

## CARRERA DE ESPECIALIZACIÓN EN SISTEMAS EMBEBIDOS

MEMORIA DEL TRABAJO FINAL

### Robot de exploración ambiental

**Autor:**  
**Ing. Gonzalo Carreño**

Director:  
Esp. Ing. Sergio Alberino (UTN.BA)

Jurados:  
Dr. Ing. Pablo Gonzalez (FIUBA)  
Mg. Bioing. Eduardo Filomena (UNER)  
Ing. Juan Manuel Cruz (FIUBA)

*Este trabajo fue realizado en la Ciudad Autónoma de Buenos Aires,  
entre marzo de 2023 y octubre de 2024.*



## *Resumen*

El presente trabajo describe un emprendimiento personal en el que se desarrolla un dispositivo robótico de exploración ambiental, controlable a distancia con las funciones básicas de desplazamiento, medición y reporte de parámetros ambientales tales como presión, temperatura, humedad y luminosidad.

El sistema cuenta con una arquitectura base robusta y flexible sobre la cual otros sensores y actuadores pueden ser adaptados para poder interactuar con el medio ambiente, y logra por tanto brindar una solución que ayuda a incrementar la oferta de robots exploradores en la industria argentina pudiendo ser utilizados en casos de uso de IoT, como por ejemplo en aplicaciones de explotación de petróleo, Smart Mining, Smart Farming, etc.

Para su implementación se utilizaron conceptos y herramientas tales como buenas prácticas en el diseño y desarrollo de firmware, la utilización de sistemas operativos de tiempo real como plataforma de ejecución base, protocolos de comunicaciones para sistemas embebidos, y técnicas y frameworks de testing para asegurar la calidad del producto final.



## *Agradecimientos*

Esta sección es para agradecimientos personales y es totalmente **OPCIONAL**.



# Índice general

<b>Resumen</b>	<b>I</b>
<b>1. Introducción general</b>	<b>1</b>
1.1. Motivación . . . . .	1
1.2. Estado del arte . . . . .	1
1.3. Alcance y objetivos . . . . .	2
1.4. Requerimientos . . . . .	2
<b>2. Introducción específica</b>	<b>5</b>
2.1. Tecnologías de hardware utilizadas . . . . .	5
2.1.1. Espressif ESP32 . . . . .	5
2.1.2. Sensor de temperatura y humedad DHT11 . . . . .	5
2.1.3. Sensor de presión BMP280 . . . . .	6
2.1.4. Fotorresistor como sensor de luminosidad . . . . .	6
2.1.5. Joystick analógico . . . . .	6
2.1.6. Display LCM1602A . . . . .	7
2.1.7. Motores de corriente continua . . . . .	7
2.1.8. Módulos L298N . . . . .	7
2.1.9. Pilas de Li-Ion 3,7 V . . . . .	8
2.1.10. Baterías AA 1,5 V . . . . .	8
2.1.11. Plaquetas genéricas PCB . . . . .	9
2.1.12. Cables de conexión DuPont . . . . .	9
2.1.13. Board Pin Headers para montaje de componentes y cables . .	10
2.1.14. Interruptores de On-Off . . . . .	10
2.1.15. Portapilas . . . . .	10
2.1.16. Ruedas . . . . .	11
2.1.17. Anemómetro digital . . . . .	11
2.2. Tecnologías de software utilizadas . . . . .	12
2.2.1. Marco de trabajo ESP-IDF . . . . .	12
2.2.2. Plataforma Docker . . . . .	12
2.2.3. Testing unitario . . . . .	13
2.2.4. Plataforma de CI/CD . . . . .	13
2.2.5. Visual Studio Code . . . . .	14
2.2.6. Sistema operativo Ubuntu . . . . .	14
<b>3. Diseño e implementación</b>	<b>15</b>
3.1. Arquitectura de software del sistema . . . . .	15
3.1.1. Arquitectura y diseño de componentes en la versión v1.0 . .	15
3.1.2. Arquitectura y diseño de componentes en la versión v2.0 . .	17
3.2. Implementación de los módulos . . . . .	18
3.2.1. Control de la red Wi-Fi . . . . .	19
3.2.2. Control del joystick analógico . . . . .	19

3.2.3. Medición de valor de luminosidad . . . . .	20
3.2.4. Medición de temperatura y humedad . . . . .	21
3.2.5. Medición de presión . . . . .	21
3.2.6. Control de motores DC . . . . .	22
3.2.7. Control del display . . . . .	23
3.3. Arquitectura de hardware . . . . .	24
3.3.1. Ensamblado final del producto v2.0 . . . . .	24
3.3.2. Conexionado lógico . . . . .	25
3.3.3. Conexionado físico . . . . .	26
3.4. Plataforma de desarrollo y ciclo de CI/CD . . . . .	27
3.5. Reportes de ejecución y cobertura de testing unitario . . . . .	31
<b>4. Ensayos y resultados</b>	<b>33</b>
4.1. Proceso de desarrollo y aseguramiento de calidad . . . . .	33
4.2. Verificación técnica de los diferentes módulos . . . . .	33
4.2.1. Verificación del módulo de joystick . . . . .	33
4.2.2. Verificación del módulo de control del display . . . . .	34
4.2.3. Verificación del módulo de control de motores . . . . .	34
4.2.4. Verificación del módulo de medición de temperatura y humedad . . . . .	34
4.2.5. Verificación del módulo de medición de presión atmosférica . . . . .	34
4.2.6. Verificación del módulo de medición de luminosidad . . . . .	34
4.2.7. Verificación del módulo de comunicación UTP sobre Wi-Fi . . . . .	34
4.3. Pruebas funcionales y validación del producto . . . . .	34
4.3.1. Prueba y validación del módulo de visualización de display . . . . .	35
4.3.2. Prueba y validación del módulo de medición de temperatura y humedad . . . . .	35
4.3.3. Prueba y validación del módulo de medición de presión atmosférica . . . . .	37
4.3.4. Prueba y validación del módulo de medición de luminosidad ambiental . . . . .	38
4.3.5. Prueba y validación del control y desplazamiento del robot . . . . .	39
4.4. Videos del producto durante el ensamblado y experimentación . . . . .	39
4.4.1. Videos demostrativos del producto final . . . . .	40
4.4.2. Videos durante el prototipado y ensamblado del robot . . . . .	40
4.5. Documentación del producto . . . . .	40
<b>5. Conclusiones</b>	<b>41</b>
5.1. Próximos pasos . . . . .	41
<b>Bibliografía</b>	<b>43</b>

