UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE GUAYAQUIL

CARRERA DE INGENIERÍA ELECTRÓNICA

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE:
INGENIERO ELECTRÓNICO

TEMA:

DISEÑO E IMPLEMENTACIÓN DE UN DRONE TERRESTRE CONTROLADO INALÁMBRICO A TRAVÉS DE UNA APLICACIÓN ANDROID

AUTORES:
JOSELYN BELÉN JORDÁN RODRÍGUEZ
ANABELL ALEXANDRA ZAMBRANO CAISE

TUTOR:
ING. BYRON XAVIER LIMA CEDILLO

GUAYAQUIL – ECUADOR
2020

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA

Nosotras, Jordán Rodríguez Josselyn Belén y Zambrano Caise Anabell Alexandra, estudiantes de Ingeniería Electrónica de la Universidad Politécnica Salesiana, certificamos que los conceptos desarrollados, análisis realizados y las conclusiones del presente trabajo son de exclusiva responsabilidad de los autores.

Guayaquil, junio del 2020

Josselyn Belén Jordán Rodríguez
C.I. 0925242620

Anabell Alexandra Zambrano Caise
C.I. 0951412147

CERTIFICADO DE CESIÓN DE DERECHOS DEL TRABAJO DE TITULACIÒN

Nosotras, Jordán Rodríguez Josselyn Belén con cédula de identidad N° 0925242620 y Zambrano Caise Anabell Alexandra con cédula de identidad N° 095141214-7, declaramos bajo juramento que el trabajo aquí descrito es nuestro proyecto técnico de titulación **"DISEÑO E IMPLEMENTACIÓN DE UN DRONE TERRESTRE CONTROLADO INALÁMBRICAMENTE A TRAVÉS DE UNA APLICACIÓN ANDROID"**; que no ha sido previamente presentado para ningún grado o calificación profesional; y que hemos consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedemos nuestro derecho de propiedad intelectual correspondiente a este trabajo, a la Universidad Politécnica Salesiana sede Guayaquil, según lo establecido por la ley de propiedad intelectual, por su reglamento y por la normativa institucional vigente.

Los conceptos desarrollados, análisis realizados y las conclusiones del presente trabajo, son de exclusiva responsabilidad de los autores.

Guayaquil, junio del 2020

Josselyn Belén Jordán Rodríguez
C.I. 0925242620

Anabell Alexandra Zambrano Caise
C.I. 0951412147

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN SUSCRITO POR EL TUTOR

Por medio de la presente.

Yo, Ing. Byron Xavier Lima Cedillo MSc. doy constancia que las Srtas. Josselyn Belén Jordán Rodríguez y Anabell Alexandra Zambrano Caise han desarrollado y elaborado satisfactoriamente el proyecto final de titulación denominado **"DISEÑO E IMPLEMENTACIÓN DE UN DRONE TERRESTRE CONTROLADO INALÁMBRICAMENTE A TRAVÉS DE UNA APLICACIÓN ANDROID"**, que se ajusta a las normas establecidas por la Universidad Politécnica Salesiana, por tanto, autorizo su presentación para los fines legales pertinentes.

Guayaquil, junio del 2020

Ing. Byron Xavier Lima Cedillo MSc.
DIRECTOR DE PROYECTO DE TITULACIÓN

DEDICATORIA

Esta dedicatoria va en primer lugar a Dios que me ha encaminado a tomar cada decisión desde que comencé mi vida universitaria, a pesar de los inconvenientes que tuve en el camino.

A mis padres el Sr. Julio Jordán y la Sra. Raquel Rodríguez, ya que son mi pilar fundamental y apoyo en mi formación académica, me han dado todo lo que soy como persona, mis valores, mis principios, mi perseverancia y mi empeño. Gracias por aquellas noches de desvelo que sin importancia me dieron para culminar mi proyecto. Y a mi hermano por su apoyo y aliento que me ayudaron a no rendirme.

A mi novio el Ing. Joshue Torres que siempre estuvo ahí desde que empecé este proyecto de titulación y me ha apoyado incondicionalmente con la culminación a pesar de las dificultades que se presentaron día a día.

Y finalmente a todas las personas que de una u otra manera se hicieron presente para terminar este proyecto.

Joselyn Belén Jordán Rodríguez
C.I. 0925242620

DEDICATORIA

Dedico este proyecto principalmente a Dios por haberme regalado una familia maravillosa.

A mis padres el Sr. Rafael Zambrano y la Sra. Proscopia Caise que son mi fuente de inspiración y motivación para crecer cada día como persona y profesionalmente, porque creyeron en mí, dándome ejemplos dignos de amor y humildad. Hoy puedo sentir culminada mi meta y les agradezco por siempre estar impulsándome en los momentos más difíciles de mi etapa universitaria de inicio a fin, y el orgullo que sienten por mí fueron mis fuerzas para perseverar hasta el final.

A mis segundos padres la Sra. Asnelia Caise y el Sr. Víctor Chere que me cuida desde el cielo, gracias por haberme ayudado e incentivado a triunfar en la vida y ser una persona de bien.

A mi hermano que siempre ha estado junto a mí, brindándome su apoyo incondicional, siguiendo mis pasos y ejemplo.

Y a todas mis amigas y amigos que me ayudaron dándome ánimos e impulsándome para culminar este proyecto.

Anabell Alexandra Zambrano Caise
C.I. 09514121417

AGRADECIMIENTO

El presente proyecto de titulación lo dedicamos en primer lugar a Dios, por ser el inspirador y darnos fuerzas para seguir y no decaer en este proceso de obtener uno de los anhelos más deseados.

A nuestros padres, por su sacrificio, amor y trabajo en estos largos años, gracias a ustedes hemos luchado por conseguir lo propuesto y también agradecemos por su educación para convertirnos en personas humildes. Para nosotras es un privilegio y orgullo ser sus hijas.

A los familiares por permanecer presentes, acompañándonos y apoyándonos moralmente, todo lo brindado a lo largo de esta gran etapa de nuestras vidas.
Y a todas las personas que han contribuido para que este trabajo se realice con éxito en especial a aquellos que nos abrieron las puertas y compartieron sus conocimientos.

A nuestros profesores quienes le debemos una gran parte de nuestro conocimiento, que nos servirá a lo largo de nuestra vida profesional, también agradecemos a nuestro tutor de tesis el Ing. Byron Lima por brindarnos su apoyo y a la Universidad Politécnica Salesiana por convertirnos en personas de bien.

"La base de toda educación es cuestión de corazón" (San Juan Bosco).

Josselyn Belén Jordán Rodríguez

C.I. 0925242620

Anabell Alexandra Zambrano Caise

C.I. 0951412147

RESUMEN

Año	Título	Alumnas/s	Director de Tesis	Tema de Tesis
2020	Ingeniero Electrónico	Josselyn Belén Jordán Rodríguez Anabell Alexandra Zambrano Caise	Ing. Byron Lima	Diseño e Implementación de un Drone Terrestre Controlado Inalámbricamente a través de una Aplicación Android

En este trabajo de titulación con su tema: “Diseño e Implementación de un Drone Terrestre Controlado Inalámbricamente a través de una Aplicación Android” se propone el desarrollo de manipular el robot con una aplicación y a su vez realizar diferentes prácticas. El prototipo consta con una aplicación android hecha con el software android studio el cual se comunica mediante una red inalámbrica wifi; en la aplicación se puede visualizar diferentes parámetros como posición, velocidad, y estado de la batería. El robot también puede realizar giros, salto, se propone que el procesamiento del streaming de video se realice en el dispositivo móvil portado por el usuario e ir hacia adelante, hacia atrás y hacia los lados.

La velocidad y orientación del dispositivo se controla mediante el sensor mpu6050 el cual ayuda en el proceso de construcción del controlador pid, este a su vez es complementado con un encoder magnético que está acoplado en uno de los motores. El dispositivo consta de un buzzer el cual se activa cuando se encuentre frente a un obstáculo.

Finalmente, se muestran y se discuten los resultados finales, los cuales prueban que la comunicación opera satisfactoriamente de manera correcta.

Palabras claves: buzzer, mpu6050, android studio, drone.

ABSTRACT

Year	Título	Student/s	Director	Topic
2020	Electronic Engineer	Josselyn Belén Jordán Rodríguez Anabell Alexandra Zambrano Caise	Ing. Byron Lima	Design and Implementation of a Wirelessly Controlled Terrestrial Drone through an Android Application

In this degree work with its theme: "Design and Implementation of a Wirelessly Controlled Terrestrial Drone through an Android Application" the development of manipulating the robot with an application and in turn performs different practices are proposed. The prototype consists of an android application made with the android studio software which communicates through a wireless wifi network; different parameters such as position, speed, and battery status can be viewed in the application. The robot can also perform turns, jumps, it is proposed that the video streaming processing be carried out on the mobile device carried by the user and go forward, backward and sideways.
The speed and orientation of the device is controlled by the mpu6050 sensor which helps in the process of building the pid controller, which in turn is complemented by a magnetic encoder that is coupled to one of the motors. The device consists of a buzzer which is activated when you are in front of an obstacle.
Finally, the final results are shown and discussed, which prove that the communication works satisfactorily.

Keywords: buzzer, mpu6050, android studio, drone

ABREVIATURAS

MCU Unidad de control multipunto

IP Protocolo de internet

SDP Protocolo de descripción de sesión

SMP Multiproceso simétrico

PCB Placa de circuito impreso

FTDI Dispositivos de tecnología futura internacional

PWM Modulación de ancho de pulsos

I2C Circuitos inter - integrados

USB Bus serie universal

IMU Unidad de medición inercial

INDICE GENERAL

INDICE DE TABLAS

INDICE DE FIGURAS

ANEXOS

INTRODUCCIÓN

En este trabajo de titulación que se desarrolla para la carrera de Ingeniería Electrónica de la Universidad Politécnica Salesiana, originalmente gracias al constante desarrollo e investigación este avance tecnológico de los drones fue empleado de manera profesional siendo así de gran importancia ya que se ha visto reflejado en el uso de despliegues táctico y seguridad de la población mundial, donde desde su aparición y su primera implementación han demostrado su potencial en distintas áreas además de que poseen un nivel de autonomía y de captura de información, brindándole al mundo una gama de aplicaciones que ayudarán a optimizar la ejecución de actividades.

En las últimas décadas, la tecnología ha avanzado a pasos agigantados en cada uno de los ámbitos que existen en la sociedad. La diversidad de las aplicaciones potenciales, y el incremento de los avances en las últimas décadas, hace predecir un progreso muy acelerado en un futuro.

Esta nueva aplicación para los vehículos terrestre no tripulado ofrece beneficios significativos con la técnica de transmisión de datos inalámbricos tomados en tiempo real al momento de realizar streaming, debido a su funcionamiento el drone necesitará de alguna persona que esté cerca de él para darle una ruta mientras va tomando nueva información a través de la aplicación android.

Debido a lo expuesto anteriormente se diseñó un dispositivo con la finalidad de que los estudiantes de la carrera puedan experimentar, realizar distintas prácticas y destrezas de este tipo de drone desarrollando los conocimientos teóricos y prácticos, la innovación y la utilización de herramientas completas como la tecnología arduino, la cual facilita crear ambientes de aprendizaje y satisfacer las necesidades educativas y formativas personales. El drone implementado es capaz de desplazarse por una superficie plana, podrá realizar saltos, giros y posee una vista panorámica debido a que tiene una cámara la cual estará enlazada en tiempo real con la aplicación.

En el **PROBLEMA,** se plantea las delimitaciones, tanto espacial, temporal y académica.
En el **ESTADO DEL ARTE**, se redactan las justificaciones teóricas conforme al trabajo de titulación.
En la **METODOLOGÍA DE LA INVESTIGACIÓN,** se desglosa detalladamente la solución, también el dispositivo que permita la adquisición de datos, en contexto se muestra la realización del algoritmo para la programación dentro de arduino, la comunicación inalámbrica con el software android studio.
En los **RESULTADOS** se da a conocer el funcionamiento del drone y los beneficios del proyecto técnico.

1 EL PROBLEMA

1.1 Planteamiento del problema

Las redes de telecomunicaciones se han basado en la creación y evolución de medios dedicados a la facilidad de conexión y de ahorro de tiempo de los usuarios brindando así una comunicación eficiente en este caso por medio del wifi. En la carrera de Ingeniería Electrónica de la Universidad Politécnica Salesiana se realizan excelentes prácticas ayudando a los estudiantes para complementar los conocimientos teóricos adquiridos durante cierto tiempo.

Hoy en día es necesario un dispositivo capaz de observar y de recorrer áreas o superficies con una diversidad de fines, para realizar investigaciones, desarrollo e incluso para diversión social.

La adversidad ocurre cuando se trata de trasladar a tecnología actual a la resolución de problemas concretos, la cual permite adaptarse al medio ambiente, la satisfacción de las necesidades esenciales y los deseos de los estudiantes de seguir el ámbito de la innovación.

1.2 Importancia y alcance

El propósito y objetivo del presente trabajo de titulación diseño e implementación de un drone terrestre controlado inalámbricamente a través de una aplicación android, se basa en la autonomía de estos ingenios tecnológicos los cuales pueden llegar a variar según los modelos, tamaños y capacidades de desarrollo.
En la actualidad ya es posible visualizar las imágenes a tiempo real de los drones desde un smartphone al mismo tiempo en el que se controla el prototipo.
El alcance de este análisis ha servido como principal fuente de información sobre el tema, a la hora de aportar ejemplo y conocer en profundidad los programas que se están empleando para uso de los drones con sus diversas funcionalidades.

1.3 Delimitación del problema

1.3.1 Delimitación espacial

La investigación se realizará en la Universidad Politécnica Salesiana sede Guayaquil desarrolladas en el laboratorio de telecomunicaciones, el cual servirá para la manipulación, estudios y prácticas de los alumnos de dichas materias.

1.3.2 Delimitación temporal

El presente trabajo de titulación empezó como punto de partida el mes de mayo del 2018 por un tiempo de 22 meses teniendo como culminación del presente año, por considerar un período que permitirá establecer los objetivos planteados.

1.3.3 Delimitación académica

El alcance del trabajo de titulación contiene conocimiento y estudios adquiridos en el transcurso de la carrera Ingeniería Electrónica, en asignaturas tales como: programación, electiva, sistemas microprocesadores, teoría de control.

1.4 Justificación del problema

El análisis se ha realizado básicamente enfocado a las necesidades del medio, ya que cada día se desarrollan avances científicos y tecnológicos, donde los límites de esta tecnología son infinitos. Estos mecanismos poseen grandes potenciales en diversas áreas ya que pueden desplazarse por zonas con una leve complejidad de acceso, esquivando cualquier tipo de obstáculo mostrando imágenes y datos.

1.5 Objetivos

1.5.1 Objetivo general

- Diseñar e implementar un drone terrestre controlado inalámbricamente a través de una aplicación android.

1.5.2 Objetivos específicos

- Lograr tener comunicación con una red wifi.
- Ejecutar conjuntos de ensayos programados para constatar el funcionamiento del drone.
- Implementar sonido (alarma) en el caso de que el drone esté obstruido mediante buzzer.
- Diseñar un algoritmo para manejar: velocidad, equilibrio, estabilidad, salto del drone.
- Diseñar un controlador PID para la velocidad del drone con un sensor acelerómetro.
- Medir y visualizar en una aplicación android las variables: altura(cm), inclinación(grados), estado de la batería(porcentaje), velocidad(cm/seg) del drone terrestre.
- Diseñar tarjeta electrónica de fuerza, control de velocidad y estabilidad con elementos smd.
- Diseñar una aplicación android para el control del drone terrestre.
- Realizar y monitorear streaming (video) en tiempo real mediante la cámara con la aplicación android.
- Realizar comunicación serie asíncrona con el módulo rs232
- Elaborar #5 prácticas didácticas para demostrar el funcionamiento del drone.

2 ESTADO DEL ARTE

En esta sección se puede encontrar el análisis y exposición de la teoría que sirve como fundamento para el desarrollo del presente trabajo, basado en fuentes documentales y/o trabajos anteriores.

2.1 Drone

Los drones son estructuras mecánicas y electrónicas, conocidas por su habilidad de transportarse por diferentes superficies ya sea aéreas, terrestres o marítimas, aunque no existe definiciones concretas de las mismas, estos pueden detectar e interactuar en su entorno. Como lo es el minidrone de parrot que puede desplazarse en la superficie terrestre, los cuales son capaces de esquivar obstáculos, realizar saltos además de ser estables gracias a la forma de sus neumáticos, siendo manipulados desde un control remoto o un smartphone, también a su cámara incorporada la cual les permite realizar streaming (video) en tiempo real y seguirlo como si uno mismo estuviera al volante.[16]

2.2 Control

Al momento de realizar un control de una variable de proceso se lo puede realizar de dos distintas maneras. Puede ser con un sistema de control en lazo abierto o también con un sistema de control en lazo cerrado. [1]

- **Método en lazo abierto:** La información de las características estáticas y dinámicas de la planta se obtienen en lazo abierto en respuesta a un escalón. [1]

Figura 1: Control lazo abierto. [1]

- **Método en lazo cerrado:** La información de las características del lazo se obtienen a partir de un test en lazo cerrado, generalmente un controlador con acción proporcional pura. [1]

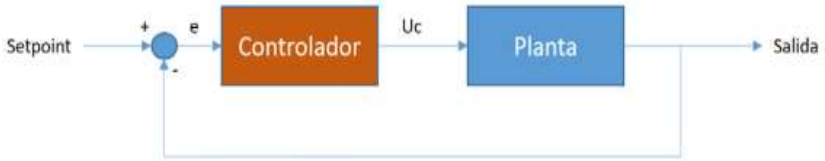


Figura 2: Control lazo cerrado. [1]

Para un sistema tradicional de control, la variable de proceso es el parámetro que se desea controlar. Por lo tanto, mediante el uso de un sensor se obtiene la variable de proceso y se envía a un sistema de control retroalimentado. Esto por lo general se lo que se conoce como un sistema de control en lazo cerrado. [1]

El ingeniero, científico y matemático ruso Nikolai Fyodorovich Minorsky es distinguido por su importante propuesta a la aplicación de los controladores PID.

La idea fundamental detrás de un controlador PID es de leer la variable del proceso en la salida y obtener a la salida del actuador un término de compensación para la entrada sumando la parte proporcional, integral y derivativa. [1]

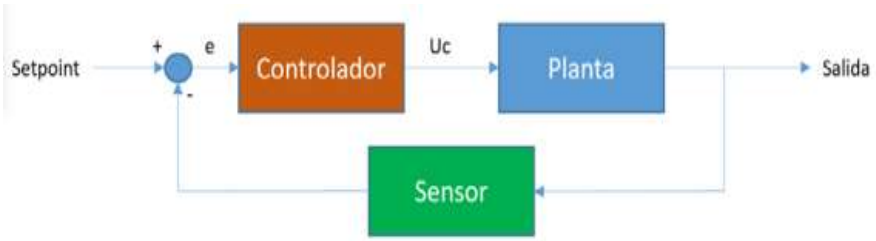


Figura 3: Esquema del control PID. [1]

La diferencia entre la variable de proceso y la referencia que se desea conseguir es utilizada por el algoritmo de control para determinar la salida (Uc(t)) que se envía a la planta del sistema. En donde la salida se obtiene a partir del algoritmo de control pid, el cual está formado por los tres parámetros distintos que son: el proporcional, el integral y el derivativo. Estos parámetros se lo suman para así obtener la salida necesaria para conseguir el control deseado. [1]

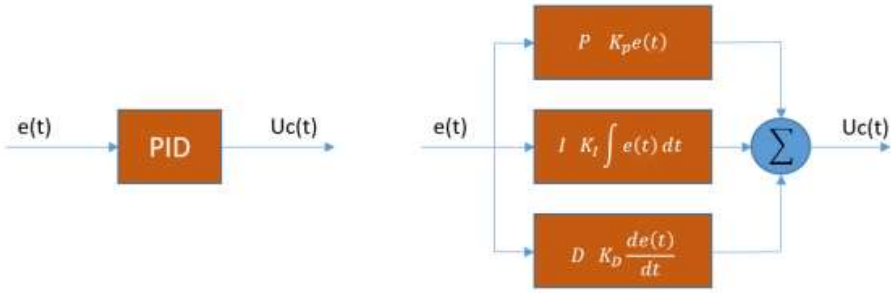


Figura 4: Algoritmo de control de la planta del sistema. [1]

Explicando lo de la imagen anterior se llega a obtener la ecuación general de un controlador PID:

$$\mathrm{Uc(t)} = \mathrm{Kp} \cdot \mathrm{e(t)} + \mathrm{Ki} \cdot \int \mathrm{e(t)}\ \mathrm{dt} + \mathrm{Kd} \cdot \frac{d}{dt} e(t)$$

Al desarrollar lo anterior, se llega a la siguiente solución:

$$\mathrm{Uc(t)} = \mathrm{Kp}\left[\mathrm{e(t)} + \frac{1}{Ti}\sum \mathrm{e(t)} + \mathrm{Td}\Delta\mathrm{e(t)}\right]$$

Donde:

- e(t) = error(t) obtenido a partir de la diferencia entre la referencia(t) y la variable de proceso(t).
- Uc(t) Salida del controlador.
- Kp Ganancia proporcional. ganancia del controlador.
- Ti Constante de tiempo integral.
- Td Constante de tiempo derivativa.

Apartado proporcional: La parte proporcional es el producto entre la señal de error y una constante proporcional para lograr que el error en estado estacionario se aproxime a cero. La parte proporcional no considera el tiempo para generar una corrección de las perturbaciones, buscando mantener la variable controlada en el punto de consigna, denominado setpoint.[1]

$$\mathrm{Psal} = \mathrm{Kp} \cdot \mathrm{e(t)}$$

Apartado integral: En el control integral se disminuye y se elimina el error en estado estacionario provocado por el modo proporcional. Actúa cuando hay desviación entre la variable y el punto de consigna, integrando la desviación en el tiempo y sumándosela a la acción proporcional. [1]

$$\mathrm{Isal} = \mathrm{Ki} \int \mathrm{e(t)dt} = \frac{Kp}{Ti} \int \mathrm{e(t)dt}$$

Apartado derivativo: La acción derivativa actúa si hay un cambio en el valor absoluto del error. La función es mantener el error mínimo corrigiéndolo proporcionalmente a la misma velocidad que se produce. [1]

$$\mathrm{Dsal} = \mathrm{Kd} \frac{d \cdot e(t)}{dt}$$

Cuando se va a definir los valores de las diferentes variables del controlador pid, se debe tener en cuenta las siguientes condiciones:

	Kp aumenta	**Ti disminuye**	**Td aumenta**
Estabilidad	Se reduce	Disminuye	Aumenta
Velocidad	Aumenta	Aumenta	Aumenta
Error estacionario	No eliminada	Eliminado	No eliminado

Tabla 1: Tabla de variables del controlador PID. [1]

Teniendo en cuenta de que los controladores PID son muy utilizados para cualquier tipo de dispositivo, estos tienden a presentar ciertas desventajas cuando se implementan para el control de sistemas en los que sea necesario tener una ganancia del lazo PID (Kp) reducida. Si la ganancia es elevada, el controlador se saturará, por lo que empezará a generar la acción de control máxima posible con valores de error muy pequeños, lo que hará perder precisión a la hora de controlar el sistema. [1]

Por otro lado, el controlador PID puede ser muy sensible a la presencia de ruido. El ruido está formado por pequeñas variaciones de tensión de muy alta frecuencia que pueden afectar especialmente a la parte derivativa del control. Esto se debe a que estas pequeñas variaciones pueden aumentar significativamente si se derivan, por lo que, en muchas ocasiones, en lugar de emplear un controlador PID, se emplea un control PI, que elimina la parte derivativa y soluciona este problema a costa de no poder controlar el estado transitorio del sistema. Otra posibilidad para resolver este problema sería la implantación de un filtro paso bajo que minimice el ruido. [1]

2.4 Módulo esp32

La placa esp32 funciona con un sistema independiente completo o como un dispositivo esclavo de un MCU host, se la utiliza para interactuar con otros sistemas para proporcionar la funcionalidad de wifi y bluetooth a través de sus interfaces. [2]

Este módulo es diseñado para dispositivos móviles, electrónicos portátiles y aplicaciones de IoT, el módulo esp32 alcanza un consumo de energía ultra bajo, también incluye características de vanguardia, varios modos de potencia y escala de potencia dinámica. [2]

Figura 5: Módulo esp32. [2]

2.4.1 Características del módulo esp32

Este módulo está alimentado hasta 3.6v, tiene ROM de 448 Kb y soporta un almacenamiento hasta 16 MB.

- Dos núcleos controlados independientemente con frecuencia de reloj ajustable, que van desde 80 MHz a 240 MHz.
- Bluetooth clásico para conexiones heredadas, compatible con l2cap, SDP, SMP.
- IEE 802.11 b/g/n wifi.
- La salida de +19.5 dBm en la antena asegura un buen rango físico.
- Integra 4mb de flash
- Totalmente certificado con una antena integrada y pilas de software.

Se destaca las siguientes aplicaciones de este módulo:

- Electrodomésticos
- Automatización industrial
- Cámaras IP
- Redes de sensores
- IoT (internet)

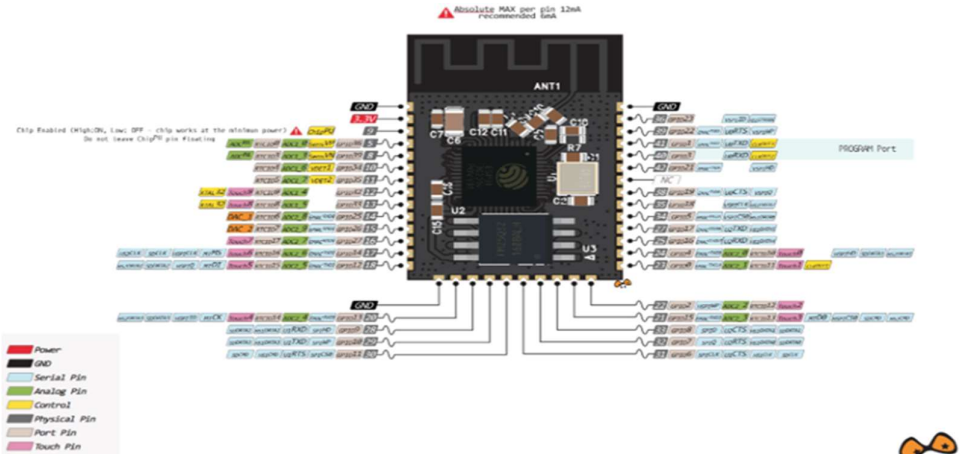


Figura 6: Características del esp32. [2]

2.4.2 Consumo esp32

Tabla de consumo del módulo

Mode	Min	Typ	Max	Unit
Transmit 802.11b, DSSS 1 Mbps, POUT = +19.5 dBm	-	240	-	mA
Transmit 802.11b, OFDM 54 Mbps, POUT = +16 dBm	-	190	-	mA
Transmit 802.11g, OFDM MCS7, POUT = +14 dBm	-	180	-	mA
Receive 802.11b/g/n	-	95 ~ 100	-	mA
Transmit BT/BLE, POUT = 0 dBm	-	130	-	mA
Receive BT/BLE	-	95 ~ 100	-	mA

Tabla 2: Consumo de módulo esp32. [2]

La placa esp32 en gran medida está integrado con interruptores de antena incorporados, balun radio frecuencia, amplificador de potencia, amplificador de recepción de bajo ruido, filtros y módulos de administración de energía. [2]

2.5 Baterías de li-po

Las baterías basadas en polímero de litio consiguen un almacenamiento muy superior de energía. Son también muy ligeras, pesando cerca de la mitad de una batería de ni-cd equivalente. [3]

El uso de estas baterías se ha desarrollado en aparatos como agendas electrónicas, teléfonos móviles, lectores de música y ordenadores portátiles. [3]

Figura 7: Batería de li-po.[3]

2.5.1 Características de la batería li-po

Un elemento de lipo tiene un voltaje nominal, cargado de 3.7v. Nunca se debe descargar una batería por debajo de 3.0v por elemento; nunca se la debe cargar más allá de 4.3v por elemento. Los elementos de lipo se pueden agrupar en serie (s), para aumentar el voltaje total, o en paralelo (p), para aumentar la capacidad total. [3]

El código 3s indica tres elementos en serie; el código 4s2p indica 2 grupos en paralelo de 4 elementos en serie. Como referencia, una batería 2s equivale aproximadamente, en voltaje de salida, a uno de 7 elementos de baterías de ni-mh; una batería 3S equivale aproximadamente a uno de 10 elementos de baterías de ni-mh. Las baterías de lipo, además de especificar su capacidad (en mAh) y su número de elementos, indican la corriente máxima que son capaces de suministrar sin sufrir daños, en múltiplos de c (capacidad). [3]

2.5.2 Ventajas de las baterías li-po

- Las baterías de litio no sufren con el efecto de memoria.
- Densidad de energía de entre 5 y 12 veces las de baterías de ni-cd o baterías de ni-mh, a igualdad de peso.
- Aumento en la duración, típicamente 5-10 min con baterías de ni-cd o baterías de ni-mh hasta entre 20 y 30 min con li-po.
- Cuatro veces más ligeras que las de baterías de ni-cd de la misma capacidad.

2.5.3 Desventajas de las baterías li-po

- Requieren un trato mucho más delicado, bajo riesgo de deteriorarlas irreversiblemente o incluso, llegar a producir su ignición o explosión.
- Precisan una carga mucho más lenta que las de baterías de ni-cd.
- Alto precio, en general alrededor del doble de una batería de ni-cd o baterías de ni-mh.
- Nunca se deberán descargar tan profundamente como las de baterías de ni-cd o baterías de ni-mh, bajo riesgo de deteriorar su capacidad de carga irreversiblemente.

2.5.4 Dispositivos que utilizan las baterías li-po

- Teléfonos móviles
- Ordenadores portátiles
- Cámaras de video
- IPod

2.6 Esp32-cam

Esta placa con sus capacidades inalámbricas de wifi permite realizar streaming de video e imágenes. Puede ser ampliamente utilizado en varias aplicaciones de IoT, adecuado para dispositivos domésticos inteligentes, control inalámbrico industrial, monitoreo inalámbrico. Todo esto es posible gracias a la gran comunidad alrededor de este chip que constantemente está expandiendo sus funcionalidades.
Se utiliza esta tarjeta como un sistema de videovigilancia, para transmitir imágenes desde un móvil, o como sensor para un sistema de visión por computadora básico. [4]

Figura 8: Esp32-cam. [4]

2.6.1 Características del esp32-cam

- Soporta cámaras ov2640 y ov7670 con flash incorporado
- Hasta 240MHz, hasta 600dmips
- CPU de 32 bits y doble núcleo de baja potencia para procesadores
- Módulo ultra pequeño 802.11b/g/n wifi + BT / BLE SOC
- Antena PCB
- Admite interfaces como UART / SPI / I2C / PWM / ADC / DCA
- Admite múltiples modos de suspensión

El módulo esp32-cam no incluye un conector USB, por lo que se necesita un programador FTDI para poder cargar el código a través de los pines our y uot.

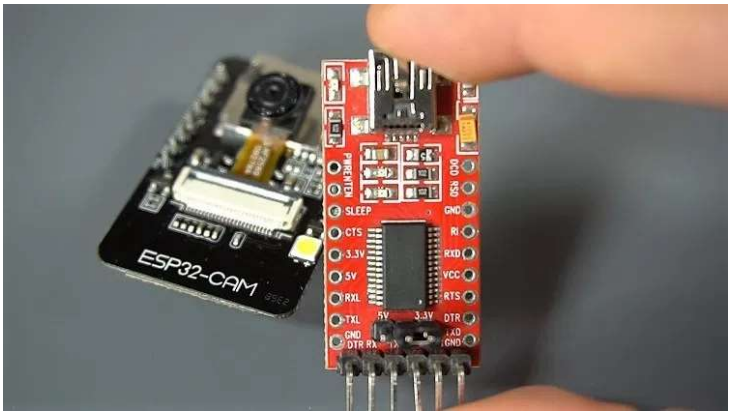


Figura 9: Programador FTDI. [7]

2.6.2 Pines de salida del esp32-cam

En la figura 10 se muestran los pines de salida del esp32-cam, existen tres pines Gnd y dos pines para alimentación de: 3.3v o 5v. Gpio1 y Gpio3 son los pines seriales, se necesita estos pines para poder cargar el código en la placa.
Además, Gpio0 es muy importante ya que determina si el esp32-cam está en modo intermitente o no. Cuando Gpio0 está conectado a Gnd, el esp32-cam está en modo intermitente. [4]

Figura 10: Pines de salida del esp32-cam. [4]

A continuación, los pines que están conectados internamente al lector de tarjetas micro sd:

- Gpio 14: clk
- Gpio 15: cmd
- Gpio 2: datos 0
- Gpio 4: datps 1 (también conectado al led incorporado)
- Gpio 12: datos 2
- Gpio 13: datos 3

2.7 Motor Reductor

Para el proyecto técnico se utiliza motor reductor, el cual es un motor de corriente continua de 6v de alta potencia con escobillas de cárbon de larga duración y una caja de engranajes de metal 51.45: 1. En donde tiene una sección transversal de 10x12mm, y el eje de salida de la caja de engranajes en forma de D tiene 9mm de largo y 3 mm de diámetro. [5]

Figura 11: Motor reductor. [5]

Especificaciones clave:

Voltaje	Rendimiento sin carga	Explotación de puesto
6v	650 rpm, 100mA	0,74 kg·cm (10 oz·in), 1,5A

Tabla 3: Especificaciones de los motores reductores. [5]

Este reducido motor reductor de corriente continua cepillados están disponibles en una amplia gama de relaciones de transmisón, desde 5:1 hasta 1000:1, y con cinco diferentes: motores de alta potencia de 6v y también de 12v con escobillas de carbón de larga duración (hpcb). Los motores hpcb de 6v y de 12v brinda el mismo rendimiento a sus respectivos voltajes nominales. [5]

Las versiones de los motores reductores también están disponible con un eje de salida adicional de 1mm de diámetro que sobresale de la parte trasera del motor. Para este eje trasero de 4.5 mm de largo gira a la misma velocidad que la entrada a la caja de engranajes y ofrece una forma de agregar un codificador, para porporcionar la velocidad o posición del motor. [5]

Figura 12: Motor reductor con eje extendido. [5]

2.7.1 Dimensiones del motor reductor

En términos de dimensiones, estos motores reductores son muy similares a los populares motores reductores na4s corriente directa de 12 mm de sanyo, y los motores reductores con este factor de forma se denomina ocasionalmente de terminal y tapa final ligeramente de diferentes a las versiones con escobillas de metales, pero todas las demás dimensiones son idénticas. [5]

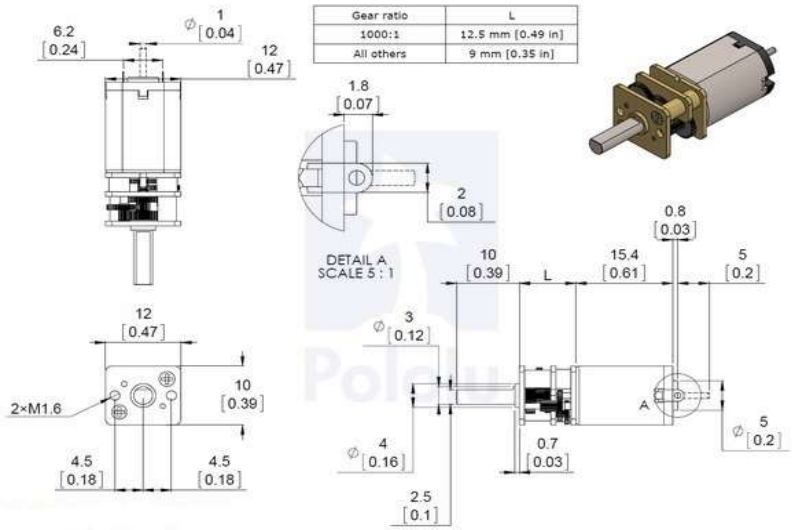


Figura 13: Dimensiones del motor reductor. [5]

2.8 Codificador óptico

El codificador óptico está diseñado para soldarse directamente a la parte posterior del motor, con el eje posterior del motor que esta sobresalido a través del orificio en el medio de la placa del circuito. Cuanto mejor alineada esté la placa, mejor será la calidad de la señal de salida. Para lograr una buena alineación con el motor es pegar la placa a un terminal y soldar el otro terminal cuando la placa está alineada. [6]

Se debe de tener cuidado evitando el calentamiento prolongado de los pasadores del motor, el cual podrían deformar la tapa de plástico o los cepillos del motor. Una vez que el codificador se suelda a los dos terminales, los cables del motor se conectan a las almohadillas m1 y m2 a lo largo del borde de la placa, junto con la potencia de los sensores y las dos salidas en cuadratura. [6]

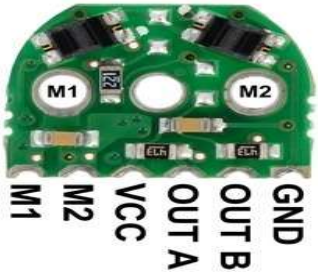


Figura 14: Pines de salida del codificador óptico. [6]

Como se muestra en la figura 15 el codificador óptico se lo puede soldar perpendicularmente al motor, las conexiones de borde tienen una dimensión de 2mm.

