UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE GUAYAQUIL

CARRERA DE ELECTRÓNICA Y AUTOMATIZACIÓN

DISEÑO E IMPLEMENTACIÓN DE UN ROBOT SEGUIDOR DE LÍNEA MEDIANTE VISIÓN ARTIFICIAL

Trabajo de titulación previo a la obtención del

Título de Ingeniero en Electrónica y Automatización

AUTOR(ES): Tom Andy Bohórquez Pinargote

Dustin Jean Martillo García

TUTOR: Ing. Víctor David Larco Torres, MSc.

GUAYAQUIL – ECUADOR

2023

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

Nosotros, Tom Andy Bohórquez Pinargote con cédula de identidad N° 0953785052 y Dustin Jean Martillo García con cédula de identidad N° 0954725263, expresamos lo siguiente:

Somos los autores y responsables quienes realizamos el presente trabajo, damos la autorización a la Universidad Politécnica Salesiana para que sin fines de lucro pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación.

Guayaquil, febrero 2023.

Atentamente,

Tom Andy Bohórquez Pinargote

C.I. 0953785052

Dustin Jean Martillo García

C.I. 0954725263

CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA

Nosotros, Tom Andy Bohórquez Pinargote con cédula de identidad N° 0953785052 y Dustin Jean Martillo García con cédula de identidad N° 0954725263, manifestamos nuestra voluntad, a través del presente documento le transferimos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos los autores del trabajo de grado intitulado: **"DISEÑO E IMPLEMENTACION DE UN ROBOT SEGUIDOR DE LINEA MEDIANTE VISION ARTIFICIAL"**, el cual ha sido desarrollado para obtener el título de: **INGENIERO EN ELECTRONICA Y AUTOMATIZACION**, quedando la Universidad facultada para ejercer plenamente los derechos transferidos anteriormente.

En concordancia, suscribimos este documento en el momento que hacemos entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

Guayaquil, febrero 2023.

Atentamente,

Tom Andy Bohorquez Pinargote

C.I. 0953785052

Dustin Jean Martillo García

C.I. 0954725263

CERTIFICADO DE DIRECIÓN DEL TRABAJO DE TITULACIÓN

Yo, MSc. Víctor Larco Torres con cédula de identidad N° 0923270136, docente de la Universidad Politécnica Salesiana, atestiguo que bajo mi tutela fue desarrollado el trabajo de titulación: **"DISEÑO E IMPLEMENTACION DE UN ROBOT SEGUIDOR DE LINEA MEDIANTE VISION ARTIFICIAL"**, realizado por **TOM ANDY BOHORQUEZ PINARGOTE** con cédula de identidad N° 0953785052 y por **DUSTIN JEAN MARTILLO GARCÍA** con cédula de identidad N° 0954725263, dando como finalizado el trabajo de titulación bajo la opción de proyecto técnico cumpliendo con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Guayaquil, febrero 2023

Atentamente,

Ing. Víctor Larco Torres, MSc.

C.I. 0923270136

AGRADECIMIENTO

En primer lugar, le agradezco a Dios por hacerme sentir seguro, confiado y animado en este proceso de titulación, asimismo por darme las fuerzas y la voluntad para realizar la tesis.

En segundo lugar, agradezco a mis progenitores por brindarme ese apoyo incondicional que me ayudó a cumplir este objetivo, de igual manera quiero agradecer a mi compañero de tesis por aportar un excelente conocimiento en el proceso para culminar nuestro trabajo de titulación, tampoco de quienes me ayudaron como club de robótica para que este proyecto salga adelante y logre ser culminado exitosamente.

Tom Bohórquez

Agradezco a Dios por darme salud y poder culminar esta etapa de mi vida, cumplir con las metas propuestas.

Agradezco a mi familia por sus grandes consejos y el apoyo brindado en el transcurso de todo este tiempo.

Agradezco a mi tutor y todos los docentes que han estado en el transcurso de mi carrera apoyando y brindando sus conocimientos en mi formación como profesional. De igual forma doy gracias a mis compañeras y compañeros del club de robótica por su motivación y consejos que me ha servido como soporte.

Dustin Martillo García

DEDICATORIA

Este logro universitario se lo dedico a Dios por darme las fuerzas y la voluntad asimismo a mis padres por estar presentes ahí dándome consejos y animándome a que salga adelante este proyecto de titulación de igual manera quiero agradecer a quienes conforman el club de robótica puesto que se ofrecieron a brindarme ayuda y facilitarme ciertos insumos que utilice para la realización del presente proyecto.

Tom Bohórquez

Dedico este logro a mis padres que siempre han estado pendientes de mí, han sido un pilar fundamental en todas las etapas de mi formación personal como profesional.

Mis familiares que han estado de apoyo ante cualquier problemática que se presenta a lo largo de mi vida y en carrerea universitaria.

Dustin Martillo García

RESUMEN

El presente proyecto de titulación tuvo como finalidad implementar un robot seguidor de línea con reconocimiento de pistas mediante visión artificial, usando la librería de Open CV y poder incursionarlo en las nuevas categorías del Club de Robótica que pertenece a la Universidad Politécnica Salesiana, Sede Guayaquil.

Para cumplir con el objetivo se usó el hardware Raspberry que sirve para programar en lenguaje Python, donde se importó la librería de Open CV, que facilitó la identificación de colores por medio de la cámara, esto se codificó mediante el software Thonny que por defecto viene incluido en el sistema operativo de la Raspberry.

El desarrollo de este proyecto se empezó con la identificación y segmentación de la pista con sus colores característicos (blanco y negro), una vez identificado se procedió a dar la rutina de seguimiento, dividiendo la pantalla por tres secciones, detectando así en las esquinas de la pantalla la curvatura, en el centro las rectas para enviar la información al Raspberry procesando los datos obtenidos y proceder con el movimiento de los motores respectivamente.

PALABRAS CLAVES: Visión Artificial, Open CV, Python, Raspberry.

ABSTRACT

The purpose of a tutorial project is presented to implement a line follower robot with path recognition through artificial vision, using the Open CV library and to be able to venture into the new categories of the Robotics Club that participates in the Salesian Polytechnic University, Headquarters Guayaquil.

To meet the objective, the Raspberry hardware used to program in Python language was used, where the Open CV library was imported, which facilitated the identification of colors through the camera, this was coded using the Thonny software that is included by default in the raspberry operating system.

The development of this project began with the identification and segmentation of the runway with its characteristic colors (black and white), once identified, the follow-up routine was carried out, dividing the screen into three sections, thus detecting in the corners of the screen the curvature, in the center the lines to send the information to the Raspberry processing the data obtained and proceeding with the movement of the motors respectively.

KEY WORDS: Artificial Vision, Open CV, Python, Raspberry.

ÍNDICE

PROBLEMA DE ESTUDIO

Un robot seguidor de línea tradicionalmente utiliza sensores infrarrojos, sin embargo, se ven afectados por la luminosidad del entorno durante el circuito de prueba.

Dado este antecedente, el prototipo a implementar contribuye al desarrollo de nuevos algoritmos que permitan trazar la trayectoria del robot seguidor de línea con una mejor precisión en la navegación mediante la visión artificial.

DELIMITACIÓN

- **Temporal:** El proyecto a realizar tiene como objetivo un tiempo estimado de 5 meses que comprenden desde el mes de octubre del 2022 hasta febrero del 2023.
- **Espacial:** El proyecto será realizado en el Club de Robótica, situado en el bloque "C", de la Universidad Politécnica Salesiana, sede Guayaquil.
- **Académico:** El proyecto a realizar se relacionó a conceptos aprendidos en clases de Programación, Robótica Móvil y adicional los conocimientos que a lo largo del tiempo hemos adquirido en el Club de Robótica.

JUSTIFICACIÓN

Actualmente la Universidad Politécnica Salesiana en sus distintas sedes por medio de los grupos ASU (Asociacionismo Salesiano Universitario), impulsa la participación de los estudiantes en diferentes áreas; entre ellos el Club de Robótica que tiene diferentes categorías de aprendizaje, que son: seguidor de líneas, soccer, drones, etc.

La mayoría de las competencias en la categoría seguidor de línea usan robot adherido a la pista y recopilan datos por medio de sensores infrarrojos, por la cual aprovechando esta categoría, se quiere incursionar en un nuevo avance con visión artificial, para el procesamiento de imágenes se necesita una CPU como Raspberry, tiene como ventaja analizar la ruta y mejorar la destreza de navegación en la pista mediante adquisición de imágenes en tiempo real del entorno, además, contribuirá para las futuras prácticas y motivación de los estudiantes.

OBJETIVOS

OBJETIVO GENERAL:

Diseñar e implementar un robot seguidor de línea mediante visión artificial, utilizando hardware como Raspberry, a través del uso de la librería OpenCV y lenguaje Python.

OBJETIVOS ESPECÍFICOS:

- Diseñar la estructura para el robot seguidor de línea.
- Programar el software para el procesamiento de imágenes mediante visión artificial.
- Integrar el hardware Raspberry con la cámara web y el controlador de motores.
- Realizar pruebas de validación en diferentes pistas de líneas continuas para comprobar su eficiencia.

MARCO TEÓRICO REFERENCIAL

INTRODUCCIÓN

A continuación, se da a conocer las características de un robot seguidor de línea, iniciando su análisis se desarrolla el algoritmo en seguidor de línea a través de visión, analizando los conceptos y sus principales características para el caso de estudio.

ROBOTS AUTÓNOMOS

Son dispositivos que constan con un sistema de sensores, sistema de control cuya finalidad principal es interactuar con el entorno y cumplir con funciones específicas en un ambiente simple o complejo sin la necesidad de estar siendo controlado por operadores humanos. (Ametic, 2021)

ROBOTS SEGUIDORES DE LÍNEA

El robot seguidor de línea como se puede observar en la figura 1, tiene como objetivo principal seguir una trayectoria con diferentes tipos de curvas desde el punto inicial al punto final, generalmente la línea a seguir es de color negro y su fondo es de color blanco.

Estos tipos de robots se los fomenta para el avance y exposición de la robótica, en la cual se agregan algunas dificultades en la categoría dando así las diferentes destrezas en: Líneas continuas, discontinuas, bifurcaciones, entre otras.

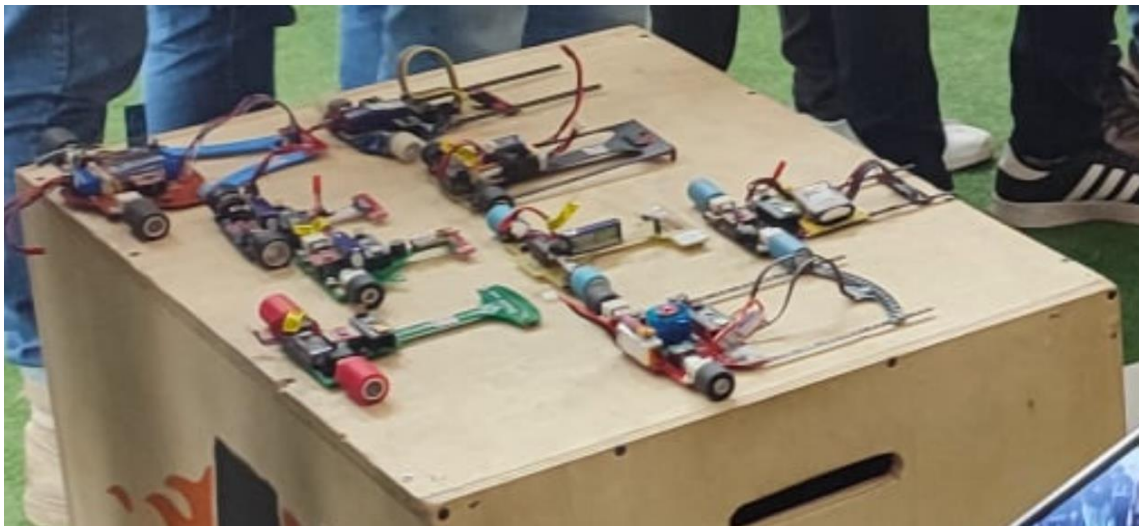

Figura 1. Robots seguidores de línea por medio de sensores infrarrojos

VISIÓN ARTIFICIAL

Esta tecnología tiene como objetivo emular la vista humana artificialmente mediante el procesamiento e interpretación de imágenes capturadas desde un hardware óptico que son las cámaras. (Clelio Daniel Endara Sumba , Emerson Joao Maigua Yánez, 2021)

Entre los softwares a usar se destaca Open CV para el uso y funcionamiento del proyecto debido a que es completo para el desarrollo y procesamiento de imágenes mediante computador

La visión artificial tiene tres principales utilidades que son:

- Distinguir personas, formas y objetos
- Analizar variación de luz, color y cinemática de objetos
- Sirve para la orientación y ubicación es espacios 2D o 3D

PROCESADO

El procesado ayuda a compensar las falencias producidas al momento de capturar una imagen mediante la cámara, ya sea contraste o brillo, los algoritmos de preprocesados permiten corregir dichas falencias por medio de operaciones básicas, filtrado, entre otras.