# Índice de figuras

2.1.	Microcontrolador ESP32-WROOM-32D . . . . .	5
2.2.	Sensor DHT11 . . . . .	6
2.3.	Sensor BMP280 . . . . .	6
2.4.	Fotorresistor . . . . .	6
2.5.	Joystick analógico . . . . .	7
2.6.	Display LCM1602A . . . . .	7
2.7.	Motor de corriente continua . . . . .	7
2.8.	Interruptores de On-Off . . . . .	8
2.9.	Baterías Li-Ion . . . . .	8
2.10.	Baterías AA . . . . .	9
2.11.	Plaquetas genéricas . . . . .	9
2.12.	Cables DuPont . . . . .	9
2.13.	Pines . . . . .	10
2.14.	Interruptores de On-Off . . . . .	10
2.15.	Portapilas . . . . .	11
2.16.	Ruedas . . . . .	11
2.17.	Anemómetro digital AOPUTTRIVER AP-007-WM . . . . .	12
2.18.	Proceso de desarrollo utilizando ESP-IDF <sup>1</sup> . . . . .	13
3.1.	Arquitectura global . . . . .	16
3.2.	Arquitectura global . . . . .	17
3.3.	Conexionado joystick . . . . .	20
3.4.	Conexionado fotorresistor . . . . .	21
3.5.	Círculo del conexionado DHT11 . . . . .	21
3.6.	Conexionado BMP280 . . . . .	22
3.7.	Conexionado motores . . . . .	23
3.8.	Conexionado display . . . . .	23
3.9.	Hardware del robot . . . . .	24
3.10.	Detalles del hardware del robot . . . . .	24
3.11.	Detalles del hardware del joystick . . . . .	25
3.12.	Conexionado del robot . . . . .	25
3.13.	Conexionado del joystick . . . . .	26
3.14.	Conexionado físico del robot . . . . .	26
3.15.	Ejecución de tests por consola . . . . .	28
3.16.	Listado de <i>commits</i> en Github . . . . .	29
3.17.	Listado de <i>builds</i> en Google CloudBuild . . . . .	30
3.18.	Listado de versiones de imágenes doker en Google ArtifactRegistry . . . . .	31
3.19.	Reportes de testing por consola . . . . .	32
3.20.	Reportes de testing web . . . . .	32
4.1.	Visualización del display en la oscuridad . . . . .	35
4.2.	Medición de humedad en el interior . . . . .	36
4.3.	Medición de temperatura en el interior . . . . .	36

4.4. Medición de humedad en el exterior. . . . .	36
4.5. Medición de temperatura en el exterior. . . . .	37
4.6. Medición de presión atmosférica en el interior. . . . .	37
4.7. Medición de presión atmosférica en el exterior. . . . .	38
4.8. Medición de luminosidad ambiental en el exterior durante el día. .	38
4.9. Medición de luminosidad ambiental en interiores durante el día. .	39
4.10. Medición de luminosidad ambiental en interiores durante la noche. .	39

# Índice de tablas

3.1. Configuración de AP WiFi . . . . .	19
3.2. Conexionado joystick . . . . .	19
3.3. Conexionado fotoresistor . . . . .	20
4.1. Resultados de mediciones de temperatura y humedad . . . . .	36
4.2. Resultados de mediciones de presión ambiental. . . . .	37



*Dedicado a mi familia y mis amigos.*



# Capítulo 1

## Introducción general

Esta sección presenta la motivación, alcance, objetivos y requerimientos del producto en el marco del estado del arte y su importancia en la industria.

### 1.1. Motivación

La motivación del presente trabajo fue primeramente volcar y unificar en un emprendimiento personal los conceptos aprendidos en la especialización de Sistemas Embebidos, con una arquitectura robusta que pueda ser extrapolada a otros casos de uso de valor en la industria como por ejemplo la exploración de suelos en el agro, la exploración submarina para la perforación de pozos de petróleo, o los mencionados más adelante en el estado del arte. Por otra parte, se buscó desarrollar un producto que pueda contribuir a aumentar la oferta de dispositivos robóticos exploradores en Argentina.

### 1.2. Estado del arte

Los robots exploradores son dispositivos robotizados capaces de moverse de forma autónoma, y/o controlados a distancia, que han sido creados con el fin de reconocer y explorar un lugar o entorno donde una persona no pueda o deba acceder ya sea por motivos de capacidad, practicidad o seguridad. Por este motivo, en función de las necesidades de desplazamiento, existen diferentes sistemas de motricidad, como son por ejemplo, los bípedos, cuadrúpedos, con ruedas, tracción oruga, acuáticos/sumergibles, aéreos, etc. En cuanto a la forma de control, los hay manejados por control remoto cableado o inalámbrico, habiendo equipos más sofisticados, que gracias a aplicaciones de Inteligencia Artificial, están preparados para desplazarse y tomar decisiones de forma autónoma. Algunos de los tipos de robots exploradores más conocidos son los espaciales, de minas, de rescate en catástrofes, de tuberías, submarinos, y de suelos.

Tanto en el ámbito académico como en la industria existen trabajos, proyectos, e implementaciones comerciales similares al presente trabajo, como por ejemplo:

- El prototipo robótico de exploración minera publicado en varios artículos [1], [2], e impulsado por el Instituto de Automática de la Facultad de Ingeniería de la Universidad Nacional de San Juan en el marco de un convenio con la Comisión Nacional de Energía Atómica y el Gobierno argentino [3].
- El robot de exploración terrestre denominado Geobot [4] desarrollado por los ingenieros Nelson Dario García Hurtado y Melvin Andrés González

Pino, de la universidad de Pamplona, capaz de realizar reconocimiento de zonas y manipulación de muestras de manera autónoma o asistida.

- El robot minero MIN-SIS 1.0 SDG-STR [5] desarrollado por los ingenieros Hernán L. Helguero Velásquez y Rubén Medinaceli Tórrez de la Universidad Técnica de Oruro, capaz de detectar gases, almacenar datos locales y enviar video e imágenes al puesto de mando.
- Spot [6], desarrollado por Boston Dynamics, un robot explorador cuadrupeado de propósito general capaz de explorar, almacenar y enviar información en tiempo real.
- BIKE [7], desarrollado por Waygate Technologies, un robot con ruedas magnéticas, muy utilizado en la industria de petróleo y gas entre otras, capaz de desplazarse por el interior de tuberías para poder realizar inspecciones y comunicar hallazgos.

