UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

Diseño e implementación de un sistema de enfoque remoto para telescopio basado en microcontrolador.

Trabajo Fin de Grado

Grado en Ingeniería Informática

AUTOR/A: Landete Exposito, David

Tutor/a: Pérez Jiménez, Alberto José

CURSO ACADÉMICO: 2021/2022

Agradecimientos

A mi tutor Alberto Pérez Jiménez por la ayuda aportada, así como su experiencia para apoyar en que sea más sencilla la creación del proyecto con sus consejos.

Del mismo modo, a mi pareja por brindarme su apoyo en todo momento en que el trabajo se realizaría de una forma adecuada, así como a mi familia por apoyarme desde que comencé el grado.

Resumen

Hoy en día, dada la utilización del internet de las cosas, existe una gran cantidad de proyectos en los cuales se utilizan las automatizaciones, las conexiones entre distintos proyectos o entornos y, por último, proyectos a menor escala en los que se utilizan microcontroladores como Arduino o ESP. Es por esto por lo que, dada la importancia de los sensores empleados para generar los datos que interpretará el programa, la utilización de los sensores de manera correcta, el correcto funcionamiento de éstos y las conexiones entre entornos o proyectos, favorecen a que, actualmente, los datos que se generan para la interpretación y toma de decisiones en un proyecto, jueguen un papel importante.

En este trabajo de fin de grado se ha desarrollado un programa para el sistema de enfoque de un telescopio utilizando un microcontrolador Arduino NANO, con lo que se podrá realizar un uso del sistema de enfoque sin tener que mover manualmente la rueda del telescopio. Con esto se conseguiría un enfoque más preciso y la posibilidad de manejarlo de manera remota.

Para el diseño del controlador se ha diseñado una serie de piezas que servirán como soporte a los pulsadores, pantalla OLED, Arduino y, por último, el motor paso a paso empleado. Así como también se ha desarrollado el software teniendo unos objetivos de cara al uso de las personas para que se pueda emplear del modo más sencillo cumpliendo con los objetivos.

Palabras clave: enfocador, telescopio, Arduino, ESP, motor paso a paso, OLED.

Abstract

Today, given the use of the Internet of Things, there are many projects in which automations are used, connections between different projects or environments, and smaller-scale projects in which microcontrollers such as Arduino or ESP are used. Therefore, given the importance of the sensors used to generate data that the program will interpret, the correct use of the sensors, their correct operation and the connections between environments or projects, favor that, currently, the data that are generated for interpretation and decision making in a project, play an important role.

In this end-of-degree project, a program for a focusing system of a telescope has been developed using an Arduino NANO microcontroller, with which the focus system can be used without having to manually move the telescope wheel. This would achieve a more precise focus and the possibility of managing it remotely.

For the design of the controller, a series of pieces have been designed that will serve as support for the push buttons, OLED screen, Arduino and, finally, the stepper motor used. As well as the software has been developed having some objectives for the use of people so that it can be used in the simplest way fulfilling the objectives.

Keywords: focuser, telescope, Arduino, ESP, stepper motor, OLED.

Índice general

Índice de ilustraciones

Índice de tablas

1. Introducción

Dada la gran cantidad de avances tecnológicos que se dan en la actualidad en la sociedad, la creación de un proyecto es una tarea más sencilla de la que muchas personas se podrían llegar a imaginar. En muchos casos, es tan sencillo como que la imaginación de una persona comience a idear un proyecto para el cual, realizando una rápida búsqueda por internet, se pueda comenzar a desarrollar éste.

Hoy en día, existe un porcentaje muy alto de personas que disponen de un *smartphone* y/o ordenador en casa. Cualquiera de estas personas, en mayor o menor medida, utiliza las automatizaciones, éstas pueden ser de procesos tan sencillos como enviar un mensaje por una aplicación o, incluso gente con más experiencia en el ámbito tecnológico, utilizando un asistente inteligente como lo son Alexa o Siri. Con esto son capaces de realizar un proyecto en el cual son capaces de conectar una gran variedad de dispositivos para controlarlos con comandos de voz o mediante la aplicación de móvil.

Actualmente, para comenzar un proyecto e investigar si ya existe o, por el contrario, si existe uno parecido con el que poder comenzar a tomar ideas, nos es muy sencillo dada la existencia de un buscador de internet tan potente como lo es Google. Por ello, la sociedad de hoy en día tiende a sentir la necesidad de buscar todo por internet cuando no se conoce sobre un tema. Esto puede tener un punto muy negativo y es la variedad de páginas existentes. Una página podría no tener una fuente fiable, por lo que no será un buen lugar en el que buscar este conocimiento.

Si, bien por el contrario, existe una cantidad todavía mayor de páginas en las cuales obtener información cuya fuente origen es más fiable, la información adecuada se encuentra al alcance de unos pocos *clicks* del *smatphone* u ordenador. Asimismo, la facilidad para encontrar información no significa que cualquier persona pueda realizar cualquier proyecto. Para poder darse el caso, la persona en cuestión deberá ser capaz de conseguir una formación adecuada de aquello que le gustaría realizar. Una persona la cual no tiene conocimientos de tecnología web, podrá crear una página muy sencilla utilizando una plantilla o una página existente que da la posibilidad de crear una página web propia. Esta misma persona, una vez transcurrido una cierta cantidad de tiempo, durante el cual ha ido obteniendo conocimientos acerca de las tecnologías web y cómo desarrollar una página podrá, ahora sí, crear su propio proyecto con la calidad que esta persona desearía.

Por todo ello, se ha llegado a la conclusión de que, desarrollar un proyecto como este, es un reto, puesto que se dispone de conocimiento para poder abordar el proyecto, pero, por otro lado, se carece de una información más técnica en otros apartados. Es por esto por lo que, siguiendo el

ejemplo anteriormente comentado, se ha realizado una tarea de investigación para poder llevar a cabo el proyecto con la calidad deseada y un correcto funcionamiento.

En este trabajo, se ha realizado un trabajo de búsqueda de información en el cual, partiendo de una base de conocimiento previa, se ha obtenido el proyecto final. El resultado del proyecto, desde un punto inicial, se ha abordado para obtener un resultado con el nivel de calidad deseado

El proyecto inicialmente parte como la propuesta del profesor, la cual el alumno la acepta dado el interés mostrado acerca del microcontrolador Arduino. Finalmente, dado el interés, la propuesta aceptada y el conocimiento previo, se ha realizado este trabajo, el cual se detallará más adelante el proceso de desarrollo y montaje.

1.1. Motivación

Para realizar este proyecto, la motivación principal es que, partiendo de una idea base, se añade la posibilidad de realizar el proyecto por uno mismo. Actualmente existe una tendencia conocida como *"do it yourself",* mediante la cual, las personas realizan las cosas por sí mismas. Este movimiento aparece ya que, visualizando un vídeo explicativo o leyendo un artículo donde se detallan los pasos, la gente es capaz de realizar las tareas por sí mismas.

En este caso, se trata de la posibilidad de realizar un proyecto, un sistema de enfoque para un telescopio empleando un microcontrolador, utilizando el conocimiento aprendido previamente.

No obstante, disponiendo de conocimientos y material, la solución por la que se ha optado es la de realizar el proyecto desde un comienzo, sin utilizar modelos, código o cualquier recurso disponible en estos foros para su uso en el proyecto. Este proyecto pretende ser realizado únicamente con los conocimientos previos, obtenidos en la titulación en asignaturas como por ejemplo programación o tecnología de computadores, así como en etapas anteriores de los estudios; y un trabajo de investigación mediante el cual se obtendrá el conocimiento faltante para emplear dispositivos tales como el motor paso a paso o la pantalla OLED de la forma correcta, obteniendo un resultado con la calidad deseada.

La idea inicial comienza con la propuesta del profesor, en este caso, el tutor del trabajo. La propuesta inicial es aceptada, ya que encaja dentro de los intereses del alumno, por lo que la motivación inicial del proyecto comienza siendo elevada.

1.2. Objetivos

Para la realización de este proyecto se han planteado una serie de objetivos, dadas las necesidades encontradas para el correcto uso de la solución propuesta.

En primer lugar, crear un programa capaz de poderse implantar en un Arduino NANO, teniendo en cuenta las limitaciones acerca de la memoria disponible.

Además de esto, crear un programa el cual sea fácil de utilizar por cualquier persona para utilizar el enfoque del telescopio sin tener que recurrir al uso manual del enfoque. Deberá tener una interfaz sencilla y entendible para que no aparezca ningún tipo de dificultad de uso ni confusiones en las opciones.

También se añade como objetivo, desarrollar una solución para el uso desde un ordenador conectando el microcontrolador Arduino mediante USB. Esto es debido a que el microcontrolador empleado no dispone de módulo WI-FI ni bluetooth. Del mismo modo, se puede realizar un uso a distancia en caso de que el ordenador esté conectado a internet, con lo que se permite realizar una mayor automatización.

Por último, diseñar las piezas necesarias para el sistema en 3D para su posterior impresión y uso. Las piezas deberán ser de las medidas necesarias para que encajen de forma correcta, sin holguras excesivas y, a su vez, sin ser tan justo que no sea posible su acople.

1.3. Expectativas

En este trabajo las expectativas se basan en la capacidad de aplicar adecuadamente los conocimientos previos, los adquiridos y los conocimientos nuevos adquiridos tras una tarea de investigación para la realización del proyecto.

No solo se trata de realizar un proyecto el cual sea funcional, también deberá ser un proyecto atractivo para quienes disponen de un telescopio y deseen emplear este proyecto. Bajo esta premisa, se ha realizado el diseño de los soportes para el motor y el cuerpo central.

En el caso de que una persona sea consciente del proyecto, compruebe que el funcionamiento es el deseado, comprobará la posibilidad de implantarlo en su propio telescopio. Es por eso por lo que partiendo de esta premisa se ha creado un sistema de enfoque capaz de ser implantado en una gran cantidad de telescopios.

Para poder llevar a cabo este deseo se ha realizado un diseño con unos detalles pequeños en los que se puede apreciar la posibilidad de ajustarse a multitud de telescopios, como lo son los orificios para poder mover el motor si se desea más arriba o más abajo, o las

aberturas para la sujeción en el telescopio. Estos agujeros de sujeción servirán para pasar a través una cuerda, una brida o cualquier elemento que se desee emplear para la fijación al telescopio.

Por otra parte, se desea realizar un sistema que sea fácil de utilizar. Para esto se desarrollarán una serie de versiones, en las que se irán haciendo progresos para descubrir los intereses del usuario final para, de esta forma, realizar un sistema lo más amigable y sencillo de usar.

1.4. Estructura

En este primer apartado se realiza una introducción, hablando acerca de la motivación por la que se ha decidido realizar este proyecto, los objetivos que se desean alcanzar tras la elaboración del proyecto y los resultados que se esperan obtener.

En el segundo apartado se realiza una comparación entre distintos microcontroladores, así como se explica la decisión para escoger el empleado en el proyecto.

En el tercer apartado se explican los componentes empleados, su función y cómo se han utilizado para completar el proyecto final, combinando todos los componentes.

En el cuarto apartado, se detalla la preparación realizada para el comienzo del proyecto, así como las pruebas realizadas para comprobar el correcto funcionamiento de los componentes.

En el quinto apartado, se describe el diseño que se ha realizado para el sistema, así como el diseño de su sistema eléctrico, como los diseños en 3D para las piezas deseadas.

En el sexto apartado, se describe el proceso de construcción del proyecto final, uniendo todas las piezas empleadas.

El séptimo apartado, es donde se explica el funcionamiento del programa, así como la inclusión de diagramas de flujo para representarse de una mejor forma.

En el octavo apartado, se detallan las conclusiones y los proyectos futuros con los que se podría mejorar el proyecto actual, explicando diversas opciones para ello.

El noveno apartado corresponde a las referencias utilizadas y, por último, el décimo apartado son los anexos, donde se encuentra el código fuente para la placa Arduino.

2. Comparando microcontroladores

Actualmente existe una gran variedad de microcontroladores, cada uno con unas características determinadas que lo diferencia del resto. Es por esto por lo que, se va a realizar una comparativa entre una serie de modelos de microcontroladores Arduino [1].

En primer lugar, aparece el Arduino UNO, probablemente el más utilizado hoy en día. Esto es debido a que una gran cantidad de gente que no tiene conocimientos de programación de Arduino adquiere por internet un *kit* en el cual se incluye un Arduino UNO, diodos LED, pulsadores, diversos sensores y una guía de programación para los elementos que incluye.

En segundo lugar, un microcontrolador casi idéntico al mencionado anteriormente, el Arduino UNO WiFi. Este microcontrolador posee unas características muy similares a las de la versión UNO. Las diferencias se encuentran en el microcontrolador empleado, siendo mejor el del modelo WiFi. La diferencia más importante que podría decantar a cualquier persona encontrándose ante la duda de adquirir uno o UNO WiFi es la conectividad WiFi. Esta conectividad puede ser un punto a favor para ampliar la posibilidad de crear proyectos más complejos en los que se requiera comunicación vía WiFi.

En tercer lugar, se encuentra el Arduino MEGA, un microcontrolador menos utilizado puesto que el tamaño es superior al del resto de los microcontroladores. Asimismo, es el microcontrolador que cuenta con la mayor cantidad de pines de entrada y salida, por lo que se recomienda para proyectos más complejos en los que se necesite una gran cantidad de pines a emplear.

En cuarto lugar, aparece el Arduino MICRO, in microcontrolador de tamaño reducido, pese a lo cual, conserva la misma cantidad de memoria *flash* como *SRAM* que el modelo UNO. Este modelo podría ser el ideal para las personas que van a realizar un proyecto simple en el cual no se utiliza un número elevado de elementos. Un proyecto en el que se utilice un interruptor o pulsador, una serie de LEDs o, incluso, un motor.

Por último, aparece el Arduino NANO, en este caso es el microcontrolador escogido. Este último empleado en la comparación es el segundo más pequeño de la familia Arduino, siendo el más pequeño el Arduino MINI. En este caso, el Arduino NANO, se trata de un microcontrolador de un tamaño ligeramente inferior al MICRO, con una cantidad de pines inferior, sin ser una diferencia amplia a favor del MICRO.

El microcontrolador escogido se trata del Arduino NANO ya que, se trata de un microcontrolador con un tamaño reducido y una cantidad de pines de entrada y salida suficiente como para un proyecto en el que se utiliza una pantalla, un motor paso a paso y 3 pulsadores.

Además de esto, se ha escogido el Arduino NANO puesto que, dadas las características que tiene, es la placa con un precio más reducido, con lo que se consigue abaratar el coste frente a un proyecto de iguales características en el que se utiliza un Arduino UNO, por ejmplo.

NOMBRE	**ARDUINO UNO**	**ARDUINO UNO WIFI**	**ARDUINO MEGA**	**ARDUINO MICRO**	**ARDUINO NANO**
Microcontrolador	ATmega328	ATmega4809	ATmega2560	ATmega32u4	ATmega328P
GPIO(PWM)	14 (6)	14 (5)	54 (15)	14 (7)	14 (6)
Entradas analógicas	6	6	16	12	8
Salidas analógicas	0	0	0	0	0
Memoria flash [kB]	32	48	256	32	32
SRAM [kB]	2	6	8	2	2
Puerto USB	B	Micro B	B/A	Micro B	Mini B

Tabla 1 - Comparativa entre placas Arduino

3. Enfocador y sus elementos

3.1. Pulsadores

Para poder realizar las acciones desde el Arduino en modo autónomo, sin conectar al ordenador, se emplean 3 pulsadores redondos. Cada uno de estos pulsadores tiene una única opción: subir, bajar y aceptar.

Para conectar los pulsadores con el microcontrolador se ha utilizado un sistema con una resistencia PULL-DOWN de 5'1KΩ. Esto es debido a que, pese a que el microcontrolador posee una resistencia PULL-UP interna, resulta más efectivo utilizar una externa de este modo. Las resistencias se pueden obtener de forma sencilla en cualquier tienda de electrónica, no es común disponer de resistencias con estos valores.

Utilizando el sistema de resistencias PULL-DOWN simplificamos las lecturas en el pin de entrada, siendo el valor para leer "alto", o activado.

Ilustración 1 - Pulsadores circulares

3.2. Motor paso a paso

Para dotar al sistema de la capacidad de giro en la rueda de enfoque, se utiliza un motor paso a paso 28BYJ-48 [2], un motor paso a paso unipolar.

Utilizando un motor paso a paso obtendremos un giro más preciso, puesto que, para que un motor paso a paso de una vuelta completa, éste debe completar 64 pasos, 5'625º. Dado que el motor dispone de una reductora con relación 1/64, los pasos totales para dar una vuelta completa serán 4096 pasos.

El movimiento del motor es posible realizarlo en una fase, dos fases o medios pasos. Cada tipo tiene un funcionamiento distinto en cuanto a las bobinas que se activan.

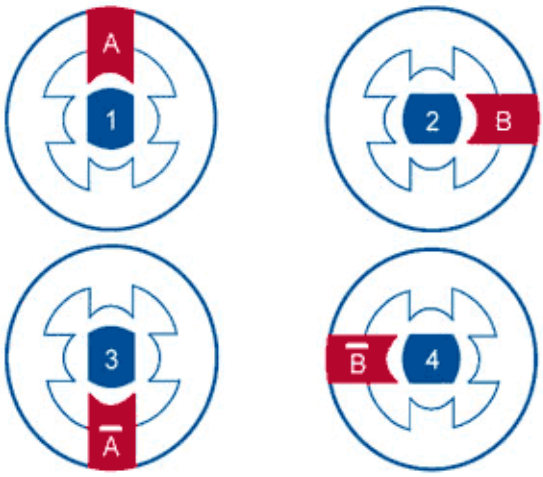

Ilustración 2 - Funcionamiento motor paso a paso en secuencia de una fase [2]

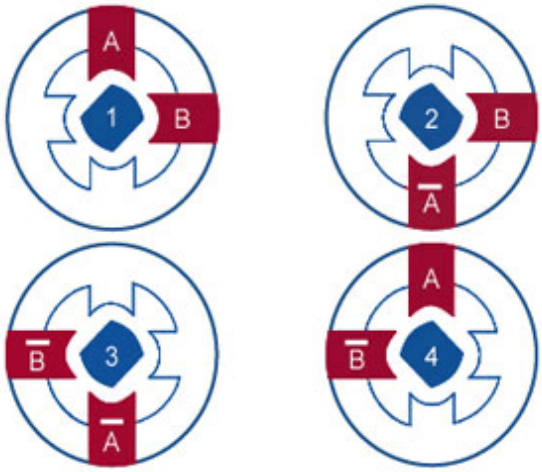

Ilustración 3 - Funcionamiento motor paso a paso en secuencia de dos fases [2]

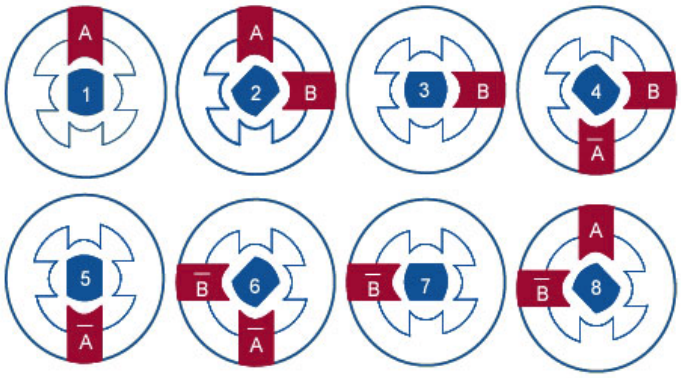

Ilustración 4 - Funcionamiento motor paso a paso en secuencia de medios pasos [2]

3.3. Módulo controlador de motores ULN2003

Para realizar la conexión del microcontrolador con el motor paso a paso bipolar se ha utilizado un módulo controlador de motores ULN2003. Mediante este módulo se podrá controlar el giro del motor controlando los campos magnéticos que se activan para provocar el giro.

En este caso, el módulo opera a un voltaje de 5V con una capacidad para manejar 500mA, unos niveles apropiados para utilizar con el microcontrolador seleccionado, puesto que se conecta a la salida de 5V estables del Arduino NANO.

