UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ETSI Aeroespacial y Diseño Industrial

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Aeroespacial y Diseño Industrial

Diseño de un sistema sobreactuado para la orientación de una cámara fotográfica.

Trabajo Fin de Máster

Máster Universitario en Ingeniería Mecatrónica

AUTOR/A: Guillén Romo, Santiago

Tutor/a: Puche Panadero, Rubén

Cotutor/a: Terrón Santiago, Carla

CURSO ACADÉMICO: 2023/2024

Resumen

En el presente trabajo de fin de máster se diseña y desarrolla un dispositivo sobreactuado para el control preciso de la posición de una cámara fotográfica. Primero, se definen los requisitos técnicos del dispositivo, que incluyen el rango de movimiento de cada eje y la velocidad mínima necesaria para la captura de imágenes en diferentes condiciones.

A continuación, se realiza una comparación exhaustiva de diferentes sistemas de actuación para seleccionar los actuadores más adecuados en términos de precisión, velocidad de respuesta y compatibilidad con otros componentes. También se seleccionan los sensores de posición, así como el microcontrolador encargado de procesar los datos y controlar los actuadores en tiempo real.

Con los componentes seleccionados, se procede al diseño de las conexiones entre cada elemento, considerando la alimentación eléctrica, la comunicación entre sensores, actuadores y el microcontrolador, y la integración de adaptadores si es necesario. Se crean esquemas y se realizan pruebas preliminares para asegurar que todas las conexiones funcionen correctamente.

El diseño de la estructura mecánica es crucial y debe permitir el movimiento en dos grados de libertad, soportando el peso de la cámara y albergar todos los componentes. Se utilizan herramientas de diseño asistido (CAD) para diseñar cada pieza y la fabricación se realiza por medio de impresión 3D

Una vez fabricada la estructura, se implementa la lógica de control mediante un sistema de comunicación por comandos, que permite al usuario controlar la posición y velocidad del dispositivo, así como las funciones de la cámara. Este sistema se programa en el microcontrolador y se realizan pruebas exhaustivas para asegurar su correcto funcionamiento.

Para verificar la efectividad del dispositivo, se llevan a cabo dos pruebas: una imagen panorámica para evaluar la capacidad de movimiento preciso y una fotografía de las estrellas para verificar la estabilidad y suavidad del movimiento durante largos períodos. Los resultados de estas pruebas demuestran que el dispositivo puede lograr un nivel de detalle y precisión imposible de alcanzar sin la asistencia de una montura motorizada como la diseñada.

Palabras clave: control; atmega238P; encoder; cámara fotográfica; dos grados de libertad; microcontrolador.

Resum

En aquest treball de final de màster es dissenya i desenvolupa un dispositiu sobreactuat per al control precís de la posició d'una càmera fotogràfica. Primer, es defineixen els requisits tècnics del dispositiu, que inclouen el rang de moviment de cada eix i la velocitat mínima necessària per a la captura d'imatges en diferents condicions.

A continuació, es realitza una comparació exhaustiva de diferents sistemes d'actuació per a seleccionar els actuadors més adequats en termes de precisió, velocitat de resposta i compatibilitat amb altres components. També es seleccionen els sensors de posició, així com el microcontrolador encarregat de processar les dades i controlar els actuadors en temps real.

Amb els components seleccionats, es procedeix al disseny de les connexions entre cada element, considerant l'alimentació elèctrica, la comunicació entre sensors, actuadors i el microcontrolador, i la integració d'adaptadors si és necessari. Es creen esquemes i es realitzen proves preliminars per a assegurar que totes les connexions funcionen correctament.

El disseny de l'estructura mecànica és crucial i ha de permetre el moviment en dos graus de llibertat, suportant el pes de la càmera i allotjant tots els components. S'utilitzen eines de disseny assistit per ordinador (CAD) per a dissenyar cada peça i la fabricació es realitza mitjançant impressió 3D.

Una vegada fabricada l'estructura, s'implementa la lògica de control mitjançant un sistema de comunicació per comandos, que permet a l'usuari controlar la posició i velocitat del dispositiu, així com les funcions de la càmera. Aquest sistema es programa en el microcontrolador i es realitzen proves exhaustives per a assegurar el seu correcte funcionament.

Per a verificar l'efectivitat del dispositiu, es duen a terme dues proves: una imatge panoràmica per a avaluar la capacitat de moviment precís i una fotografia de les estrelles per a verificar l'estabilitat i suavitat del moviment durant llargs períodes. Els resultats d'aquestes proves demostren que el dispositiu pot aconseguir un nivell de detall i precisió impossible d'assolir sense l'assistència d'una muntura motoritzada com la dissenyada.

Paraules clau: control; ATmega238P; codificador; càmera fotogràfica; dos graus de llibertat; microcontrolador.

Abstract

In this master's thesis, an over-actuated device for the precise control of a camera's position is designed and developed. First, the technical requirements of the device are defined, including the range of motion for each axis and the minimum speed necessary for capturing images in different conditions.

Next, an exhaustive comparison of different actuation systems is conducted to select the most suitable actuators in terms of precision, response speed, and compatibility with other components. Position sensors and the microcontroller responsible for processing data and controlling the actuators in real-time are also selected.

With the components selected, the design of the connections between each element is carried out, considering the electrical power supply, communication between sensors, actuators, and the microcontroller, and the integration of adapters if necessary. Schematics are created, and preliminary tests are conducted to ensure that all connections function correctly.

The design of the mechanical structure is crucial and must allow movement in two degrees of freedom, support the weight of the camera, and house all the components. Computer-aided design (CAD) tools are used to design each part, and fabrication is carried out through 3D printing.

Once the structure is fabricated, the control logic is implemented using a command-based communication system, which allows the user to control the device's position and speed, as well as the camera's functions. This system is programmed into the microcontroller, and exhaustive tests are conducted to ensure its proper functioning.

To verify the effectiveness of the device, two tests are carried out: a panoramic image to evaluate the precision of the movement and a photograph of the stars to verify the stability and smoothness of the movement over long periods. The results of these tests demonstrate that the device can achieve a level of detail and precision impossible to attain without the assistance of a motorized mount like the one designed.

Keywords: control; ATmega238P; encoder; camera; two degrees of freedom; microcontroller.

Documento:

Memoria

Índice

Índice de figuras

Índice de tablas

1. Objeto

El objetivo de esta memoria es el diseño mecánico, eléctrico, electrónico y de control de un sistema sobreactuado que permita la orientación de una cámara fotográfica en dos grados de libertad (yaw y pitch), Figura 1, proporcionando una orientación controlada y precisa. Este proyecto cumple con la normativa de trabajos finales de la ETSIADI y está diseñado para abordar los desafíos mencionados, mejorando la estabilidad y precisión en la captura de imágenes.

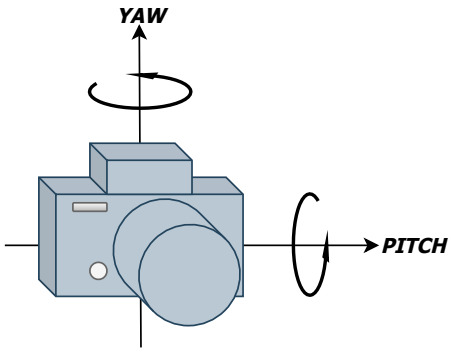


Figura 1: Ejes de movimiento yaw y pitch.

El uso de monturas fotográficas motorizadas, que permitan la orientación controlada de la cámara, abre la posibilidad a la obtención de imágenes de calidad como panorámicas, agrupaciones de fotos en intervalos amplios de tiempo (time lapse) o incluso la compensación de la rotación de la tierra para astrofotografía, donde se requiere una precisión extrema para seguir objetos en movimiento en el cielo nocturno. El movimiento y orientación de la cámara representa un desafío mayor conforme se usan objetivos de mayor aumento, pues estos harán que la presencia de vibraciones o una incorrecta orientación se magnifique en la imagen capturada.

Las principales dificultades a afrontar en el proyecto es:

1. **Tamaño**: La finalidad es la creación de una montura fotográfica que permita la orientación de manera precisa, esto se traduce en una montura ligera, firme y potable.

2. **Comunicación**: Establecer un sistema de comandos que permita controlar los distintos movimientos del sistema, como el control de la cámara.

Este proyecto abarca diversas disciplinas del Máster en Ingeniería Mecatrónica, incluyendo mecánica, eléctrica y electrónica. A lo largo del desarrollo del trabajo, se

aplicarán conocimientos y técnicas de estas áreas para diseñar un sistema innovador y eficiente que resuelva los problemas de orientación en cámaras fotográficas.

El alcance del trabajo cubre las siguientes competencias del Máster en ingeniería Mecatrónica de la UPV:

- Capacidad para proyectar, desarrollar, calcular, diseñar productos y sistemas mecatrónicos. Desarrollar y construir un argumento racional y lógico en la presentación de resultados. [32917/32918/32919]
- Capacidad para realizar investigación, desarrollo e innovación de productos, procesos y sistemas mecatrónicos. Presentación de trabajos científicos y técnicos oralmente y por escrito. [32913/32914]
- Capacidad de organización y planificación en proyectos de ingeniería mecatrónica. [32923]
- Capacidad de aplicar los conocimientos adquiridos y capacidad de resolución de problemas multidisciplinares. Toma de decisiones, iniciativa, creatividad y razonamiento crítico. [32911]
- Capacidad para el manejo, comprensión y aplicación de especificaciones, manuales, reglamentos y normas de obligado cumplimiento. [32922]
- Capacidad para la integración de tecnologías de control, automatización, electrónica, mecánica, electricidad e informática en el diseño de sistemas y productos mecatrónicos. [32920/32921]
- Capacidad para aplicar métodos y principios de calidad. Analizar y valorar el impacto social y medioambiental. [32923]
- Capacidad de comunicación, transmisión de conceptos, especificaciones y funcionalidades, tanto oralmente como de forma escrita. [32923]
- Capacidad para diseñar e implementar sistemas utilizados en mecatrónica. [32910]
- Conocimiento de algoritmos de accionamientos de motores. [32921]
- Conocimiento y utilización de arquitecturas hardware para sistemas mecatrónicos. [32920/32921]
- Conocimiento y utilización de software y algoritmos para sistemas mecatrónicos. [32920/32921]
- Capacidad para especificar, seleccionar e integrar dispositivos eléctricos y electrónicos en sistemas mecatrónicos. [32922]

- Capacidad para diseñar, modelar y seleccionar accionamientos electromecánicos utilizados en sistemas mecatrónicos. [32914/32915]

El resultado esperado es un sistema que no solo mantenga la estabilidad de la cámara, sino que también facilite la captura de imágenes de alta calidad, especialmente en aplicaciones exigentes como la fotografía astronómica.

1.1. Requerimientos

A continuación, se detallan los requerimientos necesarios para el proyecto. Es imprescindible disponer de un sistema de actuación en bucle cerrado que permita detectar perturbaciones en la posición, las cuales podrían afectar negativamente el desempeño de la escena a capturar. El rango de movimiento de la cámara en los ejes de *yaw* y *pitch* será de 90^{o} para el eje de *yaw* y de 360^{o} para el *pitch*. La estructura que contenga la parte electrónica, los sensores, motores, engranajes y la cámara debe permitir el movimiento libremente.

En cuanto a las aceleraciones y velocidades transmitidas a la cámara fotográfica, estas deben ser bajas y sin vibraciones, permitiendo transiciones fluidas. La velocidad mínima a la que debe moverse cada eje debe permitir el seguimiento de astros celestes, lo que se traduce en una velocidad de $\frac{1}{240}$ grados/segundo o $\frac{\pi}{43200}$ radianes/segundo.

Además, el dispositivo de orientación debe contar con un sistema de comunicación que permita su operación de forma remota, así como la recepción de comandos a través del puerto serie para la realización de seguimientos de trayectorias y control por coordenadas. Será necesario diseñar considerando la electrónica escogida, así como el trazado de los cables y conexiones.

1.2. Materiales y herramientas

En este apartado se indican las herramientas a las que se ha tenido acceso durante el desarrollo del trabajo:

- Ordenador con el software:
  - **SOLIDWORKS**: Software de diseño CAD utilizado para modelado 3D y simulaciones de estructuras mecánicas.
  - **MATLAB**: Entorno de programación para análisis de datos, simulaciones y desarrollo de algoritmos.
  - **SALEAE LOGIC 2**: Software libre de análisis lógico para la captura y análisis de señales digitales.
  - **CURA**: Software libre de impresión 3D que convierte modelos 3D en instrucciones para impresoras 3D.
  - **HUGIN**: Software libre para crear imágenes panorámicas a partir de múltiples fotos superpuestas.

  - **GIMP**: Software libre para la edición de imágenes.
  - **Deepskystacker**: Software libre para realizar apilado de imágenes.

- Analizador lógico de señales: AZ-Delivery, dispositivo para capturar y analizar señales digitales.
- Impresora 3D: Artillery Sidewinder X1, máquina utilizada para fabricar prototipos físicos a partir de modelos 3D.
- Soldador: Herramienta para unir componentes electrónicos mediante soldadura.
- Fuente de alimentación 12V: Dispositivo que proporciona energía eléctrica a circuitos y componentes.
- Protoboard y cables de conexión: Placa utilizada para construir circuitos electrónicos temporales sin necesidad de soldadura.
- Multímetro: Instrumento para medir magnitudes eléctricas como voltaje, corriente y resistencia.
- Calibre y regla: Instrumento para medir magnitudes de distancia con precisión.

1.3. Cámara

La cámara empleada es una Canon 250D, un modelo de cámara réflex digital (DSLR). Esta cámara está equipada con un objetivo de 50mm y cuenta con un sensor de formato APS-C, este conjunto es equivalente a trabaja un objetivo 75mm en una cámara Full-frame, como se puede ver en la Figura 2, el sensor APS-C tiene un tamaño mas reducido que el Full-frame y por tanto se pierde la información de los bordes del encuadre. Este tipo de conjunto, cámara con sensor APS-C mas objetivo de 50mm es muy popular por el rendimiento que obtiene en cuanto a calidad en fotografía general.

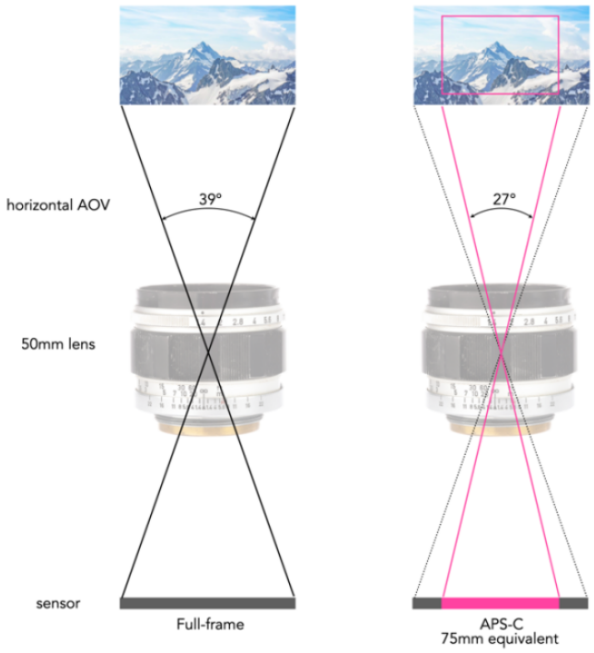


Figura 2: Equivalencia 50mm en sensor APS-C[1].

Características	**Especificaciones**
Dimensiones	122,4 x 92,6 x 69,8 mm
Peso	451 gramos (solo cuerpo con batería)
Tensión de trabajo	3,2 V (batería LP-E17 de 7,2V)
Tipo de sensor	CMOS APS-C

Tabla 1: Especificaciones de la cámara Canon 250D

2. Descripción del problema y objetos

En esta sección, se aborda en detalle el propósito del proyecto y el procedimiento seguido para su ejecución.

El uso de monturas fotográficas motorizadas es crucial para el control preciso de la orientación de una cámara fotográfica. Estas monturas permiten obtener imágenes de alta calidad en diversas situaciones, como la captura de panorámicas y la creación de secuencias time-lapse con intervalos de tiempo amplios. Además, son esenciales para la astrofotografía, donde es necesario compensar la rotación de la Tierra para seguir con precisión extrema los objetos en movimiento en el cielo nocturno. A medida que se utilizan objetivos con mayor aumento, el desafío de mantener la estabilidad y la correcta

orientación de la cámara se incrementa, ya que cualquier vibración o desalineación se magnifica en la imagen capturada. Por ello, una montura motorizada proporciona la estabilidad y el control necesarios para evitar estos problemas y asegurar la captura de imágenes nítidas y bien orientadas.

2.1. Objetivos

Para realizar el proyecto se han tenido que alcanzar los siguientes objetivos:

1. **Selección y Comparación de Componentes**: Se realizará una evaluación exhaustiva de los componentes necesarios para los sistemas de actuación, medición y control. Esto incluye motores, sensores y controladores. Cada componente será seleccionado en función de sus características.

2. **Diseño del Sistema Mecánico**: Se diseñará una estructura mecánica capaz de albergar la cámara fotográfica, permitiendo el movimiento en dos grados de libertad. Esta estructura debe ser robusta y precisa, albergando también parte del sistema de actuación y control.

3. **Integración de los Componentes**: Los componentes seleccionados se integrarán en un único dispositivo, controlado a través de un sistema de comunicaciones.

4. **Validación y Verificación**: Se llevará a cabo una serie de pruebas para verificar el comportamiento y desempeño de cada componente, así como del sistema en su conjunto. Esto asegurará que el sistema cumpla con los requisitos definidos y funcione correctamente en condiciones reales de uso.

5. **Documentación**: Se realizará un documento detallado justificativo y descriptivo de la solución adoptada que incluye el proceso de génesis de la misma, con la indicación de las principales dificultades encontradas y los procesos seguidos. en su resolución

2.2. Antecedentes

En la actualidad existen multitud de monturas que permiten la orientación y control de la orientación de una cámara fotográfica. Cada montura se enfoca a un único propósito ya sea la creación de transiciones entre escenas, la obtención de fotografías en intervalos de tiempo, o la compensación de la rotación de la tierra.

Además de los dispositivos comerciales, también existen numerosos dispositivos creados por la comunidad de "makers" utilizando plataformas de código abierto. Estos dispositivos, desarrollados y compartidos libremente, ofrecen una amplia variedad de funcionalidades y pueden ser adaptados a las necesidades específicas de cada usuario.

El objetivo no es crear un sistema que permita desempeñar un único propósito, si no, el crear un sistema motorizado que permita ser controlado mediante comandos y que por medio de dichos comandos pueda desempeñar cualquier propósito dentro de las limitaciones físicas del dispositivo.

2.2.1. Estudio de mercado

En está sección se presentan algunos productos/dispositivos destinados a orientar una cámara fotográfica:

El OlyCelotti, Figura 3, es una montura que permite la orientación de una cámara en 360º para el movimiento de yaw y 70º para el movimiento de pitch. El control se realiza por medio de un mando por radiofrecuencia, el coste del producto es de 140 €.

Figura 3: Montura OlyCelotti

El OpenAstroTracker, es una montura motorizada de dos grados de libertad enfocado a astrofotografia, de código abierto[2], la principal función es la de compensar la rotación de la tierra, controlado desde su propio software.

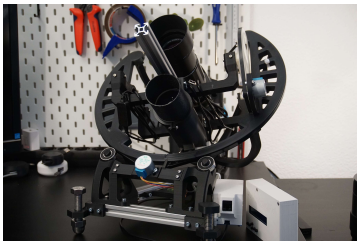


Figura 4: Montura OpenAstroTracker

Montura BRESSER StarTracker PM-100, es una montura de un único grado de libertad, controlado desde los botones y pantalla que integra, el propósito de esta montura es la de contrarrestar la rotación de la tierra para la realización de astrofotografia, el coste de este producto se encuentra en 230 €.

Figura 5: Montura BRESSER

El proyecto PanTiltMount[3], Figura 6, es un proyecto de código abierto para mover una cámara en dos grados de libertad, con la finalidad de realizar transiciones entre escenas de vídeo.

Figura 6: Montura PanTiltMount

2.3. Desarrollo del proyecto

El desarrollo del proyecto pasa por diferentes etapas, cada una de las cuales es crucial para lograr el objetivo final:

1. **Planteamiento preliminar del sistema**: Búsqueda de información sobre productos y dispositivos que realicen funciones similares, establecer objetivos a cumplir, determinar el medio por el cual alcanzar dichos objetivos.

2. **Diseño del dispositivo**: elección de los componentes a utilizar, determinar la relación entre componentes y su uso, realizar el diseño mecánico que albergue todas las partes.

3. **Ensamblado**: Con las partes diseñadas, y todos los elementos escogidos se procede a realizar el ensamblado de todas los elementos.

4. **Sistema de control**: Diseñar un sistema de control que permita la gobernabilidad del dispositivo por distintos medios.

5. **Pruebas**: Se realizan pruebas para determinar el funcionamiento de los movimientos y controles que tiene el dispositivo, búsqueda de fallos y mejoras.

6. **Documentación**: Como parte final del proyecto se realiza la presente memoria donde se detalla de manera técnica y económica los aspectos del proyecto.

2.4. Composición del trabajo

La documentación de la que se compone el presente trabajo se forma por diversos documentos, con el objetivo de cumplir con la normativa vigente relacionada a la presentación de trabajos finales de máster de la ETSIADI. El conjunto de todos los documentos descritos a continuación conforman los aspectos tecnio-economicos del proyecto. La memoria es un documento justificativo y descriptivo de la solución adoptada que incluye el proceso de génesis de la misma, con la indicación de las principales dificultades encontradas y los procesos seguidos en su resolución, la cual está estructurada de la siguiente forma:

- **Objeto**: Esta parte del documento presenta una introducción general al trabajo, explicando el contexto global en el que se enmarca y ofreciendo una introducción técnica de los elementos a tratar. Se justifica la necesidad de este proyecto y se resumen de manera breve los pasos seguidos y los objetivos a alcanzar.

- **Descripción del problema y objetos**: En esta sección se describe el problema, objetivos y alcance.

- **Planteamiento de soluciones alternativas y justificación de la solución adoptada**: Esta sección se dedica a la elección de los diferentes componentes para el montaje final, llevando a cabo comparaciones entre las opciones disponibles.

- **Descripción detallada de la solución adoptada**: Después de realizar la selección de componentes, se procede a detallar como estos van a interactuar.

- **Ensayos y resultados**: Se realizan las pruebas previas al funcionamiento y se comprueba el mismo.

- **Conclusiones**: Se efectúa un balance entre las expectativas indicadas al inicio del documento con los resultados obtenidos.

2.5. Justificación

El presente trabajo se realiza bajo el marco de la normativa aplicable a los trabajos finales de máster de la Escuela Técnica Superior de Ingeniería Aeroespacial y Diseño Industrial (ETSIADI). Según dicha normativa, para poder obtener el título de Máster en Ingeniería Mecatrónica, es necesario realizar, presentar un trabajo final de máster, el cual tiene una extensión de 12 créditos académicos.

3. Planteamiento de soluciones alternativas y justificación de la solución adoptada

En esta sección se realiza la elección de los componentes necesarios para la elaboración de la montura fotográfica motorizada que permita la orientación de la cámara de manera precisa y controlada.

El primer elemento a considerar es el sistema de actuación, este sistema es el encargado de dotar de movimiento a la montura, dada las características se precisa de un elemento de actuación, que destaque por tener una gran precisión y generar movimientos suaves, como opciones a considerar se encuentran los servos y los motores paso a paso.

Los servomotores ofrecen un valor elevado de par en comparación con su coste y tamaño, y no necesitan de ningún elemento adicional para su funcionamiento, en su encapsulado se encuentra el sensor y la electrónica que lo hacen trabajar en bucle cerrado. Sin embargo, debido a su construcción, la salida del servo se realiza a través de un tren de engranajes, lo que se traduce en movimientos lentos y con presencia de vibraciones, además de un rango de movimiento limitado.

Por otro lado, los motores paso a paso destacan por su precisión en cuanto a movimientos angulares discretos. Sin embargo, cuentan con una serie de desventajas, como la necesidad de ser controlados por un driver o la limitación en la velocidad de giro a altas velocidad.

En está aplicación se requiere de un elemento de actuación que permita mover su salida de posición con gran precisión, debido a que se plantea acoplar el elemento de actuación a un sistema de engranajes con la finalidad de aumentar la precisión en el eje del movimiento de la cámara, es crucial que el elemento de actuación permita que su salida pueda dar giros completos, esto hace que el servo como elemento de actuación quede descartado en esta etapa del diseño.

En cuanto a las consideraciones al par necesario, se ha realizado una estimación del valor de par, el cual es de 0,28Nm para el eje de movimiento pitch, este eje es el mas desfavorable porque en su rotación actúa la aceleración de la gravedad.

En cuanto al valor estimado de par necesario, tanto los servos como los motores paso a paso ofrecen un valor superior. Los servos al no ofrecer una salida con un movimiento continuo se descartan, escogiendo el motor paso a paso 42HS02, con el driver DVR8825.

Como en el apartado Objeto se mencionó, es de gran importancia que el sistema de actuación se encuentre en bucle cerrado, para garantizar que se ha alcanzado las referen-

cias en posición y poder corregir en caso de perturbaciones. Para lograr esto se pretende acoplar al eje del motor paso a paso un encoder capaz de medir la posición, los motores paso a paso nema17, suelen tener su salida en un único extremo del motor, por lo que los encoders magnéticos son ideales para esta aplicación, pues permiten alojar un imán en el lado opuesto a la salida del motor y justamente encima ubicar el sensor, de esta forma el conjunto motor más sensor están dispuestos de la manera que menos espacio requiere.

Se han comparado los siguientes sensores, el AS5600 y el AS5048A que son sensores magnéticos de posición angular que ofrecen soluciones precisas y fiables para diversas aplicaciones. El AS5600 es una excelente opción para aplicaciones generales que requieren un sensor de alta precisión y bajo consumo de energía. Sin embargo, para aplicaciones que demandan una mayor resolución y velocidad de comunicación, el AS5048A se destaca como la mejor opción. Con su resolución de 14 bits, el AS5048A proporciona una precisión significativamente mayor que el AS5600. Además, el uso de la interfaz SPI permite una comunicación más rápida, aunque requiere un mayor número de pines para funcionar. Por ello, se selecciona el encoder AS5600, debido su mayor precisión.

Como elemento de control, es necesario el microcontrolador que mejor se adapte a las necesidades del proyecto. Es fundamental definir previamente los requisitos específicos para controlar y comunicar todos los elementos anteriores. En primer lugar, el microcontrolador debe contar con 4 pines digitales para el control de los motores paso a paso. Además, se requiere comunicación SPI para obtener las lecturas de los sensores. Asimismo, se necesitan dos pines digitales adicionales para controlar la cámara (enfoque y disparo). Finalmente, es esencial que el microcontrolador disponga de comunicación serie para el envío y recepción de datos, en el Tabla 2, se encuentran los requisitos descritos.

Requisito	**Descripción**
Pines digitales para motores	4 pines digitales para el control de motores paso a paso
Comunicación SPI	Necesaria para obtener lecturas de los sensores
Pines digitales para cámara	2 pines digitales para enfoque y disparo de la cámara
Comunicación serie	Necesaria para el envío y recepción de datos

Tabla 2: Necesidades del microcontrolador

Para seleccionar el microcontrolador más adecuado para esta aplicación, se han evaluado las siguientes opciones: ATmega328P-PU, ESP32-D0WD-V3, RP2040 y STM32 (LQFP64). Tras un análisis detallado de las especificaciones y capacidades de cada uno, se ha determinado que el ATmega328P-PU es la opción más adecuada. Este microcontrolador de 8 bits cuenta con suficientes pines digitales y capacidades de comunicación necesarias para nuestro proyecto, como los 23 pines de E/S digitales, comunicación SPI y serie (UART), lo que lo hace apto para controlar los motores paso a paso, la cámara y obtener lecturas de los sensores. Por otro lado, los ESP32-D0WD-V3, RP2040 y STM32

(LQFP64) cuentan con capacidades muy superiores a los requisitos del proyecto.

Seguidamente se detallan todo los componentes comparados, los cuales se dividen en tres categorías principales: sistemas de actuación, sistemas de medición y sistemas de control. A continuación, se presenta una descripción detallada de cada uno de estos componentes y su función dentro del sistema.

3.1. Elemento de actuación

Como elemento de actuación, se pueden emplear distintos dispositivos para generar un movimiento controlado. En esta aplicación, se requiere de un actuador que permita el movimiento de rotación de manera controlada y precisa, suministrando el par necesario para mover el conjunto mecánico más la cámara fotográfica. Para lograrlo, se pueden considerar diversas alternativas, como el uso de servomotores, motores DC o motores paso a paso.

El elemento de actuación debe de cumplir con una serie de requerimientos, que permitan al sistema de orientación moverse de manera controlada y precisa. Seguidamente se detallan como se han obtenido los valores que determinan las características del elemento de actuación a seleccionar.

Para calcular el par necesario para el actuador se han tomado las siguientes suposiciones: el cuerpo de la cámara cuyas dimensiones son 122.4 x 92.6 x 69.8 mm con un peso de 449 g, puede ser considerado un paralelepípedo rectangular que rota por su eje horizontal. Como objetivo se va a considerar un teleobjetivo de 71 mm de diámetro y 122 mm de longitud con un peso de 480 g, se considera un cilindro macizo que rota por uno de sus extremos. Primero se ha calculado el par del objetivo en el punto mas desfavorable, el cual es cuando se encuentra a paralelo al suelo, después se calcula el momento de inercia del objetivo y del cuerpo de la cámara y finalmente se calcula el par para el caso de que rote a una velocidad de 1 rad/s. A continuación se muestran los pasos mencionados anteriormente.

El par (τ) necesario para mantener el objetivo en la posición mas desfavorable formando un ángulo θ de 90° con la horizontal se determina según:

$$\tau = F \cdot d \cdot \sin(\theta)$$

Donde:

$$F = m \cdot g$$
$$d = \frac{L}{2}$$

Dado que:

$$m = 0{,}48\,\text{kg}$$
$$L = 0{,}122\,\text{m}$$
$$g = 9{,}81\,\text{m/s}^2$$
$$\theta = 90^\circ \Rightarrow \sin(90^\circ) = 1$$

Primero calculamos la fuerza debido a la gravedad:

$$F = 0{,}48\,\text{kg} \cdot 9{,}81\,\text{m/s}^2 = 4{,}7088\,\text{N}$$

La distancia d es:

$$d = \frac{0{,}122\,\text{m}}{2} = 0{,}061\,\text{m}$$

El par es:

$$\tau = 4{,}7088\,\text{N} \cdot 0{,}061\,\text{m} \cdot 1 = 0{,}2872368\,\text{Nm}$$

Este valor de par será el valor máximo necesario para mantener el conjunto cuerpo de la cámara mas objetivo detenidos en una posición.

Para calcular el momento de inercia del cuerpo de la cámara que rota alrededor de su eje longitudinal:

$$I = \frac{1}{12} m(b^2 + c^2)$$

Dado que:

$$m = 0{,}449\,\text{kg}$$
$$b = 0{,}0926\,\text{m}$$
$$c = 0{,}0698\,\text{m}$$

Calculamos:

$$b^2 = 0{,}00857376\,\text{m}^2$$
$$c^2 = 0{,}00487204\,\text{m}^2$$

$$I = \frac{1}{12} \cdot 0{,}449 \cdot (0{,}00857376 + 0{,}00487204) = \frac{1}{12} \cdot 0{,}449 \cdot 0{,}0134458 = 0{,}0005045642\,\text{kg} \cdot \text{m}^2$$

El momento de inercia del objetivo se puede calcular como:

$$I = \frac{1}{3} mL^2 + \frac{1}{4} mR^2$$

Dado que:

$$m = 0{,}48\,\text{kg}$$
$$L = 0{,}122\,\text{m}$$
$$R = 0{,}0355\,\text{m}$$

Calculamos:

$$L^2 = 0{,}014884\,\mathrm{m}^2$$

$$R^2 = 0{,}00126025\,\mathrm{m}^2$$

$$I = \frac{1}{3} \cdot 0{,}48 \cdot 0{,}014884 + \frac{1}{4} \cdot 0{,}48 \cdot 0{,}00126025$$

$$I = 0{,}00238144 + 0{,}00015123 = 0{,}00253267\,\mathrm{kg} \cdot \mathrm{m}^2$$

El valor de par necesario para iniciar el movimiento puede determinarse de la siguiente manera, primero es necesario escoger el valor máximo al que esperamos mover el cuerpo de la cámara más el objetivo, para ello se ha seleccionado un valor de 0,5 rad/s, que equivale a 28,65 grados/s. En cuanto a la aceleración, debido a que la cámara es un elemento delicado, se va a escoger un valor de 2 segundos para alcanzar la velocidad seleccionada. Seguidamente se detallan los cálculos:

El momento de inercia total del conjunto es la suma de los momentos de inercia individuales:

$$I_{\text{total}} = 0{,}0005045642 + 0{,}00253267 = 0{,}0030372342\,\mathrm{kg} \cdot \mathrm{m}^2$$

La aceleración angular α es:

$$\alpha = \frac{\Delta\omega}{\Delta t} = \frac{0{,}5\,\mathrm{rad/s}}{2\,\mathrm{s}} = 0{,}25\,\mathrm{rad/s}^2$$

El par para proporcionar esta aceleración es:

$$\tau_{\text{aceleración}} = I_{\text{total}} \cdot \alpha = 0{,}0030372342 \cdot 0{,}25 = 0{,}00075930855\,\mathrm{Nm}$$

El par total necesario es la suma del par debido a la gravedad y el par para la aceleración:

$$\tau_{\text{total}} = 0{,}2872368\,\mathrm{Nm} + 0{,}00075930855\,\mathrm{Nm} = 0{,}28799610855\,\mathrm{Nm}$$

Por lo tanto, el par total necesario para empezar a mover el conjunto a 0,5 rad/s en 2 s es aproximadamente:

$$\tau_{\text{total}} \approx 0{,}288\,\mathrm{Nm}$$

A continuación, se exponen los candidatos a elemento de actuación con sus determinadas características:

3.1.1. Servomotores

Los servomotores son sistemas compuestos por un motor y un sensor, operando mediante una electrónica capaz de alcanzar una posición deseada corrigiendo errores o perturbaciones que puedan surgir. Actualmente, los servomotores son ampliamente utilizados en una multitud de aplicaciones, ofreciendo una amplia variedad de potencias. Dado el enfoque del proyecto, donde no solo se busca realizar una elección adecuada en cuanto a las especificaciones de salida (par/velocidad) del elemento de actuación, sino también se deben considerar aspectos como el peso y tamaño. Teniendo esto en cuenta, se han escogido los siguientes modelos como candidatos para el elemento de actuación del proyecto:SG90, MG995 y HS-422.

