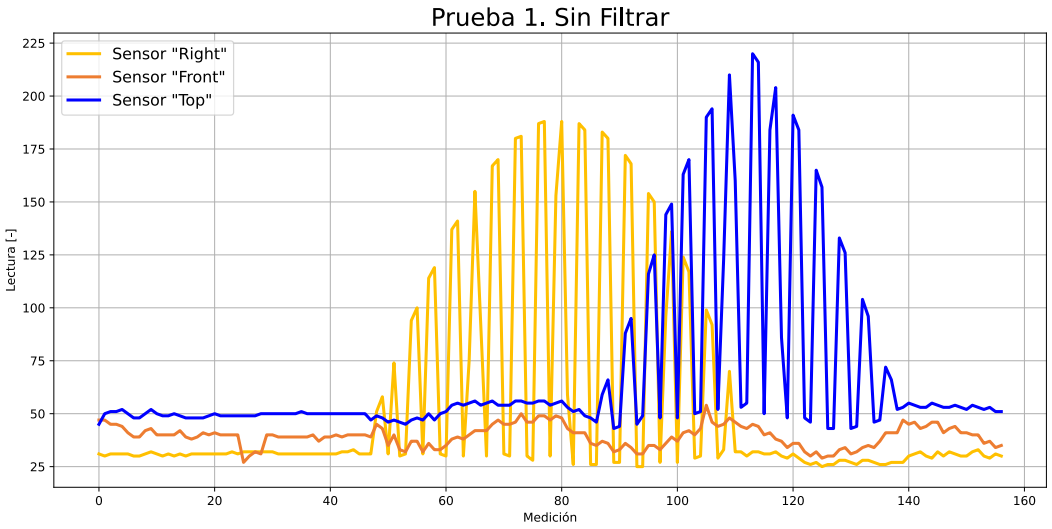



Figura 7.4: Resultados Prueba 1, sin filtrar.

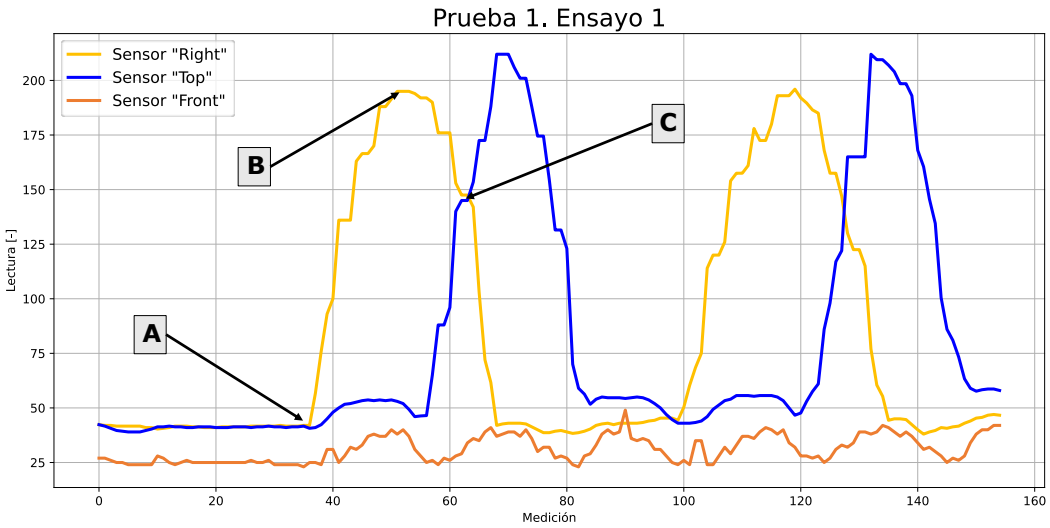



Figura 7.5: Resultados Prueba 1, Lectura 1.

En los resultados se observan diferentes hitos marcados con letras de la “A” a la “C”, a continuación se describen estos hitos acompañados de imágenes de los instantes en el banco de ensayos para comprender los resultados.

- Hito A: Aparición de la fuente de luz en el campo de visión del sensor. El sensor BPW34 tiene un campo de visión de 120 [°] [31], es decir, en el hito A la fuente de luz se encuentra a aproximadamente 60 [°] del sensor generando el inicio del aumento en la medición. En la Figura 7.6 se observa el hito A en el banco de ensayos.
- Hito B: Máxima incidencia sobre el sensor. En el hito B, la fuente de luz está a 0 [°] del sensor generando máxima incidencia sobre él y provocando el peak de 190 [-] en la medición del sensor “Right”. En la Figura 7.7 se observa el hito B en el banco de ensayos.
- Hito C: LED en el campo de visión de ambos sensores. Como se mencionó anteriormente, el sensor tiene un campo de visión de 120 [°], debido a la posición de los sensores en las diferentes caras los conos de visión de los sensores se cruzan generando áreas donde la luz incide sobre dos sensores al mismo tiempo y por tanto existen puntos donde la medición es igual en ambos sensores, este punto se observa en el gráfico como el hito C. En la Figura 7.8 se observa el momento en que ocurre el hito C.

Figura 7.6: Hito A.

Figura 7.7: Hito B.

Figura 7.8: Hito C.

### 7.2.2 Conclusiones de la prueba

Al finalizar la Prueba 1 se ha demostrado la funcionalidad de los engranajes verticales y la ejecución de las ordenes de las luces LED. Estructuralmente el banco responde bien a los cambios de sentido en los engranajes verticales, además el movimiento de estos no provoca movimientos indeseados en el engranaje horizontal. El movimiento de los engranjes es constante y la velocidad elegida en el código es correcta para no generar vibraciones que afecten al banco de ensayos o al satélite en cuestión.

Respecto a las luces LED y la interacción con los sensores, los sensores responden bien a los estímulos generados por las luces, de todas formas, los resultados debieron ser filtrados y manejados con técnicas de post-process. El sensor BPW34 no fue saturado por la luz COB de espectro completo, esto permite observar los peaks de medición en el punto A y similares dentro del gráfico.

## 7.3 Prueba 2

La Prueba 2 comienza en la misma posición inicial que la Prueba 1 y consta de dos etapas, en primer lugar se mueve el engranaje 1 hasta 90 [°] en la elevación para llegar a la máxima incidencia del sensor "Front", posteriormente y sin mover aquel engranaje, se mueve el engranaje horizontal en 360 [°] llegando al mismo punto en el sensor "Front" para descender los 90 [°] en la elevación y volver a la posición inicial del engranaje vertical. Con esta prueba se desea evaluar la estructura bajo los efectos del movimiento del engranaje horizontal, verificar que los cables no interrumpan la actividad del banco de ensayos y confirmar el desempeño del sensor "Front". A continuación se detalla el procedimiento previo para ejecutar la Prueba 2.

1. Compilar en el Arduino del banco de ensayos el código "Prueba2.ino".
2. Compilar en el Arduino del satélite el código "LecturaSensores.ino".

3. Introducir la tarjeta MicroSD al Arduino UNO en el satélite.
4. Posicionar ambas luces al contacto con el tubo de aluminio.
5. Conectar el cable rotulado con “220 [V]” a línea directa para comenzar la prueba.

### 7.3.1 Resultados

En la Figura 7.9 se grafican los resultados de los sensores para la Prueba 2 ejecutada una vez. De la misma manera, se identifican hitos en el gráfico con letras de la “D” a la “G” y a continuación se describen cada uno de los hitos acompañado de las fotografías en las Figuras 7.10-7.14, además se incluye la Figura 7.13 que representa una fotografía del banco de ensayos en un momento intermedio de la prueba para mostrar el giro en 360 [°] del banco.

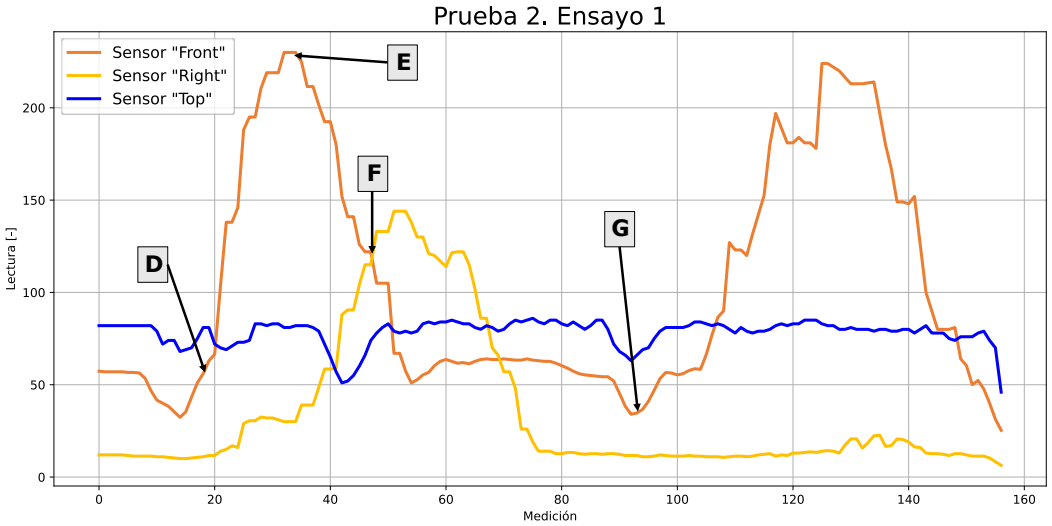



Figura 7.9: Resultados Prueba 2, Lectura 1.