Figura 15: Codificador óptico reflectante con motor reductor instalado. [6]

Una vez que la placa se suelda al motor, la rueda de codificador de plástico se puede empujar sobre el eje del motor. La rueda del codificador debe empujarse lo suficiente como para que el espacio entre la rueda y los sensores sea de aproximadamente 0,5 mm. [6]

Las placas de codificador óptico como se muestran en la figura 16 están disponibles en dos versiones, lo que significa que están optimizados para el funcionamiento a 5v y 3.3v. La única diferencia es la configuración eléctrica de los leds en los sensores de reflectancia.[6]

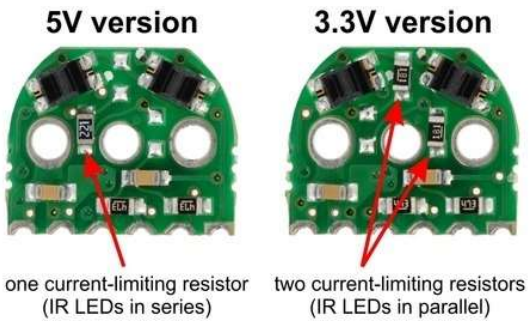


Figura 16: Diferencia de configuración eléctrica de leds. [6]

La versión de 5v con led en serie consume aproximadamente 12 mA, la versión de 3.3 v con led en paralelo consume aproximadamente a 24mA. [6]

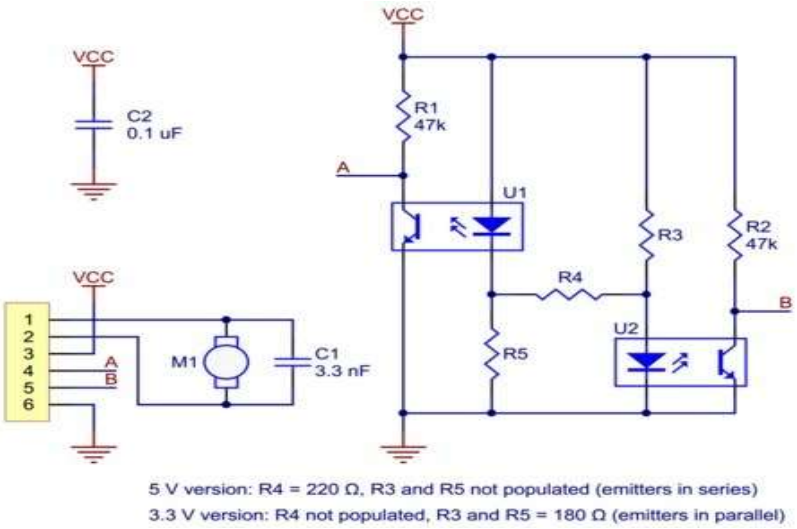


Figura 17: Diagrama esquemático del codificador óptico. [6]

2.9 Unidad de medición inercial

Este sensor, unidad de medición inercial (IMU), está formado por 3 acelerómetros y 3 giroscopios teniendo la posibilidad de añadirle a parte un magnetómetro de tres ejes que lo complemente para medir el ángulo de giro en el eje Z, a continuación, se da una explicación del funcionamiento de cada elemento.
El circuito integrado que se utiliza es concretamente un imu mpu-6050 de invesense montado sobre un tablero de desbloqueo GY-521. [7]

Figura 18: Unidad de medición inercial. [7]

2.9.1 Especificaciones técnicas

- Fuente de alimentación: 3.3v-5v
- Rango del acelerómetro: 2g-4g-8g-16g.
- Rango del giroscopio: 250, 500, 1000, 2000Grad/Seg.
- Rango magnetómetro: 4800µT.
- Interfaz: I2C y SPI
- Conversor AD: 16 Bits (salida digital)
- Grados de libertad: 6
- Regulador de voltaje integrado en placa
- Tamaño: 2.0cm x 1.6cm x 0.3cm
- Aceleración máxima: 10000g

2.10 Giroscopio

El giroscopio consiste en un aparato que mide la orientación de un objeto en el espacio, fue desarrollado por Foucault en 1852 pero ya había ideas parecidas desde 1813, con el alemán Johann Bohnenberger a quien se le atribuye el descubrimiento del efecto giroscópico. Los giroscopios se fundamentan en el efecto coriolis para tomar sus medidas. [7]

El efecto coriolis está presente en todos los cuerpos de rotación, incluida la propia tierra. Este efecto consiste en la distancia percepción del movimiento al observar desde un sistema de referencia rotatorio (no inercial) a un objeto que se encuentra fuera de este sistema de referencia. [7]
La fuerza de coriolis como tal no existe, pero es necesario incluirla si se desea explicar el funcionamiento del sistema con las leyes de newton. Estrictamente podría decirse que los sistemas inerciales como tal no existen, ya que la propia tierra es un sistema no inercial al estar girando sobre sí misma, alrededor del sol, este respecto a la vía láctea, etc.
Sin embargo, se aceptan ciertas consideraciones para simplificar los cálculos y en la mayoría de las ocasiones la tierra se considera un sistema inercial. [7]
La componente del movimiento del cuerpo paralela del eje de rotación no engendra fuerza de coriolis.

Se describe el valor de la fuerza de coriolis como Fc:

$$Fc = -2m\ (w \times v)$$

Donde:
- M: masa del cuerpo.
- W: velocidad angular del sistema en rotación vista desde un sistema inercial.
- X: producto vectorial.
- V: velocidad del cuerpo en el sistema en rotación.

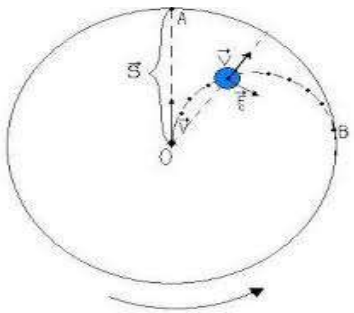
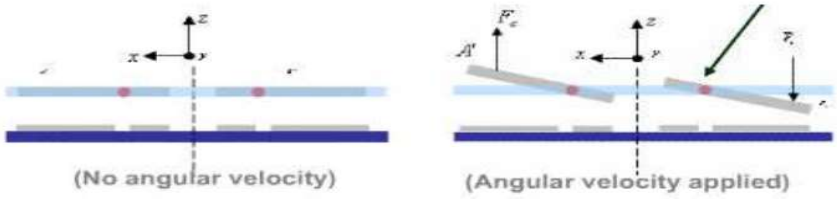


Figura 19: Valor de fuerza coriolis. [7]

En la figura 19 se observa un experimento donde un disco gira, con una velocidad angular w, respecto a un eje perpendicular a la superficie del disco. En la superficie del disco se encuentra una bola de masa m, que se desplaza a una velocidad v, en la dirección que se muestra en la figura, la trayectoria seguida por la bola en la superficie del disco debido al efecto de las fuerzas de coriolis. [7]

El giroscopio dispone de 2 masas conocidas con cierta movilidad lograda mediante materiales flexibles. Para que las masas perciban el efecto coriolis deben desplazarse, esto se logra aplicando una vibración a las mismas mediante la interacción de campos electromágneticos. Al actuar el efecto coriolis el recorrido previsto se ve alterado. [7]



En la figura 20 de velocidad no angular no se aplica ningún movimineto angular, luego las placas no se desplazan debido a ninguna fuerza. Por otro lado en la figura de velocidad angular, se aplica un movimiento angular el cual produce el par de fuerzas de coriolis que desplazan las placas tal y como se observa en la figura 20. Este

Figura 20: Movilidad del giroscopio. [7]

desplazamiento produce una variación en la distancia entre placas del condensador, lo que implica una variación de la capacidad del condensador. [7]

2.10.1 Principales características de los giroscopios electrónicos

A continuación se detallan las principales características de los giroscopios electrónicos

2.10.2 Nivel de cero

Corresponde al nivel de tensión de salida, cuando no hay presencia de velocidad angular. Se suele indicar mediante un nivel de tensión continua a una temperatura de funcionamiento. Por ejemplo 1,23 V a 25 º C. [8]

2.10.3 Sensibilidad

Se mide como la relación entre la entrada y la salida del sensor, cuando se aplica un cambio a la entrada que produce una respuesta a la salida. Se mide en mV/dps. El valor de la sensibilidad es especificado para un rango de exactitud y es variable con la temperatura. [8]

2.10.4 Densidad de ruido

El último parámetro importante a tener en cuenta es la densidad de ruido. La variación del ruido a la salida depende del ancho de banda seleccionado. Reduciéndose este mediante un filtro paso bajo a la salida, se mejora la resolución de salida. Normalmente la densidad de ruido se expresa en términos de "rate noise density" expresado en $\frac{dps}{\sqrt{Hz}}$. [8]

2.11 Acelerómetro

Los acelerómetros miden el efecto de la fuerza de gravedad sobre ellos mismos, al disponerse 3 acelerómetros formando 3 ejes ortogonales, el aumento de percepción de gravedad en uno de los ejes indica que este se está poniendo en vertical, incidiendo toda la fuerza de gravedad sobre él. [7]

Como se sabe los acelerómetros son idóneos para medir la aceleración en tres dimensiones del espacio denominados ejes X, Y y Z. Si se mueve la unidad de medición inercial hacia arriba, marca un dato el eje Z, si es hacia adelante, el eje X y si es hacia atrás, el eje Y. [7]

Suponiendo que la unidad de medición inercial esté perfectamente alineada con el suelo. Entonces, como se observa en la figura 21, el eje Z marca 9.8, y los otros ejes marcarán 0. Ahora suponiendo que gira la IMU 90 grados, es el eje X el que está perpendicular al suelo, por lo tanto marca la aceleración de la gravedad. [7]

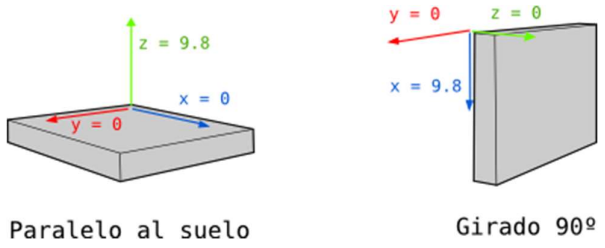


Figura 21: Sensor acelerómetro. [7]

Sabiendo que la gravedad es 9.8 m/s^2, y que mesura dan los tres ejes del acelerometro, por trigonometría es posible calcular el ángulo de inclinación de la imu. Una buena fórmula para calcular el ángulo es: [7]

$$AnguloY = atan\left(\frac{x}{\sqrt{y^2 + z^2}}\right)$$

$$AnguloX = atan\left(\frac{y}{\sqrt{x^2 + z^2}}\right)$$

Figura 22: Ángulo de inclinación de la imu. [7]

Dado que el ángulo se calcula a partir de la gravedad, no es posible calcular el ángulo Z con esta fórmula ni con ningúna otra. Para hacerlo se necesita otro componente: el magnetómetro, que es tipo de brújula digital. El mpu-6050 no lleva, y por tanto nunca podrá calcular con precisión el ángulo Z. Sin embargo, para la gran mayoría de aplicaciones solo se necesita los ejes X y Y. [7]

Este sensor permite realizar la medición de vibración, distancia y velocidad por eso es perfecto para diseñar control de robótica.
El mpu-6050 contiene un acelerómetro, un giroscópico, también un sensor de temperatura a través de un bus de comunicación (I2C), proporciona un bucle de valores conocidos como raw o “crudos” según los datos registrados. [7]

En la siguiente tabla muestra los rangos de escala y valor máximo raw.

RANGO DE ESCALA COMPLETA GIROSCOPIO	SENSIBILIDAD DEL GIROSCOPIO	RANGO DE ESCALA COMPLETA ACELERÓMETRO	SENSIBILIDAD DEL ACELERÓMETRO
250	131	2	16.384
500	65.5	4	8.192
1000	32.8	8	4.096
2000	16.4	16	2.048

Tabla 4: Rangos de escala y valor máximo raw. [7]

La aplicación de un acelerómetro en los mecanismos de un sistema permite visualizar parámetros como:

- Velocidad angular, rpm
- Aceleración, aceleración angular
- Fuerza
- Posición, ángulos
- Bandas de frecuencia
- Energía de impulso

2.11.1 Especificaciones de los acelerómetros

Rango dinámico: Es la amplitud máxima expresada en g, que miden los acelerómetros antes que la señal de salida resulte recortada o distorsionada. [9]

Rango de temperatura: Generalmente es de -50 a 120°C, se encuentra restringido por el microcircuito electrónico que convierte la carga en una salida de baja impedancia. [9]

Gravedad: Es la aceleración que está dada por la gravedad que es 9,8 m/s2. [9]

Rango de frecuencia: Generalmente es de ±5%, en el que la salida del acelerómetro se encuentra dentro de una desviación específica. [9]

Límite de alta frecuencia: Es la frecuencia que supera la desviación de una salida definida. Habitualmente establecida por la resonancia mecánica del acelerómetro.[9]

Corte de baja frecuencia: Es la frecuencia inferior a la salida establecida. La salida no se "corta", pero la sensibilidad disminuye prontamente a frecuencias inferiores. [9]

Ruido: El ruido electrónico es generado por el circuito de amplificación. Cuyo ruido puede ser de banda ancha (por encima de un espectro de frecuencias) o esprectral (a frecuencias específicas). Los niveles de ruido se especifican en "g".
Se puede minimizar el ruido aumentando las frecuencias, es decir, que el ruido es mayor cuando se trabaja a bajas frecuencias que a altas frecuencias. [9]
Sensibilidad: Se refiere a la tensión de salida que es producida por una fuerza definida, existen dos categorías de acelerómetros: los que generan 10mV/g y los que generan 100mV/g. [9]

Sensibilidad de temperatura: Estos sensores contienen un sistema de compensación de temperatura para que si se efectúan cambios en la salida estas mantengan dentro de los límites establecidos para una determinada alteración de la temperatura. [9]

2.12 Micro controlador (Arduino uno)

Para llevar a cabo el proyecto se utiliza la placa arduino uno la cual posee los pines necesarios que facilitan la lectura de dispositivos electrónicos como sensores. Este microcontrolador de código abierto basado en C con procesamiento gratuito cuya destreza es construir dispositivos digitales e interactivos que ayudan a detectar o controlar objetos del mundo real, pero incluso pone a disposición los diagramas al más mínimo detalle por si se decide crear una nueva placa arduino. [7]

Figura 23: Arduino uno. [7]

Arduino uno está basado en un microcontrolador ATmega los cuales son circuitos integrados en los que se puede definir instrucciones, posee una interfaz de entrada que permite conectar diferentes tipos de periféricos y una interfaz de salida encargada de enviar información que se ha procesado en el arduino a otros periféricos. [7]

Cuenta también con 14 entradas/salidas digitales, las cuales 6 de esas se usan como entradas y las otras 6 como salidas de modulación por ancho de pulsos, Además,

incluye un resonador cerámico de 16 MHz, conector USB, conector de alimentación, cabecera de programación serial en circuito (ICSP) y un botón de reinicio. [7]

2.13.1 Características del arduino uno

Las características básicas requeridas son las siguientes:

- Comunicación serie para intercambiar datos con un módulo esp32 (Tx y Rx)
- Comunicación I2C para los distintos sensores, en nuestro caso se utiliza únicamente mpu-6050.
- Pin de alimentación para periféricos a 5V y 3.3V.
- Microprocesador ATmega
- Voltaje de entrada (recomendado): 7-12V
- Voltaje de salida (límites): 6-20V
- Pines digitales 14 de I/O (6 salidas PWM)
- Entradas análogas (6)
- Memoria flash de 32k
- Reloj de 16MHz
- 5 voltios

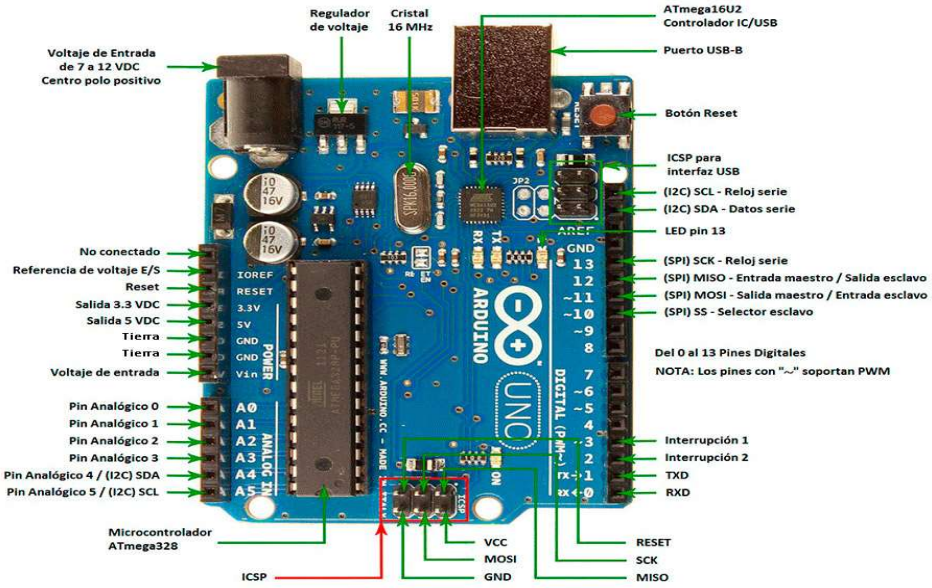


Figura 24: Características y componentes de arduino uno. [7]

Arduino uno es alimentado de diferentes formas, mediante el cable USB que se conecta a un pc o también con una fuente externa. Además, arduino uno cuenta con un zócalo donde se enlaza un jack de 2,1mm en el cual se conecta un adaptador que está en rangos de 7-12v, donde se puede ver en las características que es la tensión recomendada. [10]

Así mismo, posee un conector USB de tipo B, donde se lo conecta al pc y se puede programarlo y alimentarlo. [10]

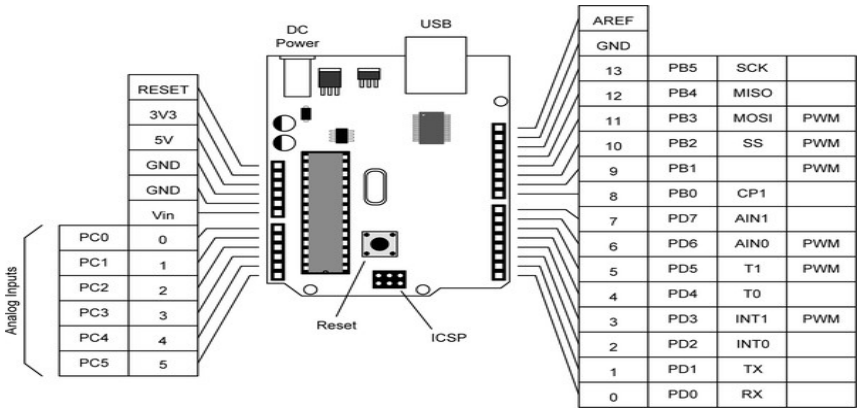


Figura 25: Alimentación del arduino uno. [10]

2.13.2 Diagrama del arduino Uno

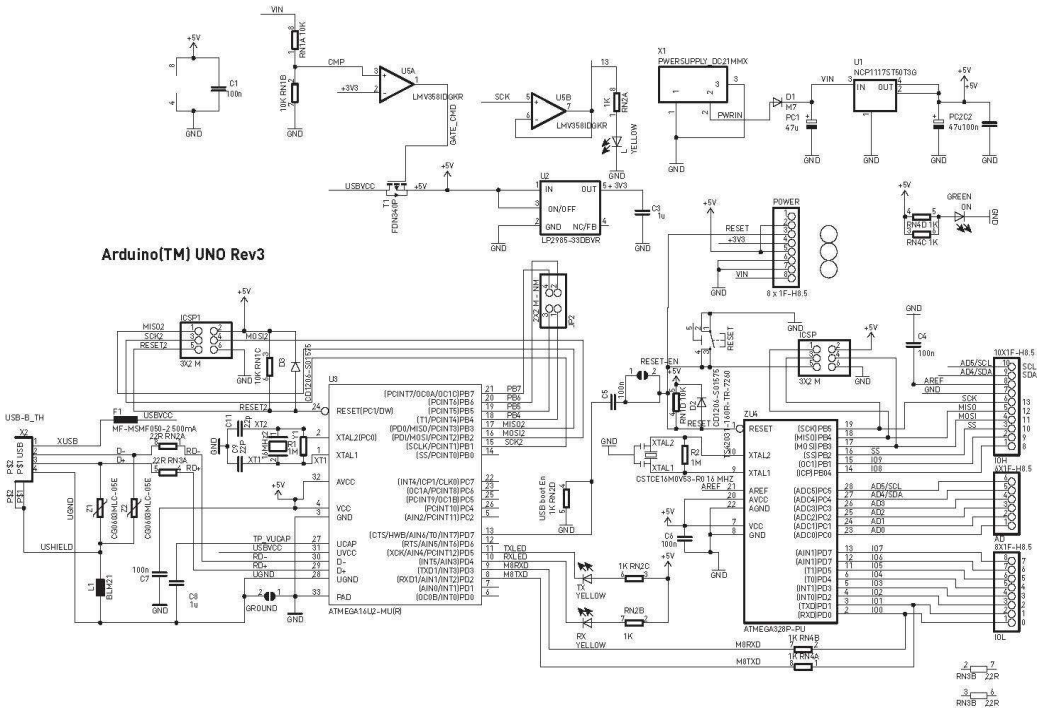
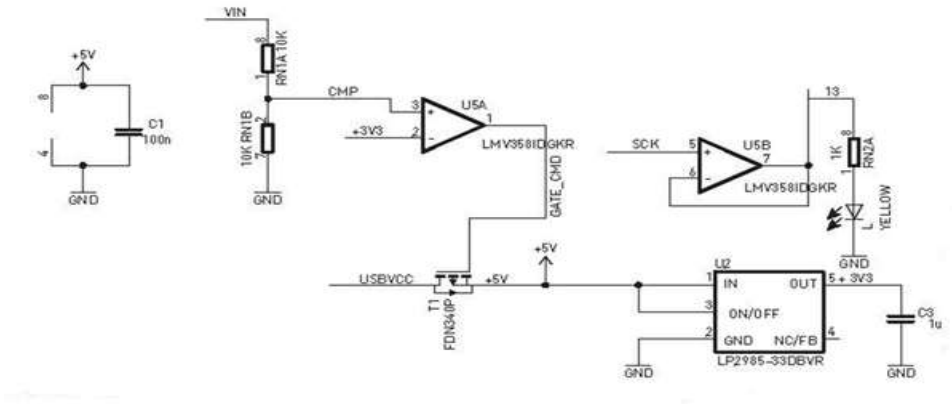


Figura 26: Esquema de arduino uno. [10]

Como se observa en la figura 27, el esquema representa la dirección de voltaje el cual alimenta la placa, y está representado por “USBVCC” para la alimentación del puerto USB [10].

Figura 27: Voltaje de alimentación de arduino uno. [10]



A continuación, el siguiente esquema muestra lo extraído del primer gráfico electrónico, donde está representado la entrada principal de tensión del jack de alimentación [10].

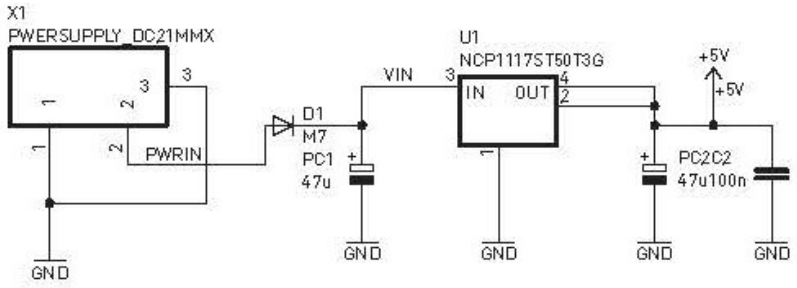


Figura 28: Tensión del Jack de alimentación. [10]

2.14 Sensor sharp

Con el sensor de distancia se adquieren datos que van desde el sharp y cierto objeto que se encuentre de 10 a 80 cm. Está constituido por tres dispositivos: un detector sensitivo de posición, un diodo emisor de infrarrojos y un circuito de señales. Además, es un sensor óptico apto para medir la distancia, en cooperación de un emisor infrarrojo y un receptor miden la distancia usando triangulación. [11]

Para aplicar el método de triangulación se debe medir uno de los ángulos que forma el triángulo emisor-objeto-receptor, el receptor es un detector sensible a la posición en la que halla el punto de incidencia el cual necesita del ángulo y a su vez de la distancia donde se encuentra el objeto [11].

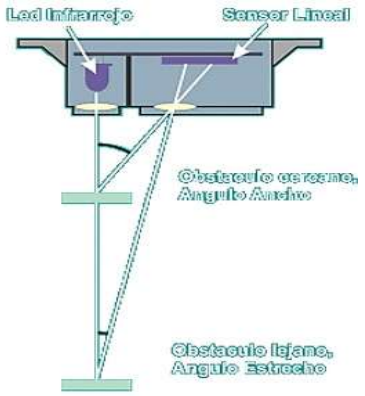


Figura 29: Sensor sharp. [11]

El rango del sensor se encuentra limitado por la geometría del sensor y de su óptica, el sensor sharp (agudo) es debido al rango de visión ya que es demasiado reducido. Esto es ya que la luz emitida es puntual, lo que facilita el uso del sensor para escanear o mapear áreas, teniendo en cuenta que se dificulta detectar objetos pequeños. [11]

El sensor sharp es una luz infrarroja intermitente con una frecuencia definida, que en el receptor es filtrada y descarta cualquier otra fuente de luz diferente a la frecuencia emitida, un punto muy importante es que no son sensibles a la luz ambiental o al sol. [11]

2.14.1 Especificaciones técnicas del sensor sharp

- Rango de distancia 10cm a 80cm
- Voltaje analógico de salida
- Corriente consumida: 30mA
- Voltaje de alimentación: 4.5v a 5.5v Dc
- Conector tipo JST PH de 3 pines
- Tamaño; 29.5x13x13.5mm

2.14.2 Pines de conexión

- Pin 1: Vo (voltaje de salida)
- Pin 2: Gnd (0v)
- Pin 3: Vcc (+5v Dc)

Como se muestra en la figura 24, es lineal la salida del sensor si tiene una forma potencial negativa a partir del rango mínimo.

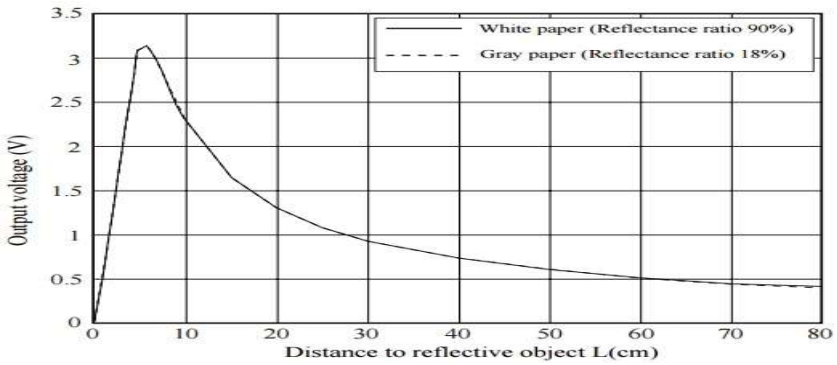


Figura 30: Salida del sensor sharp. [11]

Es recomendable asegurar que el objeto se encuentre a una distancia correcta del sensor para evitar una lectura errónea, ya que para distancias muy pequeñas el sensor arroja valores incorrectos. [11]

Mecánicamente se debe limitar los 10 cm ya que se comporta inestable, por eso se debe colocar el sensor atrás de la posición inicial. [11]

Figura 31: Posición del sensor sharp. [11]

2.14.3 Conexión entre sensor sharp y arduino

SENSOR SHARP	ARDUINO
Pin voltaje de salida	Pin A0
Pin tierra	Pin Gnd
Pin Vcc	Pin +5V

Tabla 5: Conexiones entre el sensor sharp y arduino. [11]

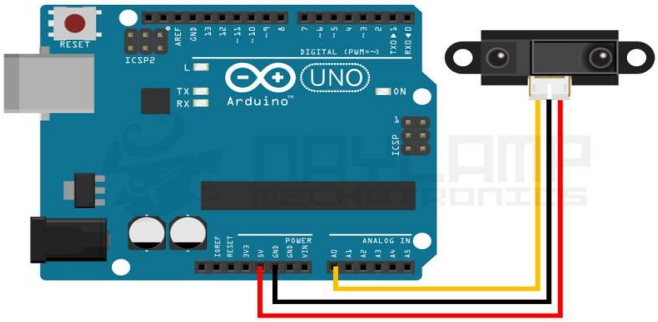


Figura 32: Conexión del sensor sharp y arduino uno. [11]

A esta conexión se le puede agregar un condensador electrolito de 10uF o más entre Vcc y Gnd siempre y cuando esté lo más cerca al sensor para eliminar el ruido en la fuente que provoca el mismo. [11]

2.15 Módulo rs232

Este módulo es ideal para la comunicación entre el puerto serial con un ordenador o un microcontrolador, a su vez soluciona el inconveniente de la diferencia de voltajes ya que el puerto serial rs232 se alimenta con un voltaje de +-12V y un microcontrolador TTL se alimenta con 5V. Transformando los niveles de voltaje bidireccionalmente y de forma trasparente. A través de este puerto serial se realiza la obtención de datos, control de procesos, debugging, etc. [12]

Una de las características principales es su funcionamiento en modo full dúplex el cual se refiere a la transmisión y recepción datos simultáneamente. [12]

Figura 33: Módulo rs232. [12]

2.15.1 Especificaciones técnicas del módulo rs232

- Chip Max3232
- Voltaje de alimentación 3V – 5.5v Dc
- Comunicación serial
- Velocidad máxima de transmisión: 120 Kbps.
- Pines Vcc, Gnd, Tx, Rx.

2.15.2 Puerto serie - paralelo

Se conocen a los interfaces físicos o virtuales como puertos, estos permiten realizar la comunicación entre dos dispositivos o pc. El puerto serie se encarga de enviar datos o información a través de una secuencia de bits, es necesario tener dos conectores Tx (transmisión) y Rx (recepción) para realizar la comunicación de datos. Sin embargo, pueden existir conectores como sincronismo de reloj, referencia de tensión, entre otros. [13]

Mientras que el puerto paralelo se encarga de enviar la información o datos a través de múltiples canales de forma simultánea, es necesario tener un número superior de conectores de comunicación, que cambian según la función del puerto. Además de los puertos de comunicación existen conectores adicionales. [13]

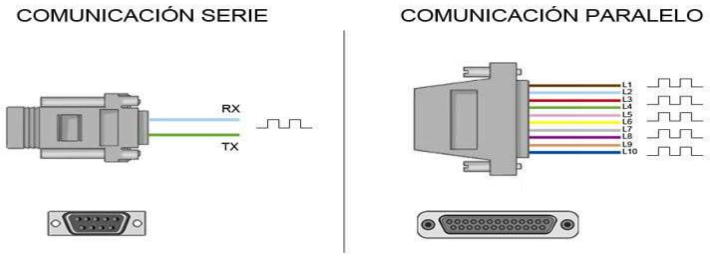


Figura 34: Puertos del módulo rs232. [13]

En los ordenadores poseen los dos tipos de puertos principalmente se han empleado los puertos paralelos para la transmisión de mayores volúmenes de datos. A pesar de ello los puertos series fueron reemplazados progresivamente por los puertos paralelos en la mayoría de las aplicaciones cuando los procesadores se hicieron mucho más rápidos. [13]

La ciencia aplicada para la construcción de circuitos electrónicos digitales se la conoce con el término TTL que significa lógica transistor a transistor donde son bipolares los elementos las entradas y salidas del dispositivo. Se realiza la comunicación cuando Vcc suele ser 3.3v o 5v. [13]

2.16 Transductor electroacústico (buzzer)

Un buzzer permiten convertir una señal eléctrica en una onda de sonido. Estos son dispositivos no contienen una electrónica interna, por lo que se tiene que suministrar una señal eléctrica para obtener el sonido anhelado. [14]

En cambio, los buzzer activos cuentan con un oscilador interno, por esta razón solo se tiene que alimentar el dispositivo para generar el sonido.
Aunque la complicación de generar y controlar la señal eléctrica, los buzzer pasivos y los altavoces tienen la ventaja de lograr diversos tonos emitidos, rectificando la señal que se asigna al altavoz, lo que permite producir melodías. [14]

Figura 35: Transductor electroacústico (buzzer). [14]

Habitualmente, el buzzer pasivo se asocia de una placa para simplificar la conexión, que integra un transistor y resistencias que son inevitables para que funcione el buzzer pasivo o altavoz sin más que conectarlo [14].

2.16.1 Funcionamiento del buzzer y altavoz

Estos transductores electroacústicos son dispositivos que transforman la energía eléctrica en ondas sonoras o sonido. Se diferencian por su funcionamiento.
Los buzzer son transductores piezoeléctricos. A través de estos elementos piezoeléctricos tienen la capacidad de modificar su volumen al ser traspasado por energías eléctricas. [14]

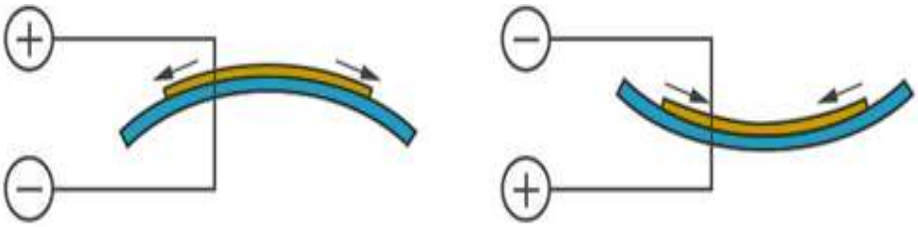


Figura 36: Funcionalidad del transductor electroacústico. [14]

Por lo general un buzzer aprovecha esta apariencia para hacer vibrar una membrana al traspasar el material piezoeléctrico con una señal eléctrica. [14]

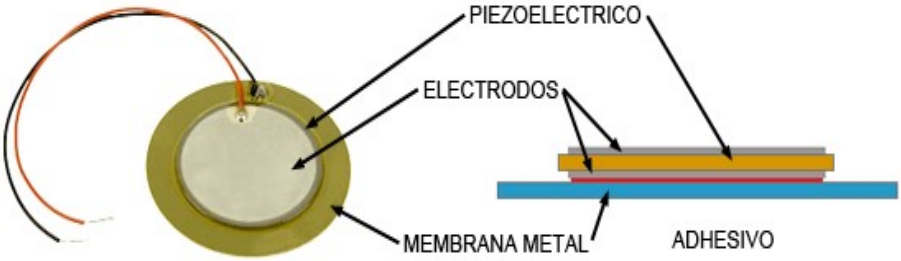


Figura 37: Comportamiento del transductor electroacústico. [14]

Estos buzzer son pequeños y compactos con una alta durabilidad y además con bajo consumo eléctrico. Sin embargo, la calidad de sonido es limitado.
Un altavoz está basado en que su manejo sea en magnetismo, este dispone de un imán fijo que usualmente es seguro a la carcasa. Además, tiene una bobina móvil que se ajusta a una capa flexible. [14]

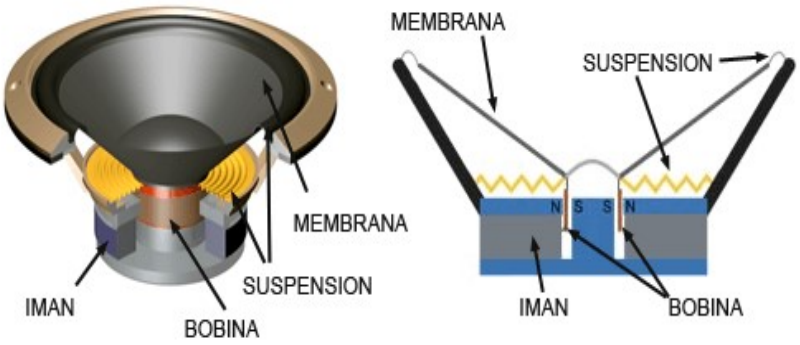


Figura 38: Funcionalidad de un altavoz. [14]

Como la corriente es circular por la bobina, el campo magnético proveniente produce una adherencia con el imán, haciendo vibrar la membrana. Los altavoces exponen una mejor calidad de sonido, sin embargo, en común se necesita una mayor potencia y es imprescindible disponer de dispositivos de amplificación para su uso. [14]

2.16.2 Esquema de montaje

Se alimenta el módulo conectado Vcc y Gnd al arduino, y la entrada de la señal a cualquier salida digital del Arduino. [14]
A continuación, el esquema de conexión visto desde el componente sería el siguiente:

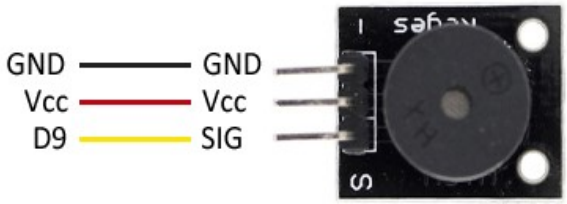


Figura 39: Alimentación del módulo buzzer. [14]

Mientras que la figura 40 muestra el esquema de conexión visto desde el arduino:

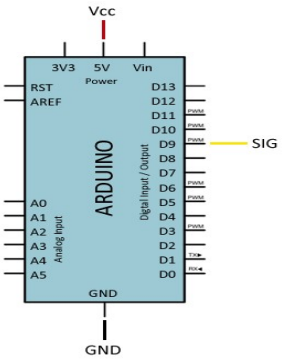


Figura 40: Alimentación desde arduino. [14]

Cuando se usa un altavoz que cumple con una suma corriente, de manera que pueda generar arduino, se considera proporcionar una fase de amplificación, como se ve en la entrada de transistores BJT. [14]

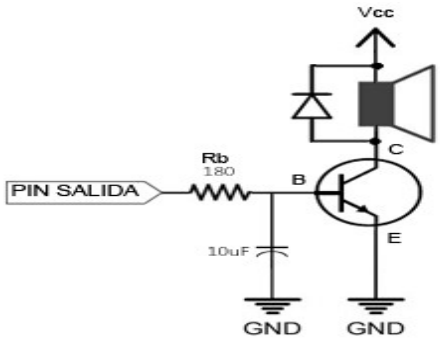


Figura 41: Esquemático de transistores BJT. [14]

2.17 Software Matlab

El software Matlab se ha convertido en una herramienta muy amplia ya que facilita los análisis matemáticos, cuenta con un gran número de instrucciones para resolver problemas científicos. [15]

Matlab es un sistema que realiza dos funciones: una super calculadora y un intérprete de un lenguaje de programación. [15]
También se puede mencionar entre sus aplicaciones a la computación y la matemática se puede aludir el desarrollo de algoritmos, el modelado y la simulación; además la exploración, visualización y análisis de datos, la creación de gráficas científicas entre otras. [15]

Figura 42: Software matlab. [15]

3 MARCO METODOLÓGICO

3.1 Diseño estructural

Para el diseño del robot se usa el software CAD SOLIDWORKS, para el desarrollo de la estructura del prototipo. Consecutivo a esto teniendo ya el diseño se procede a la impresión en 3D del diseño. En las siguientes figuras se puede observar las dimensiones y vistas de cada capa de la estructura del prototipo.

3.1.1 Prototipo (capa base)

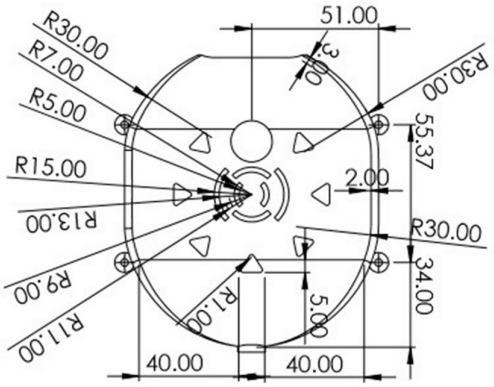


Figura 43: Vista superior capa base del prototipo.

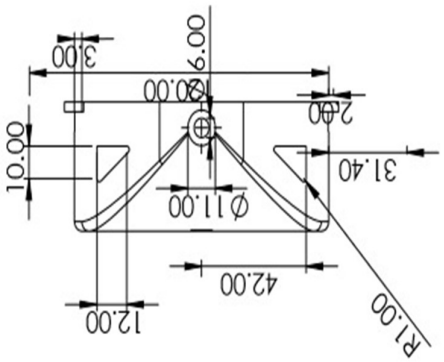


Figura 44: Vista frontal capa base del prototipo.