El módulo es capaz de operar con 7 salidas. Para el proyecto se utilizan únicamente 4, esto es debido al funcionamiento interno del motor paso a paso para la activación de los campos magnéticos.

Ilustración 5 - Módulo ULN2003

3.4. Arduino NANO

El microcontrolador Arduino NANO es el microcontrolador escogido, la elección se ha explicado con anterioridad en el apartado anterior. En este caso, el NANO está basado en el microcontrolador ATmega38P.

Además, el Arduino NANO, posee las mismas capacidades que un Arduino UNO, siendo sus únicas diferencias el conector de alimentación y los pines. La diferencia en los pines reside en que en el Arduino NANO se tratan de pines con formato *header*.

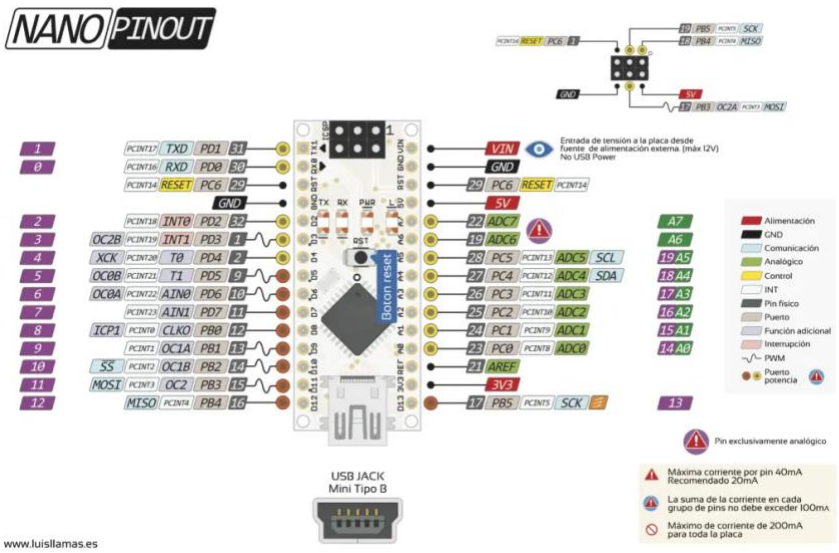

Ilustración 6 - Esquema de pines Arduino NANO

3.5. Resistencias

Las resistencias son un componente electrónico capaz de introducir una resistencia en un circuito. En este caso, se utilizan para disminuir la intensidad que circula por el circuito. Las resistencias utilizan un código de colores para distinguir si se trata de una con mayor o menor capacidad.

Las resistencias son una pieza importante para el correcto funcionamiento del circuito. En este caso, se utilizan resistencias de 5,1KΩ, cantidad suficiente como para poder utilizar el dispositivo empleando un transformador de móvil, el cual tiene una capacidad de transmitir 2,1A.

Si no utilizamos las resistencias, el sistema funcionaría cuando se utiliza con la alimentación del ordenador, pero, por otra parte, no funcionaría correctamente al utilizar un cargador de móvil.

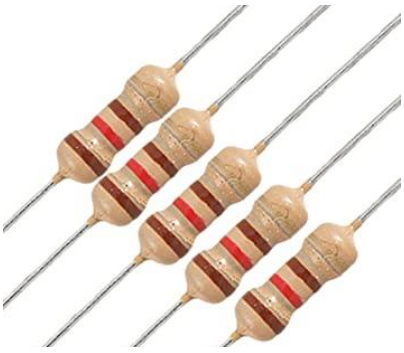

Ilustración 7 - Resistencias

3.6. Pantalla OLED

Para utilizar el sistema de enfoque de manera autónoma, es decir, sin necesidad de conectar a un ordenador, utilizamos una pantalla OLED de 0,96" y resolución de 128X64 píxeles. Esta pantalla, con unas dimensiones reducidas, no nos permite mostrar en pantalla una gran cantidad de información a la vez.

Pese a su tamaño reducido, se ha optado por una solución en la cual se muestra por pantalla una cabecera con los datos más importantes, en este caso, los pasos dados y la velocidad. Además de esta cabecera se muestra una serie de opciones, tratándose de un menú multi opción. En función de la pantalla en la que nos encontremos, podremos ver una información u otra en la pantalla.

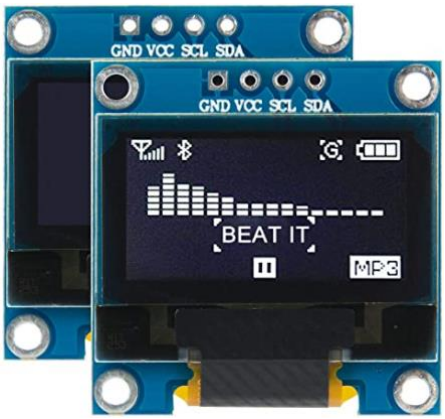

Ilustración 8 - Pantalla OLED monocromática

3.7. Placa de conexiones

La placa de conexiones o *protoboard* es una placa de pruebas utilizada para electrónica, la cual contiene numerosos orificios en los que insertar los cables, creando circuitos provisionales. Estos orificios se conectan junto a otros siguiendo un esquema definido.

Las conexiones entre estos orificios se realizan a nivel interno empleando pequeñas láminas metálicas conductoras. El patrón empleado es el que conecta los orificios por filas. Así pues, los orificios que se encuentran en la misma fila tienen conductividad entre sí y, por otro lado, los que no están en la misma fila, no tendrán conductividad.

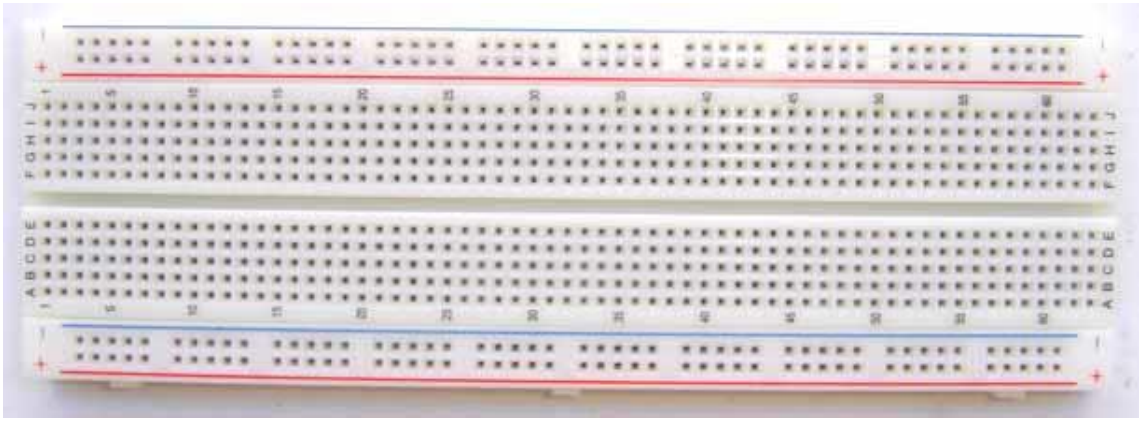

Ilustración 9 - Placa de conexiones

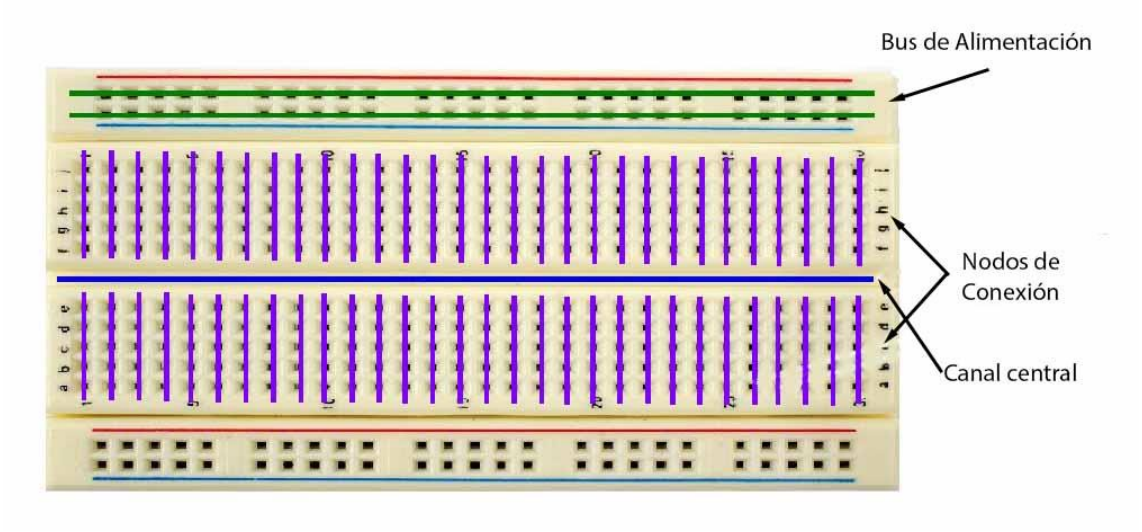

Ilustración 10 - Conexiones en la placa

3.8. Shapr3D

Shapr3D [3] se trata de un software de diseño 3D, actualmente propiedad de Siemens, empresa la cual ha realizado una gran cantidad de modificaciones en la aplicación para aplicar mejoras. En este caso se ha utilizado el plan de estudiante, con lo que se permite disponer de más de 2 diseños.

Esta aplicación se ha utilizado para crear los diseños en 3D para los soportes del motor y del cuerpo central del proyecto, el cual albergará en su interior todos los componentes a excepción del motor, el cual se encontrará separado y colocado en el telescopio junto a la rueda de enfoque.

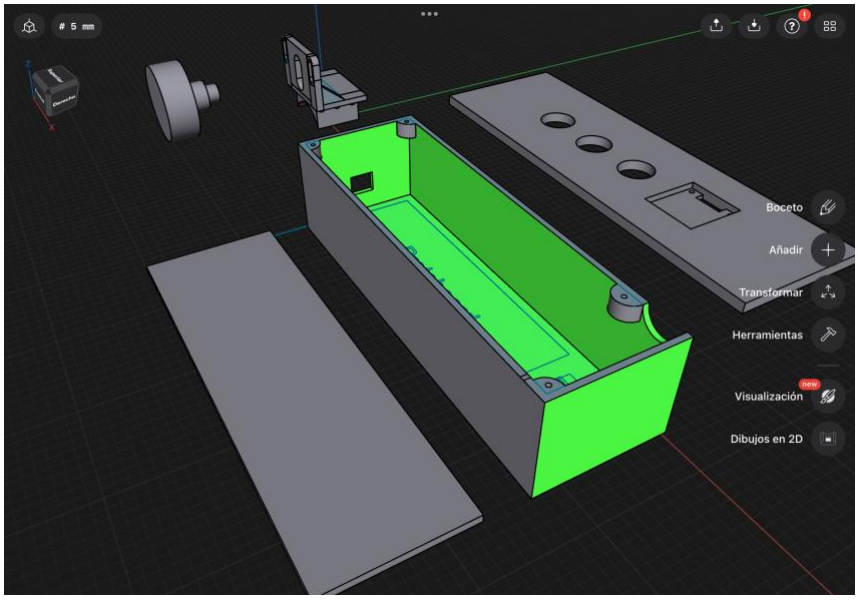

Ilustración 11 - Captura de pantalla del programa Shapr3D

3.9. Arduino IDE

Para realizar la programación se ha hecho uso del IDE de Arduino [4], un software libre creado por la empresa Arduino LLC.

Arduino utiliza un lenguaje propio basado en C++, así como también soporta lenguajes alternativos como lo es C.

Para programar en Arduino existen multitud de opciones, en este caso escogeremos Arduino IDE 1.8.19, otra opción sería utilizar Eclipse Arduino IDE, un plugin para Eclipse, el cual nos permitirá realizar la programación del dispositivo.

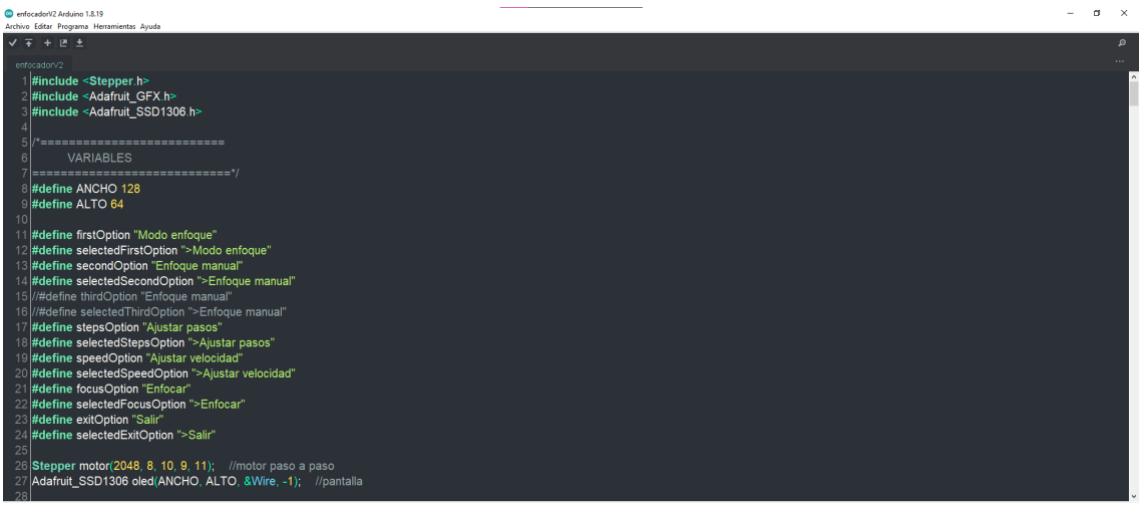

Ilustración 12 - Captura del IDE de Arduino

3.10. Fritzing

Utilizando el programa Fritzing [5], se ha realizado el diseño del esquema de conexiones de todos los componentes. Fritzing es un software libre de automatización de diseño electrónico. Utilizando este programa se ha facilitado el diseño del circuito, así como su montaje.

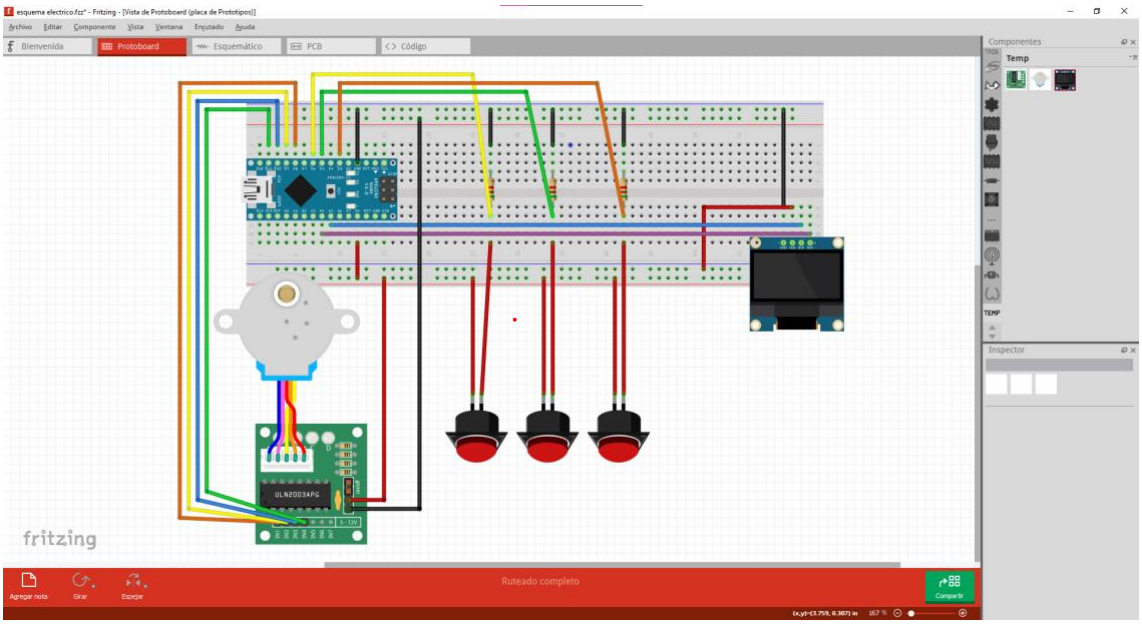

Ilustración 13 - Captura de pantalla del programa Fritzing

3.11. GitHub

Para poder trabajar en distintos dispositivos y tener un control de versiones, ya sea para tener un histórico de cambios o, simplemente para poder descargar los nuevos cambios en el momento de cambiar de dispositivo, se utiliza GitHub [6].

GitHub es una página para realizar un control de versiones de un proyecto, donde puede haber distintas ramas para cada proyecto. Por ejemplo, para el proyecto actual se ha creado un repositorio llamado “enfocador”, donde se almacena el archivo de código Arduino “*.ino”. Para cada una de las versiones se crea una nueva rama para, de esta forma, en el momento de subir los cambios, se suban a la rama.

Al subirse los cambios a la rama creada para la nueva versión se consigue que la versión actual siga en funcionamiento, puesto que se trata de la rama principal o *main*. De esta forma, se consigue que, en el caso de subir cambios en la nueva rama con errores, no afecten al funcionamiento del programa principal, alojado en la rama principal.

En este caso, para reflejar las dos versiones que se han creado del programa, existe un archivo para cada una de estas versiones, habiendo trabajado en un principio sobre el archivo “programaEnfocador.ino” y, posteriormente sobre la versión dos, en el otro archivo. Para cada uno de estos archivos se ha creado su correspondiente rama de versión, haciendo la fusión con la rama principal tras comprobar su correcto funcionamiento para aplicarse sobre el Arduino.

Para realizar las subidas del nuevo código se puede mediante consola, instalando *Git* en el ordenador, utilizando el programa de *Git* para ordenador o, en este caso, con *Visual Studio Code*, ya que permite un control de repositorio.

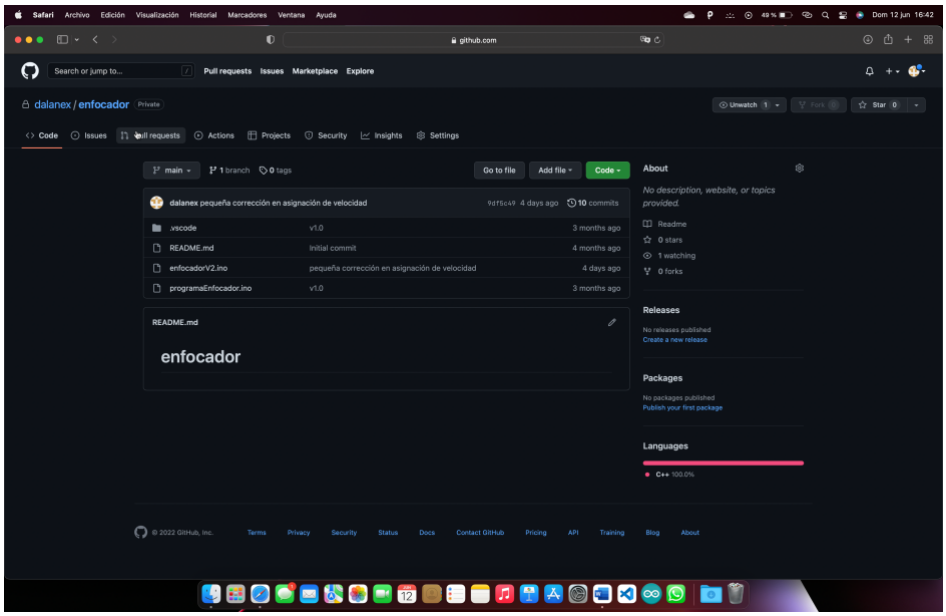

Ilustración 14 - Captura del repositorio

4. Pruebas de funcionamiento y preparación

En este apartado se explicará el proceso que se ha seguido para preparar el entorno de desarrollo y las pruebas que se han realizado para comprobar el correcto funcionamiento de los elementos.