Estos servomotores se encuentran comúnmente en juguetes RC (coches, aviones), por lo que son de un tamaño y peso reducidos. El rango de operación de estos servomotores se encuentra entre 0-180° o 0-360°, por lo que no es posible realizar giros de manera continua. Esta limitación se debe por lo general a que el sensor de posición no está preparado para dar una vuelta completa, en la Figura 7, se encuentra un diagrama que contiene todos los elementos existentes en el interior de un servo.

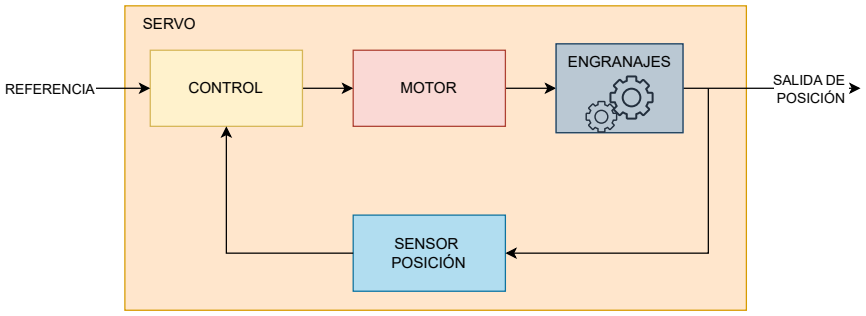


Figura 7: Diagrama elementos de un servo.

En la Figura 8, se encuentran los siguientes modelos SG90, MG995 y HS-422, los cuales son modelos muy expendidos por su ámbito en el mundo del radio control y por consiguientes son modelos fáciles de encontrar en el mercado y con un coste reducido.

Figura 8: Servos: SG90, MG995 y HS-422.

Estos servos tiene las características que se muestran en la Tabla 3:

Característica	**SG90**	**MG995**	**HS-422**
Par (N·m)	0,18	0,98	0,40
Velocidad	0,10 s/60°	0,16 s/60°	0,16 s/60°
Voltaje de Operación	4,8-6V	4,8-7,2V	4,8-6,0V
Rango de Operación	180°	180°	45°
Peso (g)	9	55	45

Tabla 3: Comparación de Modelos de Servomotores más Utilizados

3.1.2. Motor paso a paso

Los motores paso a paso están formados por un conjunto de bobinas agrupadas por polos en el estator, y un rotor de imanes permanentes. La activación de una bobina alinea el rotor; mediante la conmutación secuencial de las bobinas, crea un movimiento discreto, correspondiente a los grados eléctricos entre los polos. Además, las expansiones polares están ranuradas de manera característica para facilitar el movimiento discreto del rotor con cada conmutación de bobina.

Uno de los modelos más empleados es el 42HS02, que pertenece a la familia NEMA 17, debido a sus prestaciones en par (0,5 N·m) y precisión (1,8° por paso), lo que lo hace que se encuentre en multitud de dispositivos como impresoras, plotters, telescopios y equipos de laboratorio.

Este motor, a diferencia de los servomotores indicados anteriormente, no puede funcionar sin un controlador (driver) que realice las conmutaciones en el orden adecuado tanto para avanzar como para retroceder.

El driver más empleado para el control de motores paso a paso de este tamaño es el A4988 de la marca "POLOLU". Este driver cuenta con un rango de operación de 8 a 35 V, permite manejar corrientes de hasta 2 amperios por bobina, además de poder variar el paso del motor pudiendo escoger entre el paso completo y el paso fraccionado, como se puede observar en la Tabla 4, para variar esta configuración solo es necesario la conexión a un nivel alto o bajo en los pines **MS1**, **MS2** y **MS3**.

La manera de conseguir que el motor paso a paso consiga movimientos discretos inferiores al valor determinado por su construcción, se realiza por medio del control de las corrientes que circulan por sus devanados. A continuación en la Figura 9 se puede visualizar las variaciones de corriente. Comparando esta figura con la Figura 10, se puede apreciar que el número de saltos de corriente se ha incrementado al doble, lo que permite dividir el paso del motor a la mitad, lo que se traduce para un motor paso a paso de 1.8° en un paso de 0,9°.

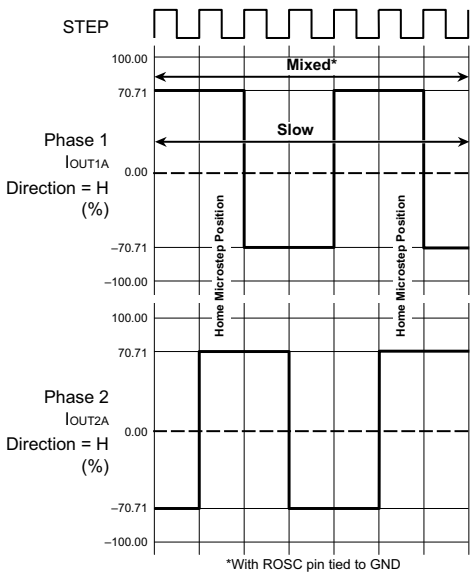


Figura 9: Corriente en modo Full Step.

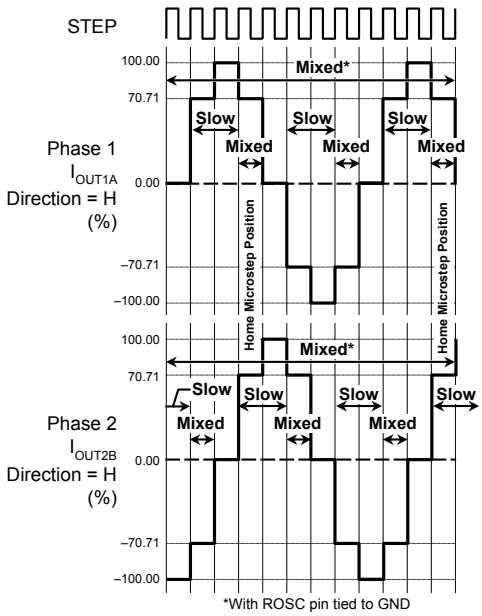


Figura 10: Corriente en modo Half Step.

En el caso extremo, el driver permite crear 16 divisiones en la corriente, lo que se traduce en un paso de 0,11°. Esta resolución en la salida del eje del motor tan elevada tiene un claro inconveniente: no es válida para realizar movimientos rápidos debido a la gran cantidad de pulsos que es necesario enviar al driver. Recalcar que cuanto mas divisiones se crean, la corriente toma la forma de onda de una señal senoidal.

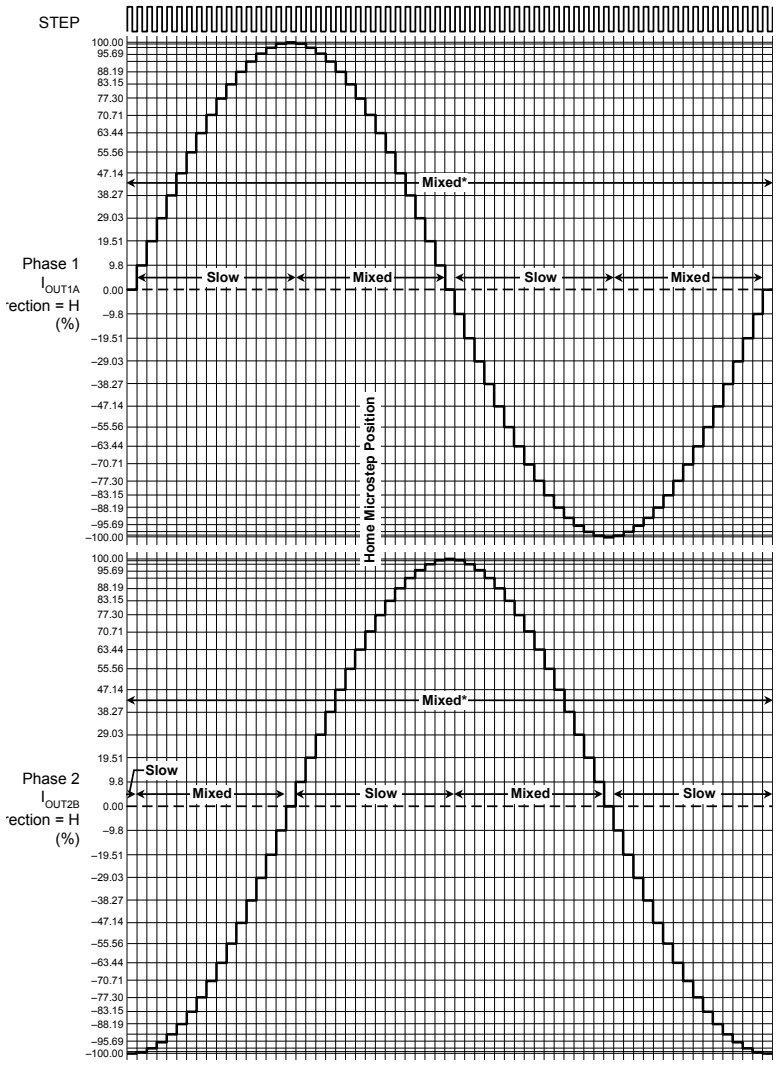


Figura 11: Configuración de corriente en microstepping.

El DVR8825 es otro driver comúnmente utilizado para controlar motores paso a paso y presenta varias ventajas sobre el A4988. La principal ventaja del DVR8825 es la reducción del ruido durante la operación del motor, lo que permite un funcionamiento más suave y silencioso. Ambos drivers pueden manejar corrientes similares y tienen configuraciones de microstepping, pero el DVR8825 tiene un rango de operación más amplio, de 8,2 a 45 V, y permite manejar corrientes de hasta 2,5 amperios por bobina en el caso de que cuente con disipador. Además, el DVR8825 proporciona hasta 32 configuraciones de microstepping en comparación con las 16 del A4988, lo que permite un movimientos discretos mas reducidos.

MS1	MS2	MS3	Resolución
Low	Low	Low	Full
High	Low	Low	Half
Low	High	Low	Quarter
High	High	Low	Eighth
High	High	High	Sixteenth

Tabla 4: Configuración de MS1, MS2, MS3 y resolución del driver A4988

MS1	MS2	MS3	Resolución
Low	Low	Low	Full
High	Low	Low	Half
Low	High	Low	Quarter
High	High	Low	Eighth
Low	Low	High	Sixteenth
High	Low	High	Thirty-second
Low	High	High	Thirty-second
High	High	High	Thirty-second

Tabla 5: Configuración de MS1, MS2, MS3 y resolución del driver DVR8825

Estas diferencias remarcan que el DVR8825 es un driver mas avanzado acontinuación se muestran en la Tabla 6 los datos mas relevantes de estos drivers. La elección entre el DVR8825 y el A4988 dependerá de los requerimientos específicos de la aplicación, incluyendo el rango de voltaje, la precisión del movimiento y las condiciones operativas.

Característica	A4988	DVR8825
Rango de Voltaje	8 a 35 V	8.2 a 45 V
Corriente por bobina	Hasta 2 A	Hasta 2.5 A
Microstepping	Hasta 16	Hasta 32
Ruido	Alto	Bajo

Tabla 6: Comparación de drivers A4988 y DVR8825

3.2. Sensores

En esta sección se detallarán los sensores empleados para medir y controlar el movimiento de la cámara. Estos sensores son cruciales para proporcionar retroalimentación al sistema de control y garantizar que el movimiento de la cámara sea preciso y estable. Los sensores incluyen encoders para medir la posición angular, acelerómetros para detectar movimientos no deseados y giróscopos para proporcionar datos de orientación.

3.2.1. Encoders

Los encoders son dispositivos que convierten el movimiento mecánico en señales eléctricas digitales que pueden ser interpretadas por un sistema de control. Existen dos tipos principales de encoders: incrementales y absolutos. Los encoders incrementales proporcionan información sobre el cambio de posición, mientras que los encoders absolutos proporcionan información sobre la posición exacta en todo momento.

A continución, se van a comparar dos encoders magnéticos el AS5600 y el AS5048A desarrollados por *ams OS*, una empresa dedicada al desarollo y fabricación de sensores,

sistemas de iluminación y chips. Estos sensores se utilizan en diversas aplicaciones que requieren una detección precisa de la posición angular, como en la automatización industrial, sistemas automotrices, y dispositivos de consumo.

El AS5600 es un sensor magnético de posición angular sin contacto que proporciona una salida analógica o digital basada en el ángulo del campo magnético. Este sensor destaca por su precisión de 12 bits, bajo consumo de energía y facilidad de integración al contar con comunicación por I2C.

El AS5048A es un sensor magnético de posición angular de 14bits de precisión. Este sensor es adecuado para situaciones que demandan una precisión mayor a de más de contar con un sistema de autocalibración y diagnóstico.

En la Figura 12, se pueden ver los dos encoders comparados, a la derecha el AS5600 y a la izquierda el AS5048A, en esta figura puede apreciarse el menor número de pines respecto al AS5048A, debido al menor número de pines de la comunicación I2C fente a la comunicación SPI. En la Tabla 7, se encuentran las características fundamentales de cada uno.

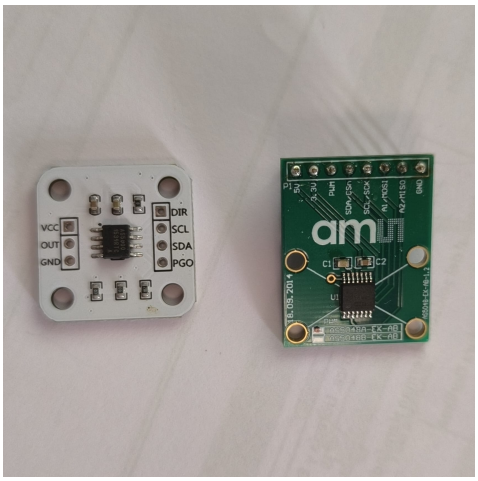


Figura 12: Encoder AS5600 y AS5048A.

Característica	**AS5600**	**AS5048A**
Rango de medición	0° a 360°	0° a 360°
Resolución	12 bits (4096 pasos por revolución)	14 bits (16384 pasos por revolución)
Comunicación	PWM, I2C	SPI, PWM
Alimentación	3,3V o 5V	3,3V o 5V
Consumo de energía	6,5mA	15mA

Tabla 7: Características entre AS5600 y AS5048A

3.3. Microcontrolador

Los microcontroladores son dispositivos electrónicos que integran los componentes esenciales de un ordenador en un solo chip. Estos sistemas embebidos incluyen un procesador, memoria y periféricos de entradas/salidas (I/O) en una única unidad. Los microcontroladores son diseñados para realizar tareas específicas, los microcontroladores juegan un papel fundamental en aplicaciones embebidas, controlando procesos y dispositivos electrónicos en una variedad de productos y sistemas, dada su capacidad para ser programados para distintas aplicaciones.

Los microcontroladores constan de una Unidad Central de Procesamiento (CPU) que ejecuta las instrucciones del programa, memoria de programa (ROM o Flash) que almacena el código a ejecutar, y memoria de datos (RAM) que proporciona almacenamiento temporal para variables del código. Además, cuentan con periféricos de entrada/salida que facilitan la interacción con otros dispositivos y componentes externos, tales como puertos digitales y analógicos, temporizadores e interfaces de comunicación (como $UART$, SPI, I^2C).

A continuación se realiza una comparación entre los siguientes microcontroladores: el ATmega328P-PU, el ESP32-D0WD-V3, el RP2040 y el STM32 (LQFP64).

- El ATmega328P-PU es un microcontrolador de 8 bits de la familia AVR, fabricado por Microchip Technology (anteriormente Atmel). Es ampliamente conocido por su uso en plataformas de desarrollo como Arduino, debido a su versatilidad. Este microcontrolador requiere de muy pocos elementos para su funcionamiento, además, es un microcontrolador del que hay mucha información y herramientas.
- El ESP32-D0WD-V3 es un microcontrolador de 32 bits fabricado por Espressif Systems. Este microcontrolador es conocido por integrar en un mismo encapsulado un microcontrolador con capacidades de conectividad, incluyendo Wi-Fi y Bluetooth, a un precio reducido.
- El RP2040 es un microcontrolador de doble núcleo basado en ARM Cortex-M0+, desarrollado por Raspberry Pi Foundation. Este microcontrolador es conocido por su alta capacidad de memoria y su bajo coste.
- El STM32 es una familia de microcontroladores basados en ARM Cortex-M, fabricados por STMicroelectronics. En particular, el modelo LQFP64 es conocido por su alta capacidad de procesamiento y su amplia gama de periféricos. Es utilizado en aplicaciones industriales y comerciales donde se requiere un rendimiento confiable y versátil.

En la Tabla 8, se ofrece una visión general de las especificaciones clave de cada microcontrolador para realizar una comparación en base a sus características.

Característica	ATmega	ESP32	RP2040	STM32
Arquitectura	AVR 8 bits	32 bits	ARM Cortex-M0+	ARM Cortex-M3
Número de núcleos	1	2	2	1
Velocidad de reloj	16 MHz	240 MHz	133 MHz	72 MHz
Memoria Flash	32 KB	Interna	2 MB	256 KB
SRAM	2 KB	520 KB	264 KB	64 KB
EEPROM	1 KB	No	No	Hasta 4 KB
Wi-Fi	No	Sí	No	No
Bluetooth	No	Sí	No	No
GPIO	23 pines	34 pines	30 pines	Hasta 80 pines
ADC	10 bits	12 bits	No	12 bits
Canales ADC	6	18	No	24
DAC	No	Sí, 8 bits	No	Sí 2 x 12 bits
UART	Sí	Sí	Sí	Sí
SPI	Sí	Sí	Sí	Sí
I2C	Sí	Sí	Sí	Sí
SDIO	No	Sí	No	Sí
CAN	No	No	No	Sí
PWM	Sí	Sí	Sí	Sí
Voltaje de operación	1,8V a 5,5V	3,0V a 3,6V	1,8V a 3,3V	2,0V a 3,6V
Precio aproximado	1,8€ - 2,7€	3,6€ - 5,4€	0,9€ - 1,8€	4,5€ - 9€

Tabla 8: Comparación de Microcontroladores

4. Descripción detallada de la solución adoptada

Este apartado engloba la parte del diseño eléctrico, mecánico y de control, detallando los aspectos relevantes, donde el propósito es el diseño de un dispositivo que englobe todas las partes y trabaje de la manera deseada.

4.1. Esquema eléctrico

En este apartado se muestran los esquemas realizados, necesarios para el funcionamiento del proyecto. Una vez escogido el microcontrolador, el Atmega328P, es necesario realizar la elección de los pines necesarios para la aplicación. En la Figura 13 se encuentran los pines del microcontrolador.

Se necesitan 4 pines digitales para controlar los dos motores paso a paso (step y dir). Para controlar el enfoque y disparo de la cámara se usarán dos salidas digitales, así como los pines para la comunicación SPI que tienen que ser del pin a 16 al 19 más un pin digital por cada dispositivo conectado.

En este caso, se usará el pin 1 como reset, los pines 2 y 3 para la comunicación por el puerto serie, los pines digitales 4 y 5 para el control de la cámara, función de enfocar y disparo, los pines 7 y 8 corresponden a la alimentación y tierra, los pines digitales 6,

11, 12 y 13 se emplearán como control para los drivers de los motores paso a paso, del pin 14 a 19 se usarán para las comunicaciones SPI y por último los pines 27 y 28 se emplearán para una comunicación por I2C o serie.

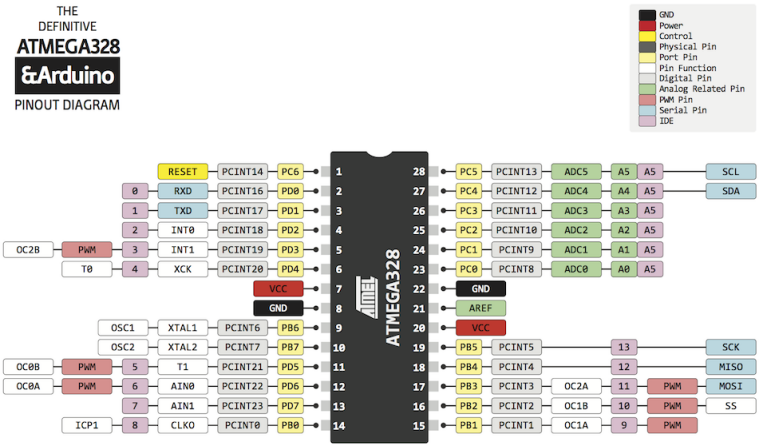


Figura 13: Pines Atmega328P-PU

Cabe mencionar que cada parte de los siguientes esquemas se han probado por separado para garantizar su funcionamiento y la correcta integración en futuros pasos del diseño como la fabricación de la PCB.

4.1.1. Esquema mínimo de funcionamiento del Atmega328P

El microcontrolador Atmega328P requiere un conjunto específico de componentes para su funcionamiento, entre los cuales se incluyen un oscilador, una fuente de alimentación y un sistema de reinicio. En la Figura 14 se presenta el circuito mínimo necesario para el funcionamiento del microcontrolador. Este esquema se ha creado siguiendo las directrices proporcionadas por la página de Arduino, aprovechando que dicha información es de acceso libre [4].

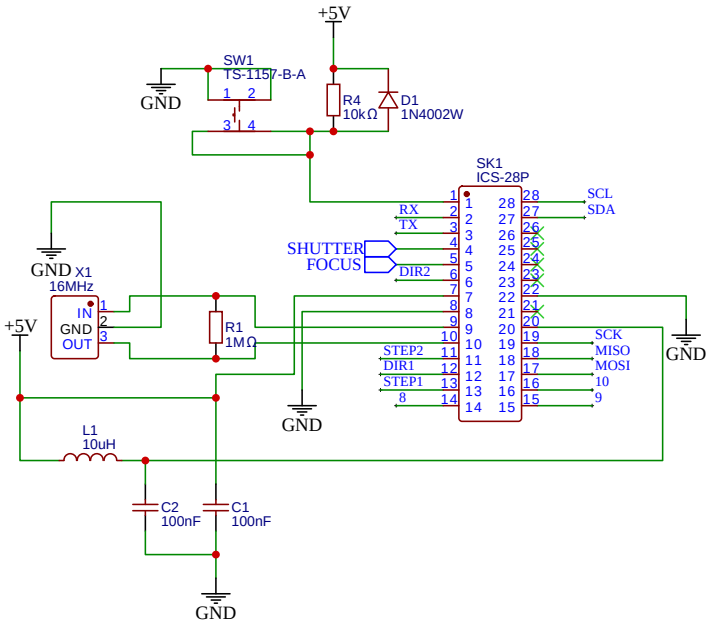


Figura 14: Esquema mínimo del Atmega328P

En el esquema de la Figura 14, se pueden ver como en la parte superior los componentes SW1, R4 y D1 forman parte del sistema de reset del microcontrolador. Como circuito oscilador se cuenta con X1 y R1, suele ser habitual el uso de condensadores para la estabilización de la frecuencia del oscilador, pero el modelo escogido *Murata Electronics CSTNE16M0VH3C000R0* cuenta dentro de su propio encapsulado estos condensadores, como se puede ver en la Figura 15. Por ultimo L1, C1 y C2 realizan la función de filtros en los pines de alimentación del integrado y alimentación del puerto C, respectivamente (pines 7 y 20).

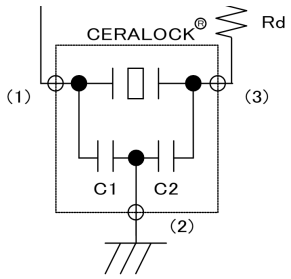


Figura 15: Simbolo del oscilador Murata Electronics

4.1.2. Adaptador de USB a TTL

Para poder programar, enviar y recibir datos desde el microcontrolador se necesita de un adaptador de USB (Universal Serial Bus) a TTL (Transistor-Transistor Logic), para ello se ha escogido el CH340C, el cual es un adaptador que no necesita ser progra-

mado ni necesita de oscilador. Para realizar el esquema se ha tenido la consideración de nombrar los pines de datos como D_P y D_N para posterior mente trazar los pares diferenciales que unirán el terminal hembra del USB con los pines 5 y 6 del CH340C, por otro lado se conectara los pines RX y TX respectivamente a los pines correspondientes del Atmega328 Tx y RX. Para finalizar el esquema se añaden los condensadores recomendados en el datasheet del fabricante y unos LEDs para poder visualizar si se están enviando o recibiendo datos.

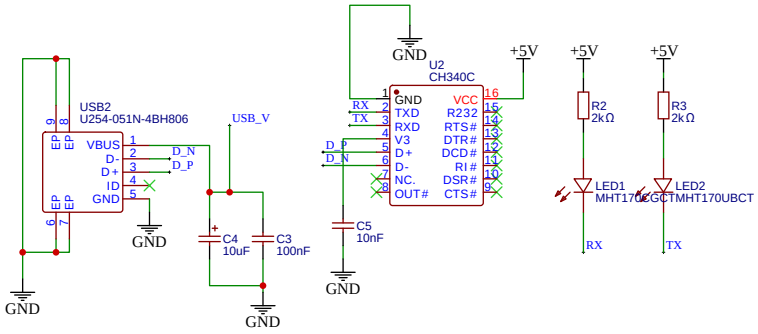


Figura 16: Adaptador USB a TTL

4.1.3. Regulador de tensión

Tanto el microcontrolador como los encoders y driver de los motores paso a paso necesitan una alimentación o de un nivel de trabajo como es el caso de los driver, para lograrlo se ha recurrido a un regulador de tensión, *MaxLinear SPX1117M3-L-5-0/TR*, el cual ofrece una salida regulada de 5V con una salida de 800mA. El regulador debe alimentar exclusivamente el Atmega328P y los encoders AS5048A, el Atmega328P puede consumir hasta un máximo de 200mA [5], mientras que éste cuenta con un consumo máximo de 15mA [6], por otro lado los los drivers de los motores paso a paso, solo necesitan un nivel lógico como referencia ya que se alimentan por medio de la entrada de 12V. Teniendo esto en cuenta, el regulador escogido es capaz de alimentar los componentes previstos de manera adecuada.

En la Figura 17, se muestra el esquema del regulador de tensión donde la entrada al regulador se realizar por medio del conector circular DC1, y pasa a traves del diodo D2, el cual es un diodo de propósito general de 1A cuya función es la de garantizar la correcta polaridad de la alimentación de entrada, seguido por el condensador C6, el cual cumple la función de filtro de entrada. Por otro lado, se ha puesto un conmutador deslizante U6, para permitir la selección de la alimentación del nivel de 5V.

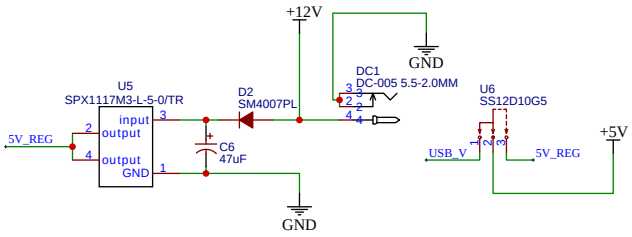


Figura 17: Regulador de tensión 12V a 5V

4.1.4. Conexion a los drivers de los motores paso a paso

Para la conexión de los drivers de los motores paso a paso, se ha seguido las indicaciones realizando el esquema recomendado para el uso indicado en la página de POLOLU [7]. En la Figura 18, se encuentra el esquema realizado, donde se ha realizado un puente en los pines RESET y SLEEP, debido a que no se va a realizar control que precise del cambio de estado en estos pines pero es necesario habilitar el drivers para su funcionamiento, se utiliza el pin de RESET, el cual es un pin que se encuentra en flotante para conectarlo al SLEEP y conseguir un nivel alto, consiguiendo habilitar la placa. Debajo de estos pines se encuentra el STEP y DIR desde donde se gobernará el driver, a continuación los pines VDD y GND se utilizan para suministrar el nivel lógico de operación, los pines 1A, 1B, 2A y 2B son los pines que se conectaran a las bobinas del motor y finalmente, VMOT es por donde se alimenta el driver.

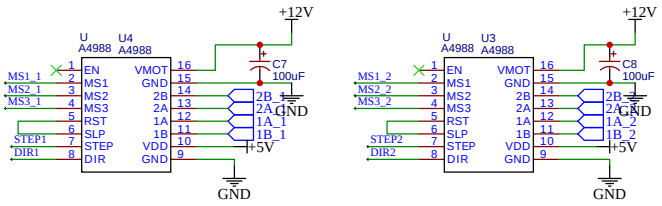


Figura 18: Dreivers motores paso a paso

Para poder configurar el paso de cada motor se han puesto unos pines con la intención de insertar unos jumpers, conmutando entre nivel alto o nivel bajo. En la Figura 19, se muestra como están realizadas las conexiones por medio de unas resistencias de PULLDOWN.

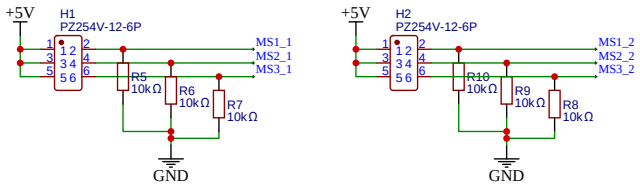


Figura 19: Configuración de pasos de los drivers

4.1.5. Circuto opto-acopladores

Para realizar el control del enfoque y del disparo de la cámara, es necesario realizarlo por medio de opto-acopladores debido a que de esta manera se incorpora un aislamiento entre circuitos que funcionan a tensiones distintas. En la Figura 20, se encuentra el 4N26 el cual es un optoacoplador de tipo transistor, para trabajar con el operaremos sobre los pines 1, 2, 4 y 5, que corresponden al ánodo, cátodo, emisor y colector, respectivamente.

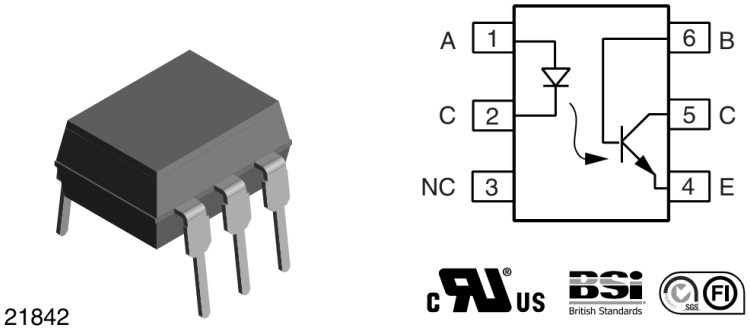


Figura 20: Optoacoplador 4N26

Primero es necesario calcular la resistencia que limita la corriente por el diodo, en el dado del transistor sólo se utilizara para conmutar los niveles lógicos de las entradas de la cámara. La resistencia se puede calcular tomando los valores típicos que se pueden encontrar en el datasheet[8], de la caída de tensión cuando se polariza directamente el diodo, la cual es de 1,2V tomando este valor y calculando para que circule una corriente de 10mA da como resultado 380Ω, como se puede observar en la Ecuación 1.

$$R = \frac{VDD - V_{diodo}}{I_{diodo}} = \frac{5 - 1,2}{0,01} = 380\Omega \tag{1}$$

Para el calculo de la corriente de colector se tiene en consideración el valor CRT, que es la relación de corriente que permite circular por cada amperio que circula por la entrada del diodo, en la Ecuación 2 se muestra está relación, en este caso el 4N26 cuenta con

un valor mínimo del 20 % y un valor típico del 50 %, lo que quiere decir que por cada 10mA que circulen por el diodo, circularan por la base aproximadamente la mitad. En el caso de no circular ninguna corriente por el LED, la corriente entre el colector y el emisor debería de ser cero, el datasheet indica que para un valor de $V_{CE} = 10V$, la corriente en este caso es de 5nA, esté valor es muy reducido y se puede aproximar a un interruptor abierto. En el caso contrario, cuando el LED se encuentre encendido la corriente del transistor (I_{CE}) corresponderá a los $70\mu A$, valor de la corriente cuando se conectan el pin de disparo a GND.

$$I_c = \left(\frac{CTR}{100}\right) I_f \tag{2}$$

Los emisores de cada optoacoplador se encuentran conectados entre sí y al negativo de la cámara, de esta forma se puede conmutar los niveles lógicos de los pines de enfoque y disparo de la cámara para controlarlo. A continuación, en la Figura 21, se encuentra el esquema utilizado.