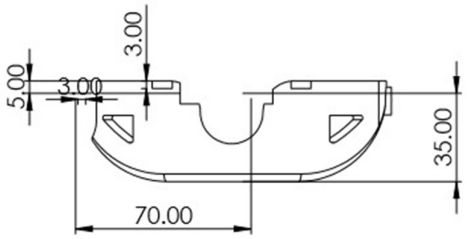


Figura 45: Vista lateral capa base del prototipo.

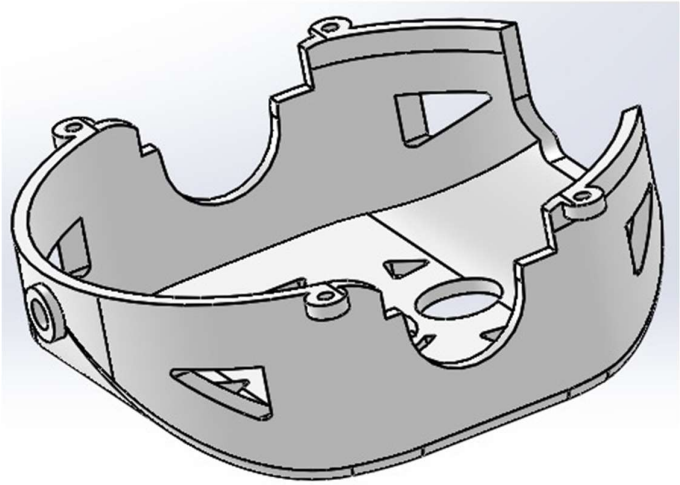


Figura 46: Capa base del prototipo en 3D.

3.1.2 Prototipo (capa media)

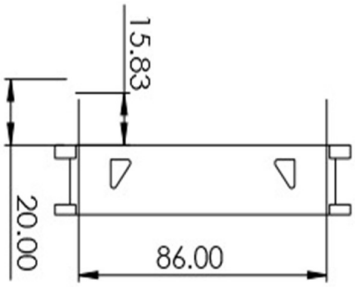


Figura 47: Vista frontal capa media del prototipo.

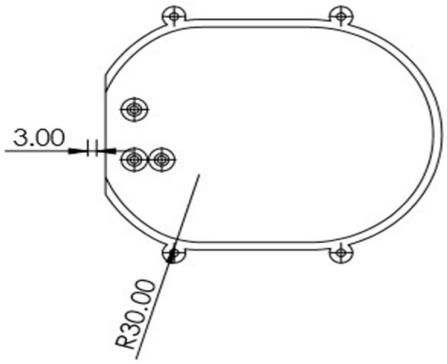


Figura 48: Vista superior capa media del prototipo.

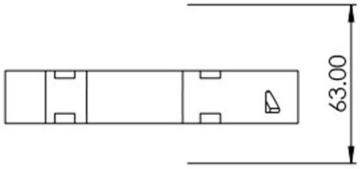


Figura 49: Vista lateral capa media del prototipo.

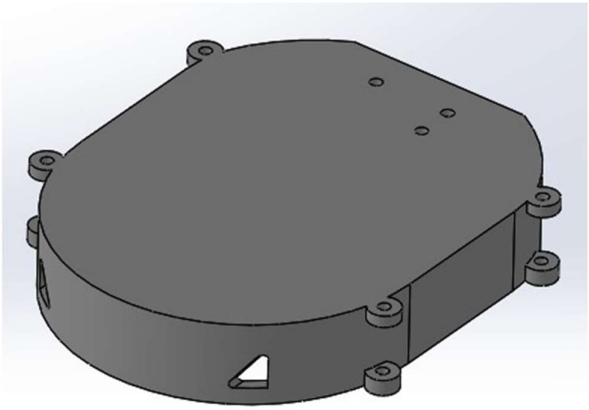


Figura 50: Capa media del prototipo en 3D.

3.1.3 Prototipo (capa superior)

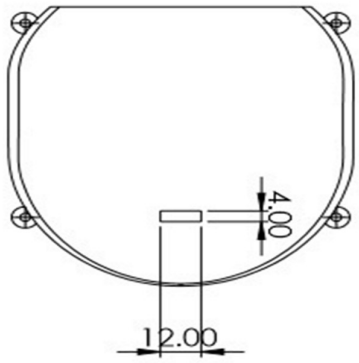


Figura 51: Vista superior capa superior del prototipo.

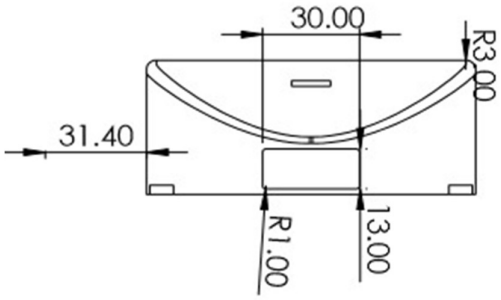


Figura 52: Vista frontal capa superior del prototipo.

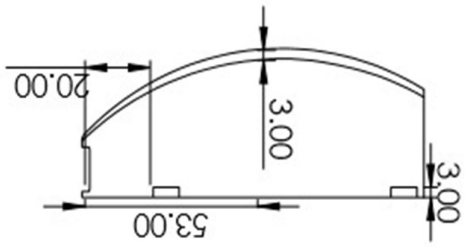


Figura 53: Vista lateral capa superior del prototipo.

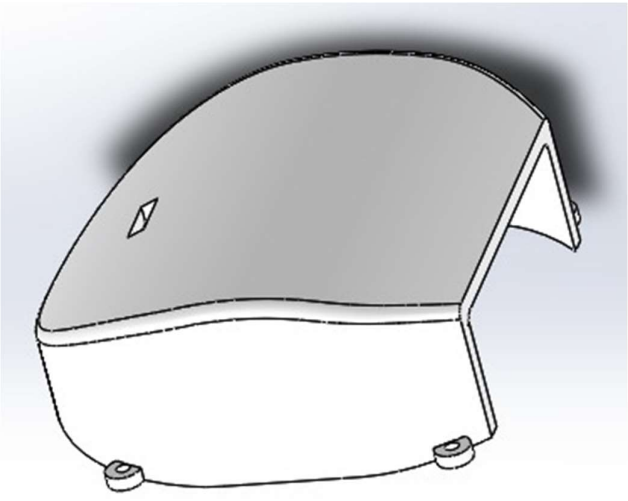


Figura 54: Capa superior del prototipo en 3D.

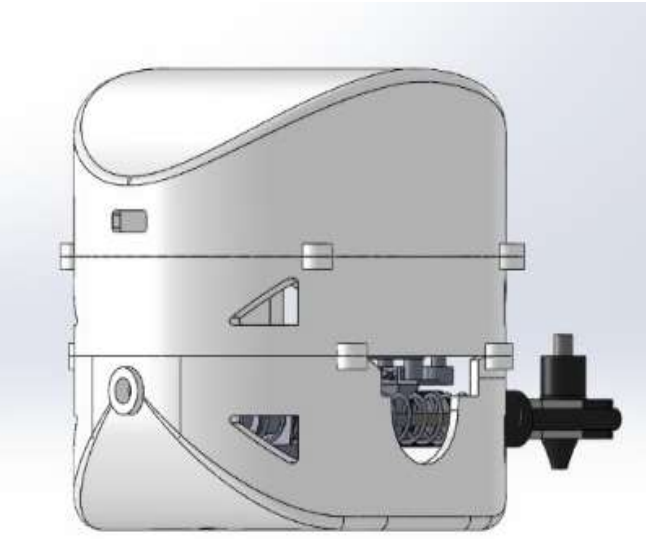


Figura 55: Diseño estructural del prototipo.

3.2 Diseño del mecanismo del prototipo

El diseño de este mecanismo nos ayuda a la compresión y expulsión del resorte para poder realizar el salto del robot. Además, sirve como punto de apoyo cuando el prototipo esté en estado de reposo como se puede observar en las siguientes figuras.

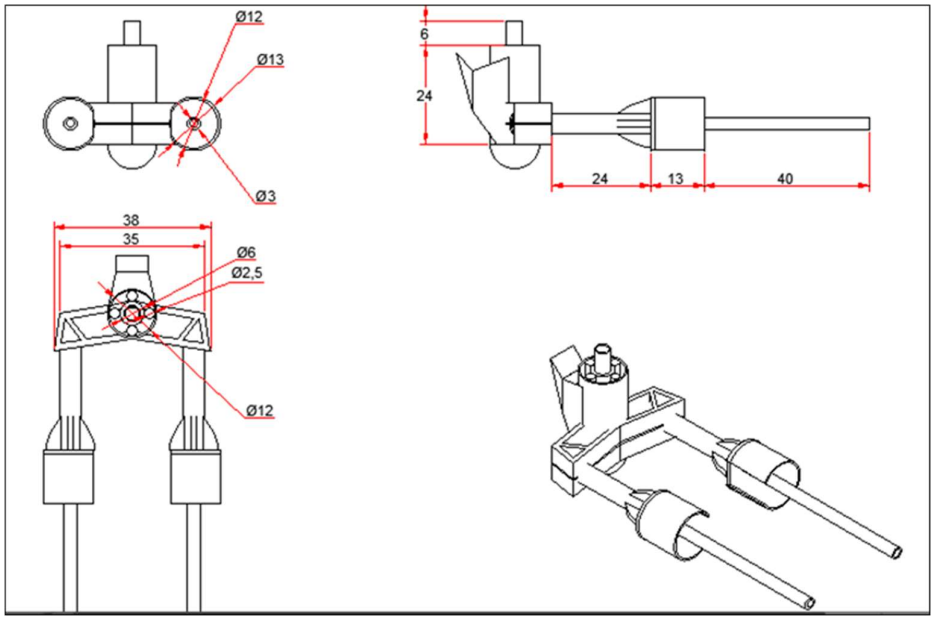


Figura 56: Piezas del mecanismo del robot en el software autocad.

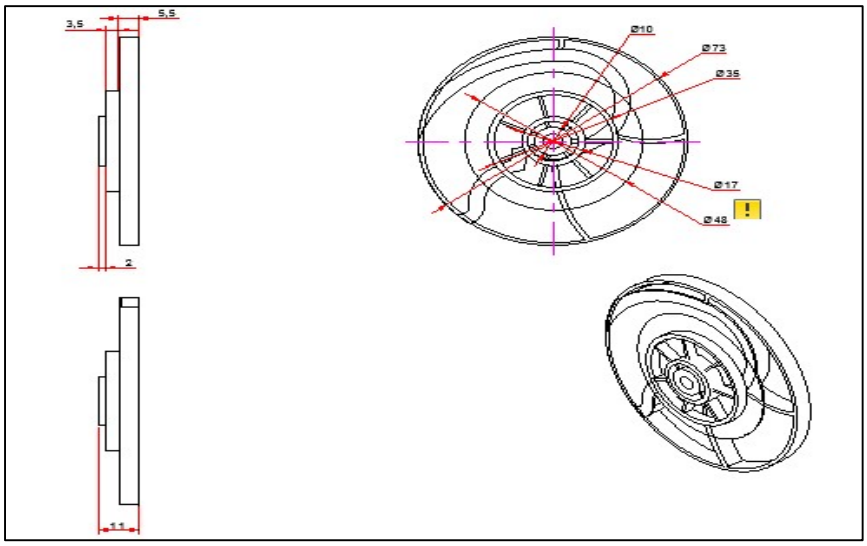


Figura 57: Piezas del mecanismo del prototipo en el software autocad.

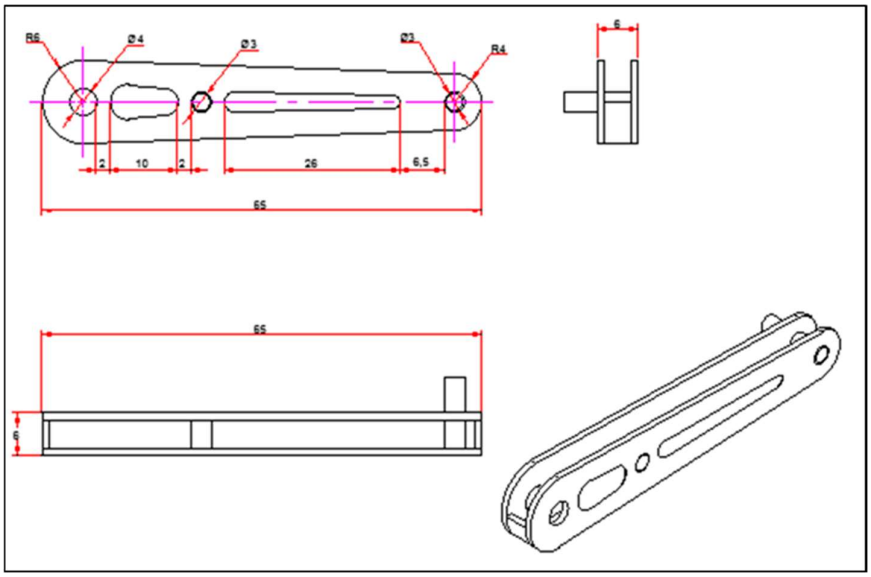


Figura 58: Piezas del mecanismo del prototipo en el software autocad.

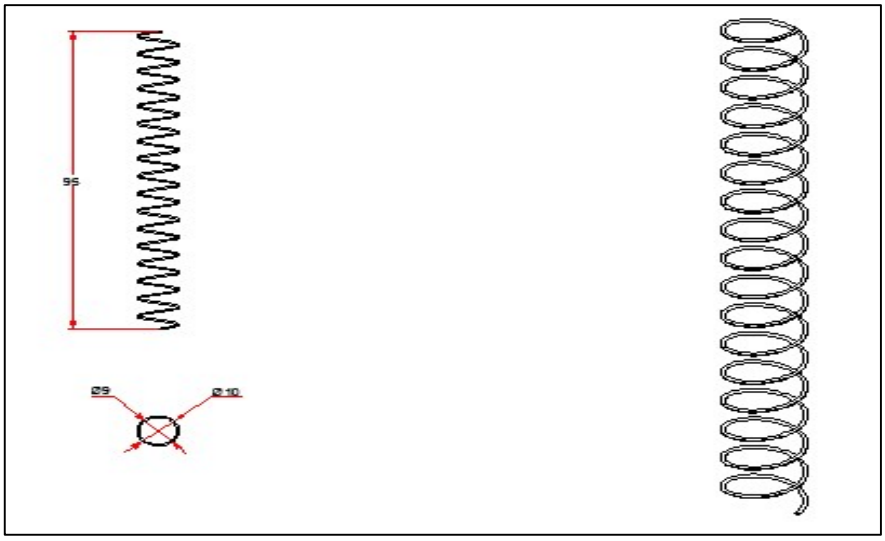


Figura 59: Piezas del mecanismo del robot en el software autocad.

3.3 Diseño electrónico y conexiones

3.3.1 Diseño de tarjeta de circuito impreso

El diseño de la tarjeta se realiza en el programa proteus 8 professional donde cada elemento ya tienen incluido su propio paquete smd, los elementos que componen la placa son: driver de motores (L293D), sensor mpu6050, sensor sharp, buzzer, rs232, batería lipo y pines de conexión.

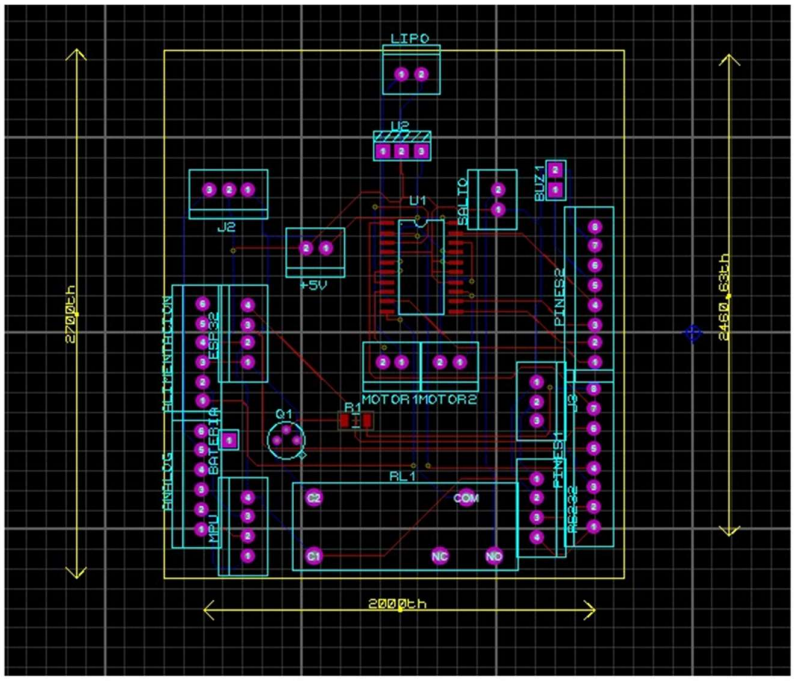


Figura 60: Diseño de tarjeta de circuito impreso.

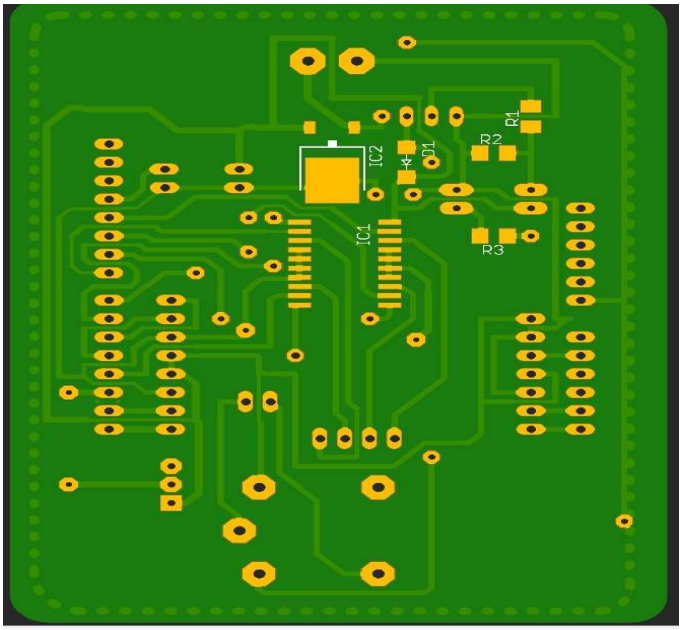


Figura 61: Diseño de tarjeta smd en el software proteus ares 3D.

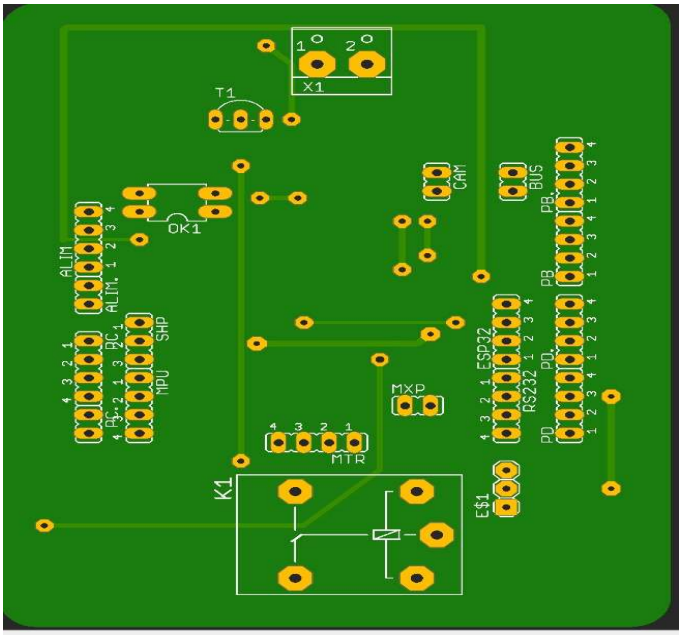


Figura 62: Diseño de tarjeta smd en el software proteus ares 3D.

3.4 Diagrama del drone terrestre

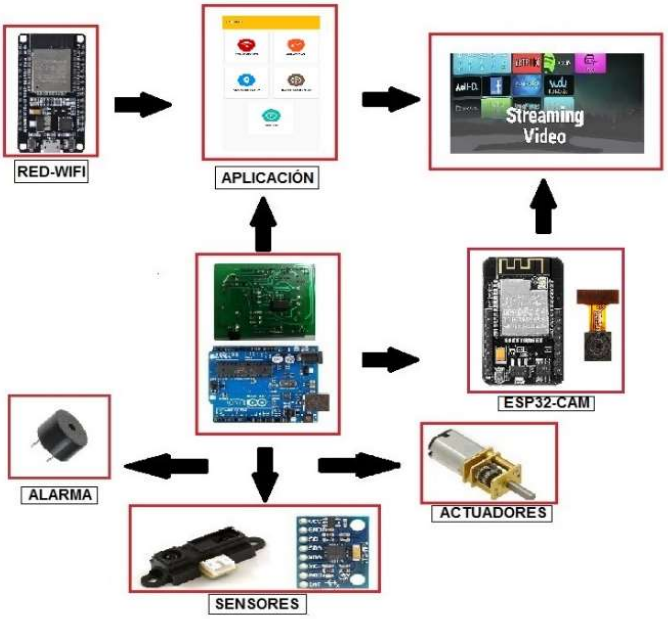


Figura 63: Diagrama de bloques del prototipo.

3.5 Conexiones de los componentes

3.5.1 Conexiones del esp32

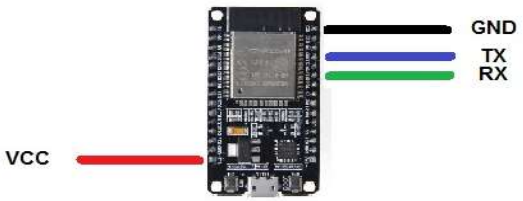


Figura 64: Conexiones del módulo esp32.

3.5.2 Conexiones del mpu6050

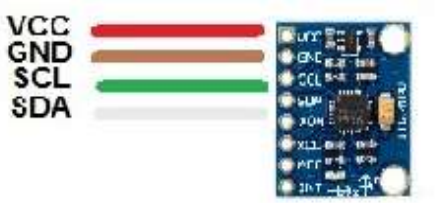


Figura 65: Conexiones del módulo mpu6050.

3.5.3 Conexiones del esp32-cam

Figura 66: Conexiones del módulo esp32-cam.

3.5.4 Conexiones del buzzer

Figura 67: Conexiones del buzzer.

3.5.5 Conexiones de los motores

Figura 68: Conexiones de los motores pololu.

3.5.6 Conexiones del rs232

Figura 69: Conexiones del módulo rs232.

3.5.7 Conexiones del sensor sharp

Figura 70: Conexiones del sensor sharp.

3.5.8 Conexiones del encoder

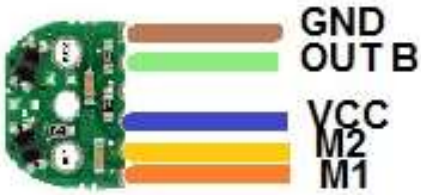


Figura 71: Conexiones del encoder.

3.6 Pines de conexión en arduino

uno

PINES DEL ARDUINO (DIGITALES)	CONEXIONES DE LOS MÓDULOS
0	Módulo rs232 (rx)
1	Módulo rs232 (tx)
2	Encoder
3	Motor izquierdo
4	Módulo esp32 (rx)
5	Módulo esp32 (tx)
6	Enable motor izquierdo
7	Salto
8	Motor izquierdo
9	Enable motor derecho
10	Motor derecho
11	Motor derecho
12	Buzzer

Tabla 6: Pines digitales de conexiones en arduino uno.

PINES DEL ARDUINO (ANÁLOGOS)	CONEXIONES DE LOS MÓDULOS
A0	Batería
A1	Sensor sharp
A4	Módulo mpu6050 (sda)
A5	Módulo mpu6050 (scl)

Tabla 7: Pines análogos de conexiones en arduino uno.

3.7 Diseño de la aplicación android

Se diseña la aplicación android para tener un destacado manejo del dispositivo, teniendo una interfaz interactiva y de fácil uso para los estudiantes de la carrera.

3.7.1 Pantalla inicio (menú)

La aplicación La aplicación se realiza por medio de codificación tal como se puede apreciar en la figura 72, se crea cada botón con las mismas dimensiones y distribuciones correctas en la pantalla inicio para tener una mejor calidad de diseño. Para ver el código completo se puede dirigir al anexo 5.

```java
package e.jordan.presentacion;

import ...


public class Inicio extends AppCompatActivity implements View.OnClickListener {


    private CardView conexionCard, pilotajesCard, mapaCard, instruccionesCard, startCard;
    private static final int PERMISSION_CODE = 1000;
    private static final int IMAGE_CAPTURE_CODE = 1001;


    Uri image_uri;

    public static final String[]requiredPermissions = {
        Manifest.permission.READ_EXTERNAL_STORAGE,
        Manifest.permission.WRITE_EXTERNAL_STORAGE,
    };

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        if(RuntimePermissionsHelper.applicationHasAllPermissions( context: this, Inicio.requiredPermissions))
        {
            setContentView(R.layout.activity_inicio);
            conexionCard = (CardView) findViewById(R.id.Conexion_card);
            pilotajesCard = (CardView) findViewById(R.id.Grafica_card);
            mapaCard = (CardView) findViewById(R.id.Mapa_card);
            instruccionesCard = (CardView) findViewById(R.id.Instrucciones_card);
```

Figura 72: Android studio codificación pantalla inicio.

La pantalla inicio tiene seis botones, donde cada botón se codifica como se muestra en el anexo 6, de acuerdo a una función específica para el control del robot las cuales son: conexiones, mapa de ruta, instrucciones, start, y subir registros como se visualiza en la figura 73.

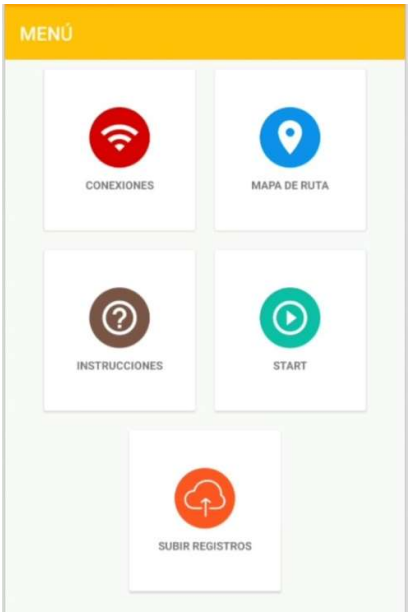


Figura 73: Android studio interfaz gráfica pantalla inicio.

3.7.2 Conexiones

El botón conexión se utiliza para tener la comunicación entre el robot y la aplicación mediante una red inalámbrica dada por el módulo esp32, el código se lo puede observar en el anexo 7.

El método this.wifiScanBroadcastReceiver realiza el escaneo a todas las redes que existen y así tener comunicación. Una vez terminado la codificación del botón conexión, se procede a evidenciar la interfaz y el muestreo de las redes habilitadas

```
icio.java | Graficas.java | Instrucciones.java | activity_registro1.xml | Conexion.java

package e.jordan.presentacion;

import ...

public class Conexion extends AppCompatActivity
{

    public static final String TAG = "Conexion";

    private WifiHelpers wifiHelper;

    private Switch switchWiFi;
    private WifiManager wifiManager;
    private ListView availableWifisListView;
    private Context context;
    private String availableWifis[];
    private Network availableWifisNetworks[];
    private WifiScanBroadcastReceiver wifiScanBroadcastReceiver;
    private SwipeRefreshLayout swipeRefreshLayout;

    private final static int PERMISSIONS_STATE = 1000;

    private void setupObjects()
    {
        Log.e(Conexion.TAG, "setupObjects");
        this.context = getApplicationContext();

        this.switchWiFi = findViewById(R.id.switchWifi);
        this.availableWifisListView = findViewById(R.id.availableWifisListView);
        this.swipeRefreshLayout = findViewById(R.id.swipeRefreshLayout);
```

Figura 74: Android studio codificación botón conexiones.

En conexiones se realiza el escaneo de las posibles redes inalámbricas que se encuentran en el momento como se muestra en la figura 75 y así lograr la conexión con la red que, proporcionada por el prototipo, el código lo puede divisar en el anexo 8.

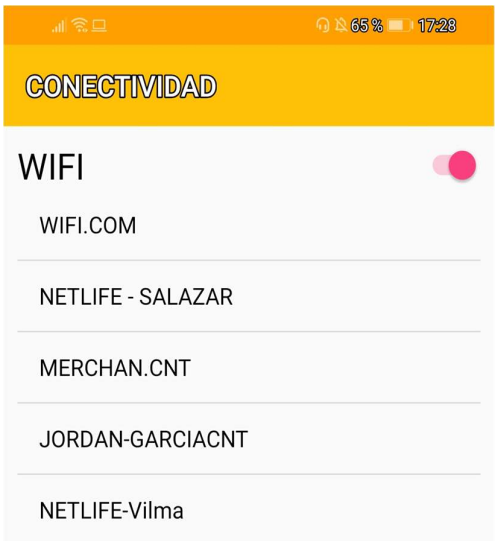


Figura 75: Android studio interfaz gráfica conexiones.

3.7.3 Mapa de ruta

En la figura 76 se muestra el código del botón mapa de ruta mediante la codificación que se visualiza en el anexo 9, se logra el manejo del robot, sus direcciones, el salto y stop.

```
@Override protected void onCreate(Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_mapa_ruta);

    this.toolbar = findViewById(R.id.toolbar);
    this.toolbar.setTitle("Mapa de ruta");
    setSupportActionBar(toolbar);

    this.editTextUrl = findViewById(R.id.editTextUrl);
    this.textViewError = findViewById(R.id.textViewError);

    this.viewPager = findViewById(R.id.viewpager);
    setupViewPager(this.viewPager);

    this.tabLayout = findViewById(R.id.tabs);
    this.tabLayout.setupWithViewPager(this.viewPager);
}
@Override public boolean onCreateOptionsMenu(Menu menu)
{
    getMenuInflater().inflate(R.menu.main, menu);
    return true;
}
@Override public boolean onOptionsItemSelected(MenuItem item)
{
    switch (item.getItemId())
    {
        case R.id.toggleIp:
            this.editTextUrl.setVisibility(this.editTextUrl.getVisibility()==View.GONE?View.VISIBLE:View.GONE);
            return true;
        default: return super.onOptionsItemSelected(item);
```

Figura 76: Android studio codificación de mapa de ruta.

En la figura 77 se puede mostrar la interfaz de mapa de ruta para que el usuario pueda controlar con el robot, por su parte observar la medición de sus variables (velocidad, inclinación, altura y batería), el código completo se encuentra en el anexo 10.

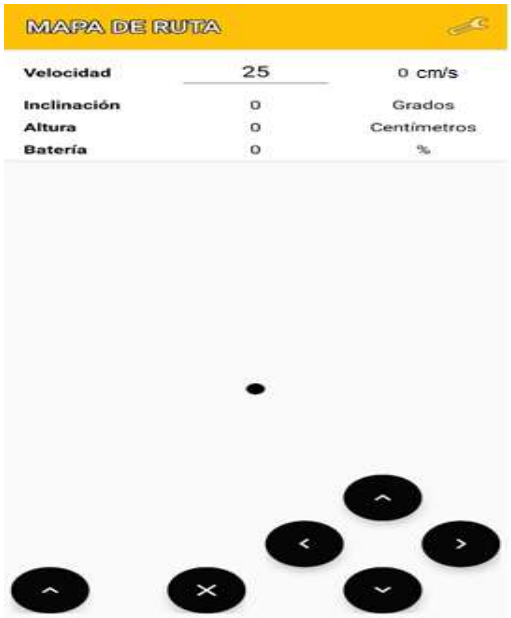


Figura 77: Android studio interfaz gráfica mapa de ruta.

3.7.4 Instrucciones

En el botón se encuentra una lista donde están todas las funciones del prototipo. Se obtiene más información de cada uno de los botones y de su uso para el manejo de una forma correcta. Se puede observar el código completo en el anexo 11

```
Inicio.java | Graficas.java | Instrucciones.java | activity_registro1.xml | Conexion.java

package e.jordan.presentacion;

import ...

public class Instrucciones extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_instrucciones);

        String[] lista ={"CONEXIONES","MAPA DE RUTA","START","SUBIR REGISTROS"};
        ListView listView =(ListView) findViewById(R.id.listViewHistory);

         ListAdapter myadapter = new ArrayAdapter<>( context: this,android.R.layout.simple_list_item_1,lista);
         listView.setAdapter(myadapter);


         listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
             @Override
             public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

                 if(position==0)
                 {
                     Intent intent = new Intent( packageContext: Instrucciones.this,conexiones1.class);
                     startActivity(intent);
                 }
```

Figura 78: Android studio codificación de instrucciones.

En la figura 79, el botón instrucciones muestra información para el fácil uso de la aplicación android, y se podrá visualizar el código en el anexo 12.

Figura 79: Android studio interfaz gráfica instrucciones.

3.7.5 Streaming (start)

En este botón start se visualiza el streaming de video en tiempo real como se aprecia en la fig. 80, según el control de sus funciones: direcciones, salto y stop, además se podrá visualizar el código en el anexo 13.

```
nicio.java | Graficas.java | Instrucciones.java | Start.java | activity_registro1.xml | Conexion.java

package e.jordan.presentacion;

import ...

public class Start extends AppCompatActivity
{
    private Toolbar toolbar;
    private WebView myWebView;
    private LinearLayout linearLayoutEditTexts;
    private EditText editTextDroneUrl;
    private EditText editTextRaspberryUrl;
    private TextView textViewError;
    private SwipeRefreshLayout swipeRefreshLayout;
    private RequestQueue requestQueue;

    public static final String UP = "UP";
    public static final String DOWN = "DOWN";
    public static final String LEFT = "LEFT";
    public static final String RIGHT = "RIGHT";
    public static final String FRONT = "FRONT";
    public static final String BACK = "BACK";
    public static final String STOP = "STOP";
    private String direction = ControlFragment.STOP;
    private boolean jump = false;

    private Javascript.Interval interval;

    private void setupHttpError() { this.textViewError = findViewById(R.id.textViewError); }
    private void onHttpError(String message)
    {
        this.textViewError.setText(message);

Start
```

Figura 80: Android studio codificación del streaming (start).

Como se muestra en la figura 81 se realiza el streaming de video en tiempo real el cual permite que el robot se desplace gracias a los botones de direcciones, salto y stop, el código completo lo puede observar en el anexo 14.

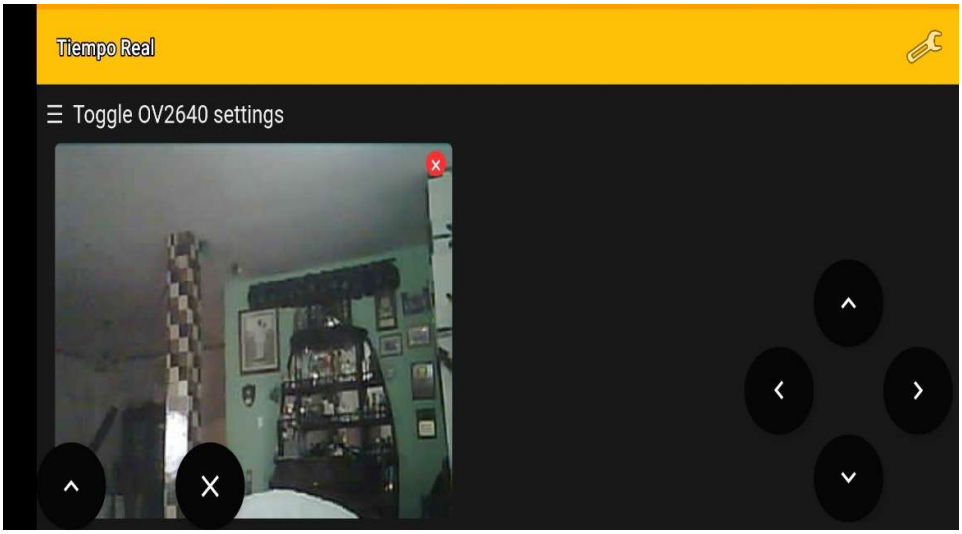


Figura 81: Android studio interfaz gráfica del botón start.

3.7.6 Actualizar base de datos

Este botón realiza la subida de datos a la nube los cuales son obtenidos dentro del botón mapa de ruta para la velocidad, inclinación, y altura del drone terrestre para luego poder ser visualizados en el software matlab.

Figura 82: Android studio codificación de subir registros.

3.8 Control PID de los motores

3.8.1 Función de transferencia del motor

En el software matlab con el comando ident se obtiene la función de transferencia.

```
Transfer Function Identification
Estimation data: Time domain data mydata
Data has 1 outputs, 1 inputs and 100 samples.
Number of poles: 2, Number of zeros: 1
Initialization Method: "iv"


Estimation Progress

Initialization complete.

Nonlinear least squares with automatically chosen line search method
---------------------------------------------------------------------------------------
                          Norm of     First-order     Improvement (%)
Iteration      Cost      step        optimality    Expected  Achieved   Bisections
---------------------------------------------------------------------------------------
    0       11692.6         -      5.68e+03    0.00248          -        -
    1       10179.5      7.38      4.89e+03    0.00248       12.9        2
    2       9739.51      5.18       4.4e+03    0.00181       4.32        3
    3       9691.36     0.791      4.35e+03    0.00164      0.494        6
    4       9679.51     0.201      4.34e+03    0.00162      0.122        8
    5       9679.32   0.00316      4.34e+03    0.00161    0.00191       14
---------------------------------------------------------------------------------------
Estimating parameter covariance...
done.

Result

Termination condition: Near (local) minimum, (norm(g) < tol).
Number of iterations: 5, Number of function evaluations: 44

Status: Estimated using TFEST with Focus = "simulation"
Fit to estimation data: 48.72%, FPE: 10840.8
```

Tabla 8: Obtención de datos de la función de transferencia en el software matlab.

Inicia el programa con la limpieza de pantalla, se procede a cargar la obtención de datos y mostrar el mensaje de la función de transferencia.

```
clear all
close all
clc

%% Obtencion de datos
load('C:\Users\Josselyn\Desktop\LYN_Materias\Tesis\Matlab\velocidad.mat');
load('C:\Users\Josselyn\Desktop\LYN_Materias\Tesis\Matlab\pulso.mat');
load('C:\Users\Josselyn\Desktop\LYN_Materias\Tesis\Matlab\tf1.mat');
load('C:\Users\Josselyn\Desktop\LYN_Materias\Tesis\Matlab\P1IZ.mat');

%% Funcion Transferencia
disp('Modelo de proceso')
disp('Integrador y Zero: ')
P1IZ

disp('-------------------------------')

disp('Funcion de transferencia')
disp('2 polos - 1 zero: ')
tf1
```

Figura 83: Obtención de datos de la función de transferencia.

```
disp('-------------------------------')

disp('Funcion de transferencia')
disp('2 polos - 1 zero: ')
tf1

%% Valores pidtunner con planta TF
%P1IZ
kp = 39.9801;
ki = 107.7312;
kd = 0;
PDI = tf([kd kp ki],[1 0]);
disp('Control Proporcional Integral Derivativo Planta TF')
Control41 = feedback(PDI*P1IZ,1)

%gráfica
figure(7)
step(Control41)
title('PIDTunner')
legend('tf1 - PIDTunner')
```

Figura 84: Control PID.