Las pruebas realizadas consisten en realizar un programa sencillo, el cual sea capaz de utilizar los elementos de una forma controlada y, en base a unos resultados deseados, compararlos con los resultados obtenidos. En el caso de que los resultados obtenidos coincidan con los deseados, aseguraremos que, para esta prueba, funciona correctamente el componente. Una vez superadas todas las pruebas para cada uno de los componentes, aseguraremos que funciona correctamente.

4.1. Preparación repositorio

Para preparar el repositorio, una forma rápida de hacerlo es desde la página de GitHub, creando el nuevo repositorio y, una vez creado, crear el archivo de programa para Arduino y subirlo desde la página. Este proceso facilitará el proceso, puesto que no requiere de un programa externo ni la instalación de Git para utilizarse en el terminal.

Una vez creado el repositorio y habiendo subido el primer archivo, las subidas de cambios y la descarga de éstos se puede realizar desde el terminal o la aplicación de GitHub para el ordenador.

Para crear un repositorio siguiendo los pasos descritos anteriormente, el primer paso a realizar será iniciar sesión en la página de GitHub. Una vez en la página, se presionará sobre el botón de “nuevo”.

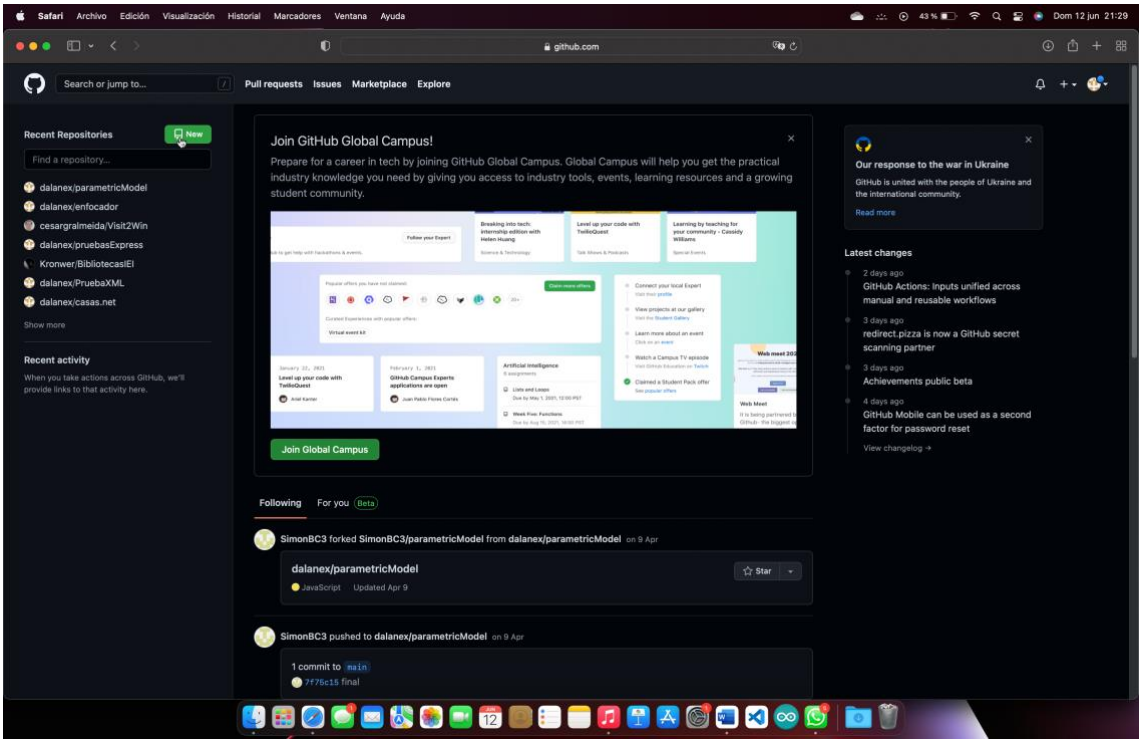

Ilustración 15 - Creación de nuevo repositorio

Una vez presionado, al estar en la siguiente pantalla se deberá asignar el nombre que se desee asignar al repositorio. Para este ejemplo, puesto que ya existe el repositorio deseado, se ha asignado el nombre "repositorioPrueba".

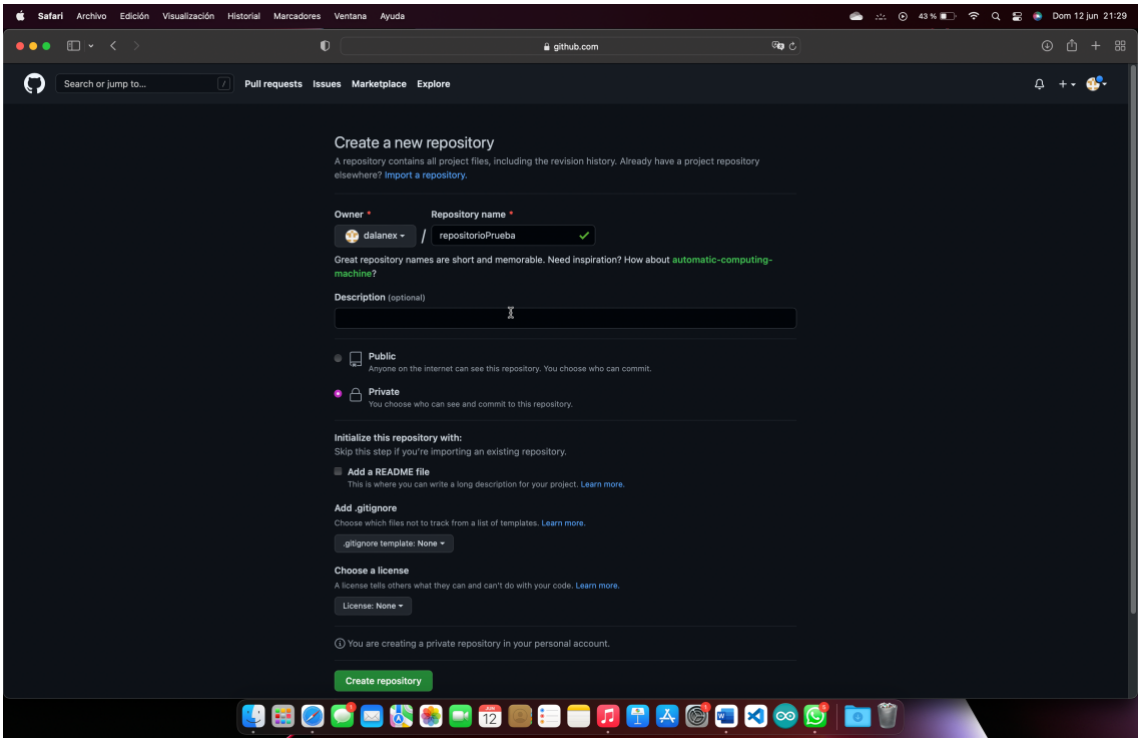

Ilustración 16 - Asignación de nombre al repositorio

Una vez asignado el nombre aparecerá una pantalla en la cual se podrá elegir si utilizar un archivo existente de código, un repositorio como base o, por otra parte, la creación del repositorio en el terminal. En este caso se ha utilizado la opción de “utilizar archivo de código existente”.

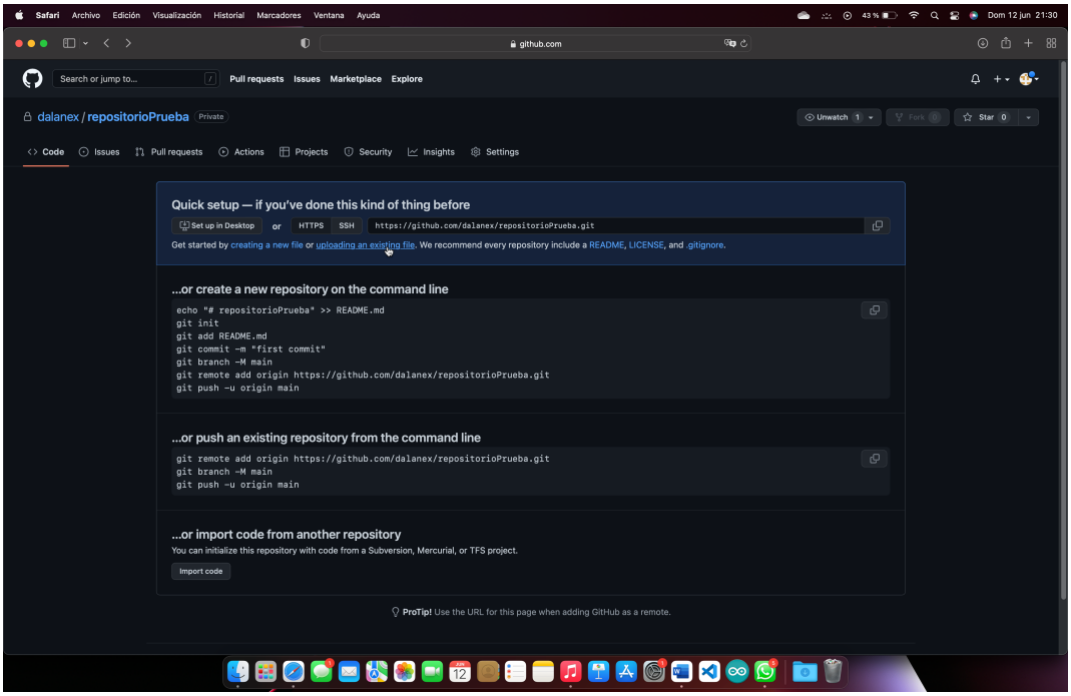

Ilustración 17 - Elección método de creación de repositorio

El último paso será seleccionar el archivo deseado para, de esta forma, crear la petición de subida al repositorio con el primer archivo. Esta es una forma muy rápida de crear un repositorio utilizando un archivo de código ya creado anteriormente.

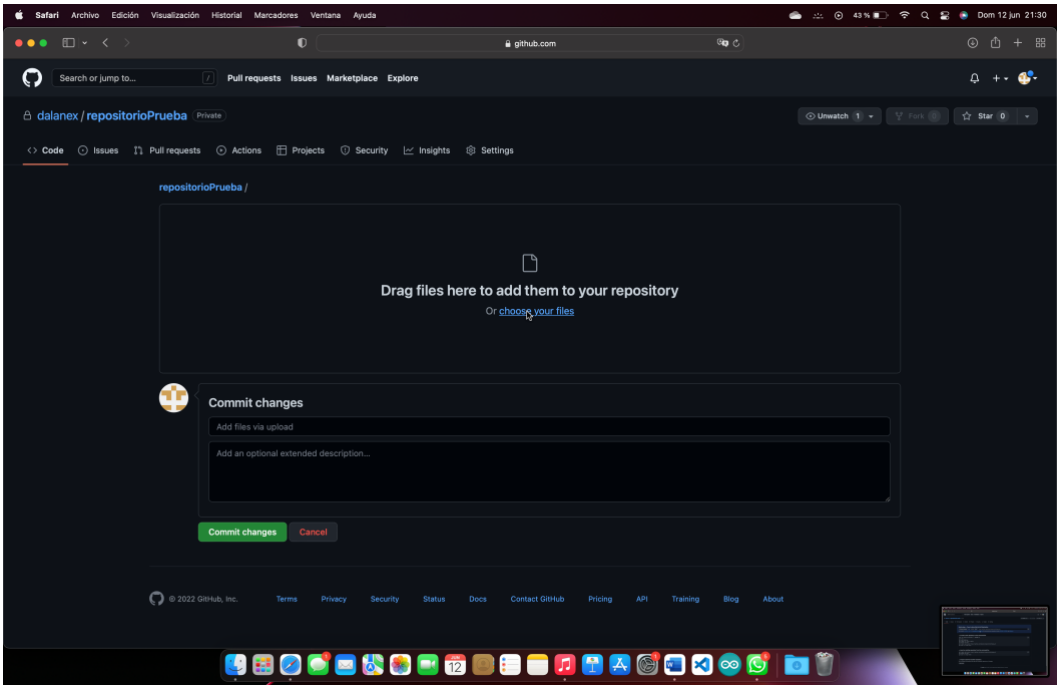

Ilustración 18 - Selección de archivo para subir

4.2. Preparación Arduino

En primer lugar, antes de comenzar a realizar el programa, se deberá seguir una serie de pasos para asegurar que la configuración sea la correcta para poder comenzar. La configuración consiste en seleccionar el tipo de placa que utilizaremos, el puerto en el que se ha conectado la placa y, por último, el *bootloader*.

Para configurar el entorno se deberán seguir los siguientes pasos en el IDE. En primer lugar, acceder al apartado "Herramientas", "Placa:", la placa se escogerá de entre una serie de posibilidades, en este caso se utiliza Arduino Nano. Por último, al tratarse de una placa de Arduino NANO, deberemos comprobar si el *bootloader* por defecto es el correcto.

El *bootloader* es un programa que se encuentra en la placa, el cual es el encargado de, si llega por el puerto serie un *sketch*, el programa que acabamos de realizar, lo cargará en la memoria *flash*. Si, por el contrario, no llega ningún *sketch*, se ejecutará el anteriormente guardado.

Para la placa Arduino NANO se dispone del *bootloader* ATMEGA328P anterior y el actual. Para comprobar si con el actual la placa será capaz de recibir el *sketch* y ejecutarlo correctamente, se escogerá un programa de ejemplo. Para escoger un programa de ejemplo se accederá al apartado "Archivo", "Ejemplos". Para esta prueba se escogerá uno sin importar cual sea, en este caso se emplea "Basics", "Blink".

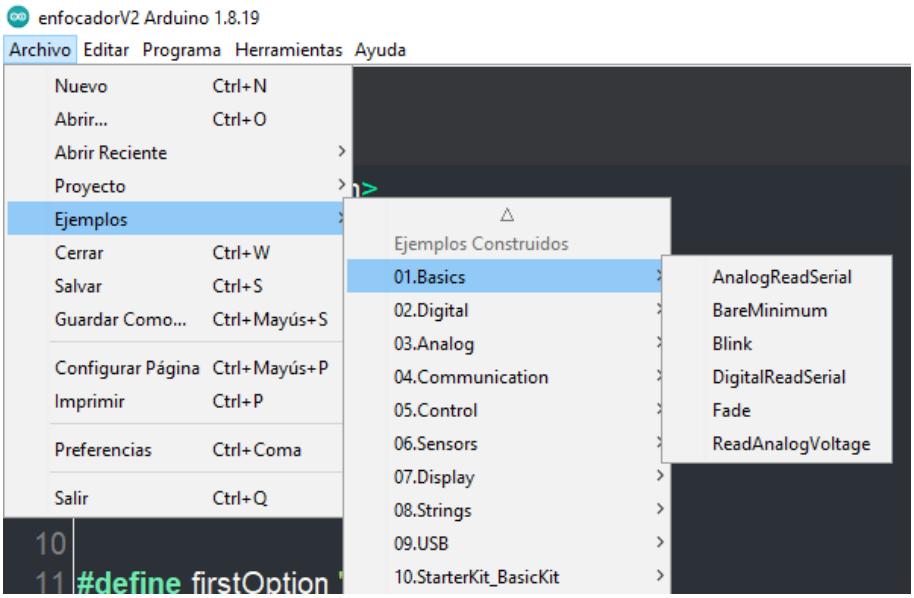

Ilustración 19 - Selección del programa de ejemplo

Para subir el *sketch* a la placa se pulsará sobre el botón "Subir", tras lo cual aparecerá en la parte inferior de la pantalla un cuadro en el que aparecerá el texto "Subido" en caso de que la

operación se complete con éxito. En caso contrario, aparecerá un mensaje de error en este cuadro que ha aparecido en la parte inferior de la pantalla.

Si aparece el error al subir, se deberá cambiar el *bootloader* al anterior. Para ello, desde el apartado "Herramientas", "Procesador:", se deberá escoger la opción "ATMEGA328P (Old Bootloader)". Una vez seguidos estos pasos, el programa se subirá correctamente a la placa y aparecerá el mensaje "Subido", con el que se podrá comprobar que el proceso de subida del *sketch* se ha completado con éxito.

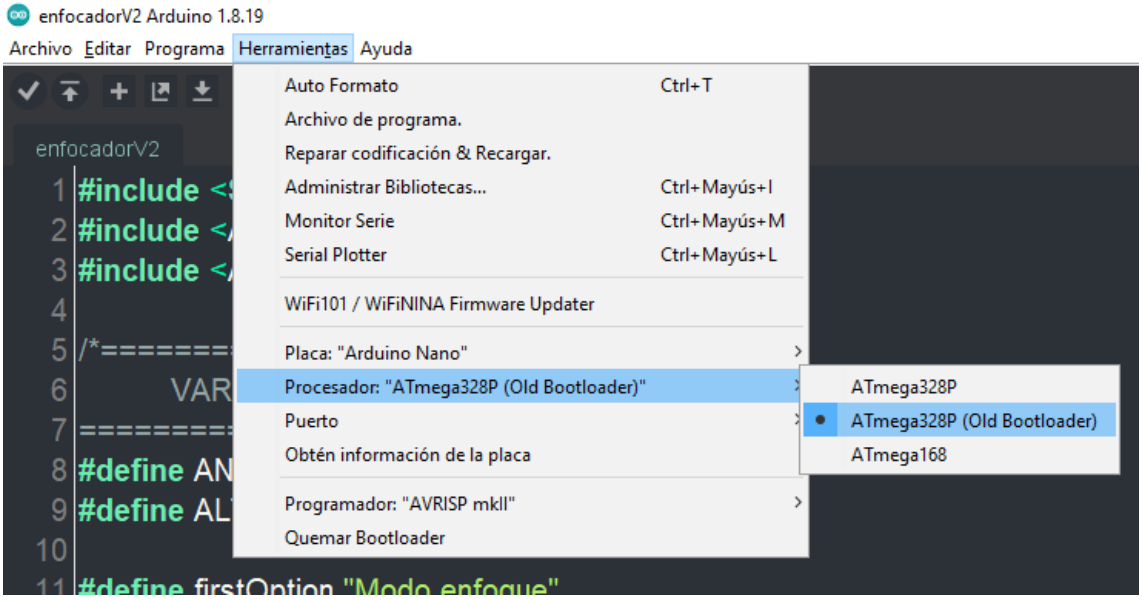

Ilustración 20 - Selección del bootloader

Una vez realizados estos pasos, lo próximo es comenzar a programar, para lo que se deberá tener en cuenta las librerías a emplear, si se debe importar una librería y, lo más importante, si se modifica el USB en el que se conecta la placa, se deberá modificar desde el apartado de "Herramientas". Esto último es importante ya que, existe la posibilidad de que no se modifique automáticamente el puerto escogido tras cambiar el microcontrolador de USB, lo que ocasionará un error en el momento de subir el *sketch* a la placa.

4.3. Pulsadores

El próximo paso para comenzar el proyecto consiste en comprobar el correcto funcionamiento de los pulsadores. Podría no requerirse, en caso de confiar en el correcto funcionamiento. No obstante, las pruebas se realizan para comprobar que los pulsadores empleados funcionen correctamente, no se registren pulsaciones cuando no se pulsan y comprobar que cuando se pulsan no se registre la pulsación de forma errónea.

Para ello, se ha realizado un montaje de un circuito sencillo empleando una resistencia para, de esta forma, crear un circuito de resistencia PULL-DOWN. Se podría no utilizar resistencia y conectar directamente el pulsador a la salida de 5V de Arduino y al pin donde se desee realizar la lectura.

La resistencia se emplea para comprobar así que no esté quemada, así como para realizar el montaje del circuito que servirá más adelante para el circuito final.