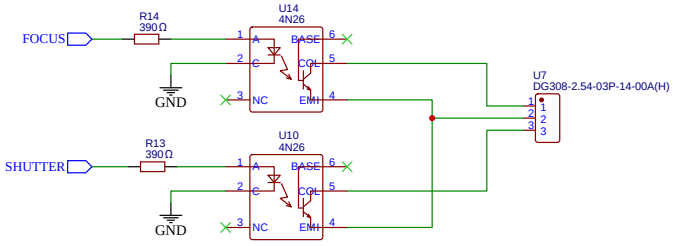


Figura 21: Opto-acopladores

4.2. Diseño PCB

Tras ensayar individualmente cada parte de los esquemas del apartado anterior, se ha decidido realizar una PCB que contenga todos los elementos anteriores, con el propósito de trabajar en un espacio mas reducido teniendo los componentes ordenados y sin los problemas que se pueden generar el trabajar mediante una protoboard.

Para el diseño de la PCB se ha escogido EasyEDA, el cual es un software de diseño electrónico de código abierto. Este programa permite realizar esquemas electrónicos y diseñar placas de circuito impreso (PCBs) de manera rápida. Una de las ventajas clave de EasyEDA es su conexión con servicios de fabricación, lo que facilita la transición desde el diseño hasta la producción. Esto lo convierte en una herramienta ideal para proyectos que requieren un desarrollo ágil y una implementación rápida.

Para la fabricación de las PCBs desde EasyEDA, se ha escogido a JLCPCB. Para

solicitar la fabricación de una PCB a través de JLCPCB, es necesario proporcionar varios archivos cruciales que definen el diseño y los requisitos de ensamblaje de la placa. Estos incluyen los archivos Gerber, que contienen toda la información gráfica de cada capa de la PCB y son esenciales para la producción; el archivo BOM (Bill of Materials), que detalla todos los componentes que deben ser montados en la placa; y los archivos Pick-and-Place, que indican la posición exacta y la orientación de cada componente en la placa, facilitando así la automatización del proceso de ensamblaje.

Partiendo del esquema, que se encuentra en el documento Planos, se ha tratado de ubicar cada elemento en un espacio de 60x75mm, tratando de facilitar el trazado de las pistas, una vez ubicados todos los elementos, se ha empleado herramientas de cálculo conforme a la IPC-2221[9] para determinar el ancho de cada pista, para ello se introducen las características de la Tabla 9, cada pista no debe soportar mas de 15mA, la fabricación se realizará a 1oz/ft^2 y las pistas aproximadamente no excederán de 20mm.

Current	0.015 Amps
Thickness	1 oz/ft^2
Temperature Rise	10 Deg C
Ambient Temperature	25 Deg C
Trace Length	20 mm

Tabla 9: Datos de entrada.

Con estos parámetros en partida se obtienen los resultados de las tablas 10 y 11. Ante estos resultados, se ha escogido un ancho de 0,254mm, siendo este el valor por defecto para realizar trazados en EasyEDA.

Required Trace Width	0.00238 mm
Resistance	4.24 Ohms
Voltage Drop	0.0635 Volts
Power Loss	0.000953 Watts

Tabla 10: Resultados para las pistas internas.

Required Trace Width	0.000916 mm
Resistance	11.0 Ohms
Voltage Drop	0.165 Volts
Power Loss	0.00248 Watts

Tabla 11: Resultados para las pistas externas.

Se repite el procedimiento para los trazados de 12v correspondientes a la alimentación de los drivers y a la alimentación de los motores paso a paso, de tal manera que el valor de corriente máxima se estima que será de 2A, con una longitud máxima de 1cm y manteniendo el resto en parámetros de entrada se obtiene un valor de ancho de pista de 0,06mm

Además de generar los archivos Gerber, esenciales para la fabricación de PCBs, se han enviado a la planta de fabricación los archivos de listado de materiales (BOM) y de

posicionamiento de componentes (Pick-and-Place). De esta manera se puede solicitar la PCB con los componentes soldados en sus respectivas ubicaciones.

Como resultado se han obtenido las capas de las figuras 22 a 25, en este caso se ha diseñado empleando un proceso de fabricación de cuatro capas, debido a que en instante de realizar el pedido, el coste de fabricación a cuatro capas era inferior.

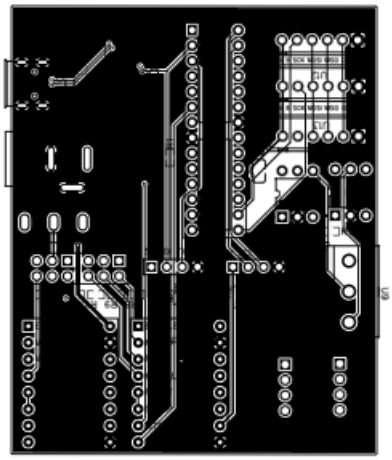


Figura 22: Capa superior.

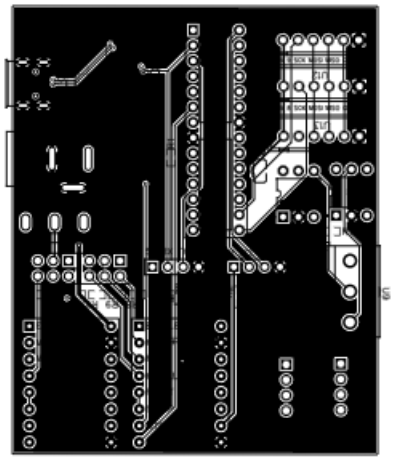


Figura 23: Capa intermedia 1.

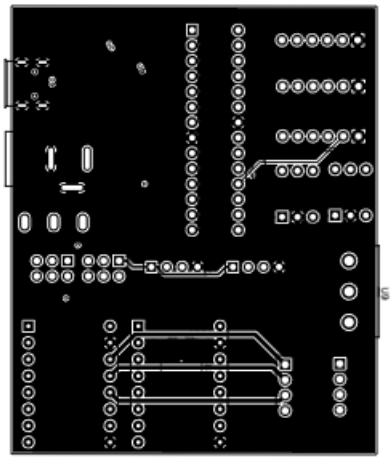


Figura 24: Capa intermedia 2.

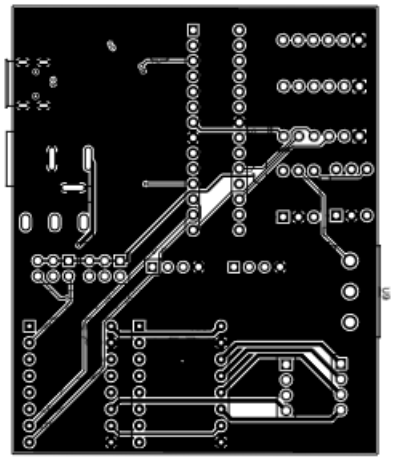


Figura 25: Capa inferior.

A continuación, en la Figura 26, se encuentra la PCB fabricada con los componentes soldados. Se ha probado cada elemento de la placa verificando su correcto funcionamiento: regulador de tensión, conexiones con entre el socket del microcontrolador y

conectores de los sensores, comunicaciones y optoacopladores.

Debido a un fallo en el diseño de la PCB, el emplazamiento de los drivers (footprint) es ligeramente mas grande, la separación entre pines es de 2,6mm en vez de 2,54mm esta diferencia hacía que el driver no encajara, por lo que se modifica el footprint correspondiente a los drivers.

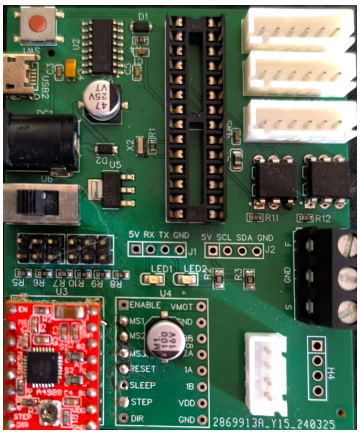


Figura 26: Fotografía del primer prototipo de PCB.

Tras detectar y corregir el error, se volvió a realizar el pedido, esta vez modificando el tamaño de la PCB a 85,6x54mm agregando unos agujeros para poder sujetarlo posteriormente a la estructura. A diferencia de la primera PCB, ésta se fabricó a dos capas, debido al coste, en esta nueva PCB no hay pistas internas como si las había en la anterior. En las figuras 27 y 28, se encuentra, el diseño del trazado más la serigrafía.

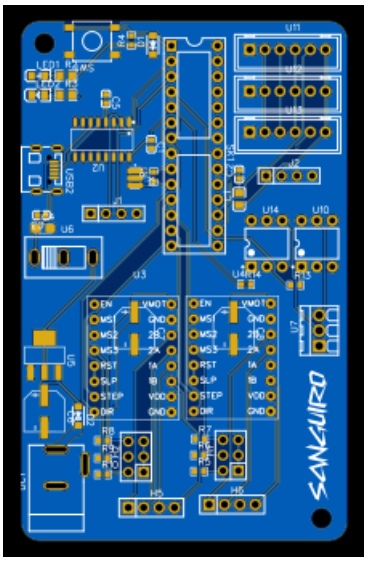


Figura 27: Capa intermedia 2.

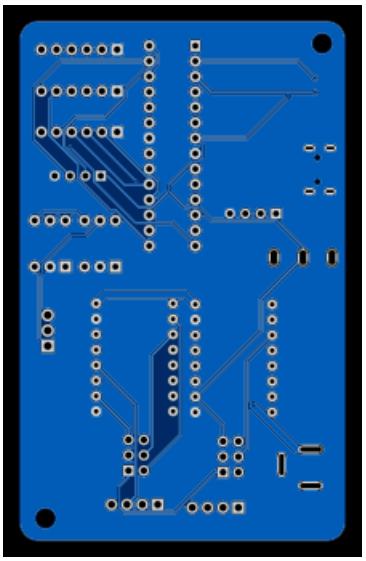


Figura 28: Capa inferior.

A continuación, en la Figura 29, se encuentra la segunda PCB realizada, a la derecha la PCB sin componentes y a la izquierda con todos los componentes ubicados. Se han realizado verificaciones para comprobar que todas las partes (tensiones, comunicaciones salidas digitales, conexiones de los drivers) son correctas después de solventar el el problema de espaciado entre los pines y reubicar de forma distinta los componentes.

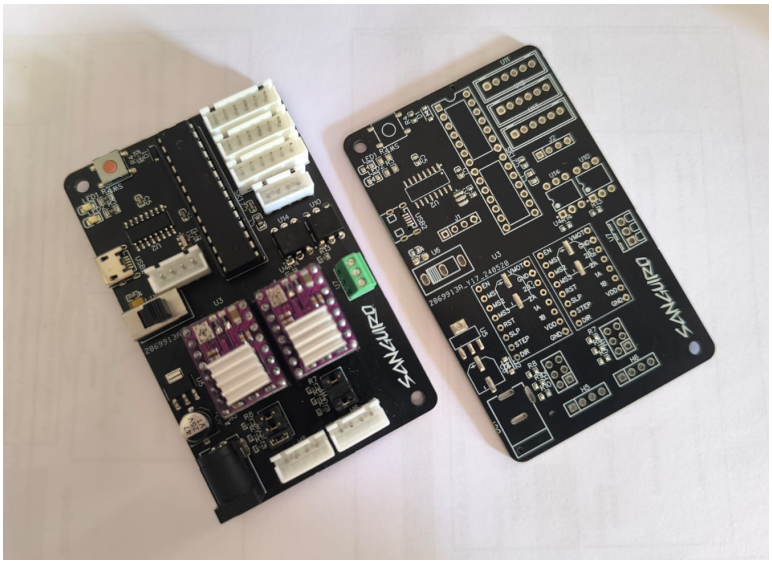


Figura 29: Fotografía del segundo prototipo de PCB.

4.3. Diseño mecánico

Como parte del trabajo, es necesario diseñar una estructura que no solo permita alojar cada componente, motores, cámara, electrónica... sino también permitir el movimiento de la cámara en dos grados de libertad, en sus ejes pitch y yaw, de manera suave. Para comenzar, partiendo de las dimensiones de la cámara se realizará el conjunto de piezas que permitirán su rotación en pitch, posteriormente se realizará el diseño para realizar la rotación en el eje yaw.

En la Figura 30, se encuentran las dimensiones de la cámara, la cual cuenta con 122mm de ancho por 93mm de alto, el centro del objetivo se encuentra a una altura de 30mm y la sujeción por medio de tornillo a 50mm del lado derecho.

Figura 30: Dimensiones de la cámara.

Seguidamente, el apartado de diseño mecánico se estructura en de la siguiente manera, se ha dividido en los movimientos de pitch y yaw, dentro de cada apartado se describe cada elemento y como esté interviene en la función del movimiento.

Reglas para el diseño de las piezas:

Dado que todas las piezas se fabricarán mediante una impresora 3D FDM (Fused Deposition Modeling), es esencial diseñarlas con caras planas que faciliten la adhesión a la base de la impresora. Por lo tanto, se deben evitar voladizos, elementos ocultos y grandes volúmenes separados de la base. Esta consideración obliga a diseñar los elementos por separado, lo que permite corregir errores en piezas concretas. Es más sencillo volver a generar una pieza pequeña de geometría sencilla que una grande con un gran volumen.

Además, las piezas se diseñan utilizando plástico tipo PLA (ácido poliláctico), con un espesor mínimo de 1 cm y un relleno elevado para garantizar la resistencia mecánica. Debido al proceso de fabricación en capas, este material presenta un comportamiento anisotrópico, es decir, sus propiedades varían según la dirección de la carga aplicada. Realizar análisis de esfuerzos en estas piezas es complicado debido a la cantidad de datos necesarios y a las variables que cambian de una pieza a otra; por esta razón, no se realiza ningún análisis de esfuerzo.

4.3.1. Movimiento pitch

El movimiento de pitch permite rotar respecto su eje longitudinal, para realizar esto se toma la distancia desde la base hasta el centro del objetivo, haciéndolo rotar en este eje.

En la Figura 31, se ve el conjunto diseñado, donde en la parte central se ubica la

cámara sujetada un tornillo emplazado en la ranura de la parte trasera, de manera que se pueda desplazar hasta encontrar un equilibrio entre el cuerpo de la cámara y el objetivo. En la parte izquierda se encuentran los engranajes, encargados de transmitir el movimiento. En los extremos se encuentran las piezas que sostienen este conjunto por medio de un par de rodamientos así como el motor paso a paso.

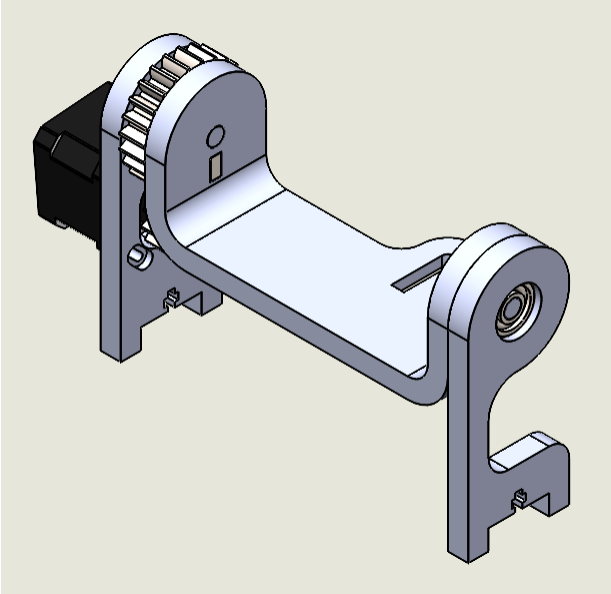


Figura 31: Mecanismo movimiento Pitch.

En la Figura 32, están todos los elementos que intervienen en el movimiento de Pitch.

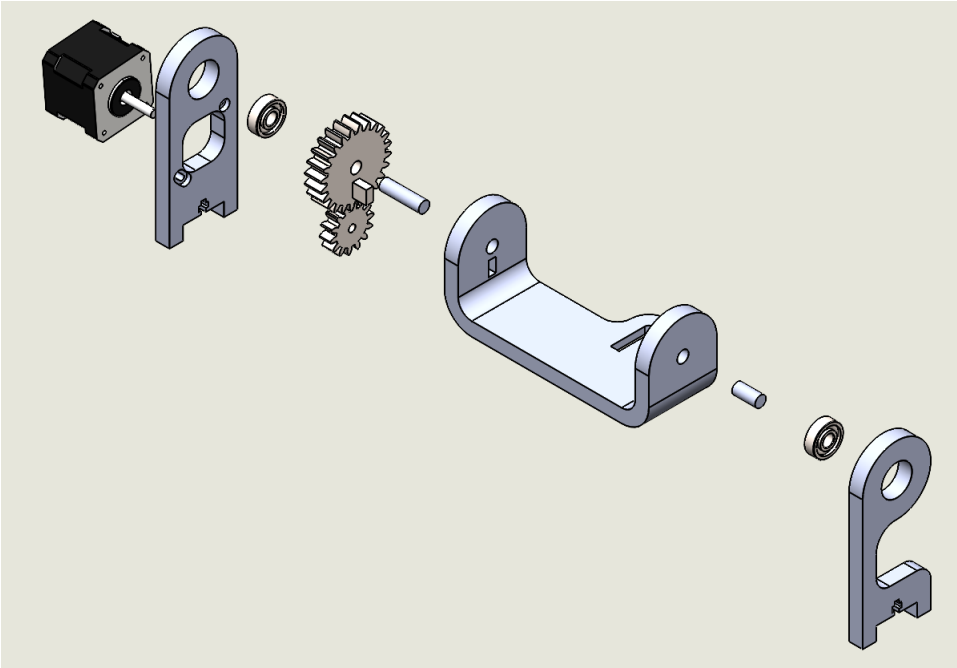


Figura 32: Explosión movimiento Pitch.

Soporte de la cámara

Esta es la pieza es la encargada de sostener la cámara fotográfica, la cual se ancla por medio de un tornillo de 1/4 de pulgada, la cámara se puede deslizar ligeramente hacia delante y atrás hasta encontrar el punto de equilibrio entre el cuerpo de la cámara y el objetivo. En uno de sus lados tiene una hendidura que permite al engranaje transmitir el movimiento de rotación, en la Figura 33 se puede ver esta hendidura en el lado izquierdo.

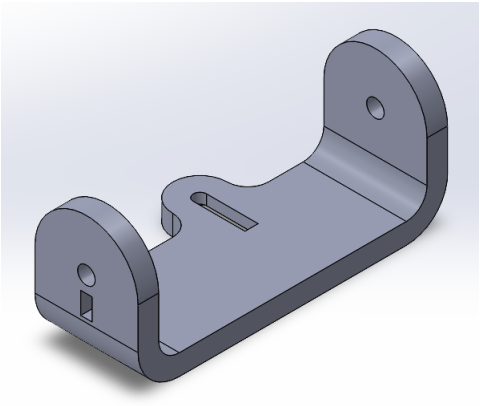


Figura 33: Soporte cámara.

Engranajes

Los engranajes cumplen la función de transmitir el movimiento, además de que la relación de transmisión desempeña un gran papel al poder obtener movimientos más suaves y precisos además de aumentar el par aplicado a la carga. Para ello se ha determinado que la distancia entre los dos ejes, salida del motor y eje de rotación de la cámara, será de 39,5 mm. Se ha determinado por medio de ensayos que el mejor módulo para poder ser fabricados por medio de impresión 3D es el módulo 2, el cual alcanza un balance entre el tamaño de los dientes y la resolución de salida de la impresora.

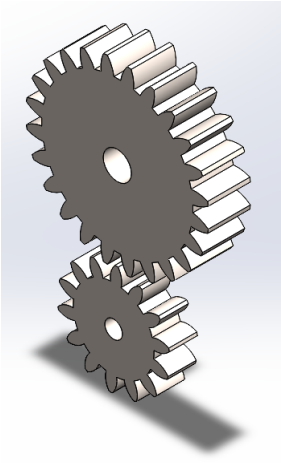


Figura 34: Engranajes movimiento Pitch.

La distancia entre centros (C) en un sistema de engranajes está dada por la fórmula:

$$C = \frac{(d_1 + d_2)}{2}$$

donde d_1 y d_2 son los diámetros primitivos de los engranajes 1 y 2, respectivamente. El diámetro primitivo (d) de un engranaje está relacionado con el número de dientes (Z) y el módulo (m) por la fórmula:

$$d = Z \times m$$

Dado que el módulo (m) es 2 y la distancia entre centros (C) es 39.5 mm, podemos escribir:

$$39{,}5 = \frac{(Z_1 \times 2 + Z_2 \times 2)}{2}$$

Simplificando la ecuación:

$$39{,}5 = Z_1 + Z_2$$

En este caso se ha escogido un valor para la rueda conductora de 14 y 24 para la conducida, dando una separación entre ejes de 38mm y una relación de transformación de 1,71. El escoger un paso que no permite la alcanzar los 39,5mm entre ejes previstos, no supone un problema pues el elemento que sostenga el motor tendrá una excentricidad que permita realizar un ajuste fino para adaptar la longitud a la que más se ajuste.

Soportes laterales

Los soportes, Figura 35, cumplen varias funciones, la principal es sostener el soporte de la cámara, de tal manera que la altura a la que la sostienen se ha fijado para que la cámara no colisione con el resto de la estructura.

Atendiendo al soporte del lado izquierdo, se puede ver que esté tiene una forma distinta, se ha dado esa forma para permitir que el conector de disparo remoto de la cámara no colisione cuando la se orienta la cámara en hacia el eje horizontal. En la parte inferior de ambos soportes, se han realizado dos salientes y en el centro un vaciado en forma de cruz, con el objetivo de insertar los salientes en la base y hacer pasar un tornillo con una trueca por la cruz, de esta manera se generar una unión formando un ángulo recto entre la base y los soportes.

El soporte de la derecha sujeta el motor paso a paso, como se puede ver se sujeta desde dos puntos, el superior, es fijo mientras que el de la parte inferior se móvil, esto produce que el eje del motor se comporte de manera excéntrica y permite variar la longitud entre el eje del motor y el eje de rotación.

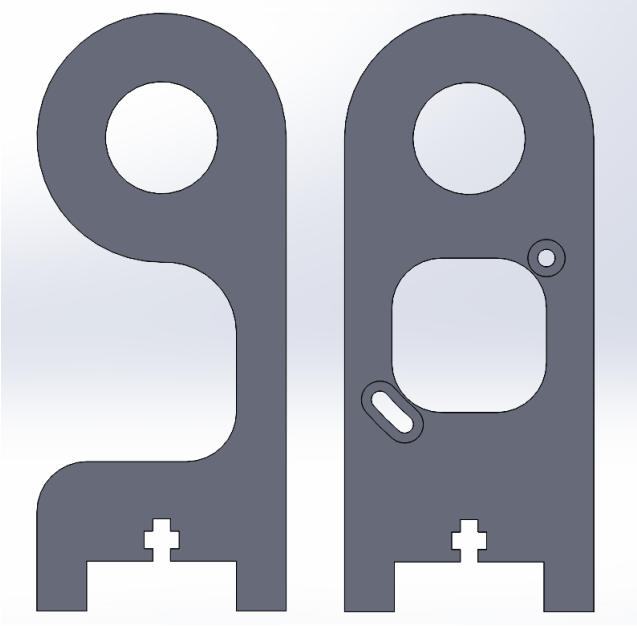


Figura 35: Soportes laterales.

Rodamientos

Para unir el soporte de la cámara con los soportes laterales se han empleado rodamientos modelo 608ZZ, que cuentan con 8mm de diámetro interior 22mm de diámetro exterior y 7mm de espesor, junto a unos pasadores de 8mm se mantienen todas las partes sujetas.

4.3.2. Movimiento yaw

El movimiento de yaw permite la rotación en el eje vertical, el conjunto de piezas que permiten esto también debe de alojar la electrónica, el motor y mecanismo que transmita el movimiento, así como un soporte firme para soportar el conjunto de piezas del movimiento yaw y pitch.

En las figuras 36 y 37, se ven el conjunto de piezas empleadas para poder realizar los propósitos antes mencionados. A continuación, se expone cada una de las piezas, comenzando de la parte superior.

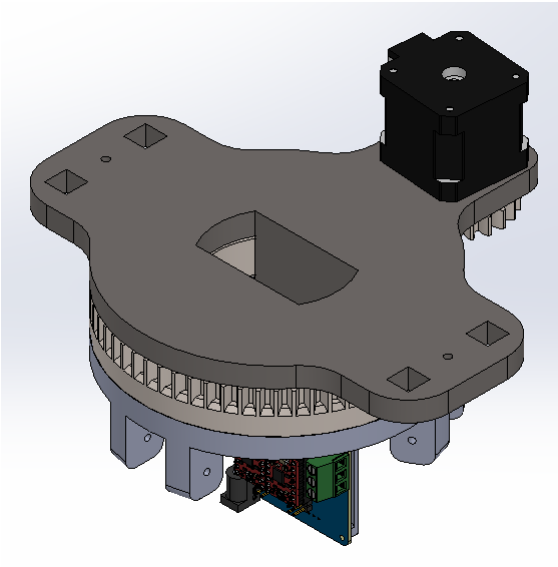


Figura 36: Mecanismo movimiento yaw.

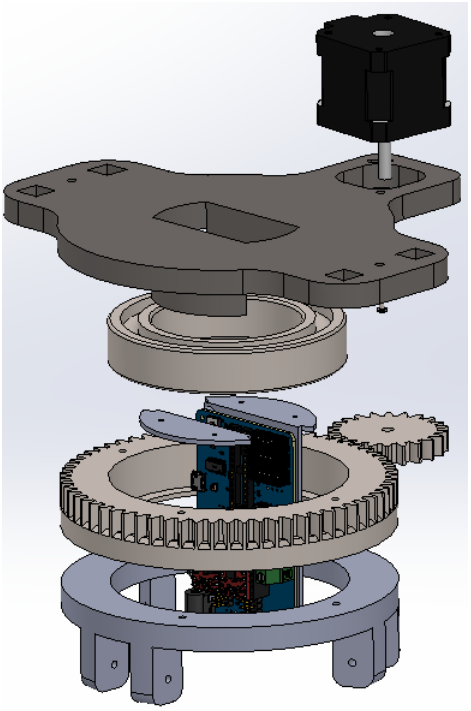


Figura 37: Explosión componentes movimiento yaw.

Base movimiento pitch y motor del eje yaw

En está pieza se alojan el mecanismo que genera el movimiento de pitch y el motor del eje yaw, se puede ver en la Figura 38 donde se alojan cada elemento, a demás, en el centro de la pieza se encuentran dos salientes con un agujero, lo que permite que un rodamiento pueda encajar dejando un hueco para alojar la electrónica.

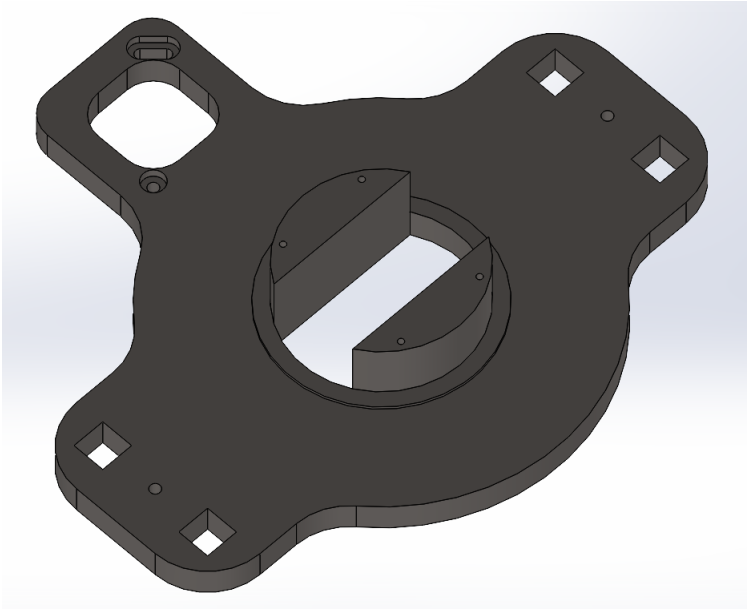


Figura 38: Base movimiento picth y motor del eje yaw.

Engranajes

Los engranajes para transmitir el movimiento en el eje yaw están dispuestos de tal forma que el engranaje conducido es fijo, lo que produce que cada movimiento del engranaje conductor se genere un movimiento angular en la base.

Para el diseño de los engranajes de este eje se han tenido las siguientes consideraciones: se emplea un rodamiento tipo 6012-2RS, con las siguientes dimensiones, 60mm diámetro interior, 95mm diámetro exterior y 18mm espesor, la distancia entre los ejes es de 87mm, se emplea modulo 2.

Teniendo en cuenta lo anterior se han escogido un engranaje conductor de 22 dientes y un engranaje conducido de 65 dientes.

El engranaje conductor debe alojar en su interior el rodamiento, en la Figura 39 se han marcado las zonas donde apoya el rodamiento entre la base, detalle A y el engranaje, detalle B. Así se puede apreciar como el engranaje conductor se sujeta por la parte exterior del rodamiento mientras que la parte interior alberga la base que mueve todo el conjunto

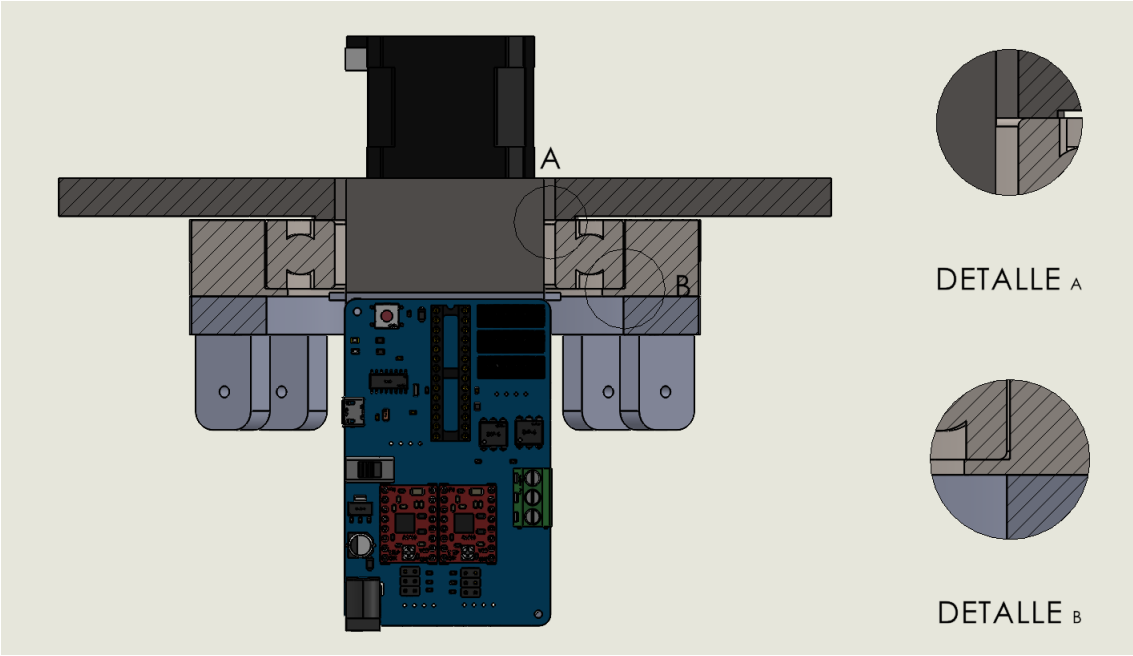


Figura 39: Detalles apoyo rodamiento eje yaw.

Topes del rodamiento y soporte de la PCB

Se han añadido dos topes que unen la parte interior del rodamiento con la base que rota, de tal forma que para realizar cualquier manipulación al extraer la parte superior, el rodamiento se separe del emplazamiento del engranaje conducido, quedando fila a la base. Además aprovechando el lugar se ha diseñado un soporte para la PCB. La PCB irá atornillada a este soporte, Figura 40.