El preprocesado se utiliza en una imagen dimensionada **M*N** que consta de 256 niveles de grises por punto, el valor **0** se define como **negro carbón** y el **255** como **blanco**. (Chango, 2018)

UMBRALIZACIÓN

La umbralización permite calcular o variar un valor de umbral dependiendo de la singularidad del entorno a evaluar. (Alvear, 2018)

Este método es muy importante aplicarlo porque a través de Umbralización podremos segmentar imágenes, debido a que su función es convertir una imagen con diferentes tonalidades a una imagen en blanco y negro, donde el color blanco es nuestra área para trabajar mientras que el color negro es el fondo donde no se trabaja. (Palomino, 2019)

Entre las formas a destacar usadas en segmentación tenemos

- Procesamiento de bordes: se identifica la continuidad y discontinuidad de la imagen tomando la mejor trayectoria

RASPBERRY PI

Una placa Raspberry como se muestra en la figura 2, consiste en una base de 8.5 x 5.4 cm, embebido por un chip BCM2835, un procesador de 1 GHz ARM y un GPU VideoCore IV.

Dentro de sus ventajas se encuentran dos puertos HDMI, un mini Jack para audio, un puerto rj45 (Ethernet) y puertos USB 2.0 donde se puede conectar tanto un mouse o teclado, beneficioso al momento de configurarlo. (Willian Chávez Laz , Gerardo Barzallo Muñoz, 2018)

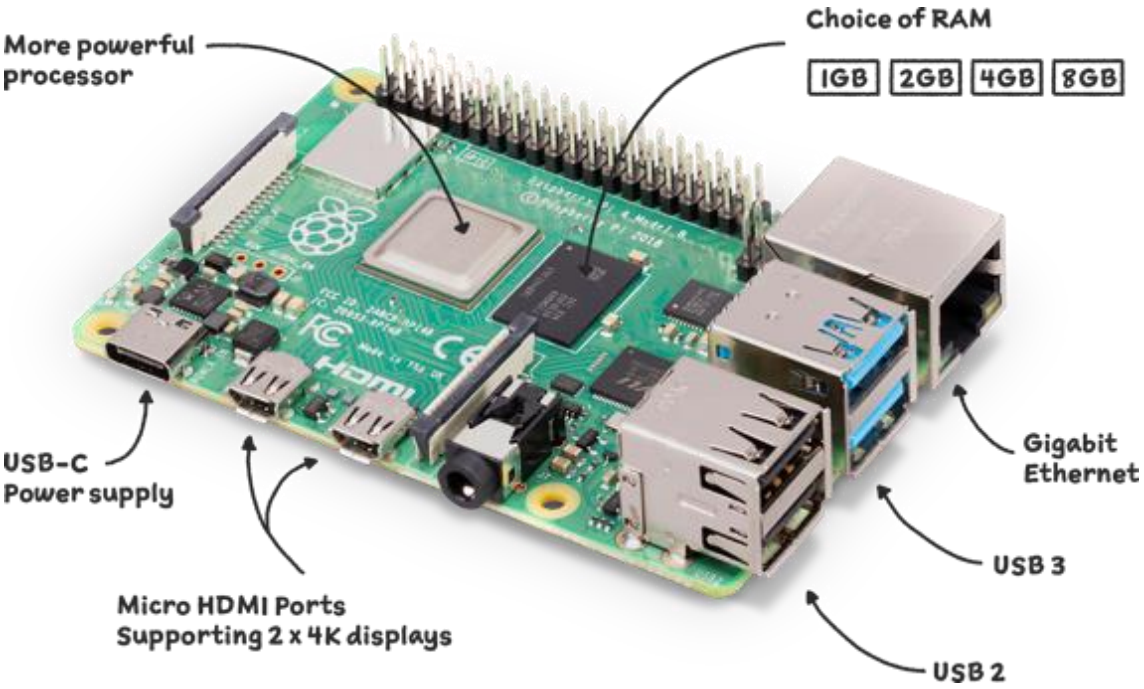

Figura 2. Hardware Raspberry PI (Raspberry, 2020)

MARCO METODOLÓGICO

Tarjeta de desarrollo

Se optó entre diversas tarjetas mediante un análisis de estudio que beneficie al proyecto tanto en hardware y software a usar.

Además de los recursos existentes en el mercado ecuatoriano para ser compaginables con el procesamiento de imágenes mediante la visión artificial, con el fin de que pueda ser manipulado para mejoras por diferentes sistemas operativos de código abierto.

Raspberry PI 4

Al ser una CPU el Raspberry PI 4 es adecuada para poder implementar en ella el sistema de seguidor de línea. Al ser una placa diseñada para recopilar una gran variedad de datos, se la utiliza para conectar la cámara, así se logra tener un ágil trabajo en el robot.

Diseño

Existen diversas plataformas para la realización de diseño en 3D, sin embargo, como se observa en la figura 3, este prototipo en particular se lo realizará en la herramienta AutoCAD.

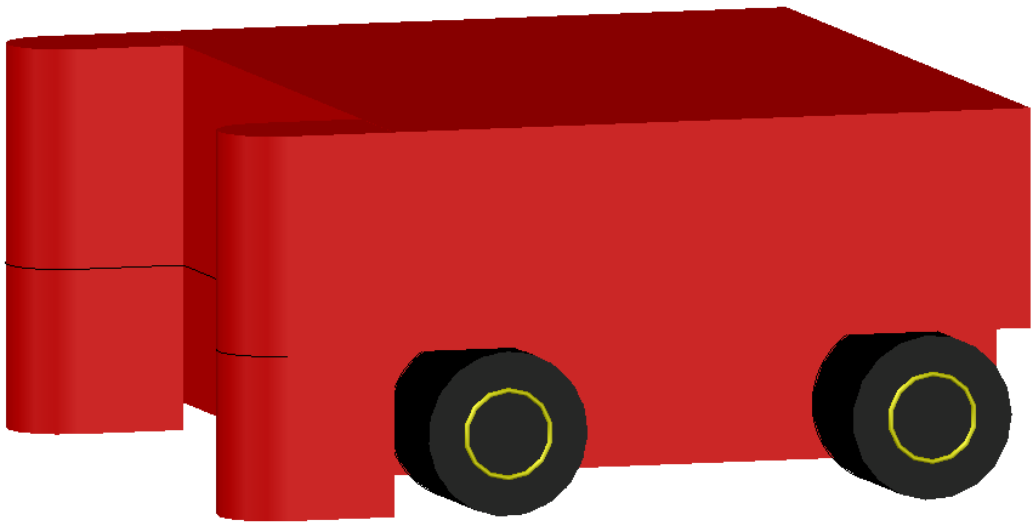

Figura 3. Vista lateral del prototipo seguidor de línea.

En la figura 4 se muestra la base con sus respectivas medidas para asentar el prototipo y la colocación de los motores, el puente H (driver) y el regulador.

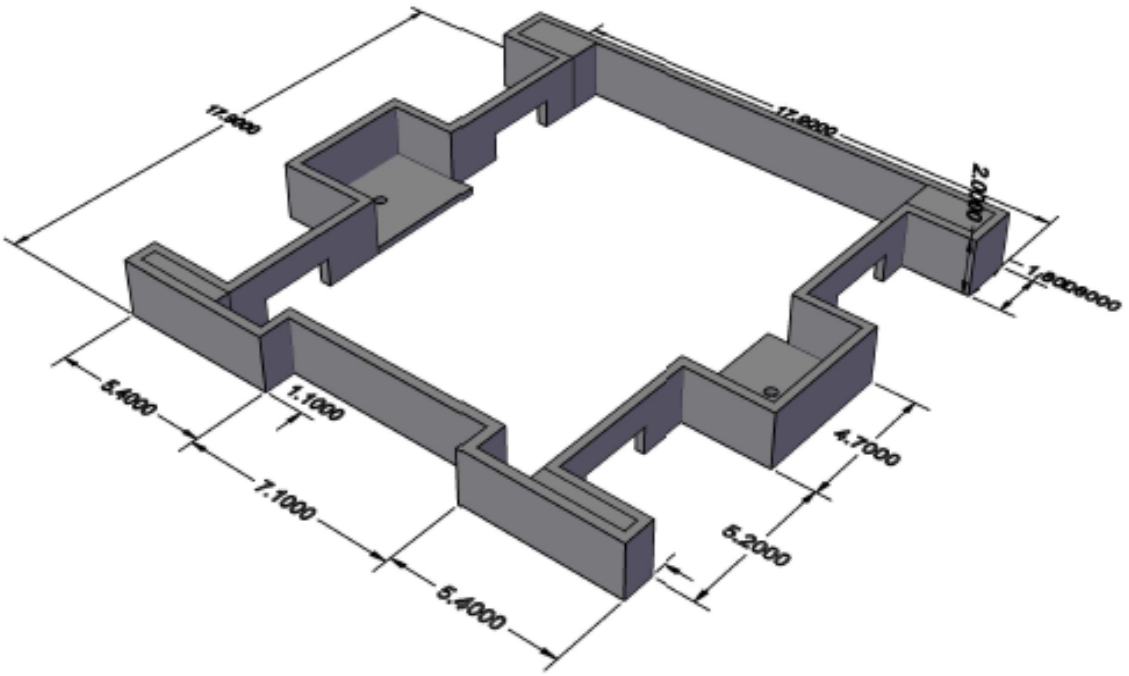

Figura 4. Diseño de la base del prototipo.

En la figura 5 se muestra la planta alta con sus respectivas medidas, la cual servirá para colocar la Raspberry y las baterías, asimismo, se dejó orificios para transportarlos los conductores hacia los elementos que se encuentran en la base.

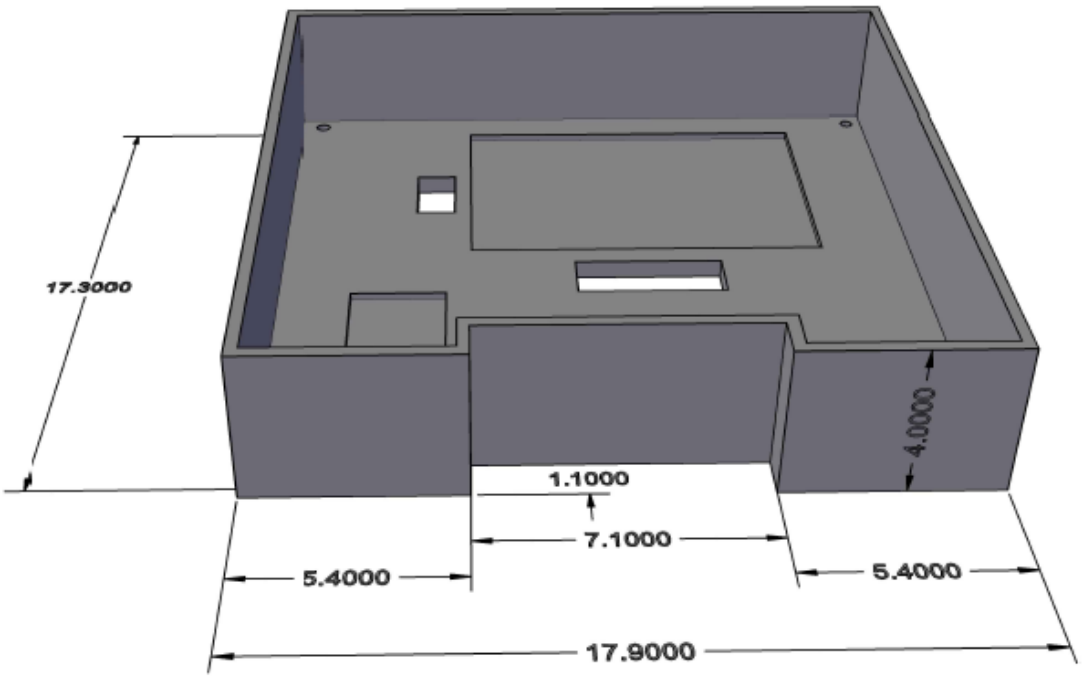

Figura 5. Diseño de la planta alta del prototipo.