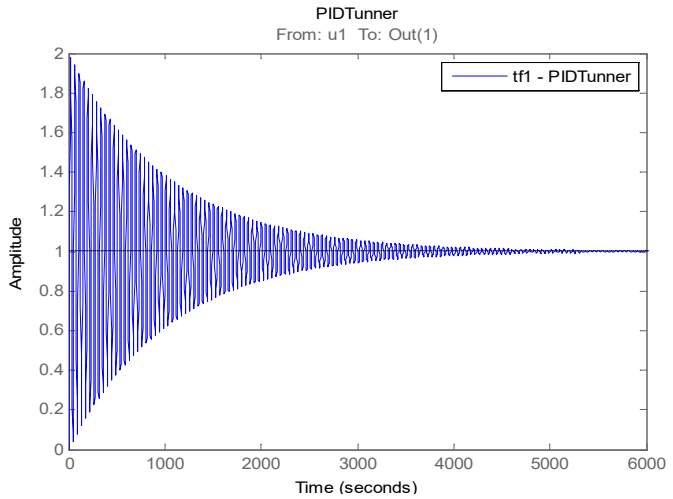


Figura 85: Gráfica de la función de transferencia.

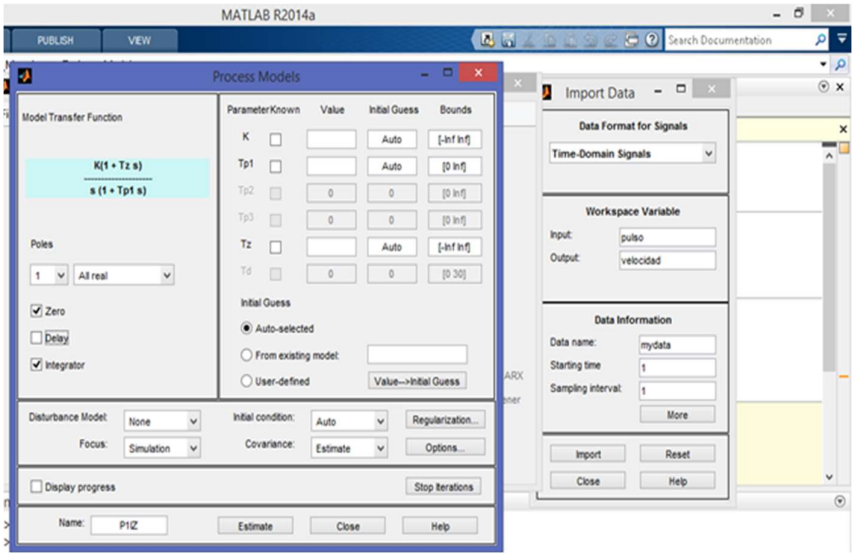


Figura 86: Control PID tuner proceso de modelación.

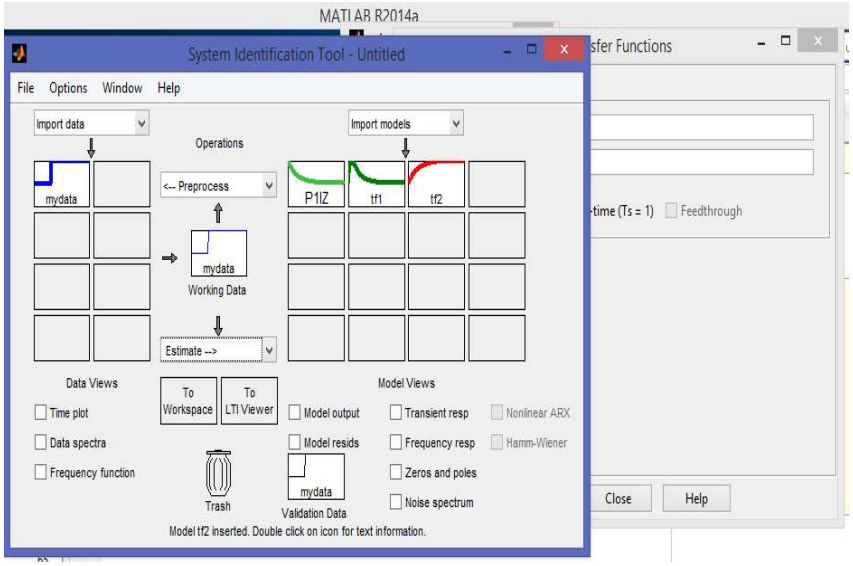


Figura 87: Creación modelos de sistemas dinámicos.

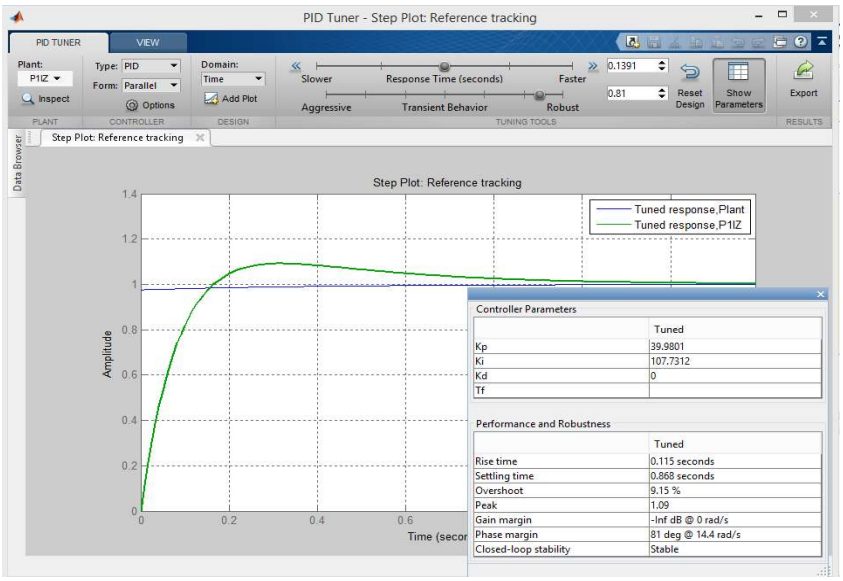


Figura 88: Demostración gráfica del PID tuner.

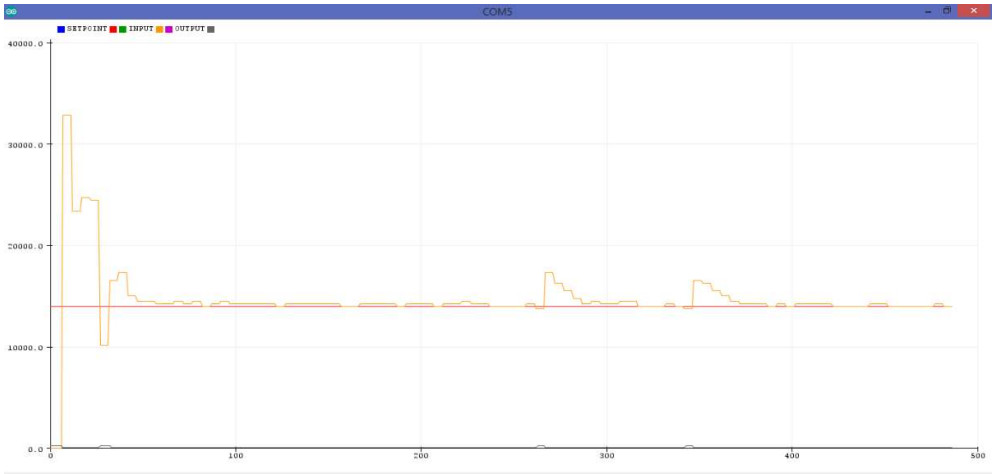


Figura 89: Gráfica del control PID.

Se realiza varias pruebas con diferentes tipos de controladores (PI PD PID) y también variando dichos valores para el mejor control de la planta.

3.9 Implementación del controlador en arduino

En arduino uno se realiza el control PID para los motores; Se inicia colocando las variables y librerías necesarias para realizar el control. Se define 3 tipos diferentes de variables que son de tipo “double” y “volatile” necesarias para el algoritmo del control y de tipo “int” para los motores y el sensor mpu-6050.

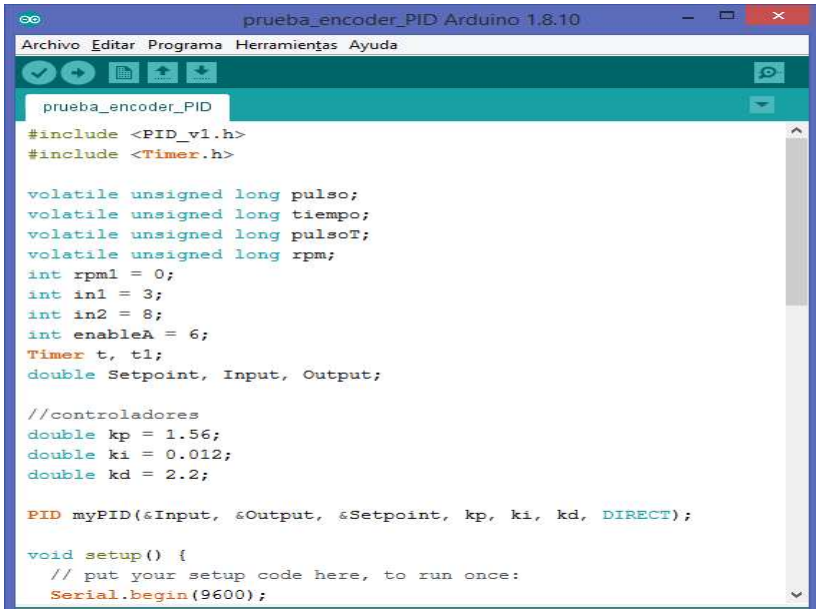


Figura 90: Ingreso de datos en el control PID.

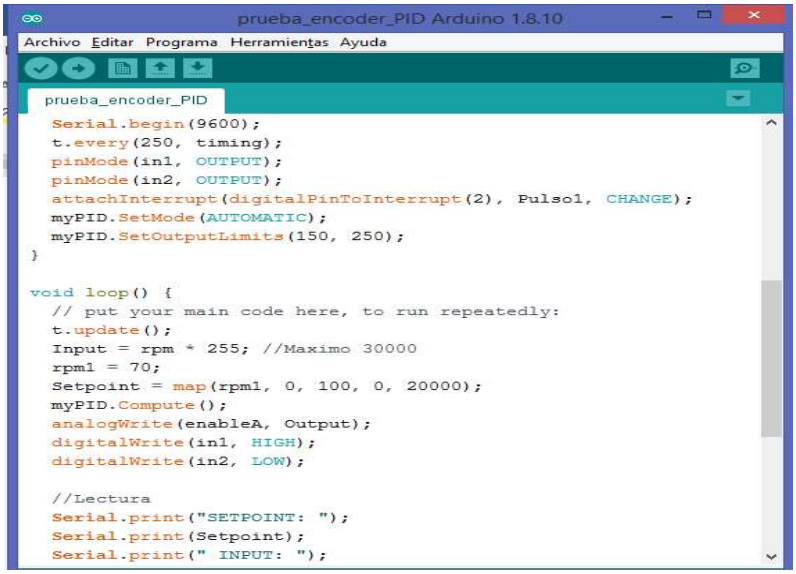


Figura 91: Inicializar comunicación serial.

Como se muestra en la Figura 91, se establece la velocidad de comunicación entre el arduino y serial, se hace uso de una variable “t.every” la cual indica que cada 250 milisegundos actualiza el tiempo en nuestra clase “timing”, teniendo una interrupción en el pin 2 donde se dirige a la clase pulso1 siempre y cuando haya un cambio, además inicializando el PID con el comando “mypid” y estableciendo parámetros para que tenga modo “automático”, el “void loop” es el que se ejecuta siempre cuando está alimentada la tarjeta electrónica arduino y se parametriza como salidas los pines que están conectados a los motores.

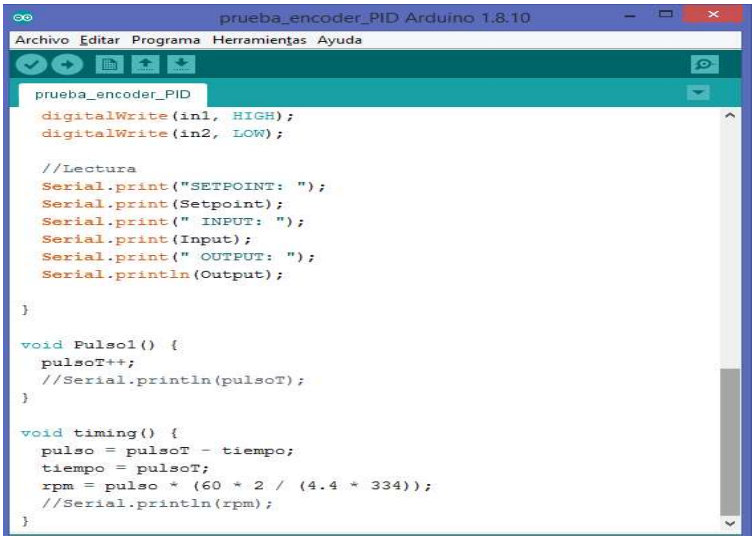


Figura 92: Lectura de datos.

En esta parte de la codificación del control con el comando “digitalWrite” se realiza el control el sentido de los motores high y low, luego la lectura mediante el puerto serial para visualizar los datos, en clase “pulso1” realiza el incremento de mi “pulsoT” es básicamente un contador. En el “void timing” son los pulsos que se obtiene según el tiempo que se demore en recopilar el dato y así las revoluciones finales transformadas a minutos.

3.10 Comunicación módulo-arduino en puerto serial

Datos recibidos de la comunicación entre el módulo esp32 y arduino en el puerto serial. Como se puede observar en la figura 93 su **speed** es 0, eso indica que no se le ha variado la velocidad y además su posición se encuentra en stop como se indica en **dirección >7<.**

Figura 93: Comunicación de módulo esp32 - arduino.

Para esta imagen primero ingresa al botón de gráficas de la aplicación para poder variar su **speed** a **33,** con una **VELOCIDAD** de **>84<**, eso indica que se le ha variado la velocidad.
Como se puede observar en esta imagen su **speed** es 42, indica que se le ha variado la velocidad y además su posición se encuentra en **FRONT** como se indica en **dirección >1<.**

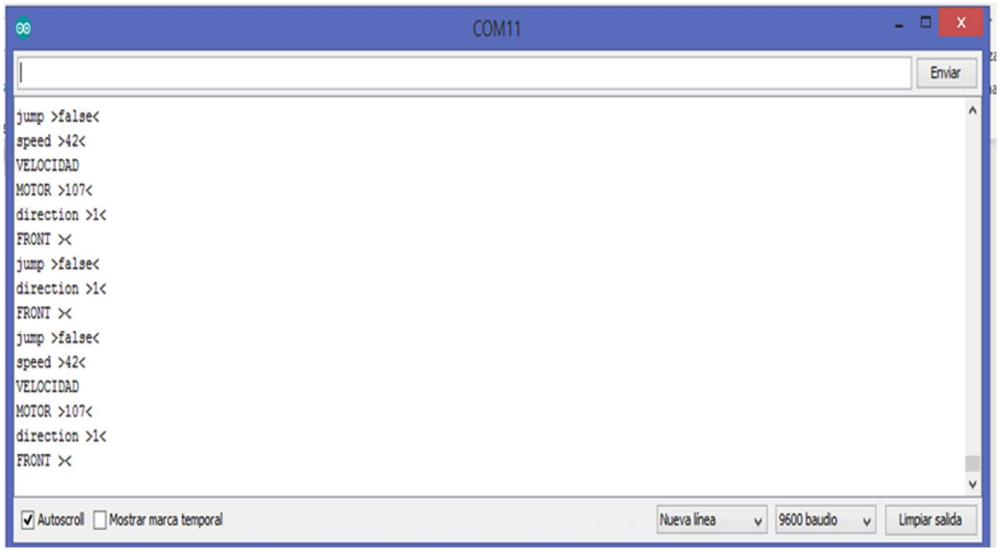


Figura 94: Datos recibidos de velocidad y posición.

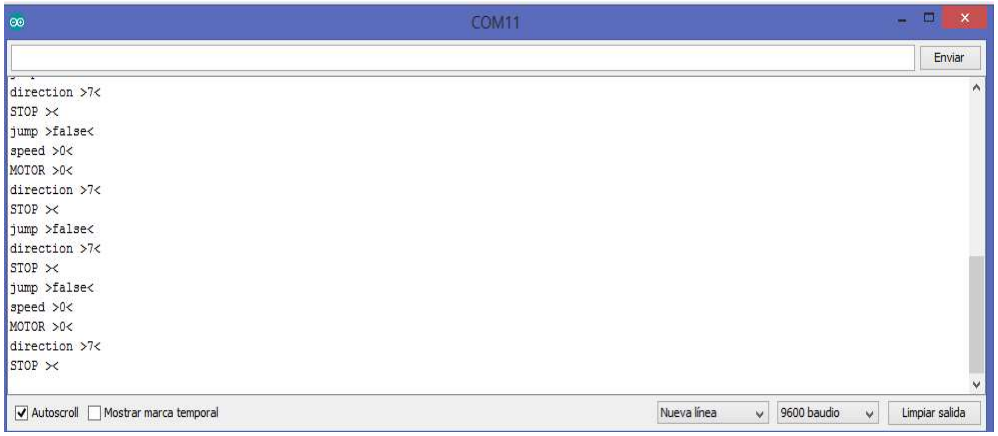


Figura 95: Datos recibidos del puerto serial.

En la figura 96 se observa su **speed** que es 42, indica que se le ha variado la velocidad y además su posición se encuentra en **BACK** como se indica en **dirección >2<.**

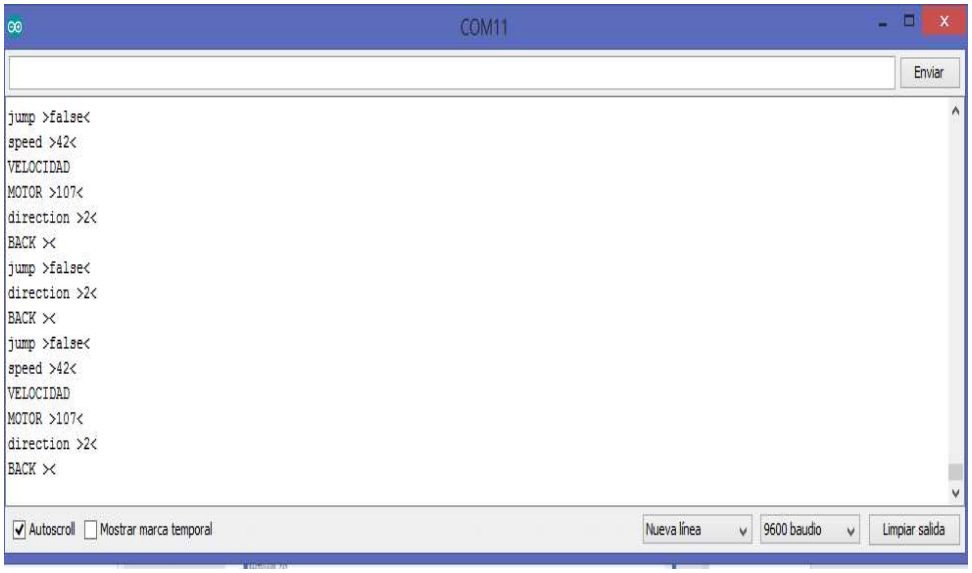


Figura 96: Datos recibidos de dirección.

4 PRÁCTICAS DE LABORATORIO

	PRÁCTICA #1
UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	CONTROL DE LOS MOTORREDUCTORES POLOLU
CARRERA	INGENIERÍA ELECTRÓNICA
SEDE	GUAYAQUIL

OBJETIVO GENERAL

- Visualizar el funcionamiento correcto de los Motores Pololu.

OBJETIVOS ESPECÍFICOS

- Cargar la programación adecuada para el control de los motores.
- Implementar el código fuente al controlar la tarjeta y los variadores de velocidad.
- Configurar parámetros principales como el máximo ciclo de trabajo, la máxima aceleración.

MATERIALES A UTILIZAR

- Pc (portátil)
- Arduino uno (drone)
- Motores pololu

INSTRUCCIONES

- Programar en arduino.
- Conectar el cable usb (entrada c) al arduino, compilar y subir el programa.
- No desconectar y sostener en el aire al drone (no de las llantas).
- Comprobar resultados.

DIAGRAMA

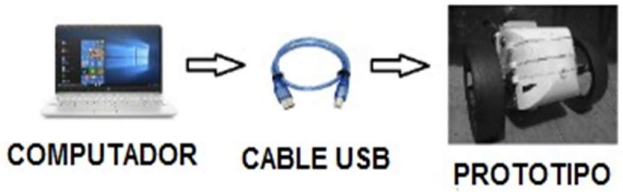


Figura 97: Diagrama de conexiones práctica 1.

	PRÁCTICA #2
UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	CONTROL DE LOS MOTORES MEDIANTE EL MÓDULO RS232
CARRERA	INGENIERÍA ELECTRÓNICA
SEDE	GUAYAQUIL

OBJETIVO GENERAL

- Conocer el protocolo de comunicación rs232, su funcionalidad y el control de las comunicaciones serie entre dos terminales.

OBJETIVOS ESPECÍFICOS

- Proporcionar un servicio de comunicación básico mediante una línea rs232
- Conocer la transmisión y recepción de datos a través de los puertos serie del ordenador.

MATERIALES A UTILIZAR

- Cable usb (entrada c)
- Pc (portátil)
- Arduino uno (drone)
- Módulo rs232

Adaptador usb a rs232 db9

INSTRUCCIONES

- Programar en arduino.
- Conectar el cable usb (entrada c) al arduino, compilar y subir el programa.
- Conectar el módulo rs232 al drone y el adaptador al portátil.
- No desconectar y sostener en el aire al drone.
- Comprobar resultados.

DIAGRAMA

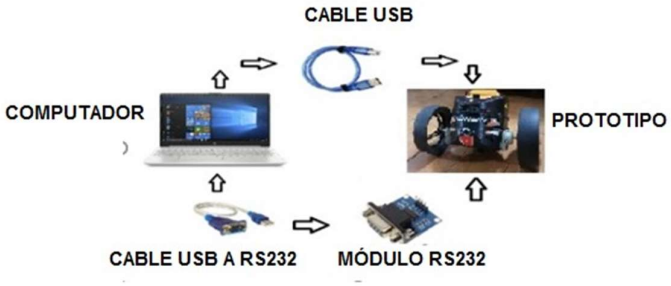


Figura 98: Diagrama de conexiones práctica 2.

	PRÁCTICA #3
UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	OBTENCIÓN Y MUESTRAS DE DATOS DEL SENSOR MPU6050

UNIVERSIDAD	POLITÉCNICA SALESIANA
CARRERA	INGENIERÍA ELECTRÓNICA
SEDE	GUAYAQUIL

OBJETIVO GENERAL

- Instruir en la aplicación de los sensores de medida inercial y realizar algoritmos que permitan la obtención del ángulo de inclinación mediante el uso de un giroscopio y un acelerómetro.

OBJETIVOS ESPECÍFICOS

- Poder visualizar datos de variables mediante el puerto serial del arduino.
- Obtener los datos del módulo mpu6050.

Configurar el arduino para que realice un control con los datos obtenidos del sensor.

MATERIALES A UTILIZAR

- Arduino uno (drone)
- Módulo mpu6050
    - Pc portátil

INSTRUCCIONES

- Programar en arduino
- Realizar la conexión vía cable usb (entrada c), entre el portátil y el drone.
- Compilar y ejecutar en el software arduino la programación la práctica a realizar
- Verificar resultados en el puerto serial los datos obtenidos con el sensor.

DIAGRAMA

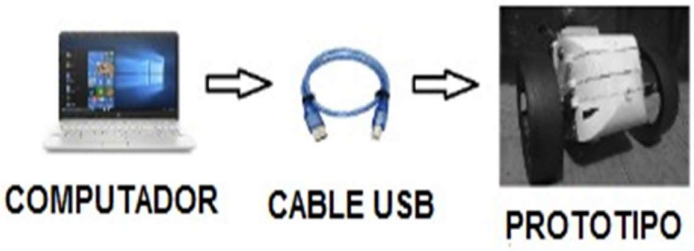


Figura 99: Diagrama de conexiones práctica 3.

	PRÁCTICA #4
UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	ACCIÓN DE LOS MOTORES MEDIANTE EL USO DEL SENSOR MPU6050

UNIVERSIDAD	POLITÉCNICA SALESIANA
CARRERA	INGENIERÍA ELECTRÓNICA
SEDE	GUAYAQUIL

OBJETIVO GENERAL

- Visualizar el accionar de los motores pololu aplicando un algoritmo que permitan el uso de un giroscopio y acelerómetro.

OBJETIVOS ESPECÍFICOS

- Realizar la programación en arduino del control de los motores con los datos obtenidos del sensor.
- Cargar el programa efectuado al arduino.
- Aplicar movimiento para poder visualizar la función correspondiente.

MATERIALES A UTILIZAR

- Módulo mpu6050
- Pc (portátil)
- Software arduino
- Cable usb (entrada c)

INSTRUCCIONES

- Programar en arduino
- Realizar la conexión vía cable usb (entrada c), entre el portátil y el drone.
- Compilar y ejecutar en el software arduino la programación la práctica a realizar.
- Verificar resultados en el puerto serial los datos obtenidos con el sensor.

DIAGRAMA

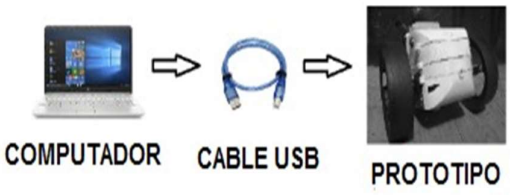


Figura 100: Diagrama de conexiones práctica 4.

	PRÁCTICA #5
UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	DETECTOR DE OBJETOS MEDIANTE EL SENSOR SHARP CON ALARMA SONORA

UNIVERSIDAD	POLITÉCNICA SALESIANA
CARRERA	INGENIERÍA ELECTRÓNICA
SEDE	GUAYAQUIL

OBJETIVO GENERAL

- Detector de objetos a corta distancia con alarma sonora.

OBJETIVOS ESPECÍFICOS

- Indagar sobre las características técnicas del sensor sharp.

Establecer comunicación con el software arduino mediante el sensor de distancia (sharp) y el buzzer.

MATERIALES A UTILIZAR

- Arduino uno (drone)
- Sensor sharp
- Buzzer
    - Pc portátil

INSTRUCCIONES

- Programar en arduino.
- Conectar el cable usb (entrada c) al arduino, compilar y subir el programa.
- Conectar el sharp y el buzzer.
- Colocar el drone en el piso y encenderlo.
- Comprobar resultados.

DIAGRAMA

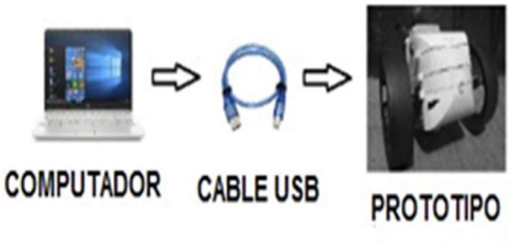


Figura 101: Diagrama de conexiones práctica 5.

5 RESULTADOS

En esta sección se muestra los resultados obtenidos en cada una de las prácticas que ayudaron a cumplir los objetivos planteados inicialmente.

5.1 Análisis de resultados

5.1.1 Práctica 1

En esta práctica se tiene como objetivo el control de los motores reductores pololu, en cuanto al resultado, es la visualización del correcto funcionamiento de los motores pololu mediante el puerto serial.

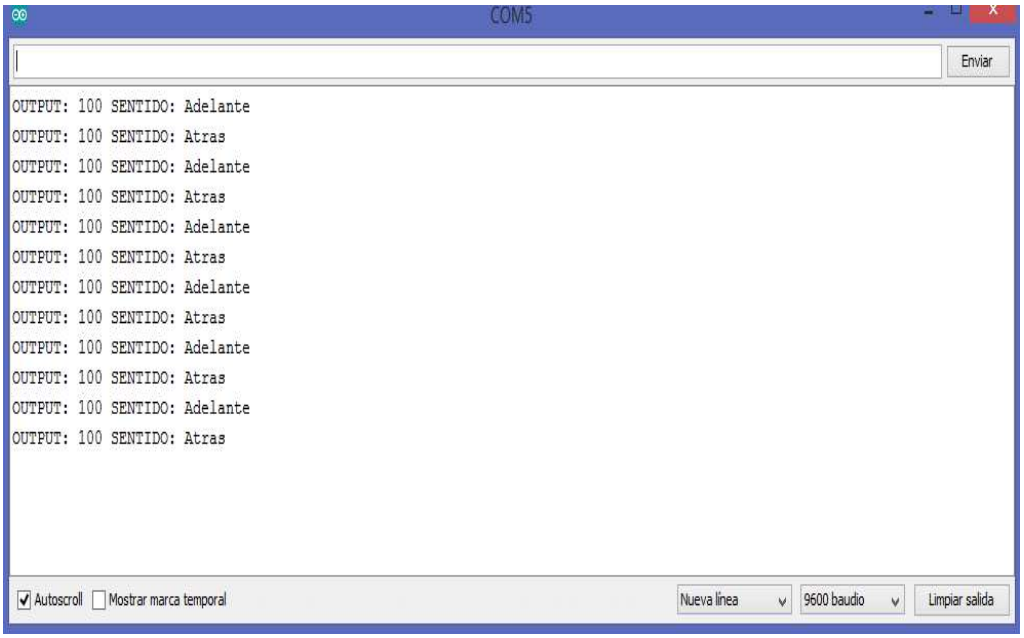


Figura 102: Demostración del control de los motores reductores pololu.

Figura 103: Demostración real de la práctica 1.

5.1.2 Práctica 2

Como tema de esta práctica se tiene el control de motores mediante el módulo rs232 y como resultado de esta práctica es tener el control de los motores reductores pololu mediante la comunicación asíncrona con el módulo rs232.

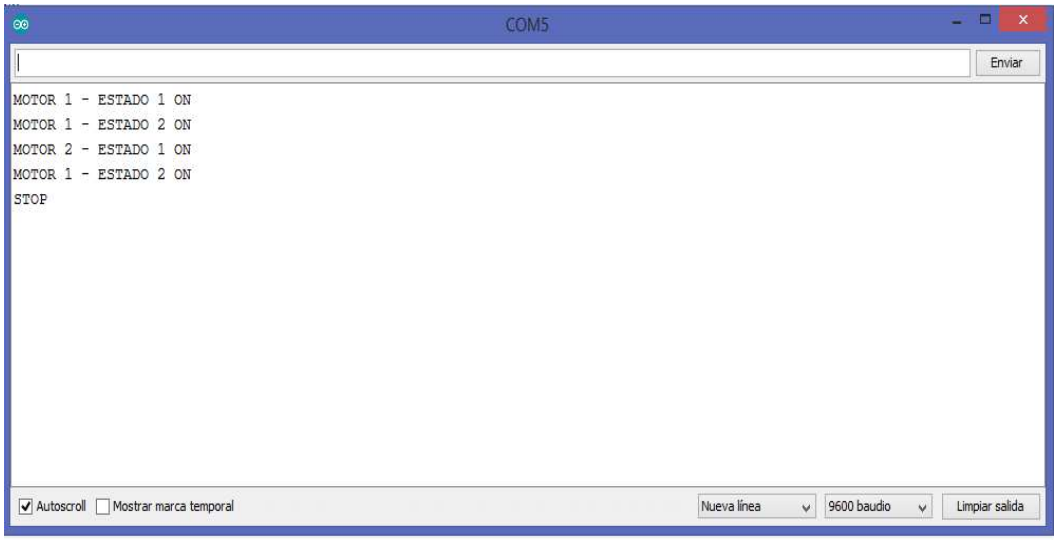


Figura 104: Demostración de manipulación de módulo asíncrono rs232.

Figura 105: Muestra real de la práctica 2.

5.1.3 Práctica 3

En esta práctica se tiene como título obtención y muestras de datos del sensor mpu6050 y para verificar el funcionamiento del sensor se logra la obtener las muestras de datos mediante el sensor acelerómetro y giroscopio según su posición.

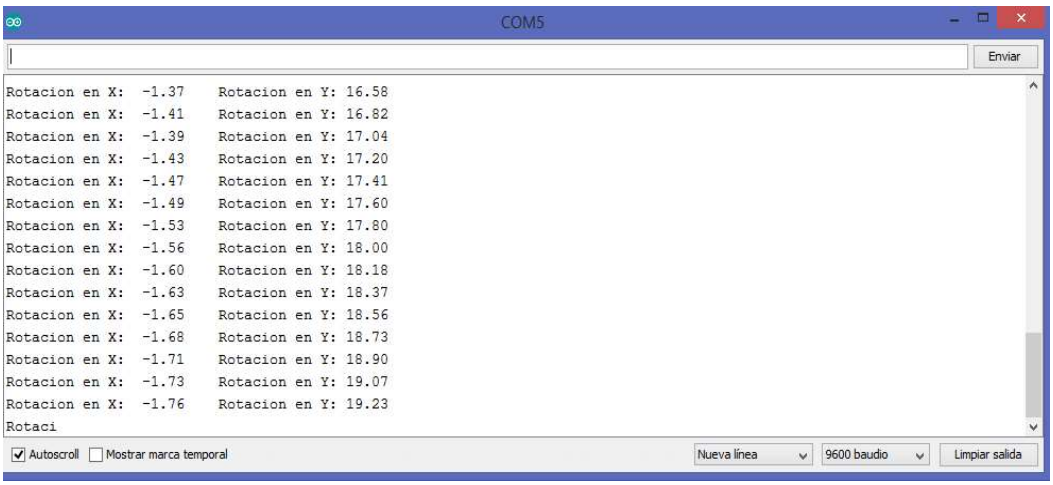


Figura 106: Demostración de obtención de datos del sensor mpu-6050.

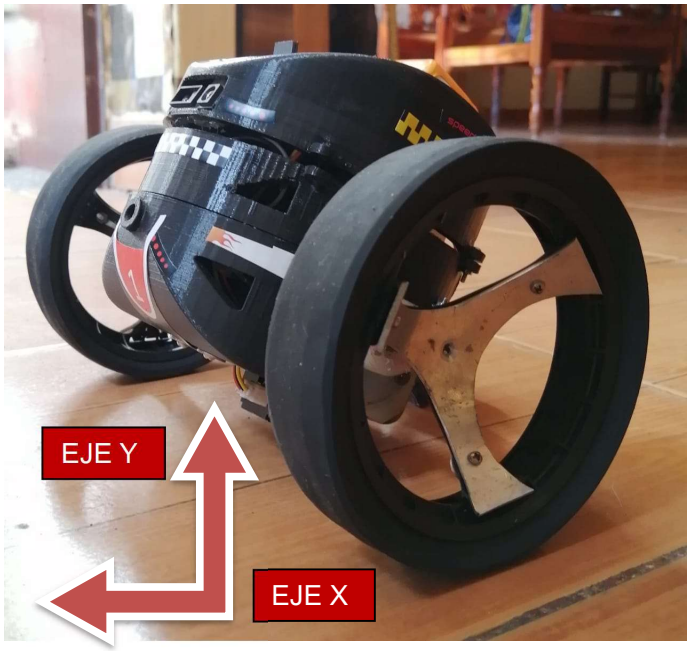


Figura 107: Verificación de obtención de datos de la práctica 3.

5.1.4 Práctica 4

En esta práctica como tema se tiene la acción de los motores mediante el uso del sensor mpu6050 en donde como resultados se realiza el control de la acción de los motores reductores pololu según la dirección del robot mediante el sensor mpu6050.

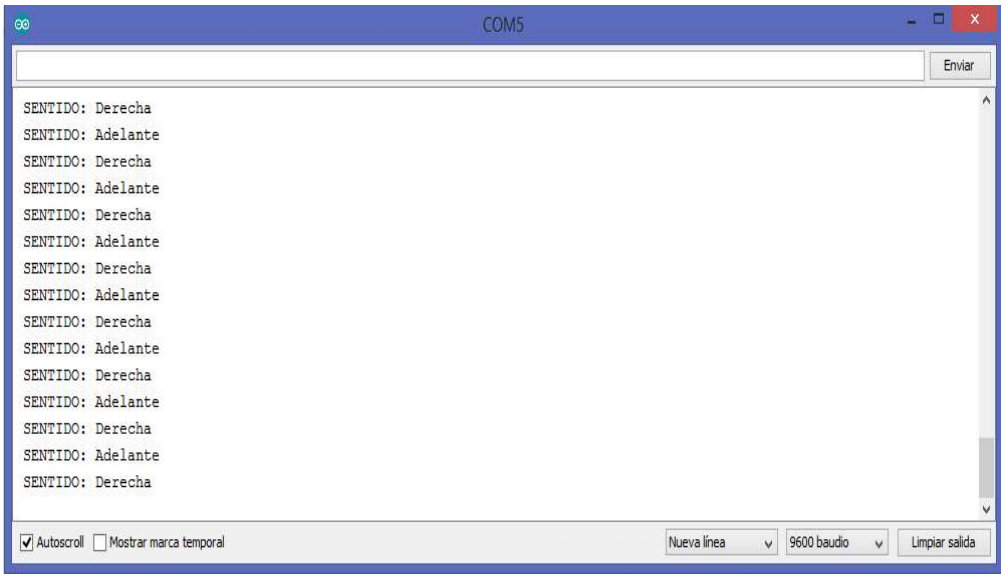


Figura 108: Control de acción de los motores mediante el sensor mpu6050.