- Hito D: Aparición de la luz en el campo de visión del sensor. De igual forma que en la prueba anterior, el sensor percibe la luz a partir de los 60 [°] aproximadamente.
- Hito E: Máxima incidencia de la luz. La fuente de luz se posiciona aproximadamente a 0 [°] del sensor generando las máximas lecturas del sensor “Front”.
- Hito F: Fuente de luz percibida por ambos sensores. Como se mencionó anteriormente, el hito F representa un punto dentro del área generada por la intersección de los conos donde la medición de dos sensores es igual en el mismo instante de tiempo.
- Hito G: Aparición de la luz en el campo de visión. De igual forma que en el hito A, la fuente de luz aparece nuevamente en el campo de visión del sensor “Front” para posteriormente posicionarse a 0 [°] con máxima intensidad sobre el sensor y descender a la posición inicial finalizando la prueba 2.

Figura 7.10: Hito D

Figura 7.11: Hito E

Figura 7.12: Hito F

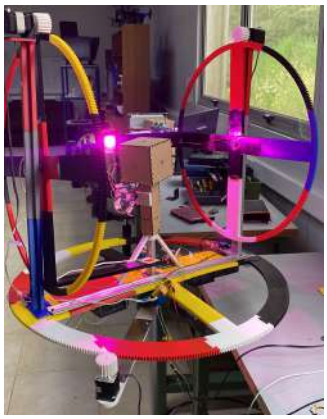


Figura 7.13: Hito Intermedio

Figura 7.14: Hito G

### 7.3.2 Conclusiones de la prueba

Luego de la prueba 2 se ha demostrado el correcto funcionamiento del engranaje horizontal y el funcionamiento del sensor "Front" que hasta ahora no había sido activado por las luces LED.

El engranaje horizontal completó de manera correcta el movimiento asignado, realizando una revolución completa sin interrupción. El movimiento fue de manera constante y suave sin mayores vibraciones en el banco de ensayos. Con los resultados de los puntos E y el homólogo punto de máxima incidencia en el intervalo [120,140] del gráfico en la Figura 7.9 se comprueba que el engranaje horizontal lleva a la estructura al mismo punto inicial luego de completar la revolución, además el movimiento horizontal del banco no afectó al engranaje vertical, manteniendo este en todo momento su posición de 90 [°].

En este caso, el sensor "Right" posee lecturas bastante menores a los otros dos sensores, esto se debe a la posición en sombra de este sensor, sumado a las condiciones lumínicas del laboratorio influenciadas por la iluminación eléctrica y natural. Los tres códigos utilizados en este Capítulo se encuentran el Anexo F.

# CAPÍTULO 8: Simulador

Una vez realizadas las pruebas de funcionamiento, se escribe el código capaz de generar las simulaciones de órbita en el banco de ensayos. En este capítulo se revisa la metodología y la línea de trabajo del código para simular el vector Sol de la órbita. La órbita elegida para validar el funcionamiento del banco de ensayos es la órbita del SUCHAI-3 dado que se cuentan con los resultados extraídos de [14].

Cabe mencionar que para este tipo de simulaciones, la posición inicial de los engranajes verticales es de la forma que se indica en la Figura 8.1, esto se debe a la definición de los ejes coordenados en el bodyframe.

Figura 8.1: Referencia de posición inicial para simulación en órbita.

## 8.1 Diagrama del código

Con el fin de comprender el orden de trabajo del código y como las funciones se complementan entre sí, en la Figura 8.2 se muestra un diagrama de flujo del código Arduino.

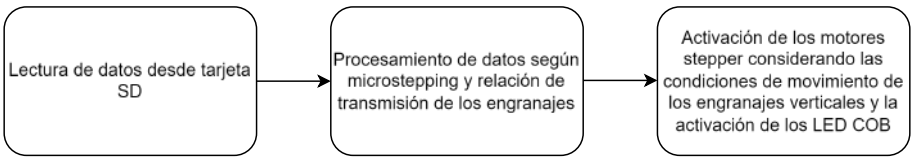



Figura 8.2: Diagrama de bloques para cálculo del vector Sol.

1. Lectura de datos desde tarjeta SD: El trabajo del código comienza con la lectura de ángulos desde la tarjeta SD. Esta función abre y lee un archivo .csv creado desde el código Python descrito en 4.1 que contiene dos columnas con las orientaciones azimuth y elevación replicarse en el banco de ensayos, a partir de los cuales se crean dos vectores columna.

2. Procesamiento de datos: Aquí, los ángulos son transformados a pasos de los motores según la preferencia de microstepping que el driver ofrece, en este caso se utilizaron 6400 $[\frac{pasos}{rev}]$. Finalmente, los pasos son multiplicados por las relaciones de transmisión entre los engranajes correspondientes según las siguientes ecuaciones:

$$\frac{Z_{Eng-Horizontal}}{Z_{NEMA}} = \frac{370}{20}$$

$$\frac{Z_{Eng-Vertical}}{Z_{NEMA}} = \frac{290}{20}$$

3. Activación de los motores y luces LED: Esta función toma los pasos calculados anteriormente y a partir de la creación de pulsos, activa los motores en los pasos y dirección correcta. Al mismo tiempo, esta función envía la señal a los relays para activar las luces COB y simular la trayectoria correcta.

## 8.2 Flujo de trabajo

### 8.2.1 Python

Como se mencionó, la primera función del código en la Figura 8.2 lee un archivo .csv, este archivo es producto del código Python por lo tanto, la ejecución del código Arduino requiere de la ejecución previa del código Python. A continuación se explican brevemente las condiciones necesarias para ejecutar el código Python, los resultados de este código y posteriormente el funcionamiento del código Arduino. El código Python se puede encontrar en el siguiente link

El código Python requiere de diferentes valores de entrada correspondientes a información del satélite, tales como:

1. Un TLE en formato .txt: El código es capaz de propagar hasta 24 [hrs] el estado de un satélite a partir de un TLE. Sobrepasar las 24 [hrs] de propagación aumenta considerablemente el porcentaje de error de los resultados. En las líneas del código, se debe ingresar el nombre completo del archivo .txt, además de asegurarse que el archivo se encuentra en la misma carpeta que el archivo .py

2. Fecha: Se debe ingresar en las lineas del código la fecha correspondiente al TLE en formato dd/mm/yyyy.
3. Inercias y condiciones iniciales del movimiento rotacional: Se deben ingresar manualmente las inercias principales del satélite y las condiciones iniciales de ángulo y velocidades angulares.

El código requiere la instalación de las siguientes librerías:

1. SGP4
2. SciPy
3. Numpy
4. Pandas
5. Math
6. ScipySignal

La ejecución del código da como resultado las Figuras en 4.2 y 4.3. Para efectos del banco de ensayos se decidió por simular utilizando máximos y mínimos locales de estos gráficos, identificando estos puntos en el gráfico se obtienen las Figuras 8.3 y 8.4.

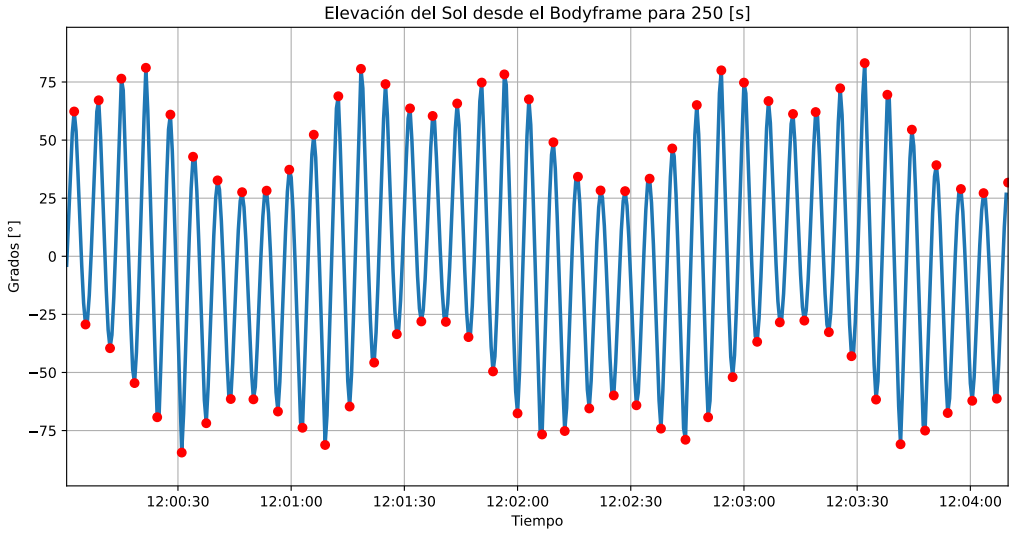



Figura 8.3: Elevación del Sol desde el bodyframe para 250 [s] [14].

Finalmente, los valores representados por los puntos en rojo se agrupan en dos columnas (una columna para elevación y una columna para azimuth) comenzando con el primer valor máximo y luego el mínimo y así sucesivamente como muestra en la Tabla 8.1

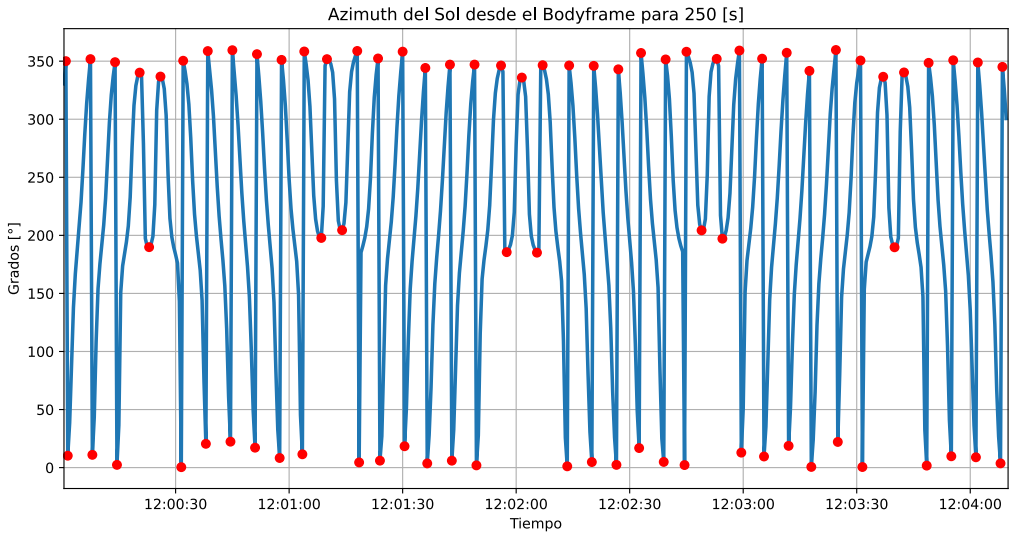



Figura 8.4: Azimuth del Sol desde el bodyframe para 250 [s] [14].