### 1.3. Alcance y objetivos

A continuación se detallan las funcionalidades incluidas en el alcance del trabajo.

- Sistema de desplazamiento terrestre.
- Operaciones de exploración
  - Medición de humedad ambiental.
  - Medición de temperatura ambiental.
  - Medición de presión ambiental.
  - Medición de luminosidad ambiental.
- Visualización de estado de exploración (lecturas de los sensores).
- Sistema de control por medio de un joystick cableado.

Queda fuera del alcance:

- Locomoción por cualquier otro medio que no sea terrestre.
- Cualquier otra función no contemplada en este alcance.

### 1.4. Requerimientos

A continuación se listan los requerimientos del producto:

1. Requerimientos funcionales
  - a) El sistema debe contar con funciones de desplazamiento para poder moverse hacia adelante y atrás, y poder girar radialmente un ángulo de 360 grados.
  - b) El sistema debe ser capaz de realizar las siguientes operaciones de exploración:
    - Medición de humedad ambiental.
    - Medición de temperatura ambiental.

- Medición de luminosidad ambiental.
  - Medición de presión ambiental.
- c) El sistema debe poder ser controlado a distancia mediante un joystick para que el dispositivo pueda realizar sus movimientos. En caso de que alguna de sus operaciones de exploración requiera algún otro mecanismo de control, el mismo también será integrado en el joystick.
- d) El sistema debe proveer un mecanismo de visualización de las operaciones de exploración al usuario que controla el dispositivo para poder ver el estado y lectura de las operaciones de exploración.
- La interfaz de usuario debe permitir visualizar las lecturas de cada uno de los sensores.
  - Debe haber una pequeña leyenda de la magnitud que se está midiendo y la unidad utilizada junto con el valor.

## 2. Requerimientos no funcionales

- a) La arquitectura del producto debe ser robusta y tolerante a fallas.
- b) A fin de maximizar la mantenibilidad, la arquitectura del producto debe estar modularizada para permitir que los diferentes módulos puedan ser integrados y orquestados separadamente.



## Capítulo 2

# Introducción específica

Esta sección presenta una breve introducción técnica a las herramientas hardware y software utilizadas en el trabajo.

### 2.1. Tecnologías de hardware utilizadas

#### 2.1.1. Espressif ESP32

ESP32 [8] es una serie de microcontroladores embebidos en un chip con Wi-Fi y Bluetooth integrados, de bajo costo y consumo, desarrollado por *Espressif Systems*. Emplea dos cores Xtensa® 32-bit LX6 CPU, incluye interruptores de antena, amplificador de potencia, amplificador de recepción de bajo ruido, un co-procesador ULP (*Ultra Low Power*), módulos de administración de energía y varios periféricos. En la siguiente imagen (2.1) se puede apreciar la placa ESP32-WROOM-32D [9] utilizada para el desarrollo del presente trabajo.



FIGURA 2.1. Microcontrolador ESP32-WROOM-32D.

#### 2.1.2. Sensor de temperatura y humedad DHT11

El DHT11 [10] es un sensor digital de temperatura y humedad relativa de bajo costo y fácil uso. Integra un sensor capacitivo de humedad y un termistor para medir el aire circundante, y muestra los datos mediante una señal digital en el pin de datos (no posee salida analógica). Entre otras aplicaciones se lo suele utilizar principalmente en aplicaciones relacionadas al control automático de temperatura, aire acondicionado y monitoreo ambiental en agricultura. En la figura 2.2 se puede apreciar una imagen del componente.



FIGURA 2.2. Sensor DHT11.

### 2.1.3. Sensor de presión BMP280

El BMP280 [11] es un sensor de presión barométrica absoluta, especialmente factible para aplicaciones móviles que puede ser utilizado con I2C o SPI. Permite alta precisión y linealidad, estabilidad a largo plazo, alta robustez a un muy bajo consumo. En la figura 2.3 se puede apreciar una imagen del componente.



FIGURA 2.3. Sensor BMP280.

### 2.1.4. Fotorresistor como sensor de luminosidad

El fotorresistor es una resistencia eléctrica que varía su valor en función de la cantidad de luz que incide sobre su superficie. Cuando el fotorresistor no está expuesto a radiaciones luminosas, los electrones están firmemente unidos en los átomos que lo conforman, por lo que alcanza su máxima resistencia eléctrica, y cuando sobre él inciden radiaciones luminosas, esta energía libera los electrones con lo cual el material se vuelve más conductor, y se disminuye su resistencia. En la figura 2.4 se puede apreciar una imagen del componente.



FIGURA 2.4. Fotorresistor.

### 2.1.5. Joystick analógico

El módulo de joystick analógico [12] está construido sobre el montaje de dos potenciómetros en un ángulo de 90 grados. Los potenciómetros están conectados a una palanca corta centrada por resortes. Este módulo produce una salida de alrededor de 2,5 V cuando la palanca se encuentra en reposo (en el centro), mientras que al desplazarse hará que la salida varíe de 0 a 5 V dependiendo de su posición. La obtención de los valores en Volts se obtiene tras convertir las lecturas en niveles lógicos mediante el módulo ADC (conversor analógico digital) [13]

del microcontrolador ESP32. En la figura 2.5 se puede apreciar una imagen del componente.



FIGURA 2.5. Joystick analógico.

#### 2.1.6. Display LCM1602A

El display LCM1602A [14] consta de una pantalla de cristal líquido que permite representar dos filas con hasta 16 caracteres alfanuméricos en cada una y dado que se encuentra integrada a una interfaz adaptadora I2C puede ser controlada por este protocolo. En la figura 2.6 se puede apreciar una imagen del componente.



FIGURA 2.6. Display LCM1602A.

#### 2.1.7. Motores de corriente continua

El motor DC (corriente continua) [15] es un electromotor que transforma energía eléctrica en energía mecánica. Estos motores operan con una tensión entre 3 y 6 V, corriente de 150 mA, permiten una velocidad de entre 90 y 200 RPM y un torque de entre 0,15 Nm y 0,60 Nm. En la figura 2.7 se puede apreciar una imagen del componente.



FIGURA 2.7. Motor de corriente continua.