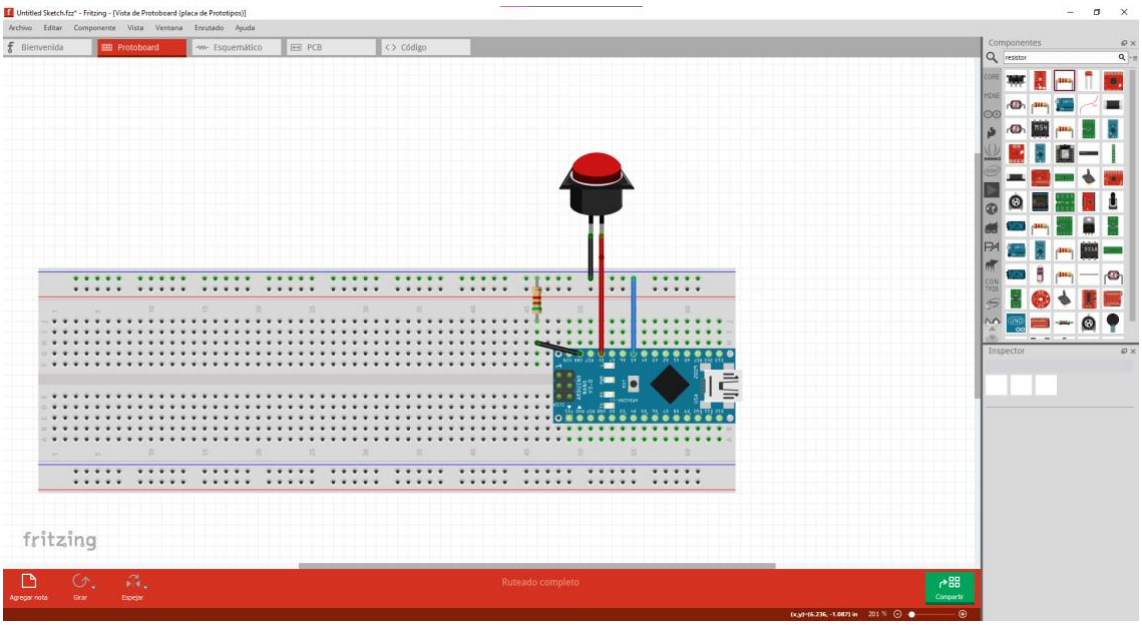

Ilustración 21 - Esquema resistencia PULL-DOWN

Para comprobar el correcto funcionamiento del pulsador se ha creado un programa muy sencillo en el que se realiza una lectura del pin declarado como pin de entrada. Se mostrará por pantalla a través del puerto serie si se encuentra pulsado o no el pulsador, si aparece pulsado únicamente cuando se pulsa y aparece como no pulsado cuando no se pulsa, en ese caso, se concluye que el pulsador funciona correctamente.

```
int pulsador = 5;

void setup() {
    pinMode(pulsador, INPUT);
    Serial.begin(9600);
}

void loop() {
    if( HIGH == digitalRead(pulsador)) Serial.println("PULSADO");
    else Serial.println("NO PULSADO");
}
```

Ilustración 22 - Código de prueba de pulsadores

4.4. Motor paso a paso y ULN2003

Las pruebas mediante las cuales se comprobará el correcto funcionamiento tanto del motor paso a paso empleado como del módulo controlador se realizarán juntas. Para ello, se utiliza un programa de ejemplo.

Este programa se puede encontrar entre los programas de ejemplo de Arduino, en este caso se utiliza el ejemplo “stepper”.

```
#include <Stepper.h>

int stepsPerRevolution = 2048;
Stepper myStepper(stepsPerRevolution, 8, 9, 10, 11);

void setup() {
  myStepper.setSpeed(10);
  Serial.begin(9600);
}

void loop() {
  Serial.println("clockwise");
  myStepper.step(stepsPerRevolution);
  delay(500);

  Serial.println("counterclockwise");
  myStepper.step(-stepsPerRevolution);
  delay(500);
}
```

Ilustración 23 - Código de pruebas de motor y módulo controlador

4.5. Pantalla OLED

Las pruebas para comprobar el correcto funcionamiento de la pantalla han sido sencillas, se ha utilizado un programa poco complejo para que aparezca “Hola Mundo!”. Una vez comprobado el funcionamiento se ha realizado una prueba más exhaustiva mediante la cual se ha creado un programa para rellenar toda la pantalla y comprobar que no haya ningún píxel muerto.

Se podría realizar directamente la primera prueba, ya que es una prueba más completa y que involucra toda la pantalla, pero, realizando las dos pruebas, se ha podido tener un mayor uso de la pantalla para crear el programa y comprender de una mejor forma los métodos existentes en la librería.

```
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define ANCHO 128
#define ALTO 64

Adafruit_SSD1306 oled(ANCHO, ALTO, &Wire, -1);
void setup() { oled.begin(SSD1306_SWITCHCAPVCC, 0X3C); }

void loop() {
    oled.clearDisplay();
    oled.println("Hola Mundo!");
    oled.display();
    delay(1000);
}
```

Ilustración 24 - Código de prueba de pantalla OLED

5. Diseño del enfocador

5.1. Esquema eléctrico

Las conexiones entre los distintos elementos se han realizado utilizando una placa de conexiones. Podría haberse empleado una placa de estaño para crear un circuito impreso en caso de requerir de una solución más permanente.

En este caso se ha utilizado una placa de conexiones dado que se disponía de una, facilita el montaje del proyecto y, en caso de conseguir un buen resultado, se puede crear un circuito impreso para un proyecto futuro, mejorando el sistema actual.

Tal y como se puede apreciar en la **Error! Reference source not found.**, la cual es una ilustración del esquema eléctrico completo del sistema.

La placa Arduino NANO es capaz de alimentar a todo el circuito, ya sea utilizando el cable para conectarlo al ordenador desde el que se podrán realizar una serie de ajustes para mover la rueda de enfoque. También es posible alimentar la placa utilizando un cargador de móvil, en este caso el sistema no se utilizará con ningún ordenador, únicamente con los pulsadore que se incluyen en la tapa del sistema centra.

5.1.1. Resistencias PULL-DOWN

En este proyecto se han empleado resistencias de 5,1kΩ para crear un circuito de resistencia PULL-DOWN [7]. En caso de no disponer de una resistencia para crear este montaje, Arduino dispone de resistencia PULL-UP incorporada, siendo únicamente necesario declarar el pin como “INPUT_PULLUP”.

No se ha optado por esta solución puesto que de cara al consumo estático se incrementará, así como el calor producido por la propia resistencia. No obstante, si se empleara un único pulsador es una solución muy acertada, puesto que no requiere de material externo. En este caso, se trata de tres pulsadores, por lo que la potencia estática y el calor aumentan, y se puede evitar utilizando la resistencia PULL-DOWN.

Empleando el sistema PULL-DOWN nos garantiza que, cuando el pulsador cierra el circuito, el pin es capaz de leer una señal de 5V. Por otro lado, en caso de utilizar PULL-UP, el pin por defecto detectará 5V, siendo necesario controlar en el programa que la lectura sea baja, es decir, 0V.

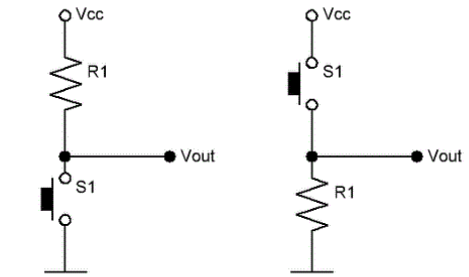

Ilustración 25 - Resistencia PULL-UP y PULL-DOWN

5.1.2. Módulo controlador de motores ULN2003 y motor paso a paso

Una vez probado el correcto funcionamiento del motor paso a paso [8], así como del módulo controlador, el próximo paso es realizar las conexiones entre la placa Arduino, el motor y el módulo controlador.

Para este proyecto, el módulo controlador utiliza 4 de las 7 entradas de las que dispone, una para cada bobina necesaria para su activación. Asimismo, se requiere de una conexión con la salida de 5V del Arduino, así como del pin de 0V.

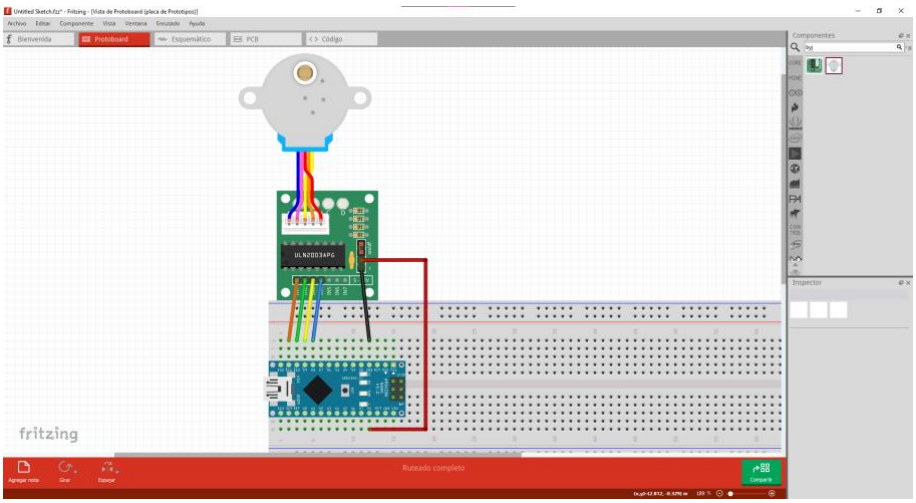

Ilustración 26 - Conexiones motor y módulo controlador

5.1.3. Pantalla OLED

La pantalla OLED [9], así como el motor y el resto de los componentes, una vez comprobado su funcionamiento, se realiza la conexión con la placa Arduino, asignando los pines utilizando el esquema por defecto. En este esquema las relaciones entre pines es la siguiente:

- SCL – A5
- SDA – A4

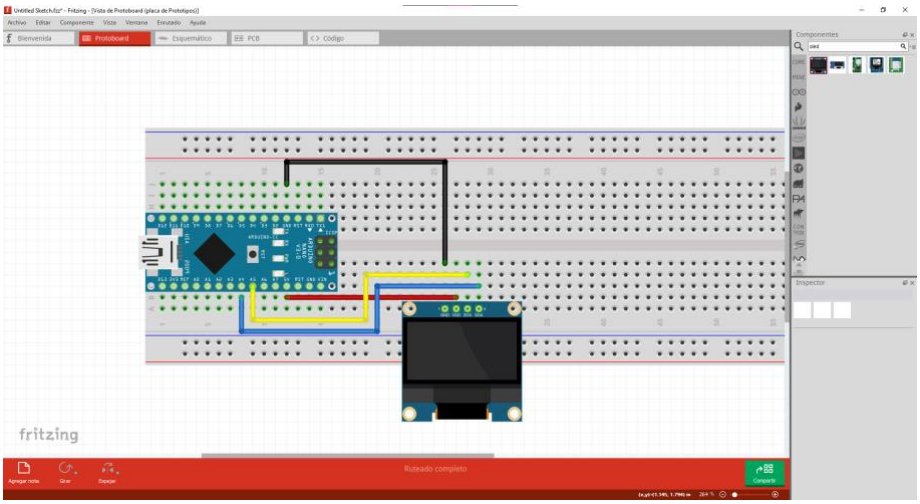

Ilustración 27 - Conexiones pantalla OLED

Durante la fase de diseño no se han tenido en cuenta únicamente las conexiones para asegurar la disponibilidad de pines suficientes por parte de la placa Arduino, sino que también se han diseñado las distintas pantallas a mostrar.

Se ha realizado una serie de diseños para que sea más sencillo el desarrollo del sistema. Estos diseños consisten en, sabiendo los datos más importantes a mostrar y teniendo una primera versión del menú de opciones a mostrar, ubicar de la mejor forma y con los tamaños de fuente adecuado cada uno de los elementos a mostrar.

Además de esto, se han creado las transiciones entre las distintas pantallas, tanto para una primera versión como para la segunda versión, donde se han realizado las modificaciones más importantes. La versión final únicamente incorpora unas modificaciones menores, para las que no se han creado diseños nuevos, sino que se ha modificado directamente desde el código fuente para crear estas pequeñas modificaciones.

Los cambios realizados en la versión final respecto de la segunda versión no ha sido necesario crear diseños nuevos, tal y como se ha explicado anteriormente. Para añadir, estos cambios han sido pequeños retoques en algunas etiquetas a mostrar, cambiando la forma en la que aparece, ningún cambio referido a la creación de nuevas pantallas o modificaciones completas de éstas.

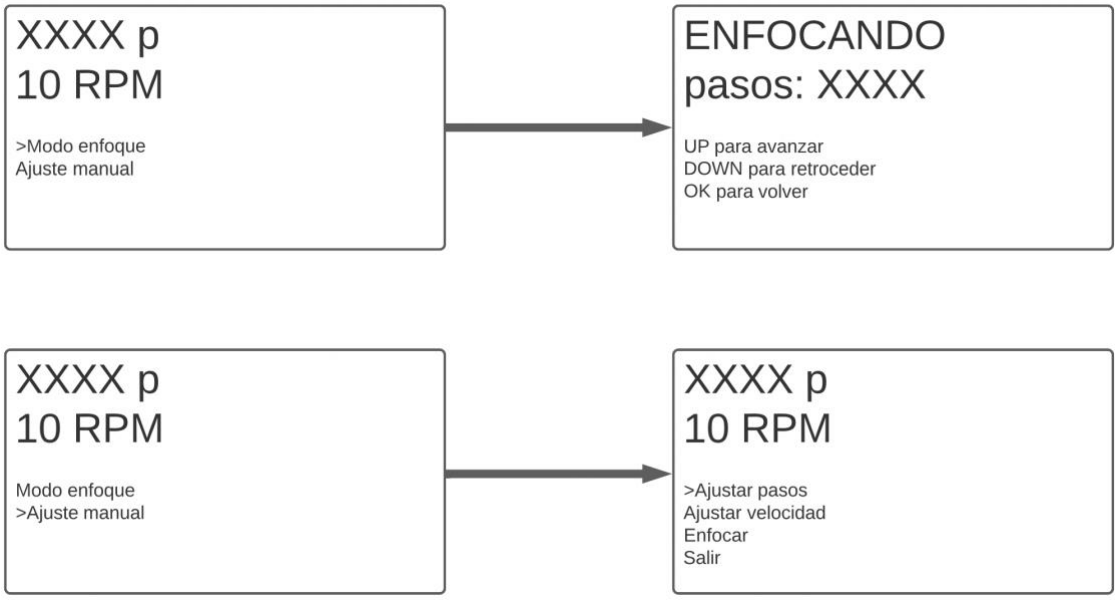

Ilustración 28 - Navegación entre pantallas principales

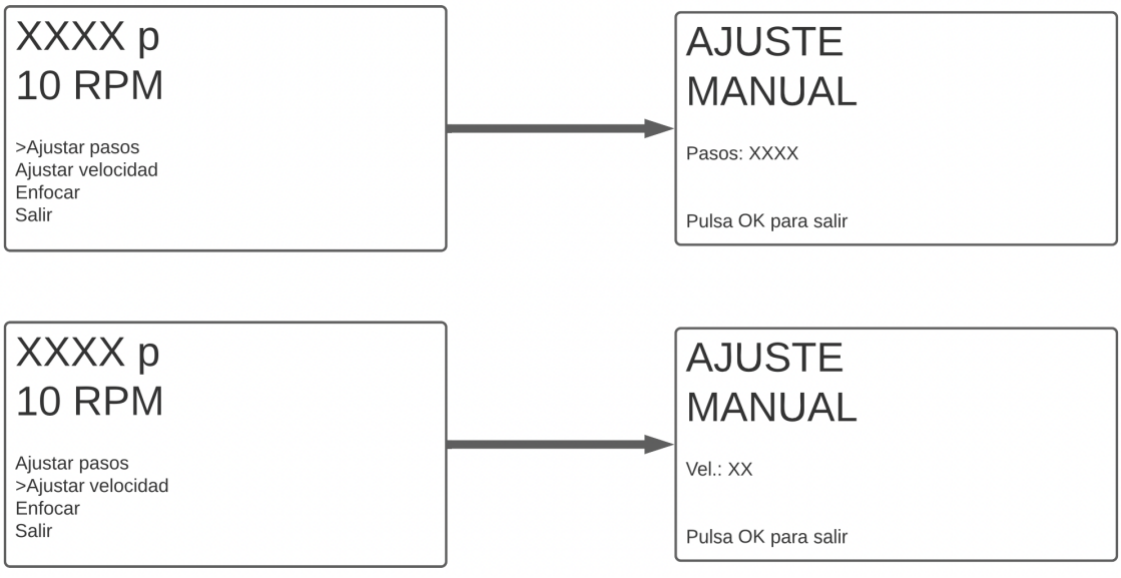

Ilustración 29 - Navegación entre menú de opciones de enfoque manual

5.1.4. Interfaz USB

En el momento en el que el sistema se conecta a un ordenador por USB en el cual se dispone del Arduino IDE o a través el monitor serie disponible en la página web de Arduino, donde se dispone también de un editor para poder modificar el programa. Se ha diseñado una sencilla interfaz en la cual se podrán elegir cuatro opciones: avanzar una cantidad de pasos, avanzar hasta un paso dado, modificar la velocidad de giro del motor y, por último, restablecer los pasos dados a cero.

La interfaz es una interfaz de tipo consola, donde aparecen los mensajes con las posibles acciones y se debe introducir la opción deseada, siendo posible escoger entre cuatro opciones descritas anteriormente.

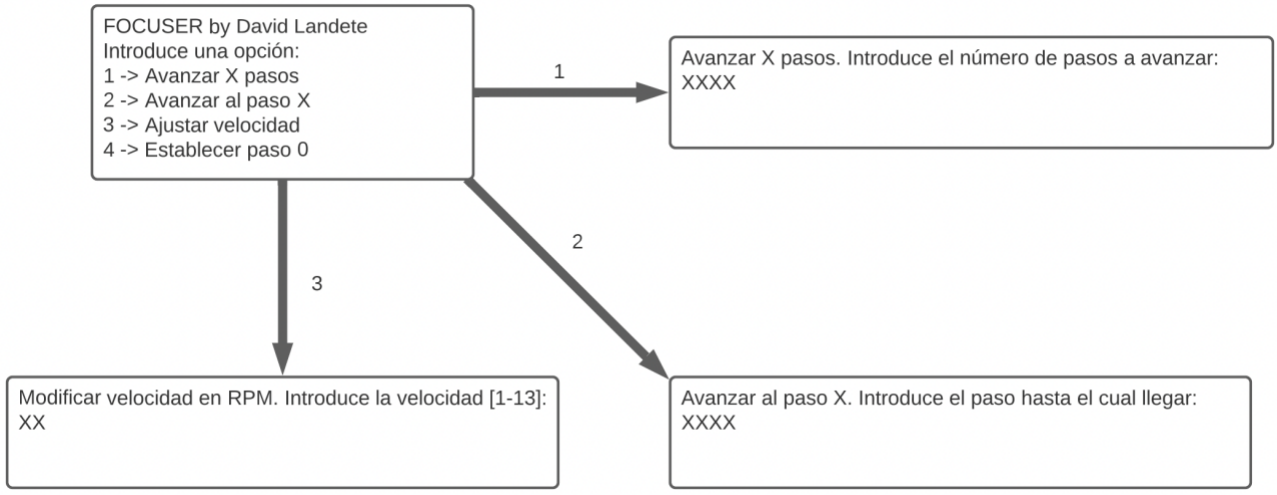

Ilustración 30 - Navegación entre opciones de la interfaz USB

Cuando se selecciona la cuarta opción, no aparece ninguna pantalla diferente, como sí ocurre al seleccionar por ejemplo la opción de avanzar X pasos. En este caso, el funcionamiento del programa resetea la cantidad de pasos dados hasta el momento, viéndose reflejado en la pantalla OLED. La cantidad de pasos dados en la interfaz USB aparecerá cuando se introduzca la cantidad de pasos a avanzar, ya sea mediante la primera o segunda opción.

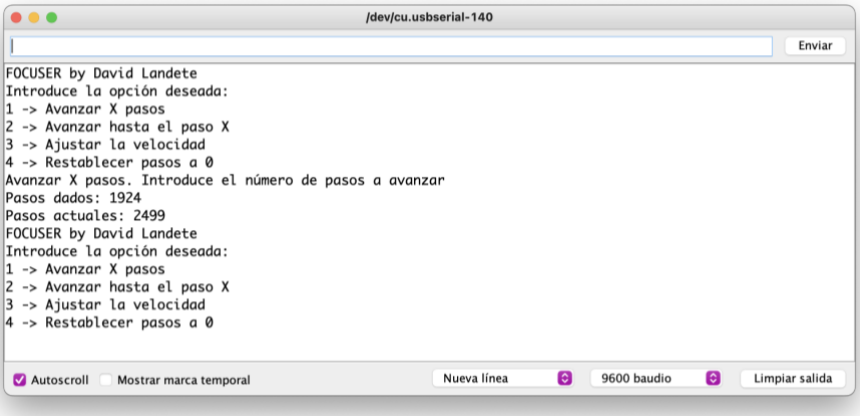

Ilustración 31 - Interfaz USB avanzando una cantidad de pasos

5.1.5. Conexiones Arduino

En este proyecto se utilizan distintos componentes, entre los cuales los detallados anteriormente. Las conexiones se han detallado a lo largo de este apartado, cada uno en el apartado correspondiente.