Tabla 8.1: Valores en archivo .csv para Arduino.

| Elevación [°] | Azimuth [°] |
|---|---|
| 62 | 360 |
| -29 | 0 |
| 67 | 360 |
| -40 | 0 |
| ⋮ | ⋮ |

En este caso, los valores de Azimuth se han guardado como 360 [°] a excepeción de datos como los del tiempo 12:02:00. Gracias a la definición del paso de tiempo para la propagación, no se obtienen valores exactos de 360 [°], sin embargo es notorio que el azimuth se mueve en ciclos completos a excepción de los datos que se observan en 190 [°] aproximadamente que representan cambios en el sentido de giro por breves instantes.

Este archivo se debe guardar en la tarjeta SD con el nombre “destinos.csv”.

### 8.2.2 Arduino

A continuación, comienza el trabajo del Arduino, al que se ha acoplado el Shield de tarjeta SD para poder leer los archivos. El código Arduino lee el archivo .csv y lee las columnas para transformarlas a pasos de los motores usando las relaciones de engranaje y la resolución de 6400 $[\frac{pasos}{rev}]$.

Posteriormente, se ejecutan los movimientos con cada uno de los datos en las columnas. Los resultados para el movimiento del engranaje horizontal se ejecutan inmediatamente y los resultados para el engranaje vertical son procesados para que en caso de que un movimiento supere los -90 o 90 [°], se realice con movimientos combinando ambos engranajes verticales. El movimiento de los motores y la activación de las luces LED se realiza sin el uso de librerías, es decir, los pulsos

se envían utilizando ciclos “for” que envíen de forma alternada señales “HIGH” y “LOW”. A continuación se muestra la función responsable de la activación de motores y luces en el código, el código en su totalidad se puede revisar en el Anexo F.

```
void stepper(int motor, int pasos, int direccion) {
int pinDireccion = pinDireccion1;
int pinStep = pinStep1;
int mot3= pinStep3;
digitalWrite(Rele1, LOW);
digitalWrite(Rele2, HIGH);
if (motor == 2) {
  pinDireccion = pinDireccion2;
  pinStep = pinStep2;
  digitalWrite(Rele2, LOW);
  digitalWrite(Rele1, HIGH);
}

digitalWrite(pinDireccion, direccion);
aux=abs(pasos);
for (unsigned int x = 0; x < aux; x++) {
  digitalWrite(pinStep, HIGH);
  digitalWrite(mot3, HIGH);
  delayMicroseconds(400);
  digitalWrite(pinStep, LOW);
  digitalWrite(mot3, LOW);
}
```

### 8.3 Dificultades y nuevas opciones

Con el código anterior, no se pudo completar la simulación de la trayectoria del Sol para la órbita del SUCHAI-3. Si bien los bloques del código de la Figura 8.2 representando las funciones del código se ejecutan de manera correcta, la mala interpretación del funcionamiento de ciclos “for” generó que los movimientos de los motores no sean coordinados y no se simule de manera correcta.

Dado los resultados, no es posible utilizar solo un ciclo for ya que se necesita que los motores tengan diferentes velocidades, tampoco es posible tener dos ciclos for, pues eso genera que se ejecute solo un ciclo for a la vez. Finalmente, el control simultáneo de dos motores paso a paso a la vez es una tarea que abre dos alternativas para los trabajos futuros:

1. Multitasking: Mover ambos motores a diferentes velocidades con dos ciclos for al mismo tiempo es un problema de “Multitasking”, en estricto rigor, un Arduino UNO es incapaz de ejecutar este tipo de tareas de manera directa. Una técnica para lograr este tipo de desafíos es el Multitasking.

   El “Multitasking” en Arduino se basa en la función millis, esta función es una función in-

tegrada de Arduino que entrega como resultado el tiempo transcurrido desde el inicio de la ejecución del programa del microcontrolador [32]. La técnica consiste en utilizar el tiempo como una condición para la ejecución de tareas, entonces, los motores se moverán si la condición de tiempo se cumple, por ejemplo si el tiempo es menor a 30 [s] entonces mover motor 1 y motor 2.

Aplicado al simulador, se deben obtener las velocidades en $[\frac{grados}{s}]$ que deben tener los motores para mantener la coordinación y simular de forma correcta los resultados, una vez obtenida las velocidades deseadas, se pueden entender los destinos de los motores como tiempos de acción en vez de ángulos y así controlar de manera simultanea los motores. Dentro de las funciones también se debe agregar la condición para cumplir con los cambios de sentido de giro.

2. Librería AccelStepper.ino: La librería AccelStepper.ino es otra opción para sortear las dificultades encontradas. La librería permite el control de velocidad, aceleración y destino de los motores sin necesidad de usar ciclos "for" ya que posee comandos integrados que logran estos objetivos [33].

   Uno de los comandos integrados es el comando Stepper.run() el que permite mover el motor a destinos y velocidades definidas previamente, la ventaja es que el comando .run() puede funcionar de manera simultanea dentro del Arduino aún cuando esté escrito en lineas sucesivas. Otra de las grandes ventajas de la librería es que las funciones guardan referencia de sus destinos, es decir, si dentro de las configuraciones de .run() estaba la condición .GoTo(Ángulo), una vez que el motor haya llegado a "Ángulo", el motor se detiene y sabe que ha llegado a destino creando una variable llamada .DistanceToGo(), por lo tanto se puede recurrir a la variable DistanceToGo() para que cuando sea igual a cero, recurra nuevamente a la columna de datos creada para continuar con la trayectoria a simular.

### 8.4 Implementación de librería AccelStepper.ino y trabajos futuros.

Con la información anterior, se decide por implementar la opción de la librería a través de las siguientes funciones.

1. Definición de velocidad con comandos "stepper.SetMaxSpeed" y "stepper.SetSpeed". Estos comandos sirven para definir la velocidad del motor en $\frac{pasos}{segundo}$. Con esto, se puede configurar la velocidad de manera directa antes que realizar un delayMicroseconds dentro de un for.

2. El control de los motores y la definción de destinos son dos funciones diferentes. Para definir el destino del motor se utiliza la función "stepper.MoveTo()" que recibe como valor de entrada los pasos a realizar y la detección de preferencia de microstepping es automática. Por otro lado, para ejecutar la tarea se utiliza el comando "stepper.runSpeedToPosition()" que moverá los motores a la velocidad constante definida anteriormiente hasta llegar a destino definido en MoveTo.

3. Control y ejecución continua con comando "stepper.DistanceToGo()". El comando DistanceToGo() es un valor que representa la cantidad de pasos faltantes por realizar. Para mantener

el movimiento de los motores se utiliza el comando “while” con la condición que DistanceTo-Go() sea distinto de cero, es decir, los motores se moverán mientras hayan pasos por realizar. Esto mantendrá el movimiento y la coordinación de los motores.

La librería incluye referencia de la posición de los motores, es decir el motor en su inicio de operación define como posición cero la posición inicial y una vez realizado el movimiento definido en MoveTo, el motor guarda una variable con su posición. Lo anterior es de gran ayuda para la simulación de la orbita ya que no será necesario guardar la posición actual ni cálculos intermedios.

El manejo de datos se debe realizar de manera diferente para aprovechar las funciones de la librería, además, ahora no solo se utilizarán los peaks independientes de los ángulos sino que se guardarán los datos de ámbos ángulos cada vez que uno de ellos alcance un peak. De la misma forma que antes, valores extremos fueron aproximados a 360 [°] y 0 [°] según corresponda. Los pares de puntos a simular se observan en la Figura 8.5 y además un extracto de los primeros valores en la Tabla 8.2.