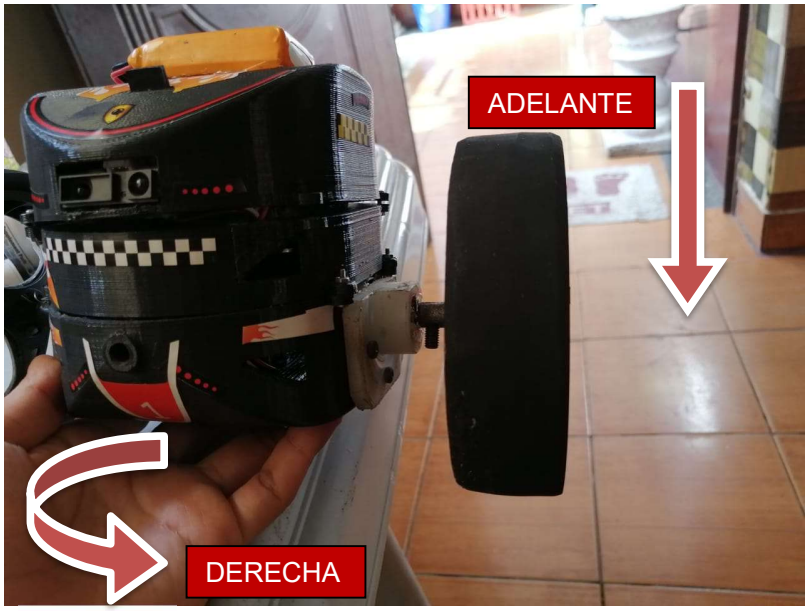


Figura 109: Demostración real práctica 4.

5.1.5 Práctica 5

En esta práctica se tiene como objetivo detector de objetos mediante el sensor sharp con alarma sonora, para comprobar y verificar que los resultados sean los adecuados se asignan parámetros con respecto a la distancia para que asi detecte los objetos mediante el sensor sharp con la alarma sonora.

Figura 110: Demostración del detector de distancia mediante el sensor sharp.

Figura 111: Verificación de distancia de la práctica 5.

CONCLUSIONES

Como conclusiones se establece los siguientes puntos:

- Se diseñó una aplicación personal en la plataforma android studio y se la enlazó con arduino, mediante el módulo esp32 con la finalidad de controlar el dispositivo.

- Se logró establecer una nueva alternativa a nivel de comunicación de dispositivos y/o equipos mediante la tecnología arduino, usando la red del módulo esp32 existente como medio de transmisión de datos.

- Para realizar el streaming de video se utilizó el módulo esp32-cam donde se le asigna una dirección IP respectiva para la conexión entre el prototipo y la aplicación android para mostrarlo en tiempo real mediante la aplicación y tener un adecuado funcionamiento.

- El proceso de colocación del módulo electrónico esp32, se realizó con cuidado para evitar daños en el prototipo, teniendo en cuenta las direcciones de los equipos sean las correctas para que sea reconocido en la red.

- Hay que tener en cuenta que todos los equipos utilizados para el desarrollo del proyecto ya que son de suma importancia para el funcionamiento del sistema.

- Con el uso del software arduino se pudo verificar mediante el puerto serial el

correcto funcionamiento de la programación, tales como el control pid, batería, comunicación de datos entre arduino uno y módulo esp32.

- El proceso de diseño de la estructura fue lo más laborioso ya que debía verificarse qué tipo de material poder utilizar, el espacio interior para las tarjetas electrónicas, peso y diseño.

- Los controladores para el control PID tuvieron que ser de un rango de 0 a 1 para que se pueda estabilizar en un tiempo menor a 3 seg.

- El mpu6050 se lo coloca de manera horizontal para obtener el correcto valor de inclinación del eje que arroja el mpu6050 y se verifica mediante un graduador donde el ángulo es de 0 a 90 grados explicando con una regla de tres dentro de la programación para poder visualizar dicho ángulo.

RECOMENDACIONES

- Desarrollar una placa electrónica la cual pueda incorporar todos los sensores junto con el módulo esp32.

- Mejorar el streaming de video con el módulo esp32-cam para obtener una mejor resolución de cámara, teniendo compatibilidad con arduino uno.

- Diseñar un filtro para tener una mejor obtención de datos del sensor mpu6050.

- Obtener unos mejores controladores para el control pid de la velocidad obteniendo más muestras de datos y a su vez implementar un mejor algoritmo para tener una respuesta más rápida.

- Investigar algún material más resistente e innovar un nuevo diseño para la estructura del robot el cual pueda resistir los saltos y que no tienda a deformarse al momento de juntar con perno y tuerca las diferentes capas que conforman el drone terrestre.

- Crear un mecanismo el cual pueda lograr una mayor fuerza de salto y así mejorar la altura entre el suelo y el prototipo.

- Mejorar la comunicación entre la aplicación android y el prototipo para corregir la latencia de datos (tiempo de retardo de 2 seg) al momento de pulsar el botón en la aplicación y realiza la orden el robot.

BIBLIOGRAFÍA

[1] A. García Moisés y S. Tainta Ausejo, «E.T.S de Ingeniería Industrial, Informática y de Telecomunicación,» de *Diseño y construcción de un robot auto-balanceado mediante Arduino*, Pamplona, 2017, pp. 20-23.

[2] «Espressif,» Sistemas Espressif, 2018. [En línea]. Available: https://www.espressif.com/en/products/hardware/esp32/overview. [Último acceso: 10 Marzo 2020].

[3] «EcuRed,» 12 Marzo 2017. [En línea]. Available: https://www.ecured.cu/Bater%C3%ADas_de_Li-Po. [Último acceso: 19 Febrero 2019].

[4] «Ramdom Nerd Tutorials,» 22 Junio 2016. [En línea]. Available: https://randomnerdtutorials.com/esp32-cam-video-streaming-face-recognition-arduino-ide/. [Último acceso: 02 Enero 2020].

[5] «Pololu Robotics & Electronics,» Pololu Corporation, 2011. [En línea]. Available: https://www.pololu.com/product/3063. [Último acceso: 09 Febrero 2020].

[6] «Pololu Robotics & Electronics,» Pololu Corporation, 2011. [En línea]. Available: https://www.pololu.com/product/2590. [Último acceso: 10 Febrero 2020].

[7] J. Etxeberria Mendez y J. Goicoechea Fernández, «E.T.S de Ingeniería Industrial, Informática y de Telecomunicación,» de *Implementación de un dron cuadricóptero con Arduino*, Pamplona, 2015, pp. 12-18.

[8] J. M. Quero Reboul, «Evaluación de Giroscopo,» de *Evaluación de un Giróscopo MEMS en un Péndulo*, Sevilla, 2011, p. 9.

[9] «Omega,» 15 Mayo 2015. [En línea]. Available:

https://mx.omega.com/prodinfo/acelerometro.html. [Último acceso: 10 Marzo 2019].

[10] «Infootec.Net,» 14 Octubre 2017. [En línea]. Available: https://www.infootec.net/arduino/. [Último acceso: 18 Marzo 2019].

[11] «Naylamp Mechatronics,» 24 Julio 2015. [En línea]. Available: https://naylampmechatronics.com/blog/55_tutorial-sensor-de-distancia-sharp.html. [Último acceso: 19 Marzo 2019].

[12] «Eneka Líder en Electrónica,» 12 Julio 2014. [En línea]. Available: http://www.eneka.com.uy/robotica/modulos-comunicacion/m%C3%B3dulo-conversor-rs232-a-ttl-max3232-detail.html. [Último acceso: 24 Marzo 2019].

[13] L. Llamas, «Ingeniría, informática y diseño,» 14 Abril 2014. [En línea]. Available: https://www.luisllamas.es/arduino-puerto-serie/. [Último acceso: 06 Mayo 2019].

[14] L. Llamas, «Ingeniría, informática y diseño,» 16 Julio 2016. [En línea]. Available: https://www.luisllamas.es/reproducir-sonidos-arduino-buzzer-pasivo-altavoz/. [Último acceso: 22 Marzo 2019].

[15] E. H. Asis López, «Universidad Nacional De Educación Enrique Guzaman y Valle,» 2015. [En línea]. Available: http://repositorio.une.edu.pe/bitstream/handle/UNE/962/TM%20CE-Du%20A814%202015.pdf?sequence=1&isAllowed=y. [Último acceso: 29 03 2020].

[16] Parrot, «My Parrot,» 25 06 2010. [En línea]. Available: https://www.parrot.com/es/minidrones/parrot-jumping-race-max. [Último acceso: 28 04 2020].

ANEXOS

Anexo 1: Desarrollo de las prácticas

PRÁCTICA 1

Paso 1

En este paso se define las variables que van a controlar la dirección y velocidad de los dos motores, así como se muestra en la figura 1.

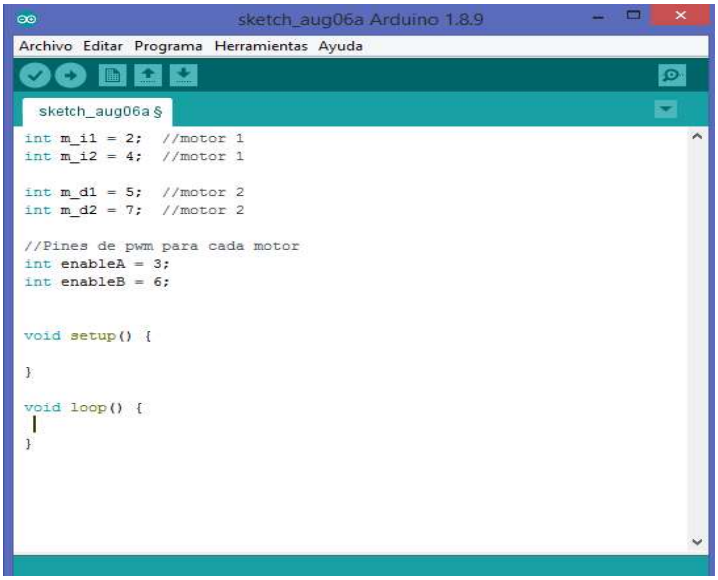


Figura 112: Declaración de variables práctica 1.

Paso 2

Ahora en el void setup (), se inicializa la comunicación por el puerto serial (PuertoCOM) y los pines para controlar el motor se los declara como puertos de salida.

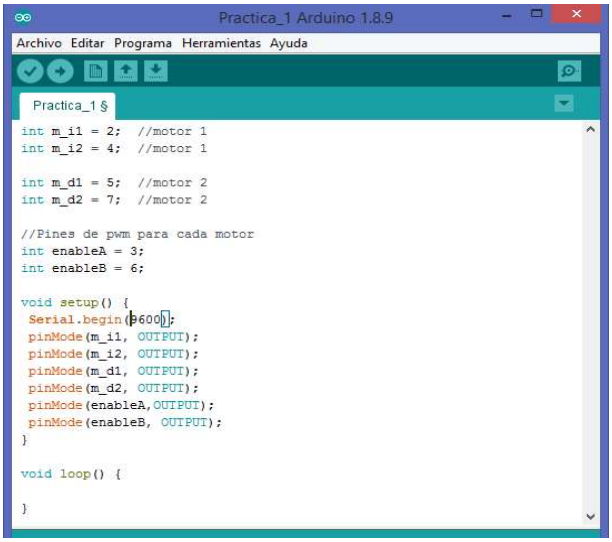


Figura 113: Comunicación puerto serial práctica 1.

Paso 3

En el void loop se configura los pines para que el motor gire en un sentido al inicio, y luego mediante una pausa de medio segundo, cambia el sentido mandando los pines que estaban en alto, ahora que estén en bajo y viceversa. Todo esto con unos pulsos de pwm de 150 (Los pulsos van de 0 a 255).

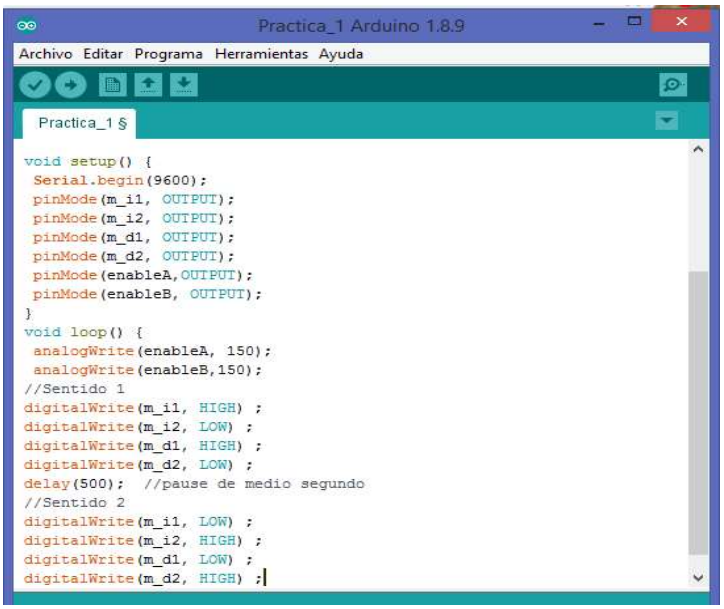


Figura 114: Configuración de pines práctica 1.

Paso 4

Se muestra la lectura de datos por el puerto serial

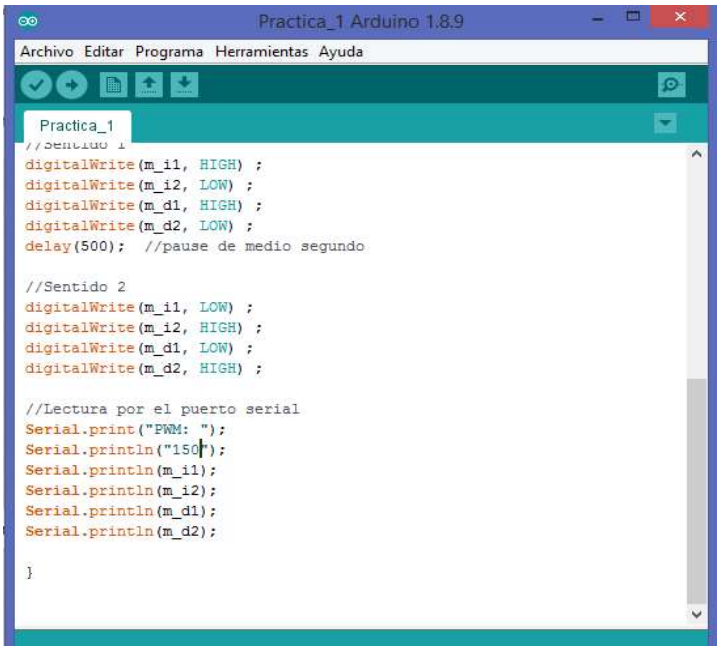


Figura 115: Lectura de los puertos seriales práctica 1.

PRÁCTICA 2

Paso 1

Abrir el software arduino para realizar la programación. Se empieza declarando las variables, especificando los puertos digitales en el arduino de cada uno de los motores.

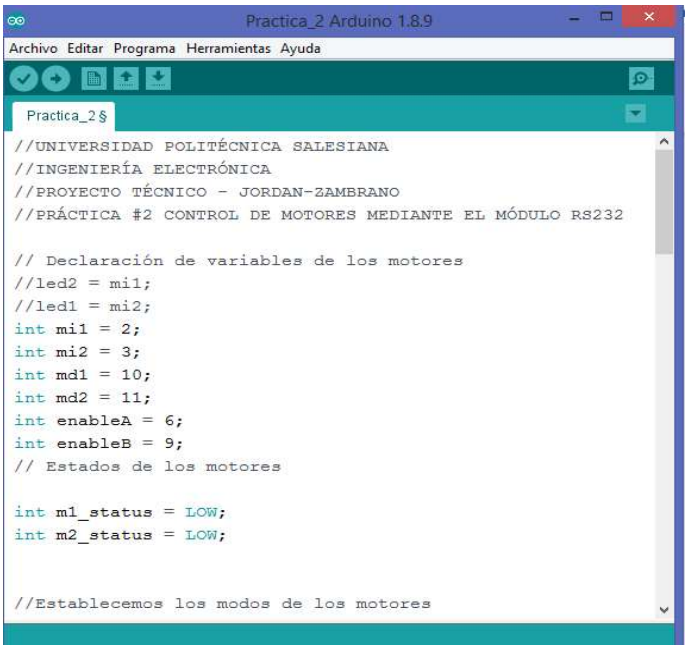


Figura 116: Declaración de variable práctica 2

Paso 2

Configurar los estados de los motores para que se inicialicen apagados en este caso tienen dos tiempos cada motor hacia adelante y hacia atrás.

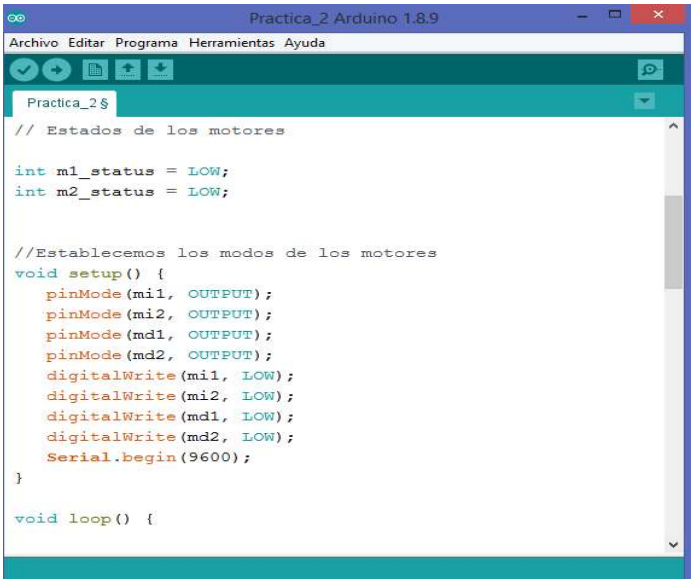


Figura 117: Establecer estados de motores práctica 2.

Paso 3

Definir las revoluciones por minuto de los motores y la condición correspondiente al motor 1 (izquierdo) cuando se ingresa en el serial “1” este va a girar hacia adelante caso contrario si ingresa en el serial “2” este va a girar hacia atrás.

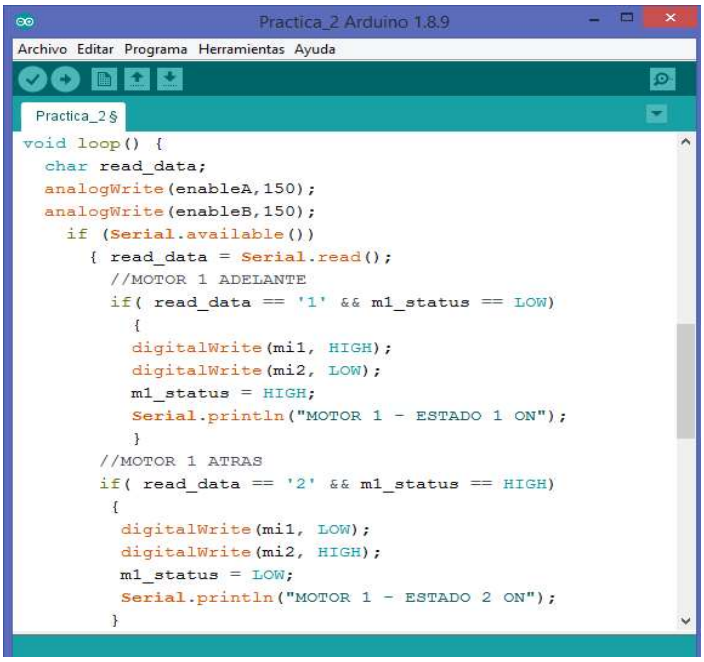


Figura 118: Definición de estado del motor izquierdo práctica 2.

Paso 4

Realizar

el mismo procedimiento ahora con respecto al motor 2 (derecho) cuando ingresa en el serial “3” este va a girar hacia adelante caso contrario si se ingresa en el serial “4” este va a girar hacia atrás.

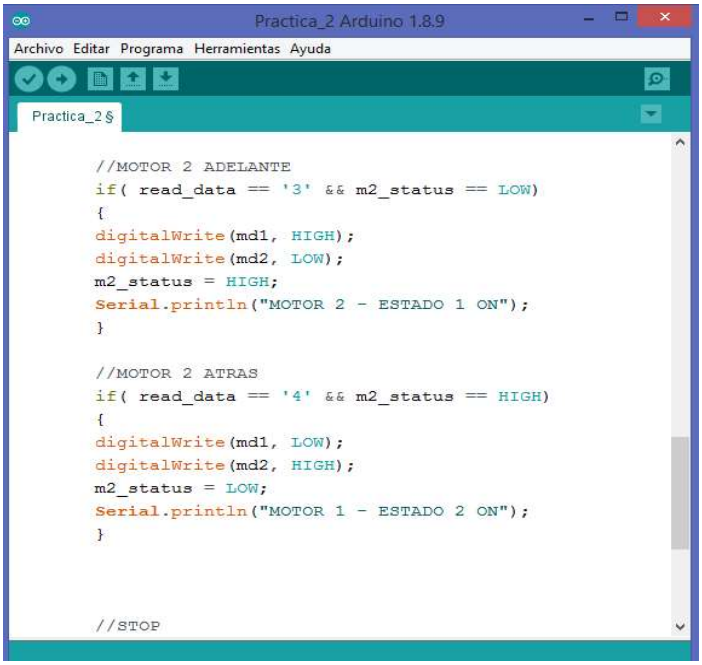


Figura 119: Definición del estado del motor derecho.

Paso 5

Para realizar el paro de los dos motores digitar en el serial “0” y estos proceden a detener su marcha.

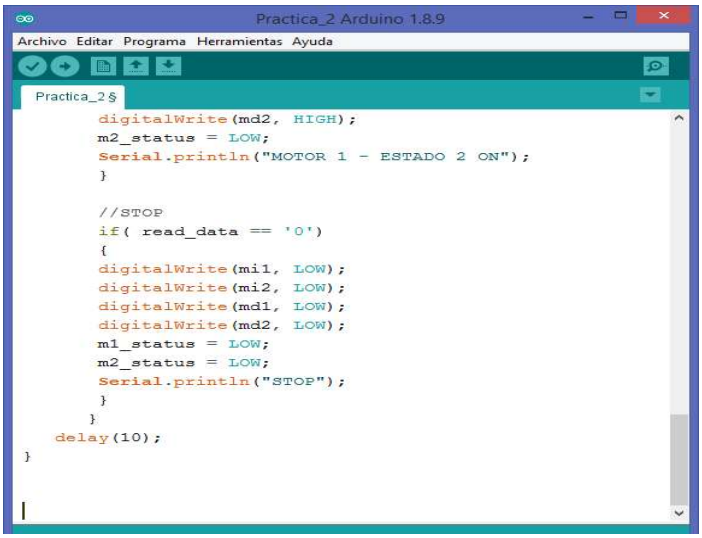


Figura 120: Detención o apagado de motores práctica 2.

Paso 6

Conectar el cable usb (entrada c) entre la portátil y el arduino, luego dirigirse a herramientas>> puerto>>com (arduino uno) se procede a compilar y cargar el programa.

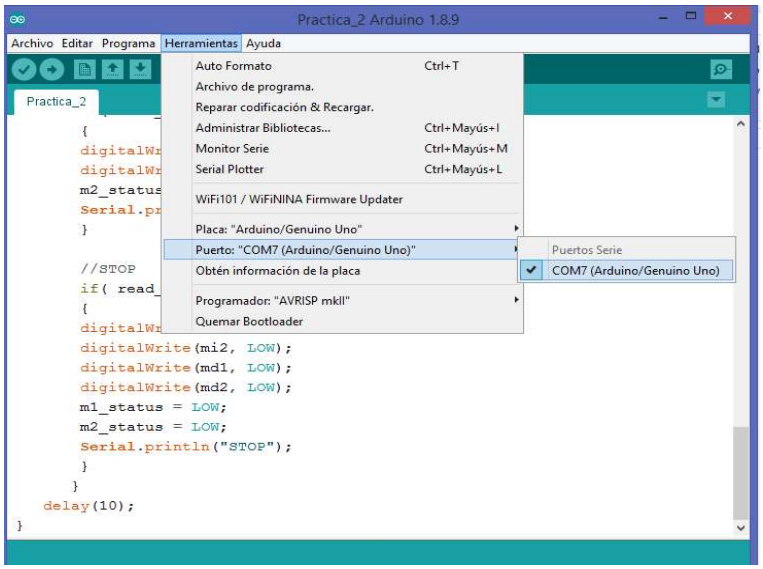


Figura 121: Seleccionar el puerto serial del arduino práctica 2.

Paso 7

Realizar la conexión del módulo rs232 al drone y a su vez el adaptador (usb a rs232) al portátil.

Paso 8

Verificar en panel de control>>hardware y dispositivos>>administrador de dispositivos que se encuentren visibles los puertos tanto del arduino como el del rs232.

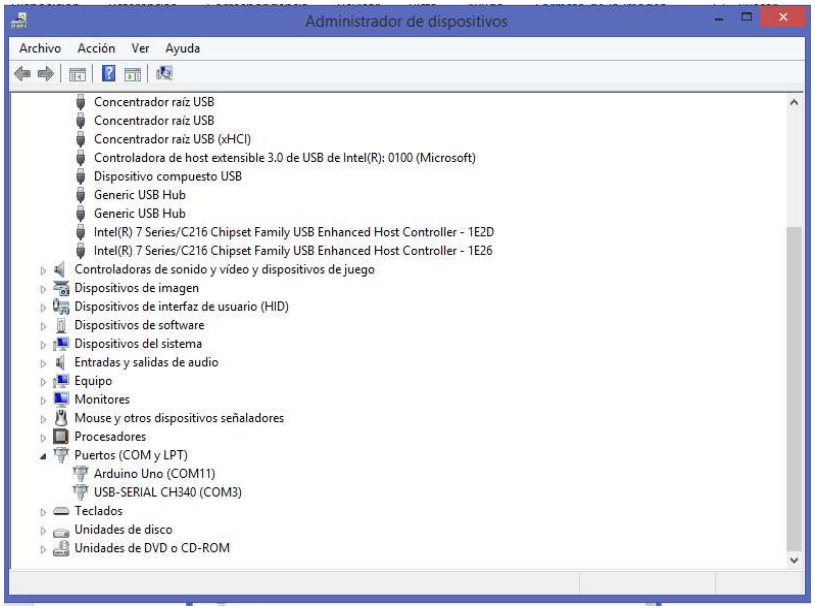


Figura 122: Verificar los puertos seriales práctica 2.

Paso 9

Dentro del software arduino seleccionar en herramientas>>puerto (serial) del rs232.

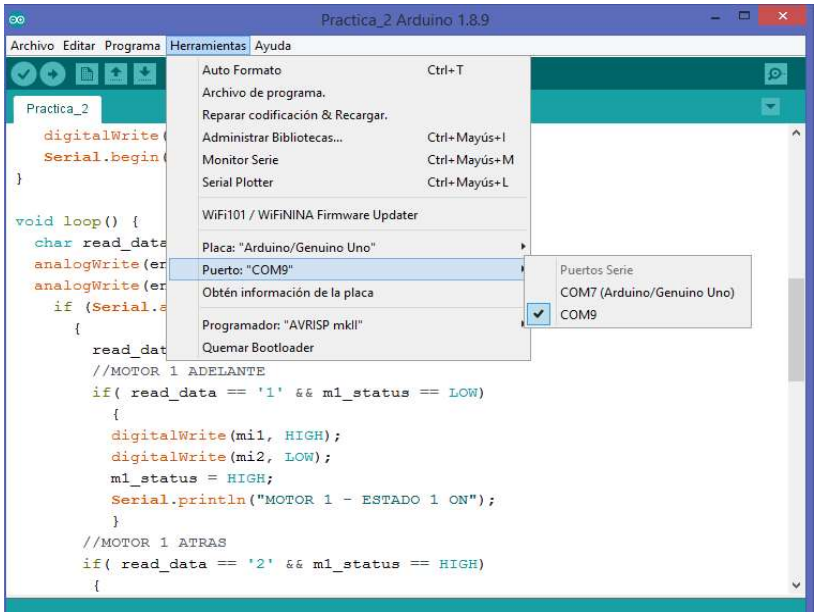


Figura 123: Seleccionar el puerto serial del rs232 práctica 2.

Paso 10

Luego en el botón monitor serie, digitar la función para que ejecuten los motores del robot mediante las variables ya declaradas. Comprobar que existe comunicación entre el rs232 y arduino.

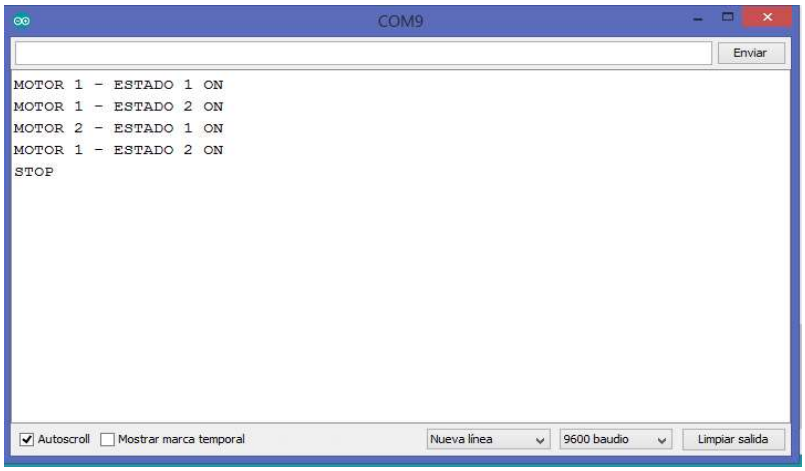


Figura 124: *Visualización del funcionamiento en el puerto serial práctica* 2.

PRÁCTICA 3

Paso 1

Se empieza incluyendo las librerías necesarias para poder realizar la comunicación con el sensor mpu6050 y también inicializar las variables, como muestra en la figura 118.

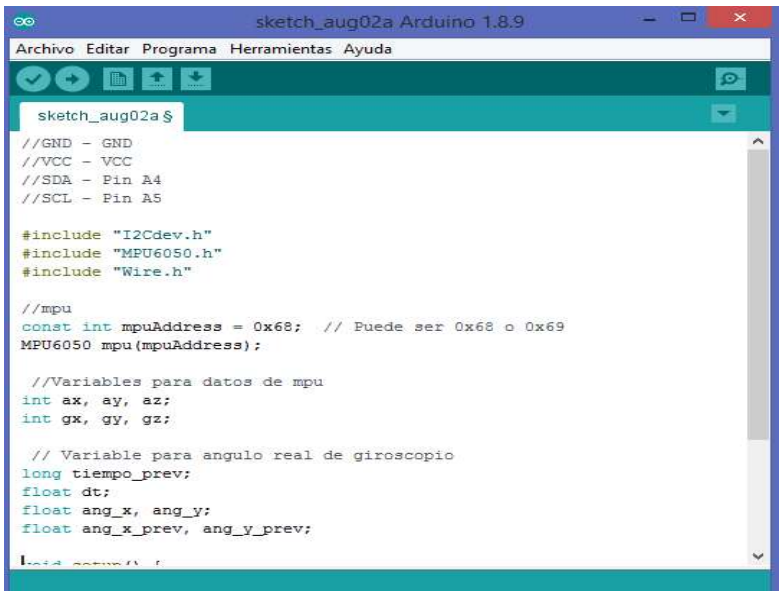


Figura 125: Declaración de librerías práctica 3.

Paso 2

En el void setup() se centra en inicializar la comunicación con el mpu6050 y también habilitar el puerto serial para poder observar los datos obtenidos de la codificación de la figura 119.

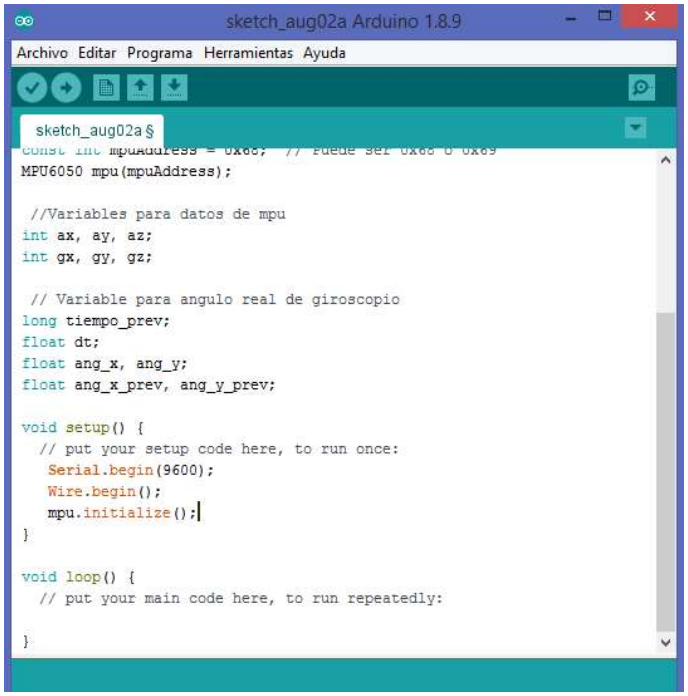


Figura 126: Iniciar comunicación con mpu6050 práctica 3.

Paso 3

Tal cual como se muestra en la figura 4, se debe empezar obteniendo los datos de los ejes x, y y z y guardándolos en sus respectivas variables para después aplicar la fórmula que convertirá los datos obtenidos en ángulos en x y en y.

Para poder obtener el mayor número de puntos de posición del sensor, se aplica un algoritmo para adquirir cada 10 milisegundos la posición.

$$\theta_x = \tan^{-1}\left(\frac{a_x}{\sqrt[2]{{a_Y}^2 + {a_z}^2}}\right)$$

$$\theta_y = \tan^{-1}\left(\frac{a_y}{\sqrt[2]{{a_x}^2 + {a_z}^2}}\right)$$

Figura 127: Fórmula de ángulos práctica 3.

Paso 4

En las variables accel_ang_x y accel_ang_y se guardan los datos de la aplicación de la formula.

Luego se aplica un filtro para tener mayor precisión en la adquisición de datos y se guarda en las variables ang_x y ang_y.

Para finalizar se guardan las variables ang_x y ang_y en otras variables tipo float las cuales son crucial para el desarrollo correcto del filtro.

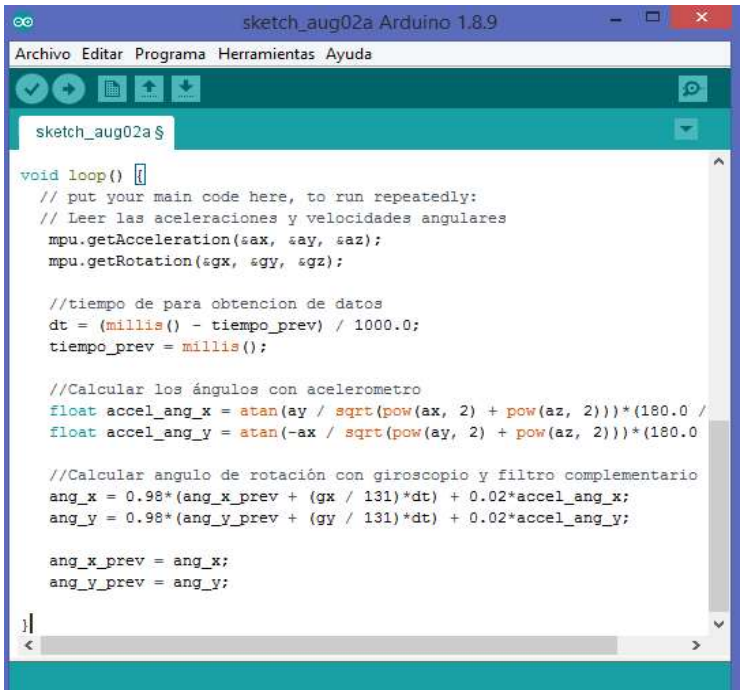


Figura 128: Obtención y filtrado de datos práctica 3.

Paso 5

Se procede a imprimir por el puerto serial el mensaje de rotación y su valor correspondiente.

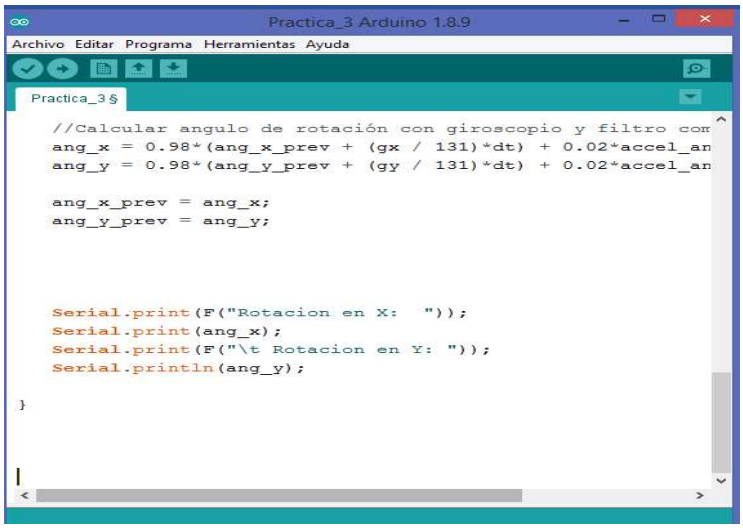


Figura 129: Mostración de los ejes con sus valores práctica 3.

Paso 6

Muestreo de datos mediante el puerto serial

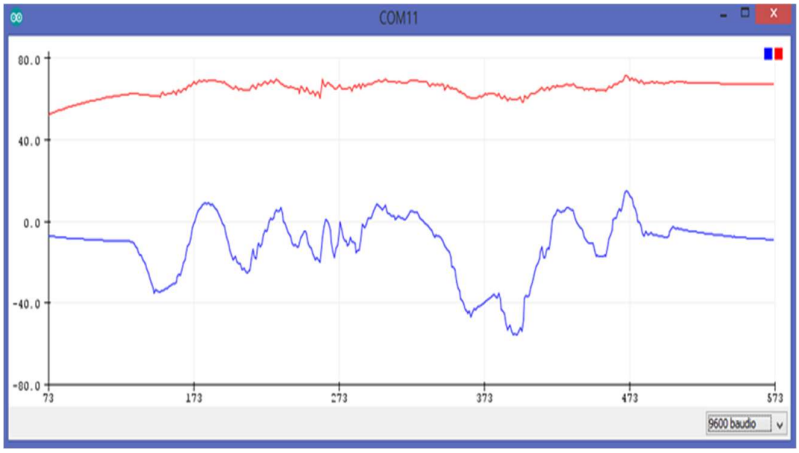


Figura 130: Muestra de datos en el puerto serial practica 3.

PRÁCTICA 4

Paso 1
Incluir todas las librerías con la que se ejecuta el sensor acelerómetro y giroscopio mpu6050 y a su vez declarar las variables de los motores para la práctica.