Dado que la placa Arduino NANO es capaz de suministrar la alimentación a todos los componentes, sin necesidad de utilizar una fuente externa, las conexiones se simplifican, dando lugar al esquema que se puede apreciar en la Ilustración 14, donde se puede apreciar el esquema eléctrico final y la asignación de pines.

La asignación de pines para el proyecto es la siguiente:

- Pulsador OK -> D3
- Pulsador SUBIR -> D6
- Pulsador BAJAR -> D5
- Módulo controlador de motor
    - IN1 -> D8
    - IN2 -> D9
    - IN3 -> D10
    - IN4 -> D11
- Pantalla OLED
    - SCL -> A5
    - SDA -> A4

5.2. Sistema central

Para el diseño del sistema central se ha utilizado la aplicación Shapr3D, una aplicación de diseño 3D que se puede encontrar en la Apple Store, estando disponible para iPad y Mac y, por otra parte, también disponible para Windows.

El sistema central consiste en una caja, la cual albergará en su interior el microcontrolador, la placa de conexiones o *protoboard* y el módulo controlador del motor paso a paso empleado.

Antes de comenzar con los diseños en 3D, se han creado una serie de bocetos en los cuales se refleja la idea para cada una de las piezas a crear. En estos bocetos se han tenido en cuenta las dimensiones de los componentes a emplear.

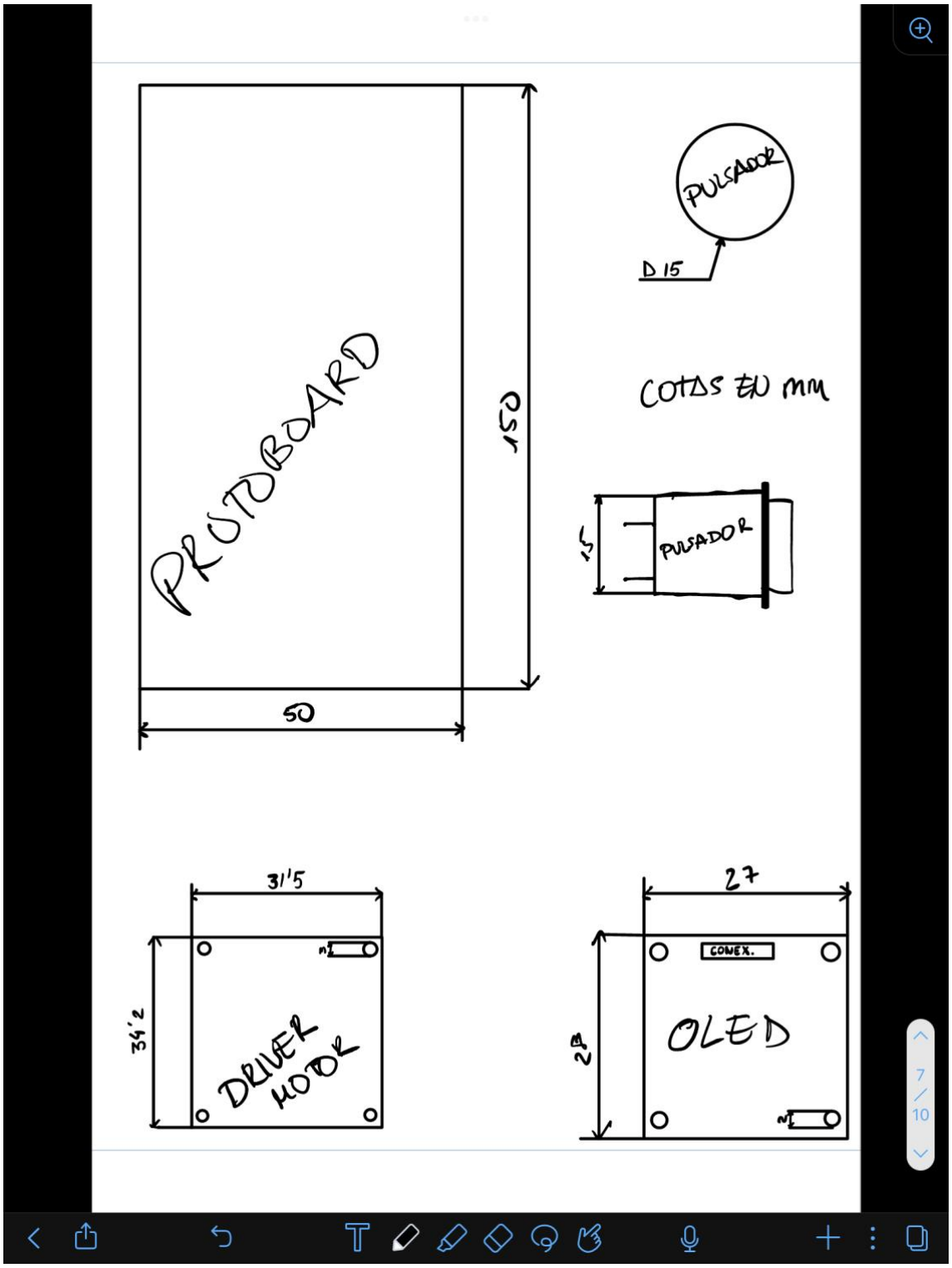

Ilustración 32 - Bocetos elementos con medidas

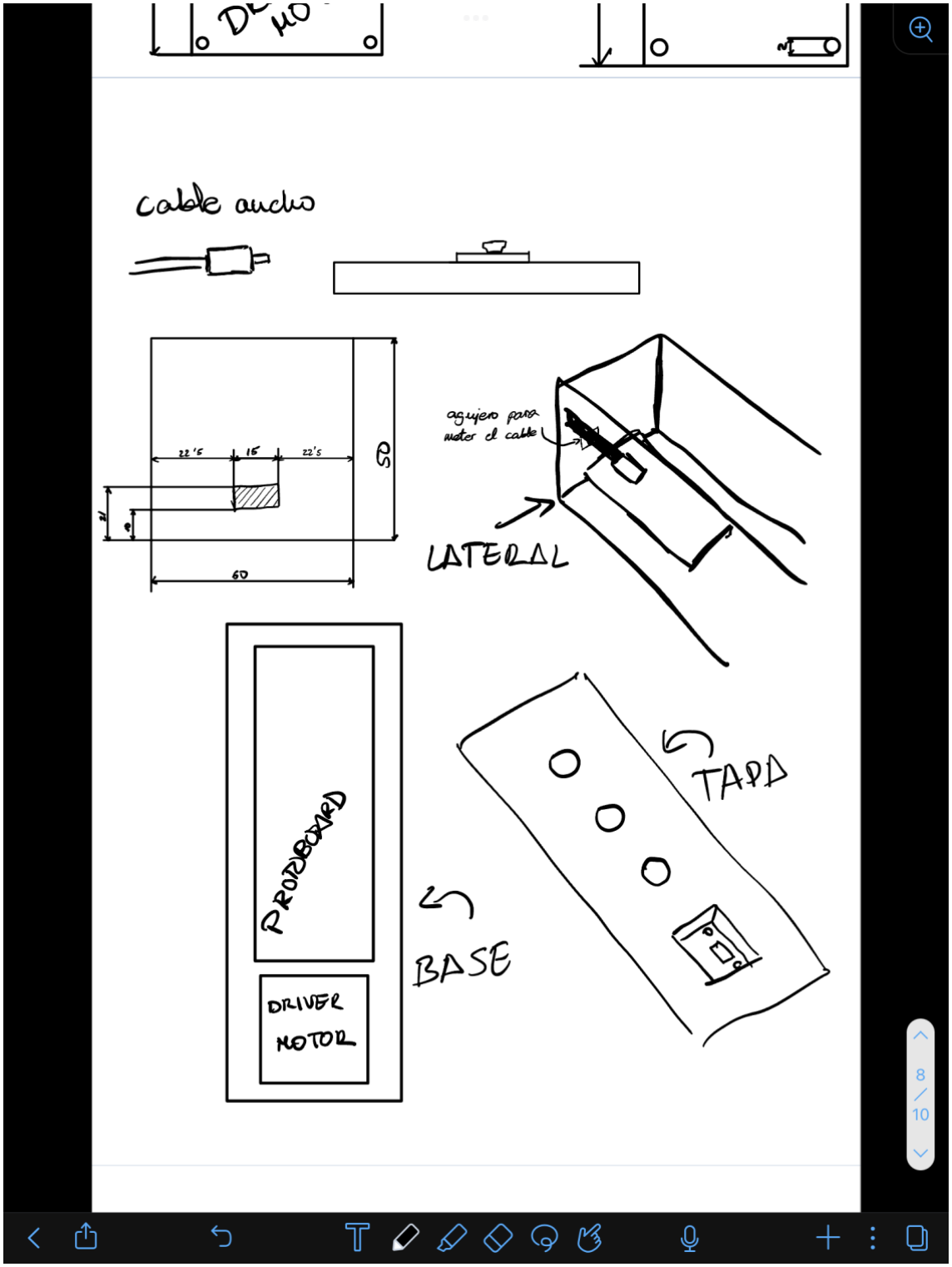

Ilustración 33 - Bocetos sistema central

5.2.1. Base

Dado que en el interior del sistema central se alberga la placa de conexiones y el módulo controlador del motor, se ha optado por diseñar una base sencilla, en la que se ha tenido en cuenta el tamaño de la placa de conexiones y del módulo controlador del motor, más un margen de separación entre ambos componentes, así como con las paredes.

Para realizar el diseño se ha partido de las medidas de los componentes para, de este modo, empezar a crear bocetos en la aplicación Shapr3D, con la cual podemos crear rectángulos con las medidas deseadas y desplazarlos hasta obtener la posición deseada.

Una vez obtenida una distribución de los componentes adecuada, se crea el diseño final en el cual, ya teniendo las medidas del primer paso, para así, obtener el resultado final, el cual se imprimirá en 3D junto con las pareces para crear el sistema central en 2 piezas, la base y las paredes y la tapa.

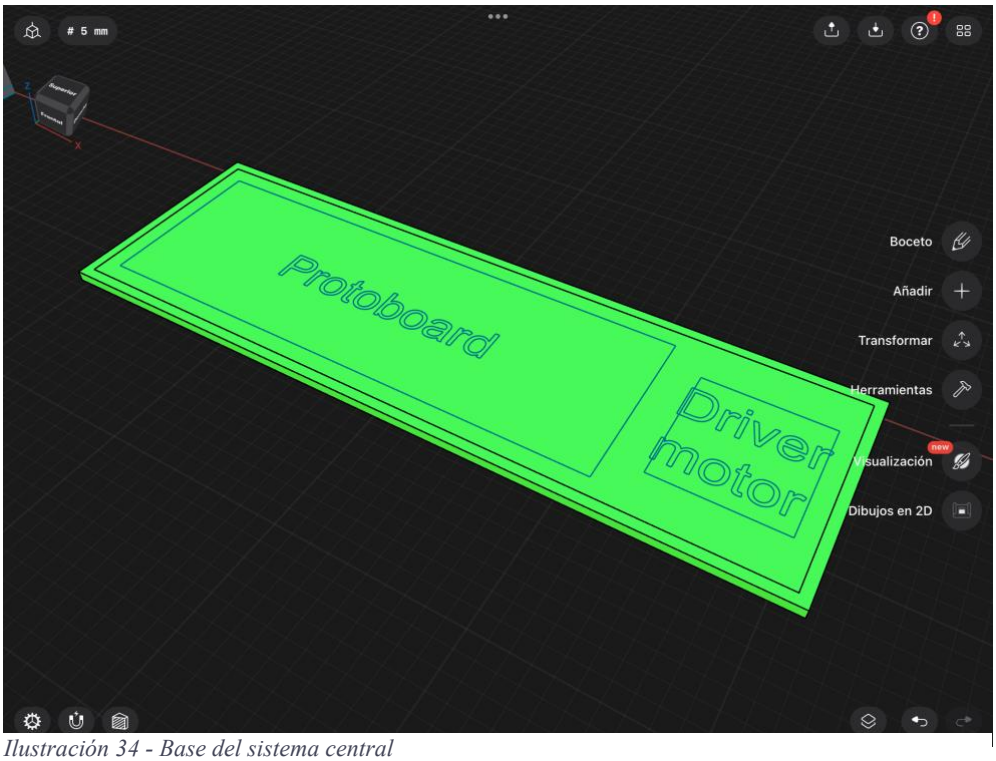

Ilustración 34 - Base del sistema central

5.2.2. Laterales

Durante la etapa del diseño se los laterales, se ha optado por utilizar unas pareces de 3mm de ancho, los cuales tienen anchura suficiente como para poder soportar la estructura.

Dado que en el interior se alberga el microcontrolador, una de las paredes posee un agujero lo suficientemente grande como para que el cable de alimentación del Arduino pase con holgura suficiente sin suponer un peligro para la estructura.

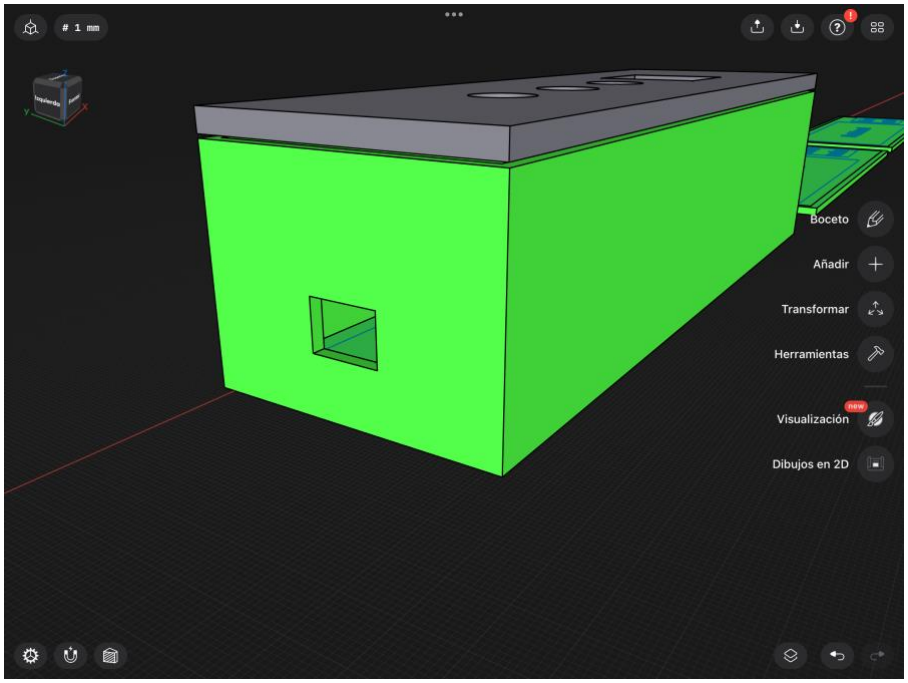

Ilustración 35 - Lateral 1 del sistema central

La otra cara lateral en la que se encuentra una diferencia es, la cara desde la que saldrán los cables correspondientes al motor paso a paso. Eso es necesario puesto que, en caso de acoplar el sistema central al telescopio, es posible que se coloque en una disposición difícil de operar desde el propio telescopio.

Separando el cuerpo central del soporte del motor se consigue que, teniendo los controles del sistema central en mano en todo momento, se necesite un cable que parta desde el interior hasta el motor con una longitud tal que se pueda colocar en cualquier posición el telescopio.

Utilizando este método de separación se consigue que el sistema central se emplee como un mando de control remoto cableado.

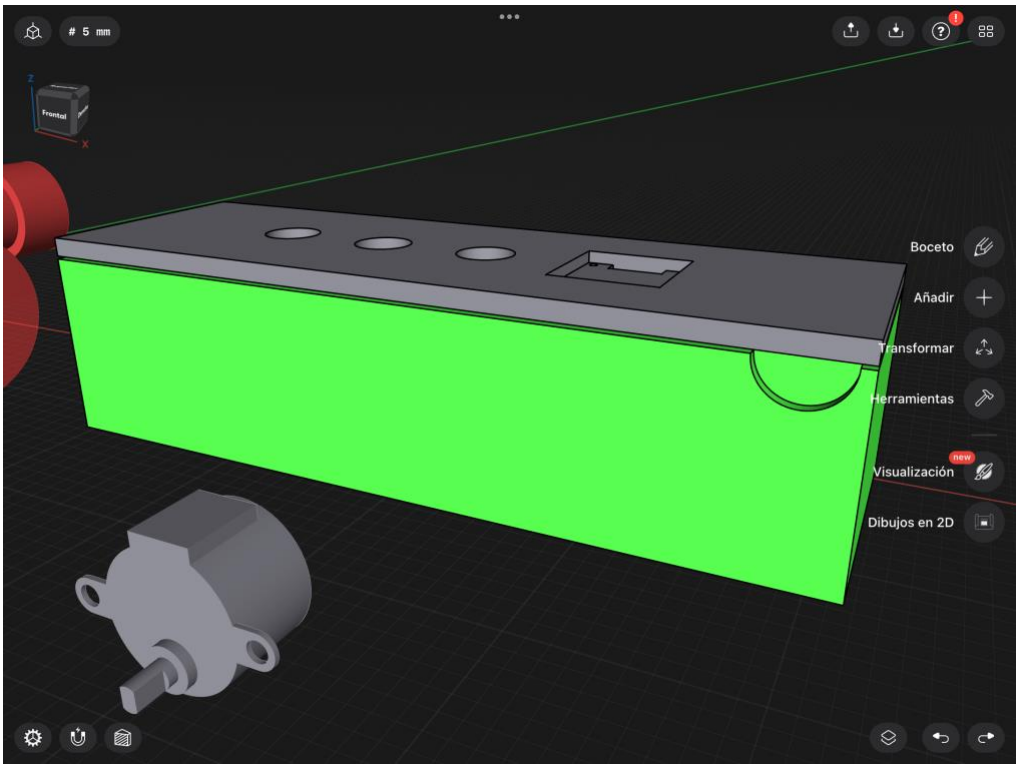

Ilustración 36 - Lateral 2 del sistema central

5.2.3. Tapa

El diseño de la tapa se ha creado específico para poder colocar los 3 pulsadores empleados en el proyecto y la pantalla en la cual se mostrarán las opciones.

El soporte de la pantalla es un hundimiento en la tapa para colocar la pantalla a un nivel más igualado que el resto de la tapa, añadiendo un agujero para que pasen los cables al interior del cuerpo central y poder realizar las conexiones, así como los 4 agujeros para fijar la pantalla con tornillos a la tapa.

También se encuentran los 3 agujeros para los pulsadores, los cuales poseen su propio sistema de anclaje empleando una tuerca.