#### 2.1.8. Módulos L298N

Los módulos L298N [16] son controladores de motores de puente H duales muy utilizados en proyectos de robótica y automoción para controlar motores de corriente continua (DC) y motores paso a paso. El módulo se basa en el IC L298N, que es un controlador de motor de doble puente H fabricado por STMicroelectronics. Este módulo posee dos puentes H que permiten controlar 2 motores DC o un motor paso a paso bipolar/unipolar. El módulo permite controlar el sentido de giro y velocidad mediante señales TTL (*transistor-transistor logic*) que se pueden obtener de microcontroladores. Tiene integrado un regulador de tensión de

5 V encargado de alimentar la parte lógica del L298N cuya configuración se hace a través de un Jumper y se puede usar para alimentar el microcontrolador.

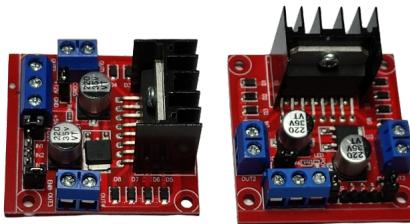


FIGURA 2.8. Interruptores de On-Off.

### 2.1.9. Pilas de Li-Ion 3,7 V

Las pilas 18650 de 3000 mAh [17] son baterías recargables que utilizan tecnología de iones de litio y se emplean comúnmente en una variedad de dispositivos electrónicos, como por ejemplo linternas de alta potencia, laptops, vehículos eléctricos (como bicicletas y scooters), etc. Las baterías de iones de litio son conocidas por su alta densidad de energía, su baja tasa de autodescarga y su larga vida útil en comparación con otras tecnologías de baterías recargables. Con 3000 mAh (miliamperios-hora), estas baterías ofrecen una duración prolongada, lo que es ideal para dispositivos que requieren mucha energía. Al ser recargables, son más ecológicas y económicas a largo plazo en comparación con las baterías desechables. Además, son generalmente seguras y tienen una buena estabilidad térmica, reduciendo el riesgo de explosiones o incendios.



FIGURA 2.9. Baterias Li-Ion.

### 2.1.10. Baterías AA 1,5 V

Las pilas AA de 1,5 V con 2600 mWh [18] son baterías de tamaño AA que proporcionan una tensión de 1,5 V y tienen una capacidad energética de 2600 mWh. Ofrecen alta capacidad pudiendo durar más tiempo entre recargas o reemplazos en comparación con las baterías de menor capacidad, y versatilidad, ya que al ser AA son compatibles con la mayoría de los dispositivos electrónicos que requieren alimentación de bajo voltaje. Las pilas AA de 1,5 V con una alta capacidad como 2600 mWh son ideales para dispositivos que requieren un mayor consumo de energía, tales como cámaras digitales, linternas, juguetes electrónicos, mandos a distancia, controles de videojuegos, etc.



FIGURA 2.10. Baterías AA.

### 2.1.11. Plaquetas genéricas PCB

Las plaquetas electrónicas genéricas del tipo PCB (*Printed Circuit Board*), también conocidas como placas de desarrollo, son herramientas utilizadas principalmente en la educación, el diseño y el prototipado de circuitos electrónicos que constan de material aislante con patrones de cobre en una o ambas caras, donde se pueden soldar componentes para crear circuitos permanentes. Permiten el montaje de circuitos electrónicos de manera más duradera y fiable que el uso de un Protoboard y son asequibles de forma online a un bajo costo.

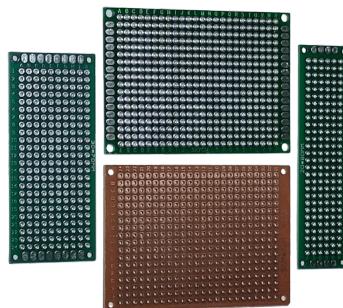


FIGURA 2.11. Plaquetas genéricas.

### 2.1.12. Cables de conexión DuPont

Los cables Dupont son cables de conexión eléctrica utilizados comúnmente en prototipado y proyectos de electrónica. Estos cables son muy populares en la comunidad de los entusiastas y estudiantes de electrónica debido a su facilidad de uso y versatilidad. Están hechos de hilos de cobre con una cubierta de plástico aislante. Los conectores suelen ser de metal chapado para asegurar una buena conductividad. Los hay del tipo Hembra-Hembra, Hembra-Macho y Macho-Macho. Entre sus usos más comunes se encuentran la conexión de dispositivos en Protoboard sin necesidad de soldadura y la interconexión entre sensores, actuadores y terminales en pines o bahías de plaquetas impresas.



FIGURA 2.12. Cables DuPont.

### 2.1.13. Board Pin Headers para montaje de componentes y cables

Los pines de cabecera son componentes electrónicos que consisten en filas de pines metálicos montados en una base plástica. Estos pines se utilizan para realizar conexiones eléctricas entre diferentes componentes y placas en proyectos electrónicos. La cantidad de pines puede variar en número, desde unos pocos pines hasta decenas, dependiendo de las necesidades del proyecto. Pueden encontrarse en varias configuraciones, como ser *Single Row* (una sola fila de pines), *Double Row* (dos filas de pines), que aumentan la capacidad de conexión en un espacio compacto, y *Right Angle* (pines que se extienden en ángulo recto respecto a la base, útiles para conexiones horizontales).



FIGURA 2.13. Pines.

### 2.1.14. Interruptores de On-Off

Los interruptores de encendido-apagado (on-off) son componentes fundamentales en circuitos electrónicos que permiten controlar el flujo de corriente eléctrica. Son utilizados para conectar o desconectar la corriente en un circuito, funcionando como un medio sencillo y efectivo para gestionar la energía de dispositivos electrónicos. Se los puede encontrar en diferentes tipos, como por ejemplo, *Rocker Switches* (interruptores basculantes que se operan basculando de una posición a otra), *Slide Switches* (interruptores deslizantes que se operan deslizando un pequeño botón a lo largo de una pista), *Toggle Switches* (interruptores de palanca que utilizan una palanca que se mueve hacia arriba y hacia abajo para cambiar entre las posiciones on y off) y *Push-Button Switches* (interruptores de pulsador que al presionar el botón se conecta o desconecta el circuito).



FIGURA 2.14. Interruptores de On-Off.

### 2.1.15. Portapilas

Los portapilas son componentes diseñados para alojar y conectar baterías en un circuito electrónico. Estos dispositivos, generalmente hechos de plástico resistente con contactos metálicos (normalmente de níquel o acero) para asegurar una buena conductividad, facilitan la instalación y el reemplazo de baterías, asegurando una conexión segura y confiable. Se los puede encontrar comúnmente para los tipos de pilas y baterías más utilizadas, como por ejemplo AA, AAA, 9 V,

18650, CR2032, etc. Además, suelen venir en tres posibles configuraciones, un solo compartimento (para una sola batería), múltiples compartimentos (para conectar varias baterías en serie o en paralelo), y con interruptor (algunos portapilas incluyen un interruptor para encender o apagar la conexión de las baterías al circuito).