En la figura 6 se muestra la tapa con sus respectivas medidas, aquí se dejaron los orificios para colorar los pulsadores, y el interruptor para la energizar el sistema eléctrico.

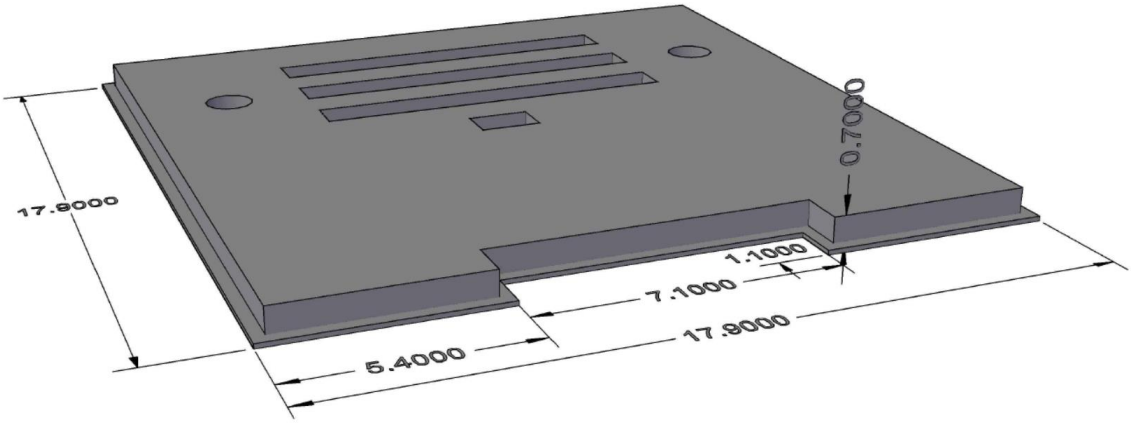

Figura 6. Diseño de la tapa del prototipo.

Impresión

Tomando en cuenta las dimensiones del prototipo se optó por realizar las impresiones en tres partes por motivos al tiempo de impresión de este.

Para dichas impresiones se usó el software Ultimaker Cura 5.0.

En la figura 7, usamos el Ultimaker Cura para imprimir la base del prototipo, damos toda la característica a la impresión, en este diseño seleccionamos la capa de relleno de la impresión a 2mm como inicial y de 3mm para todo el cuerpo mientras que la temperatura de la impresión a 200 grados centígrados y 60 grados centígrados para la cama (placa) de impresión este parámetro es de acuerdo al material que se va utilizar (PLA).

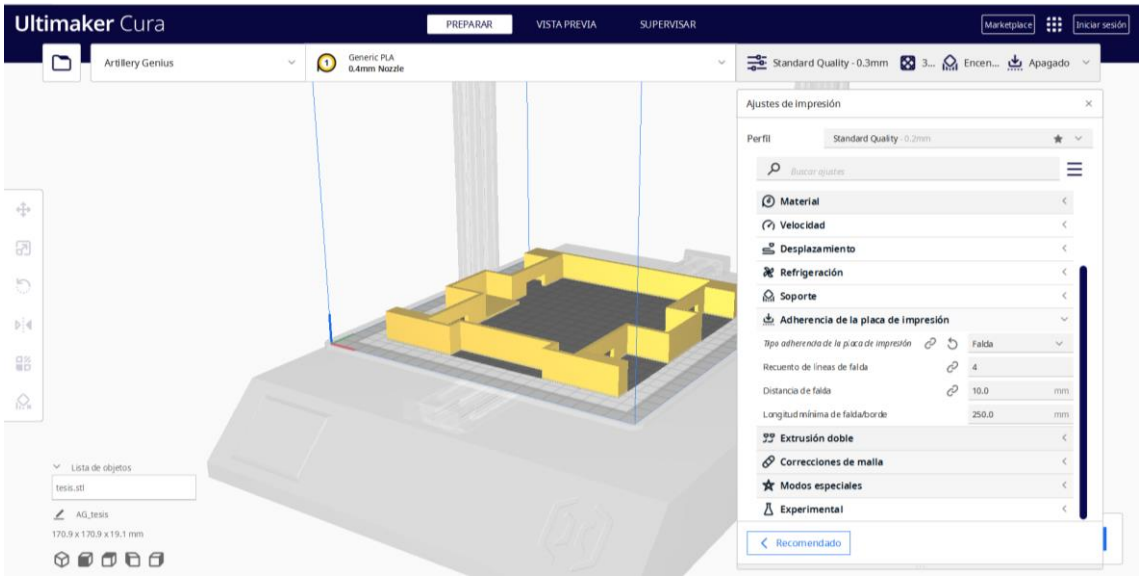

Figura 7. Vista de impresión para la base del prototipo.

En la figura 8, se usó el Ultimaker Cura para imprimir la planta alta del prototipo, se dá todas la características de la impresión, en este diseño se selecciona la capa de relleno de la impresión a 2mm como inicial y de 3mm para todo el cuerpo mientras que la temperatura de la impresión a 200 grados centígrados y 60 grados centígrados para la cama (placa) de impresión este parámetro es de acuerdo al material que se va utilizar (PLA).

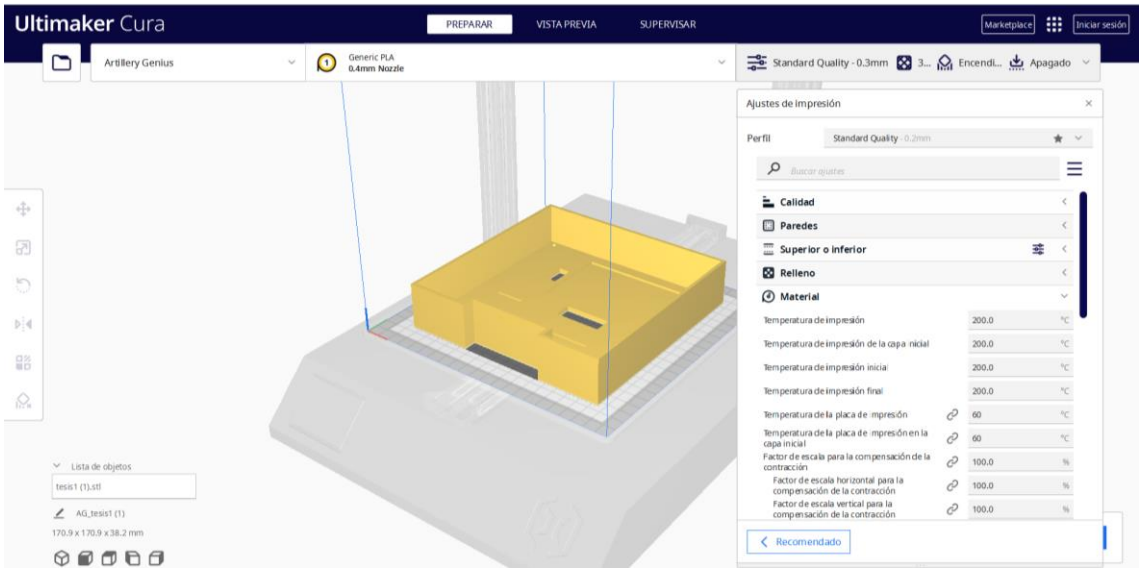

Figura 8. Vista de impresión para la planta alta del prototipo.

En la figura 9, usamos el Ultimaker Cura para imprimir la tapa del prototipo, se dan todas las características de la impresión, en este diseño se selecciona la capa de relleno de la impresión a 2mm como inicial y de 3mm para todo el cuerpo mientras que la temperatura de la impresión a 200 grados centígrados y 60 grados centígrados para la cama (placa) de impresión este parámetro es de acuerdo al material que se va utilizar (PLA).

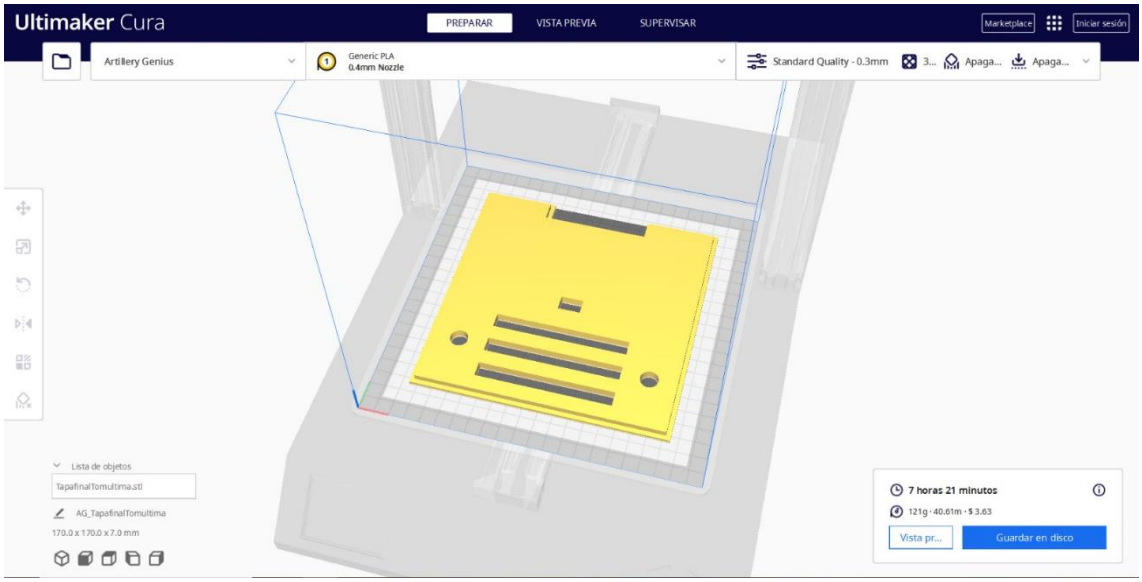

Figura 9. Vista de impresión para la tapa del prototipo.

Implementación

Antes de implementar algún dispositivo electrónico se debe realizar un análisis para el correcto funcionamiento del prototipo.

Dada la circunstancia en la tabla 1, se detalla el consumo eléctrico de cada componente que conforma el circuito eléctrico, asimismo se toma en cuenta la conexión de la cámara a la Raspberry en este proyecto.

Tabla1. Tabla del consumo eléctrico.

		CONSUMO	
	ELEMENTOS	VOLTAJE	AMPERAJE
	RASPBERRY	5	3
	LUCES LED	12	2
	MOTORES	5	1.2
	VENTILADOR RASPBERRY	5	1.25
	CAMARA	5	2
	PUENTE H	12	0.036
TOTAL		44	9.486

Usando el software Proteus en la figura 10 se presenta el esquema eléctrico de las conexiones realizadas en el circuito, exceptuando la cámara del proyecto, para poder tener en cuenta los materiales a usar y realizar una correcta distribución de las baterías, optimizando y garantizando su funcionalidad.