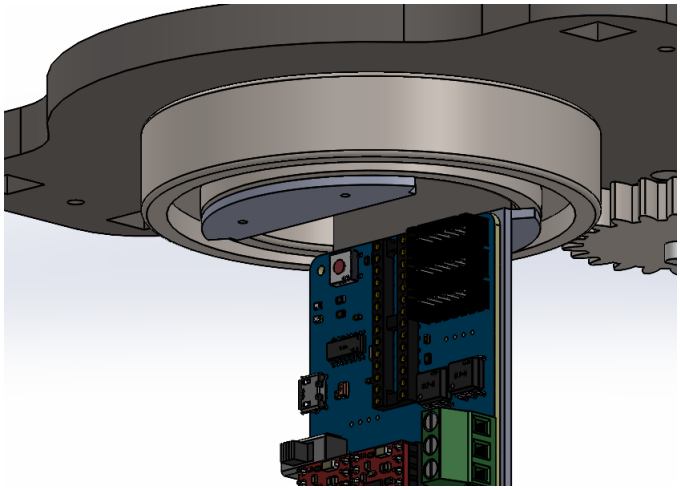


Figura 40: Topes del rodamiento y soporte de la PCB.

En la Tabla 12, se encuentran todas las piezas que se han diseñado, incluyendo la

sujeción del encoder al motor la cual se ha diseñado como si fuese un capuchón que se inserta sobre la parte trasera del motor paso a paso. Todos las piezas se encuentran detalladas en el documento Planos.

NOMBRE	Nº PLANO	Cantidad
Soporte cámara	TFM-P001	1
Rodamiento 608ZZ	-	2
Soporte lateral izquierdo	TFM-P002	1
Pasador	-	2
Base	TFM-P003	1
Rodamiento 6012-2RS	TFM-P004	1
Tope rodamiento	TFM-P005	1
PCB	-	1
Engranaje conducido_H	TFM-P006	1
Sujeción patas	TFM-P007	1
Engranaje conductor_H	TFM-P008	1
Soporte PCB	TFM-P009	1
Tornillo M3x15mm	-	4
Motor nema 17	-	2
Engranaje conductor_V	TFM-P010	1
Engranaje conducido_V	TFM-P011	1
Soporte lateral derecho	TFM-P012	1
Soporte encoder	TFM-P013	2
Tapa encoder	TFM-P014	2

Tabla 12: Lista de componentes

4.4. Código

En esta sección se detallan las librerías y funciones empleadas para el funcionamiento del control de la montura motorizada. Se ha seguido el diagrama de flujo de la Figura 41 como diagrama conceptual de las acciones que debe realizar el control. El control que debe realizar el microcotrolador debe ser el control de los motores, para ello debe consultar el estado de los encoders y calcular el error y actuar en consecuencia para reducir el error. Se han planteado dos controles, el control de posición y el de velocidad, el control de posición se podrá aplicar a los dos ejes mientras que el de velocidad sólo al eje yaw, aplicable a la aplicación de contrarrestar la velocidad de la tierra. Por otro lado, cuando se dese realizar una fotografía se debe de activar las salidas correspondientes al FOCUS y SHUTTER de la cámara.

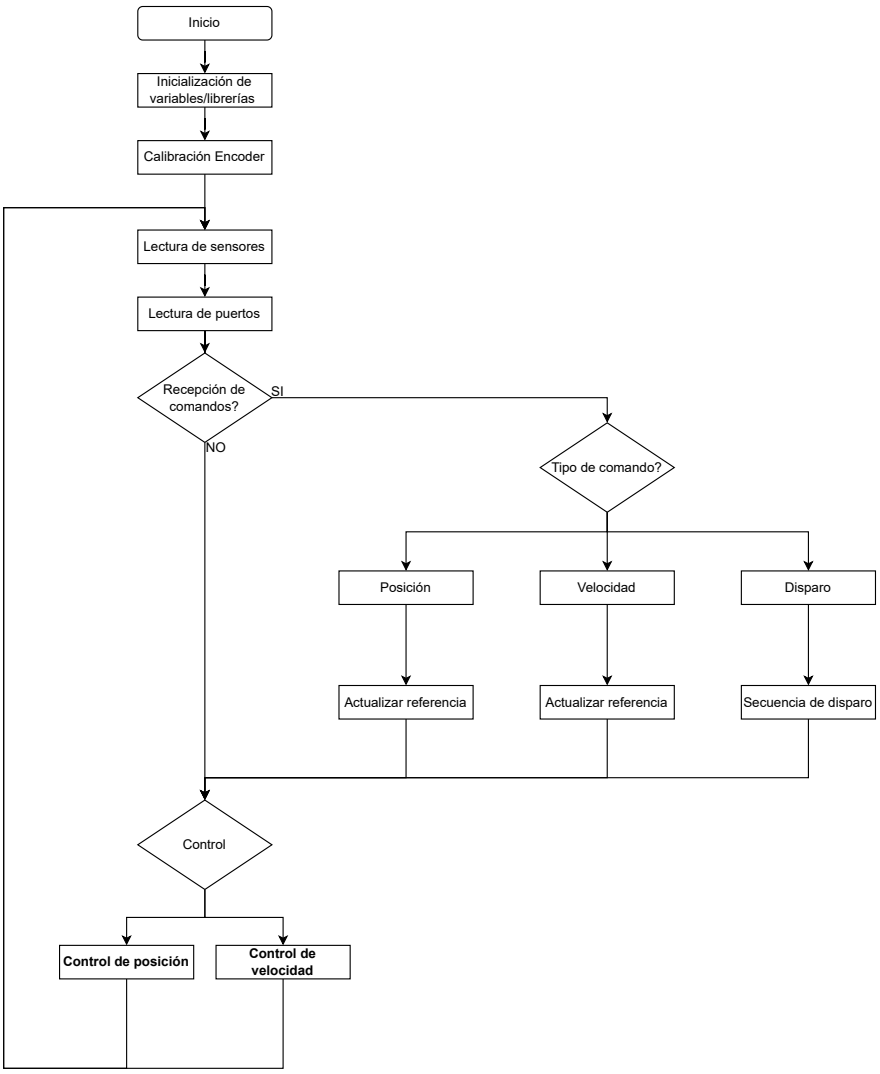


Figura 41: Diagrama de flujo del funcionamiento del dispositivo.

Seguidamente se detallan como se ha realizado el control de los motores paso a paso, librería y algoritmo de control, funciones para la obtención de la posición desde los encoders y comunicaciones.

4.4.1. Librería de control para el motor paso a paso

Trabajar con librerías en programación tiene múltiples ventajas. Las librerías permiten reutilizar código, mantener un código limpio y organizado. Al encapsular funcionalidades específicas en librerías, se puede abstraer la complejidad y mejorar la mantenibilidad del código.

En C++, las librerías suelen dividirse en dos tipos de archivos: archivos de cabecera (`.h`) y archivos de implementación (`.cpp`). Los archivos de cabecera (`.h`) contienen las declaraciones de funciones, clases y variables globales, mientras que los archivos de implementación (`.cpp`) contienen las definiciones y la lógica de las funciones y clases declaradas en los archivos de cabecera.

Los motores paso a paso se controlan mediante pulsos y una señal de sentido. Cada pulso enviado al pin de paso (step) hace que el motor avance un paso. La dirección del movimiento se controla mediante un pin de dirección (dir). Cambiar el estado de este pin determina si el motor gira en el sentido de las agujas del reloj o en sentido contrario.

La librería `ControlST` presentada a continuación permite controlar la velocidad de giro de los motores paso a paso, ajustando la frecuencia de los pulsos enviados al motor.

Definición de la Librería

El primer paso es definir la librería mediante un archivo de cabecera (`ControlST.h`) que declare las funciones y las variables necesarias:

```
#ifndef ControlST_h
#define ControlST_h

#include "Arduino.h"

class ControlST {
public:
    ControlST(uint8_t pinStep, uint8_t pinDir);
    void velocidad(float speed); // Establece la velocidad en pasos por segundo
    void marcha(); // Debe llamarse con frecuencia para actualizar la posición del motor

private:
    uint8_t _pinStep, _pinDir;
    unsigned long _lastStepTime;
    long _stepInterval; // Intervalo entre pasos en microsegundos, calculado a partir de la velocidad
    static const int _maxSpeed = 1500; // Máxima velocidad permitida,
};

#endif
```

El archivo `ControlST.h` define la clase `ControlST` que incluye las funciones para establecer la velocidad del motor (`velocidad`) y actualizar su posición (`marcha`). También declara las variables privadas necesarias para el control del motor.

Implementación de la Librería

El siguiente paso es implementar las funciones declaradas en el archivo de cabecera en un archivo de código `ControlST.cpp`:

```
#include "Arduino.h"
#include "ControlST.h"

ControlST::ControlST(uint8_t pinStep, uint8_t pinDir) : _pinStep(
    pinStep), _pinDir(pinDir), _lastStepTime(0) {
    pinMode(_pinStep, OUTPUT);
    pinMode(_pinDir, OUTPUT);
}

void ControlST::velocidad(float speed) {
    speed = constrain(speed, -_maxSpeed, _maxSpeed);
    if (speed == 0.0) {
        _stepInterval = 0;
    } else {
        _stepInterval = abs(1000000.0 / speed);
        digitalWrite(_pinDir, speed > 0 ? HIGH : LOW);
    }
}

void ControlST::marcha() {
    if (_stepInterval == 0) return;

    unsigned long time = micros();
    if (time - _lastStepTime >= _stepInterval) {

        digitalWrite(_pinStep, HIGH);
        delayMicroseconds(50);  // Ancho de pulso
        digitalWrite(_pinStep, LOW);

        _lastStepTime = time;
    }
}
```

En el archivo `ControlST.cpp`, se implementan las funciones declaradas. La función `velocidad` ajusta la velocidad del motor en pasos por segundo y la dirección del movimiento, mientras que la función `marcha` se llama periódicamente para mover el motor en intervalos de tiempo calculados.

Uso de la Librería

Finalmente, se muestra un ejemplo de uso de la librería en un archivo de programa principal (`main.ino`):

```
#include "ControlST.h"

// Definición de los pines para los motores
const uint8_t motor1PinStep = 2; // Pin de paso del motor 1
```

```
const uint8_t motor1PinDir = 3;  // Pin de dirección del motor 1
const uint8_t motor2PinStep = 4; // Pin de paso del motor 2
const uint8_t motor2PinDir = 5;  // Pin de dirección del motor 2

// Creación de objetos motor
ControlST motor1(motor1PinStep, motor1PinDir);
ControlST motor2(motor2PinStep, motor2PinDir);

void setup() {
  // Configuración inicial
  Serial.begin(9600); // Inicia la comunicación serial

  // Establecer la velocidad de los motores en pulsos por segundo
  motor1.velocidad(1000); // 1000 pulsos por segundo para el motor 1
  motor2.velocidad(500);  // 500 pulsos por segundo para el motor 2
}

void loop() {
  // Hacer que los motores giren
  motor1.marcha();
  motor2.marcha();
}
```

En el archivo principal (`main.ino`), se crean dos objetos `ControlST` para controlar dos motores paso a paso. En la función `setup`, se inicializan los motores y se establece su velocidad. La función `loop` llama a `marcha` repetidamente para mover los motores según la velocidad configurada.

Explicación del Código

El archivo `ControlST.h` define la clase `ControlST` que incluye las funciones para establecer la velocidad del motor (`velocidad`) y actualizar su posición (`marcha`). En el archivo `ControlST.cpp`, se implementan estas funciones. La función `velocidad` ajusta la velocidad del motor en pasos por segundo, mientras que la función `marcha` se llama periódicamente para mover el motor.

En el archivo principal (`main.ino`), se crean dos objetos `ControlST` para controlar dos motores paso a paso. En la función `setup`, se inicializan los motores y se establece su velocidad. La función `loop` llama a `marcha` repetidamente para mover los motores según la velocidad configurada.

Esta librería proporciona una manera sencilla de controlar motores paso a paso en aplicaciones donde se requiere de un control de velocidad.

En la Figura 42 se observa una representación de los pulsos generados. Para obtener dicha representación se ha empleado el analizador de señales digitales de 24 MHz. En esta representación se puede apreciar que la frecuencia es ligeramente inferior a la

que se pretende generar debido al coste en tiempo de cada función de código.

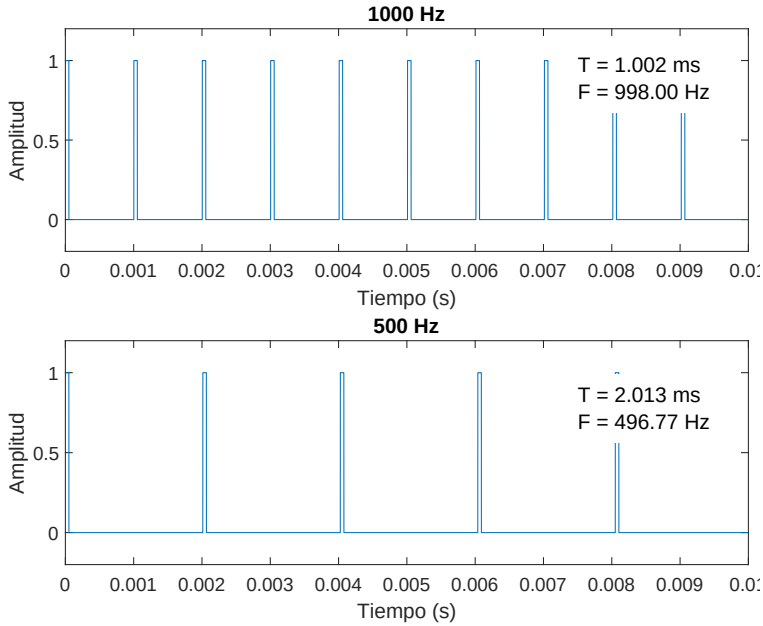


Figura 42: Pulsos generados por el microcontrolador.

Lógica de control

Una vez visto como se generan los pulsos para controlar el driver del motor paso a paso, es necesario de una lógica para calcular la velocidad de los pulsos por segundo y determinar cuando es necesario detenerse. La lógica empleada se encuentra en la Figura 43, donde lo primero es necesario conocer la posición actual de la salida del sistema, posteriormente se calcula el error entre la posición actual y la referencia, si dicho error se encuentra dentro de un umbral no se realiza acción debido a que el error es muy pequeño, si por el contrario el error es mayor al umbral se detecta el sentido de la acción y, se calcula la velocidad multiplicando el error por una ganancia y final mente se generan los pulsos.

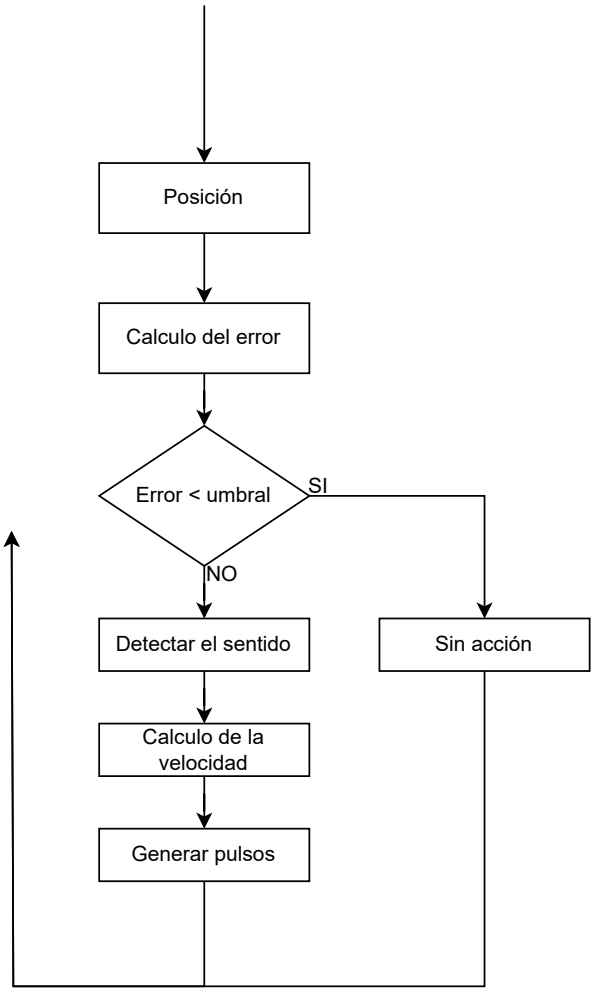


Figura 43: Algoritmo de control de posición

Se ha añadido una velocidad mínima de 200 pulsos por segundo para alcanzar la referencia, debido a que sin esta velocidad mínima el tiempo que toma en alcanzar la referencia por medio del regulador proporcional es muy elevado. A continuación, se presenta la lógica de control implementada en el código:

```
Error1 = Posicion1 - posicionAbsoluta1;
if (control == 1){
  if (abs(Error1) < 1) {
   Upk1 = 0; // Acción proporcional
  } else {
   Upk1 = (Error1 * KP1) * (-1); // Acción proporcional
```

```
7       if (Upk1 > 0) {
8         if (Upk1 < 200) {
9           Upk1 = 200; // Asegurar que Upk1 no sea menor a 100 en
      positivo
10        }
11      }
12      if (Upk1 < 0) {
13        if (Upk1 > -200) {
14          Upk1 = -200; // Asegurar que Upk1 no sea menor a 100 en
      negativo
15        }
16      }
17     }
18   }
19
20
21   motor1.velocidad(Upk1);
22   motor1.marcha();
```

El control de velocidad se realiza en bucle abierto, debido a que el driver con el que se está trabajando no permite variar la tensión o corriente para aumentar el par de salida en el eje. Esto hace que la única manera para mantener una velocidad se pueda realizar por medio de la relación entre la frecuencia de los pulsos y la configuración de microstepping. Para evitar que al salir del control de velocidad el dispositivo vuelva a la posición de inicio debido a no haber actualizado la variable de posición y referencia, se actualizan estas dos variables al mismo valor de posición medido a cada vuelta de ciclo mientras se efectué el control de velocidad.

4.4.2. Función obtención posición encoder

Para realizar medidas de posición del encoder se ha empleado la librearía `AS5048A.h`[10], esta sencilla librería ofrece las funcionalidades necesarias para inicializar, obtener la posición y realizar un manejo de errores por comunicación SPI.
A continuación se muestra un ejemplo de como obtener la posición. Como se está empleando comunicación SPI es necsario indicar el pin de selección del sensor, en este ejemplo se declara el pin 10 en la linea 5 y se usa como argumento en la declaración de el objeto `angleSensor`, posteriormente se inicializan las comunicaciones y en el bucle sin fin se muestran le valor de posición.

```
1  #include <SPI.h>
2  #include <AS5048A.h>
3
4  // Define the CS pin
5  const int csPin = 10;
6
7  // Create an instance of the AS5048A class
8  AS5048A encoder(csPin);
9
10 void setup() {
```

```
  // Initialize serial communication
  Serial.begin(9600);

  // Initialize the encoder
  encoder.init();
}

void loop() {
  // Read the raw angle
  uint16_t angle = encoder.getRawRotation();

  // Print the angle to the serial monitor
  Serial.print("Angle: ");
  Serial.println(angle);

  // Wait for a short period
  delay(500);
}
```

Este código sólo proporciona el valor de posición en valores de 0 a 16383, para pasar de este valor a grados se ha creado la siguiente función, `getPosicionAbsoluta`, esta función tiene como argumentos el sensor que se dese consultar, orden para asignar el cero y las variables necesarias para el cálculo de la posición como el valor de las vueltas completadas y offsets.

Lo primero que se realiza es la lectura del sensor, valor de 0 a 16383, después se calcula la diferencia de la posición entre el medido en la actual vuelta de ciclo y la anterior, por medio de esta diferencia se puede determinar si se ha completado una vuelta y el sentido, ejemplo: si se completa una vuelta completa el valor en la vuelta de ciclo actual será próximo a 0, mientras que en la vuelta de ciclo anterior será un valor cercano al máximo del sensor, 16383, haciendo la diferencia como resultado se obtendrá un valor negativo, clasificando la diferencia que se produce al cruzar del limite máximo al mínimo según su signo se determina las vueltas completadas y el sentido.
El valor de la posición se determina como el valor en grados de las vueltas completadas mas la posición actual que se pasa a grados por medio de la función `map`, esta función solo trabaja con variables de tipo `int` esto quiere decir que si se desea optener un decimal de precisión se puede multiplicar los valores de salida de escalado por 10 y dividir el resultado por 10.

Para resetear la posición se asigna el valor de cero a la variable de `vueltasCompletas` y se calcula un offset, que será la posición actual, la cual se restará a las medidas que se tomen a continuación.

```
float getPosicionAbsoluta(AS5048A &encoder, bool resetear, long &
    vueltasCompletas, long &offset, word &valorAnterior) {
  word valorActual = encoder.getRawRotation();

  if (resetear) {
```

```
    vueltasCompletas = 0;
    offset = map(valorActual, 0, 16383, 0, 3600) / 10; // Ajustar el offset al valor actual del encoder
    valorAnterior = valorActual;
    return 0.0;
  }

  int diferencia = int(valorActual) - int(valorAnterior);

  if (diferencia > 8192) {
    vueltasCompletas -= 1;
  } else if (diferencia < -8192) {
    vueltasCompletas += 1;
  }

  float posicionAbsoluta = 360.0 * vueltasCompletas + (map(valorActual, 0, 16383, 0, 3600) / 10.0) - offset;
  valorAnterior = valorActual;
  return posicionAbsoluta;
}
```

4.4.3. Función lectura de comandos

Para interpretar los comandos recibidos primeros es necesario reconstruir la información recibida. A continuación se observa el fragmento de código empleado para reconstruir la información recibida por el puerto serie, en este código siempre que se realice la recepción de datos por el puerto serie el carácter se asignan a la variable `c` y ésta se concatena carácter a carácter en la variable `Serialdata`, cuando se recibe el salto de línea (`\n`) indica que se ha finalizado la recepción de información por el puerto serie y se procede a interpretar el comando, para interpretar el comando se ha creado la función `processCommand()`.

```
  while (Serial.available() > 0) {
    char c = Serial.read();
    Serialdata += c;
    if (c == '\n') {
      Serialdata.trim(); // Recorta los espacios en blanco
      processCommand(Serialdata); // Procesar comandos recibidos desde el monitor serie
      Serialdata = ""; // Limpiar el buffer después de procesar el comando
    }
  }
```

La función `processCommand` se encarga de ejecutar los comandos de la cadena de caracteres montada, realizando comparaciones mediante bucles "if", a continuación se encuentra un ejemplo de como se realiza la ejecución de los comandos:

```
void processCommand(String command) {
  command.trim(); // Eliminar espacios en blanco al principio y al final
```

```
  if (command == "S") {
    digitalWrite(FOCUS, HIGH); // Activar salida digital
    delay(500);
    digitalWrite(SHUTTER, HIGH); // Activar salida digital
    delay(200);
    digitalWrite(SHUTTER, LOW);  // Desactivar salida SHUTTER
    digitalWrite(FOCUS, LOW);
  } else if (command.startsWith("V ")) {
    Velocidad = command.substring(2).toFloat();
    Serial.print("Velocidad: ");
    Serial.println(Velocidad);
    if (Velocidad != 0){
      control = 0;

  } else {
    int commaIndex = command.indexOf(',',); // Comando de posicion 0,0
   posición en grados eje pitch, eje yaw
    if (commaIndex != -1) {
      String pos1Str = command.substring(0, commaIndex); // extraer el
    valor en grados del eje pitch
      String pos2Str = command.substring(commaIndex + 1);// extraer el
    valor en grados del eje yaw
      Posicion1 = pos1Str.toFloat();
      Posicion2 = pos2Str.toFloat();
      Serial.print("Posicion1 actualizada a: ");
      Serial.println(Posicion1);
      Serial.print("Posicion2 actualizada a: ");
      Serial.println(Posicion2);
    } else {
      Serial.println("Comando no reconocido");
    }
  }
}
```

4.4.4. Envió de comandos desde un terminal Bluetooth

Para enviar y recibir datos por Bluetooth se ha empleado el HM-10, un dispositivo Bluetooth que se comunica por medio de comunicación serial. De este modo se puede enviar comandos desde un dispositivo móvil, para ello es necesario instalar una aplicación que realice la función de terminal, en este caso se ha probado la aplicación dabble, la cual es un terminal que ofrece distintas interfaces. A continuación se muestran la aplicación Dabble, Figura 44:

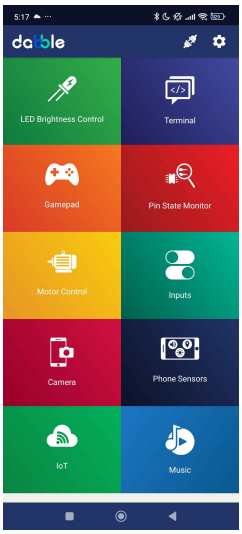


Figura 44: Aplicación Dabble.

Las interfaces que se han utilizado para el envió y recepción de datos son la de gamepad y terminal, que se muestran en las figuras 45 y 46. Desde la pantalla del terminal se puede enviar cualquier comando, muentras que desde la pantalla de gamepad se utiliza los botones de las flechas como si fuera un modo jog, mientras que el botón del circulo establece un cero en el sistema de referencia y el cuadrado realiza el disparo de la cámara.

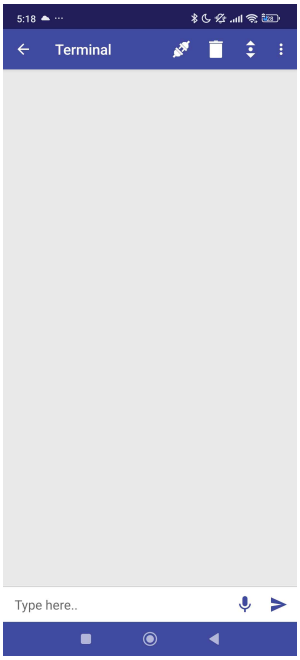


Figura 45: Dabble interfaz terminal.

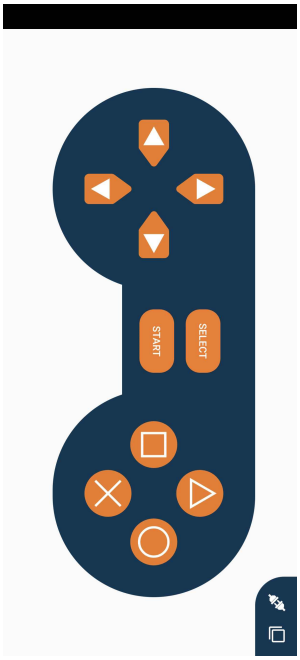


Figura 46: Dabble interfaz gamepad.

En la Tabla 13, se encuentran todos los comandos que se pueden enviar por el puerto

serie.

Comando	Descripción
S	Activa la salida digital para FOCUS y SHUTTER durante un breve periodo.
A	Resetea la posición absoluta de ambos encoders y actualiza las posiciones objetivo de los motores.
V +*velocidad*	Establece la velocidad manual de los motores. Si la velocidad es diferente de cero, el control de velocidad se activa. Si la velocidad es cero, el control de posición se reactiva. Ejemplo: `V 100`.
T +*tiempo,repeticiones*	Activa las salidas digitales FOCUS y SHUTTER durante un *tiempo* en segundos, repitiendo el ciclo el número de *repeticiones* especificadas. Ejemplo: `T 2,5`.
KP1 +*valor*	Actualiza la ganancia proporcional (KP1) del motor 1. Ejemplo: `KP1 50`.
KP2 +*valor*	Actualiza la ganancia proporcional (KP2) del motor 2. Ejemplo: `KP2 75`.
posición1,posición2	Actualiza las posiciones objetivo de los motores 1 y 2. Las posiciones se separan por una coma. Ejemplo: `10.5,20.3`.

Tabla 13: Lista de comandos y sus descripciones

4.4.5. Funciones desde Matlab

En este apartado se explican las funciones creadas desde Matlab para generar y enviar comandos a la montura

Generar vector de coordenadas trayectoria lineal

La función `trayectoria_recta` crea un vector de coordenadas que une la coordenada inicial con la coordenada final en un números de coordenadas intermedias, para ello se emplea la función `linspace` para crear los valores intermedios según cuantas coordenadas se desee, finalmente se crea el vector que reúne las dos componentes de cada coordenada. A continuación se ve el código de Matlab que realiza lo descrito anteriormente:

```
function coords = trayectoria_recta(lim1_start, lim1_end, lim2_start,
    lim2_end, num_divisiones)
    % lim1_start: inicio de la primera coordenada
    % lim1_end: fin de la primera coordenada
    % lim2_start: inicio de la segunda coordenada
    % lim2_end: fin de la segunda coordenada
    % num_divisiones: número de divisiones
    coords = {};
```

```
8
9       % Generar los vectores de divisiones
10      coord1 = linspace(lim1_start, lim1_end, num_divisiones);
11      coord2 = linspace(lim2_start, lim2_end, num_divisiones);
12
13      % Generar las coordenadas
14      for i = 1:(num_divisiones)
15          coords{end+1} = sprintf('%.2f,%.2f', coord1(i), coord2(i));
16      end
17  end
```

Generar vector de coordenadas recorrer una matriz

La función `trayectoria_panoramica` crea un vector de coordenadas que recorre las posiciones de una matriz de dimensiones mxn, como el vector de coordenadas contiene las posiciones de una matriz, el incremento de una posición a otra contigua varía en una unidad una sus sus coordenadas, conociendo esto, multiplicando esta unidad por un valor en grados se transforman las posiciones de una matriz de coordenadas a grados. Estas coordenadas necesitan ser ordenadas para que al recorrer el vector solo pueda haber un cambio de valor de una de sus componentes. A continuación se ve el código de Matlab que realiza lo descrito anteriormente:

```
1  function coords = trayectoria_panoramica(m, n, factor)
2      % m: número de filas
3      % n: número de columnas
4      % factor: número por el cual se multiplican las coordenadas
5      coords = {};
6
7      % Variable para controlar la dirección
8      direction = 1; % 1 para izquierda a derecha, -1 para derecha a
       izquierda
9
10     for i = 1:m
11         for j = 1:n
12             if direction == 1
13                 % Coordenadas multiplicadas por el factor
14                 coords{end+1} = sprintf('%d,%d', ifactor, jfactor);
15             else
16                 coords{end+1} = sprintf('%d,%d', ifactor, (n-j+1)
       factor);
17             end
18         end
19         % Cambiar la dirección
20         direction = -direction;
21     end
22 end
```

Envió de comandos desde Matlab

Para enviar los vectores de coordenadas desde MATLAB, el siguiente código configura y gestiona la comunicación con un dispositivo a través de un puerto serie. Primero, cierra y elimina cualquier conexión previa para evitar conflictos. Luego, muestra los puertos seriales disponibles y configura el puerto seleccionado con una tasa de baudios específica. Se crea un objeto de puerto serie y se define una función callback para manejar los datos recibidos. Después, inicializa variables globales y abre el puerto serie, enviando el primer comando. Un bucle infinito mantiene el script en ejecución.

La función callback lee los datos recibidos y muestra el contenido. El código recorre el vector comandos y no envía el siguiente comando hasta que se ha recibido una "R" y se ha enviado una "S" como respuesta, asegurando una comunicación bidireccional continua con el dispositivo. Esto garantiza que cada comando se procese en el orden correcto y que el dispositivo responda adecuadamente antes de recibir el siguiente comando.

```
% Verificar y cerrar puertos abiertos previamente
if ~isempty(instrfind)
    fclose(instrfind);
    delete(instrfind);
end

% Mostrar puertos disponibles
serialInfo = instrhwinfo('serial');
disp('Puertos seriales disponibles:');
disp(serialInfo.AvailableSerialPorts);

% Configuración del puerto serie
puerto = 'COM5'; % Cambia esto por un puerto disponible de la lista
baudRate = 9600; % Configura la tasa de baudios adecuada para tu
   dispositivo

% Crear el objeto de puerto serie
s = serial(puerto, 'BaudRate', baudRate, 'Terminator', 'LF');

% Configurar el callback para la recepción de datos
s.BytesAvailableFcnMode = 'terminator'; % Ejecutar callback al recibir
    el terminador
s.BytesAvailableFcn = @serialCallback;  % Especificar la función
   callback

% Inicializar variables globales
global comandos i s
%comandos = {'180,90','90,0','260,45'};
i = 1; % Inicializar el índice en 1

% Abrir el puerto serie
try
    fopen(s);
    disp('El puerto serie está abierto.');
```

```
catch
    disp(['No se pudo abrir el puerto serie ', puerto]);
    delete(s);
    return;
end

% Enviar el primer comando
fprintf(s, comandos{i});
disp(['Comando enviado: ', comandos{i}]);

% Bucle infinito para mantener el script en ejecución
while true

end

% Función callback para manejar los datos recibidos
function serialCallback(obj, ~)
    global comandos i s
    data = fscanf(obj); % Leer los datos disponibles
    disp(['Datos recibidos: ', data]);

    % Enviar 'S' si se recibe 'R'
    if contains(data, 'R')
        fprintf(obj, 'S');
        disp('Comando enviado: S');
        pause(1);

        % Verificar el valor de i y la longitud de comandos
        if i < length(comandos)
            i = i + 1;
            fprintf(obj, comandos{i});
            disp(['Comando enviado: ', comandos{i}]);
        else
            disp('Todos los comandos han sido enviados.');
        end
    end
end
```

5. Pruebas y resultados

5.1. Comprobaciones

Lo primero que se realiza tras la recepción de la PCB fue verificar su estado, para ello se inspecciona la placa en busca de defectos o daños que se puedan dar por el transporte, como arañazos. Después se comprueba la continuidad entre las diferentes pistas, asegurando que las conexiones son las mismas que las de el circuito diseñado y que conectan de la manera adecuada los distintos elementos y, que no hay presencia de cortocircuitos que puedan generar un problema.