FIGURA 2.15. Portapilas.

### 2.1.16. Ruedas

Las ruedas de plástico son componentes genéricos utilizados en la comunidad hobbista y estudiantil de electrónica para el armado de sistemas con desplazamiento. Se las puede comprar online a un bajo costo, usualmente vienen en kit de dos o cuatro unidades, y son compatibles con los motores de DC utilizados para moverlas.



FIGURA 2.16. Ruedas.

### 2.1.17. Anemómetro digital

Para la validación de los parámetros ambientales medidos por el robot se utilizó un dispositivo anemómetro digital AOPUTTRIVER AP-007-WM [19] capaz de medir presión, temperatura y humedad ambiental, y de esta manera poder comparar los valores medidos. En la siguiente figura puede apreciarse una imagen del mismo.



FIGURA 2.17. Anemómetro digital AOPUTTRIVER AP-007-WM.

## 2.2. Tecnologías de software utilizadas

### 2.2.1. Marco de trabajo ESP-IDF

*Espressif Systems* proporciona recursos básicos de hardware y software para ayudar a los desarrolladores de aplicaciones a realizar sus ideas utilizando el hardware de la serie ESP32. El framework de software de Espressif está destinado al desarrollo de aplicaciones de IoT (Internet de las cosas) con Wi-Fi, Bluetooth, administración de energía y varias otras características del sistema. Sus componentes son:

1. Toolchain, utilizado para compilar el código para ESP32.
2. Build tools, que provee utilidades como CMake [20] y Ninja [21] para construir la aplicación completa para ESP32.
3. ESP-IDF [22], que brinda la API de desarrollo para ESP32 y scripts para ejecutar Toolchain.

Además de las herramientas mencionadas se utilizó el conjunto de bibliotecas y drivers provistos por el proyecto ESP-IDF-Lib [23] basados en el framework ESP-IDF.

En la figura 2.18 se puede apreciar una imagen del proceso de desarrollo y despliegue usando el framework ESP-IDF.

### 2.2.2. Plataforma Docker

Docker [25] es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software, proporcionando una capa adicional de abstracción y automatización de virtualización de aplicaciones en múltiples sistemas operativos. Docker utiliza características de aislamiento de recursos del kernel Linux, tales como cgroups y espacios de nombres (namespaces) para

---

<sup>1</sup>Imagen tomada de [24]

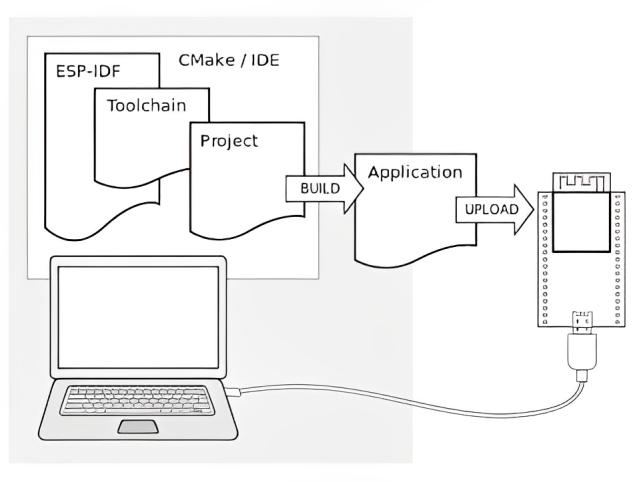


FIGURA 2.18. Proceso de desarrollo utilizando ESP-IDF<sup>1</sup>.

permitir que contenedores livianos independientes se ejecuten en paralelo de manera aislada evitando la sobrecarga de iniciar y mantener máquinas virtuales.

### 2.2.3. Testing unitario

Con el fin de maximizar la calidad durante el proceso de desarrollo del producto se implementaron test unitarios para todos los servicios del robot y del joystick. El conjunto de herramientas utilizadas para tal fin fueron:

- Ceedling [26]: herramienta de orquestación de tests unitarios, inyección de objetos mocks.
- CMock [27]: framework de mock objects para sistemas embebidos.
- Unity [28]: framework de unit testing para sistemas embebidos.
- Gcov [29]: plugin the ceedling para evaluar y reportar la cobertura.

En los capítulos siguientes se describe la configuración de dichas herramientas y se presentan los resultados tras su ejecución.

### 2.2.4. Plataforma de CI/CD

Durante el proceso de desarrollo del producto se utilizó CI/CD (*continuous integration / continuous delivery*) mediante la integración de las siguientes herramientas:

- Github [30]: servicio de repositorio y control de versiones de código fuente.
- Google Cloud Build [31]: servicio de compilación, empaquetado y ejecución *builds*.
- Google Artifact Registry [32]: servicio de repositorio y control de versiones de imágenes Docker.

El objetivo de esta configuración de servicios es permitir que por cada cambio en el código fuente versionado en el controlador de versiones Github, se dispare un proceso de compilación y ejecución de tests unitarios notificando en tiempo real si dicho cambio agrega o no una falla al actual estado del desarrollo. En caso

de pasar satisfactoriamente la compilación y ejecución de los tests entonces se genera una nueva imagen docker con la última versión del código compilado y se versiona en Artifact Registry.

### **2.2.5. Visual Studio Code**

Visual Studio Code [33] es un editor de código fuente desarrollado por Microsoft para Windows, Linux, macOS y Web. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código.

### **2.2.6. Sistema operativo Ubuntu**

Ubuntu [34] es una distribución Linux basada en Debian GNU/Linux y patrocinado por Canonical, que incluye principalmente software libre y de código abierto. Puede utilizarse en ordenadores y servidores, está orientado al usuario promedio, con un fuerte enfoque en la facilidad de uso y en mejorar la experiencia del usuario.

## Capítulo 3

# Diseño e implementación

Esta sección presenta los detalles técnicos del diseño e implementación de las diferentes funcionalidades del producto, la arquitectura hardware y software, y finalmente la interfaz de usuario para el control y reporte de las operaciones del robot.

El trabajo fue realizado siguiendo una metodología basada en crear un prototipo funcional con una arquitectura escalable que cumpla con alcance básico especificado en el plan de proyecto, y una vez conseguido agregar en lo posible y de acuerdo a la capacidad, funcionalidades adicionales. De esta forma, una vez que se logró el alcance básico, considerado como la versión v1.0 del producto, se continuó con el desarrollo y se agregó la funcionalidad adicional de control inalámbrico por medio de un joystick, en lo que se considera la versión v2.0 del mismo.