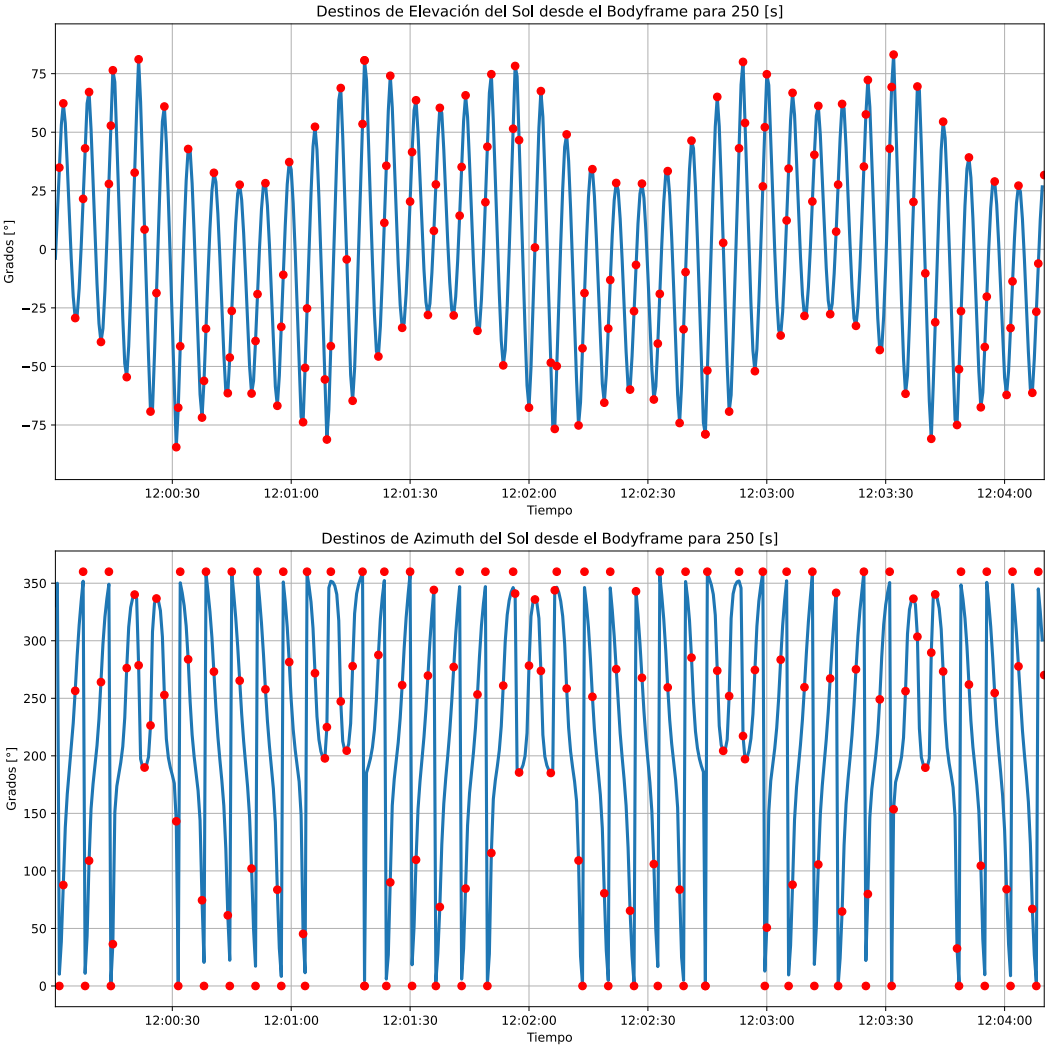



Figura 8.5: Destinos marcados para ambos ángulos.

Tabla 8.2: Valores en archivo .csv para movimiento con librería.

| Elevación [°] | Azimuth [°] |
|---|---|
| 34 | 10 |
| 62 | 87 |
| -29 | 256 |
| 21 | 360 |
| 43 | 10 |
| ⋮ | ⋮ |

El código desarrollado en base a la libería AccelStepper se encuentra en el Anexo F bajo el nombre "Orbita Librería". En él, se utiliza el mismo mecanismo para lectura de datos, sin embargo, esta vez son tres datos a leer y no dos como en el código anterior. Este tercer dato, llamado "aux" en el código dentro de la función mover, se utiliza para configurar la posición del motor horizontal. Como se mencionó, los resultados del azimuth se traducen en revoluciones y dada la naturaleza con la que el motor funciona guardando su posición, lo más conveniente es configurar su posición cada vez que termine alguna de las revoluciones.

El valor de aux varía entre tres valores: 0, 1 y 2. Si aux es igual a 0, entonces no es necesario configurar la posición actual; Si aux es igual a 1, entonces el motor configura su posición actual como 0 [°] y se dirige a 360 [°] en dirección horaria; Si aux es igual a 2, entonces el motor configura su posición actual en 360 [°] y se dirige a 0 [°] en dirección antihoraria. En la Figura 8.6 se indican los instantes en que aux funciona para ejecutar la línea "stepper3.setCurrentPosition()" según lo indicado.

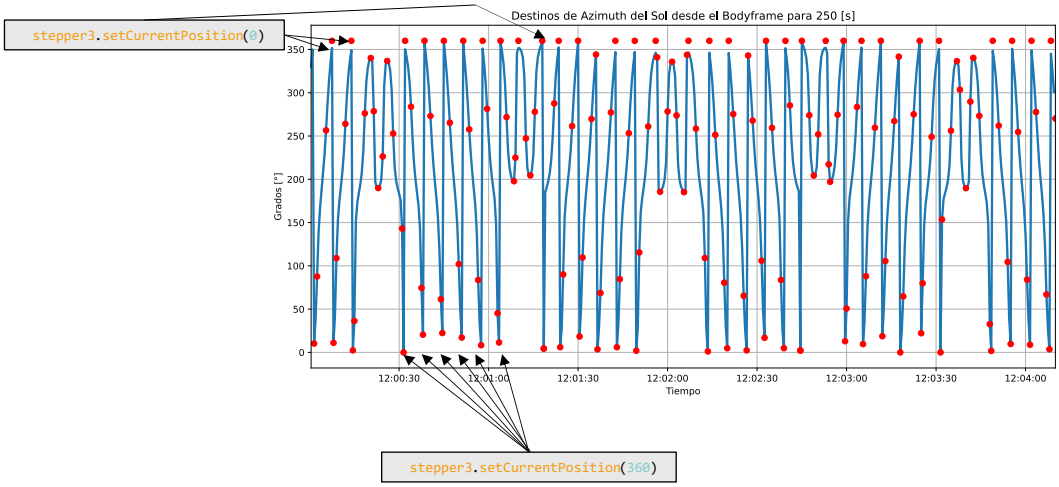



Figura 8.6: Indicaciones de configuración de posición actual.

Para el caso de la órbita del SUCHAI-3, la elevación del Sol no excede los 90 [°] por lo que no es necesario el trabajo de ángulos entre los dos engranajes verticales.

Lamentablemente, no se pudo probar el código implementado con libería en el simulador. Sin embargo, el código se ha verificado dentro de la plataforma Arduino IDE confirmando la sintaxis

del código, además fue probado en herramientas de simulación en internet pudiendo ejecutar los comandos en los motores utilizando los códigos de las librerías. Finalmente, se ha dejado para trabajos futuros los siguientes puntos:

1. Prueba del código en el banco de ensayos: Ejecutar el código en el banco de ensayos con su microcontrolador Arduino y sus conexiones.
2. Verificación de velocidad de los motores: Al utilizar el condicional "while" para la ejecución de movimientos, basta con que un solo motor tenga pasos por ejecutar para que el otro motor esté detenido esperando una nueva orden. La correcta configuración de velocidad solo se logra realizando pruebas en el banco de ensayos, de esta forma se asegura la coordinación de los motores para llegar a destinos según los pares en la Tabla 8.2.
3. Pruebas con diferentes órbitas y automatización del código: Gracias a que el código Python es un código universal para órbitas LEO, es necesario ejecutar pruebas con diferentes órbitas, esto generará diferentes escenarios con límites de ángulos y velocidades distintas a las mencionadas en este trabajo y uno de los desafíos futuros es la automatización de las condiciones de simulación.

# CAPÍTULO 9: Conclusiones

En base a los resultados obtenidos en [14] se trabajaron tres propuestas de banco de ensayos, iterando sobre su estructura, fuente de luz y control del funcionamiento. A través de reuniones con stakeholders y presentación de alcances y dificultades de las propuestas, se define cual será el diseño final a implementar.

Una vez aprobada la propuesta y con el diseño definitivo del banco de ensayos se elabora el presupuesto de la estructura para ser presentado a las autoridades DIM. El presupuesto considera la adquisición de componentes electrónicos adicionales a lo mínimo necesario en caso que la falla o mal funcionamiento de alguna de ellas no genere retrasos en el trabajo. La cantidad de PLA se dimensiona utilizando las configuraciones en 5.1.

La construcción del banco de ensayos se lleva a cabo en la sala de electrónica del Laboratorio de Técnicas Aeroespaciales utilizando las herramientas disponibles en el taller del primer piso. La construcción sufre de demoras debido a la alta demanda de impresión en el Laboratorio de Fabricación de la Universidad, por otro lado, la estructura sufre de flexión cuando la electrónica, en especial los motores paso a paso, son instalados sobre ella. Luego de evaluación del montaje y de la flexión en la estructura, se añade el rigidizador y se diseña el seguro para el rodamiento, de esta forma, la estructura está lista para sostener el montaje de la electrónica. Inmediatamente, se realizan los trabajos de soldadura para montar los componentes electrónicos en la estructura, una vez realizado el montaje y durante las pruebas básicas de funcionamiento se evidencia un mal funcionamiento de los motores. Con ayuda de equipos electrónicos se determina que el Arduino UNO y los driver 8825 son causantes de las fallas, ambos fueron reemplazados inmediatamente. Las conexiones electrónicas definitivas cuentan con los driver DM542 de mejores prestaciones y un Arduino UNO original.

Se realiza la verificación de funcionamiento del banco de ensayos ejecutando dos pruebas básicas y obteniendo resultados de la emisión de luz a través de un dispositivo de adquisición de datos. Las pruebas de funcionamiento validan la rigidez del banco de ensayos, el funcionamiento de los engranajes tanto en precisión como en velocidad y la continuidad e intensidad de la emisión de luz de los LED COB.