Después de comprobar el estado de la PCB se ha cargado una fragemto de código donde se testean las lecturas de posición desde el encoder y se acciona un motor paso a paso. En esta primera prueba se trata de probar la lógica del regulador implementado, verificar que acciones de control positivas corresponden a salidas del sistema positivas y cuál es la capacidad para alcanzar una velocidad de giro.

Antes de conectar los motores paso a paso, se ha ajustado la corriente que circulará por los motores, para realizar el ajuste el driver tiene un pequeño potenciómetro con el que se puede ajustar la corriente. Siguiendo las indicaciones de la documentación técnica [11], entre el potenciómetro y tierra debe haber el doble de tensión que corriente se desea, se ha ajustado hasta tener un valor entre estos dos puntos de 2V.

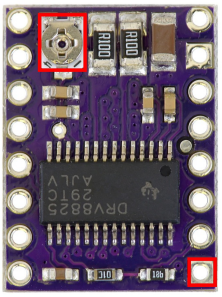


Figura 47: Ajuste de corriente DVR8825.

En la Figura 48, se ve el comportamiento obtenido, donde se ha escogido una velocidad máxima de 1000 pasos por segundo, el primer escalón se realiza a 5000 grados, del eje del motor, lo que es aproximadamente 14 vueltas que se realizan en 2,8 segundos. En esta figura se ve como debido a la ganancia escogida la acción de control (velocidad), enseguida satura y se observa como la posición se incrementa de manera lineal.

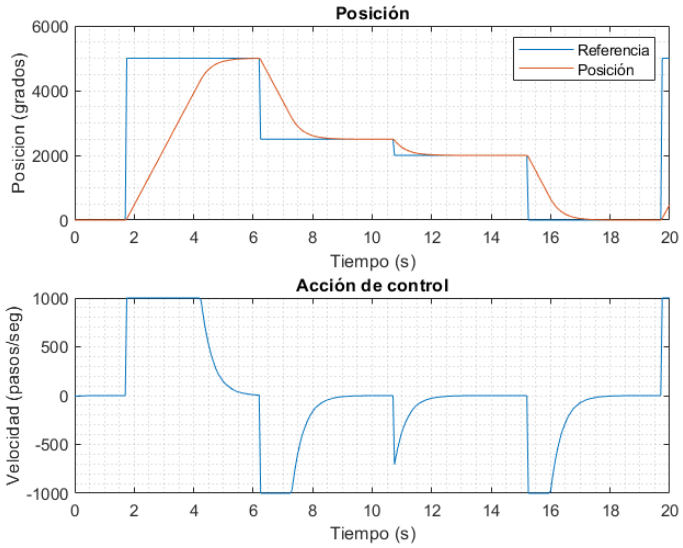


Figura 48: Ensayo control de posición.

Después de este ensayo se ha cargado el código completo, incluyendo las comunicaciones por puerto serie, para medir la vuelta de ciclo. Para ello se ha añadido en el código que el pin 8 alterna su estado a cada vuelta de ciclo, midiéndolo con el analizador de señales. El resultado es una onda cuadrada de 0,94ms de periodo (1,06kHz), tiempo en en que se realizan las mediciones de los dos encoders, se verifica si hay algún comando. Cuando se está ejecutando algún movimiento el tiempo en el que se completa una vuelta de ciclo aumenta, hasta los 1,14ms (870Hz). Estos valores son mas que suficientes para obtener movimientos rápidos en la cámara, sí se escoge el valor mas bajo, de 870Hz, y se multiplica por el paso máximo, de 1,8º, se obtiene un valor de 1566º/s, lo que se traduce a una velocidad de salida de 4,35 revoluciones por segundo.

Tras verificar que la PCB funciona correctamente, se realiza el ensamblaje de todos los elementos, como se puede ver en la Figura 49, donde se han añadido tres perfiles de aluminio para realizar la función de patas.

Figura 49: Dispositivo ensamblado.

Con el ensamblaje realizado se procede a comprobar que todos los elementos móviles, encajan y permiten el moviente de manera adecuada sin colisionar con otros elementos. En la Figura 50, se puede ver como el conector de tipo jack que permite el control del disparo de la cámara no colisionará cuando éste se oriente hacia arriba.

Figura 50: Comprobación de movimientos del sistema

5.2. Panorámicas

Una de las posibilidades de contar con un control de la posición para una cámara es la de crear panorámicas, para ello sólo es necesario realizar un conjunto de fotografías donde entre fotografías contiguas exista un solape de información, esto puede ser una tarea sencilla cuando el número de fotografías que componen la panorámica es reducido pero cuando el número aumenta, debe realizarse de manera estructurada. Esto es una tarea sencilla para el dispositivo donde se han realizado 21 fotografías con incrementos de 7º de una a otra, este reducido valor entre fotografías hace que exista mucha información solapada facilitando la reconstrucción de la imagen por el software *HUGIN*. A continuación, en la Figura 51, se ven todas las fotografías realizadas.

Figura 51: Imágenes capturadas para panorámica

En el programa *HUGIN*, se debe de cargar las imágenes y realizar la operación de alinear, esta operación establece las relaciones de posición entre fotografías ordenándolas, de no encontrar de manera automática estas relaciones se deberán de realizar de manera manual, como todas las fotografías se han realizado de forma estructurada el software es capaz de establecer todas las relaciones, como resultado se obtiene la salida de la Figura 52.

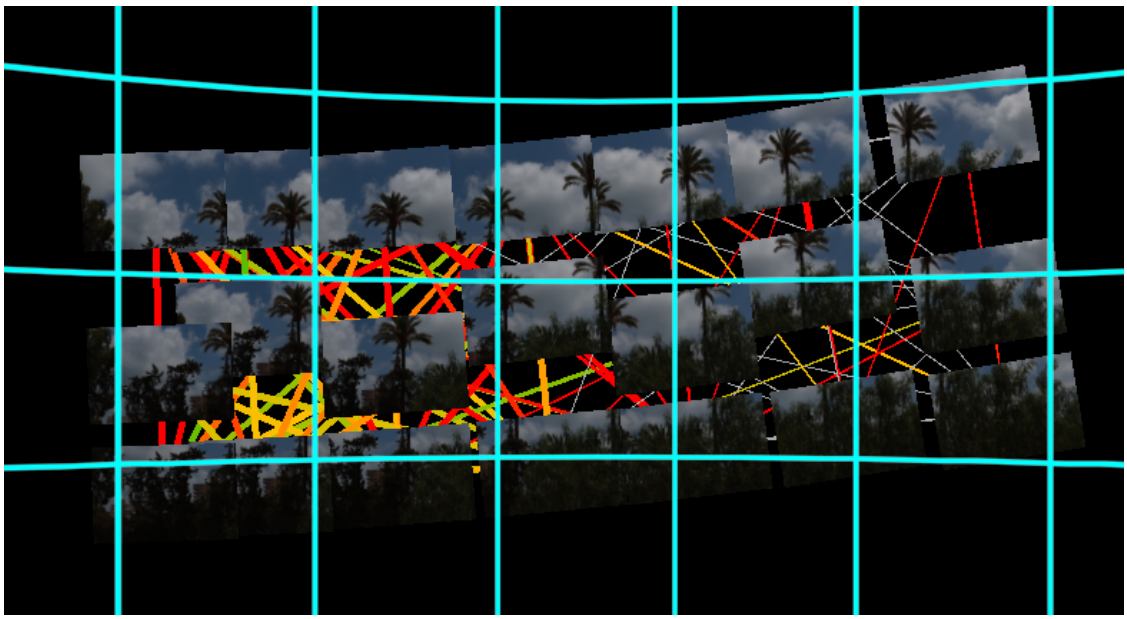


Figura 52: Programa Hugin identificación de relaciones

Como paso final, se realiza la operación de ensamblado de las imágenes, obteniendo el resultado de la Figura 53.

Figura 53: Resultado panorámica

5.3. Astrofotografía

La montura motoriza diseñada permite realizar un seguimiento de cuerpos celestes igualando la velocidad de rotación del eje yaw con la velocidad de rotación de la tierra, de esta forma el eje pitch se encarga en apuntar a la región del cielo a observar. Para conseguir esto primero es necesario es necesario orientar el eje de rotación yaw con el eje de rotación de la tierra, para ello solo se necesita inclinar el eje jaw un valor equivalente al de la latitud de la zona en la se ubique y orientarlo hacia el norte, en la Figura 54 se ve como se han efectuado las fotografías.

Figura 54: Inclinación dispositivo astrofotografía.

Antes de comenzar la sesión de astrofotografía se ha tomado una imagen con un tiempo de exposición de 80 segundos para mostrar la necesidad de operar con una montura que compense la rotación de la tierra, en la Figura 55 se puede ver el desplazamiento de las estrellas producido durante los 80 segundos de exposición.

Figura 55: Fotografía sin compensación de la rotación.

Como primera secuencia de fotos, se ha orientado el eje yaw hacia el norte y con una inclinación de 38,34º, que corresponden a la latitud de Alicante, posteriormente se ha programado una exposición de 40 segundos, con lo que se ha obtenido la Figura 56. En la figura, se pueden apreciar con bastante nitidez los puntos de las estrellas aun cuando se ha tomando la foto en el interior de una ciudad como Alicante y tendiendo la Luna en su fase creciente.

Figura 56: Primera toma astrofotografía-A.

Debido a que los engranajes que conforman el eje yaw presentan pequeñas deformaciones e irregularidades, en muchas de las tomas realizadas salen movidas como en la Figura 57:

Figura 57: Primera toma astrofotografía-B.

Para dar solución a este problema se ha replanteado el sistema de transmisión del eje yaw y se ha sustituido por una correa, donde el engranaje conductor ahora es de 20 dientes y en el lado conducido, para mantener el diámetro del anterior engranaje se le ha dado 211 dientes, la correa es de tipo GT-2, siendo la separación entre los dientes de 2mm y contando con un alma de fibra de vidrio, esto garantiza que tendrá un buen comportamiento ante esfuerzos. Esta modificación aparece en el plano TFM-P014 dentro del documento de Planos

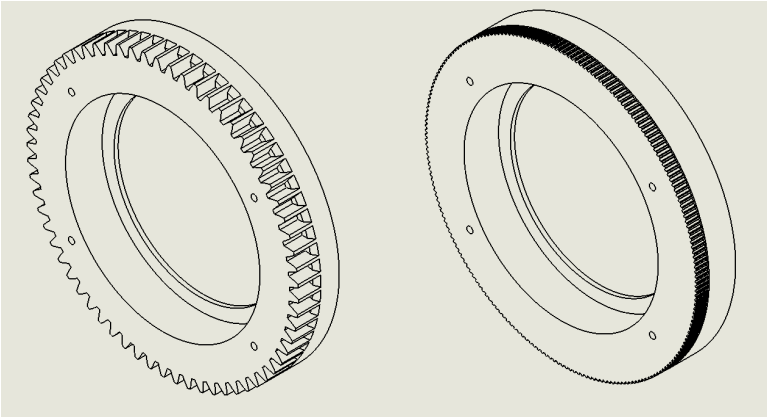


Figura 58: Modificación engranaje.

Tras realizar la modificación de engranaje y actualizar la nueva relación de transmisión en el código, se ha realizado una nueva toma nocturna. Con la modificación realizada se ha conseguido una exposición total de 20 minutos, en la Figura 59, se ve el resultado. Esta fotografía se ha realizado por medio de un proceso de apilado de imágenes, que consiste en realizar un grupo de fotografías de una zona y por medio de un software extraer la información de cada fotografía para conformar una nueva imagen con la información de todo el grupo de fotografías realizadas.
La Figura 59, se ha realizado tomando 30 tomas de 40 segundos de exposición, dejando entre un intervalo de 60 segundos entre tomas, este intervalo entre tomas es necesario para que la cámara integre toda la información adquirida durante los 40 segundos y crear el archivo .RAW.
Como consecuencia, para realizar la siguiente imagen se ha tenido que estar enfocando la misma región del espacio durante un mínimo de 50 minutos y gracias a la modificación realizada se ha podido lograr sin afectar al resultado.

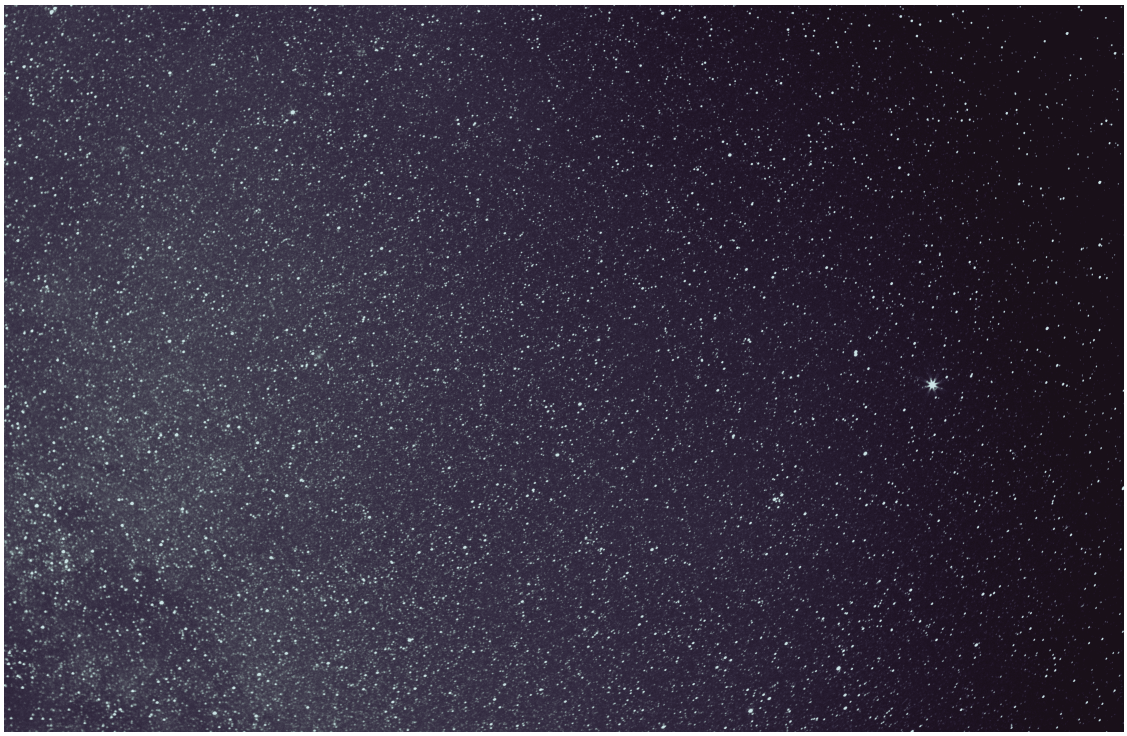


Figura 59: Segunda toma astrofotografía.

6. Compromiso ODS

Los Objetivos de Desarrollo Sostenible (ODS) son una serie de 17 objetivos establecidos por las Naciones Unidas en 2015 como parte de la Agenda 2030 para el Desarrollo Sostenible. Estos objetivos buscan abordar los desafíos globales más urgentes, como la pobreza, la desigualdad, el cambio climático, la degradación ambiental, la paz y la justicia. Los ODS son un llamado universal a la acción para acabar con la pobreza, proteger el planeta y mejorar las vidas y las perspectivas de todas las personas en todas partes.

Tabla de los ODS

ODS	Nombre	Descripción
1	Fin de la pobreza	Poner fin a la pobreza en todas sus formas en todo el mundo.
2	Hambre cero	Poner fin al hambre, lograr la seguridad alimentaria y la mejora de la nutrición, y promover la agricultura sostenible.
3	Salud y bienestar	Garantizar una vida sana y promover el bienestar para todos en todas las edades.

4	Educación de calidad	Garantizar una educación inclusiva, equitativa y de calidad, y promover oportunidades de aprendizaje durante toda la vida para todos.
5	Igualdad de género	Lograr la igualdad entre los géneros y empoderar a todas las mujeres y niñas.
6	Agua limpia y saneamiento	Garantizar la disponibilidad y la gestión sostenible del agua y el saneamiento para todos.
7	Energía asequible y no contaminante	Garantizar el acceso a una energía asequible, fiable, sostenible y moderna para todos.
8	Trabajo decente y crecimiento económico	Promover el crecimiento económico sostenido, inclusivo y sostenible, el empleo pleno y productivo, y el trabajo decente para todos.
9	Industria, innovación e infraestructura	Construir infraestructuras resilientes, promover la industrialización inclusiva y sostenible, y fomentar la innovación.
10	Reducción de las desigualdades	Reducir la desigualdad en y entre los países.
11	Ciudades y comunidades sostenibles	Lograr que las ciudades y los asentamientos humanos sean inclusivos, seguros, resilientes y sostenibles.
12	Producción y consumo responsables	Garantizar modalidades de consumo y producción sostenibles.
13	Acción por el clima	Adoptar medidas urgentes para combatir el cambio climático y sus efectos.
14	Vida submarina	Conservar y utilizar sosteniblemente los océanos, los mares y los recursos marinos para el desarrollo sostenible.
15	Vida de ecosistemas terrestres	Proteger, restablecer y promover el uso sostenible de los ecosistemas terrestres, gestionar sosteniblemente los bosques, luchar contra la desertificación, y detener e invertir la degradación de las tierras y detener la pérdida de biodiversidad.
16	Paz, justicia e instituciones sólidas	Promover sociedades pacíficas e inclusivas para el desarrollo sostenible, facilitar el acceso a la justicia para todos y crear instituciones eficaces, responsables e inclusivas a todos los niveles.
17	Alianzas para lograr los objetivos	Fortalecer los medios de implementación y revitalizar la Alianza Mundial para el Desarrollo Sostenible.

Relación del trabajo con los ODS

El dispositivo diseñado para orientar una cámara fotográfica que permite realizar panorámicas y astrofotografía tiene varias relaciones con los ODS, en particular:

- **ODS 4: Educación de calidad**: El proyecto de montura motorizada para fotografía puede ser una herramienta de bajo coste para el aprendizaje de los astros, en zonas donde el acceso a herramientas de observación astronómica es limitado, favoreciendo en consecuencia la igualdad de acceso a una formación profesional asequible, eliminar las disparidades de género y riqueza y lograr el acceso universal a una educación superior de calidad. Este objetivo alcanza su máxima necesidad en países del África subsahariana, en donde a las comunidades locales, con frecuencia, se les trasladan los principios de la educación occidental, desarraigándolas de sus costumbres y conocimientos locales, y en consecuencia, perpetuando el ciclo de dependencia cultura y económica. Sin embargo, estas culturas presentan vínculos ancestrales respecto a los ciclos astronómicos que deberían ser reforzados en herramientas de aprendizaje adaptadas a sus tradiciones y cultura propias. Por ello, la montura motorizada para la observación de astros, por su escaso coste puede ser susceptible de contribuir a este objetivo.

- **ODS 8: Trabajo decente y crecimiento económico**: El sistema, con cualquiera de sus distintas aplicaciones, puede ser susceptible de fabricación, y adiestramiento en su manejo, contribuyendo a la creación de puestos de trabajo productivos, con ingresos dignos, seguridad en el lugar de trabajo y, protección social para sus familias. El colectivo más beneficiado sería el de jóvenes, que es donde, por ejemplo, de España, mayor incidencia presentan los índices de desempleo, al estar más relacionados con las nuevas tecnologías.

- **ODS 9: Industria, innovación e infraestructura**: La creación de un dispositivo tecnológico innovador contribuye al desarrollo de infraestructuras resilientes y fomenta la industrialización inclusiva y sostenible. La innovación tecnológica es, en consecuencia, un motor clave para el desarrollo económico y el progreso social.

- **ODS 11: Ciudades y comunidades sostenibles**: La capacidad de capturar imágenes panorámicas y de astrofotografía puede contribuir a la documentación y monitoreo del entorno urbano y natural, facilitando la planificación urbana sostenible y la conservación de espacios naturales. Otro aspecto en relación a la astrofotografía es el relativo a la contaminación lumínica, el cual, en futuros desarrollos del proyecto, podría incorporar sensores para que, corrigiendo aspectos meteorológicos, pudiera valorar el grado de contaminación lumínica en las ciudades, contribuyendo a la toma de decisiones para su mitigación.

- **ODS 13: Acción por el clima**: Las aplicaciones en astrofotografía y el seguimiento del movimiento de las estrellas pueden ayudar a mejorar nuestra comprensión del cambio climático y sus efectos, proporcionando datos valiosos para

la investigación científica y la formulación de políticas ambientales. Otro aspecto a desarrollar es monitorizar eventos, tales como la subida del nivel del mar o la erosión costera, el retroceso de glaciares, las pérdidas de capacidad de los embalses por insolación o falta de lluvias, etc.

- **ODS 14: Vida submarina**: Son múltiples los efectos del cambio climático y la sobreexplotación de los recursos de mares y océanos, por lo que el sistema montura motorizada puede ser válido para su seguimiento y, posterior toma de medidas de gestión. Ya se ha mencionado en el ODS 13 , su potencial uso para monitorizar los cambios en el nivel del mar; y otros como la fotografía de cetáceos a su paso por determinadas zonas. A su vez, con la implementación de compartimentos estancos que permitan su uso submarino, puede ser una eficaz herramienta para monitorizar ciertos procesos en la ecología marina, como colonización de especies invasoras en entornos controlados, poblaciones de especies bentónicas de interés, tales como meros o langostas, o bien, en el control de las poblaciones de peces estabulados en acuicultura.

- **ODS 15: Vida de ecosistemas terrestres**: Del mismo modo que se ha expuesto en el ODS 14, también puede emplearse para monitorizar poblaciones de aves, grandes mamíferos en su paso por determinados enclaves, la evolución de enfermedades fitosanitarias en cultivos o bosques, por ejemplo, en este caso, la plaga del Tomicus en los pinares mediterráneos, etc.

- **ODS 17: Alianzas para lograr los objetivos**: El desarrollo y la implementación de este dispositivo pueden fomentar la colaboración entre diferentes sectores y disciplinas, promoviendo alianzas que son esenciales para alcanzar los ODS. La cooperación internacional en investigación y tecnología es fundamental para el progreso hacia un desarrollo sostenible.

El trabajo realizado no sólo aporta una innovación tecnológica significativa, sino que también apoya múltiples objetivos de desarrollo sostenible, contribuyendo al progreso global en áreas clave como la innovación, la sostenibilidad urbana, la acción climática y las alianzas estratégicas.

7. Conclusiones

El objetivo del trabajo era diseñar un sistema para orientar una cámara fotográfica, en el desarrollo de este trabajo se ha abarcado todos los aspectos del diseño, desde la idea, búsqueda de información, planteamiento, elección de componentes, diseño mecánico y electrónico, finalizando en las pruebas realizadas. Después de este desarrollo, y de cumplir con los requerimientos indicados, se puede decir que el se ha cumplido con los objetivos del proyecto.

Dicho lo anterior, se pueden sacar las siguiente conclusiones:

- Se ha diseñado un sistema compuesto por una parte mecánica, eléctrica y de control, que conforma un dispositivo que cumple con los requerimientos establecidos.
- Según los resultados obtenidos en los apartados de panorámica y astrofotografía, se puede afirmar que el comportamiento es el deseado.

El desarrollo del proyecto finaliza con el presente documento, aplicando en el desarrollo todas las competencias impartidas en el Máster en Ingeniería Mecatrónica, como lo son, el análisis y diseño de sistemas mecatrónica, la capacidad para integrar diferentes técnicas y tecnologías mecánicas, electrónicas, eléctricas, de control e informática enfocadas al desarrollo de un producto o investigación.

Recopilando, en este trabajo se ha realizado una evaluación de requerimientos, la selección de componentes que satisfagan los requerimientos, el diseño de una PCB enfocado a la conexión de los sensores y elementos a actuación, se ha realizado el diseño mecánico de una estructura firme que permite la orientación de la cámara en dos grados de libertad y se ha desarrollado el código para el funcionamiento del dispositivo.

7.1. Mejoras

Debido al tiempo limitado de desarrollo no se ha podido implementar mas funcionalidades al dispositivo, aun así se han conseguido cumplir con los requerimientos.

Sin embargo, como en cualquier proyecto, siempre hay aspectos que podrían mejorarse. Por ello, se propondrá una serie de posibles proyectos futuros que permitan aumentar las capacidades de estos dispositivos.

Primero, el tamaño del dispositivo puede reducirse por medio de un diseño mecánico mas compacto y con menor número de piezas, esto se traduce en piezas que cumplen mas de una función, es decir un diseño mas complejo lo que también es un diseño en su fabricación .

Segundo, en cuanto al control y configuración de los drivers de los motores paso a paso, la configuración del microstepping podría realizarse desde el microcontrolador, para conseguir movimientos rápidos y precisos.

Por último, el diseño de una aplicación dedicada a las funcionalidades del dispositivo de orientación, para la realización de panorámicas, generar movimientos a velocidades de rotación constantes, o la astrofotografía.

Referencias

[1] spqr, "Are all 50mm lenses equivalent?," October 14 2022. Crafting Pixels.

[2] OpenAstroTech Community, "OpenAstroTracker," 2024. Accessed: 2024-06-20.

[3] Isaac879, "Pan-tilt mount," 2024. Accessed: 2024-06-20.

[4] Arduino, "Uno-th_rev3e_sch." https://content.arduino.cc/assets/UNO-TH_Rev3e_sch.pdf, 2024. Accessed: 2024-06-12.

[5] M. T. Inc., *ATmega328P Datasheet*, 2024. Accessed: 2024-06-12.

[6] ams AG, *AS5048 Datasheet*, 2024. Accessed: 2024-06-12.

[7] P. Corporation, *Pololu A4988 Stepper Motor Driver Carrier*, 2024. Accessed: 2024-06-12.

[8] V. Intertechnology, *4N26 Datasheet*, 2024. Accessed: 2024-06-12.

[9] Brad, "Pcb trace width calculator." https://circuitcalculator.com/wordpress/2006/01/31/pcb-trace-width-calculator/, 2006. Accessed: 2024-06-14.

[10] Z. Labs, "As5048a arduino library." https://github.com/ZoetropeLabs/AS5048A-Arduino/tree/master/lib/AS5048A, 2023. GitHub repository.

[11] Pololu Robotics and Electronics, "Drv8825 stepper motor driver carrier, high current." https://www.pololu.com/product/2133, 2024. Accessed: 2024-07-18.

Documento:
Pliego de condiciones

Índice

1. Introducción y Objeto del Pliego de Condiciones

El presente Pliego de Condiciones tiene como finalidad regular los niveles técnicos y de calidad exigibles en el desarrollo del proyecto en cuestión. Se detallan las intervenciones necesarias de acuerdo con el contrato establecido y la legislación aplicable, especificando las responsabilidades de la parte contratante y la parte técnica, así como las relaciones y obligaciones derivadas del cumplimiento del contrato de desarrollo.

La parte contratante, **SANTIAGO GUILLÉN ROMO**, tiene la obligación de verificar los documentos del proyecto inmediatamente después de recibirlos. En caso de cualquier discrepancia, contradicción u omisión, debe comunicarlo a la parte técnica sin demora. Además, la parte contratante debe proporcionar todo el material necesario para el desarrollo del proyecto, ya sea en forma física o mediante el pago de las facturas correspondientes en un plazo máximo de cinco días laborables, conforme a lo estipulado en el contrato legal.

La parte técnica asume las responsabilidades facultativas del proyecto, las cuales incluyen la planificación y establecimiento de las condiciones técnicas, el cumplimiento de la normativa aplicable, el control de calidad y el control económico. También debe realizar la planificación temporal, seleccionar los materiales más adecuados, desarrollar, ensamblar y comprobar el sistema. Los documentos que integran el contrato, en orden de importancia en cuanto al valor de sus especificaciones en caso de omisión o aparente contradicción, son los siguientes:

1. Las condiciones fijadas en el propio documento de contrato.
2. El presente Pliego de Condiciones.
3. El resto de la documentación del proyecto: memoria, planos y presupuestos.

Cualquier propuesta de modificación del proyecto por parte de la parte contratante deberá ser presentada a la parte técnica con tiempo suficiente para su consideración. Si la propuesta es aprobada, la modificación del proyecto se realizará de mutuo acuerdo. En caso de que la parte contratante desee cesar el proyecto, deberá abonar la inversión en personal correspondiente a 40 días laborales, sin posibilidad de reclamar los gastos materiales ocasionados.

La fecha límite para el desarrollo del prototipo y el firmware se establece para el 3 de mayo de 2024. Toda la documentación actualizada del proyecto, incluyendo la memoria, planos, presupuesto y el presente Pliego de Condiciones, deberá entregarse a la parte contratante antes del 25 de julio de 2024. El incumplimiento de estas obligaciones conllevará la paralización del proyecto, sin derecho a reclamación alguna y con detrimento económico y de otro tipo para la parte incumplidora, hasta que la deficiencia sea solucionada. En caso de que no sea posible solucionar la deficiencia, se procederá a la compensación económica correspondiente.

Este Pliego de Condiciones pretende asegurar la correcta ejecución del proyecto, garantizando la calidad y el cumplimiento de todas las obligaciones contractuales por ambas partes.

2. Requisitos Técnicos y Normativos

Las condiciones técnicas son la serie de puntos que el sistema a desarrollar debe cumplir. El sistema por desarrollar, objetivo del presente proyecto, es un dispositivo que permita la orientación de una cámara con las siguientes características:

- **Comunicación**: el dispositivo debe de ser capaz de comunicar no solo debe de establecer una comunicación entre los sensores para la regulación de su posición, también debe de ser capaz de enviar y recibir comandos.
- **Estructura**: las diferentes partes deben de ensamblarse en una sola integrando todos los elementos que permiten la actuación y control de la orientación de la cámara fotográfica.

2.1. Condiciones Técnicas

Las condiciones técnicas son la serie de puntos que el sistema a desarrollar debe cumplir. El sistema por desarrollar, objetivo del presente proyecto, debe adherirse a los siguientes requisitos técnicos y normativos:

- El material de la PCB debe ser FR4-Standard TG 135-140.
- Debe contar con conexión micro USB.
- Debe contar con 3 conectores tipo JST B6B-XH-A y un conector.
- Todos los elementos electrónicos deben ser compatibles con la normativa RoHS y con la normativa de emisiones electromagnéticas, como la normativa EMC de la UE, Directiva (UE) 2014/30 del Parlamento Europeo y del Consejo.
- Los cables empleados deben ser de 0.0205mm^2 o 24AWG.
- Los plásticos empleados deben adherirse a la Directiva (UE) 2019/904 sobre la reducción del impacto de determinados productos de plástico en el medio ambiente y al Reglamento (UE) 2023/2055 relativo a la gestión sostenible de materiales plásticos. Para este proyecto, se utilizará PLA para la impresión 3D de componentes, que cumple con estas normativas.
- Las tolerancias en cuanto a las dimensiones de cada pieza deberán cumplir la ISO 2768-1.

2.2. Ensayos y test de verificación a realizar

Las siguientes comprobaciones se realizarán secuencialmente para asegurar la calidad y funcionalidad del sistema desarrollado:

- **Inspección de las soldaduras:** Verificar que todas las conexiones soldadas son correctas y no existen cortocircuitos.
- **Verificación de dimensiones y tolerancias:** Asegurar que las piezas cumplen con las dimensiones especificadas y las tolerancias permiten un ensamblaje adecuado sin holguras ni interferencias.
- **Inspección de los conectores de cables:** Comprobar que todos los conectores están bien crimpados y no presentan defectos.
- **Pruebas de funcionamiento:** Evaluar el desempeño del sistema en conjunto, verificando el movimiento de los ejes, la corrección de posición ante perturbaciones y la ejecución de comandos.

Documento:

Presupuesto

Índice

Índice de tablas

1. Objeto

En este documento se detalla el coste aproximado del desarrollo del trabajo, expresando los valores elementales, descompuestos, de ejecución de material, de contrata y por producción.

En el desarrollo del presupuesto se han tenido encanta los materiales empleados, el tiempo de desarrollo, tanto para las estructuras como para el desarrollo de la PCB y del firmware.

Se ha estructurado el documento del presupuesto siguiendo las distinciones de la memoria, separando por ejes, pitch y yaw, el diseño de la PCB y diseño del firmware del dispositivo, formando todas las partes el dispositivo completo.

2. Cuadro de precios elementales

Esta sección divide las tablas de precios en tres categorías: mano de obra, materiales y amortización, detallando en cada apartado los factores tomados en cuenta para su elaboración.

2.1. Mano de obra

En la Tabla 1, se encuentra el coste de la mano de obra del ingeniero encargado del desarrollo del dispositivo, valor respaldado por Convenio colectivo nacional de empresas de ingeniería, establecido en la: `BOE-A-2023-11785`.

MANO DE OBRA			
Ref.	**Unidad**	**Descripción**	**Precio unitario [€]**
ING	h	Ingeniero	20,00

Tabla 1: Precios unitarios mano de obra.