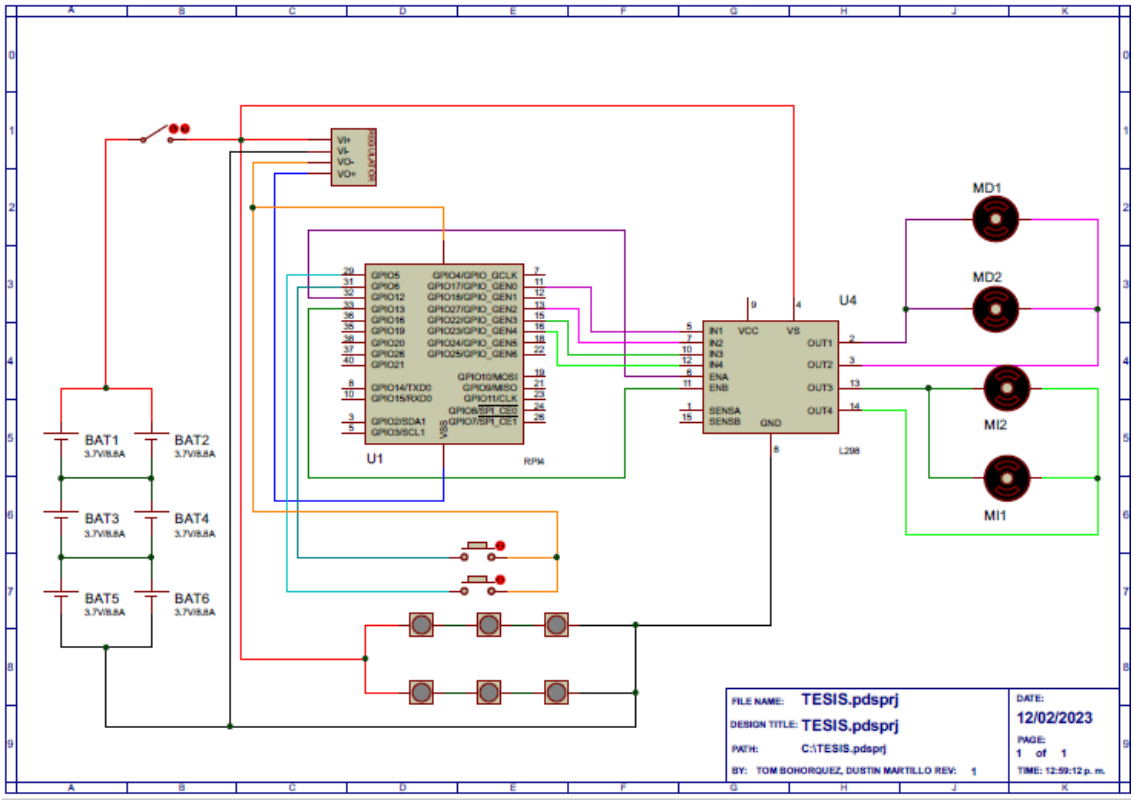

Figura 10. Diseño del esquema eléctrico en Proteus.

Como se muestra en la misma figura, y basándose en la tabla 1 la cual contiene los resultados de consumo eléctrico se optó por utilizar 6 pilas de 3.7V(voltios) / 8.8A(amperios), conectadas en 3 pares en paralelo para después conectarlas en serie, con la finalidad de poder distribuir el amperaje y sumar los voltajes para cumplir con el consumo eléctrico requerido.

En la figura 11, para llevar a cabo el desarrollo del armado, se debió seleccionar los componentes adecuados

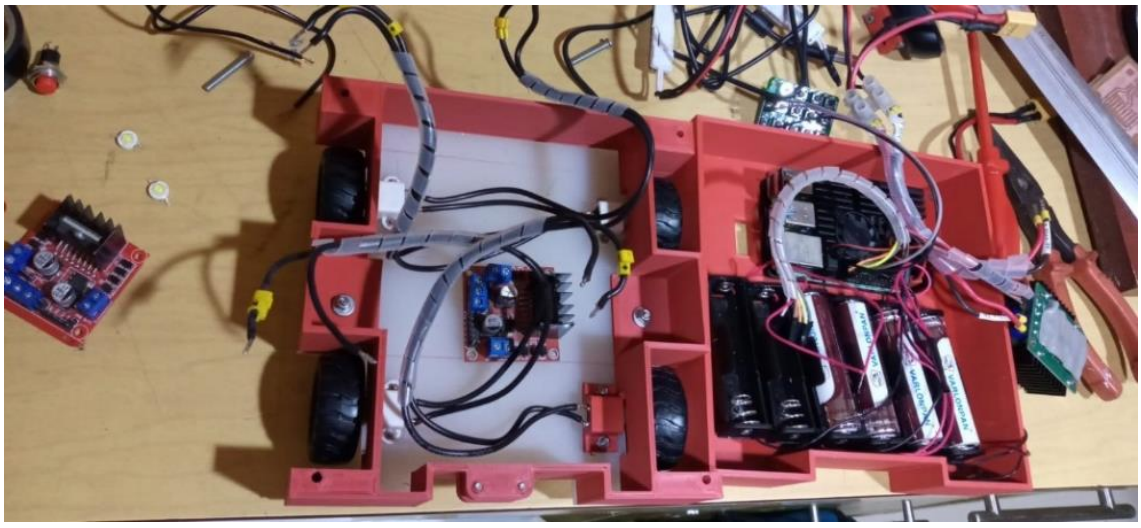

Figura 11. Armado del prototipo.

La pista será de color blanco en el fondo con la línea de color negro tal como se muestra en la figura 12, se arma la trayectoria que el robot seguirá

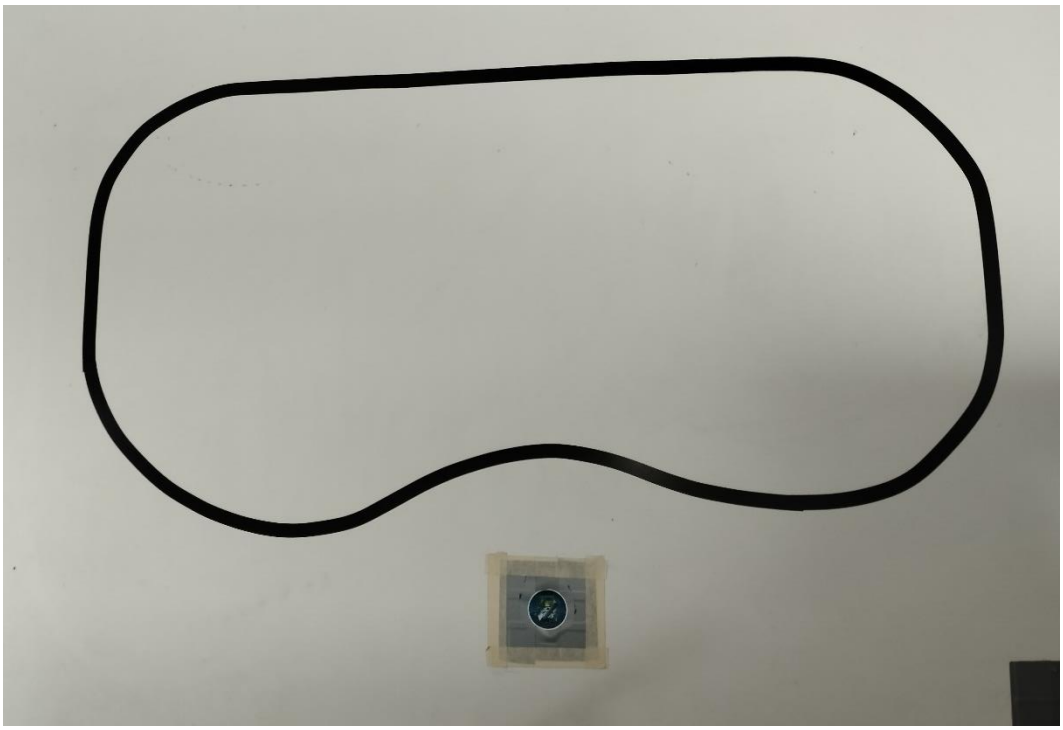

Figura 12. Pista para el robot seguidor de línea

Programación

Como la Raspberry es un ordenador al momento de energizarla se abrirá el escritorio, es por ello por lo que se realizó el siguiente código para que inicie el programa seguidor de línea a penas la Raspberry se energice.

Las siguientes líneas de código que se muestra en la figura 13 cumplen con dicha función:

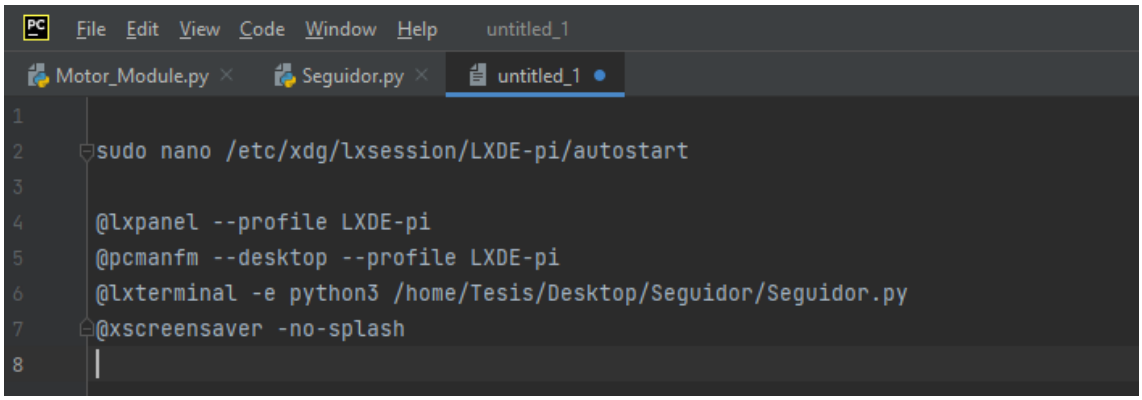

Figura 13. Inicialización de la programación mediante CMD de la Raspberry

Este código se lo hace desde la CMD de la Raspberry se inicializa el código principal sin estar accediendo mediante la pantalla.

Se divide el código en dos, por eso hay que importar las siguientes librerías, en este caso (Motor_Module) código desarrollado para la tracción de los motores.

A continuación, en la figura 14 se muestra una parte del código y las librerías para definir los puertos que controlaran los motores.