Se definió la estructura del código Arduino para simular la trayectoria del Sol en la órbita deseada. Se definen los requerimientos para la ejecución del código Python y posterior código Arduino, cuidando los archivos resultantes y el manejo de los datos para la correcta ejecución. Al ejecutar estas tareas en el banco de ensayos se encuentran dificultades para simular de manera correcta las órbitas y obtener los resultados deseados. Realizar el movimiento de los motores únicamente enviando pulsos “HIGH” y “LOW” dentro de ciclos es incorrecto y no permite el control simultáneo de dos

motores a diferentes velocidades y destinos. Por lo anterior, se proponen dos alternativas que den solución a la dificultad encontrada. Una alternativa consiste en el concepto de "Multitasking" en Arduino, esta se basa en el uso de la función "millis" que utiliza el tiempo como variable auxiliar para la ejecución de tareas simultáneas. Por otro lado, la utilización de la librería AccelStepper.ino, esta librería permite el control de motores paso a paso de manera simultánea y controlando parámetros como aceleración, velocidad y destino. En base a los argumentos desarrollados, se implementa un código en Arduino basado en el funcionamiento de la librería AccelStepper con el fin de utilizar todas sus funcionalidades y códigos integrados de referencias de posición y velocidad. Esta nueva implementación obliga a redefinir el manejo de datos y las variables utilizadas para simular las trayectorias. Con las dos orientaciones a simular y una variable auxiliar para la definición de posición actual se manejan las funciones de la librería para la ejecución de pruebas de órbita. Se propusieron los trabajos futuros con el código implementado para validar el funcionamiento del banco de ensayos, entre ellos se encuentra la definición de velocidad de los motores para la coordinación entre ambos ángulos a simular.

Finalmente, se ha construido y verificado el funcionamiento de un simulador solar para verificación de algoritmo de actitud construido en los laboratorios de la Universidad de Concepción a un costo menor que los bancos de ensayos mencionados en el Capítulo 1. La implementación requirió de constantes mejoras y evaluaciones del funcionamiento para finalizar con el banco de ensayos operativo que se mostró en las imágenes. Ha quedado para trabajos futuros la validación del banco de ensayos simulando la trayectoria solar para la órbita del SUCHAI-3 con la utilización del código AccelStepper.

# Bibliografía


[1] E. Kulu, "Nanosatellite launch forecasts-traack record and latest predictions," 2022.

[2] N. Web, "What are smallsats and cubesats?" [Online]. Available: https://www.nasa.gov/content/what-are-smallsats-and-cubesats[Acceso:(2023-08-20)]/

[3] NASA, "Nasa systems engineering handbook," 2007.

[4] Y. Zheng, *Ground Tests of Micro/Nano Satellites [Space Microsystems and Micro/nano Satellites, Capítulo 5, 147-166].* Cambridge, MA: Elsevier, 2018.

[5] J. Guo, L. Monas, and E. Gill, "Statistical analysis and modelling of small satellite reliability," *Acta Astronautica*, vol. 98, pp. 97–110, 5 2014.

[6] J. Bouwmeester, A. Menicucci, and E. K. Gill, "Improving cubesat reliability: Subsystem redundancy or improved testing?" *Reliability Engineering and System Safety*, vol. 220, 4 2022.

[7] T. Villela, C. A. Costa, A. M. Brandão, F. T. Bueno, and R. Leonardi, "Towards the thousandth cubesat: A statistical overview," 2019.

[8] F. L. Markley, F. H. Bauer, J. J. Deily, and M. D. Femiano, "Attitude control system conceptual design for geostationary operational environmental satellite spacecraft series," *Journal of Guidance, Control, and Dynamics*, vol. 18, pp. 247–255, 1995.

[9] T. Iwata, T. Yoshizawa, H. Hoshino, and K. Maeda, "Precision attitude and orbit control system for the advanced land observing satellite (alos)," 2003.

[10] FACh, "Fuerza Áerea de chile, grupo de operaciones espaciales." [Online]. Available: www.fach.mil.cl/gos/sistemassot.php[Acceso:(2023-08-26)]

[11] P. Ortega, G. López-Rodríguez, J. Ricart, M. Domínguez, L. M. Castañer, J. M. Quero, C. L. Tarrida, J. García, M. Reina, A. Gras, and M. Angulo, "A miniaturized two axis sun sensor for attitude control of nano-satellites," *IEEE Sensors Journal*, vol. 10, pp. 1623–1632, 2010.

[12] D. Modenini, A. Bahu, G. Curzi, and A. Togni, "A dynamic testbed for nanosatellites attitude verification," *Aerospace*, vol. 7, 3 2020.

[13] Adobe, "Waterfall methodology: A complete guide." [Online]. Available: https://business.adobe.com/blog/basics/waterfall#:~:text=The%20Waterfall%20methodology%20%20also%20known,before%20the%20next%20phase%20begins.[Acceso:(2023-08-20)]/

[14] A. Maricahuin, "Diseño de un simulador solar para la determinación de actitud de cubesats en leo basado en un sistema de iluminación controlado y orientable." *Bibiliotecas UdeC*, 2023.

[15] H. Curtis, *Orbital Mechanics for Engineering Students*. Cambridge: Elsevier, 2020.

[16] P. Rochus, "The orbit in time [astrodynamics lecture]," 2021.

[17] D. A. Vallado and P. Crawford, "Sgp4 orbit determination." [Online]. Available: http://CelesTrak.com

[18] Celestrak, "Celestrak." [Online]. Available: https://celestrak.org[Acceso:(2023-05-9)]

[19] Astrofein, "Astro-und feinwerktechnik adlreshof gmb." [Online]. Available: https://www.astrofein.com[Acceso:(2023-05-11)]

[20] S. Roemer, "Experiences with the bird acs test facility and the resulting design of a state of the art mini-and micro-satellite acs test facility maroc-tubsat view project semi conductor physics view project," 2007. [Online]. Available: https://www.researchgate.net/publication/224986339

[21] D. I. of Space Systems, "Dlr-institute of space systems laboratories and facilities."

[22] J. A. Olsen, "Attitude determination and control system testbed for hardware and software testing and verification for small satellites," 2021.

[23] komplettbedrift, "Flashlights." [Online]. Available: https://www.komplettbedrift.no/product/1118357[Acceso:(2023-10-19)]

[24] Sputnix, "Dynamic simulator for spacecraft adcs tests."

[25] HowToMechatronics, "Stepper motors and arduino - the ultimate guide." [Online]. Available: https://howtomechatronics.com/tutorials/arduino/stepper-motors-and-arduino-the-ultimate-guide/#google_vignette[Acceso:(2023-10-19)]

[26] WOKOST, "Tappered roller bearing 30205- catalogue."

[27] AOF, "Deep groove ball bearing 6002-2rs."

[28] Senring, "Through hole slip ring." [Online]. Available: https://www.senring.com/through-hole-slip-ring/medium-hole/h2586.html[Acceso:(2023-10-19)]

[29] StepIM, "Integrated closed-loop stepper motor datasheet. nema 17-23-34."

[30] LeadShine, "User's manual for dm542, fully digital stepper driver."

[31] V. electronics, "Silicon pin photodiode."

[32] Arduino, "Millis() function documentation." [Online]. Available: https://www.arduino.cc/reference/es/language/functions/time/millis/[Acceso:(2024-01-31)]

[33] M. McCauley, "Accelstepper library for arduino." [Online]. Available: https://www.airspayce.com/mikem/arduino/AccelStepper/index.html[Acceso:(2024-01-31)]

# Anexo A: Carta Gantt

CARTA GANTT - Memoria de Título, Alvaro Maricahuin Monsalves

| OBJETIVOS | ACTIVIDADES | HITO | INICIO DE LA ACTIVIDAD | DURACION DE LA ACTIVIDAD |
|---|---|---|---|---|
| Objetivo 1.- Validación del diseño preliminar con stakeholders para asegurar una comprensión de las capacidades y limitaciones técnicas del banco de ensayos. | 1.- Programar reuniones individuales con cada stakeholder para discutir expectativas y requisitos específicos. | | 1 | 1 |
| | 2.- Organizar una sesión de revisión conjunta con todos los stakeholders para presentar el diseño y recopilar comentarios. | | 2 | 2 |
| | 3.- Analizar los comentarios recibidos y realizar modificaciones necesarias en base a las sugerencias de los stakeholders. | | 2 | 2 |
| Objetivo 2.- Elaboración del presupuesto del banco de ensayos en función de las características y alcanzable por el DIM UdeC. | 1.- Identificar y listar todos los componentes necesarios para la construcción del banco de ensayos | | 3 | 1 |
| | 2.- Investigar los costos asociados con cada componente incluyendo costos de envío e importación. | | 3 | 2 |
| | 3.- Calcular y elaborar un presupuesto detallado y realista en función de las características necesarias y las limitancias presupuestarias de DIM UdeC. | | 4 | 1 |
| | 4.- Redactar informe de avance 1. | Hito 1 - Entrega de Avance 1 | 4 | 1 |
| Objetivo 3.- Construcción del banco de ensayos, incluyendo hardware y software, en el espacio físico y tiempo previamente definidos. | 1.- Preparar el espacio físico designado para la construcción del banco. | | 5 | 1 |
| | 2.- Preparar y revisar los componentes asegurando su buen funcionamiento. | | 5 | 1 |
| | 3.- Construir el banco de ensayos definido anteriormente y elaborar el software y códigos necesarios para realizar las respectivas simulaciones. | | 6 | 4 |
| Objetivo 4.- Ejecución de pruebas en el banco de ensayos de al menos una órbita LEO para evaluar rendimiento y funcionalidad. | 1.- Desarrollar un plan detallado de pruebas que incluya al menos un escenario de prueba, los parámetros a medir y los procedimientos de ejecución. | | 10 | 1 |
| | 2. - Redactar informe de avance 2. | Hito 2 - Entrega Avance 2 | 10 | 1 |
| | 3.- Ejecutar las pruebas, registrar y analizar los resultados. | | 10 | 4 |
| | 4.- Repetir las pruebas en caso de ser necesario y añadir diferentes escenarios para probar los resultados posibles. | | 14 | 3 |
| Objetivo 5.- Redacción del informe final del proyecto, incoporando documentación de la construccipon y los resultados de las pruebas ejectuadas que validan el funcionamiento del simulador. | 1.- Organizar resultados de las pruebas de validación. | | 14 | 3 |
| | 2.- Comparación de resultados en simulador con datos experimentales y los calculados en PIA. | | 15 | 2 |
| | 3.- Redacción de informe final y preparación de presentación de defensa. | Hito 3 - Entrega Informe Patrocinante | 16 | 2 |