2.2. Materiales

En la siguiente tabla, Tabla 2, se encuentran los precios unitarios de los materiales empleados en la construcción del dispositivo, donde se tiene encuentra el emplazamiento utilizado, despacho y taller, para su desarrollo.

MATERIALES			
Ref.	**Unidad**	**Descripción**	**Precio unitario [€]**
DESP	h	Despacho	20,00
TALL	h	Taller	30,00
DVR	ud	DVR8826	2,40
JST-XH	ud	Conector JST-XH	0,25
PLA	g	Filamento PLA	0,02
ROD60	ud	Rodamiento 6012-2RS	9,58
ROD8	ud	Rodamiento 608ZZ	1,20
M3x15	ud	Tornillo M3 de 15mm	0,12
TORM3	ud	Tuerca M3	0,10
SEN	ud	Encoder magnético AS5048A	20,74
AWG	cm	Cale AWG 4P	0,16
F4P	ud	Conector hembra XH 2.54	0,21

Tabla 2: Precios unitarios materiales.

En la Tabla 3, se encuentra la estimación de coste de fabricación cuantificando el tiempo por pieza.

Datos de Construcción por Dispositivo						
Pieza por Dispositivo	**Plástico Unitario (g)**	**Unidades producidas**	**Plástico Total (g)**	**Tiempo de impresión**	**Coste (€)**	**N.º de Dibujo**
Pitch						
Engranaje conductor	4	1	4	00:27:00	0,07	TFM-P010
Engranaje conducido	14	1	14	01:35:00	0,25	TFM-P011
Base	113	1	113	11:06:00	2,02	TFM-P003
Soporte Derecho	26	1	26	02:40:00	0,46	TFM-P012
Soporte Izquierdo	19	1	19	01:58:00	0,34	TFM-P002
Soporte Cámara	64	1	64	09:58:00	1,14	TFM-P001
Sujección encoder	14	1	14	02:00:00	0,25	TFM-P013
Tapa encoder	5	1	5	00:42:00	0,09	TFM-P014
Total	259	8	259	30:26:00	4,62	
Yaw						
Engranaje conducido	70	1	70	08:56:00	1,25	TFM-P008
Engranaje conductor	8	1	8	00:50:00	0,14	TFM-P006
Soporte PCB	9	1	9	01:40:00	0,16	TFM-P009
Tope rodameinto	2	1	2	00:19:00	0,04	TFM-P005
Sujección encoder	14	1	14	02:00:00	0,25	TFM-P013
Tapa encoder	5	1	5	00:42:00	0,09	TFM-P014
Soporte Patas	58	1	58	08:20:00	1,03	TFM-P007
Total	166	7	166	22:47:00	2,96	

Tabla 3: Coste construcción estructura dispositivo.

2.3. Amortización

La amortización asigna al coste del producto o servicio la depreciación o pérdida de valor de los elementos del activo no corriente, distribuyendo este gasto a lo largo

del tiempo. En nuestro caso, los elementos amortizables incluyen los equipos empleados en los ensayos y el software informático. El cálculo de la amortización se realiza de la siguiente manera:

$$A = \left(\frac{C}{H}\right) \cdot h$$

Donde A es la amortización, C es el coste anual en euros, H es la vida útil en horas y h son las horas de funcionamiento.

La vida útil, o coeficiente de amortización lineal, se ha determinado utilizando los pasos indicados en la Agencia Tributaria , apartado: manuales-videos-folletos. Consideraremos una vida útil máxima de 5 años para las herramientas y entre 1 y 2 años para el software informático.

Para calcular el número de horas trabajadas al año, primero determinamos el total de semanas disponibles (52) restando las semanas de vacaciones (4) y días festivos (14) del total de semanas en un año. Luego multiplicamos este resultado por las horas de trabajo semanales. La fórmula es la siguiente:

$$H_{\text{año}} = (52 - (4 + 2)) \times 40$$

Simplificando esta ecuación, tenemos:

$$H_{\text{año}} = 46 \times 40 = 1840\,\text{horas/año}$$

Con estos datos, obtenemos las tablas de amortización de herramientas, software informático y envejecimiento de equipos, tablas 4, 5 y 6.

AMORTIZACIÓN Herramientas						
Ref.	**Descripción**	**Precio [€]**	**Vida útil [años]**	**Coste de uso [€/h]**	**Uso [h]**	**Coste de Amortización [€]**
3DP	Artillery sidewinder x1	550,00	5	0,06	53,00	3,17
GENU	Fuente de tensión de laboratorio	219,00	5	0,02	25,00	0,60
LOGIC	Analizador lógico de señales	80,00	5	0,01	5,00	0,04
ORD	Ordenador	1.200,00	5	0,13	111,00	14,48
THER	Herramientas de taller	550,00	5	0,06	6,00	0,36
TOTAL						**18,64 €**

Tabla 4: Amortización herramientas

AMORTIZACIÓN software informático						
Ref.	**Descripción**	**Precio [€]**	**Vida útil [años]**	**Coste de uso [€/h]**	**Uso [h]**	**Coste de Amortización [€]**
MO	Microsoft Office	99,00	1	0,05	2,00	0,11
SWL	SolidWorks Standar licencia	4.230,00	1	2,30	16,00	36,78
MAT	Matlab licencia	186,00	2	0,05	10,00	0,51
TOTAL						**37,40 €**

Tabla 5: Amortización software informático

AMORTIZACIÓN por Envejecimiento de los Equipos						
Ref.	**Descripción**	**Precio [€]**	**Vida útil [años]**	**Coste de uso [€/h]**	**Uso [h]**	**Coste de Amortización [€]**
3DP	Artillery sidewinder x1	550,00	5	0,06	320,00	19,13
GENU	Fuente de tensión de laboratorio	219,00	5	0,02	320,00	7,62
LOGIC	Analizador logico de señales	80,00	5	0,01	320,00	2,78
THER	Herramientas de taller	550,00	5	0,06	320,00	19,13
ORD	Ordenador	1.200,00	5	0,13	320,00	41,74
TOTAL						**90,40 €**

Tabla 6: Amortización por envejecimiento de los equipos.

3. Precios Descompuestos

En este apartado se incluyen los cuadros de precios descompuestos, divididos en cuatro unidades de obra: Sistema de control, que engloba el diseño de la PCB, Movimiento yaw diseño y fabricación de los componentes que intervienen en el movimiento del eje yaw y Movimiento pitch diseño y fabricación de los componentes que intervienen en el movimiento del eje pitch. Estas unidades de obra abarcan los diferentes aspectos económicos del proyecto.

Unidad de Obra 1: Sistema de control					
Ref.	**Unidad**	**Descripción**	**Precio**	**Cantidad**	**Parcial**
PCB	ud	**Diseño de la PCB, sistema de control de la montura motorizada y ensamblado de conectores**			
		Materiales			
ORD	h	Ordenador	0,16 €	70	11,20 €
DESP	h	Despacho	20,00 €	70	1.400,00 €
TALL	h	Taller	30,00 €	2	60,00 €
DVR	ud	DVR8826	2,40 €	2	4,80 €
JST-XH	ud	Conector JST-XH	0,25 €	4	1,00 €
		SERVICIOS			
PCB	ud	Custom PCB	2,00 €	1	2,00 €
ENSPCB	ud	Ensamblaje	69,55 €	1	69,55 €
ENV	ud	Envio	20,00 €	1	20,00 €
		BOM			
C1,C2	ud	CC0603JRX7R9BB104	0,0037 €	2	0,01 €
C3	ud	0603B104K500NT	0,0019 €	1	0,00 €
C5	ud	CC0603KRX7R9BB103	0,0019 €	1	0,00 €
C4	ud	TC211A106K016Y	0,07 €	1	0,07 €
C6	ud	VT1E470M0605	0,03 €	1	0,03 €
C7,C8	ud	UWX1C101MCL1GB	0,06 €	2	0,12 €
D1	ud	1N4002W	0,01 €	1	0,01 €
D2	ud	SM4007PL	0,01 €	1	0,01 €
DC1	ud	DC-005 2.0	0,05 €	1	0,05 €
H1,H2	ud	PZ254V-12-6P	0,03 €	2	0,06 €
H3,H4	ud	210S-1*4P L=11.6MMGold-plated black	0,04 €	2	0,08 €
J1,J2	ud	?		2	0,00 €
L1	ud	MLF2012E100JT000	0,04 €	1	0,04 €
LED1	ud	MHT170CGCT	0,03 €	1	0,03 €
LED2	ud	MHT170UBCT	0,03 €	1	0,03 €
R1	ud	0603WAJ0105T5E	0,0009 €	1	0,00 €
R4,R5,R6,R7, R8,R9,R10	ud	0603WAF1002T5E	0,0009 €	7	0,01 €
R11,R12	ud	0603WAF3900T5E	0,0009 €	2	0,00 €
R2,R3	ud	0805W8F2001T5E	0,0019 €	2	0,00 €
SK1	ud	Z-ICS0S28P-NG00	0,08 €	1	0,08 €
U2	ud	CH340C	0,47 €	1	0,47 €
U3,U4	ud	A4988		2	0,00 €
U5	ud	SPX1117M3-L-5-0/TR	0,11 €	1	0,11 €
U6	ud	SS12D10G5	0,13 €	1	0,13 €
U7,U8	ud	4N26	0,37 €	2	0,73 €
U9	ud	DB128L-5.08-3P-BK-S	0,27 €	1	0,27 €
U11,U12,U13	ud	B6B-XH-A(LF)(SN)	0,08 €	3	0,25 €
X2	ud	CSTCE16M0V53-R0	0,24 €	1	0,24 €
		MEDIOS AUXILIARES			
	%	Medios auxiliares sobre costes directos	5,00 %	94,39 €	99,10 €

Tabla 7: Cuadro Precios Descompuestos Unidad de Obra 1.

Unidad de Obra 2: Movimiento yaw					
Ref.	**Unidad**	**Descripción**	**Precio**	**Cantidad**	**Parcial**
YAW	ud	**Piezas que forman del eje yaw, que se compone por las piezas fabricadas en PLA, tornillos, rodamientos, motor y sensor.**			
		Materiales			
DESP	h	Despacho	20,00 €	8	160,00 €
ORD	h	Ordenador	0,16 €	8	1,28 €
SWL	h	SolidWorks Standar licencia	2,30 €	8	18,40 €
TALL	h	Taller	30,00 €	23	690,00 €
THER	h	Herramientas taller	0,06 €	3	0,18 €
PLA	g	Filamento PLA	0,02 €	166	2,96 €
3DP	h	Artillery sidewinder x1	0,06 €	23	1,38 €
M3x15	ud	Tornillo M3 de 15mm	0,12 €	4	0,48 €
MOT	ud	Motor nema 17 42HS02	12,00 €	1	12,00 €
SEN	ud	Encoder magnético AS5048A	20,74 €	1	20,74 €
ROD60	ud	Rodamiento 6012-2RS	9,58 €	1	9,58 €
AWG	cm	Cale AWG 4P	0,16 €	20	3,20 €
F4P	ud	Conector hembra XH 2.54	0,21 €	1	0,21 €
		MEDIOS AUXILIARES			
	%	Medios auxiliares sobre costes directos	10,00 %	920,41 €	1.012,45 €

Tabla 8: Cuadro Precios Descompuestos Unidad de Obra 2.

Unidad de Obra 3: Movimiento pitch					
Ref.	**Unidad**	**Descripción**	**Precio**	**Cantidad**	**Parcial**
PICTH	ud	**Piezas que forman del eje pitch, que se compone por las piezas fabricadas en PLA, tornillos, rodamientos, motor y sensor.**			
		Materiales			
DESP	h	Despacho	20,00 €	10	200,00 €
ORD	h	Ordenador	0,13 €	10	1,30 €
SWL	h	SolidWorks Standar licencia	2,30 €	8	18,40 €
TALL	h	Taller	30,00 €	30	900,00 €
THER	h	Herramientas taller	0,06 €	3	0,18 €
PLA	g	Filamento PLA	0,02 €	259	4,62 €
3DP	h	Artillery sidewinder x1	0,06 €	30	1,80 €
M3x15	ud	Tornillo M3 de 15mm	0,12 €	2	0,24 €
MOT	ud	Motor nema 17 42HS02	12,00 €	1	12,00 €
SEN	ud	Encoder magnético AS5048A	20,74 €	1	20,74 €
ROD8	ud	Rodamiento 608ZZ	1,20 €	2	2,40 €
TORM3	ud	Tuerca M3	0,10 €	2	0,20 €
AWG	cm	Cale AWG 4P	0,16 €	20	3,20 €
F4P	ud	Conector hembra XH 2.54	0,21 €	1	0,21 €
		MEDIOS AUXILIARES			
	%	Medios auxiliares sobre costes directos	10,00 %	1.165,29 €	1.281,82 €

Tabla 9: Cuadro Precios Descompuestos Unidad de Obra 3.

Unidad de Obra 4: Diseño del código, Puesta en marcha, Ensayos y Resultados					
Ref.	**Unidad**	**Descripción**	**Precio**	**Cantidad**	**Parcial**
FIRM	ud	**Proceso de creación del código necesario para el correcto funcionamiento del sistema sobreactuado para la orientación de una cámara fotográfica.**			
		Materiales			
ORD	h	Ordenador	0,13 €	25	3,25 €
DESP	h	Despacho	20,00 €	15	300,00 €
ENSAY	h	Ensayos de funcionamiento	20,00 €	60	1.200,00 €
MEM	h	Recopilación, tratamiento y formato de resultados	21,00 €	80	1.680,00 €
TALL	h	Taller	30,00 €	10	300,00 €
GENU	h	Fuente de tensión de laboratorio	0,02 €	25	0,50 €
LOGIC	h	Analizador logico de señales	0,01 €	5	0,05 €
MAT	h	Matlab licencia	0,05 €	10	0,50 €
	%	Medios auxiliares sobre costes directos	10,00 %	3.484,30 €	3.832,73 €

Tabla 10: Cuadro Precios Descompuestos Unidad de Obra 4.

Mano de Obra por Unidad de Obra					
DES	**Unidad**	**Descripción**	**Precio**	**Cantidad**	**Parcial**
PCB	ud	**Diseño de la PCB, sistema de control de la montura motorizada y ensamblado de conectores**			
		MANO DE OBRA			
ING	h	Ingeniero	20,00	72	1.440,00 €
YAW		**Piezas que forman del eje yaw, que se compone por las piezas fabricadas en PLA, tornillos, rodamientos, motor y sensor.**			
		MANO DE OBRA			
ING	h	Ingeniero	20,00	13	260,00 €
PICTH		**Piezas que forman del eje pitch, que se compone por las piezas fabricadas en PLA, tornillos, rodamientos, motor y sensor.**			
		MANO DE OBRA			
ING	h	Ingeniero	20,00	13	260,00 €
FIRM		**Proceso de creación del código necesario para el funcionamiento del sistema sobreactuado para la orientación de una cámara fotográfica, ensayos y pruebas**			
ING	h	Ingeniero	20,00	165	3.300,00 €
Total					**5.260,00 €**

Tabla 11: Cuadro Precios Descompuestos Mano de Obra por Unidad de Obra.

4. Mediciones

En las mediciones, consideramos aquellas unidades de obra que se definen como elementos físicos, es decir, los elementos y su cantidad.

Tabla de Mediciones			
Ref.	**Unidad**	**Descripción**	**Cantidad**
PCB	ud	**Diseño de la PCB, sistema de control de la montura motorizada y ensamblado de conectores**	1
YAW	ud	**Piezas que forman del eje yaw, que se compone por las piezas fabricadas en PLA, tornillos, rodamientos, motor y sensor.**	1
PICTH	ud	**Piezas que forman del eje pitch, que se compone por las piezas fabricadas en PLA, tornillos, rodamientos, motor y sensor.**	1

Tabla 12: Cuadro de Mediciones.

5. Presupuesto Ejecución de Material

Ref.	**Unidad**	**Descripción**	**Precio**	**Cantidad**	**Total**
PCB	ud	**Diseño de la PCB, sistema de control de la montura motorizada y ensamblado de conectores.**	1.539,10 €	1	1.539,10 €
YAW	ud	**Piezas que forman del eje yaw, que se compone por las piezas fabricadas en PLA, tornillos, rodamientos, motor y sensor.**	1.316,80 €	1	1.316,80 €
PICTH	ud	**Piezas que forman del eje pitch, que se compone por las piezas fabricadas en PLA, tornillos, rodamientos, motor y sensor.**	1.541,82 €	1	1.541,82 €
FIRM	ud	**Proceso de creación del código necesario para el funcionamiento del sistema sobreactuado para la orientación de una cámara fotográfica, ensayos y pruebas.**	7.132,73 €	1	7.132,73 €
ENV	ud	**Amortización envejecimiento de equipos.**	90,40 €	1	90,40 €
TOTAL: PRESUPUESTO DE EJECUCIÓN MATERIAL					**11.620,86 €**

Tabla 13: Presupuesto de ejecución material.

6. Presupuesto de Contrata

El presupuesto de contrata es la estimación total del coste del proyecto. Incluye el coste de materiales, mano de obra y equipos (presupuesto de ejecución material), además de los gastos generales, beneficio industrial e impuestos. Este presupuesto ofrece una visión completa del coste total del proyecto.

Resumen de presupuesto		
TOTAL: PRESUPUESTO DE EJECUCIÓN MATERIAL		11.620,86 €
COSTE DE LOS MATERIALES		6.270,46 €
COSTE DE LA MANO DE OBRA		5.260,00 €
AMORTIZACIÓN ENVEJECIMIENTO EQUIPOS		90,40 €
GASTOS GENERALES	10 %	1.162,09 €
BENEFICIO INDUSTRIAL	20 %	2.324,17 €
SUMA PARCIAL		**12.782,94 €**
IVA	21 %	2.684,42 €
Total		**15.467,36 €**

Tabla 14: Presupuesto de contrata.

El presupuesto total aproximado de la inversión asciende a **QUINCE MIL CUATROCIENTOS SESENTA Y SIETE CON TREINTA Y SEIS CÉNTIMOS** , impuestos incluidos.

7. Gastos por Volumen de Producción

En esta sección, se presenta un análisis de los gastos asociados al volumen de producción, utilizando un valor simbólico de 100 unidades para facilitar la visión de costes. Es importante destacar que la empresa con la que se ha fabricado la PCB, jlcpcb, acepta pedidos de miles de unidades, lo que no es un limitante en cuanto a solicitar volúmenes altos. Además, para calcular el coste de cada pieza necesaria para generar el movimiento de cada eje, se ha empleado el mismo método de fabricación utilizado durante el desarrollo del dispositivo.

7.1. Ejecución de Material

Presupuesto de Ejecución de Material para la Producción de 100 unidades					
DES	**Unidad**	**Descripción**	**Precio**	**Cantidad**	**Parcial**
PCB100	ud	100 PCB ensamblados	3,37	100	337,20 €
PITCH100	ud	Elementos intervinientes en el movimiento del eje pitch	4,62	100	462,06 €
YAW100	ud	Elementos intervinientes en el movimiento del eje yaw	2,96	100	296,14 €
		MEDIOS AUXILIARES			
	%	Medios auxiliares sobre costes directos	5 %	1.095,40 €	54,77 €
Total					**1.150,17 €**

Tabla 15: Presupuesto de Ejecución de Material para la Producción de 100 unidades.

7.2. Ejecución de Mano de Obra

Presupuesto de Mano de Obra para la Producción de 100 unidades					
DES	**Unidad**	**Descripción**	**Precio**	**Cantidad**	**Parcial**
PCB100C	h	Inspección/Control de calidad y carga de firmware	20,00	8	160,00 €
PITCH100C	h	Inspección de piezas del eje pitch	15,00	2	30,00 €
YAW100C	h	Inspección de piezas del eje yaw	15,00	2	30,00 €
		MEDIOS AUXILIARES			
	%	Medios auxiliares sobre costes directos	5 %	220,00 €	11,00 €
Total					**231,00 €**

Tabla 16: Presupuesto de Mano de Obra para la Producción de 100 unidades.

7.3. Presupuesto de Contrata por Dispositivo y PVP

Resumen de presupuesto unitario por 100 unidades		
PCB		
TOTAL: PRESUPUESTO DE EJECUCIÓN MATERIAL		5,22 €
COSTE DE LOS MATERIALES		3,54 €
COSTE DE LA MANO DE OBRA		1,68 €
Eje pitch		
TOTAL: PRESUPUESTO DE EJECUCIÓN MATERIAL		5,17 €
COSTE DE LOS MATERIALES		4,85 €
COSTE DE LA MANO DE OBRA		0,32 €
Eje yaw		
TOTAL: PRESUPUESTO DE EJECUCIÓN MATERIAL		3,42 €
COSTE DE LOS MATERIALES		3,11 €
COSTE DE LA MANO DE OBRA		0,32 €
AMORTIZACIÓN DE I+D	15.467,36 €	154,67 €
GASTOS GENERALES	10 %	16,85 €
BENEFICIO INDUSTRIAL	15 %	27,80 €
SUMA PARCIAL		**204,49 €**
IVA	21 %	42,94 €
Total		**247,43 €**

Tabla 17: Resumen de presupuesto unitario por 100 unidades.

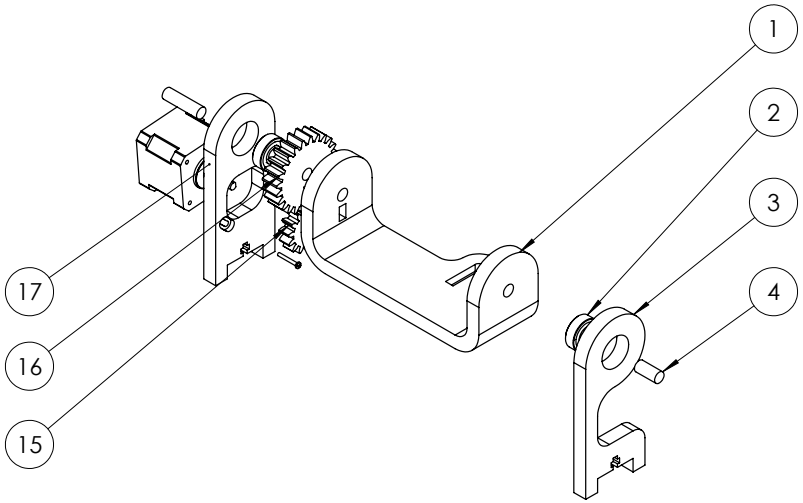
El precio de venta al público total aproximado para el dispositivo completo asciende a **DOSCIENTOS CUARENTA Y SIETE CON CUARENTA Y TRES CÉNTIMOS** , impuestos incluidos.

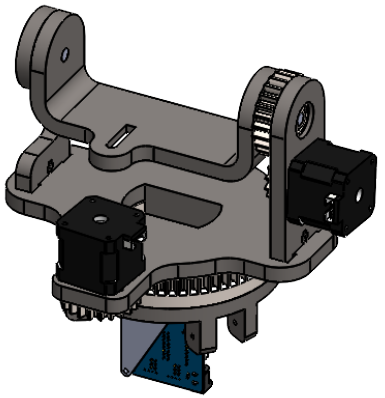
Documento:

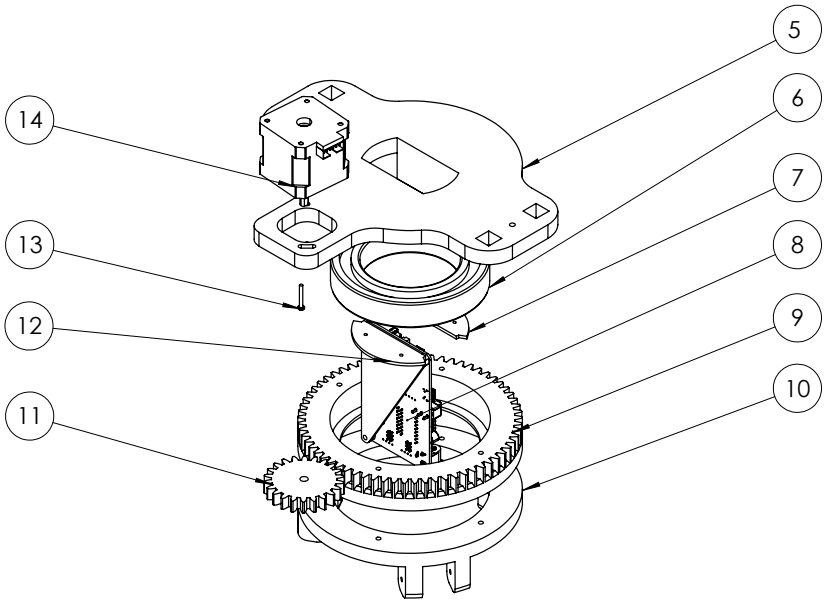
Planos

Índice

Piezas







NÚMERO	NOMBRE	Nº PLANO	Cantidad
1	Soporte cámara	TFM-P001	1
2	Rodamiento 608ZZ	-	2
3	Soporte lateral izquierdo	TFM-P002	1
4	Pasador	-	2
5	Base	TFM-P003	1
6	Rodamiento 6012-2RS	-	1
7	Tope rodamiento	TFM-P005	1
8	PCB	-	1
9	Engranaje conducido_H	TFM-P006	1
10	Sujección patas	TFM-P007	1
11	Engranaje contuctor_H	TFM-P008	1
12	Soporte PCB	TFM-P009	1
13	Tornillo M3x15mm	-	4
14	Motor nema 17	-	2
15	Engranaje conductor_V	TFM-P010	1
16	Engranaje conducido_V	TFM-P011	1
17	Soporte lateral derecho	TFM-P012	1

SI NO SE INDICA LO CONTRARIO:
LAS COTAS SE EXPRESAN EN MM
ACABADO SUPERFICIAL:
TOLERANCIAS:
LINEAL:
ANGULAR:

REBARBAR Y ROMPER ARISTAS VIVAS

NO CAMBIE LA ESCALA | REVISIÓN

	NOMBRE	FIRMA	FECHA
DIBUJ.	SANGUIRO		04/07/2024
VERIF.	SANGUIRO		04/07/2024
APROB.	SANGUIRO		04/07/2024
FABR.	SANGUIRO		04/07/2024
CALID.	SANGUIRO		04/07/2024

TÍTULO:
Diseño de un sistema sobreactuado para la orientación de una cámara fotográfica

MATERIAL: PLA

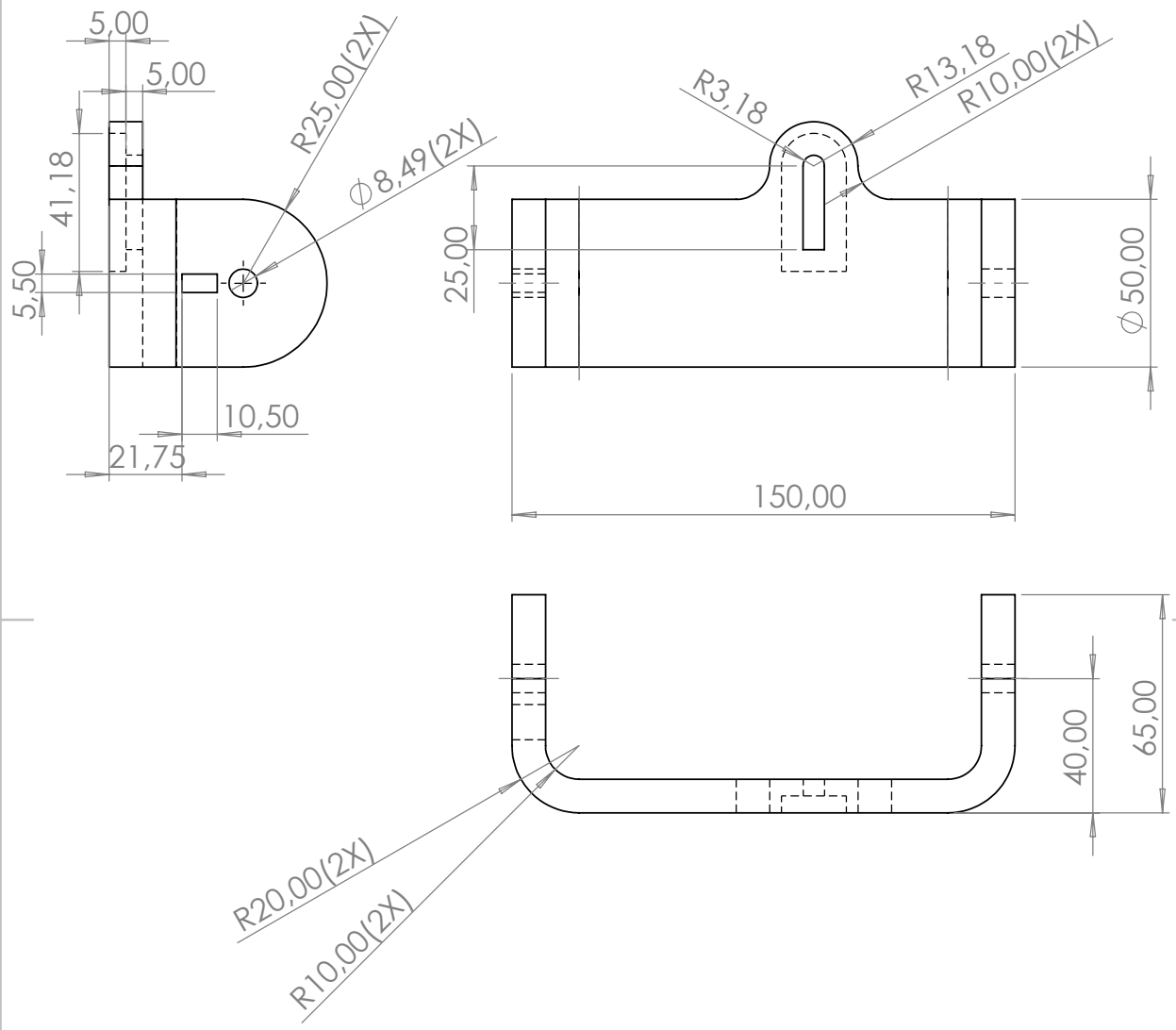
N.º DE DIBUJO: TFM-C001

A3

PESO:

ESCALA:1:3

HOJA 1 DE 15



SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:		REBARBAR Y ROMPER ARISTAS VIVAS	NO CAMBIE LA ESCALA / REVISIÓN

	NOMBRE	FIRMA	FECHA
DIBUJ.	SANGUIRO		04/07/2024
VERIF.	SANGUIRO		04/07/2024
APROB.	SANGUIRO		04/07/2024
FABR.	SANGUIRO		04/07/2024
CALID.	SANGUIRO		04/07/2024

TÍTULO: Soperte cámara

MATERIAL: PLA

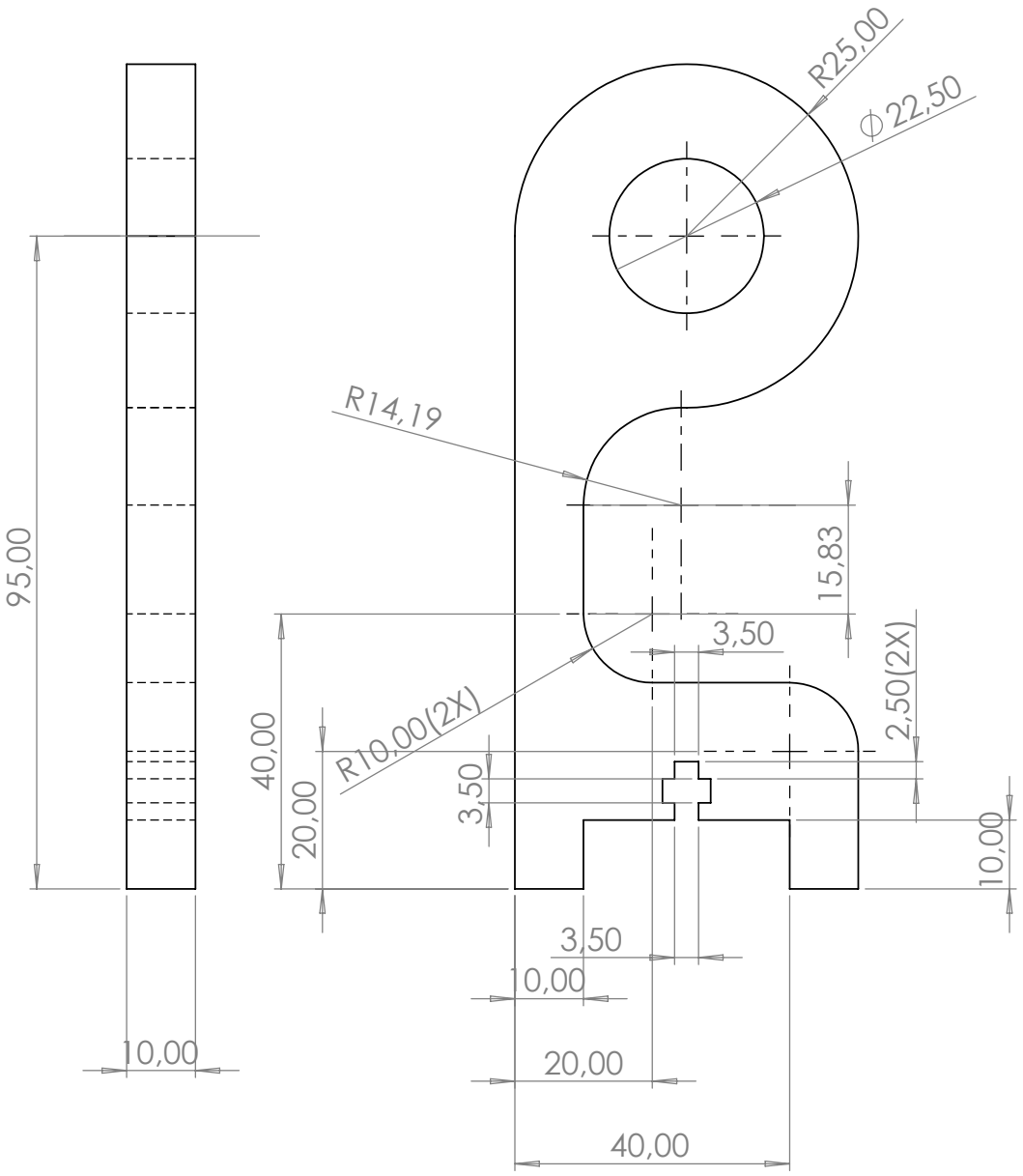
N.º DE DIBUJO: TFM-P001

A4

PESO:

ESCALA:1:2

HOJA 2 DE 15



SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:		REBARBAR Y ROMPER ARISTAS VIVAS	NO CAMBIE LA ESCALA / REVISIÓN

	NOMBRE	FIRMA	FECHA
DIBUJ.	SANGUIRO		04/07/2024
VERIF.	SANGUIRO		04/07/2024
APROB.	SANGUIRO		04/07/2024
FABR.	SANGUIRO		04/07/2024
CALID.	SANGUIRO		04/07/2024

TÍTULO: Soporte lateral izquierdo

MATERIAL: PLA

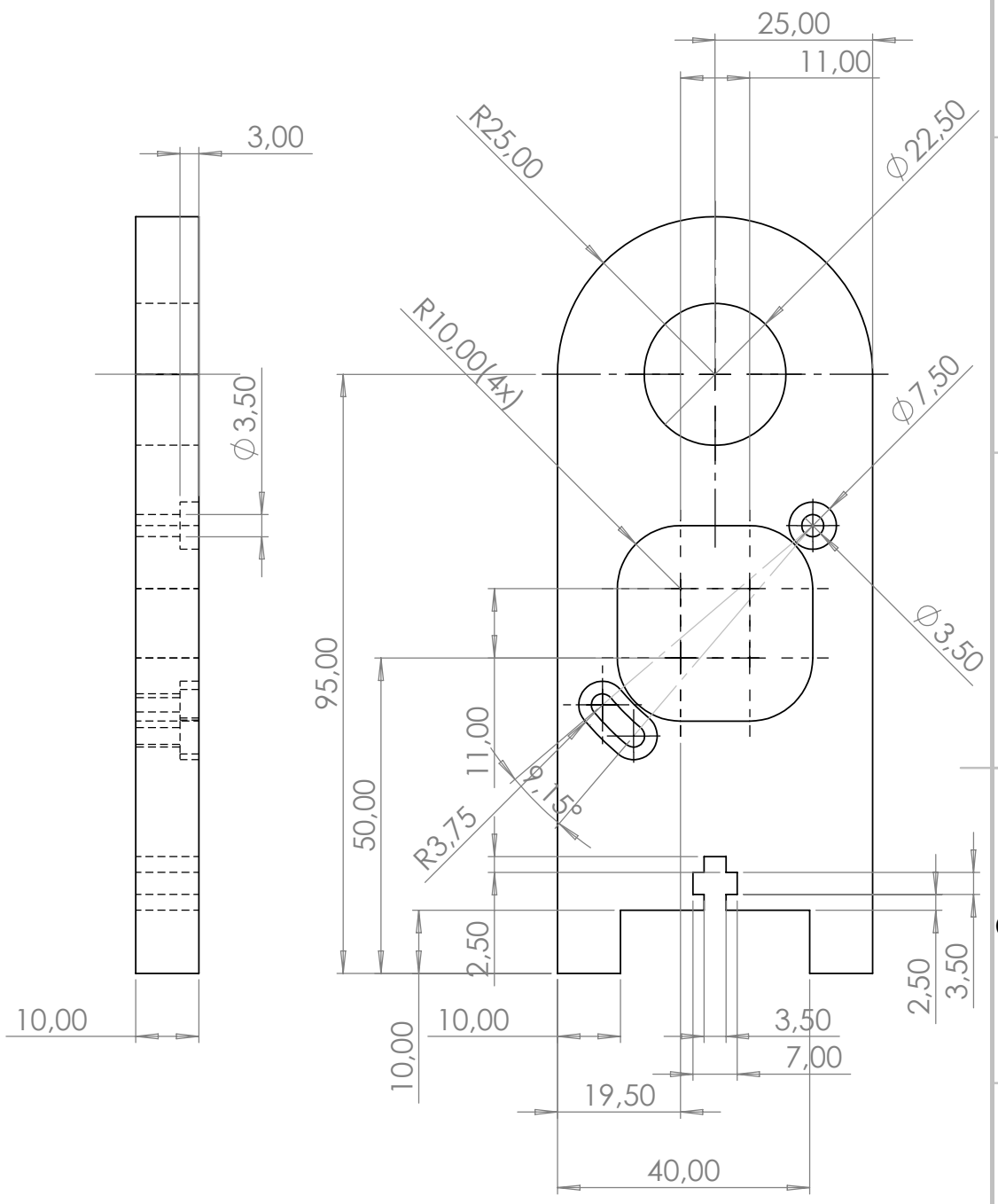
N.º DE DIBUJO: TFM-P002

A4

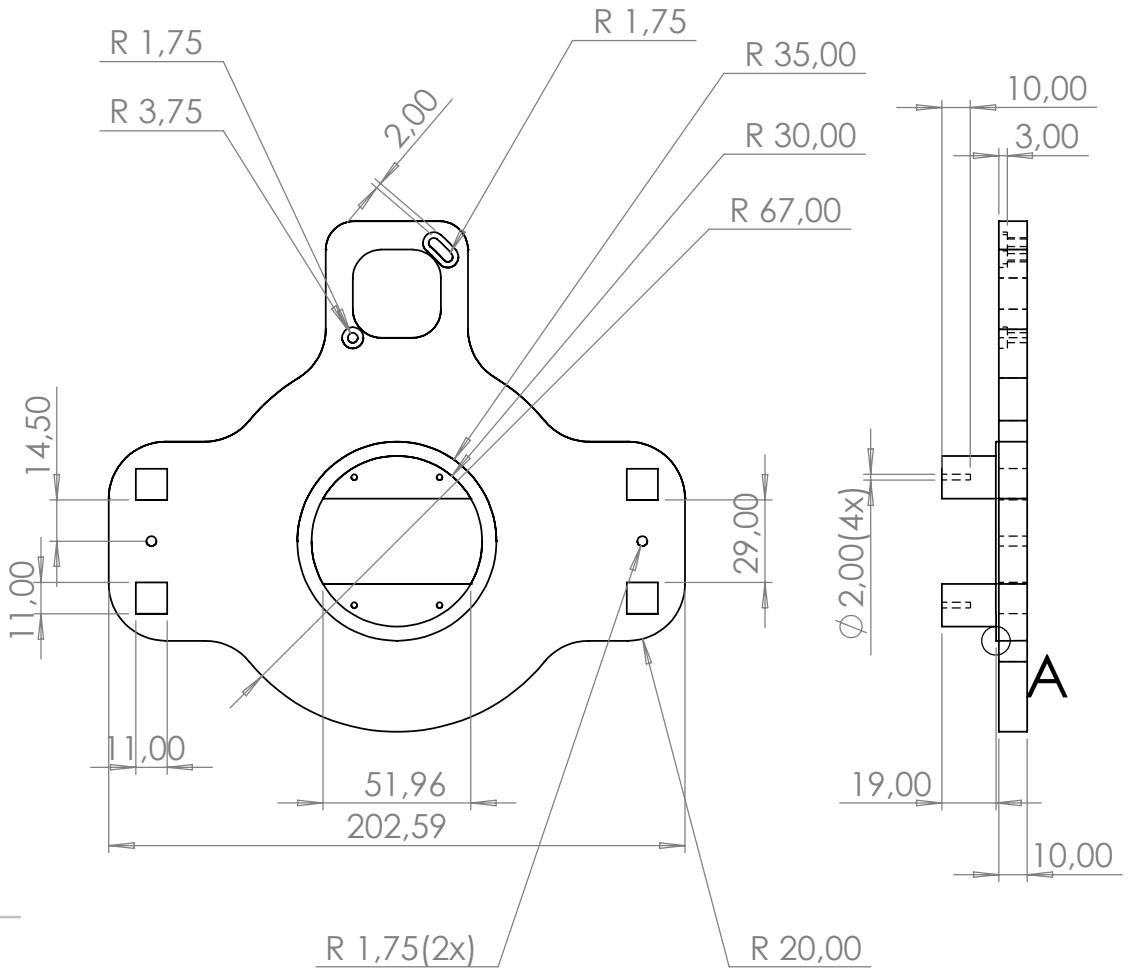
PESO:

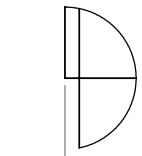
ESCALA:1:1

HOJA 3 DE 15



SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:					REBARBAR Y ROMPER ARISTAS VIVAS	NO CAMBIE LA ESCALA	REVISIÓN
	NOMBRE	FIRMA	FECHA			TÍTULO: Soporte lateral derecho	
DIBUJ.	SANGUIRO		04/07/2024				
VERIF.	SANGUIRO		04/07/2024				
APROB.	SANGUIRO		04/07/2024				
FABR.	SANGUIRO		04/07/2024				
CALID.	SANGUIRO		04/07/2024	MATERIAL: PLA		N.º DE DIBUJO TFM-C012	A4
				PESO:		ESCALA:1:1	HOJA 4 DE 15

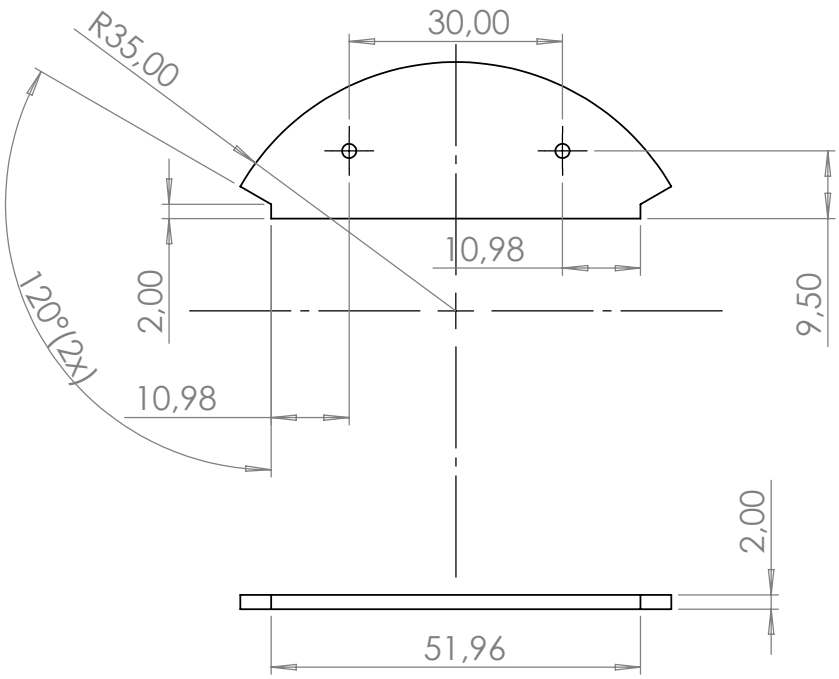




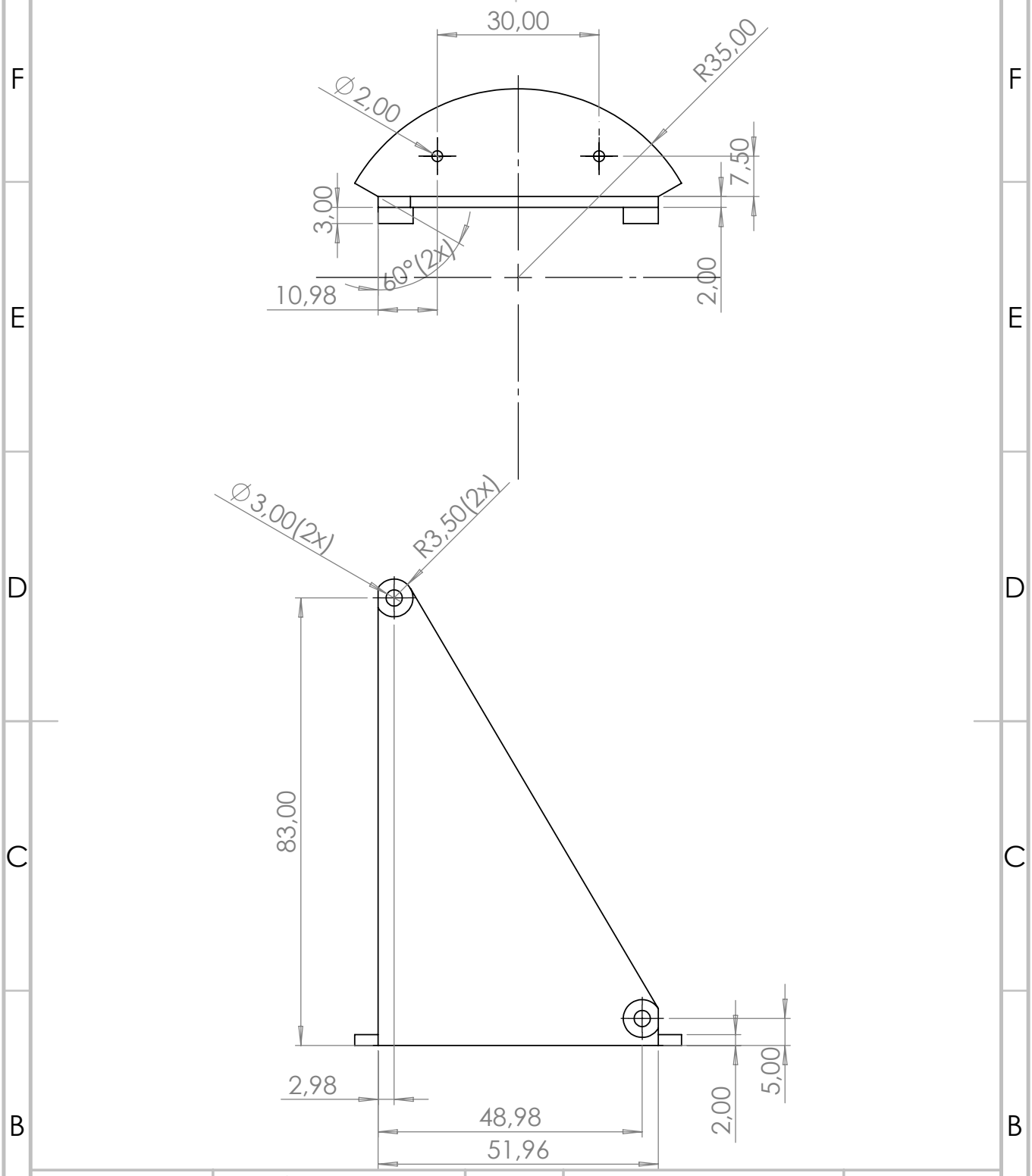
DETALLE A

ESCALA 2 : 1

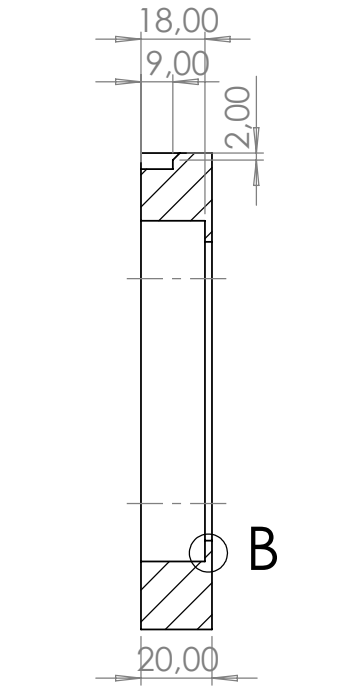
SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:				REBARBAR Y ROMPER ARISTAS VIVAS		NO CAMBIE LA ESCALA	REVISIÓN
	NOMBRE	FIRMA	FECHA			TÍTULO:	
DIBUJ.	SANGUIRO		04/07/2024			Base	
VERIF.	SANGUIRO		04/07/2024				
APROB.	SANGUIRO		04/07/2024				
FABR.	SANGUIRO		04/07/2024				
CALID.	SANGUIRO		04/07/2024	MATERIAL:		N.º DE DIBUJO	A4
				PLA		TFM-C003	
				PESO:		ESCALA:1:5	HOJA 5 DE 15



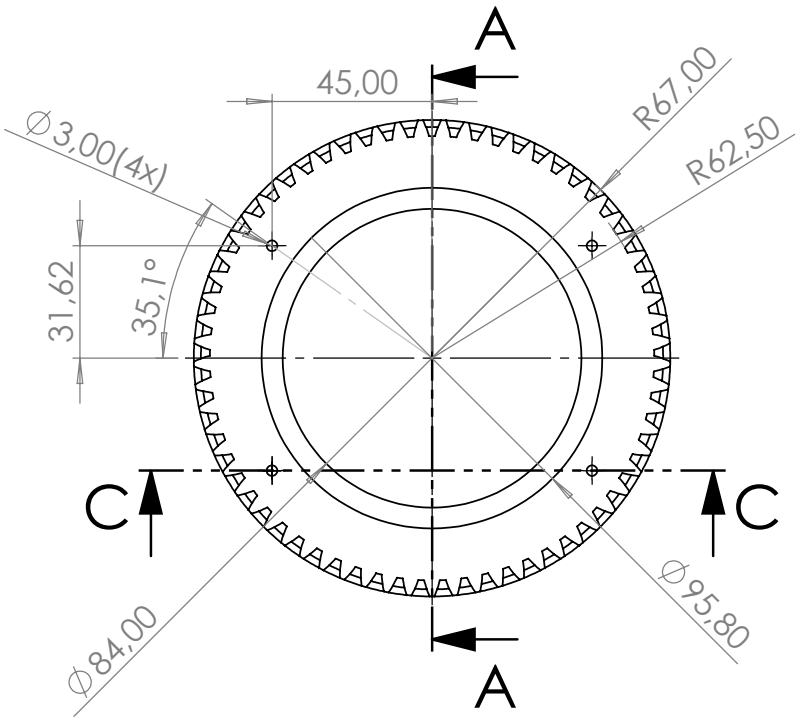
SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:			REBARBAR Y ROMPER ARISTAS VIVAS	NO CAMBIE LA ESCALA	REVISIÓN		
	NOMBRE	FIRMA	FECHA	TÍTULO: Tope rodamiento			
DIBUJ.	SANGUIRO		04/07/2024				
VERIF.	SANGUIRO		04/07/2024				
APROB.	SANGUIRO		04/07/2024				
FABR.	SANGUIRO		04/07/2024				
CALID.	SANGUIRO		04/07/2024	MATERIAL: PLA	N.º DE DIBUJO TFM-C005	A4	
				PESO:	ESCALA:1:1	HOJA 6 DE 15	



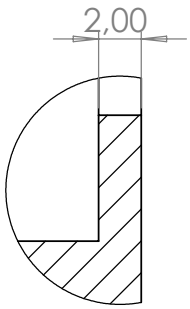
SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:			REBARBAR Y ROMPER ARISTAS VIVAS		NO CAMBIE LA ESCALA		REVISIÓN
	NOMBRE	FIRMA	FECHA				TÍTULO:
DIBUJ.	SANGUIRO		04/07/2024				Soporte PCB
VERIF.	SANGUIRO		04/07/2024				
APROB.	SANGUIRO		04/07/2024				
FABR.	SANGUIRO		04/07/2024				
CALID.	SANGUIRO		04/07/2024	MATERIAL:		N.º DE DIBUJO	A4
				PLA		TFM-C009	
				PESO:		ESCALA:1:1	HOJA 7 DE 15



SECCIÓN A-A

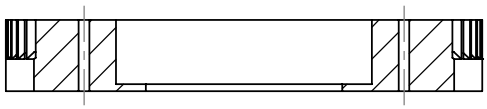


Modulo: 2
Ángulo de presión: 20°
Número de dientes: 65



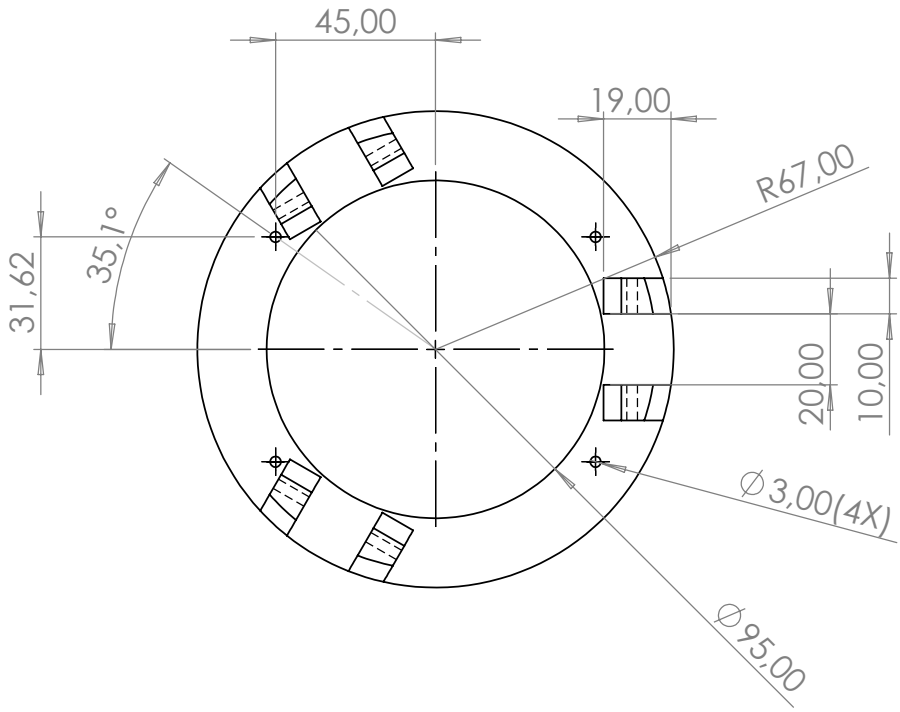
DETALLE B

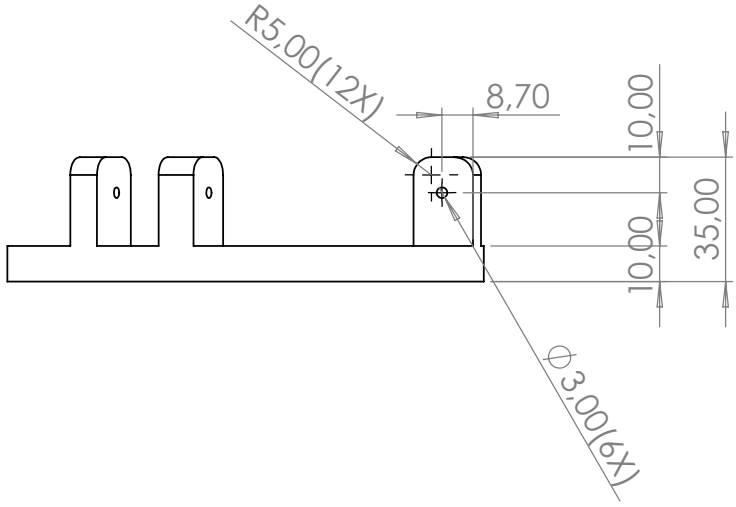
ESCALA 3 : 1



SECCIÓN C-C

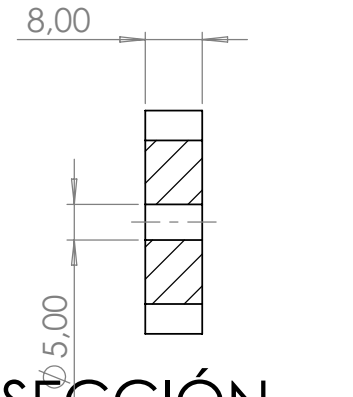
SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:					REBARBAR Y ROMPER ARISTAS VIVAS	NO CAMBIE LA ESCALA	REVISIÓN
	NOMBRE	FIRMA	FECHA			TÍTULO: Emgranaje conducido_H	
DIBUJ.	SANGUIRO		04/07/2024				
VERIF.	SANGUIRO		04/07/2024				
APROB.	SANGUIRO		04/07/2024				
FABR.	SANGUIRO		04/07/2024				
CALID.	SANGUIRO		04/07/2024	MATERIAL: PLA		N.º DE DIBUJO TFM-P006	A4
				PESO:		ESCALA:1:2	HOJA 8 DE 15



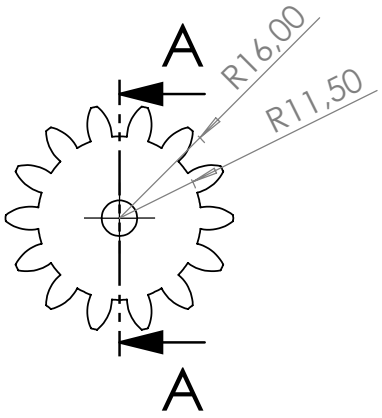


SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:					REBARBAR Y ROMPER ARISTAS VIVAS	NO CAMBIE LA ESCALA	REVISIÓN
	NOMBRE	FIRMA	FECHA			TÍTULO: Sujección patas	
DIBUJ.	SANGUIRO		04/07/2024				
VERIF.	SANGUIRO		04/07/2024				
APROB.	SANGUIRO		04/07/2024				
FABR.	SANGUIRO		04/07/2024				
CALID.	SANGUIRO		04/07/2024	MATERIAL: PLA		N.º DE DIBUJO TFM-P007	A4
				PESO:		ESCALA:1:2	HOJA 9 DE 15

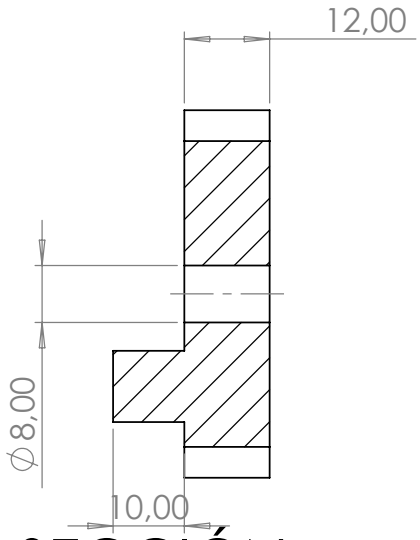
Modulo: 2
Ángulo de presión: 20º
Número de dientes: 14



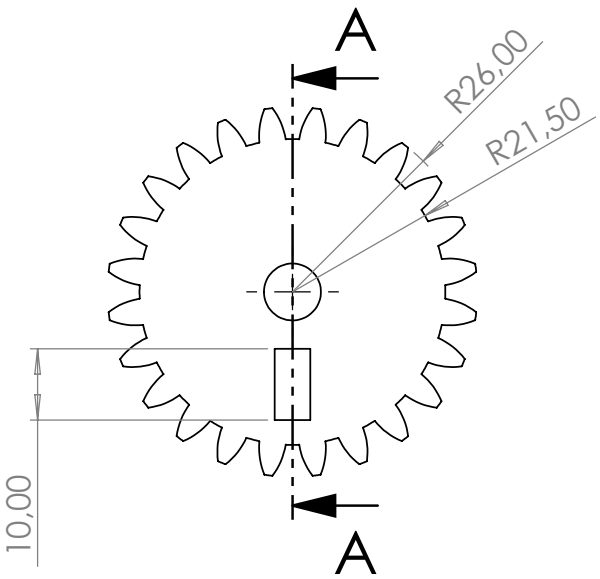
SECCIÓN A-A



SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:			REBARBAR Y ROMPER ARISTAS VIVAS	NO CAMBIE LA ESCALA	REVISIÓN		
	NOMBRE	FIRMA	FECHA	TÍTULO: Engranaje conductor_V			
DIBUJ.	SANGUIRO		04/07/2024				
VERIF.	SANGUIRO		04/07/2024				
APROB.	SANGUIRO		04/07/2024				
FABR.	SANGUIRO		04/07/2024				
CALID.	SANGUIRO		04/07/2024	MATERIAL: PLA	N.º DE DIBUJO TFM-P010	A4	
				PESO:	ESCALA:1:3	HOJA 10 DE 15	



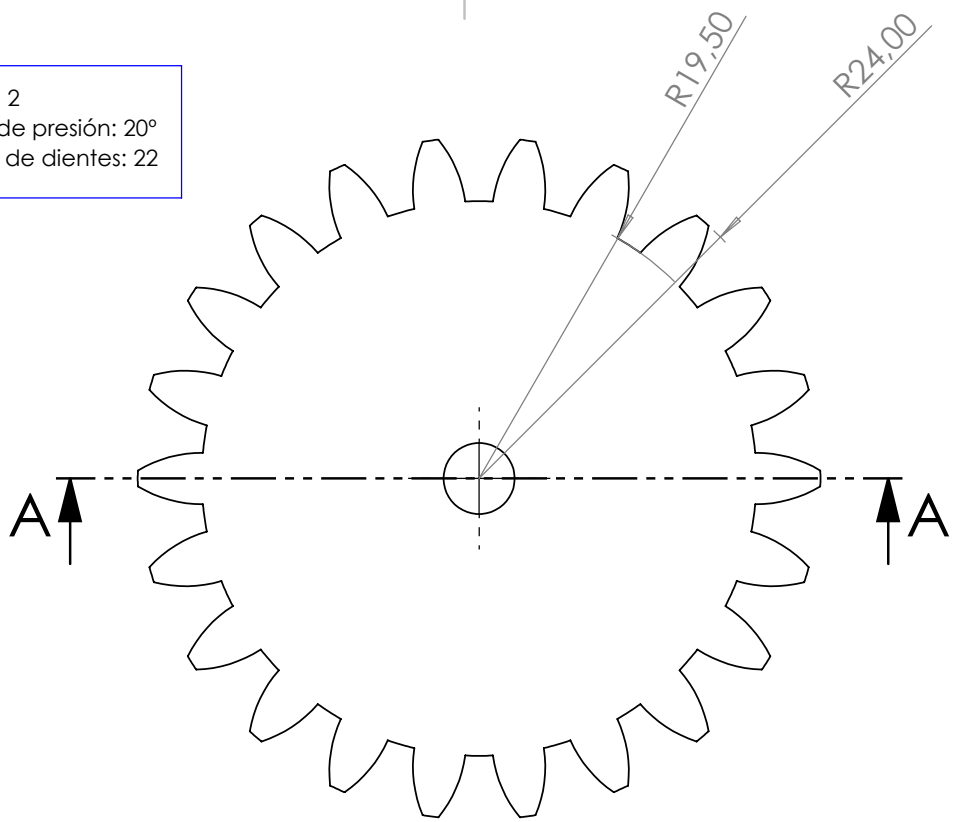
SECCIÓN A-A

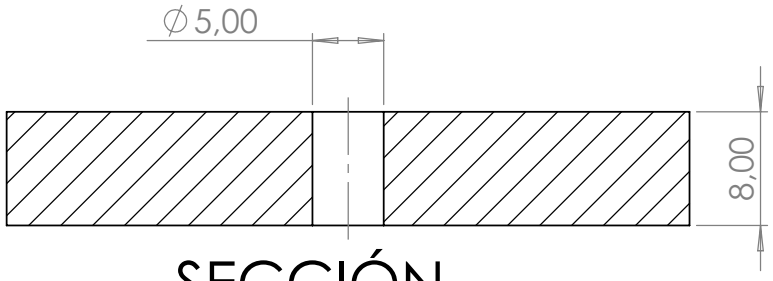


Modulo: 2
Ángulo de presión: 20°
Número de dientes: 24

SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:					REBARBAR Y ROMPER ARISTAS VIVAS	NO CAMBIE LA ESCALA	REVISIÓN
	NOMBRE	FIRMA	FECHA			TÍTULO: Engranaje conducido_V	
DIBUJ.	SANGUIRO		04/07/2024				
VERIF.	SANGUIRO		04/07/2024				
APROB.	SANGUIRO		04/07/2024				
FABR.	SANGUIRO		04/07/2024				
CALID.	SANGUIRO		04/07/2024	MATERIAL: PLA		N.º DE DIBUJO TFM-P011	A4
				PESO:		ESCALA:1:3	HOJA 11 DE 15