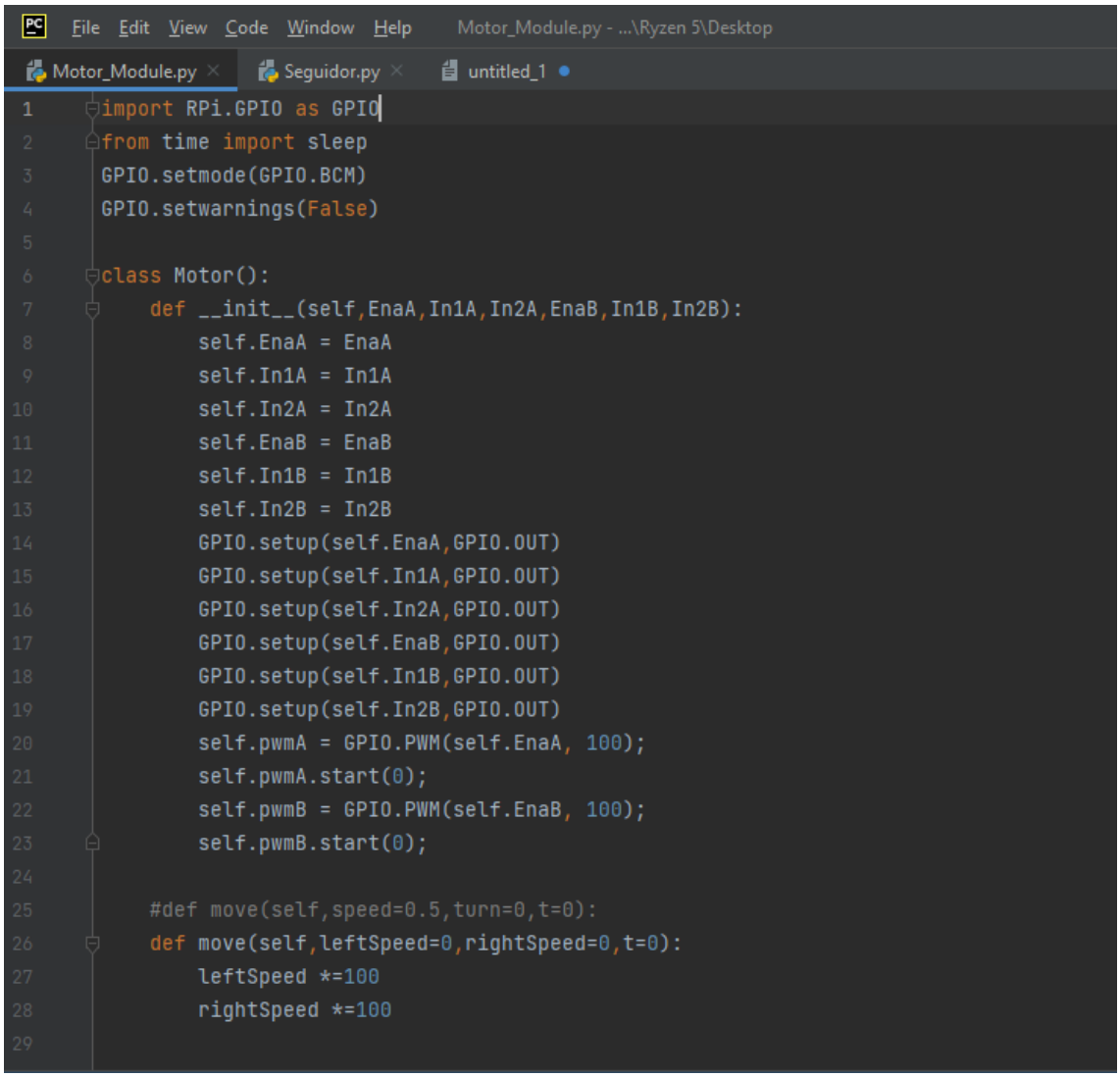

Figura 14. Declaración de puertos GPIO

En este caso al desarrollarse el robot seguidor de línea mediante visión artificial se importa la librería de OpenCV y las habilitaciones de los puertos GPIO

La siguiente imagen muestra en el código el llamado de las librerias:

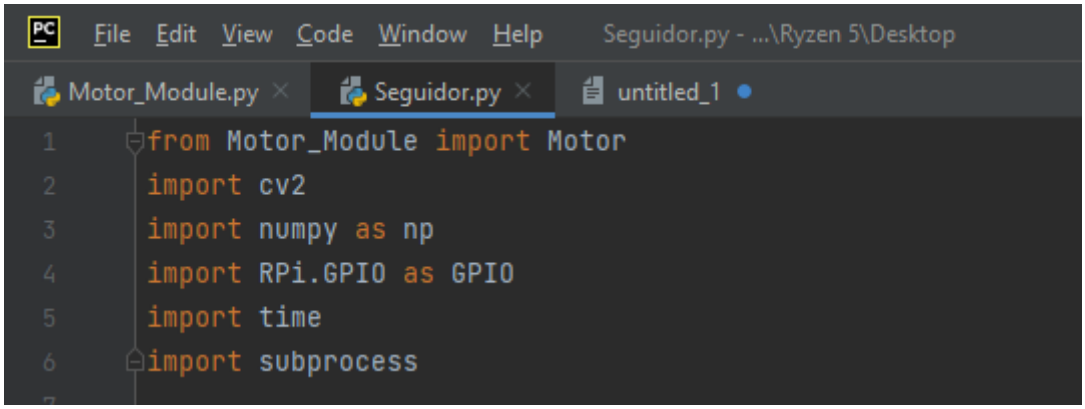

Figura 15. Importación de las librerías.

Entre la codificación se muestra lo más relevante haciendo referencia a la librería OpenCV

Primera línea, se inicializa la cámara web

Segunda línea, realiza la captura del objeto asignado en la cámara web

Tercera línea, se muestra la imagen en tiempo real

```
cv2.VideoCapture(0)
ret, frame = cap.read()
cv2.imshow("Fotograma",frame)
cv2.moveWindow("Fotograma",10,60)
```

Como se puede observar en la figura 16, se asignan dos teclas en la programación la cual nos dirige a dos modos de manejo el prototipo

Tanto de forma autómata y de forma manual como un plus adicional, todo eso visualizado en tiempo real.

```
while True:
    key = cv2.waitKeyEx(33)         #función de enlace con el teclado. Su argumento es tiempo en milisegundos.

    if(key == ord('1')):
        print("Inicio de Seguidor")     #permite mostrar texto en pantalla
        Button_estado_1 = not Button_estado_1   #Combio de valor lógico a variable booleana
        Button_estado_2 = False
        print("valor de variable 1" + str(Button_estado_1))
        time.sleep(0.5)             #añade un retraso a la ejecucion de un programa

    if(key == ord('2')):

        print("Inicio de Control Remoto")       #permite mostrar texto en pantalla
        Button_estado_2 = not Button_estado_2
        Button_estado_1 = False                 #Combio de valor lógico a variable booleana
        print("valor de variable 2" + str(Button_estado_1))
        time.sleep(0.5)
```

Figura 16. Importación de las librerías.

En este caso las pruebas realizadas, se segmenta en 3 espacios la detección de la línea la cual cada segmentación tiene su propio análisis, por tal motivo en la prueba se pudo llegar a la conclusión de que el punto exacto para realizar una correcta trayectoria es en la segmentación dos para obtener un mejor ángulo al girar.

Como se puede observar en la figura 17 dividimos en tres contours, donde el contours2 es la sección con mayor prioridad a evaluar debido a la altura y el ángulo de la cámara, en este caso usamos el CX y CY para encontrar el centro de cada segmento y evaluamos mediante IF para poder avanzar o realizar los giros necesarios. En el caso de que no detecte más pista en el centro, da por finalizado el programa, pero si la pista se llega a perder por un tiempo en las segmentaciones laterales, toma como referencia lo último visualizado y procede a buscar la línea del lado indicado.

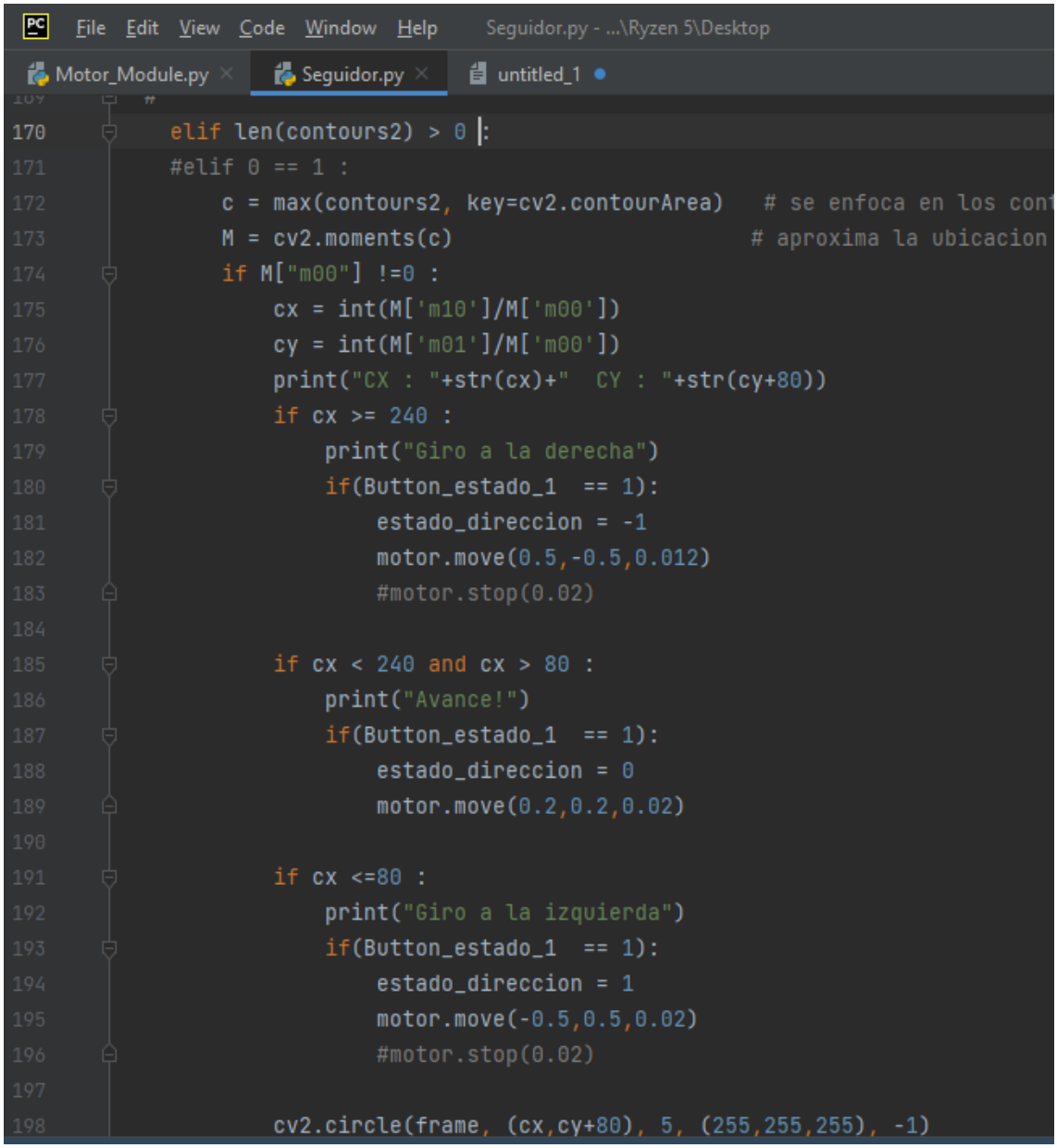

Figura 17. División de los contornos.

CRONOGRAMA

El presente proyecto se lo realizará en un lapso de cinco meses que comprenden desde el mes de octubre del 2022 hasta el mes de febrero del 2023, las actividades a realizar se las detallará en la tabla 2 de la siguiente manera:

Tabla 2. Semanas para la elaboración de proyecto.

SEMANAS DE TRABAJO		ACTIVIDADES		
		ENSAMBAJE	PROGRAMACION	PRUEBA - ERROR
OCTUBRE	4ta SEMANA	4 HORAS		
NOVIEMBRE	1ra SEMANA	4 HORAS		
	2da SEMANA	6 HORAS		
	3ra SEMANA	8 HORAS		
	4ta SEMANA	8 HORAS		
DICIEMBRE	1ra SEMANA	8 HORAS		
	2da SEMANA		8 HORAS	
	3ra SEMANA		6 HORAS	
ENERO	1ra SEMANA		8 HORAS	
	2da SEMANA		8 HORAS	
	3ra SEMANA		10 HORAS	
	4ta SEMANA		10 HORAS	
FEBRERO	1ra SEMANA			10 HORAS
	2da SEMANA			10 HORAS
	3ra SEMANA			10 HORAS

PRESUPUESTO

Para la implementación de este proyecto se va a necesitar comprar materiales, se realizó un presupuesto el cual se lo detalla en la tabla 3.

Tabla 3. Presupuesto para la implementación de la tesis.

	CANTIDAD	MATERIALES	COSTO
	1	FILAMENTO PLA	$ 20.00
	6	BATERIAS	$ 15.00
	1	REGULADOR	$ 11.50
	1	RASPBERRY + KIT	$ 265.00
	1	KIT MOTORES + RUEDAS	$ 30.00
	1	CAMARA	$ 58.00
	1	PUENTE H	$ 3.60
	2	LUCES LED	$ 2.00
	2	PULADORES	$ 0.70
	2	JUMPERS	$ 4.00
	1	INTERRUPTOR	$ 0.50
		GASTOS VARIOS	$ 100.00
TOTAL	19		$ 510.30

CONCLUSIONES

Con las investigaciones se determinó que el robot seguidor de línea mostró un resultado optimo la cual se podría ir puliendo e incursionarlo en nuevas categorías para las competencias futuras conjuntas al Club de Robótica de la UPS sede Guayaquil – centenario.

También se pudo obtener el reconocimiento de la trayectoria a través del lenguaje de programación en Python y las librerías de OpenCV.