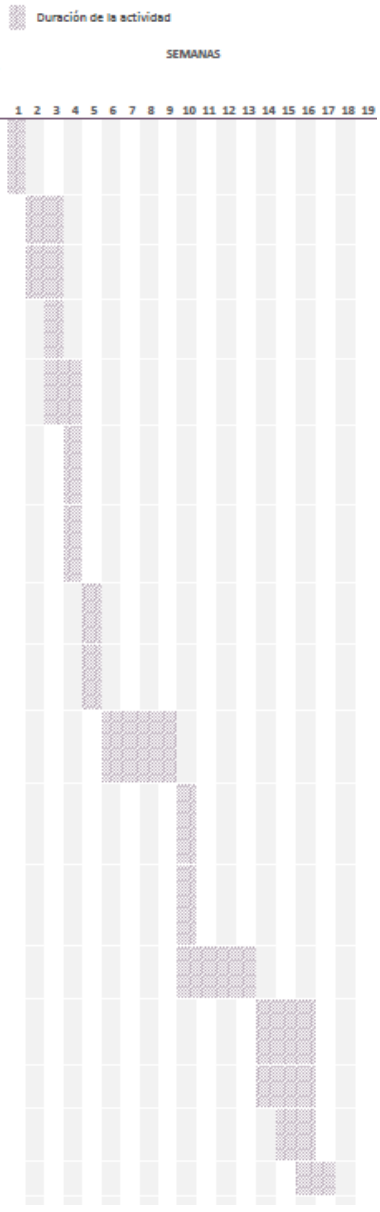



Figura A.1: Carga Gantt del proyecto.

# Anexo B: Planos

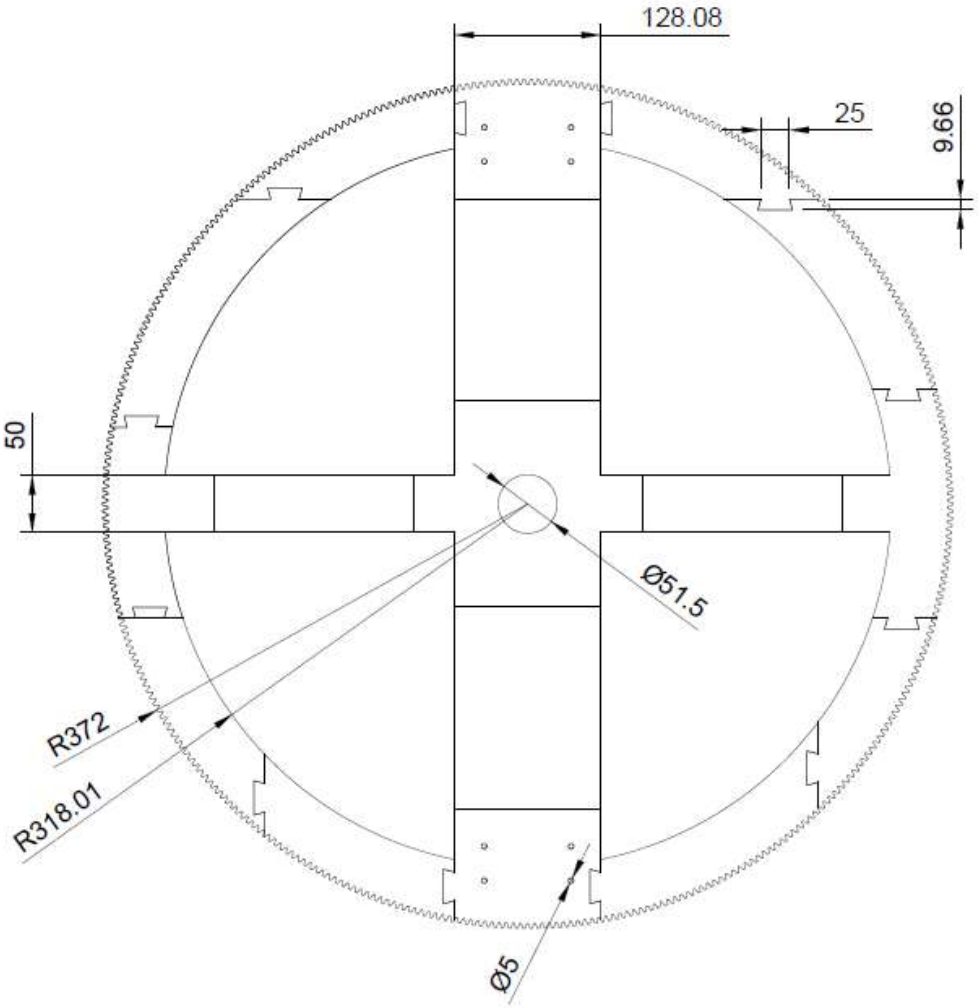



Figura B.1: Plano engranaje horizontal.

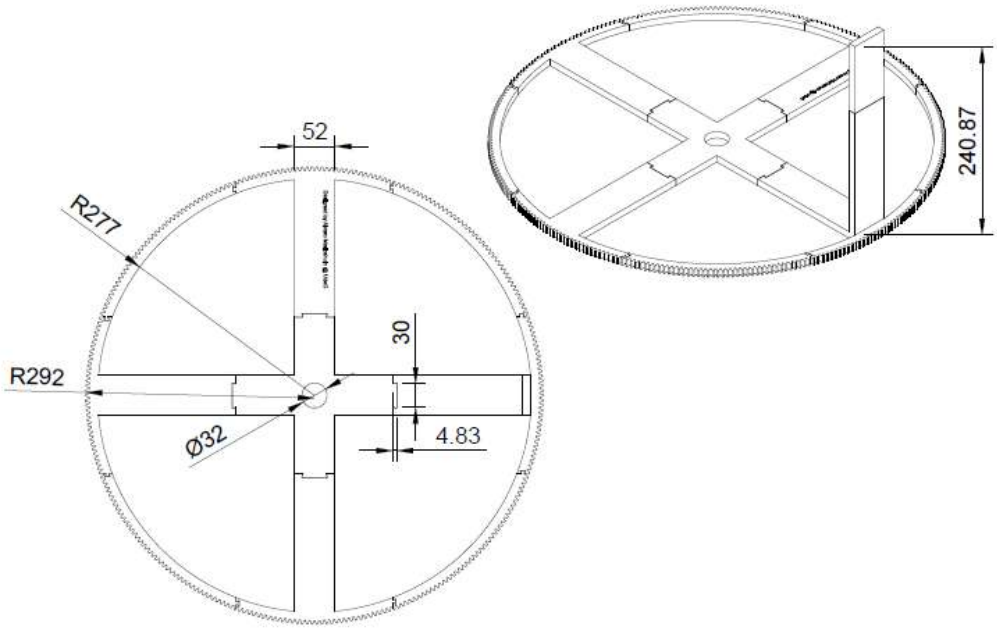



Figura B.2: Plano engranaje vertical.

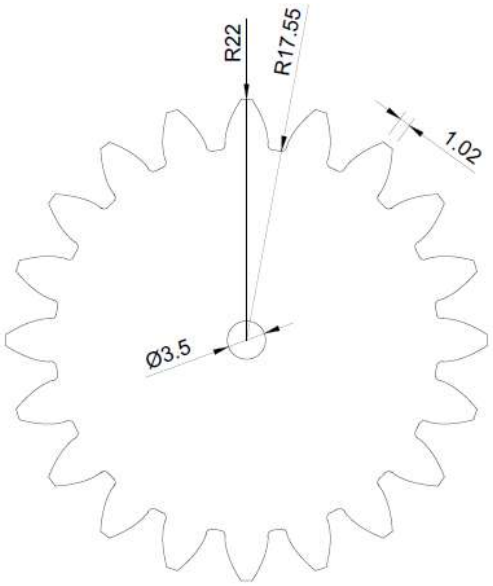



Figura B.3: Plano engranaje Nema.

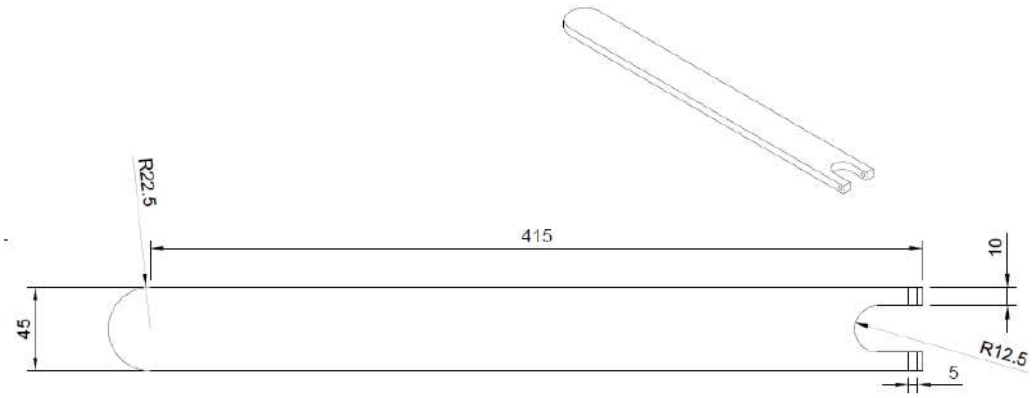



Figura B.4: Plano soporte horizontal.