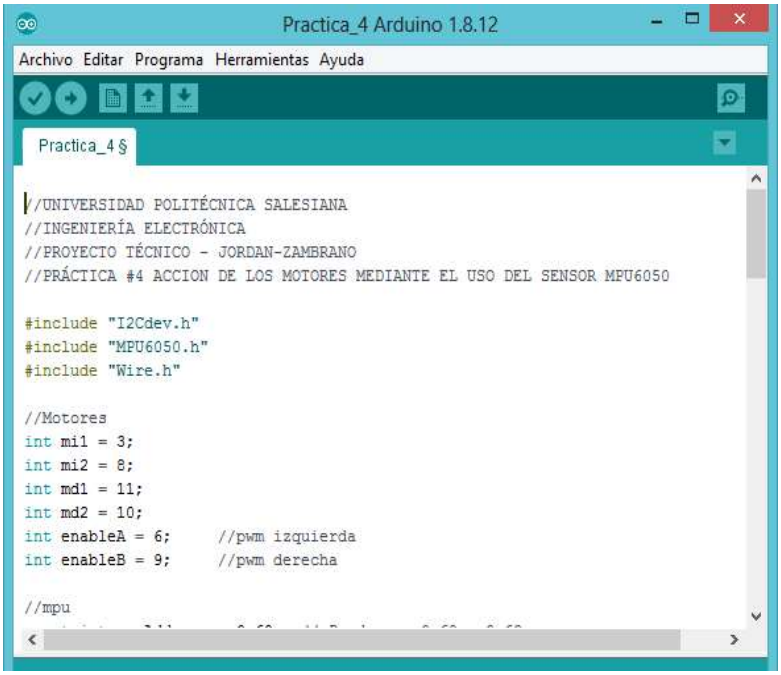


Figura 131: Declaración de variables práctica 4.

Paso 2
Especificar el direccionamiento del sensor el cual puede ser 0x68 o 0x69 y realizar la declaración de las variables para los datos del mpu (aceleración y giroscopio).

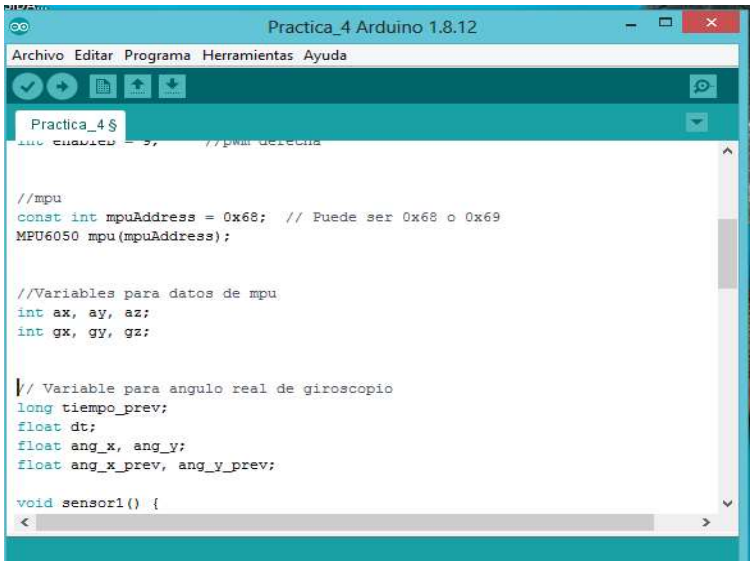


Figura 132: Especificación de direccionamiento práctica 4.

Paso 3
Lectura de las aceleraciones y velocidades angulares, para poder realizar la práctica.

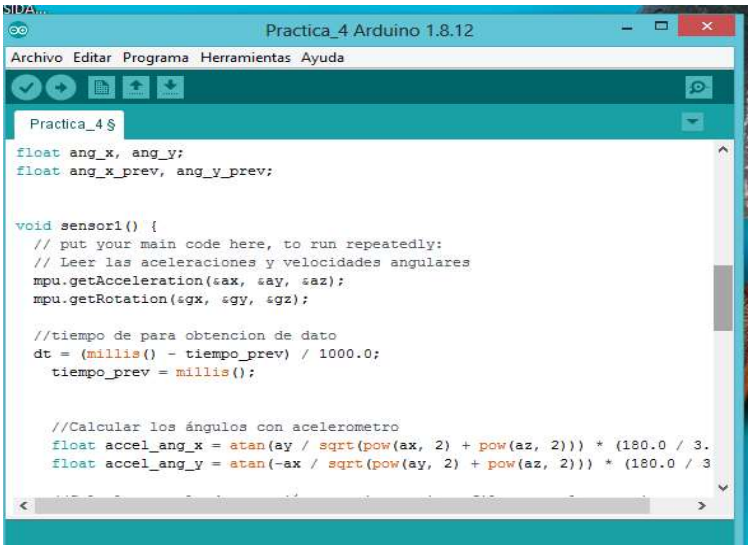


Figura 133: Lectura de variables del mpu6050 práctica 4.

Paso 4
En este paso muestra la fórmula para poder realizar el cálculo de ángulos con acelerómetro y con giroscopio, expuesto a un filtro complementario.

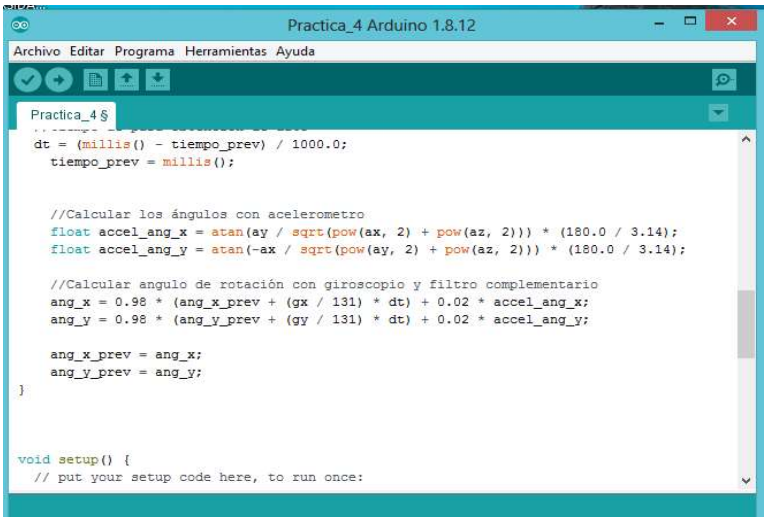


Figura 134: Fórmula para calcular ángulos práctica 4.

Paso 5
Se establece la velocidad y se inicializa el mpu con sus motores en bajo.

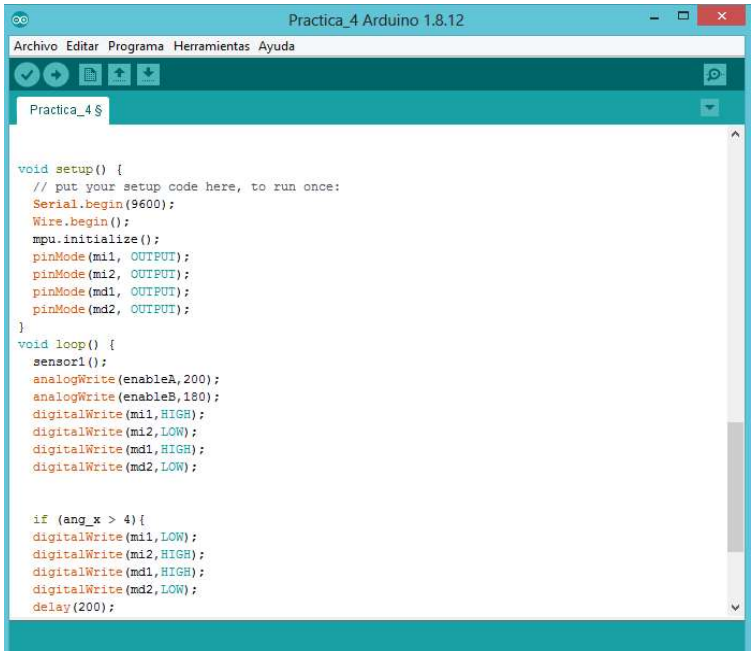


Figura 135: Inicialización del mpu6050 práctica 4.

Paso 6
Se realiza la condición, el drone terrestre va hacia adelante si alguien lo desvía de su trayectoria hacia la izquierda, él gira hacia la derecha a su posición inicial caso

contrario si alguien desvía de su trayectoria hacia la derecha, él gira hacia la izquierda a su posición inicial.

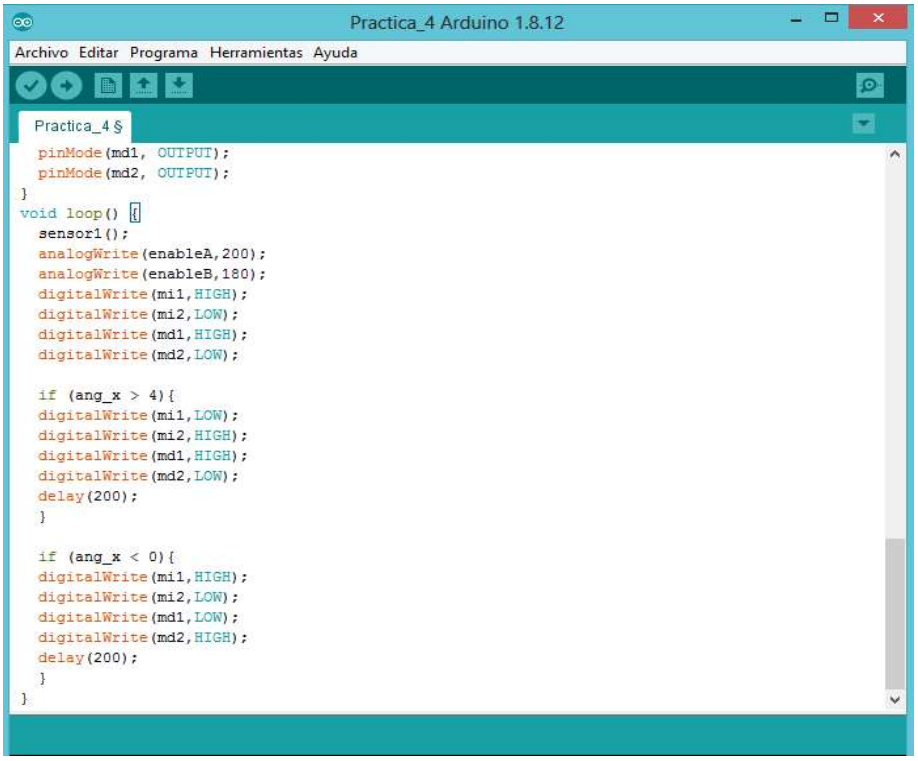


Figura 136: Control de movimiento del drone terrestre práctica 4.

PRÁCTICA 5

Paso 1

Declarar las variables del sharp y buzzer con la respectiva entrada analógica y pin.

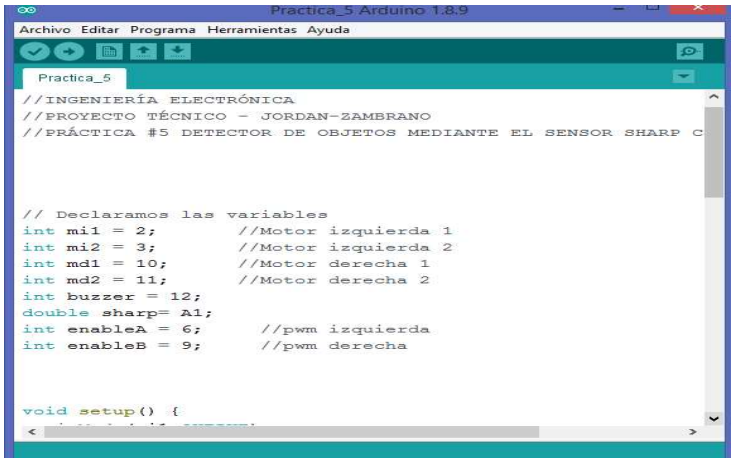


Figura 137: Declaración de variables práctica 5.

Paso 2

En el void setup() se establece la comunicación serial mediante la tasa de baudios que se refiere al número de unidades de la señal por segundo y también especificar el pin digital de salida.

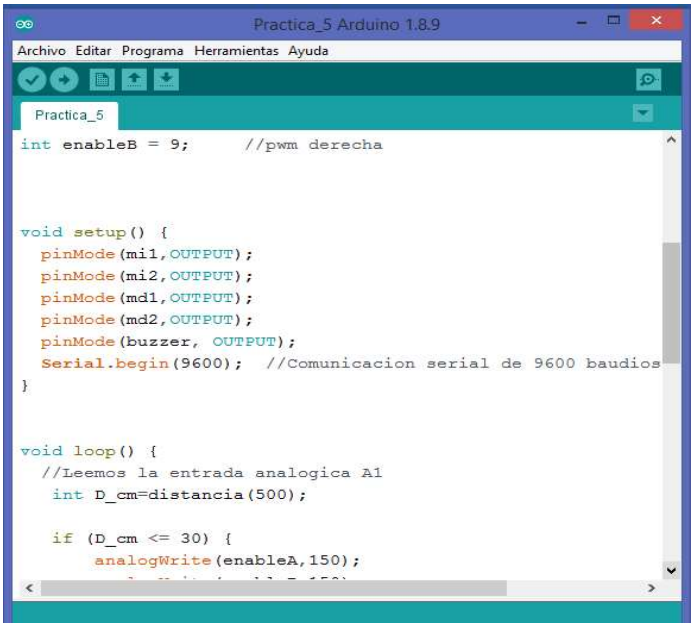


Figura 138: Establecer la comunicación serial práctica 5.

Paso 3

Establecer una condición en el void loop() para ampliar o disminuir el umbral de detección el cual si se activa el pin digital aparece un mensaje “Objeto Detectado” y el buzzer sonará caso contrario; si se apaga el pin digital aparece un mensaje “Objeto Ausente”. Y luego se especifica el tiempo de espera.

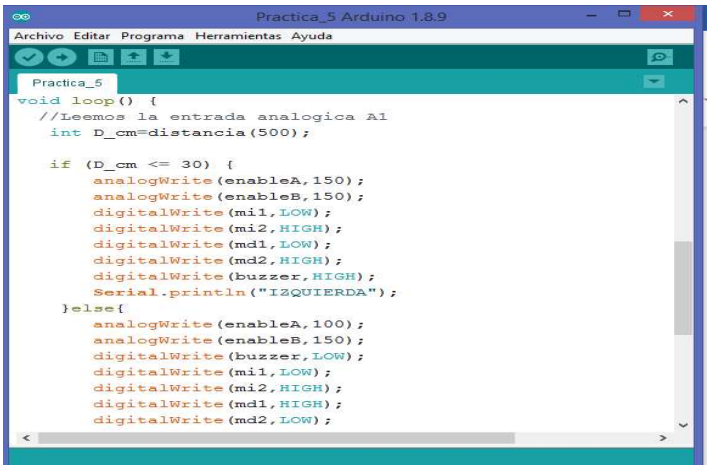


Figura 139: Se establece el umbral de detección práctica 5.

Paso 4

Se realiza la validación de la lectura del adc, donde se compara el valor del umbral de detección para determinar la distancia de los objetos.

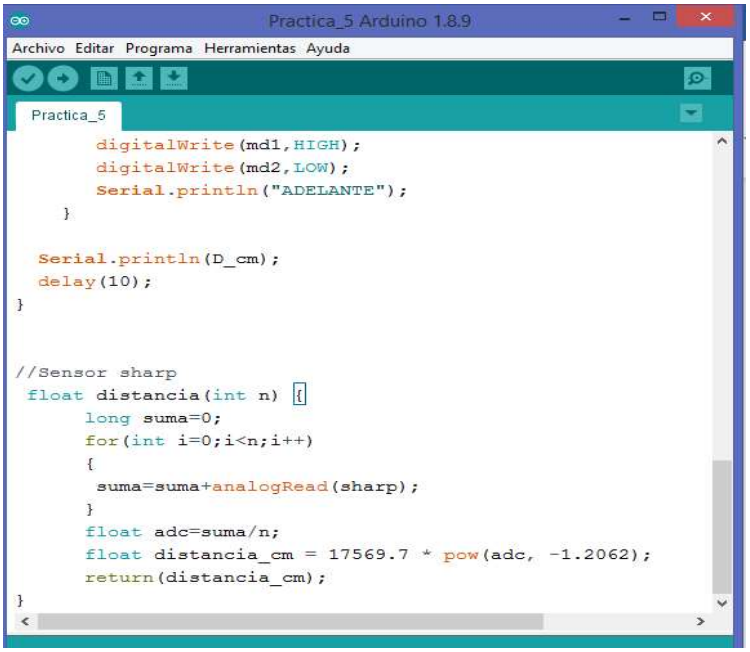


Figura 140: Lectura del adc práctica 5.

Paso 5

Conectar el cable usb (entrada c) entre la portátil y el arduino, luego dirigirse a herramientas>> puerto>>com (arduino uno) proceder a compilar y cargar el programa.

Paso 6

Una vez culminado se observa la salida del monitor serial cuando el objeto tiene una distancia cercana al sensor este proporciona un valor el cual varia cuando el objeto está más distante.

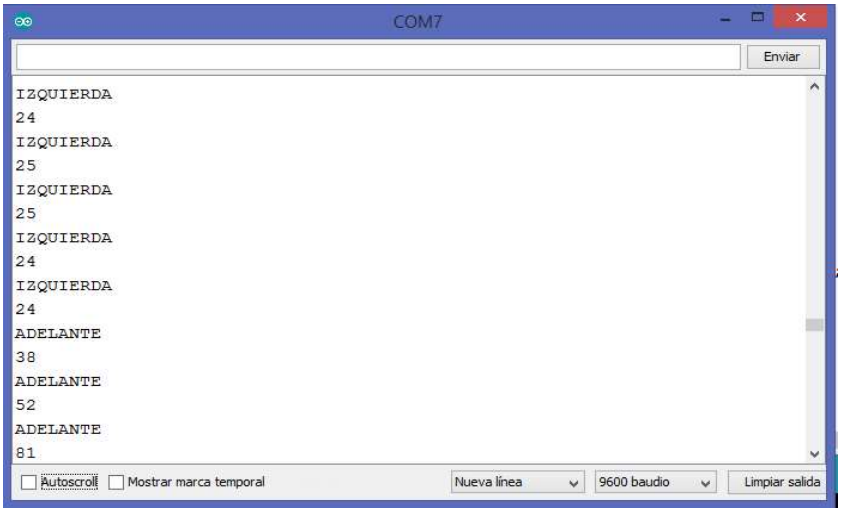


Figura 141*:* Salida del monitor serial práctica 5.

Anexo 2: Programación del esp32

```
#include <WiFi.h>
#include <WiFiClient.h>
#include <WebServer.h>
#include <WiFiAP.h>

const char* ssid     = "DRONE-WIFI";
const char* password = "WIFI-DRONE";
WebServer server(80);

//WIFI TO ARDUINO
const String UP       = "5";
const String DOWN     = "6";
const String LEFT     = "3";
const String RIGHT    = "4";
const String FRONT    = "1";
const String BACK     = "2";
const String STOP     = "7";
String speedRequested = "0";
String jumRequested = "0";
String batteryRequested = "0";
String inclinationRequested = "0";
String heightRequested = "0";

//ARDUINO TO WIFI
int velocidad = 0;
int dato = 0;
String direction = STOP;
int battery = 0;
int height = 0;
int inclination = 0;
#define SERIAL_TIMEOUT 1000
#define ARDUINO_INTERVAL 100
#define BAUD_RATE 9600
#define DEBUG false
void debug(char * message)
{
#if DEBUG
  Serial.println(message);
#endif
}
void debug(String message)
{
#if DEBUG
  Serial.println(message);
#endif
}
```

```
void setup()
{
  Serial.begin(BAUD_RATE);
  debug("Serial ready");

  WiFi.softAP(ssid, password);
  debug("Creating network");

  const IPAddress IP = WiFi.softAPIP();
  debug("IP: ");
  debug(IP.toString());

  debug("Configuring urls");
  server.on("/speed", onSpeedRequested);
  server.on("/move/up", onMoveUpRequested);
  server.on("/move/down", onMoveDownRequested);
  server.on("/move/left", onMoveLeftRequested);
  server.on("/move/right", onMoveRightRequested);
  server.on("/move/front", onMoveFrontRequested);
  server.on("/move/back", onMoveBackRequested);
  server.on("/move/stop", onMoveStopRequested);
  server.on("/battery", onBatteryRequested);
  server.on("/inclination", onInclinationRequested);
  server.on("/height", onHeightRequested);
  server.onNotFound(onNotFoundRequested);
  debug("Urls ready");

  debug("Begining server");
  server.begin();
  debug("Server ready");
}

String * split(String message, char separator, String * splitted)
{
  for (int index = 0; index < message.length(); index++)
  {
    char currentCharacter = message.charAt(index);
    if (currentCharacter == separator)
    {
      splitted[0] = message.substring(0, index);
      splitted[1] = message.substring(index + 1, message.length() - 1);
    }}}
long lastMillisComunication = millis();
String arduinoReadLine()
{
  long startTime = millis();
  while (!Serial.available())
  {
```

```
    long currentTime = millis();
    long elapsedTime = currentTime - startTime;
    if (elapsedTime < SERIAL_TIMEOUT)
    {
      return "";
    }
  }
  String line = "";
  while (true)
  {
    char character = Serial.read();
    line += character;
    if (character == '\n')
    {
      break;
    }
    while (!Serial.available());
  }
  return line;
}

void arduinoPrintln(String message)
{
  for (int index = 0; index < message.length(); index++)
  {
    Serial.print(message.charAt(index));
    delayMicroseconds(100);
  }
  Serial.println();
}
void loop()
{
  long elapsedTime = millis() - lastMillisComunication;
  if (elapsedTime > ARDUINO_INTERVAL)
  {
    dato = direction.toInt();
    arduinoPrintln("direction=" + String(direction));
    if (jumRequested == "false") {
      arduinoPrintln("jump=" + String(0));
    }
    if (jumRequested == "true") {
      arduinoPrintln("jump=" + String(1));
    }
    arduinoPrintln("speed=" + String(speedRequested));
    arduinoPrintln("battery=" + String(batteryRequested));
    arduinoPrintln("inclination=" + String(inclinationRequested));
    arduinoPrintln("height=" + String(heightRequested));
```

```
String line = arduinoReadLine();
    if (line != "")
    {
      String splitted[] = {"", ""};
      split(line, '=', splitted);
      if (splitted[0] == "velocidad") {
        String speedStr = splitted[1];

        if (!splitted[1].equals("INVALID_CMD\r"))
        {
          velocidad = speedStr.toInt();
        }
      }

      if (splitted[0] == "bateria") {
        String batteryStr = splitted[1];

        if (!splitted[1].equals("INVALID_CMD\r"))
        {
          battery = batteryStr.toInt();
        }
      }

      if (splitted[0] == "inclinacion") {
        String inclinationStr = splitted[1];

        if (!splitted[1].equals("INVALID_CMD\r"))
        {
          inclination = inclinationStr.toInt();
        }
      }

      if (splitted[0] == "altura") {
        String heightStr = splitted[1];

        if (!splitted[1].equals("INVALID_CMD\r"))
        {
          height = heightStr.toInt();
        }
      }
    }
    lastMillisComunication = millis();
  }
  server.handleClient();
}
void onNotFoundRequested()
{
  debug("{\"state\":\"NOT_FOUND\"}");
  server.send(404, "application/json", "{\"state\":\"NOT_FOUND\"}");
```

```
void onSpeedRequested()
{
  if (server.args() != 1 || server.arg(0) == "")
  {
    debug("{\"state\":\"MISSING_SPEED\"}");
    server.send(422, "application/json", "{\"state\":\"MISSING_SPEED\"}");
  }
  else
  {
    speedRequested = server.arg(0);
    debug("{\"state\":\"OK\",\"speed\":\"" + String(velocidad) + "\",\"requested\":\""
+ String(speedRequested) + "\"}");
    server.send(200,      "application/json",      "{\"state\":\"OK\",\"speed\":\""      +
String(velocidad) + "\",\"requested\":\"" + String(speedRequested) + "\"}");
  }
}
void onMoveUpRequested ()
{
  if (server.args() != 1 || server.arg(0) == "")
  {
    debug("{\"state\":\"MISSING_JUMP\"}");
    server.send(422, "application/json", "{\"state\":\"MISSING_JUMP\"}");
  }
  else
  {
    direction = UP;
    jumRequested = server.arg(0);
    debug("{\"state\":\"OK\",\"direction\":\"UP\"}");
    server.send(200, "application/json", "{\"state\":\"OK\",\"direction\":\"UP\"}");
  }
}
void onMoveDownRequested ()
{
  if (server.args() != 1 || server.arg(0) == "")
  {
    debug("{\"state\":\"MISSING_JUMP\"}");
    server.send(422, "application/json", "{\"state\":\"MISSING_JUMP\"}");
  }
  else
  {
    direction = DOWN;
    jumRequested = server.arg(0);
    debug("{\"state\":\"OK\",\"direction\":\"DOWN\"}");
    server.send(200, "application/json", "{\"state\":\"OK\",\"direction\":\"DOWN\"}");
  }
}
```

```
void onMoveLeftRequested ()
{
  if (server.args() != 1 || server.arg(0) == "")
  {
    debug("{\"state\":\"MISSING_JUMP\"}");
    server.send(422, "application/json", "{\"state\":\"MISSING_JUMP\"}");
  }
  else
  {
    direction = LEFT;
    jumRequested = server.arg(0);
    debug("{\"state\":\"OK\",\"direction\":\"LEFT\"}");
    server.send(200,                                        "application/json",
"{\"state\":\"OK\",\"direction\":\"LEFT\"}");
  }
}
void onMoveRightRequested ()
{
  if (server.args() != 1 || server.arg(0) == "")
  {
    debug("{\"state\":\"MISSING_JUMP\"}");
    server.send(422, "application/json", "{\"state\":\"MISSING_JUMP\"}");
  }
  else
  {
    direction = RIGHT;
    jumRequested = server.arg(0);
    debug("{\"state\":\"OK\",\"direction\":\"RIGHT\"}");
    server.send(200,                                        "application/json",
"{\"state\":\"OK\",\"direction\":\"RIGHT\"}");
  }
}
void onMoveFrontRequested ()
{
  if (server.args() != 1 || server.arg(0) == "")
  {
    debug("{\"state\":\"MISSING_JUMP\"}");
    server.send(422, "application/json", "{\"state\":\"MISSING_JUMP\"}");
  }
  else
  {
    direction = FRONT;
    jumRequested = server.arg(0);
    debug("{\"state\":\"OK\",\"direction\":\"FRONT\"}");
    server.send(200,                                        "application/json",
"{\"state\":\"OK\",\"direction\":\"FRONT\"}");
  }
}
```

```
void onMoveBackRequested ()
{
  if (server.args() != 1 || server.arg(0) == "")
  {
    debug("{\"state\":\"MISSING_JUMP\"}");
    server.send(422, "application/json", "{\"state\":\"MISSING_JUMP\"}");
  }
  else
  {
    direction = BACK;
    jumRequested = server.arg(0);
    debug("{\"state\":\"OK\",\"direction\":\"BACK\"}");
    server.send(200,                                        "application/json",
"{\"state\":\"OK\",\"direction\":\"BACK\"}");
  }
}
void onMoveStopRequested ()
{
  if (server.args() != 1 || server.arg(0) == "")
  {
    debug("{\"state\":\"MISSING_JUMP\"}");
    server.send(422, "application/json", "{\"state\":\"MISSING_JUMP\"}");
  }
  else
  {
    direction = STOP;
    jumRequested = server.arg(0);
    debug("{\"state\":\"OK\",\"direction\":\"STOP\"}");
    server.send(200,                                        "application/json",
"{\"state\":\"OK\",\"direction\":\"STOP\"}");
  }
}
void onBatteryRequested()
{
  batteryRequested = server.arg(0);
  debug("{\"state\":\"OK\",\"battery\":\"" + String(battery) + "\",\"requested\":\"" +
String(inclinationRequested) + "\"}");
  server.send(200,      "application/json",      "{\"state\":\"OK\",\"battery\":\""      +
String(battery) + "\",\"requested\":\"" + String(inclinationRequested) + "\"}");
}
void onInclinationRequested()
{
```

```
inclinationRequested = server.arg(0);
  debug("{\"state\":\"OK\",\"inclination\":\"" + String(inclination) + "\",\"requested\":\"" + String(inclinationRequested) + "\"}");
  server.send(200, "application/json", "{\"state\":\"OK\",\"inclination\":\"" + String(inclination) + "\",\"requested\":\"" + String(inclinationRequested) + "\"}");
}
void onHeightRequested()
{
  heightRequested = server.arg(0);
  debug("{\"state\":\"OK\",\"height\":\"" + String(height) + "\",\"requested\":\"" + String(heightRequested) + "\"}");
  server.send(200, "application/json", "{\"state\":\"OK\",\"height\":\"" + String(height) + "\",\"requested\":\"" + String(heightRequested) + "\"}");
}
```

Anexo 3: **Programación global del drone terrestre**

```
//Librerias mpu
#include "I2Cdev.h"
#include "MPU6050.h"
#include "Wire.h"
//Librerias PID
#include <PID_v1.h>
#include <Timer.h>
volatile unsigned long pulso;
volatile unsigned long tiempo;
volatile unsigned long pulsoT;
volatile unsigned long rpm;
int rpm1 = 0;
Timer t, t1;
//Motores
int mi1 = 3; //motor izquierda atras
int mi2 = 8; //motor izquierda adelante
int md1 = 11; //motor derecha atras
int md2 = 10; //motor derecha adelante
int enableA = 9; //PWM MOTOR 2
int enableB = 6; //PWM MOTOR 1

//valor PID
double kp = 2.016e+04;
double ki = 3.919 + 06;
double kd = 0;
double Setpoint, Input, Output;
PID myPID(&Input, &Output, &Setpoint, kp, ki , kd, DIRECT);
//Salto
int s1 = 7;
int cont = 1;   // Un solo salto
int flag = 0;   // validador

//Buzzer
int buzzer = 12;

//sensor sharp
double sharp = A1;

//bateria
const int bateria = A0;

//mpu
const int mpuAddress = 0x68;  // Puede ser 0x68 o 0x69
float accel_ang_x = 0;
float accel_ang_y = 0;
MPU6050 mpu(mpuAddress);
```

```
//Variables para datos de mpu
int ax, ay, az;     //acelerómetro
int gx, gy, gz;     //giroscopio

// Variable para ángulo real de giroscopio
long tiempo_prev;
float dT;
float ang_x, ang_y;
float ang_x_prev, ang_y_prev;

//MPU
void sensor() {
  mpu.getAcceleration(&ax, &ay, &az);
  mpu.getRotation(&gx, &gy, &gz);

  //tiempo de para obtencion de datos
  dT = (millis() - tiempo_prev) / 1000.0;
  tiempo_prev = millis();

  //Calcular los ángulos con acelerometro
  accel_ang_x = atan(ay / sqrt(pow(ax, 2) + pow(az, 2))) * (180.0 / 3.14);
  accel_ang_y = atan(-ax / sqrt(pow(ay, 2) + pow(az, 2))) * (180.0 / 3.14);

  //Calcular angulo de rotación con giroscopio y filtro complementario
  ang_x = 0.98 * (ang_x_prev + (gx / 131) * dT) + 0.02 * accel_ang_x;
  ang_y = 0.98 * (ang_y_prev + (gy / 131) * dT) + 0.02 * accel_ang_y;

  ang_x_prev = ang_x;
  ang_y_prev = ang_y;
}

//SERIAL
#define SERIAL_BAUD_RATE 9600
void serialBegin()
{
  Serial.begin(SERIAL_BAUD_RATE);
  while (!Serial);
  debug("Serial ready");
}

//DEBUG
#define DEBUG false
void debug(String message)
{
#if DEBUG
  Serial.println(message);
#endif
```

```
}
void debug(char * message)
{
#if DEBUG
  Serial.println(message);
#endif
}
//WIFI
#define Rx 4
#define Tx 5
#define SERIAL_TIMEOUT 1000
#define WIFI_BAUD_RATE 9600
#include <SoftwareSerial.h>
SoftwareSerial wifi(Rx, Tx);

void wifiBegin()
{
  wifi.begin(WIFI_BAUD_RATE);
  while (!wifi);
  debug("wifi ready");
}
void wifiPrintln(String message)
{
  debug("Sending to wifi: " + message);
  wifi.println(message);
}
String wifiReadLine()
{
  long startTime = millis();
  while (!wifi.available())
  {
    long currentTime = millis();
    long elapsedTime = currentTime - startTime;
    if (elapsedTime > SERIAL_TIMEOUT)
    {
      debug("timeout no data from wifi");
      return "";
    }
  }
  String line = "";
  while (true)
  {
    char character = wifi.read();
    line += character;
    debug("character received: " + character);
    if (character == '\n')
    {
      break;
```

```
}
    while (!wifi.available());
  }
  debug("received from wifi: " + line);
  return line;
}

//Sensor sharp
float distancia(int n) {
  long suma = 0;
  for (int i = 0; i < n; i++)
  {
    suma = suma + analogRead(sharp);
  }
  float adc = suma / n;
  float distancia_cm = 17569.7 * pow(adc, -1.2062);
  return (distancia_cm);
}

//MAIN
String * split(String message, char separator, String * splitted)
{
  for (int index = 0; index < message.length(); index++)
  {
    char currentCharacter = message.charAt(index);
    if (currentCharacter == separator)
    {
      splitted[0] = message.substring(0, index);
      splitted[1] = message.substring(index + 1, message.length() - 1);
    }
  }
}

void setup()
{
  pinMode(mi1, OUTPUT);
  pinMode(mi2, OUTPUT);
  pinMode(md1, OUTPUT);
  pinMode(md2, OUTPUT);
  pinMode(s1, OUTPUT);
  pinMode(buzzer, OUTPUT);
  //LOW
  digitalWrite(mi1, LOW);
  digitalWrite(mi2, LOW);
  digitalWrite(md1, LOW);
  digitalWrite(md2, LOW);
  digitalWrite(buzzer, LOW);
  serialBegin();
```

```
  wifiBegin();
  Serial.begin(9600);
  Wire.begin();
  mpu.initialize();
  t.every(250, timing);
  Setpoint = 0;
  myPID.SetMode(AUTOMATIC);
  myPID.SetOutputLimits(150, 250);
  attachInterrupt(digitalPinToInterrupt(2), Pulso1, CHANGE);  //FALLING
  Serial.println(mpu.testConnection() ? F("IMU iniciado correctamente") : F("Error al
iniciar IMU"));
}
void Pulso1() {
  pulsoT++;
}
void timing() {
  pulso = pulsoT - tiempo;
  tiempo = pulsoT;
  rpm = pulso * (60 * 2 / (4.4 * 334));
  //Serial.println(rpm);
}
void loop()
{
  String line = wifiReadLine();
  Serial.print(line);
  if (line != "") {
    String splitted[] = {"", ""};
    split(line, '=', splitted);

    if (splitted[0] == "direction") {
      long tiempo = millis(); //tiempo antes de iniciar la lectura
      int D_cm = distancia(20); //lectura de distancia (muestras)
      tiempo = millis() - tiempo; //milisegundos que duró la lectura
      if (D_cm <= 25) {
        digitalWrite(buzzer, HIGH);
      } else {
        digitalWrite(buzzer, LOW);
      }
      int dato = splitted[1].toInt();
      if (dato == 7) {  //STOP
        analogWrite(enableA, 0);
        analogWrite(enableB, 0);
        digitalWrite(mi1, LOW);
        digitalWrite(mi2, LOW);
        digitalWrite(md1, LOW);
        digitalWrite(md2, LOW);
      }
```

```
if (dato == 1) {  //ADELANTE
      analogWrite(enableA, 170);
      analogWrite(enableB, 140); //Encoder
      digitalWrite(mi1, HIGH);
      digitalWrite(mi2, LOW);
      digitalWrite(md1, HIGH);
      digitalWrite(md2, LOW);
    }
    if (dato == 2) {   //ATRAS
      analogWrite(enableA, 140); //130
      analogWrite(enableB, 110); //100
      digitalWrite(mi1, LOW);
      digitalWrite(mi2, HIGH);
      digitalWrite(md1, LOW);
      digitalWrite(md2, HIGH);
    }

    if (dato == 3) {   //IZQUIERDA
      analogWrite(enableA, 150);
      analogWrite(enableB, 250);
      digitalWrite(mi1, LOW);
      digitalWrite(mi2, HIGH);
      digitalWrite(md1, HIGH);
      digitalWrite(md2, LOW);
      delay(200);
      digitalWrite(mi1, LOW);
      digitalWrite(mi2, LOW);
      digitalWrite(md1, LOW);
      digitalWrite(md2, LOW);
      delay(750);
    }

    if (dato == 4) {  //DERECHA
      analogWrite(enableA, 200); //150
      analogWrite(enableB, 250); //250
      digitalWrite(mi1, HIGH);
      digitalWrite(mi2, LOW);
      digitalWrite(md1, LOW);
      digitalWrite(md2, HIGH);
      delay(200);
      digitalWrite(mi1, LOW);
      digitalWrite(mi2, LOW);
      digitalWrite(md1, LOW);
      digitalWrite(md2, LOW);
      delay(750);
    }
```

```
  }
  else if (splitted[0] == "speed") {
    int ref = splitted[1].toInt();  //VALOR OBTENIDO DE LA APLICACION

    if (ref > 0) {
      t.update();
      Input = rpm * 255; //Maximo 20000
      Setpoint = map(ref, 0, 100, 0, 20000);
      myPID.Compute();
      analogWrite(enableA, floor(Output));
      analogWrite(enableB, floor(Output - 30));
      digitalWrite(mi1, HIGH);
      digitalWrite(mi2, LOW);
      digitalWrite(md1, HIGH);
      digitalWrite(md2, LOW);
      int velocidad = map(Input, 0, 20000, 0, 100);
      wifiPrintln("velocidad=" + String(velocidad));
    }
  }
  else if (splitted[0] == "jump") {
    int val = splitted[1].toInt();
    if (val == 1 && cont == 1 && flag == 0) {
      digitalWrite(mi1, LOW);
      digitalWrite(md2, LOW);
      digitalWrite(mi2, LOW);
      digitalWrite(md1, LOW);
      delay(1000);
      digitalWrite(s1, HIGH);
      delay(410);
      digitalWrite(s1, LOW);
      delay(100);
      cont = 2;
    }
    if (val == 1 && cont == 2 && flag == 1) {
      digitalWrite(mi1, LOW);
      digitalWrite(md2, LOW);
      digitalWrite(mi2, LOW);
      digitalWrite(md1, LOW);
      delay(1000);
      digitalWrite(s1, HIGH);
      delay(370);
      digitalWrite(s1, LOW);
      delay(100); //200
      cont = 3;
    }
    if (val == 0) {
      digitalWrite(s1, LOW);
      if (flag == 0 && cont == 2) {
        flag = 1;
      }
```

```
if (cont == 3) {
        flag = 0;
        cont = 1;
      }
    }
  }
  else if (splitted[0] == "battery") {
    int voltaje = analogRead(bateria);
    int dato = map(voltaje, 0, 1023, 0, 100);
    wifiPrintln("bateria=" + String(dato));
  }
  else if (splitted[0] == "inclination") {
    sensor();
    int real_y = -2 * 0.99 * ang_y + 33.23;
    if (real_y < 0) {
      real_y = 0;
    }
    if (real_y > 100) {
      real_y = 100;
    }
    wifiPrintln("inclinacion=" + String(real_y));
  }
  else if (splitted[0] == "height") {
    sensor();
    int valor  = map(gz, 0, 5000, 0, 100);
    if (valor < 0) {
      valor = 0;
    }
    if (valor > 100) {
      valor = 100;
    }
    wifiPrintln("altura=" + String(valor));
  }
  else
  {
    wifiPrintln("Value=INVALID_CMD");
  }
 }
}
```