RECOMENDACIONES

El proyecto presentado está enfocado a pistas básicas seguidoras de línea para el robot con visión artificial debido a que cuenta con una sola cámara

Para competencias a nivel de destreza profesional se recomienda el incremento de cámaras o una mayor altura en la ubicación de la cámara para que la segmentación de la pista se realice con mayor fluidez y eficacia.

REFERENCIAS

Alvear, J. L. (2018). *CONTROL DE UN ROBOT SEGUIDOR DE LÍNEA TIPO DIFERENCIAL MEDIANTE VISIÓN ARTIFICIAL*. Obtenido de https://dspace.ups.edu.ec/bitstream/123456789/15958/1/UPS-ST003718.pdf

Ametic. (2021). *AMETIC*. Obtenido de https://ametic.es/es/publicaciones/industria-40-que-es-un-robot-autonomo-0

Chango, J. (2018). Obtenido de https://dspace.ups.edu.ec/bitstream/123456789/15958/1/UPS-ST003718.pdf

Clelio Daniel Endara Sumba , Emerson Joao Maigua Yánez. (2021). *DESARROLLO DE UN ALGORITMO DE TRAYECTORIA PARA UN ROBOT SEGUIDOR DE LÍNEA DESTREZA DE COMPETENCIA MEDIANTE VISIÓN E INTELIGENCIA ARTIFICIAL*. Obtenido de https://dspace.ups.edu.ec/bitstream/123456789/19880/1/UPS%20-%20TTS281.pdf

López, Álvaro y García. (2010). *DESARROLLO DEL SISTEMA DE LOCOMOCIÓN DE UNA PLATAFORMA HARDWARE PARA ROBOCUP SMALL SOCCER LEAGUE (SSL)*. Obtenido de https://core.ac.uk/download/pdf/30043662.pdf

LUIS, C. A. (2018). Obtenido de https://dspace.ups.edu.ec/bitstream/123456789/15958/1/UPS-ST003718.pdf

Palomino, L. S. (2019). *Revista De investigación De Sistemas E Informática,*. Obtenido de https://revistasinvestigacion.unmsm.edu.pe/index.php/sistem/article/view/3299

Raspberry. (2020). *RASPBERRY PI*. Obtenido de https://www.raspberrypi.com/products/raspberry-pi-4-model-b/

Willian Chávez Laz , Gerardo Barzallo Muñoz. (2018). *DISEÑO E IMPLEMENTACIÓN DE UN MÓDULO DIDÁCTICO BASADO EN RASPBERRY PI3 Y ARDUINO MEGA PARA CONTROL Y MONITOREO DE UN BRAZO ESTABILIZADOR*. Obtenido de https://dspace.ups.edu.ec/bitstream/123456789/16353/1/UPS-GT002366.pdf

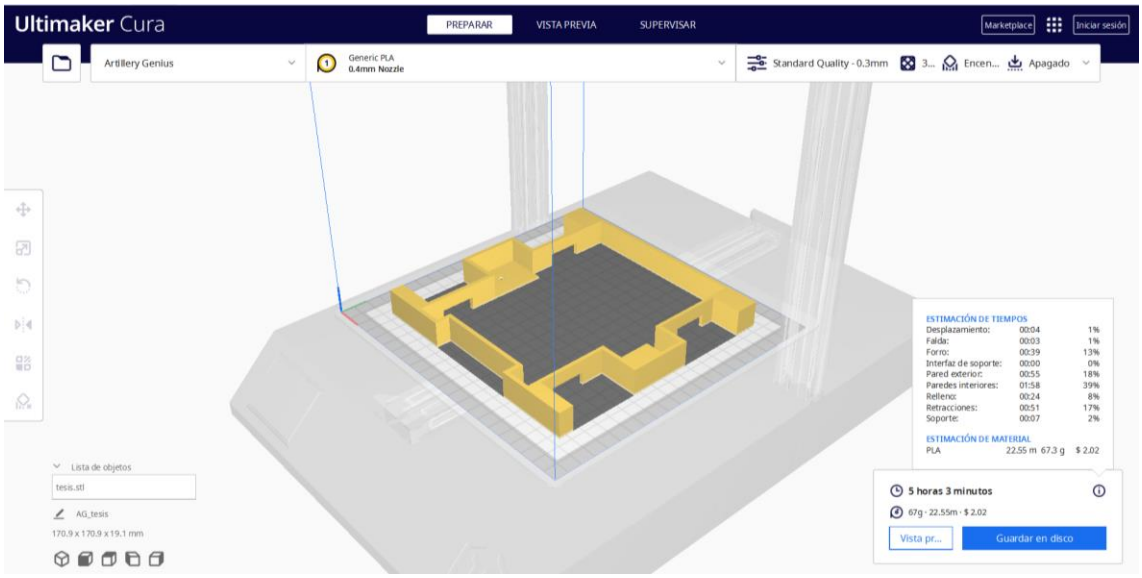

Figura 18. Tiempo de espera, impresión para la base del prototipo 5 horas 3 minutos.

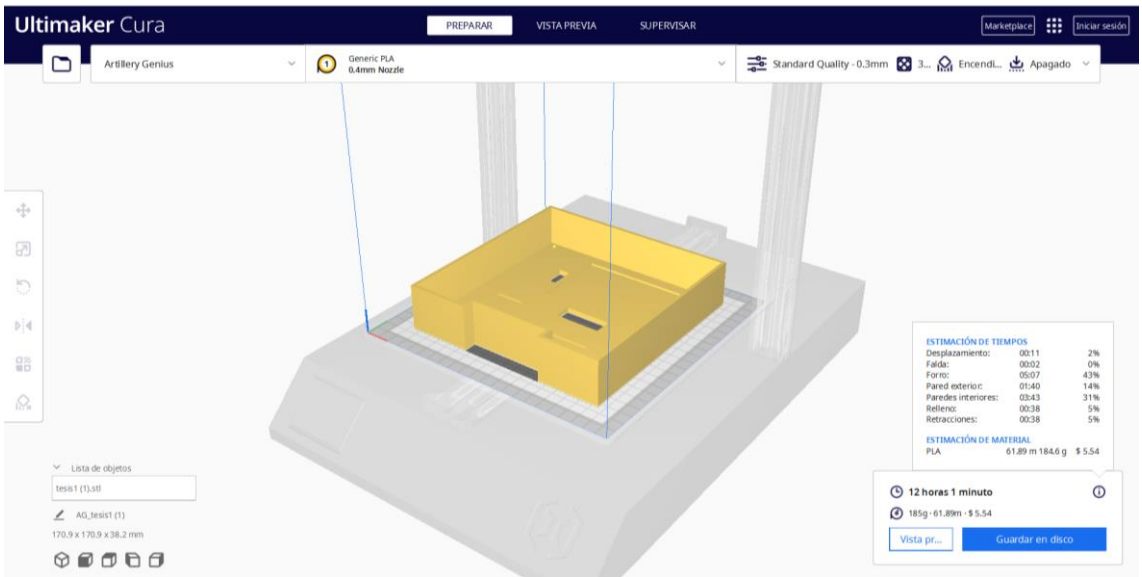

Figura 19. Tiempo de espera, impresión para la planta alta del prototipo 12 horas 1 minuto.

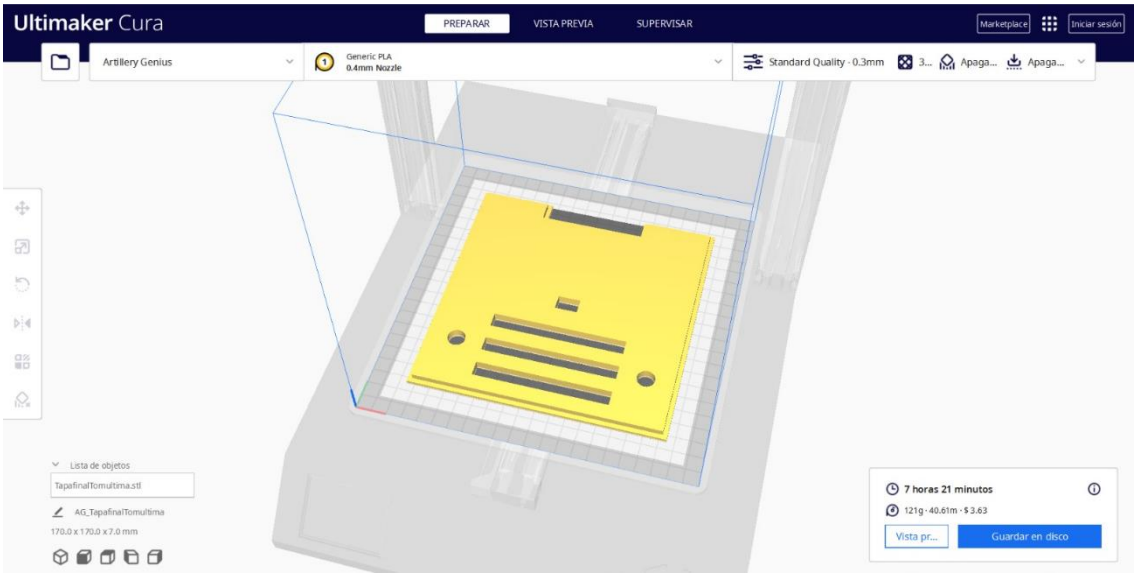

Figura 20. Tiempo de espera, impresión para la tapa del prototipo 7 horas 21 minutos.

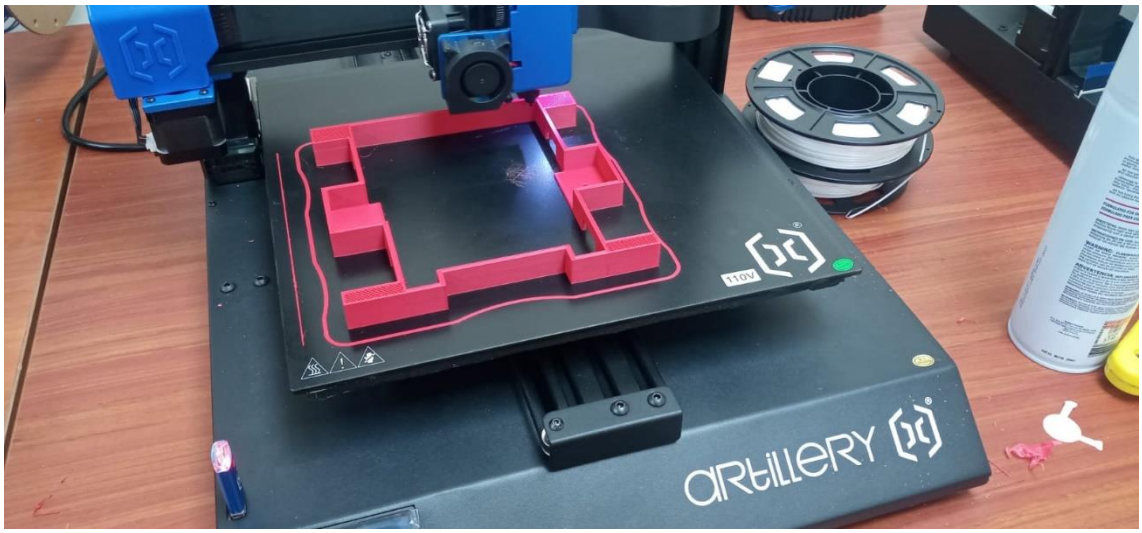

Figura 21. Vista de la impresión para la base del prototipo mediante los laboratorios de la UPS.