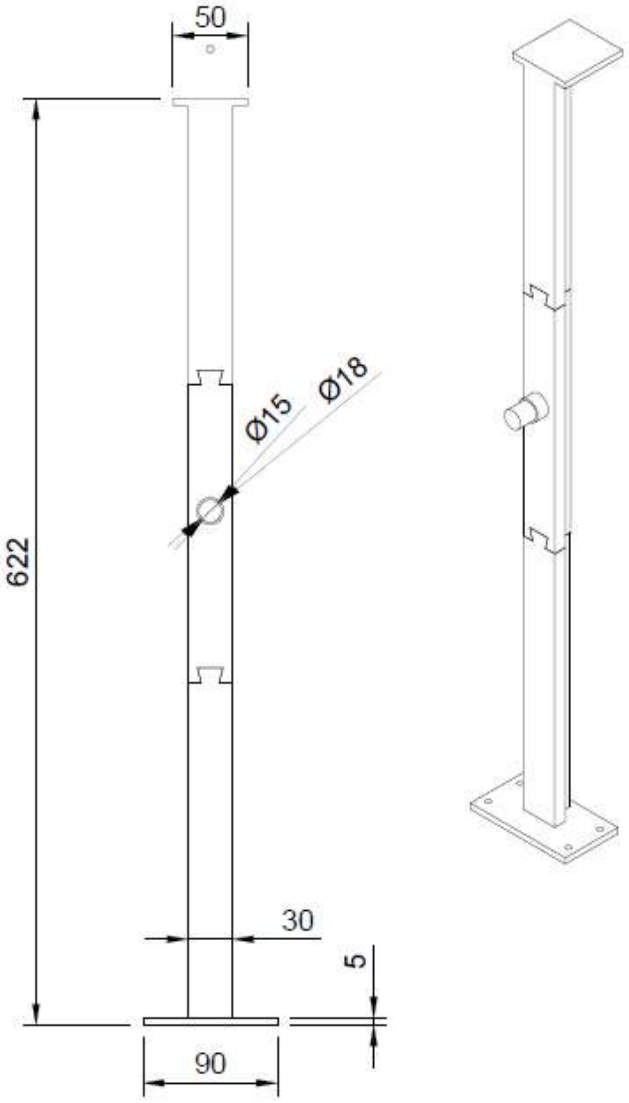
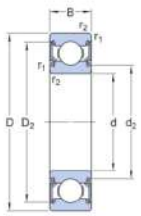



Figura B.5: Plano soporte vertical.

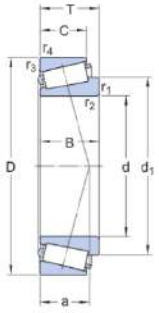
# Anexo C: Diagramas rodamientos




Dimensiones

| | | |
|---|---|---|
| d | 15 mm | Diámetro interno |
| D | 32 mm | Diámetro exterior |
| B | 9 mm | Ancho |
| $d_2$ | ≈ 18.3 mm | Diámetro del rebaje |
| $D_2$ | ≈ 28.2 mm | Diámetro del rebaje |
| $r_{1,2}$ | min. 0.3 mm | Dimensión del chaflán |

Figura C.2: Dimensiones rodamiento AOF 6002-2RS.




Dimensiones

| | | |
|---|---|---|
| d | 25 mm | Diámetro interno |
| D | 52 mm | Diámetro exterior |
| T | 16.25 mm | Ancho total |
| $d_1$ | ≈ 38 mm | Diámetro del resalte del aro interior |
| B | 15 mm | Ancho del aro interior |
| C | 13 mm | Ancho del aro exterior |
| $r_{1,2}$ | min. 1 mm | Dimensión del chaflán del aro interior |
| $r_{3,4}$ | min. 1 mm | Dimensión del chaflán del aro exterior |
| a | 12.33 mm | Distancia de la cara lateral al punto de presión |

Figura C.1: Dimensiones rodamiento WOKOST 30205.

# Anexo D: Fotografías del banco de ensayos

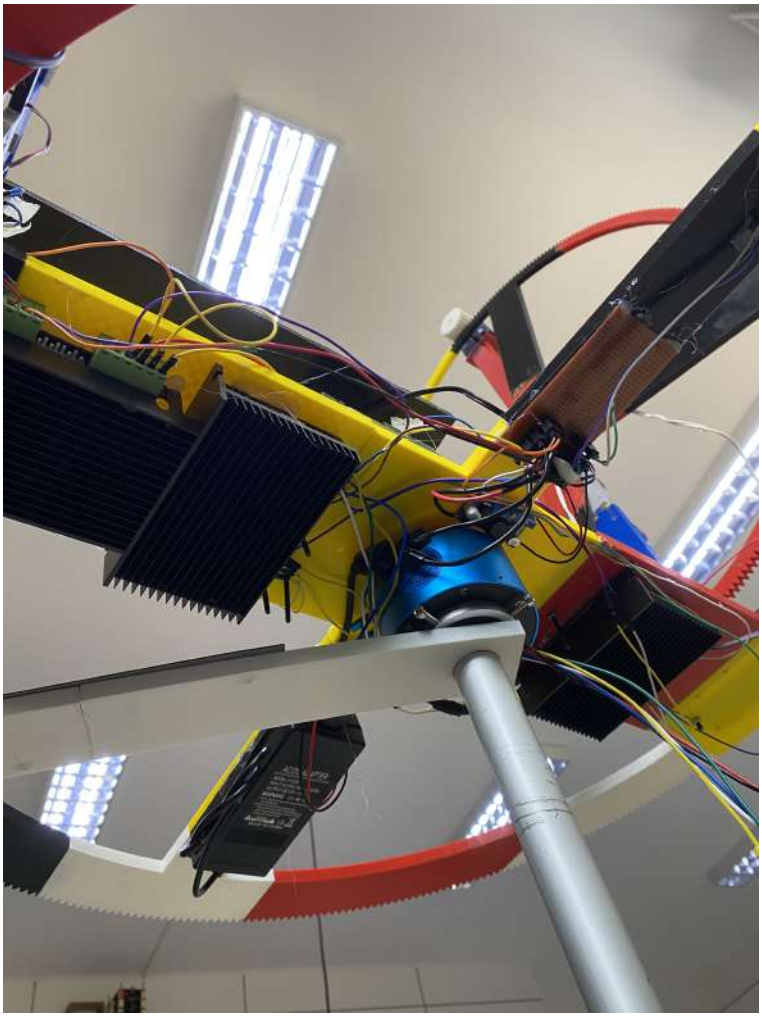


Figura D.1: Cara inferior del engranaje horizontal del banco de ensayos.

Figura D.2: Cara inferior del engranaje horizontal del banco de ensayos.


Figura D.3: Fotografía del banco de ensayos.

Figura D.4: Fotografía del banco de ensayos.

# Anexo E: Circuito del dispositivo de adquisición de datos

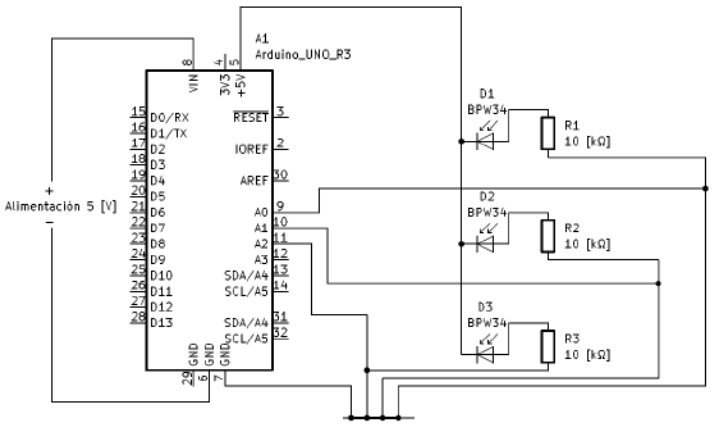



Figura E.1: Circuito electrónico del dispositivo de adquisición de datos.

# Anexo F: Códigos

### LecturaSensores

```
#include <SPI.h>
#include <SD.h>

const int pinFotodiodo1 = A0;
const int pinFotodiodo2 = A1;
const int pinFotodiodo3 = A2;

File dataFile;

void setup() {
  Serial.begin(9600);


  if (!SD.begin(10)) {
    Serial.println("SD Card initialization failed!");
    return;
  }


  dataFile = SD.open("datalog.txt", FILE_WRITE);

  if (dataFile) {
    Serial.println("Logging data to SD card...");
  } else {
    Serial.println("Error opening datalog.txt");
  }
}

void loop() {
  int ValorLuz1 = analogRead(pinFotodiodo1);
  int ValorLuz2 = analogRead(pinFotodiodo2);
  int ValorLuz3 = analogRead(pinFotodiodo3);

```

```
34
35    Serial.print(ValorLuz1);
36    Serial.print(",");
37    Serial.print(ValorLuz2);
38    Serial.print(",");
39    Serial.println(ValorLuz3);
40
41    if (dataFile) {
42      dataFile.print(ValorLuz1);
43      dataFile.print(",");
44      dataFile.print(ValorLuz2);
45      dataFile.print(",");
46      dataFile.println(ValorLuz3);
47      dataFile.flush();
48    } else {
49      Serial.println("Error␣writing␣to␣datalog.txt");
50    }
51    delay(500);
52  }
```