Anexo 4: Programación del esp32 – cam

```
#include "esp_camera.h"
#include <WiFi.h>
// // WARNING!!! PSRAM IC required for UXGA resolution and high JPEG quality
//              Ensure ESP32 Wrover Module or other board with PSRAM is selected
//          Partial images will be transmitted if image exceeds buffer size
//
// Select camera model
//#define CAMERA_MODEL_WROVER_KIT // Has PSRAM
//#define CAMERA_MODEL_ESP_EYE // Has PSRAM
//#define CAMERA_MODEL_M5STACK_PSRAM // Has PSRAM
//#define CAMERA_MODEL_M5STACK_WIDE        // Has PSRAM
#define CAMERA_MODEL_AI_THINKER // Has PSRAM
//#define CAMERA_MODEL_TTGO_T_JOURNAL // No PSRAM

#include "camera_pins.h"
const char* ssid = "DRONE-WIFI";
const char* password = "WIFI-DRONE";
void startCameraServer();
void setup() {
  Serial.begin(115200);
  Serial.setDebugOutput(true);
  Serial.println();
  camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
  config.pin_d5 = Y7_GPIO_NUM;
  config.pin_d6 = Y8_GPIO_NUM;
  config.pin_d7 = Y9_GPIO_NUM;
  config.pin_xclk = XCLK_GPIO_NUM;
  config.pin_pclk = PCLK_GPIO_NUM;
  config.pin_vsync = VSYNC_GPIO_NUM;
  config.pin_href = HREF_GPIO_NUM;
  config.pin_sscb_sda = SIOD_GPIO_NUM;
  config.pin_sscb_scl = SIOC_GPIO_NUM;
  config.pin_pwdn = PWDN_GPIO_NUM;
  config.pin_reset = RESET_GPIO_NUM;
  config.xclk_freq_hz = 20000000;
  config.pixel_format = PIXFORMAT_JPEG;
```

```
  // if PSRAM IC present, init with UXGA resolution and higher JPEG quality
  //                      for larger pre-allocated frame buffer.
  if(psramFound()){
    config.frame_size = FRAMESIZE_UXGA;
    config.jpeg_quality = 10;
    config.fb_count = 2;
  } else {
    config.frame_size = FRAMESIZE_SVGA;
    config.jpeg_quality = 12;
    config.fb_count = 1;  }
#if defined(CAMERA_MODEL_ESP_EYE)
  pinMode(13, INPUT_PULLUP);
  pinMode(14, INPUT_PULLUP);
#endif
  // camera init
  esp_err_t err = esp_camera_init(&config);
  if (err != ESP_OK) {
    Serial.printf("Camera init failed with error 0x%x", err);
    return;}
  sensor_t * s = esp_camera_sensor_get();
  // initial sensors are flipped vertically and colors are a bit saturated
  if (s->id.PID == OV3660_PID) {
    s->set_vflip(s, 1); // flip it back
    s->set_brightness(s, 1); // up the brightness just a bit
    s->set_saturation(s, -2); // lower the saturation }
  // drop down frame size for higher initial frame rate
  s->set_framesize(s, FRAMESIZE_QVGA);
#if defined(CAMERA_MODEL_M5STACK_WIDE)
  s->set_vflip(s, 1);
  s->set_hmirror(s, 1);
#endif
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print("."); }
  Serial.println("");
  Serial.println("WiFi connected");
  startCameraServer();
  Serial.print("Camera Ready! Use 'http://");
  Serial.print(WiFi.localIP());
  Serial.println("' to connect");
}
void loop() {
  // put your main code here, to run repeatedly:
  delay(10000);
}
```

Anexo 5: Programación android studio (Menú)

```
package e.jordan.presentacion;

import android.Manifest;
import android.app.AlertDialog;
import android.app.ProgressDialog;
import android.content.ContentValues;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.net.Uri;
import android.os.AsyncTask;
import android.provider.BaseColumns;
import android.provider.MediaStore;
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import androidx.cardview.widget.CardView;
import android.util.Log;
import android.view.View;

import android.view.WindowManager;
import android.widget.ProgressBar;
import android.widget.Toast;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.Task;
import com.google.android.gms.tasks.Tasks;
import com.google.android.material.snackbar.Snackbar;
import com.google.firebase.firestore.FieldValue;
import com.google.firebase.firestore.FirebaseFirestore;
import com.google.firebase.firestore.SetOptions;

import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashMap;
import java.util.List;
import java.util.Locale;
import java.util.Map;

import e.jordan.database.DatabaseHelper;
import e.jordan.database.MeasurementsContract;
import e.jordan.libraries.Faker;
import e.jordan.presentacion.tools.RuntimePermissionsHelper;

import e.jordan.database.DatabaseHelper;
import e.jordan.database.MeasurementsContract;
import e.jordan.libraries.Faker;
import e.jordan.presentacion.tools.RuntimePermissionsHelper;
```

```
public class Inicio extends AppCompatActivity implements
View.OnClickListener {


   private CardView galeriaCard, conexionCard, pilotajesCard, mapaCard,
instruccionesCard, startCard, uploadCard;
   private ProgressBar progressBar;
   public View loadingBackground;
   private static final int PERMISSION_CODE = 1000;
   private static final int IMAGE_CAPTURE_CODE = 1001;

   DatabaseHelper databaseHelper;
   SQLiteDatabase sqLiteDatabase;
Uri image_uri;

public static final String[]requiredPermissions = {
   Manifest.permission.READ_EXTERNAL_STORAGE,
   Manifest.permission.WRITE_EXTERNAL_STORAGE,
};
@Override
protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);

   if(RuntimePermissionsHelper.applicationHasAllPermissions(this,
Inicio.requiredPermissions))
   {
      setContentView(R.layout.activity_inicio);

      this.databaseHelper = new DatabaseHelper(this.getApplicationContext());
      this.sqLiteDatabase = this.databaseHelper.getReadableDatabase();

            this.galeriaCard = findViewById(R.id.Galeria_card);
      this.conexionCard = findViewById(R.id.Conexion_card);
      this.pilotajesCard = findViewById(R.id.Grafica_card);
      this.mapaCard = findViewById(R.id.Mapa_card);
      this.instruccionesCard =  findViewById(R.id.Instrucciones_card);
      this.startCard = findViewById(R.id.Start_card);
      this.uploadCard = findViewById(R.id.Upload_card);

 this.progressBar = findViewById(R.id.progressBar);
 this.loadingBackground = findViewById(R.id.loadingBackground);
this.galeriaCard.setOnClickListener(this);
this.conexionCard.setOnClickListener(this);
this.pilotajesCard.setOnClickListener(this);
this.mapaCard.setOnClickListener(this);
this.instruccionesCard.setOnClickListener(this);
this.startCard.setOnClickListener(this);
this.uploadCard.setOnClickListener(this);
this.startCard.setOnClickListener((View view) -> {
   Intent intent = new Intent(Inicio.this, Start.class);
   startActivity(intent);
});
   }
```

```
 else
   {
      Log.e(Conexion.TAG,          "onCreate          no          permissions");
      setContentView(R.layout.activity_no_permisions);
   }
}

public                void                getAllPermissions(View                view)
{
   Log.e(Conexion.TAG,                                        "getAllPermissions");
   RuntimePermissionsHelper.getAllRequiredPermissions(this,
Inicio.requiredPermissions);
}
public void uploadData()
{
   String[] projection = {
      BaseColumns._ID,
      MeasurementsContract.MeasurementEntry.COLUMN_NAME_SPEED,
      MeasurementsContract.MeasurementEntry.COLUMN_NAME_INCLINATION,
      MeasurementsContract.MeasurementEntry.COLUMN_NAME_HEIGHT,
      MeasurementsContract.MeasurementEntry.COLUMN_NAME_BATTERY,
      MeasurementsContract.MeasurementEntry.COLUMN_NAME_DIRECTION,
      MeasurementsContract.MeasurementEntry.COLUMN_NAME_CREATED_AT,
   };
Cursor cursor = this.sqLiteDatabase.query(
   MeasurementsContract.MeasurementEntry.TABLE_NAME,
   projection,
   null,
   null,
   null,
   null,
   null
);
Log.e("data", "cursor: " + cursor.getCount());
if(cursor.getCount() == 0)
{
   Snackbar.make(progressBar, "No hay registros pendientes de subir",
Snackbar.LENGTH_SHORT).show();
}
else
{
      AlertDialog.Builder builder = new AlertDialog.Builder(this);
      builder.setMessage(String.format(Locale.US, "Desea cargar \"%d\"
registros", cursor.getCount()));
      builder.setPositiveButton("Si, cargar", (DialogInterface dialog, int id) -> {
         new UploadTask(cursor).execute();
      });
      builder.setNegativeButton("No", (DialogInterface dialog, int id) -> { });
      AlertDialog alertDialog = builder.create();
      alertDialog.show();
   }
}
```

```
class UploadTask extends AsyncTask<Void, Integer, Boolean>
{
   private Cursor cursor;
   private int progress = 0;
   private List<Exception> exceptions;

   UploadTask(Cursor cursor) {
      this.cursor = cursor;
      this.exceptions = new ArrayList<>();
   }
protected void onPreExecute()
{
   progressBar.setVisibility(View.VISIBLE);
   loadingBackground.setVisibility(View.VISIBLE);
   getWindow().setFlags(WindowManager.LayoutParams.FLAG_NOT_TOUCHABLE,
WindowManager.LayoutParams.FLAG_NOT_TOUCHABLE);
}

protected Boolean doInBackground(Void... nothings)
{
   List<Integer> speeds = new ArrayList<>();
   List<Integer> inclinations = new ArrayList<>();
   List<Integer> heights = new ArrayList<>();
   List<Integer> batteries = new ArrayList<>();
   List<Integer> directions = new ArrayList<>();
   List<String> datetimes = new ArrayList<>();
while(this.cursor.moveToNext())
{
   int speed =
this.cursor.getInt(cursor.getColumnIndexOrThrow(MeasurementsContract.Measurem
entEntry.COLUMN_NAME_SPEED));
   int inclination =
this.cursor.getInt(cursor.getColumnIndexOrThrow(MeasurementsContract.Measurem
entEntry.COLUMN_NAME_INCLINATION));
   int height =
this.cursor.getInt(cursor.getColumnIndexOrThrow(MeasurementsContract.Measurem
entEntry.COLUMN_NAME_HEIGHT));
Int battery =
this.cursor.getInt(cursor.getColumnIndexOrThrow(MeasurementsContract.Measurem
entEntry.COLUMN_NAME_BATTERY));
   int direction =
this.cursor.getInt(cursor.getColumnIndexOrThrow(MeasurementsContract.Measurem
entEntry.COLUMN_NAME_DIRECTION));
   String datetime = this.cursor.getString
(cursor.getColumnIndexOrThrow(MeasurementsContract.MeasurementEntry.COLUM
N_NAME_CREATED_AT));
speeds.add(speed);
   inclinations.add(inclination);
   heights.add(height);
   batteries.add(battery);
   directions.add(direction);
   datetimes.add(datetime);
```

```
    this.publishProgress(this.progress++);
}
this.cursor.close();

final Map<String, Object> data = new HashMap<>();
data.put("speed", speeds);
data.put("inclination", inclinations);
data.put("height", heights);
data.put("battery", batteries);
data.put("direction", directions);
data.put("datetime", datetimes);
FirebaseFirestore firebaseFirestore = FirebaseFirestore.getInstance();
Task<Void> task = firebaseFirestore
        .collection("sensors")
        .document(datetimes.get(0))
        .set(data, SetOptions.merge());

try
{
    Log.e("data", "submitting task");
    Tasks.await(task);

    String selection = MeasurementsContract.MeasurementEntry._ID + " > ?";
    String[] selectionArgs = { "0" };

sqLiteDatabase.delete(MeasurementsContract.MeasurementEntry.TABLE_NAME,
selection, selectionArgs);
    Log.e("data", "deleted");
}
catch (Exception exception)
{
    Log.e("data", exception.getMessage());

    StackTraceElement[] stackTraceElements = exception.getStackTrace();
    for(int index = 0; index < stackTraceElements.length; index ++)
    {
            Log.e("data", stackTraceElements[index].toString());
        }

        this.exceptions.add(exception);
    }

    return true;
}

protected void onProgressUpdate(Integer... progress)
protected void onPostExecute(Boolean result)
{
    if(this.exceptions.size() > 0)
```

```
    {
        Snackbar snackbar = Snackbar.make(progressBar, "No todos los datos fueron
cargados",                                          Snackbar.LENGTH_SHORT);
        snackbar.setAction("Reintentar",          (View          view)          ->          {
            uploadData();
        });
        snackbar.show();
    }
    else
    {
        Snackbar.make(progressBar,   "Todos   los   datos   fueron   cargados",
Snackbar.LENGTH_SHORT).show();
    }
progressBar.setVisibility(View.GONE);
        loadingBackground.setVisibility(View.GONE);

getWindow().clearFlags(WindowManager.LayoutParams.FLAG_NOT_TOUCHABLE);
    }
}

@Override
public void onClick(View v) {

    Intent i;
    switch (v.getId()) {
        case R.id.Conexion_card:
            i = new Intent(this, Conexion.class); startActivity(i); break;
case R.id.Mapa_card:
    i = new Intent(this, MapaRuta.class);  startActivity(i); break;

case R.id.Instrucciones_card:
    i = new Intent(this, Instrucciones.class);  startActivity(i); break;
case R.id.Start_card:
    i = new Intent(this, Start.class); startActivity(i); break;

case R.id.Upload_card:
    uploadData();
    break;
default:break;

    }


}
private                void                openCamera()                {
    ContentValues                values=new                ContentValues();
    values.put(MediaStore.Images.Media.TITLE,"New                Picture");
    values.put(MediaStore.Images.Media.DESCRIPTION,"From    the    CAMERA");

image_uri=getContentResolver().insert(MediaStore.Images.Media.EXTERNAL_CON
TENT_URI,values);
```

```
    Intent cameraIntent= new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
    cameraIntent.putExtra(MediaStore.EXTRA_OUTPUT,image_uri);
    startActivityForResult (cameraIntent,IMAGE_CAPTURE_CODE);
}
public void onRequestPermissionsResult(int requestCode, @NonNull String[]
permissions, @NonNull int[] grantResults) {

    switch (requestCode){
        case PERMISSION_CODE:
            if(grantResults.length>0&&
grantResults[0]==PackageManager.PERMISSION_GRANTED){
                openCamera();
            } else{
                Toast.makeText(this,"Permission
deniend",Toast.LENGTH_LONG).show();
            }
            break;
        case RuntimePermissionsHelper.PERMISSIONS_REQUIRED:
            if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED)
            {
                Log.e(Conexion.TAG, "onActivityResult OK");
                this.finish();

                Intent intent = new Intent(this, Inicio.class);
                startActivity(intent);
            }
            break;
    }
}
}
```

Anexo 6: Programación android studio (Inicio)

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical"
   tools:context=".Inicio"
   tools:ignore="ExtraText">

   <androidx.appcompat.widget.Toolbar
      android:id="@+id/toolbar"
      android:layout_width="match_parent"
      android:layout_height="?attr/actionBarSize"
      android:background="?attr/colorPrimary"
      android:gravity="center"
      app:popupTheme="@style/AppTheme.PopupOverlay"
      app:title="MENÚ"
      app:titleTextColor="@color/blanco" />

   <ProgressBar
      android:id="@+id/progressBar"
      android:visibility="gone"
      android:indeterminate="true"
      style="@style/Widget.AppCompat.ProgressBar.Horizontal"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"/>
<FrameLayout
   android:layout_width="match_parent"
   android:layout_height="match_parent">
   <ScrollView
      android:layout_width="match_parent"
      android:layout_height="match_parent">

      <LinearLayout
         android:layout_width="match_parent"
         android:layout_height="match_parent"
         android:gravity="center"
         android:orientation="vertical">

         <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:gravity="center"
            android:orientation="horizontal">

            <androidx.cardview.widget.CardView
               android:visibility="gone"
               android:id="@+id/Galeria_card"
               android:layout_width="160dp"
               android:layout_height="170dp"
               android:layout_margin="10dp">
```

```
<LinearLayout
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:gravity="center"
      android:orientation="vertical">

      <ImageView
         android:layout_width="64dp"
         android:layout_height="64dp"
         android:background="@drawable/circlebakgroundpurple"
         android:padding="10dp"
         android:src="@drawable/galeria" />

      <TextView
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:layout_marginTop="10dp"
         android:text="GALERÍA"
         android:textStyle="bold" />

   </LinearLayout>

</androidx.cardview.widget.CardView>

<androidx.cardview.widget.CardView
   android:visibility="gone"
   android:id="@+id/Grafica_card"
   android:layout_width="160dp"
   android:layout_height="170dp"
   android:layout_margin="10dp">


   <LinearLayout
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:gravity="center"
      android:orientation="vertical">

      <ImageView
         android:layout_width="64dp"
         android:layout_height="64dp"
         android:background="@drawable/circlebackgroundorange"
         android:padding="1dp"
         android:src="@drawable/ic_grafica" />

      <TextView
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:layout_marginTop="10dp"
         android:text="GRÁFICAS"
         android:textStyle="bold" />

   </LinearLayout>
```

```
</androidx.cardview.widget.CardView>

</LinearLayout>

<LinearLayout
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:clipToPadding="false"
   android:gravity="center"
   android:orientation="horizontal">

   <androidx.cardview.widget.CardView
      android:id="@+id/Conexion_card"
      android:layout_width="160dp"
      android:layout_height="170dp"
      android:layout_margin="10dp">

      <LinearLayout
         android:layout_width="match_parent"
         android:layout_height="match_parent"
         android:gravity="center"
         android:orientation="vertical">
<ImageView
         android:layout_width="64dp"
         android:layout_height="64dp"
         android:background="@drawable/circlebackgroundred"
         android:padding="10dp"
         android:src="@drawable/wifi" />

      <TextView
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:layout_marginTop="10dp"
         android:text="CONEXIONES"
         android:textStyle="bold" />
   </LinearLayout>

</androidx.cardview.widget.CardView>
<androidx.cardview.widget.CardView
   android:id="@+id/Mapa_card"
   android:layout_width="160dp"
   android:layout_height="170dp"
   android:layout_margin="10dp">

   <LinearLayout
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:gravity="center"
      android:orientation="vertical">
```

```
<ImageView
            android:layout_width="64dp"
            android:layout_height="64dp"
            android:background="@drawable/circlebackgroundceleste"
            android:padding="10dp"
            android:src="@drawable/mapa"                                    />

         <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginTop="10dp"
            android:text="MAPA                    DE                    RUTA"
            android:textStyle="bold"                                        />
      </LinearLayout>

   </androidx.cardview.widget.CardView>

</LinearLayout>

<LinearLayout
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:clipToPadding="false"
   android:gravity="center"
   android:orientation="horizontal">
<androidx.cardview.widget.CardView
     android:id="@+id/Instrucciones_card"
     android:layout_width="160dp"
     android:layout_height="170dp"
     android:layout_margin="10dp">
   <LinearLayout
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:gravity="center"
      android:orientation="vertical">

      <ImageView
         android:layout_width="64dp"
         android:layout_height="64dp"
         android:background="@drawable/circlebackgroundcafe"
         android:padding="10dp"
         android:src="@drawable/instrucciones" />

      <TextView
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:layout_marginTop="10dp"
         android:text="INSTRUCCIONES"
         android:textStyle="bold" />

   </LinearLayout>

</androidx.cardview.widget.CardView>
```

```
<androidx.cardview.widget.CardView
      android:id="@+id/Start_card"
      android:layout_width="160dp"
      android:layout_height="170dp"
      android:layout_margin="10dp">

      <LinearLayout
         android:layout_width="match_parent"
         android:layout_height="match_parent"
         android:gravity="center"
         android:orientation="vertical">

         <ImageView
            android:id="@id/Start_card"
            android:layout_width="64dp"
            android:layout_height="64dp"
            android:background="@drawable/circlebackgroundgreen"
            android:padding="10dp"
            android:src="@drawable/start"                                    />

         <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginTop="10dp"
            android:text="START"
            android:textStyle="bold"                                         />
      </LinearLayout>

   </androidx.cardview.widget.CardView>

</LinearLayout>
<LinearLayout
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:clipToPadding="false"
   android:gravity="center"
   android:orientation="horizontal">

   <androidx.cardview.widget.CardView
      android:id="@+id/Upload_card"
      android:layout_width="160dp"
      android:layout_height="170dp"
      android:layout_margin="10dp">

      <LinearLayout
         android:layout_width="match_parent"
         android:layout_height="match_parent"
         android:gravity="center"
         android:orientation="vertical">

         <ImageView
            android:layout_width="64dp"
            android:layout_height="64dp"
            android:background="@drawable/circlebackgroundorange"
            android:padding="10dp"
            android:src="@drawable/upload" />
```

```
<TextView
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_marginTop="10dp"
                    android:text="SUBIR                    REGISTROS"
                    android:textStyle="bold"                    />

            </LinearLayout>

          </androidx.cardview.widget.CardView>

        </LinearLayout>
      </LinearLayout>
    </ScrollView>

    <View
      android:id="@+id/loadingBackground"
      android:visibility="gone"
      android:layout_height="match_parent"
      android:layout_width="match_parent"
      android:background="#CC000000"/>
  </FrameLayout>

</LinearLayout>
```

Anexo 7: Programación android studio (Conexión)

```
package e.jordan.presentacion;

import android.app.Dialog;
import android.app.ProgressDialog;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.content.pm.PackageManager;
import android.net.wifi.ScanResult;
import android.net.wifi.WifiManager;
import androidx.annotation.NonNull;
import com.google.android.material.snackbar.Snackbar;
import androidx.swiperefreshlayout.widget.SwipeRefreshLayout;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.CompoundButton;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.Switch;
import android.widget.TextView;

import com.thanosfisherman.wifiutils.WifiUtils;
import com.thanosfisherman.wifiutils.wifiConnect.ConnectionSuccessListener;

import java.util.List;

import e.jordan.presentacion.tools.IntentFilters;
import e.jordan.presentacion.tools.RuntimePermissionsHelper;
import e.jordan.presentacion.tools.wifi.Network;
import e.jordan.presentacion.tools.wifi.WifiHelpers;


public class Conexion extends AppCompatActivity
{
public static final String TAG = "Conexion";

private WifiHelpers wifiHelper;

private Switch switchWiFi;
private WifiManager wifiManager;
private ListView availableWifisListView;
private Context context;
private String availableWifis[];
private Network availableWifisNetworks[];
private WifiScanBroadcastReceiver wifiScanBroadcastReceiver;
private SwipeRefreshLayout swipeRefreshLayout;
```

```
private final static int PERMISSIONS_STATE = 1000;
private void setupObjects()
{
   Log.e(Conexion.TAG, "setupObjects");
   this.context = getApplicationContext();

   this.switchWiFi = findViewById(R.id.switchWifi);
   this.availableWifisListView = findViewById(R.id.availableWifisListView);
   this.swipeRefreshLayout = findViewById(R.id.swipeRefreshLayout);

   this.wifiHelper = new WifiHelpers(this);
   this.wifiManager = (WifiManager)
this.context.getSystemService(Context.WIFI_SERVICE);
   this.wifiScanBroadcastReceiver = new WifiScanBroadcastReceiver();
}

private void setupUi()
{
   Log.e(Conexion.TAG, "setupUi");
   this.switchWiFi.setChecked(this.wifiHelper.isWifiEnabled());
}

private void setupListeners()
{
Log.e(Conexion.TAG, "setupListeners");
this.switchWiFi.setOnCheckedChangeListener(new
CompoundButton.OnCheckedChangeListener()
{
   @Override public void onCheckedChanged(CompoundButton buttonView,
boolean isChecked)
   {
      Log.e(Conexion.TAG, "setupObjects::onCheckedChanged");
      turnOnOffWifi(isChecked);
   }
});
this.swipeRefreshLayout.setOnRefreshListener(new
SwipeRefreshLayout.OnRefreshListener()
{
   @Override public void onRefresh()
   {
      Log.e(Conexion.TAG, "setupObjects::onRefresh");
      emptyAvailableWifisList();
      wifiHelper.startScan();
   }
});
this.availableWifisListView.setOnItemClickListener(new
AdapterView.OnItemClickListener()
{
   @Override public void onItemClick(AdapterView<?> adapterView, View view,
int i, long l)
```

```
{
        Log.e(Conexion.TAG,                    "setupObjects::onItemClick");
        showConnectToSSIDDialog(availableWifisNetworks[i]);
    }
  });
}

private    void    showConnectToSSIDDialog(final    Network    wifiSSID)
{
  Log.e(Conexion.TAG,                    "showConnectToSSIDDialog");
  final    Dialog    dialog    =    new    Dialog(this);
  dialog.setContentView(R.layout.dialog_connect);
  dialog.setTitle("Conectarse    a    la    red");
  TextView    textSSID    =    dialog.findViewById(R.id.textSSID1);

  Button    dialogButton    =    dialog.findViewById(R.id.okButton);
  final    EditText    pass    =    dialog.findViewById(R.id.textPassword);
  textSSID.setText(wifiSSID.get("SSID"));

  dialogButton.setOnClickListener(new    View.OnClickListener()
  {
    @Override    public    void    onClick(View    v)
    {
        Log.e(Conexion.TAG,    "showConnectToSSIDDialog::onClick");

        String    ssid    =    wifiSSID.get("SSID");
        String    password    =    pass.getText().toString();

        ProgressDialog    progressDialog    =    ProgressDialog.show(Conexion.this,
"Conectando",    "Conectado    con    la    red    wifi    "    +    ssid);
        progressDialog.show();
WifiUtils
    .withContext(context)
    .connectWith(ssid, password)
    .onConnectionResult(new ConnectionSuccessListener()
    {
        @Override public void isSuccessful(boolean isSuccess)
        {
            Log.e(Conexion.TAG, "showConnectToSSIDDialog:isSuccessful");
            progressDialog.hide();
            if(isSuccess)
            {
  Log.e(Conexion.TAG, "showConnectToSSIDDialog::isSuccessful true");
  Snackbar.make(availableWifisListView,    "Conectado",
Snackbar.LENGTH_SHORT).show();
}
else
{
  Log.e(Conexion.TAG,    "showConnectToSSIDDialog::isSuccessful    false");
  Snackbar.make(availableWifisListView,    "Problemas    conectado",
Snackbar.LENGTH_SHORT).show();
} }
```

```
})
                .start();
            dialog.dismiss();
        }
    });
    dialog.show();
}

private void emptyAvailableWifisList()
{
    Log.e(Conexion.TAG, "emptyAvailableWifisList");
    String[] emptyArray = new String[0];
    ArrayAdapter<String> wifisAvailable = new ArrayAdapter<>(context
,android.R.layout.simple_list_item_1, android.R.id.text1, emptyArray);
    this.availableWifisListView.setAdapter(wifisAvailable);
}

private void turnOnOffWifi(boolean isChecked)
{
    Log.e(Conexion.TAG, "turnOnOffWifi");
    this.wifiHelper.setWifiEnabled(isChecked);
    if (!isChecked) {
        emptyAvailableWifisList();
    }
}
@Override protected void onPause() {
    super.onPause();
    Log.e(Conexion.TAG, "onPause");
    if(RuntimePermissionsHelper.applicationHasAllPermissions(this,
WifiHelpers.requiredPermissions))
    {
        Log.e(Conexion.TAG, "onPause::refresh");
        this.swipeRefreshLayout.setRefreshing(false);
        unregisterReceiver(this.wifiScanBroadcastReceiver);
    }
}

@Override protected void onStop() {
    super.onStop();
    Log.e(Conexion.TAG, "onStop");
}
@Override protected void onDestroy() {
    super.onDestroy();
    Log.e(Conexion.TAG, "onDestroy");
}

@Override protected void onStart() {
    super.onStart();
    Log.e(Conexion.TAG, "onStart");
    if(RuntimePermissionsHelper.applicationHasAllPermissions(this,
WifiHelpers.requiredPermissions))
    {
```

```
Log.e(Conexion.TAG, "onStart::refresh");
    this.swipeRefreshLayout.setRefreshing(true);
    IntentFilter intentFilter = IntentFilters.create(
        WifiManager.SCAN_RESULTS_AVAILABLE_ACTION
    );
    registerReceiver(this.wifiScanBroadcastReceiver, intentFilter);
    this.wifiHelper.startScan();
  }
}

@Override protected void onRestart() {
  super.onRestart();
  Log.e(Conexion.TAG, "onRestart");
}

protected void onResume()
{
  super.onResume();
  Log.e(Conexion.TAG, "onResume");
}

public void getAllPermissions(View view)
{
  Log.e(Conexion.TAG, "getAllPermissions");
  RuntimePermissionsHelper.getAllRequiredPermissions(this,
WifiHelpers.requiredPermissions);
}

@Override public void onRequestPermissionsResult(int requestCode, @NonNull
String[] permissions, @NonNull int[] grantResults)
{
  switch (requestCode)
  {
    case RuntimePermissionsHelper.PERMISSIONS_REQUIRED:
      if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED)
      {
        Log.e(Conexion.TAG, "onActivityResult OK");
        this.finish();

        Intent intent = new Intent(this, Conexion.class);
        startActivity(intent);
      }
  }
}
@Override protected void onCreate(Bundle savedInstanceState)
{
  super.onCreate(savedInstanceState);
  Log.e(Conexion.TAG, "onCreate");
  if(RuntimePermissionsHelper.applicationHasAllPermissions(this,
WifiHelpers.requiredPermissions))
```

```
{
        Log.e(Conexion.TAG, "with permissions");
        setContentView(R.layout.activity_conexion);

setupObjects();
        setupUi();
        setupListeners();
    }
    else
    {
        Log.e(Conexion.TAG, "onCreate no permissions");
        setContentView(R.layout.activity_no_permisions);

    }
}

class WifiScanBroadcastReceiver extends BroadcastReceiver
{
    public void onReceive(Context c, Intent intent)
    {
        Log.e(Conexion.TAG, "WifiScanBroadcastReceiver::onReceive");
        String action = intent.getAction();
        Log.e(Conexion.TAG, "WifiScanBroadcastReceiver::onReceive " + action);
        if(action != null)
        {
            switch (action)
            {
    case WifiManager.SCAN_RESULTS_AVAILABLE_ACTION:
    List<ScanResult> wifiScanList = wifiManager.getScanResults();
    availableWifis = new String[wifiScanList.size()];
    availableWifisNetworks = new Network[wifiScanList.size()];
    for(int index = 0; index < wifiScanList.size(); index++)
    {
        String wifi = wifiScanList.get(index).toString();

        Log.e(Conexion.TAG, "WifiScanBroadcastReceiver::onReceive " + wifi);

        String[] temp = wifi.split(",");
        availableWifis[index] = temp[0].substring(5).trim
availableWifisNetworks[index] = new Network(wifi);
    }
    ArrayAdapter<String> wifisAvailable = new ArrayAdapter<>(context
,android.R.layout.simple_list_item_1, android.R.id.text1, availableWifis);
    availableWifisListView.setAdapter(wifisAvailable);

swipeRefreshLayout.setRefreshing(false);
                break;
            }
        }
    }
}
```

Anexo 8: Programación android studio (Conexión)

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context=".Conexion">
   <LinearLayout
      android:orientation="vertical"
      android:layout_width="match_parent"
      android:layout_height="match_parent">
      <androidx.appcompat.widget.Toolbar
         android:id="@+id/toolbar"
         android:layout_width="match_parent"
         android:layout_height="?attr/actionBarSize"
         android:gravity="center"
         android:layout_weight="0"
         android:background="?attr/colorPrimary"
         app:popupTheme="@style/AppTheme.PopupOverlay"
         app:title="CONECTIVIDAD"
         app:titleTextColor="@color/blanco" />
<LinearLayout
      android:padding="10dp"
      android:orientation="vertical"
      android:layout_width="match_parent"
      android:layout_height="match_parent">
      <Switch
         android:id="@+id/switchWifi"
         android:layout_width="match_parent"
         android:layout_height="wrap_content"
         android:text="WIFI"
         android:textSize="25sp"
         android:textOn="Wi-Fi On"
         android:textOff="Wi-Fi Off" />
      <androidx.swiperefreshlayout.widget.SwipeRefreshLayout
         android:id="@+id/swipeRefreshLayout"
         android:layout_width="match_parent"
         android:layout_height="match_parent">
         <ListView
            android:id="@+id/availableWifisListView"
            android:layout_width="match_parent"
            android:layout_height="match_parent"/>
      </androidx.swiperefreshlayout.widget.SwipeRefreshLayout>
   </LinearLayout>
</LinearLayout>
```

Anexo 9: Programación android studio (Mapa Ruta)

```
package e.jordan.presentacion;

import android.os.Bundle;
import com.google.android.material.tabs.TabLayout;
import androidx.fragment.app.Fragment;
import androidx.fragment.app.FragmentManager;
import androidx.fragment.app.FragmentPagerAdapter;
import androidx.viewpager.widget.ViewPager;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;
import android.text.Editable;
import android.text.TextWatcher;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.EditText;
import android.widget.TextView;

import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

import e.jordan.presentacion.fragments.mapa_de_ruta.ControlFragment;
import e.jordan.presentacion.tools.File;

public class MapaRuta extends AppCompatActivity
{
   private Toolbar toolbar;
   private TabLayout tabLayout;
   private ViewPager viewPager;
   private int tabVisible = 0;

   private EditText editTextUrl;
   private TextView textViewError;

   @Override protected void onCreate(Bundle savedInstanceState)
   {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_mapa_ruta);

      this.toolbar = findViewById(R.id.toolbar);
      this.toolbar.setTitle("MAPA DE RUTA");
      setSupportActionBar(toolbar);

      this.editTextUrl = findViewById(R.id.editTextUrl);
      try {
         String ip = File.read(MapaRuta.this, "IP.txt");
         this.editTextUrl.setText(ip);
      }catch(IOException e) {
         this.editTextUrl.setText("http://192.168.4.1");
      }
```

```
this.editTextUrl.addTextChangedListener(new TextWatcher() {
    @Override public void beforeTextChanged(CharSequence charSequence, int i,
int i1, int i2) { }
    @Override public void onTextChanged(CharSequence charSequence, int i, int
i1, int i2) {
      try {
        File.write(MapaRuta.this, "IP.txt", charSequence.toString());
      }catch (Exception e) {
        e.printStackTrace();
      }
    }
    @Override public void afterTextChanged(Editable editable) { }
  });
  this.textViewError = findViewById(R.id.textViewError);

  this.viewPager = findViewById(R.id.viewpager);
  setupViewPager(this.viewPager);

  this.tabLayout = findViewById(R.id.tabs);
  this.tabLayout.setupWithViewPager(this.viewPager);
}
@Override public boolean onCreateOptionsMenu(Menu menu)
{
  getMenuInflater().inflate(R.menu.main, menu);
  return true;
}
@Override public boolean onOptionsItemSelected(MenuItem item)
{
  switch (item.getItemId())
  {
    case R.id.togglelp:
      this.editTextUrl.setVisibility(this.editTextUrl.getVisibility() ==
View.GONE?View.VISIBLE:View.GONE);
      return true;
    default: return super.onOptionsItemSelected(item);
  }
}
private void setupViewPager(ViewPager viewPager)
{
ControlFragment controlFragment = new ControlFragment();
controlFragment.setOnCurrentTabChangedListener(() -> this.tabVisible);
controlFragment.setOnIpChangedListener(() -> this.editTextUrl.getText().toString());
controlFragment.setOnHttpErrorListenerListener((String message) ->
  {
    if(message.isEmpty())
    {
      this.textViewError.setVisibility(View.GONE);
    }
    else
    {
      this.textViewError.setText(message);
      this.textViewError.setVisibility(View.VISIBLE);
    } });
```

```
ViewPagerAdapter adapter = new
ViewPagerAdapter(getSupportFragmentManager());
adapter.addFragment(controlFragment, "Control");
viewPager.setAdapter(adapter);
   viewPager.addOnPageChangeListener(new
ViewPager.OnPageChangeListener()
   {
      @Override public void onPageScrolled(int position, float positionOffset, int
positionOffsetPixels)
      {
         if(position != tabVisible)
         {
            tabVisible = position;
            Log.e("DEBUG",
"MapaRuta.setupViewPager.OnPageChangeListener.onPageScrolled
"+position);
         }
      }

      @Override public void onPageSelected(int position) { }
      @Override public void onPageScrollStateChanged(int state) { }
   });
}
class ViewPagerAdapter extends FragmentPagerAdapter
{
   private final List<Fragment> mFragmentList = new ArrayList<>();
   private final List<String> mFragmentTitleList = new ArrayList<>();

   public ViewPagerAdapter(FragmentManager manager) {
      super(manager);
   }
  @Override
   public Fragment getItem(int position) {
      return mFragmentList.get(position);
   }

   @Override
   public int getCount() {
      return this.mFragmentList.size();  }

public void addFragment(Fragment fragment, String title)
   {
      this.mFragmentList.add(fragment);
      this.mFragmentTitleList.add(title);
   }

   @Override
   public CharSequence getPageTitle(int position) {
      return this.mFragmentTitleList.get(position);
   }
}
```

Anexo 10: Programación android studio (Mapa Ruta)

```
<LinearLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical"
   xmlns:tools="http://schemas.android.com/tools"
   tools:context=".MapaRuta">
   <com.google.android.material.appbar.AppBarLayout
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar">
      <androidx.appcompat.widget.Toolbar
         android:id="@+id/toolbar"
         android:layout_width="match_parent"
         android:layout_height="?attr/actionBarSize"
         android:background="?attr/colorPrimary"
         app:layout_scrollFlags="scroll|enterAlways"
         app:popupTheme="@style/ThemeOverlay.AppCompat.Light">
         <EditText
            android:visibility="gone"
            android:id="@+id/editTextUrl"
            android:textColor="#fff"
            android:text="http://192.168.4.1"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"                                    />
      </androidx.appcompat.widget.Toolbar>

<com.google.android.material.tabs.TabLayout
      android:visibility="gone"
      android:id="@+id/tabs"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      app:tabMode="fixed"
      app:tabGravity="fill"/>
</com.google.android.material.appbar.AppBarLayout>

<LinearLayout
      android:orientation="vertical"
      android:layout_width="match_parent"
      android:layout_height="match_parent">
      <TextView
         android:id="@+id/textViewError"
         android:layout_width="match_parent"
         android:layout_height="wrap_content"
         android:layout_gravity="end"
         android:background="#ff0000"
         android:padding="10dp"
         android:text="asdasd"
         android:visibility="gone"
         android:textAlignment="textEnd"
         android:textColor="#ffffff" />
      <androidx.viewpager.widget.ViewPager
         android:id="@+id/viewpager"
         android:layout_width="match_parent"
         android:layout_height="match_parent"/>
   </LinearLayout>
</LinearLayout>
```