En las siguientes subsecciones se detallan los detalles del diseño e implementación realizados para ambas versiones.

### 3.1. Arquitectura de software del sistema

Con el fin de poder lograr una arquitectura de software modular, se implementaron diferentes componentes software y servicios que abstraen el acceso a los módulos de hardware (explicados en la siguiente sección) y permite un escalamiento fácil de funcionalidades en la expansión de las funcionalidades del robot entre su versión v1.0 y la v2.0.

#### 3.1.1. Arquitectura y diseño de componentes en la versión v1.0

A continuación se puede apreciar la arquitectura de software del sistema en su versión v1.0 y el detalle de sus componentes y servicios.

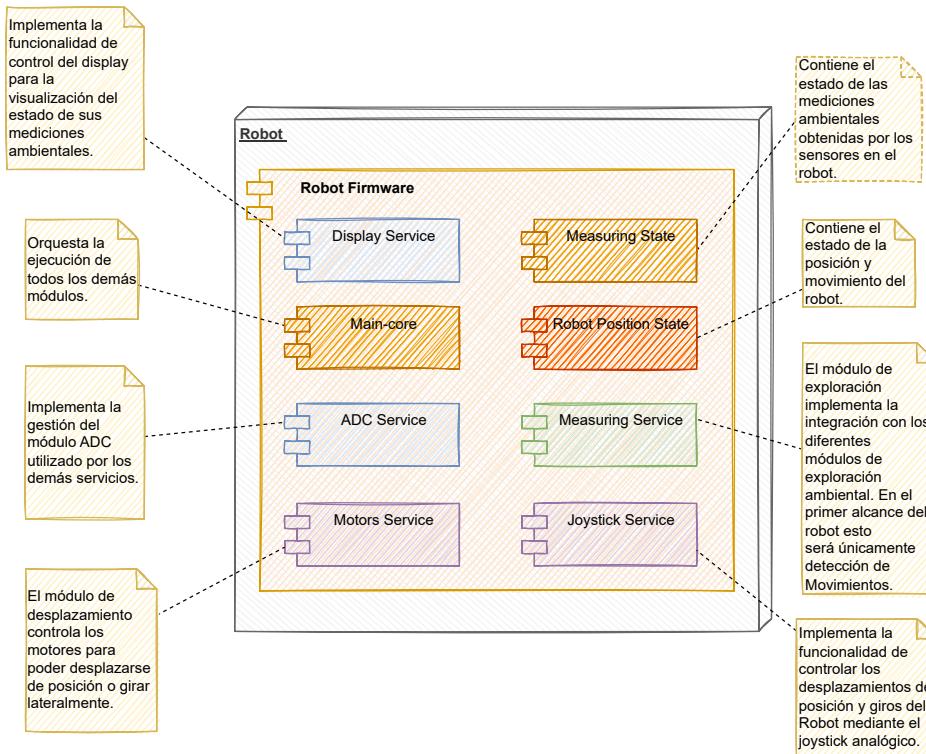


FIGURA 3.1. Arquitectura global.

Los componentes y servicios de software del Robot en la versión v1 son los siguientes:

## 1. Componentes de software

- Main-core: orquesta las diferentes tareas desde las que se invocan los demás componentes de software.
- Measuring Service: abstrae el acceso a los módulos de medición de temperatura, humedad, presión y luminosidad.
- Measuring State: mantiene el estado de cada uno de los parámetros ambientales medidos.
- Robot Position State: mantiene el estado del movimiento actual del robot.
- ADC Service: abstrae el acceso a los módulos que hacen uso de los canales que requieren conversiones ADC (analógico/digital), como el joystick analógico y el fotoresistor.
- Motors Service: abstrae el acceso al módulo de control de motores.
- Display Service: abstrae el acceso al módulo de control del display mediante I2C.
- Joystick Service: abstrae el acceso al módulo de control del joystick.

## 2. Servicios (tareas)

- Display Task
- Measuring Task

c) Joystick Task

d) Motors Task

### 3.1.2. Arquitectura y diseño de componentes en la versión v2.0

Luego de expandir su funcionalidad, los componentes y servicios de software se separaron físicamente en el hardware del robot y el del joystick, agregando además los necesarios para la comunicación inalámbrica. A continuación se puede apreciar la arquitectura de software del sistema en su versión v2.0 y el detalle de sus componentes y servicios.