**Prueba1**

```
1  #define stepPin1 2
2  #define dirPin1 3
3
4  #define stepPin2 4
5  #define dirPin2 5
6
7  #define stepPin3 6
8  #define dirPin3 7
9
10 #define Rele1 11
11 #define Rele2 12
12
13
14 void setup() {
15
16   pinMode(stepPin1, OUTPUT);
17   pinMode(dirPin1, OUTPUT);
18   pinMode(stepPin2, OUTPUT);
19   pinMode(dirPin2, OUTPUT);
20   pinMode(stepPin3, OUTPUT);
21   pinMode(dirPin3, OUTPUT);
```

```
22
23    pinMode(Rele1, OUTPUT);
24    pinMode(Rele2, OUTPUT);
25  }
26
27  void loop() {
28
29    digitalWrite(dirPin1, HIGH);
30    digitalWrite(dirPin2, HIGH);
31    digitalWrite(Rele1, LOW);
32    digitalWrite(Rele2, HIGH);
33
34    for(unsigned int x=0; x<40000;x++){
35      digitalWrite(stepPin1, HIGH);
36      digitalWrite(stepPin2, HIGH);
37      delayMicroseconds(400);
38      digitalWrite(stepPin1, LOW);
39      digitalWrite(stepPin2, LOW);
40    }
41
42    digitalWrite(dirPin1, LOW);
43    digitalWrite(dirPin2, LOW);
44    digitalWrite(Rele1, HIGH);
45    digitalWrite(Rele2, LOW);
46
47    for (unsigned int x=0; x<40000;x++){
48      digitalWrite(stepPin1, HIGH);
49      digitalWrite(stepPin2, HIGH);
50      delayMicroseconds(400);
51      digitalWrite(stepPin1, LOW);
52      digitalWrite(stepPin2, LOW);
53    }
54  }
```

**Prueba2**

```
1    #define stepPin1 2
2    #define dirPin1 3
3
4    #define stepPin2 4
5    #define dirPin2 5
6
7    #define stepPin3 6
```

```
8   #define dirPin3 7
9
10  #define Rele1 12
11  #define Rele2 11
12
13  void setup() {
14
15    pinMode(stepPin1, OUTPUT);
16    pinMode(dirPin1, OUTPUT);
17    pinMode(stepPin2, OUTPUT);
18    pinMode(dirPin2, OUTPUT);
19    pinMode(stepPin3, OUTPUT);
20    pinMode(dirPin3, OUTPUT);
21
22    pinMode(Rele1, OUTPUT);
23    pinMode(Rele2, OUTPUT);
24  }
25
26  void loop() {
27    digitalWrite(Rele1, LOW);
28    digitalWrite(Rele2, HIGH);
29
30    digitalWrite(dirPin1, HIGH);
31
32    for (unsigned int x=0;x<20500; x++){
33      digitalWrite(stepPin1, HIGH);
34      delayMicroseconds(400);
35      digitalWrite(stepPin1, LOW);
36    }
37
38    digitalWrite(dirPin3, HIGH);
39    for (unsigned int x=0;x<59200;x++){
40      digitalWrite(stepPin3, HIGH);
41      delayMicroseconds(400);
42      digitalWrite(stepPin3, LOW);
43    }
44    for (unsigned int x=0;x<59200;x++){
45      digitalWrite(stepPin3, HIGH);
46      delayMicroseconds(400);
47      digitalWrite(stepPin3, LOW);
48    }
49
50
51    digitalWrite(dirPin1, LOW);
52    for (unsigned int x=0; x<20000;x++){
```

```
      digitalWrite(stepPin1, HIGH);
      delayMicroseconds(400);
      digitalWrite(stepPin1, LOW);
    }
    digitalWrite(Rele1, HIGH);
    delay(1000);

  }
```

**Orbita**

```

  #include <SD.h>
  #include <SPI.h>

  #define anguloAPasos (6400 / 360) * (290/20)

  #define pinStep1 2
  #define pinDireccion1 3
  #define pinStep2 4
  #define pinDireccion2 5
  #define pinStep3 6
  #define pinDireccion3 7

  #define Rele1 8
  #define Rele2 9

  int ahora = 0;
  int aux=0;

  File archivoCSV;

  void mover(int destino);
  void moverMotor(int motor, int destino);
  void stepper(int motor, int steps, int direccion);
  void leerArchivoCSV();

  void setup() {
    Serial.begin(9600);
    pinMode(pinStep1, OUTPUT);
    pinMode(pinDireccion1, OUTPUT);
    pinMode(pinStep2, OUTPUT);
    pinMode(pinDireccion2, OUTPUT);
```

```
33      pinMode(pinStep3, OUTPUT);
34      pinMode(pinDireccion3, OUTPUT);
35      pinMode(Rele1, OUTPUT);
36      pinMode(Rele2, OUTPUT);
37      SD.begin(10);
38      leerArchivoCSV();
39    }
40
41    void loop() {}
42
43    void leerArchivoCSV() {
44      archivoCSV = SD.open("destinos.csv");
45      if (archivoCSV) {
46        while (archivoCSV.available()) {
47          String linea = archivoCSV.readStringUntil('\n');
48          int valor = linea.toInt();
49          mover(valor);
50        }
51        archivoCSV.close();
52      } else {
53        Serial.println("No se pudo abrir el archivo CSV.");
54      }
55    }
56
57    void mover(int destino) {
58      if (ahora > 0) {
59        moverMotor(1, 0);
60        moverMotor(2, destino);
61      } else {
62        moverMotor(2, 0);
63        moverMotor(1, destino);
64      }
65    }
66
67    void moverMotor(int motor, int destino) {
68      int pasos = (abs(destino) - ahora) * anguloAPasos;
69      stepper(motor, pasos, (abs(destino) > 0) ? 1: 0);
70      ahora = destino;
71    }
72
73    void stepper(int motor, int pasos, int direccion) {
74      int pinDireccion = pinDireccion1;
75      int pinStep = pinStep1;
76      int mot3= pinStep3;
77      digitalWrite(Rele1, LOW);
```

```
78      digitalWrite(Rele2, HIGH);
79      if (motor == 2) {
80        pinDireccion = pinDireccion2;
81        pinStep = pinStep2;
82        digitalWrite(Rele2, LOW);
83        digitalWrite(Rele1, HIGH);
84      }
85
86      digitalWrite(pinDireccion, direccion);
87      aux=abs(pasos);
88      for (unsigned int x = 0; x < aux; x++) {
89        digitalWrite(pinStep, HIGH);
90        digitalWrite(mot3, HIGH);
91        delayMicroseconds(400);
92        digitalWrite(pinStep, LOW);
93        digitalWrite(mot3, LOW);
94      }
```

**Órbita Librería**

```
1  #include <AccelStepper.h>
2  #include <SD.h>
3  #include <SPI.h>
4
5  #define anguloAPasos (6400 / 360)* (290/20)
6  #define anguloAPasos2 (6400/360)*(370/20)
7
8  #define pinStep1 2
9  #define pinDireccion1 3
10 #define pinStep2 4
11 #define pinDireccion2 5
12 #define pinStep3 6
13 #define pinDireccion3 7
14
15 #define Rele1 11
16 #define Rele2 12
17 #define MAX_SIZE 100
18
19 AccelStepper stepper1(AccelStepper::DRIVER, pinStep1, pinDireccion1);
20 AccelStepper stepper2(AccelStepper::DRIVER, pinStep2, pinDireccion2);
21 AccelStepper stepper3(AccelStepper::DRIVER, pinStep3, pinDireccion3);
22
23 void mover(int destino1, int destino2, int aux);
```

```
24 void leerArchivoCSV();
25
26 void setup() {
27   Serial.begin(9600);
28   pinMode(Rele1, OUTPUT);
29   pinMode(Rele2, OUTPUT);
30   SD.begin(10);
31   leerArchivoCSV();
32 }
33
34 void loop() {}
35
36 void leerArchivoCSV() {
37   File archivoCSV = SD.open("destinos.csv");
38   if (archivoCSV) {
39     while (archivoCSV.available()) {
40       String linea = archivoCSV.readStringUntil('\n');
41       int comaIndex1 = linea.indexOf(',');
42       int comaIndex2 = linea.indexOf(',', comaIndex1 + 1);
43       if (comaIndex1 != -1 && comaIndex2 != -1) {
44         String valor1_str = linea.substring(0, comaIndex1);
45         String valor2_str = linea.substring(comaIndex1 + 1,
              comaIndex2);
46         String valor3_str = linea.substring(comaIndex2 + 1);
47         int valor1 = valor1_str.toInt();
48         int valor2 = valor2_str.toInt();
49         int valor3 = valor3_str.toInt();
50         mover(valor1, valor2, valor3);
51       }
52     }
53     archivoCSV.close();
54   } else {
55     Serial.println("No se pudo abrir el archivo CSV.");
56   }
57 }
58
59 void mover(int destino1, int destino2, int aux) {
60   Serial.print("Moving stepper1 to: ");
61   Serial.println(destino1);
62   Serial.print("Moving stepper2 to: ");
63   Serial.println(destino2);
64
65   stepper1.moveTo(destino1*anguloAPasos);
66   stepper3.moveTo(destino2*anguloAPasos2);
67
```

```
68    while (stepper1.distanceToGo() != 0 || stepper3.distanceToGo() !=
          0) {
69
70      stepper1.setMaxSpeed(2000.0);
71      stepper1.setSpeed(1000.0);
72      stepper1.runSpeedToPosition();
73
74      stepper3.setMaxSpeed(4000.0);
75      stepper3.setSpeed(2000.0);
76      stepper3.runSpeedToPosition();
77    }
78    if (aux == 1){
79      stepper3.setCurrentPosition(0);
80      Serial.println("set␣en␣0");
81    }
82    if (aux == 2){
83      stepper3.setCurrentPosition(360);
84      Serial.println("Set␣en␣360");
85    }
86  }
```