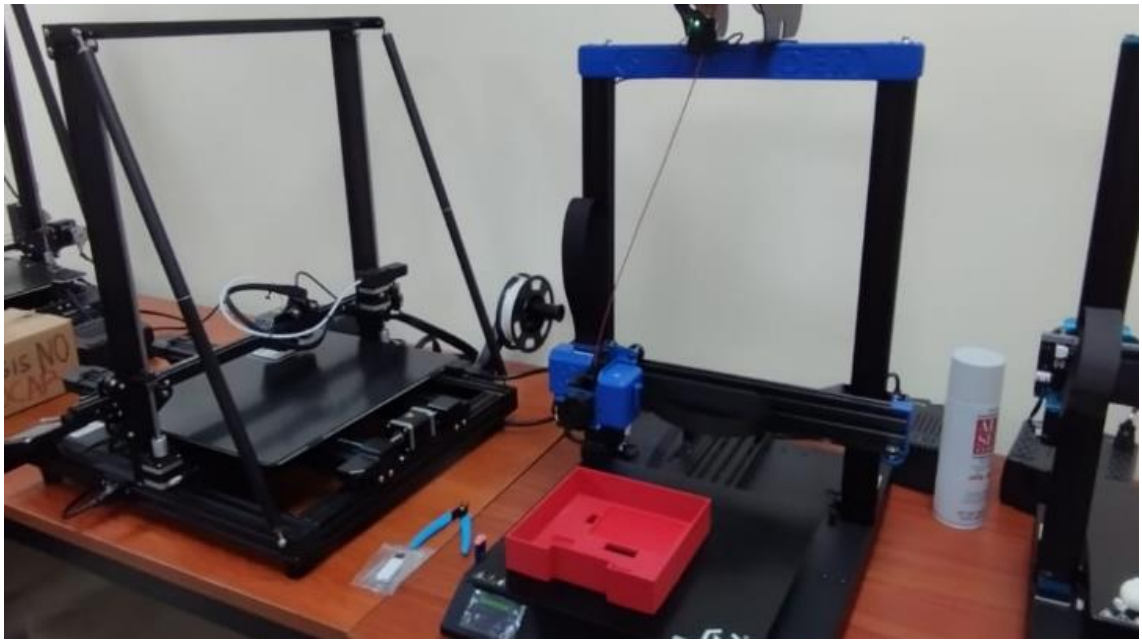

Figura 22. Imprimiendo para la planta alta del prototipo mediante los laboratorios de la UPS.

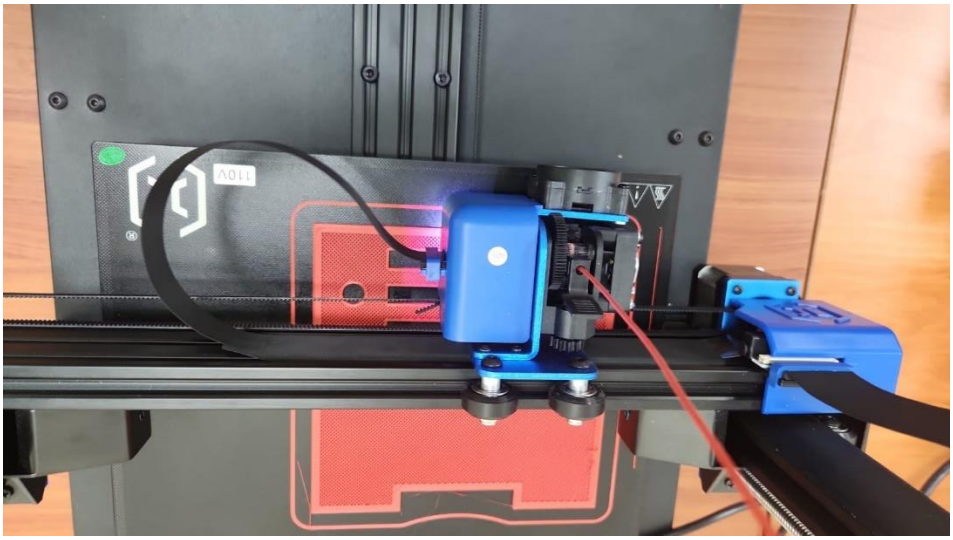

Figura 23. Imprimiendo para la tapa del prototipo mediante los laboratorios de la UPS.

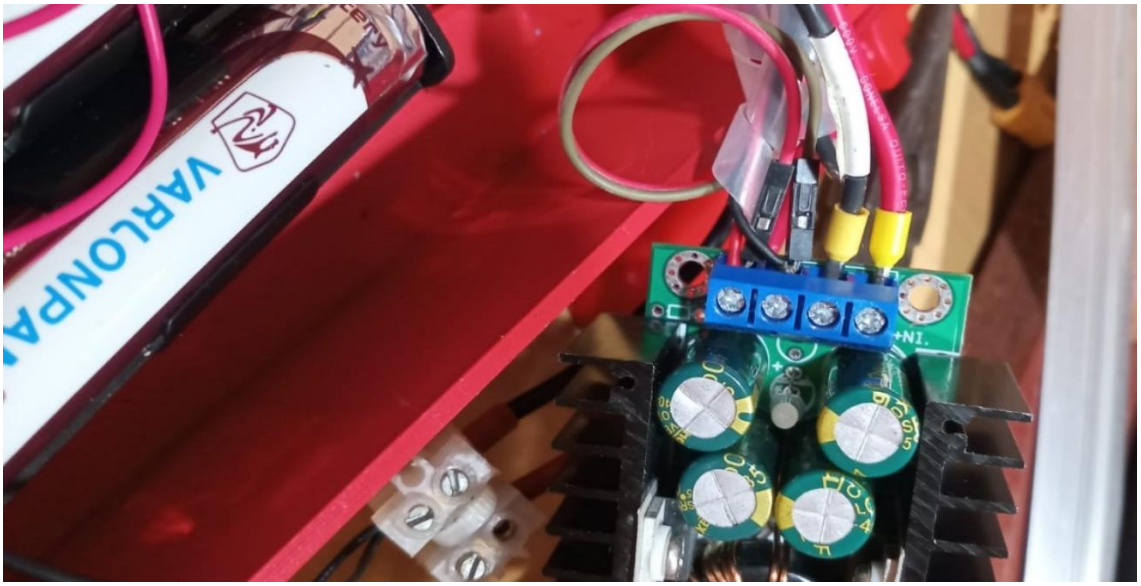

Figura 24. Conexión del regulador de voltaje a 5V.

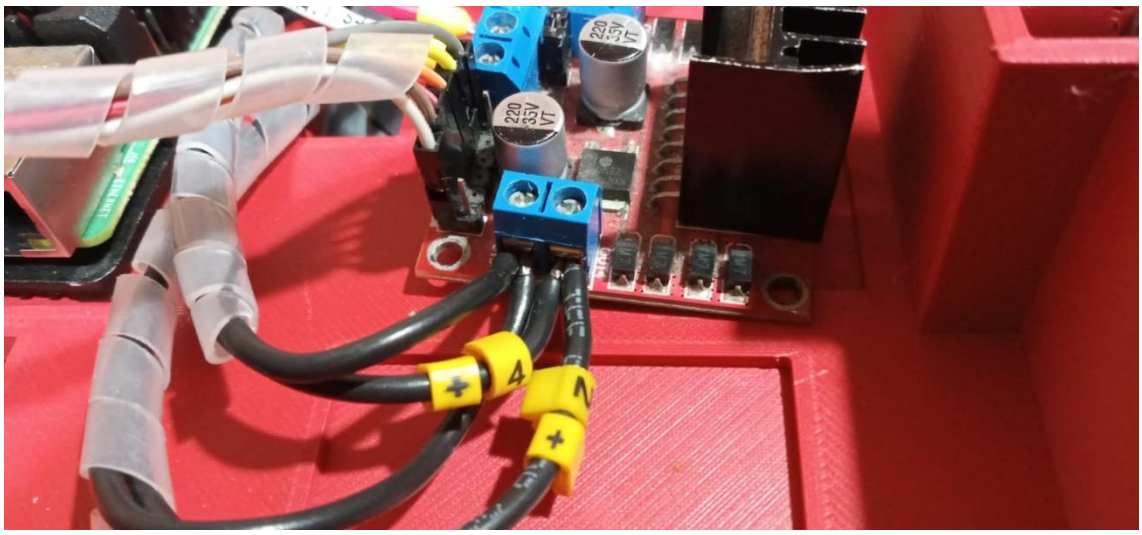

Figura 25. Conexión del drive para los motores.

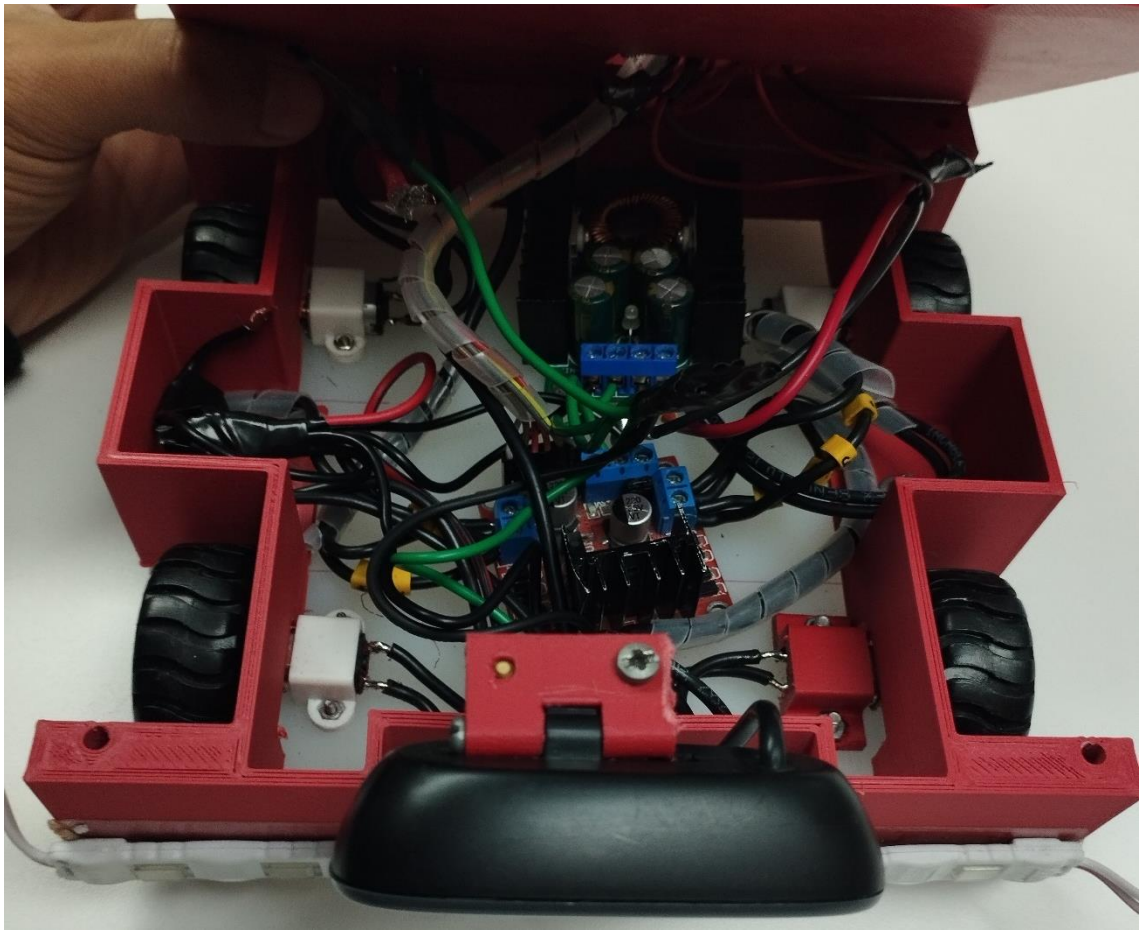

Figura 26. Ubicación del drive y el regulador en la base del prototipo.