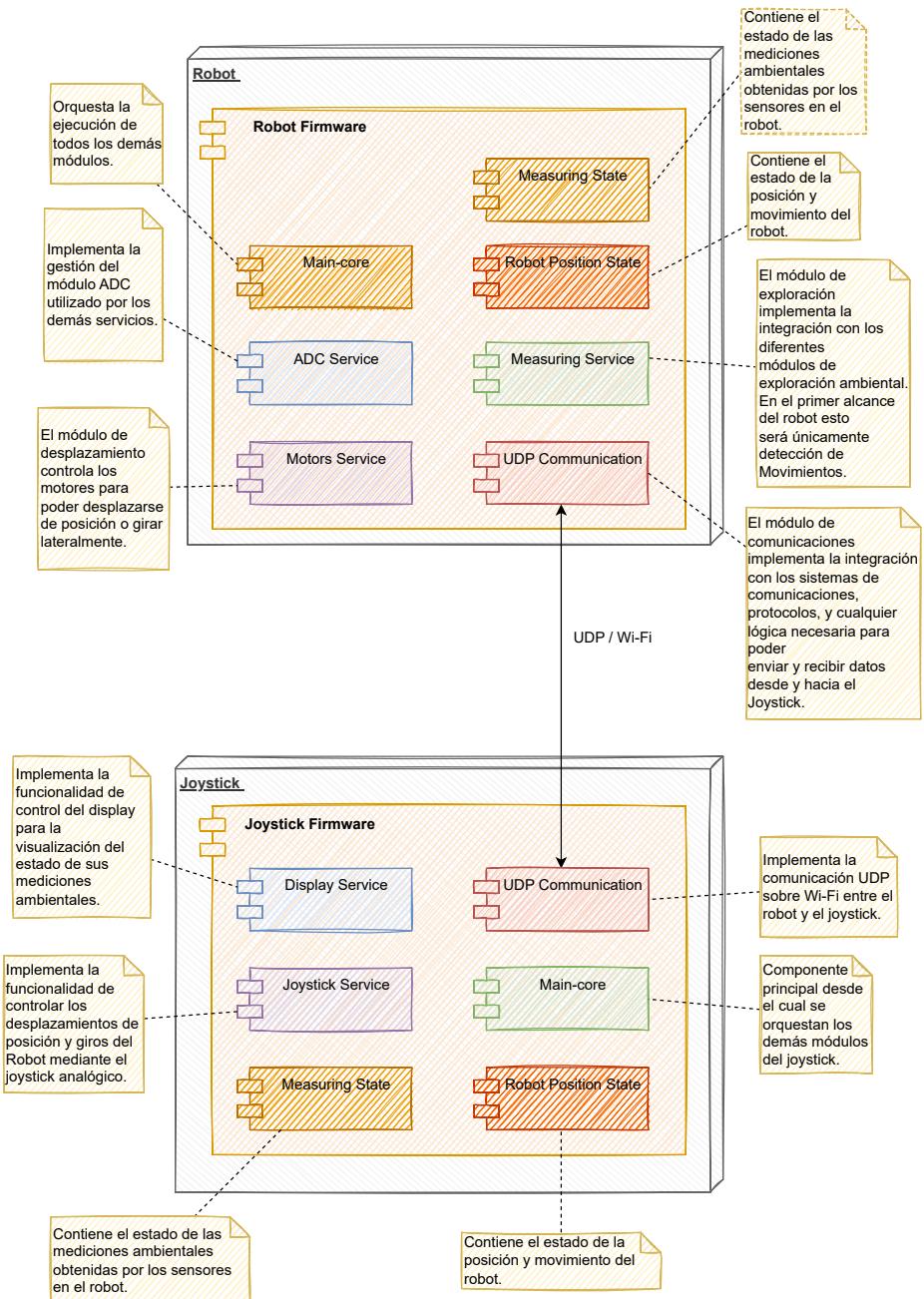


FIGURA 3.2. Arquitectura global.

Los componentes de software y servicios del robot son:

**1. Componentes de software**

- a) Main-core
- b) Measuring Service
- c) Measuring State
- d) Robot Position State
- e) ADC Service
- f) Motors Service
- g) WIFI Service
- h) UDP Service

**2. Servicios (tareas)**

- a) Measuring Task
- b) Motors Task
- c) UDP Server Task

Los componentes de software y servicios del joystick son:

**1. Componentes de software**

- a) main-core
- b) Measuring State
- c) Robot Position State
- d) UDP Service

**2. Servicios (tareas)**

- a) Display Task
- b) Joystick Task
- c) UDP Server Task

### **3.2. Implementación de los módulos**

Los macro componentes de hardware presentes en la arquitectura de la versión v2.0 son el robot y el joystick, que constituyen dos sistemas embebidos independientes implementados con dos microcontroladores ESP32, integrados entre sí por medio de una red Wi-Fi y una comunicación UDP. La arquitectura de hardware del sistema esta compuesta por los siguientes módulos:

- En el robot:
  - Control de los motores DC.
  - Control de los sensores de medición (DHT11, BMP280 y fotorresistor).
  - Gestión de la comunicación inalámbrica vía Wi-Fi (en la versión v2.0).
- En el joystick:

- Control del display.
- Control del joystick.
- Gestión de la comunicación inalámbrica vía Wi-Fi (en la versión v2.0).

La integración de los módulos se realizó mediante el diseño y construcción de una placa integradora central, que conecta los dispositivos hardware con el microcontrolador ESP-32.

### 3.2.1. Control de la red Wi-Fi

El módulo de red Wi-Fi está integrado en el chip ESP32, este soporta múltiples características [35] por lo tanto a nivel hardware no fue necesario realizar ningún conexiónado. A nivel de software, la gestión del módulo Wi-Fi está incluida en el SDK ESP-IDF, y el acceso a este se realiza desde el módulo ADC 2 [13]. Por este motivo, cuando el sistema embebido utiliza el módulo Wi-Fi, el uso del ADC2 queda restringido a esta funcionalidad por lo tanto cualquier otro dispositivo que deba hacer uso del ADC debe ser configurado para utilizar el ADC1, como en el caso de los módulos de joystick y detección de luminosidad, que se explican en las siguientes secciones.

La configuración del *soft access point Wi-Fi* implementado se realizó en base a los parámetros de red detallados en la siguiente tabla 3.1:

TABLA 3.1. Configuración de AP Wi-Fi.

Parámetro	Valor
SSID	Robot
Password	Robot

El desarrollo de este módulo se basó en el ejemplo provisto por Espressif [36]. El código fuente del prototipo realizado puede apreciarse en el siguiente enlace [37].

### 3.2.2. Control del joystick analógico

Para el desarrollo de este prototipo se utilizó el SDK ESP-IDF y se configuró el módulo ADC1 de acuerdo a los siguientes detalles en la tabla 3.3:

TABLA 3.2. Conexionado joystick.

Channel	Unit	Pin GPIO
6	1	34
7	1	35

A continuación, se puede apreciar el conexionado físico en la figura 3.3.

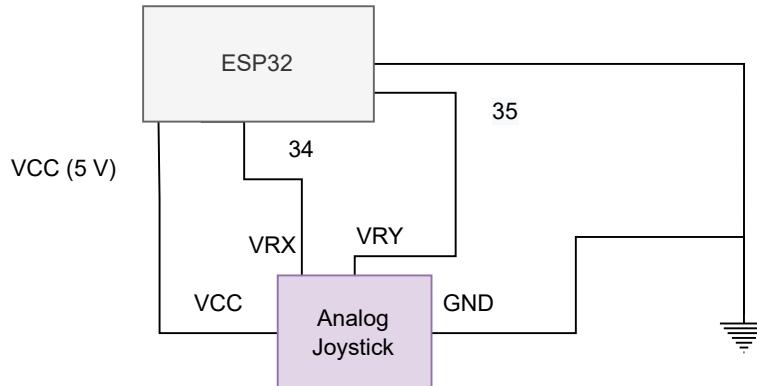


FIGURA 3.3. Conexionado joystick.

El desarrollo de este módulo se basó en el ejemplo provisto por Espressif [38]. El código fuente del prototipo realizado puede apreciarse en el siguiente enlace [39].