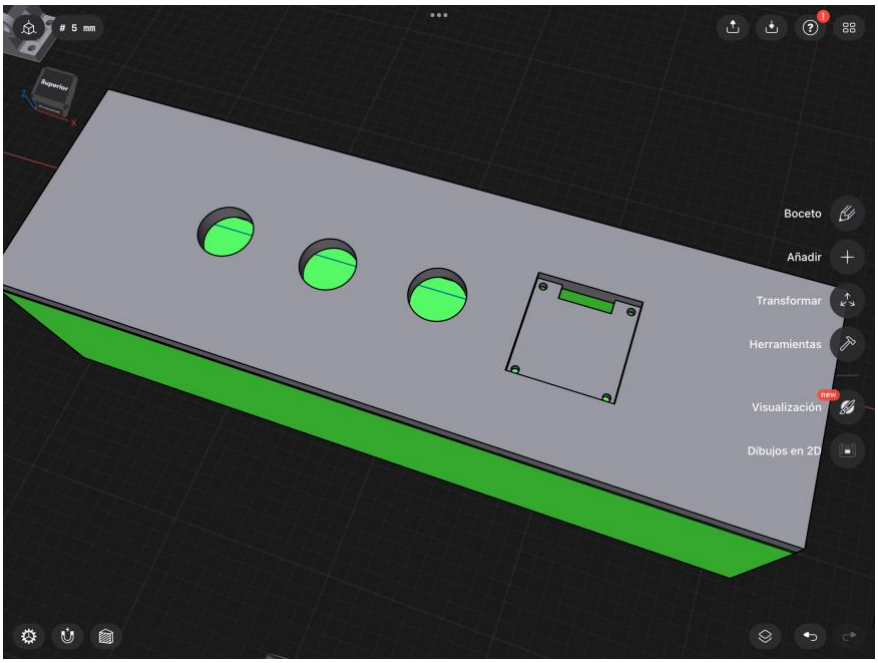

Ilustración 37 - Tapa del sistema central

5.3. Soporte del motor

El soporte para el motor diseñado es un soporte sencillo en el cual el motor se anclará utilizando 2 tornillos pasándolos por los agujeros de los que dispone el motor.

Para poder crear el diseño lo más genérico posible, se ha creado un soporte el cual el agujero donde se fijará el tornillo es una hendidura con forma ovalada, pero manteniendo rectos los laterales que unen las circunferencias superior e inferior, por lo que no se trata de un óvalo, sino de una figura con una forma similar.

Empleando estos soportes se permite la posibilidad de ajustar el motor para colocarlo más alto o bajo para así, poder adaptarse mejor a distintos telescopios. Además, dada la posibilidad del ajuste, se podría utilizar con otras ruedas de acople al telescopio de diferentes tamaños en caso de disponer de otras más pequeñas o grandes.

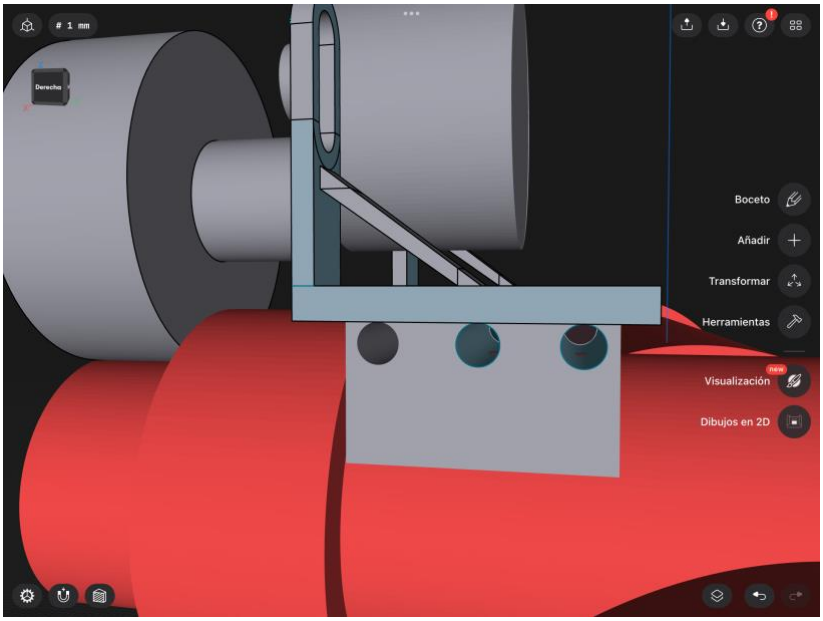

Ilustración 38 - Soporte de motor

5.4. Acople motor - telescopio

Esta pieza ha sido diseñada para dotar al sistema de enfoque creado de la posibilidad de acoplarse a un telescopio para, de este modo, mover la rueda del enfoque. Empleando esta pieza se logra el acople del motor con la rueda de enfoque del telescopio, permitiendo que no sea necesario mover el telescopio a mano, utilizando el sistema creado.

La pieza creada se ha diseñado teniendo en cuenta el acople con el motor, utilizando las medidas que se pueden obtener del manual de especificaciones del motor.

El material empleado será un material el cual permita que el rozamiento con la rueda de enfoque sea muy alto para evitar deslizamientos. La elección del material se explicará en el punto siguiente, donde se detalla el proceso de construcción del sistema.

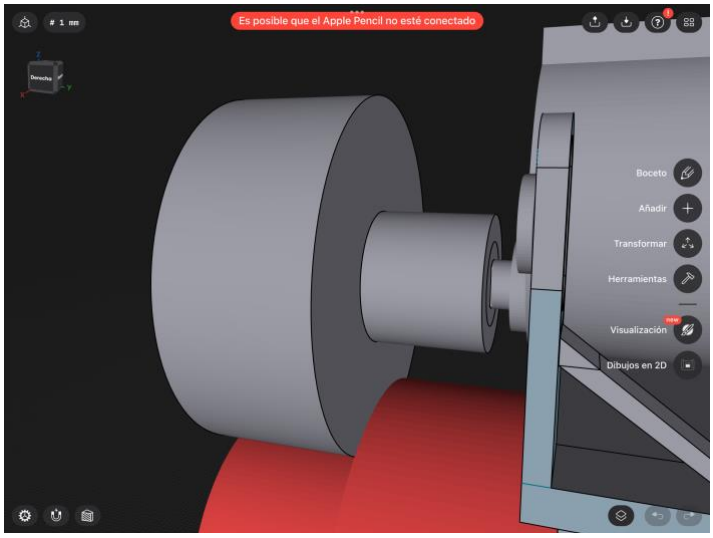

Ilustración 39 - Acople motor - telescopio

6. Construcción enfocador

En este apartado se detalla el proceso de impresión utilizando una impresora 3D, la cual es la propia del departamento al que pertenece el profesor tutor. Además del proceso de impresión, se detallará el proceso de montaje de los componentes en los modelos impresos.

En primer lugar, antes de comenzar con los inconvenientes encontrados a la hora de realizar la impresión de los modelos y detallar el resultado final de la construcción, se detallará el proceso para la impresión en 3D.

Para imprimir en 3D un objeto, antes se debe crear su modelo en formato STL, este modelo se ha creado utilizando la aplicación Shapr3D, anteriormente mencionada.

Una vez se ha creado el diseño, se debe exportar para utilizar un *slicer* y general un archivo de extensión GCODE. Este archivo será el encargado de almacenar la información sobre el objeto, el cual leerá la impresora 3D, puesto que el archivo describe cómo la impresora deberá realizar la impresión. Este archivo almacena toda la información para realizar la impresión, tales son como por ejemplo la temperatura o la posición, entre otros.

Se han producido diversos inconvenientes para la impresión de los modelos. Por ejemplo, la base es un modelo junto con los laterales y, dado un proceso de unión en la aplicación utilizada, el programa para crear la impresión no lo reconoce de forma correcta y la base no aparece, únicamente aparecen los laterales. Así como el soporte del motor, en el cual no es posible obtener el objeto completo, encontrando una serie de errores en las operaciones de unión y sustracción.

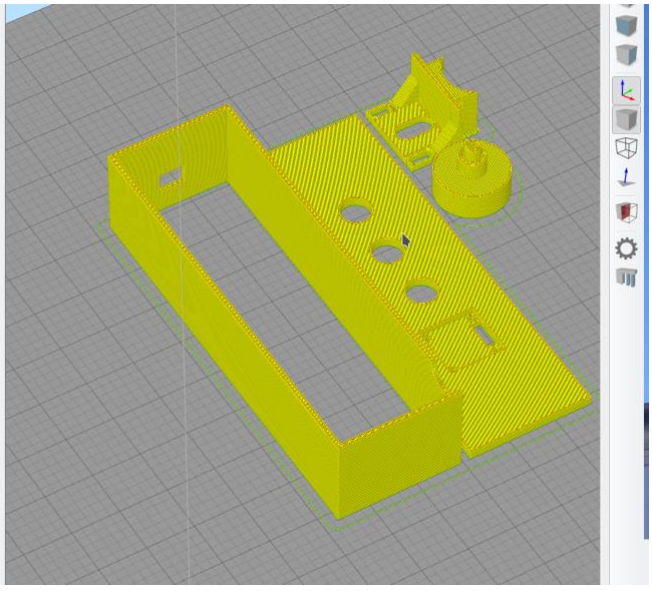

Ilustración 40 - Error en la base

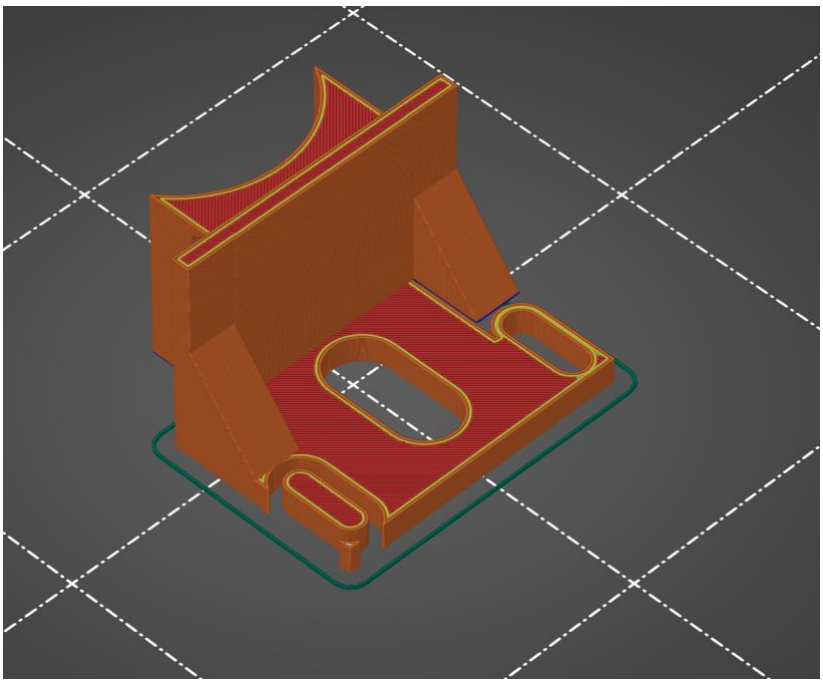

Ilustración 41 - Error en el soporte del motor

Tras la obtención de estos errores se han modificado los modelos 3D de los objetos, creándolos de una forma distinta.

Esta vez, se ha utilizado una página web para la obtención del archivo en formato GCODE [10] y otra página para la visualización del archivo [11] y la comprobación de que no exista ningún error como sí que ocurría en los modelos anteriores. Se ha utilizado dos páginas para visualizar el archivo GCODE [12].

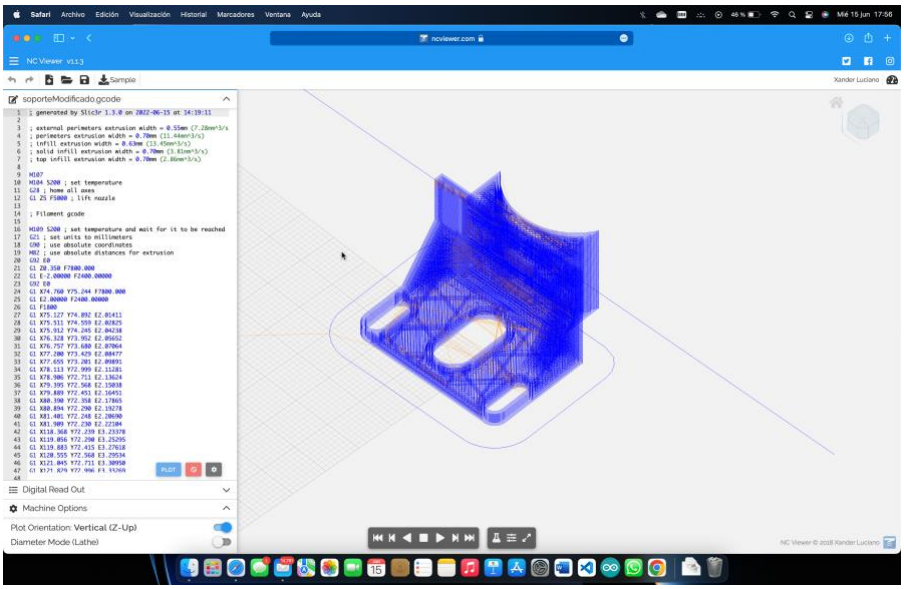

Ilustración 42 - Visualización del archivo GCODE

6.1. Base

La base del sistema central es la pieza más sencilla, puesto que únicamente es una pieza rectangular en la cual se encuentran la placa de conexiones y el módulo controlador del motor paso a paso. Es por esto por lo que, dada su sencillez, se trata de la pieza más sencilla para imprimir y colocar las piezas en ella.

Ilustración 43 - Base impresa con placa de conexiones y módulo controlador de motor

6.2. Laterales

Los laterales constan de cuatro rectángulos que servirán de paredes para completar junto con la base y la tapa, el sistema central. Dos de los laterales son rectángulos simples, mientras que los otros dos tienen cada uno un agujero. Uno de los laterales tiene el agujero para que pase el cable de alimentación, mientras que el otro lateral tiene el agujero para que salgan los cables del motor.

Ilustración 44 - Lateral con hueco para el paso del cable

Ilustración 45 - Lateral con hueco para salida de cables del motor

6.3. Tapa

La tapa es la pieza encargada de contener los tres pulsadores empleados más la pantalla. Una vez impresa la pieza correspondiente a la pantalla, se realiza la prueba para comprobar que el modelo es correcto y todos los componentes encajan de la forma adecuada.

Ilustración 46 - Tapa con los pulsadores y la pantalla colocados

6.4. Pulsadores

Los pulsadores se encuentran en la tapa. Los tres pulsadores se encuentran junto a la pantalla OLED. Cada pulsador se ha ubicado para que el uso del sistema sea sencillo, teniendo una distribución coherente, donde se encuentran el pulsador de bajar a la izquierda, el de subir en el medio y el de aceptar en la derecha.

Los pulsadores se fijarán en la tapa utilizando tuerca que tienen incorporada, con la que se quedarán fijados a la tapa, impidiendo la separación de los pulsadores con ésta.

6.5. Pantalla

La pantalla se ubica también en la tapa y, para ello, se ha creado un hundimiento para que, al colocar la pantalla, quede al nivel de la tapa. Se han añadido una serie de agujeros para que la pantalla se pueda fijar con tornillos a la tapa y no sea posible que la pantalla se separe de la tapa.

6.6. Motor paso a paso

El motor paso a paso se encuentra separado, montado sobre el telescopio utilizando el soporte creado para ello. En este caso el motor se ha colocado sobre el soporte en una posición intermedia para que sea posible un acople óptimo con el telescopio.

6.7. Soporte motor

El soporte del motor es la pieza encargada de sostener el motor paso a paso, fijándolo sobre el telescopio. La fijación sobre el telescopio no se ha realizado de la mejor manera, puesto que lo mejor es crear una pieza que encaje y no sea necesario utilizar unas bridas, cuerda o cualquier otro material que desee el usuario.

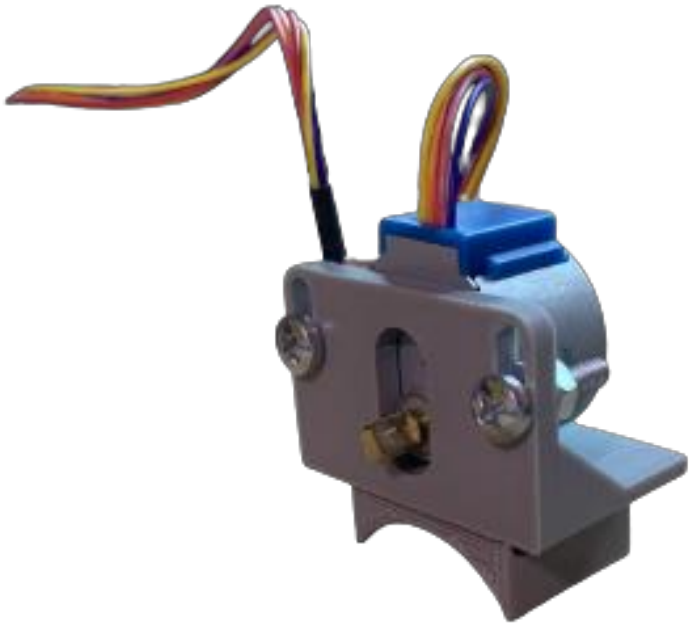

Ilustración 47 - Motor colocado en el soporte

6.8. Acople motor - telescopio

La rueda de acople del motor con el telescopio es la pieza encargada de transmitir el movimiento del motor a la rueda de enfoque del telescopio.

La rueda empleada tiene un tamaño, sin embargo, es posible crear una rueda distinta de otro material, color o tamaño para que el usuario pueda utilizar en otro telescopio o, simplemente por estética tener distintas ruedas, cada una de un color diferente.

A la rueda creada se le ha colocado un fragmento de cámara de goma de una rueda de bicicleta, con el fin de aumentar el agarre y provocar que la rueda de enfoque no patine.

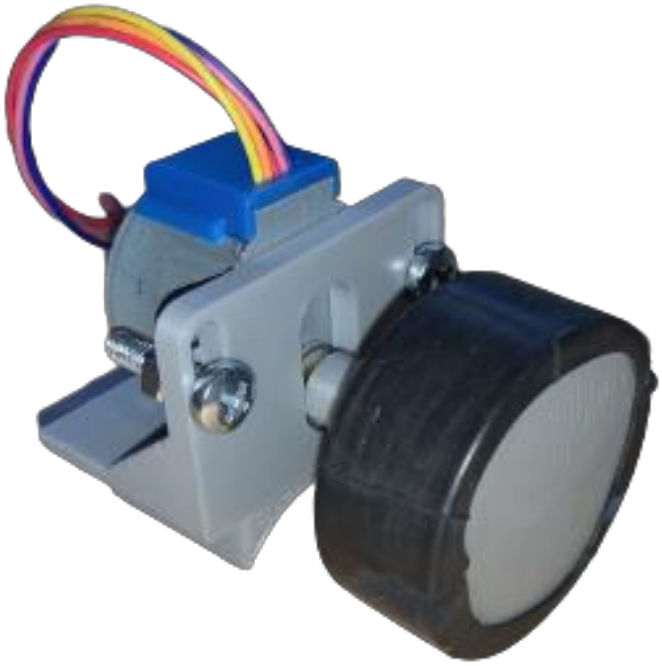

Ilustración 48 - Acople motor - telescopio colocado en el motor

6.9. Resultado final

El resultado final obtenido es el esperado, en el cual todas las piezas encajan de forma correcta y el programa tiene un funcionamiento adecuado, con unas funcionalidades esperadas para un sistema de enfoque.

El motor se ha ubicado en la posición deseada sin problemas, con lo que se ha conseguido que la rueda de enfoque sea posible moverla utilizando el motor paso a paso.