Anexo 11: Programación android studio (Instrucciones)

```
package e.jordan.presentacion;

import android.content.Intent;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListAdapter;
import android.widget.ListView;


public class Instrucciones extends AppCompatActivity {
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_instrucciones);
String[] lista ={"CONEXIONES","MAPA DE RUTA","START","SUBIR
REGISTROS"};
   ListView listView =(ListView) findViewById(R.id.listViewHistory);

    ListAdapter myadapter = new
ArrayAdapter<>(this,android.R.layout.simple_list_item_1,lista);
    listView.setAdapter(myadapter);
    listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
       @Override
       public void onItemClick(AdapterView<?> parent, View view, int position,
long id) {
          if(position==0)
          {
                        Intent intent = new
Intent(Instrucciones.this,conexiones1.class);
             startActivity(intent);
          }
          if(position==1)
          {
             Intent intent = new Intent(Instrucciones.this,mapaderuta1.class);
             startActivity(intent);
          }
          if(position==2)
          {
             Intent intent = new Intent(Instrucciones.this,start1.class);
             startActivity(intent);
          }
          if(position==3)
          {
             Intent intent = new Intent(Instrucciones.this,registro1.class);
             startActivity(intent);
          }} });}
```

Anexo 12: Programación android studio (Instrucciones)

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical"
   tools:context=".Instrucciones">


   <ListView
      android:id="@+id/listViewHistory"
      android:layout_width="334dp"
      android:layout_height="588dp"
      android:layout_below="@+id/toolbar"
      android:layout_alignParentStart="true"
      android:layout_alignParentEnd="true"
      android:layout_marginStart="17dp"
      android:layout_marginTop="17dp"
      android:layout_marginEnd="60dp"
      android:layout_marginBottom="16dp" />
<androidx.appcompat.widget.Toolbar
      android:id="@+id/toolbar"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_alignParentStart="true"
      android:layout_alignParentLeft="true"
      android:layout_alignParentTop="true"
      android:layout_marginStart="0dp"
      android:layout_marginLeft="0dp"
      android:layout_marginTop="0dp"
      android:background="@color/colorPrimary"
      app:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar"
      app:title="INSTRUCCIONES" />

   <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      />

</RelativeLayout>
```

Anexo 13: Programación android studio (Start)

```
package e.jordan.presentacion;

import android.graphics.Bitmap;
import android.media.MediaScannerConnection;
import android.net.Uri;
import android.os.Environment;
import com.google.android.material.floatingactionbutton.FloatingActionButton;
import androidx.swiperefreshlayout.widget.SwipeRefreshLayout;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import androidx.appcompat.widget.Toolbar;
import android.text.Editable;
import android.text.TextWatcher;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.webkit.WebChromeClient;
import android.webkit.WebSettings;
import android.webkit.WebView;
import android.webkit.WebViewClient;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.LinearLayout;
import android.widget.TextView;
import android.widget.Toast;

import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.ImageRequest;
import com.android.volley.toolbox.JsonObjectRequest;
import com.android.volley.toolbox.Volley;

import org.json.JSONException;
import org.json.JSONObject;

import java.io.ByteArrayOutputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.sql.Timestamp;
import java.util.Date;

import e.jordan.presentacion.fragments.mapa_de_ruta.ControlFragment;
import e.jordan.presentacion.tools.File;
import e.jordan.presentacion.tools.Javascript;
public class Start extends AppCompatActivity
{
   private Toolbar toolbar;
   private WebView myWebView;
```

```
private EditText editTextDroneUrl;
private EditText editTextRaspberryUrl;
private TextView textViewError;
private SwipeRefreshLayout swipeRefreshLayout;
private RequestQueue requestQueue;

public static final String UP = "UP";
public static final String DOWN = "DOWN";
public static final String LEFT = "LEFT";
public static final String RIGHT = "RIGHT";
public static final String FRONT = "FRONT";
public static final String BACK = "BACK";
public static final String STOP = "STOP";
private String direction = ControlFragment.STOP;
private boolean jump = false;

private Javascript.Interval interval;

private void setupHttpError()
{
   this.textViewError = findViewById(R.id.textViewError);
}
private void onHttpError(String message)
{
   this.textViewError.setText(message);
   this.textViewError.setVisibility(message.equals("") ? View.GONE :
View.VISIBLE);
   }
private String readDataFromFile(String filename)
{
   try {
      return File.read(Start.this, filename);
   }catch(IOException e) {
      return "http://192.168.4.1";
   }
}
private void setupUrlsListeners()
{
this.editTextRaspberryUrl.addTextChangedListener(new TextWatcher() {
   @Override public void beforeTextChanged(CharSequence charSequence, int
i, int i1, int i2) { }
   @Override public void onTextChanged(CharSequence charSequence, int i,
int i1, int i2) {
      try {
         String ip = charSequence.toString();
         File.write(Start.this, "RASPBERRY_IP.txt", ip);
         Start.this.myWebView.loadUrl(ip);
      }catch (Exception e) {
         e.printStackTrace();
      }
   }
```

```
@Override public void afterTextChanged(Editable editable) { }
    });

    this.editTextDroneUrl.addTextChangedListener(new TextWatcher() {
        @Override public void beforeTextChanged(CharSequence charSequence,
int i, int i1, int i2) { }
        @Override public void onTextChanged(CharSequence charSequence, int i,
int i1, int i2) {
            try {
                String ip = charSequence.toString();
                File.write(Start.this, "IP.txt", ip);
            }catch (Exception e) {
                e.printStackTrace();
            }
        }
        @Override public void afterTextChanged(Editable editable) { }
    });
}
private void setupToolbar()
{
    this.toolbar = findViewById(R.id.toolbar);
    this.toolbar.setTitle("Tiempo Real");
    setSupportActionBar(toolbar);
this.linearLayoutEditTexts = this.findViewById(R.id.linearLayoutEditTexts);

    this.editTextDroneUrl = this.findViewById(R.id.editTextDroneUrl);
    String ip = this.readDataFromFile("IP.txt");
    this.editTextDroneUrl.setText(ip);

    this.editTextRaspberryUrl= this.findViewById(R.id.editTextRaspberryUrl);
    String raspberryIp = this.readDataFromFile("RASPBERRY_IP.txt");
    this.editTextRaspberryUrl.setText(raspberryIp );
}
private void setupWebView()
{
    this.myWebView = this.findViewById(R.id.webview);

    this.myWebView.setHorizontalScrollBarEnabled(false);
    this.myWebView.clearCache(true);
    this.myWebView.clearHistory();
    this.myWebView.setWebChromeClient(new WebChromeClient());
    this.myWebView.setWebViewClient(new WebViewClient(){
        @Override
        public void onPageStarted(WebView view, String url, Bitmap favicon) {
            super.onPageStarted(view, url, favicon);

            Start.this.swipeRefreshLayout.setRefreshing(false);
        }
    });
```

```
WebSettings webSettings = this.myWebView.getSettings();
   webSettings.setJavaScriptEnabled(true);
}
private void setupSwipeRefresh()
{
   this.swipeRefreshLayout = this.findViewById(R.id.swipeRefreshLayout);
   this.swipeRefreshLayout.setOnRefreshListener(new
SwipeRefreshLayout.OnRefreshListener() {
      @Override public void onRefresh() {
         String ip = Start.this.editTextRaspberryUrl.getText().toString();
         Start.this.myWebView.loadUrl(ip);
         Start.this.swipeRefreshLayout.setRefreshing(true);
      }
   });
}
private void setupButtons()
{
   FloatingActionButton buttonLeft = this.findViewById(R.id.buttonLeft);
   FloatingActionButton buttonRight = this.findViewById(R.id.buttonRight);
   FloatingActionButton buttonFront = this.findViewById(R.id.buttonFront);
   FloatingActionButton buttonBack = this.findViewById(R.id.buttonBack);
   FloatingActionButton buttonJump = this.findViewById(R.id.buttonJump);
   FloatingActionButton buttonStop = this.findViewById(R.id.buttonStop);
   FloatingActionButton buttonTakePicture =
this.findViewById(R.id.buttonTakePicture);
buttonLeft.setOnClickListener(new View.OnClickListener() {@Override public void
onClick(View view) { leftDrone(view); }});
   buttonRight.setOnClickListener(new View.OnClickListener() {@Override public
void onClick(View view) { rightDrone(view); }});
   buttonFront.setOnClickListener(new View.OnClickListener() {@Override public
void onClick(View view) { frontDrone(view); }});
   buttonBack.setOnClickListener(new View.OnClickListener() {@Override public
void onClick(View view) { backDrone(view); }});
   buttonJump.setOnClickListener(new View.OnClickListener() {@Override public
void onClick(View view) { jumpDrone(view); }});
   buttonStop.setOnClickListener(new View.OnClickListener() {@Override public
void onClick(View view) { stopDrone(view); }});
   buttonTakePicture.setOnClickListener(new View.OnClickListener() {@Override
public void onClick(View view) { takePicture(view); }});
}
private void setupHttpConnection()
{
   this.requestQueue = Volley.newRequestQueue(this);
}
@Override protected void onCreate(Bundle savedInstanceState)
{
   super.onCreate(savedInstanceState);
   setContentView(R.layout.activity_start);
   this.setupToolbar();
   this.setupUrlsListeners();
   this.setupWebView();
   this.setupSwipeRefresh();
   this.setupButtons();
   this.setupHttpError();
   this.setupHttpConnection();
   this.move();
```

```
}
private void sendHttp(String url) {
   Log.e("ControlFragment", url);
   JsonObjectRequest jsonObjectRequest = new
JsonObjectRequest(Request.Method.GET, url, null, new
Response.Listener<JSONObject>()
   {
      @Override public void onResponse(JSONObject response)
      {
         try
         {
            Log.e("ControlFragment", response.toString());

            String state = response.getString("state");
            if(state.equals("OK"))
            {
               String direction = response.getString("direction");
            }

            if(direction.toLowerCase().equals("left") ||
direction.toLowerCase().equals("right"))
            {
               direction = "STOP";
            }

            onHttpError("");
            if(jump)
            {
               jump = false;
            }
         }
         catch (JSONException jsonexcption)
         {
            jsonexcption.printStackTrace();
            onHttpError(jsonexcption.getClass().getCanonicalName());
         }
      }
}, new Response.ErrorListener()
{
   @Override public void onErrorResponse(VolleyError error)
   {
      Log.e("ControlFragment", error.getClass().getCanonicalName());
      error.printStackTrace();
      onHttpError(error.getClass().getCanonicalName());
   }});
jsonObjectRequest.setTag(ControlFragment.VOLLEY_TAG);
   this.requestQueue.add(jsonObjectRequest);
}
private void move()
{
this.interval = Javascript.setInterval(() -> {
      Log.e("ControlFragment", "move");
      String droneIp = this.editTextDroneUrl.getText().toString();
      String url = droneIp + "/move/" + this.direction.toLowerCase() + "?jump="
+ this.jump;
      sendHttp(url);
   }, 1000);}
```

```
public void jumpDrone(View view)
{
   Log.e("ControlFragment", "jumpDrone");
   this.jump = true;
}
public void leftDrone(View view)
{
   Log.e("ControlFragment", "leftDrone");
   this.direction = ControlFragment.LEFT;
}
public void rightDrone(View view)
{
   Log.e("ControlFragment", "rightDrone");
   this.direction = ControlFragment.RIGHT;
}
public void frontDrone(View view)
{
   Log.e("ControlFragment", "frontDrone");
   this.direction = ControlFragment.FRONT;
}
public void backDrone(View view)
{
   Log.e("ControlFragment", "backDrone");
   this.direction = ControlFragment.BACK;
}
public void stopDrone(View view)
{
   Log.e("ControlFragment", "stopDrone");
   this.direction = ControlFragment.STOP;
}
public void takePicture(View view)
{
   Log.e("ControlFragment", "takePicture");

   String raspberryIp = this.editTextRaspberryUrl.getText().toString();
   String url = raspberryIp  + "/picture";
   Log.e("ControlFragment", url);

   ImageRequest imageRequest = new ImageRequest(url, new
Response.Listener<Bitmap>()
   {
      @Override public void onResponse(Bitmap response)
      {
         try
         {
Date date = new Date();
long time = date.getTime();
Timestamp timestamp = new Timestamp(time);
java.io.File root =
Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_PICT
URES);
java.io.File directory = new java.io.File(root + "/Drone");
directory.mkdirs();
```

```
String name = timestamp + ".jpg";
String path = directory + java.io.File.separator + name;
java.io.File file = new java.io.File(path);
if(file.exists())
{
   file.delete();
}

Log.e("START-DEBUG", path);

ByteArrayOutputStream byteArrayOutputStream = new ByteArrayOutputStream();
response.compress(Bitmap.CompressFormat.JPEG, 90, byteArrayOutputStream);
file.createNewFile();

FileOutputStream fileOutputStream = new FileOutputStream(file);
fileOutputStream.write(byteArrayOutputStream.toByteArray());
fileOutputStream.close();

MediaScannerConnection.scanFile(getApplicationContext(), new
String[]{file.toString()}, null,
            new MediaScannerConnection.OnScanCompletedListener() {
              public void onScanCompleted(String path, Uri uri) {
                Log.e("START-DEBUG", "Scanned " + path + ":");
                Log.e("START-DEBUG", "-> uri=" + uri);
              }
            });

        Toast.makeText(getApplicationContext(), "Image \"" + name + "\" saved",
Toast.LENGTH_SHORT).show();
      }
      catch(IOException e)
      {
        Log.e("START-DEBUG", e.getMessage());

        for(StackTraceElement stackTraceElement : e.getStackTrace())
        {
          Log.e("START-DEBUG", stackTraceElement.toString());
        }

        Toast.makeText(getApplicationContext(), "Image can not be saved",
Toast.LENGTH_SHORT).show();
      }
      Log.e("START-DEBUG", "image saved");
   }
}, 600, 600, ImageView.ScaleType.CENTER_CROP, null, error -> {
   Log.e("START-DEBUG", "Error de comunicacion con el servidor");
   Log.e("START-DEBUG", error.toString());
});
imageRequest.setTag(ControlFragment.VOLLEY_TAG);
this.requestQueue.add(imageRequest);
```

```
}

    @Override protected void onStart()
    {
        super.onStart();

        String ip = this.editTextRaspberryUrl.getText().toString();
        this.myWebView.loadUrl(ip);
        this.swipeRefreshLayout.setRefreshing(true);
    }
    @Override public boolean onCreateOptionsMenu(Menu menu)
    {
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
    @Override public boolean onOptionsItemSelected(MenuItem item)
    {
        switch (item.getItemId())
        {
            case R.id.toggleIp:

this.linearLayoutEditTexts.setVisibility(this.linearLayoutEditTexts.getVisibility()==View.GONE?View.VISIBLE:View.GONE);
            return true;
            default: return super.onOptionsItemSelected(item);
        }
    }

    @Override
    protected void onStop() {
        super.onStop();

        Javascript.clearInterval(this.interval);

        Log.e("ControlFragment", "move");
        String droneIp = this.editTextDroneUrl.getText().toString();
        String url = droneIp + "/move/stop?jump=" + this.jump;
        sendHttp(url);
    }
}
```

Anexo 14: Programación android studio (Start)

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:orientation="vertical"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context=".Start">
   <com.google.android.material.appbar.AppBarLayout
      android:id="@+id/appBarLayout"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar">
      <androidx.appcompat.widget.Toolbar
         android:id="@+id/toolbar"
         android:layout_width="match_parent"
         android:layout_height="wrap_content"
         android:background="?attr/colorPrimary"
         app:layout_scrollFlags="scroll|enterAlways"
         app:popupTheme="@style/ThemeOverlay.AppCompat.Light">
         <LinearLayout
            android:id="@+id/linearLayoutEditTexts"
            android:visibility="gone"
            android:orientation="vertical"
            android:layout_width="match_parent"
            android:layout_height="wrap_content">
<EditText
            android:id="@+id/editTextDroneUrl"
            android:textColor="#fff"
            android:text="http://27.7.99.106"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" />
         <EditText
            android:id="@+id/editTextRaspberryUrl"
            android:textColor="#fff"
            android:text="http://27.7.99.106"
            android:layout_width="match_parent"
            android:layout_height="wrap_content" />
      </LinearLayout>
   </androidx.appcompat.widget.Toolbar>
</com.google.android.material.appbar.AppBarLayout>
<TextView
   android:layout_below="@+id/appBarLayout"
   android:id="@+id/textViewError"
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:layout_gravity="end"
   android:background="#ff0000"
   android:padding="10dp"
   android:text="asdasd"
   android:visibility="gone"
   android:textAlignment="textEnd"
   android:textColor="#ffffff" />
```

```
<RelativeLayout
   android:layout_width="match_parent"
   android:layout_height="match_parent">
   <androidx.swiperefreshlayout.widget.SwipeRefreshLayout
      android:id="@+id/swipeRefreshLayout"
      android:layout_width="match_parent"
      android:layout_height="match_parent">
      <WebView
         android:layout_below="@+id/textViewError"
         android:id="@+id/webview"
         android:layout_width="match_parent"
         android:layout_height="match_parent"                                    />
   </androidx.swiperefreshlayout.widget.SwipeRefreshLayout>

   <GridLayout
      android:columnCount="3"
      android:layout_weight="1"
      android:layout_alignParentBottom="true"
      android:layout_alignParentLeft="true"
      android:layout_width="match_parent"
      android:layout_height="wrap_content">
```

Anexo 15: Programación en arduino de la práctica 1

```
//UNIVERSIDAD POLITÉCNICA SALESIANA
//INGENIERÍA ELECTRÓNICA
//PROYECTO TÉCNICO – JORDÁN-ZAMBRANO
//PRÁCTICA #1: CONTROL DE LOS MOTORREDUCTORES POLOLU

//Motor 1
int m11 = 2;
int m12 = 3;
int enableA = 6;

//Motor 2
int m21 = 10;
int m22 = 11;
int enableB = 9;

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  Serial.begin(9600);
  pinMode(m11, OUTPUT);
  pinMode(m12, OUTPUT);
  pinMode(m21, OUTPUT);
  pinMode(m22, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  analogWrite(enableA,100);
  analogWrite(enableB,100);
  digitalWrite(m11, HIGH);   // turn the LED on (HIGH is the voltage level)
  digitalWrite(m12, LOW);   // turn the LED on (HIGH is the voltage level)
  digitalWrite(m21, HIGH);   // turn the LED on (HIGH is the voltage level)
  digitalWrite(m22, LOW);   // turn the LED on (HIGH is the voltage level)
  delay(2000);                      // wait for a second
  digitalWrite(m11, LOW);   // turn the LED on (HIGH is the voltage level)
  digitalWrite(m12, HIGH);   // turn the LED on (HIGH is the voltage level)
  digitalWrite(m21, LOW);    // turn the LED off by making the voltage LOW
  digitalWrite(m22, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(2000);                      // wait for a second
}
```

Anexo 16: Programación en arduino de la práctica 2

```
//UNIVERSIDAD POLITÉCNICA SALESIANA
//INGENIERÍA ELECTRÓNICA
//PROYECTO TÉCNICO - JORDAN-ZAMBRANO
//PRÁCTICA #2 CONTROL DE MOTORES MEDIANTE EL MÓDULO RS232

// Declaración de variables de los motores
//led2 = mi1;
//led1 = mi2;
int mi1 = 2;
int mi2 = 3;
int md1 = 10;
int md2 = 11;
int enableA = 6;
int enableB = 9;
// Estados de los motores
int m1_status = LOW;
int m2_status = LOW;
//Establecer los modos de los motores
void setup() {
   pinMode(mi1, OUTPUT);
   pinMode(mi2, OUTPUT);
   pinMode(md1, OUTPUT);
   pinMode(md2, OUTPUT);
   digitalWrite(mi1, LOW);
   digitalWrite(mi2, LOW);
   digitalWrite(md1, LOW);
   digitalWrite(md2, LOW);
   Serial.begin(9600);
}
void loop() {
  char read_data;
  analogWrite(enableA,150);
  analogWrite(enableB,150);
    if (Serial.available())
      {
        read_data = Serial.read();
        //MOTOR 1 ADELANTE
        if( read_data == '1' && m1_status == LOW)
          {
          digitalWrite(mi1, HIGH);
          digitalWrite(mi2, LOW);
          m1_status = HIGH;
          Serial.println("MOTOR 1 - ESTADO 1 ON");
          }
       //MOTOR 1 ATRAS
       if( read_data == '2' && m1_status == HIGH)
        {
         digitalWrite(mi1, LOW);
         digitalWrite(mi2, HIGH);
         m1_status = LOW;
         Serial.println("MOTOR 1 - ESTADO 2 ON");
        }
```

```
//MOTOR 2 ADELANTE
      if( read_data == '3' && m2_status == LOW)
      {
      digitalWrite(md1, HIGH);
      digitalWrite(md2, LOW);
      m2_status = HIGH;
      Serial.println("MOTOR 2 - ESTADO 1 ON");
      }

      //MOTOR 2 ATRAS
      if( read_data == '4' && m2_status == HIGH)
      {
      digitalWrite(md1, LOW);
      digitalWrite(md2, HIGH);
      m2_status = LOW;
      Serial.println("MOTOR 1 - ESTADO 2 ON");
      }

      //STOP
      if( read_data == '0')
      {
      digitalWrite(mi1, LOW);
      digitalWrite(mi2, LOW);
      digitalWrite(md1, LOW);
      digitalWrite(md2, LOW);
      m1_status = LOW;
      m2_status = LOW;
      Serial.println("STOP");
      }
     }
   delay(10);
}
```

Anexo 17: Programación en arduino de la práctica 3

```
//UNIVERSIDAD POLITÉCNICA SALESIANA
//INGENIERÍA ELECTRÓNICA
//PROYECTO TÉCNICO – JORDAN-ZAMBRANO
//PRÁCTICA #3: OBTENCIÓN Y MUESTRAS DE DATOS DEL SENSOR MPU6050

//GND - GND
//VCC - VCC
//SDA - Pin A4
//SCL  Pin A5


#include "I2Cdev.h"
#include "MPU6050.h"
#include "Wire.h"

//mpu
const int mpuAddress = 0x68;  // Puede ser 0x68 o 0x69
MPU6050 mpu(mpuAddress);

 //Variables para datos de mpu
int ax, ay, az;
int gx, gy, gz;

 // Variable para angulo real de giroscopio
long tiempo_prev;
float dt;
float ang_x, ang_y;
float ang_x_prev, ang_y_prev;
void setup() {
  // put your setup code here, to run once:
   Serial.begin(9600);
   Wire.begin();
   mpu.initialize();
}

void loop() {
  // put your main code here, to run repeatedly:
  // Leer las aceleraciones y velocidades angulares
   mpu.getAcceleration(&ax, &ay, &az);
   mpu.getRotation(&gx, &gy, &gz);

   //tiempo de para obtencion de datos
   dt = (millis() - tiempo_prev) / 1000.0;
   tiempo_prev = millis();
```

```
//Calcular los ángulos con acelerometro
  float accel_ang_x = atan(ay / sqrt(pow(ax, 2) + pow(az, 2)))*(180.0 / 3.14);
  float accel_ang_y = atan(-ax / sqrt(pow(ay, 2) + pow(az, 2)))*(180.0 / 3.14);

  //Calcular angulo de rotación con giroscopio y filtro complementario
  ang_x = 0.98*(ang_x_prev + (gx / 131)*dt) + 0.02*accel_ang_x;
  ang_y = 0.98*(ang_y_prev + (gy / 131)*dt) + 0.02*accel_ang_y;

  ang_x_prev = ang_x;
  ang_y_prev = ang_y;

  Serial.print(F("Rotacion en X:  "));
  Serial.print(ang_x);
  Serial.print(F("\t Rotacion en Y: "));
  Serial.println(ang_y);

}
```

Anexo 18: Programación en arduino de la práctica 4

```
//UNIVERSIDAD POLITÉCNICA SALESIANA
//INGENIERÍA ELECTRÓNICA
//PROYECTO TÉCNICO - JORDÁN-ZAMBRANO
//PRÁCTICA #4 ACCION DE LOS MOTORES MEDIANTE EL USO DEL SENSOR MPU6050
#include "I2Cdev.h"
#include "MPU6050.h"
#include "Wire.h"

//Motores
int mi1 = 3;
int mi2 = 8;
int md1 = 11;
int md2 = 10;
int enableA = 6;     //pwm izquierda
int enableB = 9;     //pwm derecha

//mpu
const int mpuAddress = 0x68;  // Puede ser 0x68 o 0x69
MPU6050 mpu(mpuAddress);

//Variables para datos de mpu
int ax, ay, az;
int gx, gy, gz;

// Variable para angulo real de giroscopio
long tiempo_prev;
float dt;
float ang_x, ang_y;
float ang_x_prev, ang_y_prev;

void sensor1() {
  // put your main code here, to run repeatedly:
  // Leer las aceleraciones y velocidades angulares
  mpu.getAcceleration(&ax, &ay, &az);
  mpu.getRotation(&gx, &gy, &gz);

  //tiempo de para obtencion de dato
  dt = (millis() - tiempo_prev) / 1000.0;
    tiempo_prev = millis();

    //Calcular los ángulos con acelerometro
    float accel_ang_x = atan(ay / sqrt(pow(ax, 2) + pow(az, 2))) * (180.0 / 3.14);
    float accel_ang_y = atan(-ax / sqrt(pow(ay, 2) + pow(az, 2))) * (180.0 / 3.14);

    //Calcular angulo de rotación con giroscopio y filtro complementario
    ang_x = 0.98 * (ang_x_prev + (gx / 131) * dt) + 0.02 * accel_ang_x;
    ang_y = 0.98 * (ang_y_prev + (gy / 131) * dt) + 0.02 * accel_ang_y;

    ang_x_prev = ang_x;
    ang_y_prev = ang_y;
}
```

Anexo 19: Programación en arduino de la práctica 5

```
//UNIVERSIDAD POLITÉCNICA SALESIANA
//INGENIERÍA ELECTRÓNICA
//PROYECTO TÉCNICO - JORDAN-ZAMBRANO
//PRÁCTICA #5 DETECTOR DE OBJETOS MEDIANTE EL SENSOR SHARP CON
ALARMA SONORA
// Declarar las variables
int mi1 = 2;       //Motor izquierda 1
int mi2 = 3;       //Motor izquierda 2
int md1 = 10;      //Motor derecha 1
int md2 = 11;      //Motor derecha 2
int buzzer = 12;
double sharp= A1;
int enableA = 6;     //pwm izquierda
int enableB = 9;     //pwm derecha
void setup() {
  pinMode(mi1,OUTPUT);
  pinMode(mi2,OUTPUT);
  pinMode(md1,OUTPUT);
  pinMode(md2,OUTPUT);
  pinMode(buzzer, OUTPUT);
  Serial.begin(9600);  //Comunicacion serial de 9600 baudios
} void loop() {
  //Leer la entrada analogica A1
   int D_cm=distancia(500);
   if (D_cm <= 30) {
       analogWrite(enableA,150);
       analogWrite(enableB,150);
       digitalWrite(mi1,LOW);
       digitalWrite(mi2,HIGH);
       digitalWrite(md1,LOW);
       digitalWrite(md2,HIGH);
       digitalWrite(buzzer,HIGH);
       Serial.println("IZQUIERDA");
    }else{
       analogWrite(enableA,100);
       analogWrite(enableB,150);
       digitalWrite(buzzer,LOW);
       digitalWrite(mi1,LOW);
       digitalWrite(mi2,HIGH);
       digitalWrite(md1,HIGH);
       digitalWrite(md2,LOW);
       Serial.println("ADELANTE");  }
  Serial.println(D_cm);
  delay(10);}
//Sensor sharp
 float distancia(int n) {
      long suma=0;
      for(int i=0;i<n;i++)
      { suma=suma+analogRead(sharp); }
      float adc=suma/n;
      float distancia_cm = 17569.7 * pow(adc, -1.2062);
      return(distancia_cm);}
```

Anexo 20: Tabla de datos de la función de transferencia

PULSO	VELOCIDAD
0	0
0	0
0	0
0	0
0	0
50	47,6100000000000
50	47,6100000000000
50	47,6100000000000
50	47,6100000000000
50	47,6100000000000
50	47,6100000000000
50	47,6100000000000
50	47,6100000000000
50	47,6100000000000
50	47,6100000000000
50	47,6100000000000
50	47,6100000000000
50	47,6100000000000
50	47,6100000000000
50	47,6100000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000

50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
50	49,1500000000000
100	81,4000000000000
100	81,4000000000000
100	81,4000000000000
100	81,4000000000000
100	81,4000000000000
100	81,4000000000000

100	81,4000000000000
100	81,4000000000000
100	81,4000000000000
100	81,4000000000000
100	79,8700000000000
100	79,8700000000000
100	79,8700000000000
100	79,8700000000000
100	79,8700000000000
100	82,9400000000000
100	82,9400000000000
100	82,9400000000000
100	79,8700000000000
100	79,8700000000000
100	79,8700000000000
100	79,8700000000000
100	79,8700000000000
100	79,8700000000000
100	79,8700000000000
100	79,8700000000000
100	79,8700000000000
100	79,8700000000000
100	79,8700000000000
100	81,4000000000000
100	81,4000000000000
100	81,4000000000000
100	81,4000000000000
100	81,4000000000000
100	81,4000000000000
100	81,4000000000000
100	81,4000000000000
100	82,9400000000000
100	82,9400000000000
100	82,9400000000000
100	82,9400000000000
100	82,9400000000000
100	82,9400000000000
100	82,9400000000000
100	82,9400000000000
100	82,9400000000000
100	82,9400000000000
100	81,4000000000000
100	81,4000000000000
100	81,4000000000000
100	81,4000000000000
100	81,4000000000000

100	81,4000000000000
100	81,4000000000000
100	81,4000000000000
100	81,4000000000000
100	81,4000000000000
100	81,4000000000000
100	81,4000000000000
100	81,4000000000000
100	81,4000000000000
100	81,4000000000000
100	81,4000000000000
100	78,3300000000000
100	78,3300000000000
100	78,3300000000000
100	78,3300000000000
100	78,3300000000000
100	78,3300000000000
100	78,3300000000000
100	78,3300000000000
100	79,8700000000000
100	79,8700000000000
100	79,8700000000000
100	79,8700000000000
100	79,8700000000000
100	78,3300000000000
100	78,3300000000000
100	78,3300000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000

150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	92,1500000000000
150	92,1500000000000
150	92,1500000000000
150	92,1500000000000
150	92,1500000000000
150	95,2300000000000
150	95,2300000000000
150	95,2300000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	89,0800000000000
150	90,6200000000000
150	90,6200000000000
150	90,6200000000000
150	90,6200000000000
150	90,6200000000000
150	90,6200000000000
150	90,6200000000000
150	87,5500000000000

150	87,5500000000000
150	87,5500000000000
150	87,5500000000000
150	87,5500000000000
150	87,5500000000000
150	87,5500000000000
150	87,5500000000000
150	87,5500000000000
150	87,5500000000000
150	87,5500000000000
150	87,5500000000000
150	87,5500000000000
150	87,5500000000000
150	87,5500000000000
150	87,5500000000000
200	95,2300000000000
200	95,2300000000000
200	95,2300000000000
200	95,2300000000000
200	95,2300000000000
200	96,7600000000000
200	96,7600000000000
200	96,7600000000000
200	96,7600000000000
200	96,7600000000000
200	99,8300000000000
200	99,8300000000000
200	99,8300000000000
200	99,8300000000000
200	99,8300000000000
200	98,3000000000000
200	93,6900000000000
200	93,6900000000000
200	93,6900000000000
200	93,6900000000000
200	93,6900000000000
200	93,6900000000000
200	93,6900000000000
200	95,2300000000000
200	95,2300000000000
200	95,2300000000000
200	95,2300000000000
200	95,2300000000000
200	96,7600000000000
200	96,7600000000000

200	96,7600000000000
200	93,6900000000000
200	93,6900000000000
200	93,6900000000000
200	93,6900000000000
200	93,6900000000000
200	93,6900000000000
200	93,6900000000000
200	93,6900000000000
200	93,6900000000000
200	93,6900000000000
200	93,6900000000000
200	93,6900000000000
200	93,6900000000000
200	93,6900000000000
200	93,6900000000000
200	95,2300000000000
200	95,2300000000000
200	95,2300000000000
200	95,2300000000000
200	95,2300000000000
200	95,2300000000000
200	95,2300000000000
200	95,2300000000000
200	95,2300000000000
200	95,2300000000000
200	95,2300000000000
200	95,2300000000000
200	95,2300000000000
200	95,2300000000000
200	95,2300000000000
200	95,2300000000000
200	95,2300000000000
200	95,2300000000000
200	96,7600000000000

Tabla 9: Visualización de datos de la función de transferencia.

Anexo 21: Proceso de ensamblaje del prototipo

Figura 142: Capas del robot.

Figura 143: Estructura del prototipo.

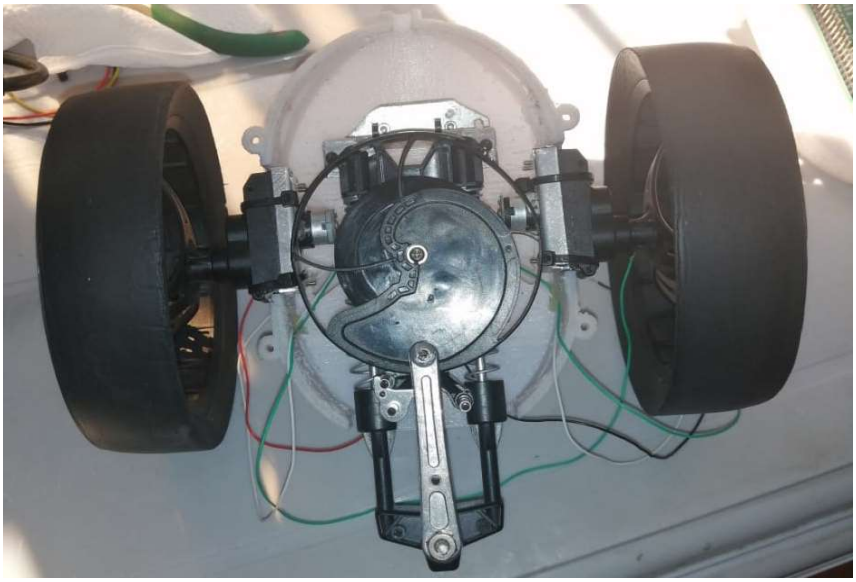


Figura 144: Mecanismo del robot.

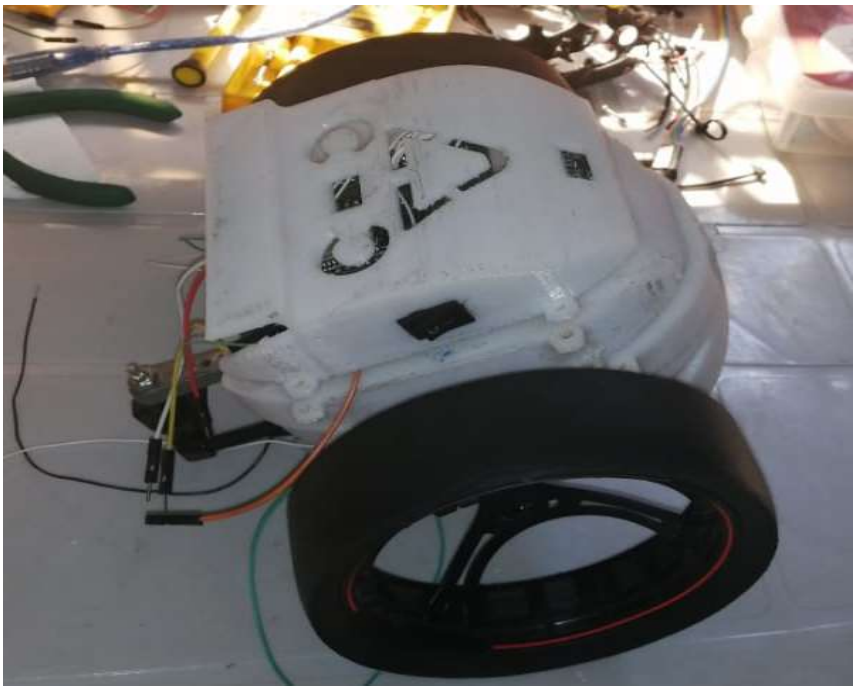


Figura 145: Drone terrestre.

Anexo 22: Cronograma del proyecto de titulación

ACTIVIDAD	MES 1				MES 2				MES 3				MES 4				MES 5				MES 6			
	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	S19	S20	S21	S22	S23	S24
Requerimientos Tecnológicos del Proyecto																								
Capacitación para crear App para Android																								
Diseño y Esquema Físico del drone terrestre																								
Diseño de Programación																								
Realizar Calculos Matematicos para el Diseño																								
Pruebas y Evaluación del drone terrestre																								
Realizar Prácticas Didácticas																								

Tabla 10: Cronograma del proyecto

Anexo 23: Presupuesto del proyecto de titulación

PRESUPUESTO DEL PROYECTO			
PRODUCTO	**CANTIDAD**	**PRECIO UNITARIO**	**PRECIO TOTAL**
Módulo de comunicación Wifi ESP32	2	$ 15,00	$ 30,00
Cámara Raspberry PI3	1	$ 22,00	$ 22,00
Módulo MPU6050	1	$ 3,50	$ 3,50
Buzzer Transductor Electroacústico	1	$ 2,00	$ 2,00
RS232 + Adaptador	1	$ 14,00	$ 14,00
Baterías de Polímero de Litio	1	$ 30,00	$ 30,00
Dispositivo Android	1	$ 100,00	$ 100,00
Motor Pololu	2	$ 15,00	$ 30,00
Diseño de Estructura del Drone	1	$ 200,00	$ 200,00
Diseño Tarjeta Electrónica	1	$ 150,00	$ 150,00
Sensores (Acelerómetro, Giroscopio)	1	$ 10,00	$ 10,00
Sensor Sharp	1	$ 12,50	$ 12,50
Multímetro	1	$ 5,50	$ 5,50
Motor Drive	1	$ 70,00	$ 70,00
Cargador de Batería	1	$ 18,20	$ 18,20
Tornillos + Brocas	25	$ 1,00	$ 25,00
Diseño Autocad	8	$ 5,00	$ 40,00
		TOTAL BRUTO	$ 762,70
		IVA 12%	$ 91,52
		TOTAL	$ 854,22

.

Tabla 11: Presupuesto del proyecto