Figura 27. Ubicación de las baterías y la Raspberry en la planta alta.

```
File  Edit  Format  Run  Options  Window  Help
import RPi.GPIO as GPIO
from time import sleep
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

class Motor():
    def __init__(self,EnaA,In1A,In2A,EnaB,In1B,In2B):
        self.EnaA = EnaA
        self.In1A = In1A
        self.In2A = In2A
        self.EnaB = EnaB
        self.In1B = In1B
        self.In2B = In2B
        GPIO.setup(self.EnaA,GPIO.OUT)
        GPIO.setup(self.In1A,GPIO.OUT)
        GPIO.setup(self.In2A,GPIO.OUT)
        GPIO.setup(self.EnaB,GPIO.OUT)
        GPIO.setup(self.In1B,GPIO.OUT)
        GPIO.setup(self.In2B,GPIO.OUT)
        self.pwmA = GPIO.PWM(self.EnaA, 100);
        self.pwmA.start(0);
        self.pwmB = GPIO.PWM(self.EnaB, 100);
        self.pwmB.start(0);

    #def move(self,speed=0.5,turn=0,t=0):
    def move(self,leftSpeed=0,rightSpeed=0,t=0):
        leftSpeed *=100
        rightSpeed *=100


Ln: 1  Col: 0
```

Figura 28. Código para mover los motores.

```
File  Edit  Format  Run  Options  Window  Help
from Motor_Module import Motor
import cv2
import numpy as np
import RPi.GPIO as GPIO
import time
import subprocess



AnchoFoto = 320
AltoFoto= 240
cap = cv2.VideoCapture(0) #objeto de capturadora de video, acceso a la camara web

cap.set(3, AnchoFoto)
cap.set(4, AltoFoto)


#Orden de pines en el modulo python (EnA,In1A,In2A,EnB,In1B,In2B)
motor= Motor(13,19,16,17,22,27)
Button_pin_1 = 5
Button_pin_2 = 6

estado_direccion = 0
Button_estado_1 = False  # True
Button_estado_2 = False
Button_estado_paro = False

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

                                                                    Ln: 177  Col: 55
```

Figura 29. Código para configurar los pulsadores y el tamaño de imagen.

```
File  Edit  Format  Run  Options  Window  Help
ret, frame = cap.read()                 #ret is una variable booleana que retorna verdadero si el fotograma es accesible,
frame21 = frame[0:80,0:320]                     #Un nuevo fotograma a partir del recorte de un fotograma previo
frame22 = frame[80:160,0:320]
frame23 = frame[160:240,0:320]
#captura de fotograma desde la camara
low_b = np.uint8([40,40,40])
high_b = np.uint8([0,0,0])
mask1 = cv2.inRange(frame21, high_b, low_b) #vuelve la imagen binaria
      #"frame21" la imagen oriinal
      #rango de color alto
      #rango de color bajo
contours, hierarchy = cv2.findContours(mask1, 1, cv2.CHAIN_APPROX_NONE) #funcion de contornos para diferenciar la pista con el fondo
        # "hierarchy" en caso de existir otro contorno interno o externo
        # "mask1" imagen Binaria
        # "1" modo recuperacion de contorno
        # "cv2.CHAIN_APPROX_NONE" Método de aproximacion de contorno, en este casp guarda cada punto del contorno

low_b = np.uint8([40,40,40])
high_b = np.uint8([0,0,0])
mask2 = cv2.inRange(frame22, high_b, low_b)
contours2, hierarchy = cv2.findContours(mask2, 1, cv2.CHAIN_APPROX_NONE) #funcion de contornos para diferenciar la pista con el fondo
low_b = np.uint8([40,40,40])
high_b = np.uint8([0,0,0])
mask3 = cv2.inRange(frame23, high_b, low_b)
contours3, hierarchy = cv2.findContours(mask3, 1, cv2.CHAIN_APPROX_NONE) #funcion de contornos para diferenciar la pista con el fondo



#if len(contours) > 0 :
                                                                    Ln: 1  Col: 0
```

Figura 30. Definición para el rango de colores y los contornos para la pista.

```
File  Edit  Format  Run  Options  Window  Help

    cv2.imshow("Mascara3",mask3)              #Muestra el fotograma
    cv2.moveWindow("Mascara3",500,250)
    cv2.imshow("Mascara2",mask2)              #Muestra el fotograma
    cv2.moveWindow("Mascara2",500,140)
    cv2.imshow("Mascara1",mask1)              #Muestra el fotograma
    cv2.moveWindow("Mascara1",500,30)

    cv2.imshow("Fotograma",frame)             #Muestra el fotograma
    cv2.moveWindow("Fotograma",10,30)

    if(Button_estado_1  == 0):
        motor.stop(0.02)
        print("Seguidor detenido")
        #break


        #if(Button_estado_paro == 1  ):
        #       print("Programa detenido")
        #       Button_estado_paro = not Button_estado_paro

    #if cv2.waitKey(1) & 0xff == ord('q') :   # 1 is tiempo en ms, tecla "q" para salida del bucle
    #if (key == ESC) :
    if (key == ord('3')) :
        motor.stop(0.02)
        print("Programa detenido")
        break
cap.release()                       #libera la camara
cv2.destroyAllWindows()                                 #destruye todas las ventanas que hemos creado
                                                                                    Ln: 1  Col: 0
```

Figura 31. Definición de fotogramas para el seguidor de línea.

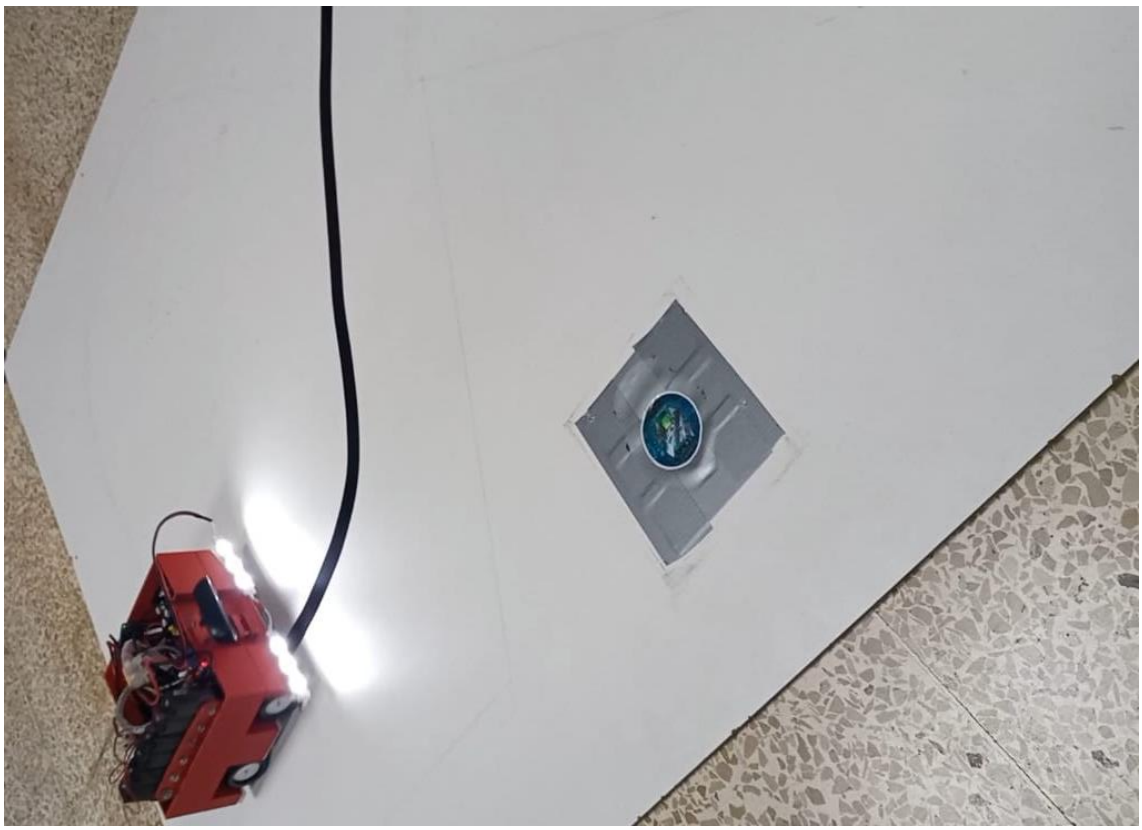

Figura 32. Funcionamiento del robot seguidor de línea en una pista abierta.

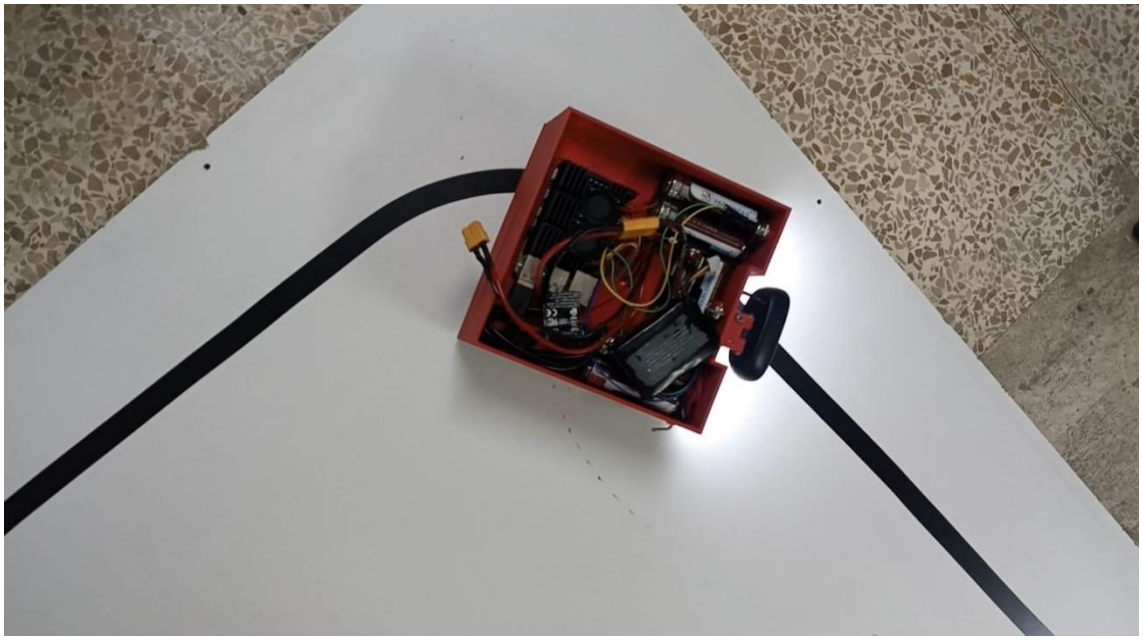

Figura 33. Funcionamiento del robot seguidor de línea en una pista cerrada.

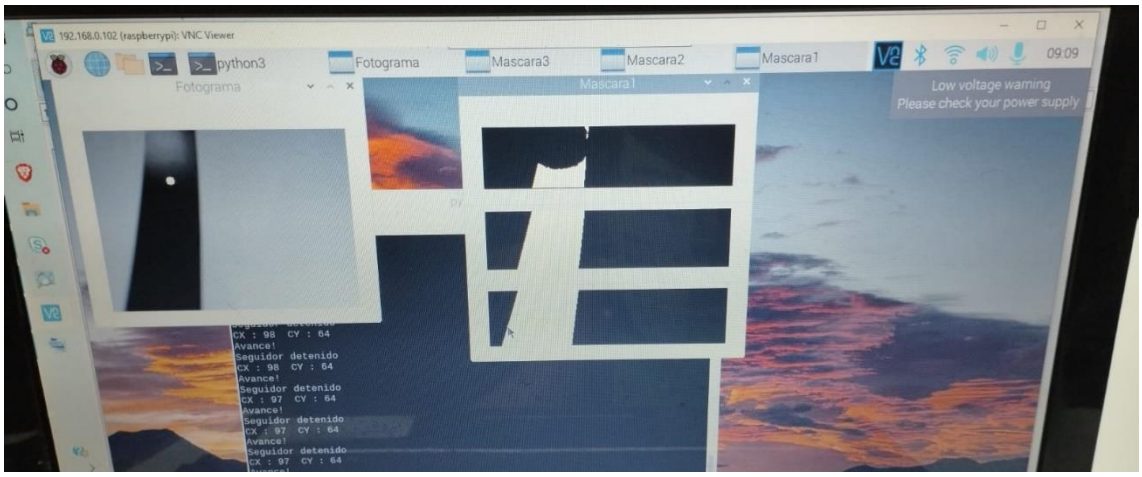

Figura 34. Monitoreo de código para el robot seguidor de línea.

Figura 35. Robot armado y funcionando en su totalidad.