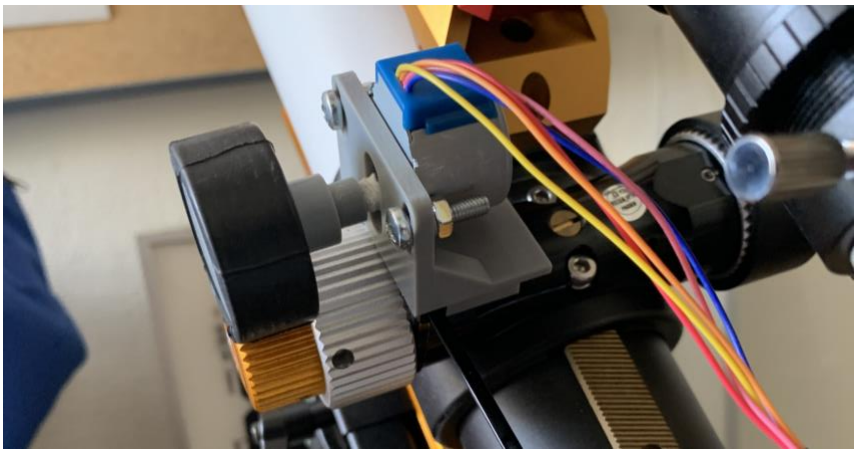

Ilustración 49 - Motor paso a paso colocado en su posición

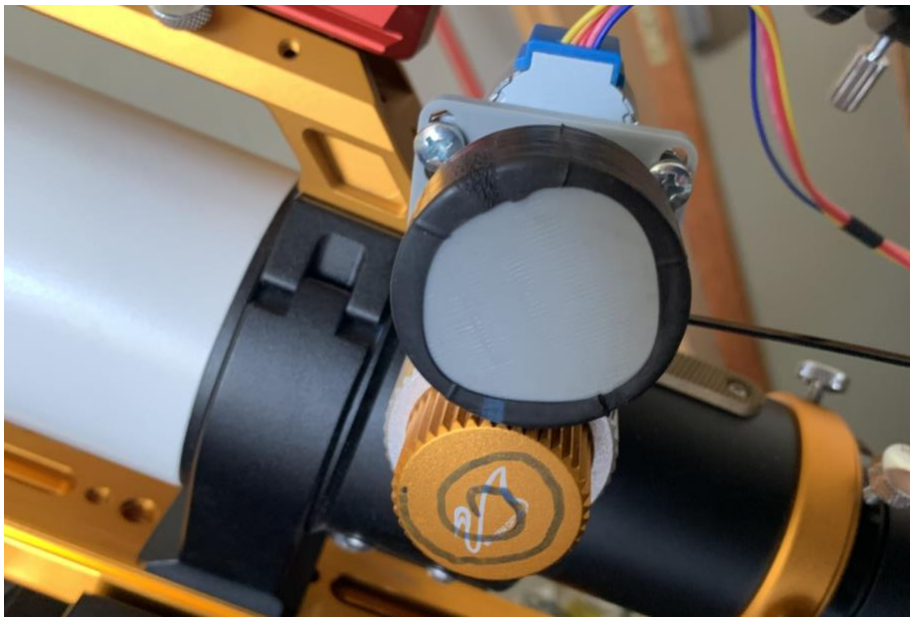

Ilustración 51 - Motor colocado en su posición con la rueda de acople bien colocada

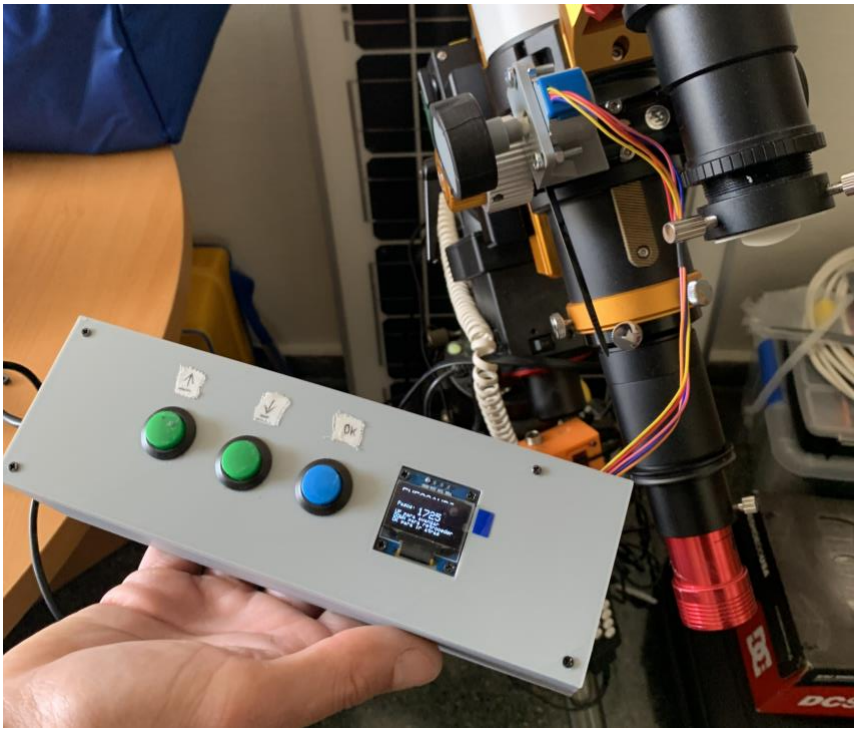

Ilustración 50 - Sistema central conectado al motor y funcional

7. Código

El código del programa está escrito en C++, ya que es el lenguaje de programación estándar para las placas Arduino. El programa se divide en dos bloques, el control por puerto serial, mediante la comunicación a través del cable USB empleado para la alimentación además de la comunicación entre ordenador y microcontrolador; y el bloque de control utilizando los pulsadores físicos.

El primer bloque consiste en el control del sistema de enfoque utilizando el ordenador a través del puerto serie. Utilizando el ordenador se podrán realizar distintas acciones: avanzar una cantidad de pasos dada por el usuario, avanzar o retroceder hasta el paso indicado por el usuario, seleccionar la velocidad de giro del motor paso a paso y, por último, restablecer la cantidad de pasos dados a cero.

Para las tres primeras opciones se muestra por pantalla un segundo mensaje indicando, en función de la elección, los pasos para proceder. Tras indicar los pasos a mover el motor o, en su defecto, la nueva velocidad del motor, la pantalla se actualizará mostrando los nuevos pasos o velocidad y en el ordenador aparecerá un mensaje indicando los pasos efectuados, así como la velocidad del motor.

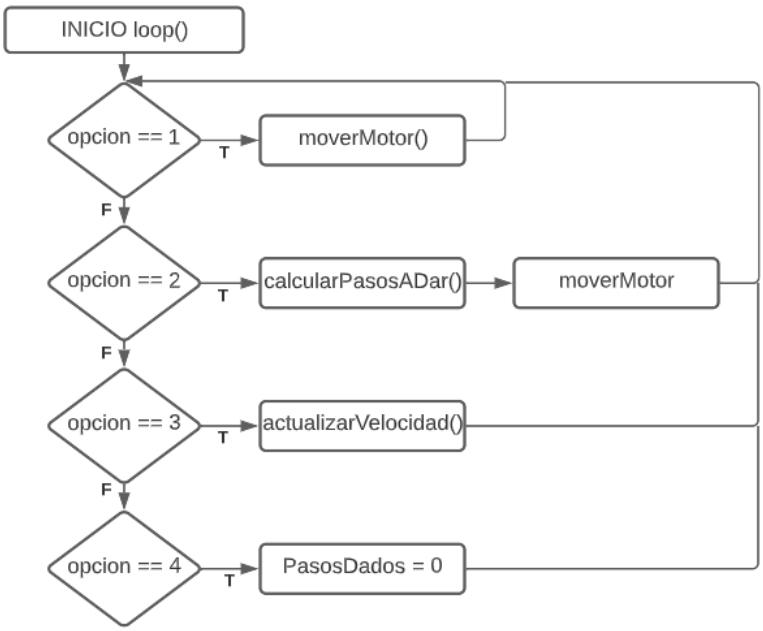

Ilustración 52 - Diagrama de flujo bloque de código 1

A continuación, en el segundo bloque, se hace uso únicamente de los pulsadores físicos empleados en el proyecto, mediante los cuales se realiza la interacción con el sistema. El botón de aceptar será el botón con el cual se puede entrar a las distintas pantallas, así como volver atrás en aquellas pantallas en las que se realizan cambios en ajustes como por ejemplo en el ajuste de la velocidad, así como en la pantalla del modo enfoque, donde al pulsar el botón de *OK* se vuelve a la pantalla anterior.

En la primera pantalla aparece la opción del mofo enfoque, así como la posibilidad de entrar a un menú para ajustar manualmente un enfoque por parámetros. En el modo enfoque se habilita la posibilidad de mover el motor en un sentido u otro en función del botón presionado. El motor se mueve mientras el pulsador se encuentre en posición pulsada.

En la segunda pantalla, el menú para configurar el enfoque por parámetros hay cuatro opciones: ajustar pasos a dar, ajustar velocidad de rotación del motor, enfocar y salir. La opción de ajustar velocidad del motor se puede utilizar para modificar la velocidad de giro del motor durante el modo enfoque. La velocidad por defecto del motor es de diez revoluciones por minuto, teniendo un máximo de 13 revoluciones por minuto y un mínimo de una revolución por minuto.

En la opción de enfocar se realiza el movimiento del motor utilizando los parámetros introducidos en el resto de las opciones. Si los pasos no se modifican, cada vez que se pulse sobre la opción de enfocar, el motor se moverá la misma cantidad de pasos. En el momento en el que se modifiquen los pasos a dar, el motor avanzará o retrocederá la nueva cantidad de pasos indicados.

Por último, mediante la opción de salir, la pantalla retrocederá y se volverá a la pantalla principal, donde se encuentra la opción del modo enfoque y la opción para entrar nuevamente en la pantalla del enfoque por parámetros.

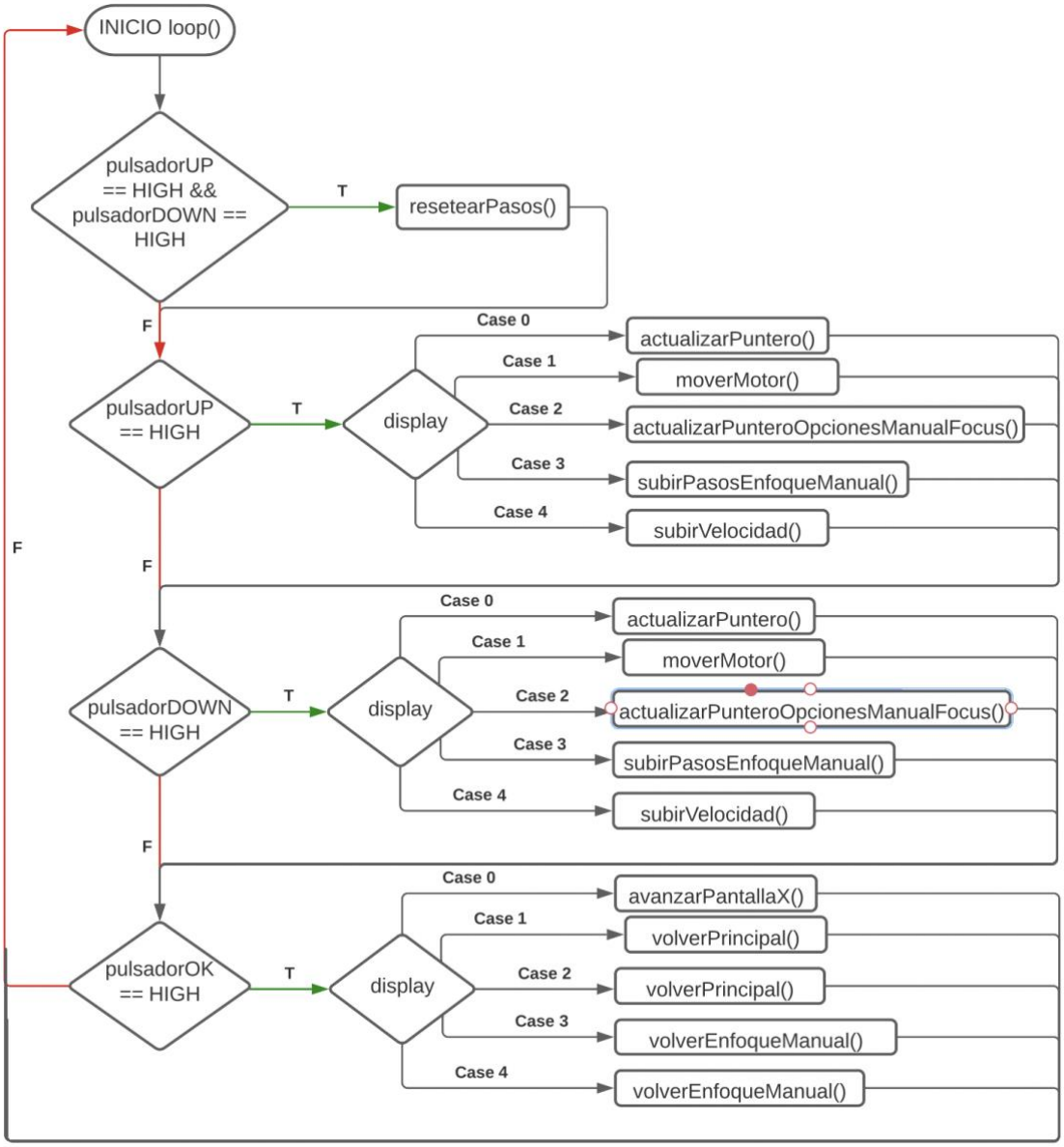

Ilustración 53 - Diagrama de flujo bloque de código 2

Para añadir, se ha requerido el uso de la macro "F()" [13], mediante la cual se consigue que las constantes se almacenen en la memoria *flash* en lugar de la *SRAM,* con lo que se obtiene un uso de la *SRAM* inferior, como por ejemplo, la constante "firstOption", una constante que almacena la primera opción que aparecerá en la pantalla. Por el contrario, la cantidad de memoria *flash* empleada, aumenta significativamente.

```
"C:\\Program Files (x86)\\Arduino\\hardware\\tools\\avr/bin/avr-size" -A "C:\\Users\\lande\\AppData\\Local\\Temp\\arduino_build_742045/enfocadorV2.ino.elf"
El Sketch usa 20714 bytes (67%) del espacio de almacenamiento de programa. El máximo es 30720 bytes.
Las variables Globales usan 605 bytes (29%) de la memoria dinámica, dejando 1443 bytes para las variables locales. El máximo es 2048 bytes.
```

Ilustración 54 - Cantidad de memoria utilizada

8. Conclusiones y proyectos futuros

Para concluir con el trabajo, se expresa la consecución de los objetivos por parte del alumno, puesto que los tres objetivos se han completado.

En primer lugar, se ha desarrollado el programa capaz de implantarse en el microcontrolador Arduino NANO. La limitación en cuando a memoria ha sido un elemento muy determinante para el desarrollo. La memoria *flash* no ha sido un impedimento, mientras que la *SRAM*, teniendo una capacidad de 2KB.

El segundo objetivo también se ha cumplido, puesto que la solución desarrollada es sencilla de utilizar. Las opciones son sencillas de comprender y es intuitivo el uso de este. En cuanto a la facilidad utilizando el ordenador es todavía mayor, ya que las opciones son más directas y, tal y como se ha mencionado anteriormente, facilitaría el control remoto del enfoque. No es necesario presionar botones para ajustar el motor, basta con indicar los pasos a mover o, en su defecto, el paso hasta el que se desea llegar.

Además de esto, el tercer objetivo también se ha conseguido, debido a que, de los dos bloques de funcionamiento del programa, uno es el dedicado al manejo del sistema empleando el puerto serie.

Por último, el cuarto objetivo se ha logrado cumplir puesto que los diseños creados han sido del tamaño adecuado, encajando todas las piezas de la forma correcta, facilitando su montaje en el sistema.

En cuanto a proyectos futuros, cabe recalcar que existe una gran cantidad de modificaciones que se pueden realizar en el proyecto. Estas modificaciones mejorarán la apariencia, dando una impresión de tratarse de un proyecto más profesional.

Para comenzar, se puede optar por crear un circuito impreso, utilizando una lámina de cobre sobre la que se creará el diseño del circuito.

Existe una diversidad de formas para realizar esta placa de circuito impreso. Se pueden adquirir unas placas con agujeros ya hechos donde se colocan las conexiones de los componentes para soldarlos.

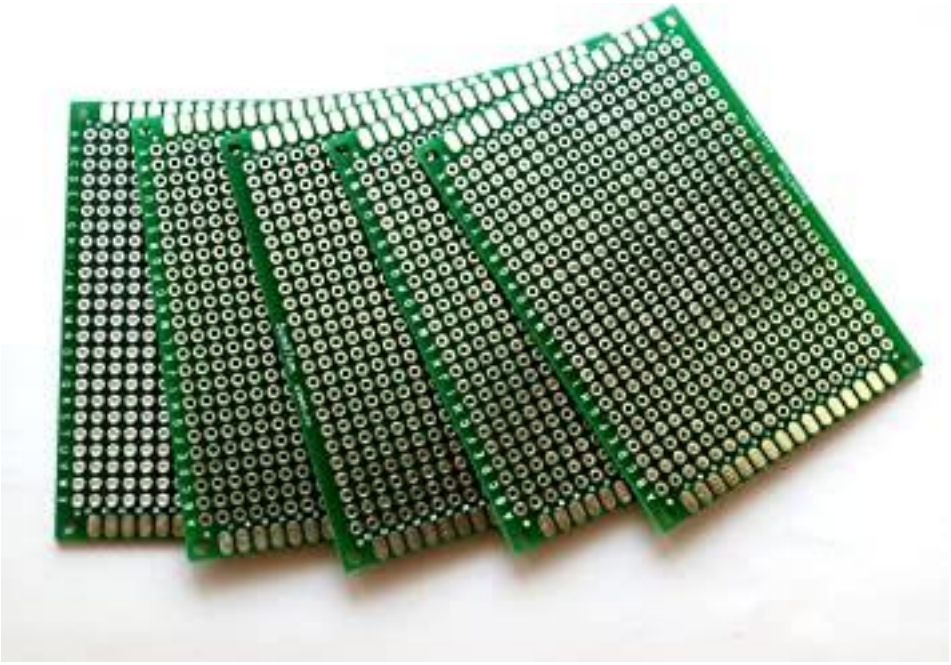

Ilustración 55 - PCB para conexiones

Otra de las opciones y, por otro lado, más artesanal es, utilizando una placa de cobre, utilizar un programa para diseñar el circuito. Una vez diseñado e impreso en un papel el circuito, se dibuja sobre la placa de cobre para, posteriormente, sumergirla en una disolución ácida y eliminar el cobre no deseado, dejando únicamente el cobre para el circuito.

Por otro lado, también es posible dibujar directamente el circuito con un rotulador. Esto es posible pero, a su vez, requiere más experiencia y tener una idea muy clara del circuito para no equivocarse.

De esta forma, se consigue un resultado más profesional visualmente, así como funcionalmente, puesto que es una solución más definitiva que la placa de conexiones utilizada, donde las conexiones pueden soltarse de una forma mucho más fácil.

Ilustración 56 - Circuito impreso casero

Otra modificación para mejorar el proyecto consiste en mejorar los diseños 3D. Para ello, se puede modificar el soporte para el motor, creado un nuevo diseño para facilitar el acople con el telescopio, creando una figura que complete la circunferencia, acoplándose con el diseño original mediante unas pestañas para que el resultado sea una pieza más firme.

También es posible modificar la caja del sistema central, mejorando la fijación de la tapa. Es posible crear unos agujeros en las esquinas para pasar unos tornillos o, en caso de optar por otra solución, preparar una de las paredes y la tapa para colocar una bisagra y poder abrir y cerrar la tapa para acceder al circuito de una forma más sencilla y rápida.

Para terminar la explicación de posibles proyectos futuros, existe la posibilidad de cambiar el microcontrolador por otro distinto, por ejemplo, un ESP8266 [14], un microcontrolador con unas características similares a las del Arduino empleado, salvo que utiliza 3,3V en lugar de 5V. utilizando esta placa se logra abaratar los costes del proyecto, puesto que el precio de esta placa es inferior al Arduino NANO.

Asimismo, se podría utilizar la placa mencionada anteriormente, la ESP8266, puesto que dispone de conexión WI-FI, así como el Arduino UNO WIFI. Con este cambio de microcontrolador se consigue un proyecto mediante el cual se podrá conectar la placa a internet, dando la posibilidad de automatizar en una escala mayor. Este cambio, se puede utilizar por ejmplo para automatizar el sistema de enfoque de un observatorio.

9. Referencias

[1] «SolectroShop,» 25 03 2020. [En línea]. Available: https://solectroshop.com/es/blog/comparacion-de-placas-arduino-con-recomendaciones-de-solectroshop-n5.

[2] «LuisLlamas,» 13 08 2016. [En línea]. Available: https://www.luisllamas.es/motor-paso-paso-28byj-48-arduino-driver-uln2003/. [Último acceso: 10 06 2022].

[3] «Shapr3D,» [En línea]. Available: https://www.shapr3d.com.