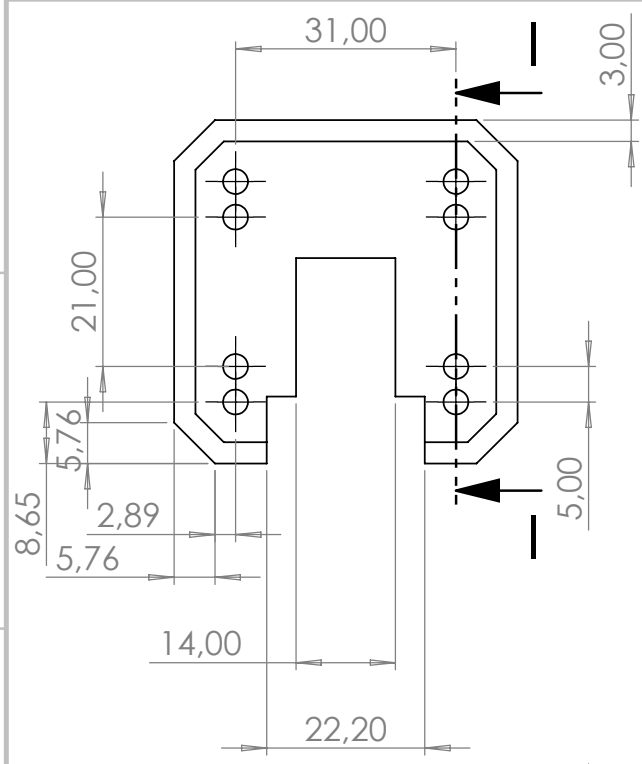
Modulo: 2
Ángulo de presión: 20º
Número de dientes: 22

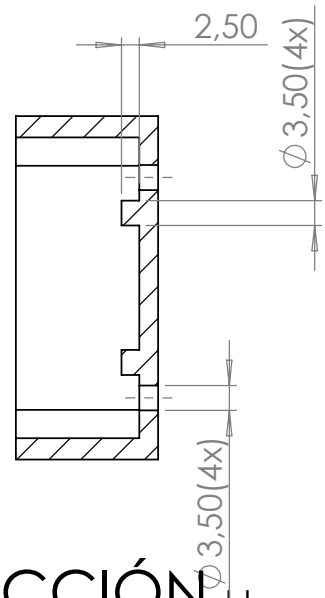




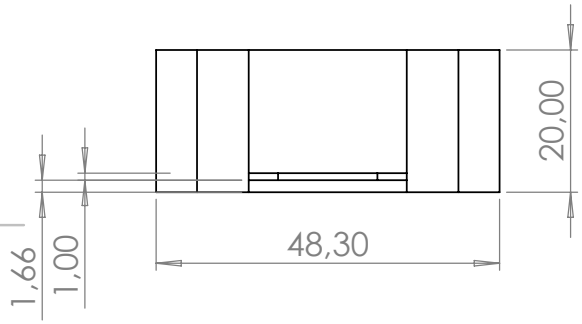
SECCIÓN A-A

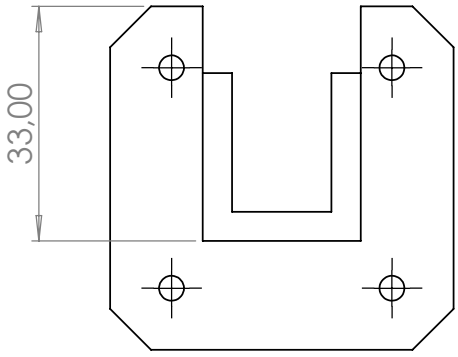
SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:				REBARBAR Y ROMPER ARISTAS VIVAS		NO CAMBIE LA ESCALA	REVISIÓN
	NOMBRE	FIRMA	FECHA			TÍTULO: Engranaje conductor_H	
DIBUJ.	SANGUIRO		04/07/2024				
VERIF.	SANGUIRO		04/07/2024				
APROB.	SANGUIRO		04/07/2024				
FABR.	SANGUIRO		04/07/2024				
CALID.	SANGUIRO		04/07/2024	MATERIAL: PLA		N.º DE DIBUJO TFM-P008	A4
				PESO:		ESCALA:2:1	HOJA 12 DE 15



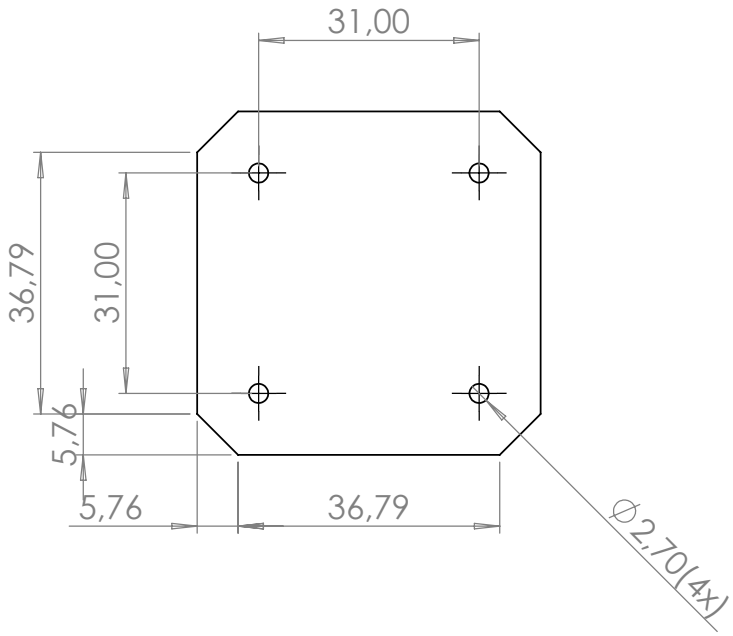


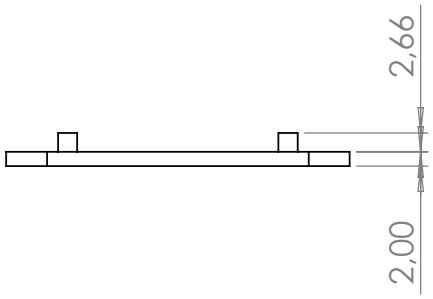
SECCIÓN I-I



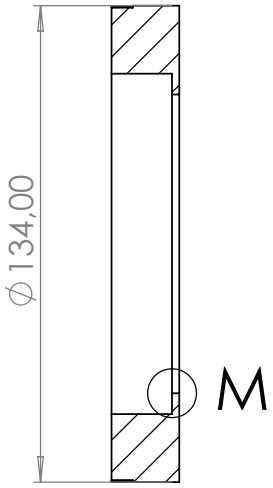


SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:					REBARBAR Y ROMPER ARISTAS VIVAS	NO CAMBIE LA ESCALA	REVISIÓN
	NOMBRE	FIRMA	FECHA			TÍTULO: Sujección encoder	
DIBUJ.	SANGUIRO		04/07/2024				
VERIF.	SANGUIRO		04/07/2024				
APROB.	SANGUIRO		04/07/2024				
FABR.	SANGUIRO		04/07/2024				
CALID.	SANGUIRO		04/07/2024	MATERIAL: PLA		N.º DE DIBUJO TFM-P013	A4
				PESO:		ESCALA:1:1	HOJA 13 DE 15

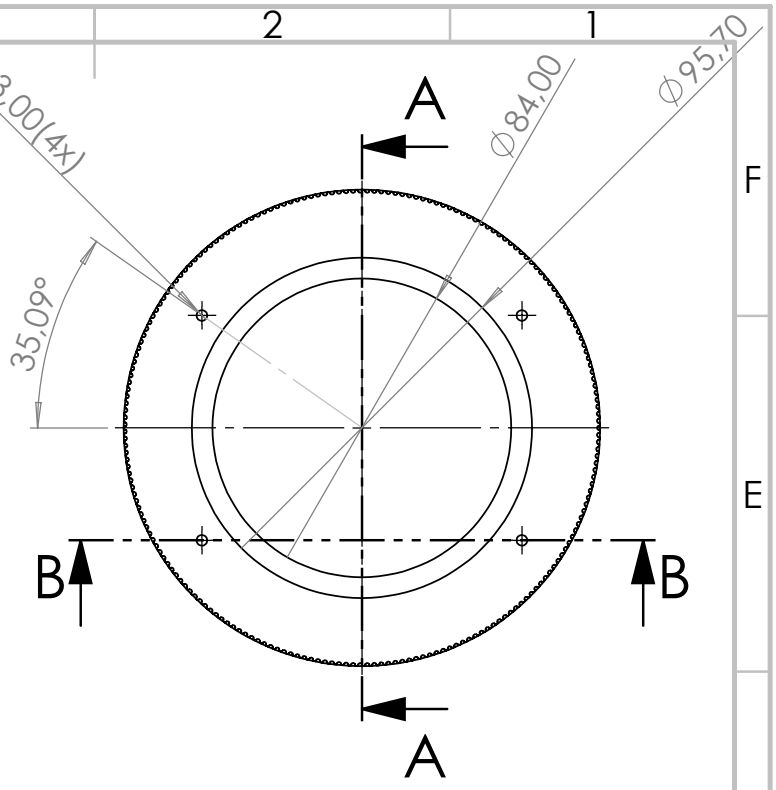


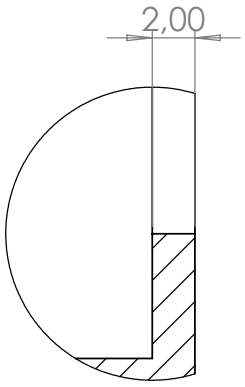


SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:					REBARBAR Y ROMPER ARISTAS VIVAS	NO CAMBIE LA ESCALA	REVISIÓN
	NOMBRE	FIRMA	FECHA			TÍTULO: Tapa encoder	
DIBUJ.	SANGUIRO		04/07/2024				
VERIF.	SANGUIRO		04/07/2024				
APROB.	SANGUIRO		04/07/2024				
FABR.	SANGUIRO		04/07/2024				
CALID.	SANGUIRO		04/07/2024	MATERIAL: PLA		N.º DE DIBUJO TFM-P014	A4
				PESO:		ESCALA:2:1	HOJA 14 DE 15



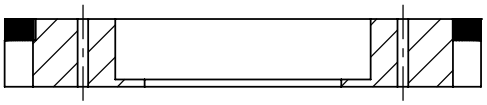
SECCIÓN A-A





DETALLE M

ESCALA 3 : 1

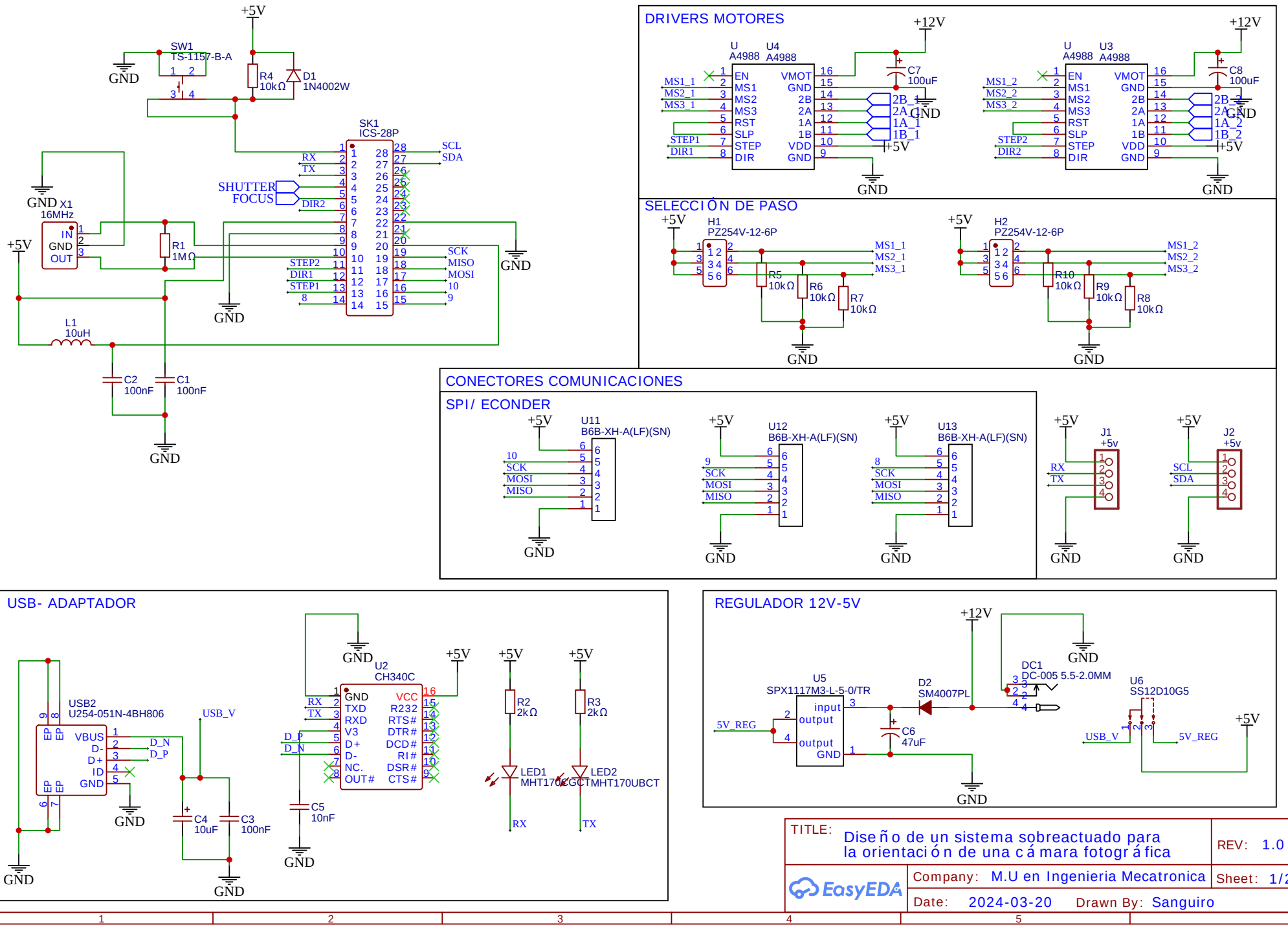


SECCIÓN B-B

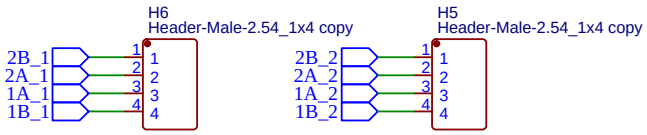
Tipo: GT-2
Número de dientes: 211

SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:					REBARBAR Y ROMPER ARISTAS VIVAS	NO CAMBIE LA ESCALA	REVISIÓN
	NOMBRE	FIRMA	FECHA			TÍTULO: Engranaje tipo GT-2	
DIBUJ.	SANGUIRO		04/07/2024				
VERIF.	SANGUIRO		04/07/2024				
APROB.	SANGUIRO		04/07/2024				
FABR.	SANGUIRO		04/07/2024				
CALID.	SANGUIRO		04/07/2024	MATERIAL: PLA		N.º DE DIBUJO TFM-P015	A4
				PESO:		ESCALA:1:2	HOJA 15 DE 15

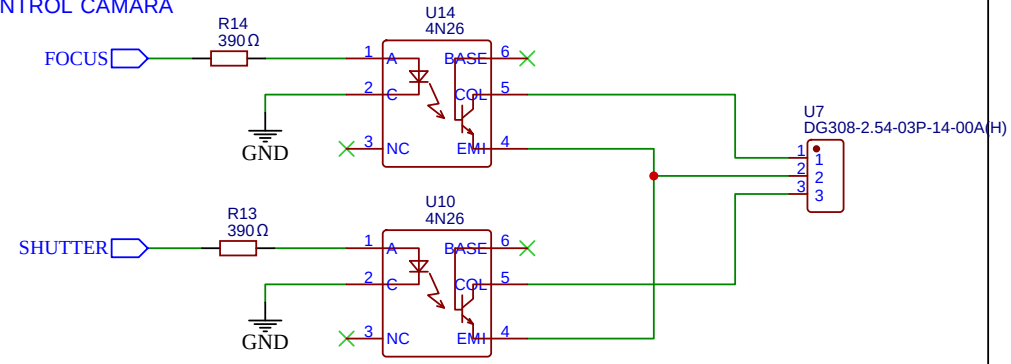
Esquemático


DRIVERS MOTORES
SELECCIÓN DE PASO
CONECTORES COMUNICACIONES
SPI/ ECONDER
USB- ADAPTADOR
REGULADOR 12V-5V
SW1
TS-1157-B-A
R4
10kΩ
D1
1N4002W
SK1
ICS-28P
SHUTTER
FOCUS
X1
16MHz
R1
1MΩ
L1
10uH
C2
100nF
C1
100nF
U4
A4988
U3
A4988
C7
100uF
C8
100uF
H1
PZ254V-12-6P
H2
PZ254V-12-6P
R5
10kΩ
R6
10kΩ
R7
10kΩ
R10
10kΩ
R9
10kΩ
R8
10kΩ
U11
B6B-XH-A(LF)(SN)
U12
B6B-XH-A(LF)(SN)
U13
B6B-XH-A(LF)(SN)
J1
J2
USB2
U254-051N-4BH806
U2
CH340C
C4
10uF
C3
100nF
C5
10nF
R2
2kΩ
R3
2kΩ
LED1
LED2
MHT170UBCT
U5
SPX1117M3-L-5-0/TR
D2
SM4007PL
DC1
DC-005 5.5-2.0MM
C6
47uF
U6
SS12D10G5
TITLE: Diseño de un sistema sobreactuado para la orientación de una cámara fotográfica
REV: 1.0
Company: M.U en Ingenieria Mecatronica
Sheet: 1/2
Date: 2024-03-20
Drawn By: Sanguiro
EasyEDA

CONECTORES MOTORES PASO A PASO



CONTROL CAMARA



TITLE: Diseño de un sistema sobreactuado para la orientación de una cámara fotográfica		REV: 1.0
EasyEDA	Company: M.U en Ingenieria Mecatronica	Sheet: 2/2
	Date: 2024-03-20 Drawn By: Sanguiro	

Documento:

Anexo

Índice

Código

```
#define CUSTOM_SETTINGS
#define INCLUDE_TERMINAL_MODULE
#define INCLUDE_GAMEPAD_MODULE
#include <Dabble.h>
#include <AS5048A.h>
#include <SoftwareSerial.h>
#include "Arduino.h"
#include "ControlST.h"

// Definir pines para SoftwareSerial
#define RX_PIN A5
#define TX_PIN A4

// Crear una instancia de SoftwareSerial
SoftwareSerial bluetoothSerial(RX_PIN, TX_PIN);

#define PIN_STEP_1 7
#define PIN_DIR_1 6
#define PIN_STEP_2 5
#define PIN_DIR_2 4

#define FOCUS 2 // Definir el pin para la salida digital
#define SHUTTER 3 // Definir el pin para la salida digital

// Crear instancias de AccelStepper para cada motor

ControlST motor1(PIN_STEP_1, PIN_DIR_1);
ControlST motor2(PIN_STEP_2, PIN_DIR_2);

// Instancias de AS5048A para dos encoders en pines diferentes
AS5048A encoder1(10); // El pin 10 para el primer encoder
AS5048A encoder2(9);  // El pin 11 para el segundo encoder

unsigned long tiempoAhora = 0;
unsigned long tiempoAnterior = 0;
const long intervalo = 2000; // Ajustar el intervalo a un valor más
   razonable

float Posicion1 = 0.0; // Señal de referencia para el primer motor
float Posicion2 = 0.0; // Señal de referencia para el segundo motor
float KP1 = 100;       // Ganancia proporcional
float KP2 = 50;        // Ganancia proporcional
float Error1 = 0.0;    // Señal de error para el primer motor
float Error2 = 0.0;    // Señal de error para el segundo motor
float Upk1 = 0.0;     // Acción proporcional para el primer motor
float Upk2 = 0.0;     // Acción proporcional para el segundo motor

float Velocidad = 0.0;

float posicionAbsoluta1;
float posicionAbsoluta2;

```

```
unsigned int control = 1;       //1=control de posición; 0=control de velocidad

float RT1 = 0.0947;     // Relación de transformación eje rotación
float RT2 = 0.5833;     // Relación de transformación eje cabeceo

// Variables globales para cada encoder
long vueltasCompletas1 = 0;
long offset1 = 0;
word valorAnterior1 = 0;

long vueltasCompletas2 = 0;
long offset2 = 0;
word valorAnterior2 = 0;

String Serialdata = "";
String Terminaldata = "";
bool dataflag = 0;

bool positionsReached = false;

// Variables para la función activarSalidas
bool activar = false;
bool espera = false;
unsigned long tiempoActivar = 0;
int repeticionesP = 0;
int contadorRepeticiones = 0;
unsigned long startTime = 0;
unsigned long tiempoAnteriorSalidas = 0;
int estadoSalidas = LOW; // Estado inicial de las salidas

unsigned long intervaloApagado; // 1 segundo

bool estado = false;

void setup() {
  Serial.begin(9600);             // Inicializar comunicación serie para el monitor serie
  bluetoothSerial.begin(9600);    // Inicializar comunicación Bluetooth

  Dabble.begin(bluetoothSerial);  // Iniciar Dabble usando SoftwareSerial

  pinMode(FOCUS, OUTPUT); // Configurar el pin digital como salida
  pinMode(SHUTTER, OUTPUT); // Configurar el pin digital como salida
  pinMode(8, OUTPUT);

  encoder1.init(); // Inicializar el primer encoder
  encoder2.init(); // Inicializar el segundo encoder
  delay(100);      // Pequeño retraso para asegurar que los encoders estén listos
```

```
99
100   valorAnterior1 = encoder1.getRawRotation(); // Establecer la lectura
        inicial para el primer encoder
101   valorAnterior2 = encoder2.getRawRotation(); // Establecer la lectura
        inicial para el segundo encoder
102
103   getPosicionAbsoluta(encoder1, true, vueltasCompletas1, offset1,
       valorAnterior1);
104   getPosicionAbsoluta(encoder2, true, vueltasCompletas2, offset2,
       valorAnterior2);
105 }
106
107 void loop() {
108   estado = !estado;  // Cambia el estado (de verdadero a falso, o de
       falso a verdadero)
109   digitalWrite(8, estado);
110   Dabble.processInput(); // Refrescar los datos obtenidos del
       smartphone
111
112   // Obtener el valor de posición del encoder
113   posicionAbsoluta1 = RT1 * getPosicionAbsoluta(encoder1, false,
       vueltasCompletas1, offset1, valorAnterior1);
114   posicionAbsoluta2 = RT2 * getPosicionAbsoluta(encoder2, false,
       vueltasCompletas2, offset2, valorAnterior2);
115
116   // Lectura de datos desde el monitor serie
117   while (Serial.available() > 0) {
118     char c = Serial.read();
119     Serialdata += c;
120     if (c == '\n') {
121       Serialdata.trim(); // Recorta los espacios en blanco
122       processCommand(Serialdata); // Procesar comandos recibidos desde
        el monitor serie
123       Serialdata = ""; // Limpiar el buffer después de procesar el
      comando
124     }
125   }
126
127   // Lectura de datos desde el Terminal y envío al monitor serie
128   if (Terminal.available()) {
129     while (Terminal.available() != 0) {
130       char c = Terminal.read();
131       Terminaldata += c;
132       Serial.write(c); // Envía el carácter al monitor serie
133     }
134     Serial.println();
135     Terminaldata.trim(); // Recorta los espacios en blanco
136     processCommand(Terminaldata); // Procesar comandos recibidos desde
        el terminal
137     Terminaldata = ""; // Limpiar el buffer después de procesar el
      comando
138   }
```

```
139
140   float manualUpk1 = 0.0;
141   float manualUpk2 = 0.0;
142
143   // Controlar motores basado en la entrada del gamepad
144   if (GamePad.isRightPressed()) {
145     Posicion1 = posicionAbsoluta1;
146     manualUpk1 = -500;
147   }
148   if (GamePad.isLeftPressed()) {
149     Posicion1 = posicionAbsoluta1;
150     manualUpk1 = 500;
151   }
152   if (GamePad.isUpPressed()) {
153     Posicion2 = posicionAbsoluta2;
154     manualUpk2 = -200;
155   }
156   if (GamePad.isDownPressed()) {
157     Posicion2 = posicionAbsoluta2;
158     manualUpk2 = 200;
159   }
160
161   // Activar la salida digital cuando se presione el botón cuadrado
162   if (GamePad.isSquarePressed()) {
163     digitalWrite(FOCUS, HIGH); // Activar salida digital
164     delay(250);
165     digitalWrite(SHUTTER, HIGH); // Activar salida digital
166     delay(250);
167     digitalWrite(FOCUS, LOW); // Desactivar salida digital
168     digitalWrite(SHUTTER, LOW); // Desactivar salida digital
169   }
170
171   // Poner a cero la referencia
172   if (GamePad.isCirclePressed()){
173     posicionAbsoluta1 = RT1 * getPosicionAbsoluta(encoder1, true,
      vueltasCompletas1, offset1, valorAnterior1);
174     posicionAbsoluta2 = RT2 * getPosicionAbsoluta(encoder2, true,
      vueltasCompletas2, offset2, valorAnterior2);
175     posicionAbsoluta1 = RT1 * getPosicionAbsoluta(encoder1, false,
      vueltasCompletas1, offset1, valorAnterior1);
176     posicionAbsoluta2 = RT2 * getPosicionAbsoluta(encoder2, false,
      vueltasCompletas2, offset2, valorAnterior2);
177     Posicion1 = posicionAbsoluta1;
178     Posicion2 = posicionAbsoluta2;
179   }
180
181   //================control por posición=====================
182
183   // Regulador P para el primer motor
184   Error1 = Posicion1 - posicionAbsoluta1;
185   if (control == 1){
186     if (abs(Error1) < 1) {
```

```
    Upk1 = 0; // Acción proporcional
   } else {
    Upk1 = (Error1 * KP1) * (-1); // Acción proporcional
    if (Upk1 > 0) {
      if (Upk1 < 200) {
        Upk1 = 200; // Asegurar que Upk1 no sea menor a 100 en positivo
      }
    }
    if (Upk1 < 0) {
      if (Upk1 > -200) {
        Upk1 = -200; // Asegurar que Upk1 no sea menor a 100 en negativo
      }
    }
   }
}

// Regulador P para el segundo motor
Error2 = Posicion2 - posicionAbsoluta2;
if (abs(Error2) < 1) {
  Upk2 = 0; // Acción proporcional
} else {
  Upk2 = (Error2 * KP2) * (-1); // Acción proporcional
  if (Upk2 > 0) {
    if (Upk2 < 200) {
      Upk2 = 200; // Asegurar que Upk2 no sea menor a 100 en positivo
    }
  }
  if (Upk2 < 0) {
    if (Upk2 > -200) {
      Upk2 = -200; // Asegurar que Upk2 no sea menor a 100 en negativo
    }
  }
}

// Enviar "R" si ambos errores son menores de 2
if (abs(Error1) < 1 && abs(Error2) < 1 && !positionsReached) {
  Serial.println("R");
  positionsReached = true;
}

motor1.velocidad(Upk1 + manualUpk1 + Velocidad);
motor1.marcha();

motor2.velocidad(Upk2 + manualUpk2);
motor2.marcha();

tiempoAhora = millis();

```

```
  // Gestionar la activación de salidas sin bloquear
  if (espera) {
    intervaloApagado = 1000; // 1 segundo de espera entre repeticiones
  } else {
    intervaloApagado = tiempoActivar + tiempoActivar; // Tiempo de
   apagado como 1.5 veces el tiempo de activación
  }
  if (activar) {
    unsigned long tiempoActual = millis(); // Obtener el tiempo actual

    if (repeticionesP > contadorRepeticiones) {

      if (estadoSalidas == LOW) {
        if (tiempoActual - tiempoAnteriorSalidas >= intervaloApagado){

          espera = false;
          estadoSalidas = HIGH; // Cambiar el estado a encendido

          digitalWrite(FOCUS, HIGH); // Encender la salida
          Serial.println("FOCUS");
          delay(200);
          digitalWrite(SHUTTER, HIGH); // Encender la salida

          tiempoAnteriorSalidas = millis();
          Serial.println("Encendido");
        }

      } else if (estadoSalidas == HIGH) {
        if (tiempoActual - tiempoAnteriorSalidas >= tiempoActivar) {

          estadoSalidas = LOW; // Cambiar el estado a apagado

          digitalWrite(FOCUS, estadoSalidas); // Apagar la salida
          digitalWrite(SHUTTER, estadoSalidas); // Apagar la salida
          Serial.println("Apagado");
          contadorRepeticiones++; // Incrementar el contador de ciclos
          espera = true; // Indicar que estamos en período de espera
          tiempoAnteriorSalidas = millis();
        }
      }
    } else {
      activar = false; // Desactivar el proceso después de completar
   las repeticiones
      Serial.println("Desactivar");
    }
  }
}

void processCommand(String command) {
  command.trim(); // Eliminar espacios en blanco al principio y al
   final
  if (command == "S") {
```

```
284     digitalWrite(FOCUS, HIGH); // Activar salida digital
285     delay(500);
286     digitalWrite(SHUTTER, HIGH); // Activar salida digital
287     delay(200);
288     digitalWrite(SHUTTER, LOW);  // Desactivar salida SHUTTER
289     digitalWrite(FOCUS, LOW);
290   } else if ((command == "A")) {
291     posicionAbsoluta1 = RT1 * getPosicionAbsoluta(encoder1, true,
      vueltasCompletas1, offset1, valorAnterior1);
292     posicionAbsoluta2 = RT2 * getPosicionAbsoluta(encoder2, true,
      vueltasCompletas2, offset2, valorAnterior2);
293     posicionAbsoluta1 = RT1 * getPosicionAbsoluta(encoder1, false,
      vueltasCompletas1, offset1, valorAnterior1);
294     posicionAbsoluta2 = RT2 * getPosicionAbsoluta(encoder2, false,
      vueltasCompletas2, offset2, valorAnterior2);
295     Posicion1 = posicionAbsoluta1;
296     Posicion2 = posicionAbsoluta2;
297   } else if (command.startsWith("V ")) {
298     Velocidad = command.substring(2).toFloat();
299     Serial.print("Velocidad: ");
300     Serial.println(Velocidad);
301     if (Velocidad != 0){
302       control = 0;
303     } else if (Velocidad == 0){
304       Posicion1 = posicionAbsoluta1;
305       Velocidad = 0;
306       control =1;
307     }
308   } else if (command.startsWith("T ")) {
309     int commaIndex = command.indexOf(',');
310     if (commaIndex != -1) {
311       int tiempo = command.substring(2, commaIndex).toInt();
312       int repeticiones = command.substring(commaIndex + 1).toInt();
313       activar = true;
314       espera = true;
315       repeticionesP = repeticiones;
316       tiempoActivar = (unsigned long)tiempo * 1000;
317       Serial.print("Tiempo Activar: ");
318       Serial.println(tiempoActivar); // Debug: Imprimir tiempoActivar
319       contadorRepeticiones = 0;
320       digitalWrite(FOCUS, LOW); // Asegurar que las salidas estén
      apagadas al inicio
321       digitalWrite(SHUTTER, LOW); // Asegurar que las salidas estén
      apagadas al inicio
322       tiempoAnteriorSalidas = millis(); // Reiniciar el tiempo
      anterior
323       Serial.println("Activar");
324     } else {
325       Serial.println("Comando T mal formado. Use el formato T tiempo,
      repeticiones");
326     }
327   } else if (command.startsWith("KP1 ")) {
```

```
    KP1 = command.substring(4).toFloat();
    Serial.print("KP1 actualizado a: ");
    Serial.println(KP1);
  } else if (command.startsWith("KP2 ")) {
    KP2 = command.substring(4).toFloat();
    Serial.print("KP2 actualizado a: ");
    Serial.println(KP2);
  } else {
    int commaIndex = command.indexOf(',');
    if (commaIndex != -1) {
      String pos1Str = command.substring(0, commaIndex);
      String pos2Str = command.substring(commaIndex + 1);
      Posicion1 = pos1Str.toFloat();
      Posicion2 = pos2Str.toFloat();
      Serial.print("Posicion1 actualizada a: ");
      Serial.println(Posicion1);
      Serial.print("Posicion2 actualizada a: ");
      Serial.println(Posicion2);
      positionsReached = false; // Reset position reached flag
    } else {
      Serial.println("Comando no reconocido");
    }
  }
}

float getPosicionAbsoluta(AS5048A &encoder, bool resetear, long &
    vueltasCompletas, long &offset, word &valorAnterior) {
  word valorActual = encoder.getRawRotation();

  if (resetear) {
    vueltasCompletas = 0;
    offset = map(valorActual, 0, 16383, 0, 3600) / 10; // Modificar
    aqui cambiar el offset por valorActual = encoder.setZeroPosition();
    valorAnterior = valorActual;
    return 0.0;
  }

  int diferencia = int(valorActual) - int(valorAnterior);

  if (diferencia > 8192) {
    vueltasCompletas -= 1;
  } else if (diferencia < -8192) {
    vueltasCompletas += 1;
  }

  float posicionAbsoluta = 360.0 * vueltasCompletas + (map(valorActual
    , 0, 16383, 0, 3600) / 10.0) - offset;
  valorAnterior = valorActual;
  return posicionAbsoluta;
}
```