[4] «Arduino,» [En línea]. Available: https://www.arduino.cc/en/Guide.

[5] «Fritzing,» [En línea]. Available: https://fritzing.org/.

[6] «GitHub,» [En línea]. Available: https://github.com.

[7] L. d. V. Hernández, «ProgramarFacil,» [En línea]. Available: https://programarfacil.com/blog/arduino-blog/resistencia-pull-up-y-pull-down/. [Último acceso: 24 04 2022].

[8] L. d. V. Hernández, «ProgramarFacil,» 04 05 2022. [En línea]. Available: https://programarfacil.com/blog/motor-paso-a-paso/.

[9] L. d. V. Hernández, «ProgramarFacil,» [En línea]. Available: https://programarfacil.com/blog/arduino-blog/ssd1306-pantalla-oled-con-arduino/. [Último acceso: 28 04 2022].

[10] «CNCApps,» [En línea]. Available: https://cnc-apps.com/en/app/stl2gcode.

[11] «NCViewer,» [En línea]. Available: https://ncviewer.com/.

[12] «GCODEViewer,» [En línea]. Available: https://gcode.ws.

[13] J. Lewis, «BaldEngineer,» 20 11 2013. [En línea]. Available: https://www.baldengineer.com/arduino-f-macro.html. [Último acceso: 15 05 2022].

[14] «ElOsciloscopio,» [En línea]. Available: https://elosciloscopio.com/comparacion-arduino-vs-esp8266-vs-esp32/#Premio..

[15] «LucidChart,» [En línea]. Available: https://lucid.app/.

[16] «FileInfo,» 03 07 2018. [En línea]. Available: https://fileinfo.com/extension/gcode.

[17] «Dyor,» [En línea]. Available: http://dyor.roboticafacil.es/creacion-de-circuitos-impresos-con-medios-caseros/. [Último acceso: 30 05 2022].

10. Anexo

10.1. Código fuente

```
#include <Stepper.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

/*============================
        VARIABLES
============================*/
#define ANCHO 128
#define ALTO 64

#define firstOption "Modo enfoque"
#define selectedFirstOption ">Modo enfoque"
#define secondOption "Enfoque manual"
#define selectedSecondOption ">Enfoque manual"
//#define thirdOption "Enfoque manual"
//#define selectedThirdOption ">Enfoque manual"
#define stepsOption "Ajustar pasos"
#define selectedStepsOption ">Ajustar pasos"
#define speedOption "Ajustar velocidad"
#define selectedSpeedOption ">Ajustar velocidad"
#define focusOption "Enfocar"
#define selectedFocusOption ">Enfocar"
#define exitOption "Salir"
#define selectedExitOption ">Salir"

Stepper motor(2048, 8, 10, 9, 11);    //motor paso a paso
Adafruit_SSD1306 oled(ANCHO, ALTO, &Wire, -1);    //pantalla

int selected = 1;   //only works in initial display / 1 -> modo enfoque / 2 -> ajuste manual / 3 -> antiguo enfoque manual
int selectedManualFocus = 1;  // 1 -> ajustar pasos / 2 -> ajustar velocidad / 3 -> enfocar / 4 -> salir
int display = 0;    // 0 -> initial / 1 -> modo enfoque / 2 -> ajustar posicion 0 / 3 -> ajuste manual enfoque
int displaySettings = 0;  // 0 -> initial / 1 -> steps to move / 2 -> speed in RPM / 3 -> focus
```

```
int pulsadorUP = 6, pulsadorDOWN = 5, pulsadorOK = 3;
int stepsDone = 0, speed = 10;    //10 RPM by default
int stepsToDo = 0;  //steps defined in manual focus
int stepsFocusing = 0;  //steps to calculate with the map function
int selectedSerial; //selection made from serial port
float stepsSerial;  //steps read by serial port
boolean focusMode = false;
boolean printed = false;
boolean fromDisplay0 = false;
boolean display1Serial = false, display2Serial = false, display3Serial = false; //to control which display we
are

/*========================
      MAIN PROGRAM
  ========================*/
void setup() {
  pinMode(pulsadorUP, INPUT);
  pinMode(pulsadorDOWN, INPUT);
  pinMode(pulsadorOK, INPUT);
  Serial.begin(9600);

  Wire.begin();
  oled.begin(SSD1306_SWITCHCAPVCC, 0X3C);
  motor.setSpeed(10);

  oled.clearDisplay();
  setDisplayX();
}

void loop() {
  stepsFocusing = map(speed, 1, 13, 10, 150);
  if ( !printed ) { printOptionsSerial(); printed = true; }
  if ( Serial.available() > 0  && !display1Serial && !display2Serial && !display3Serial) {
    selectedSerial = Serial.parseFloat();
    if ( selectedSerial == 1 ) printOption1Serial();
    if ( selectedSerial == 2 ) printOption2Serial();
    if ( selectedSerial == 3 ) printOption3Serial();
    if ( selectedSerial == 4 ) { stepsDone = 0; refreshDisplay(); }
  }
  if ( Serial.available() > 0 && display1Serial ) {
```

```
  stepsSerial = Serial.parseFloat();
  if ( stepsSerial != 0 ) {
      stepsDone += (int) stepsSerial;
      moveMotor((int) stepsSerial);
      display1Serial = false;
      printed = false;
      Serial.print(F("Pasos dados: "));
      Serial.println((int) stepsSerial);
      Serial.print(F("Pasos actuales: "));
      Serial.println(stepsDone);
      refreshDisplay();
  }
}
if ( Serial.available() > 0 && display2Serial ) {
  stepsSerial = Serial.parseFloat();
  if ( stepsSerial != 0 ) {
      int stepsToDoSerial = (int) stepsSerial - stepsDone;
      stepsDone += stepsToDoSerial;
      moveMotor(stepsToDoSerial);
      display2Serial = false;
      printed = false;
      Serial.print(F("Pasos dados: "));
      Serial.println(stepsToDoSerial);
      Serial.print(F("Pasos actuales: "));
      Serial.println(stepsDone);
      refreshDisplay();
  }
}
if ( Serial.available() > 0 && display3Serial ) {
  int speedSerial = Serial.parseInt();
  if ( speedSerial != 0 ) {
      if ( speedSerial >= 1 && speedSerial <= 13) {
          display3Serial = false;
          printed = false;
          speed = speedSerial;
          motor.setSpeed(speed);
          Serial.print(F("Pasos actuales: "));
          Serial.println(stepsDone);
          Serial.print(F("Velocidad: "));
```

```
          Serial.println(speed);
          refreshDisplay();
        }
      }
    }
  if ( HIGH == digitalRead(pulsadorUP) && HIGH == digitalRead(pulsadorDOWN)) setDisplayResetSteps();
  if ( HIGH == digitalRead(pulsadorDOWN) )  pulsadoDOWN();
  if ( HIGH == digitalRead(pulsadorUP) ) pulsadoUP();
  if ( HIGH == digitalRead(pulsadorOK) ) pulsadoOK();
}

/*========================
        CHECKS
  ========================*/

void pulsadoUP() {
  if ( display == 0 && selected > 1 ) {
    selected--;
    setDisplayX();
  }
  if ( display == 1 ) {
    while( HIGH == digitalRead(pulsadorUP) ) {
```

```
      moveMotor(stepsFocusing);
      stepsDone += stepsFocusing;
    }
    setDisplayFocus();
  }
  if ( display == 2 ) moveMotor(1);
  if ( display == 3 && selectedManualFocus > 0 ) {
    selectedManualFocus--;
    setDisplayManualFocusX();
  }
  if ( display == 4 ) {
    stepsToDo++;
    setDisplaySetManualSteps();
  }
  if ( display == 5 && speed < 13 ) {
    speed++;
    setDisplaySetManualSpeed();
  }
  delay(100);
}

void pulsadoDOWN() {
  if ( display == 0 && selected < 2 ) {
    selected++;
    setDisplayX();
  }
  if ( display == 1 )  {
    while( HIGH == digitalRead(pulsadorDOWN) ) {
      moveMotor(-stepsFocusing);
      stepsDone -= stepsFocusing;
    }
    setDisplayFocus();
  }
  if ( display == 2 ) moveMotor(-1);
  if ( display == 3 && selectedManualFocus < 4 ) {
    selectedManualFocus++;
    setDisplayManualFocusX();
  }
```

```
 if ( display == 4 ) {
   stepsToDo--;
   setDisplaySetManualSteps();
 }
 if ( display == 5 && speed > 1 ) {
   speed--;
   setDisplaySetManualSpeed();
 }
 delay(100);
}

void pulsadoOK() {
 if ( display == 0 && selected == 1 ) {
   display = 1;
   focusMode = true;
   setDisplayFocus();
 }
 if ( display == 0 && selected == 2 ) {
   display = 3;
   fromDisplay0 = true;
   setDisplayManualFocusX();
 }
 /*if ( display == 0 && selected == 3 ) {
   display = 3;
   fromDisplay0 = true;
   setDisplayManualFocusX();
 }*/
 if ( display == 1 ) {
   if ( !focusMode ) {
     selected = 1;
     display = 0;
     setDisplayX();
   } else focusMode = false;
 }
 /*if ( display == 2 ) {
   if ( !focusMode ) {
     stepsDone = 0;
     selected = 2;
     display = 0;
```

```
    setDisplayX();
  } else focusMode = false;
 }*/
 if ( display == 3 && selectedManualFocus == 1 ) {
  if ( fromDisplay0 == false ) {
      display = 4;
      selectedManualFocus = 1;
      fromDisplay0 = true;
      setDisplaySetManualSteps();
  } else fromDisplay0 = false;
 }

 if ( display == 3 && selectedManualFocus == 2 ) {
  display = 5;
  selectedManualFocus = 1;
  fromDisplay0 = true;
  setDisplaySetManualSpeed();
 }
 if ( display == 3 && selectedManualFocus == 3 ) {
  moveMotor(stepsToDo );
  stepsDone += stepsToDo;
  setDisplayManualFocusX();
 }
 if ( display == 3 && selectedManualFocus == 4 ) {
  display = 0;
  selectedManualFocus = 1;
  selected = 2;
  setDisplayX();
 }
 if ( display == 4 || display == 5 ) {
  if ( fromDisplay0 == false ) {
      display = 3;
      motor.setSpeed(speed);
      setDisplayManualFocusX();
  } else fromDisplay0 = false;
 }
 delay(100);
}
```

```
/*========================
        DISPLAY
  ========================*/

/**
   Method that prints the options in the serial port
*/
void printOptionsSerial() {
  Serial.println(F("FOCUSER by David Landete"));
  Serial.println(F("Introduce la opción deseada:"));
  Serial.println(F("1 -> Avanzar X pasos"));
  Serial.println(F("2 -> Avanzar hasta el paso X"));
  Serial.println(F("3 -> Ajustar la velocidad"));
  Serial.println(F("4 -> Restablecer pasos a 0"));
}

void printOption1Serial() {
    Serial.println(F("Avanzar X pasos. Introduce el número de pasos a avanzar"));
    display1Serial = true;
    display2Serial = false;
    display3Serial = false;
}

void printOption2Serial() {
    Serial.println(F("Avanzar hasta el paso X. Introduce el paso hasta el cual llegar"));
    display1Serial = false;
    display2Serial = true;
    display3Serial = false;
}

void printOption3Serial() {
    Serial.println(F("Modificar velocidad en RPM. Introduce la velocidad [1-13]"));
    display1Serial = false;
    display2Serial = false;
    display3Serial = true;
}

void refreshDisplay() {
    switch ( display ) {
        case 0: setDisplayX(); break;
```

```
      case 1: setDisplayFocus(); break;
      case 3: setDisplayManualFocusX(); break;
      case 4: setDisplaySetManualSteps(); break;
      case 5: setDisplaySetManualSpeed(); break;
    }
}

void setHeader() {
  oled.clearDisplay();
  oled.setTextColor(WHITE);
  oled.setCursor(0, 0);
  oled.setTextSize(2);
  oled.print(stepsDone);
  oled.println(F("p"));
  oled.print(F("RPM:"));
  oled.println(speed);
  oled.setTextSize(1);
}

void setDisplay1() {
  setHeader();
  oled.setCursor(0, 35);
  oled.println(F(selectedFirstOption));
  oled.println(F(secondOption));
  //oled.println(F(thirdOption));
  oled.display();
}




void setDisplay2() {
  setHeader();
  oled.setCursor(0, 35);
  oled.println(F(firstOption));
  oled.println(F(selectedSecondOption));
  //oled.println(F(thirdOption));
  oled.display();
```

```
}
/*
void setDisplay3() {
  setHeader();
  oled.setCursor(0, 35);
  oled.println(F(firstOption));
  oled.println(F(secondOption));
  oled.println(F(selectedThirdOption));
  oled.display();
}
*/
void setDisplayX() {
  switch (selected) {
    case 1: setDisplay1(); break;
    case 2: setDisplay2(); break;
    //case 3: setDisplay3(); break;
  }
}

void setDisplayFocus() {
  oled.clearDisplay();
  oled.setCursor(0, 0);
  oled.setTextSize(2);
  oled.print("ENFOCANDO");
  oled.setCursor(0, 25);
  oled.setTextSize(1);
  oled.print(F("Pasos: "));
  oled.setTextSize(2);
  oled.println(stepsDone);
  oled.setTextSize(1);
  oled.println(F("UP para avanzar"));
  oled.println(F("DOWN para retroceder"));
  oled.println(F("OK para ir atras"));
  oled.display();
}
/*
void setDisplayStep0() {
  oled.clearDisplay();
  oled.setCursor(0, 0);
  oled.setTextSize(2);
```

```
  oled.print("PASO 0");
  oled.setCursor(0, 25);
  oled.setTextSize(1);
  oled.println("UP para avanzar");
  oled.println("DOWN para retroceder");
  oled.println("OK para establecer\nel paso 0");
  oled.display();
  delay(100);
}
*/
void setDisplayManualFocusX() {
  switch ( selectedManualFocus ) {
    case 1: setDisplayManualSteps(); break;
    case 2: setDisplayManualSpeed(); break;
    case 3: setDisplayManualFocus(); break;
    case 4: setDisplayManualExit(); break;
  }
}

void setDisplayManualSteps() {
  setHeader();
  oled.println(F(selectedStepsOption));
  oled.println(F(speedOption));
  oled.println(F(focusOption));
  oled.println(F(exitOption));
  oled.display();
}

void setDisplayManualSpeed() {
  setHeader();
  oled.println(F(stepsOption));
  oled.println(F(selectedSpeedOption));
  oled.println(F(focusOption));
  oled.println(F(exitOption));
  oled.display();
}

void setDisplayManualFocus() {
  setHeader();
```

```
  oled.println(F(stepsOption));
  oled.println(F(speedOption));
  oled.println(F(selectedFocusOption));
  oled.println(F(exitOption));
  oled.display();
}

void setDisplayManualExit() {
  setHeader();
  oled.println(F(stepsOption));
  oled.println(F(speedOption));
  oled.println(F(focusOption));
  oled.println(F(selectedExitOption));
  oled.display();
}

void setHeaderManualSets() {
    oled.clearDisplay();
    oled.setCursor(0, 0);
    oled.setTextSize(2);
    oled.println(F("AJUSTE\nMANUAL"));
    oled.setCursor(0, 35);
}

void setSignManualSets() {
    oled.setTextSize(1);
    oled.setCursor(0, 55);
    oled.println("Pulsa OK para volver");
    oled.display();
}

void setDisplaySetManualSteps() {
    setHeaderManualSets();
    oled.print(F("Pasos:"));
    oled.println(stepsToDo);
    setSignManualSets();
}

void setDisplaySetManualSpeed() {
    setHeaderManualSets();
```

```
  oled.print(F("Vel.:"));
  oled.println(speed);
  setSignManualSets();
}

void setDisplayResetSteps() {
  oled.clearDisplay();
  oled.setCursor(0, 0);
  oled.setTextSize(2);
  oled.println(F("RESET\nDE\nPASOS"));
  oled.setTextSize(1);
  oled.display();
  stepsDone = 0;
  delay(200);
}

/*========================
       MOTOR
  ========================*/

void moveMotor(int stepsToMove) {
  motor.step(stepsToMove);
  delay(50);
}
```

10.2. OBJETIVOS DE DESARROLLO SOSTENIBLE

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenibles	**Alto**	**Medio**	**Bajo**	**No Procede**
ODS 1. **Fin de la pobreza.**				**X**
ODS 2. **Hambre cero.**				**X**

ODS 3. **Salud y bienestar.**				**X**
ODS 4. **Educación de calidad.**				**X**
ODS 5. **Igualdad de género.**				**X**
ODS 6. **Agua limpia y saneamiento.**				**X**
ODS 7. **Energía asequible y no contaminante.**				**X**
ODS 8. **Trabajo decente y crecimiento económico.**				**X**
ODS 9. **Industria, innovación e infraestructuras.**	**X**			
ODS 10. **Reducción de las desigualdades**.				**X**
ODS 11. **Ciudades y comunidades sostenibles**.				**X**
ODS 12. **Producción y consumo responsables.**				**X**
ODS 13. **Acción por el clima.**				**X**
ODS 14. **Vida submarina.**				**X**
ODS 15. **Vida de ecosistemas terrestres.**				**X**
ODS 16. **Paz, justicia e instituciones sólidas.**				**X**
ODS 17. **Alianzas para lograr objetivos.**				**X**

Reflexión sobre la relación del TFG/TFM con los ODS y con el/los ODS más relacionados.

Los objetivos de desarrollo sostenibles (ODS) son una serie de objetivos, en total 17, definidos por la Organización de las Naciones Unidas en 2015, cuando se aprobó la Agenda 2030. Se trata de una serie de objetivos de aplicación universal para impulsar el crecimiento económico, el compromiso con las necesidades sociales y la protección del medio ambiente.

El objetivo con el cual tiene mayor relación este trabajo de fin de grado es el nueve, en el cual se ve reflejada la industrialización inclusiva y sostenible. Esto es así, puesto que es posible la unión económica y de cara a una serie de objetivos entre distintas personas, tengan relación entre sí o no.

Se vela por la industria inclusiva, dado que el objetivo es promover la posibilidad de que cualquier persona pueda ser partícipe de este proyecto, ya sea mejorándolo personalmente para su caso o, todo lo contrario, utilizando los mismos modelos creados para este proyecto, siendo que encajan a la perfección con su caso y no es necesario modificar nada.

El objetivo principal por el que se cumple el objetivo es que cada persona puede aportar sus conocimientos para crear un proyecto con un nivel superior cada vez, hasta que podría llegar a ser un producto capaz de ser producido por una empresa para todas aquellas personas que deseen hacerse con uno. Esto es posible, ya que existe una gran cantidad de personas que disponen de un telescopio y, utilizando otro proyecto con el que se pueda capturar lo que se observa por el telescopio, otro proyecto para controlar el movimiento de este se pueda juntar con este para así crear también el sistema de enfoque.

Hoy en día, la cantidad de empresas que utilizan energías renovables se encuentra en alza, por lo que, si una empresa se encuentra interesada en crear un proyecto similar empleando energía renovable para su producción, se estará velando por el bienestar del planeta.

Asimismo, se desea que cualquier persona pueda ser capaz de participar en el proyecto, apostando para que las mujeres, en un menor, pero a su vez similar número en la industria tecnológica, se interesen en adentrarse en el mundo y conseguir cifras de mujeres más altas.

En conclusión, obtenemos un trabajo en el cual se vela por el crecimiento del sector manufacturero, puesto que una empresa puede decidir poner en producción el proyecto creado, dando la posibilidad a países menos desarrollados en este sector de experimentar un acelerón en el desarrollo de este sector.

Además de apostar por las energías renovables que, cada vez están más presentes en los métodos de alimentación de las empresas hoy en día. Empresas como Apple, quienes poseen el Apple Park, la sede de la empresa en Cupertino, teniendo el techo cubierto con una gran cantidad de paneles fotovoltaicos, produciendo 17 megavatios, junto con los situados en otras oficinas, tienen la capacidad de generar 393,3 megavatios en un año.

También, como conclusión obtenemos que se desea fomentar la inclusión, puesto que este proyecto está destinado al uso libre por cualquier persona que lo desee, sin importar quién sea.