### 3.2.3. Medición de valor de luminosidad

Debido a su fuerte dependencia con la temperatura, y especialmente a que su distribución espectral no resulta adecuada para la medición de iluminancia, los fotoresistores no pueden proporcionar una medición precisa de iluminancia como lo haría un luxómetro. No obstante, el fotorresistor resulta que puede ser utilizado como sensor para proporcionar medidas cuantitativas sobre el nivel de luz, tanto en interiores como en exteriores. Para leer los valores del fotorresistor se utilizó la función `adc1_get_raw` del SDK ESP-IDF, donde el valor devuelto se encuentra en el rango [0 - 2050], con 0 el valor de mayor iluminación y 2050 el menor. Finalmente, para calcular el nivel de iluminación como valor porcentual, en el rango [0-100], siendo cero el nivel más oscuro y cien el más iluminado, se utilizó la siguiente función matemática:

$$\text{Nivel de iluminación} = \frac{\text{MaxReading} - \text{reading}}{\text{MaxReading}} \times 100$$

Donde:

- El valor *reading* es la lectura analógica del valor del fotorresistor.
- El valor *MaxReading* es el máximo valor analógico posible de ser entregado por el fotorresistor, en este caso 2050.
- El valor *MaxReading* es el máximo valor analógico posible de ser entregado por el fotorresistor, en este caso 0.

Para el desarrollo de este prototipo se utilizó el SDK de ESP-IDF y se configuró el módulo ADC1 de acuerdo a los detalles provistos en la tabla 3.3.

TABLA 3.3. Conexionado fotorresistor.

Channel	Unit	Pin GPIO
0	1	36

A continuación, se puede apreciar un diagrama de su conexionado físico en la figura 3.4.

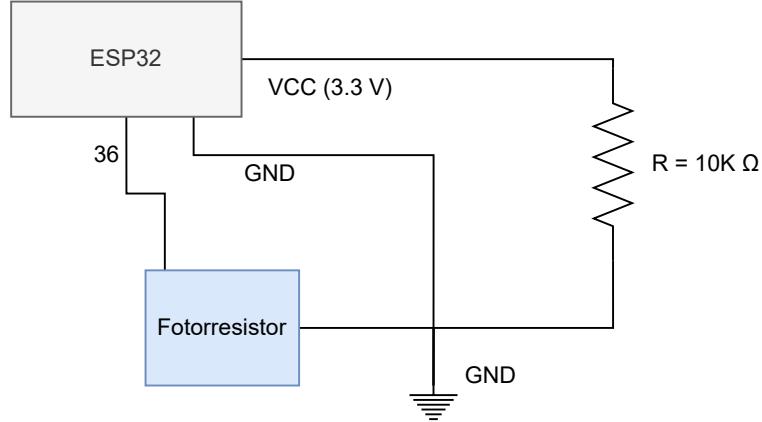


FIGURA 3.4. Conexión fotorresistor.

El desarrollo de este módulo se basó en el ejemplo provisto por Espressif [38]. El código fuente del prototipo realizado puede apreciarse en el siguiente enlace [40].

### 3.2.4. Medición de temperatura y humedad

Para el desarrollo de este módulo se utilizó la biblioteca de código ESP-IDF-Lib Components Library [23] que provee el soporte para gestionar el DHT11. Para acceder a las lecturas del dispositivo se abstrajo mediante el componente Measuring Service, este es invocado por la tarea Measuring Task y el estado de la lectura es almacenado en el componente Measuring State. A continuación, se puede apreciar el conexionado del prototipo en la figura 3.5.

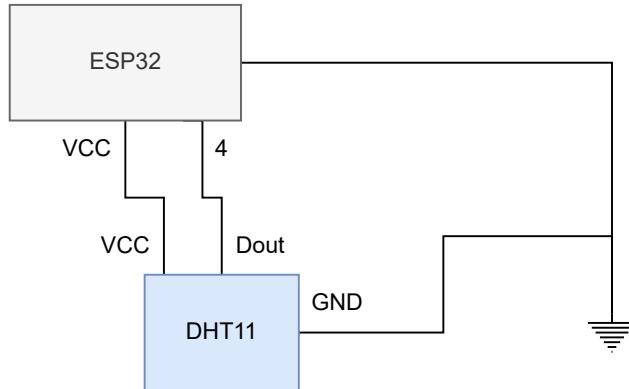


FIGURA 3.5. Circuito del conexionado DHT11.

El desarrollo de este módulo se basó en el ejemplo provisto por la biblioteca ESP-IDF-Lib [41]. El código fuente del prototipo realizado puede apreciarse en el siguiente enlace [42].

### 3.2.5. Medición de presión

Para el desarrollo de este módulo se utilizó el framework ESP-IDF y la biblioteca de código ESP-IDF Components que provee el soporte para gestionar el dispositivo BMP280 por medio del protocolo I2C. El driver es inicializado en el componente main-core para ser posteriormente invocado desde el MeasuringService

en la tarea MeasuringTask. Sus lecturas son guardadas en el MeasuringState. A continuación, se puede apreciar el conexionado del prototipo en la figura 3.6.

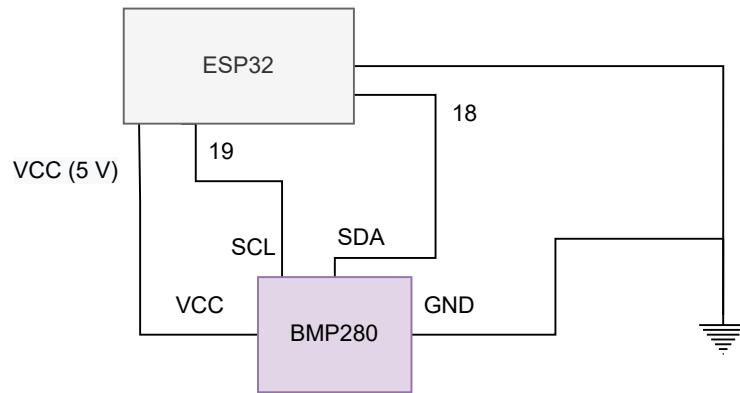


FIGURA 3.6. Conexión BMP280.

El desarrollo de este módulo se basó en el ejemplo provisto por la biblioteca ESP-IDF-Lib [43]. El código fuente del prototipo realizado puede apreciarse en el siguiente enlace [44].

### 3.2.6. Control de motores DC

Para la implementación del módulo de control de los motores de corriente continua se utilizaron a nivel de hardware dos módulos puentes L298N [45], que permiten integrar y controlar dos motores cada uno. Con el fin de alimentar los módulos con una fuente de poder de corriente y tensión consistente, se utilizaron dos baterías de Li-Ion de 3,7 V y 2000 mA/h conectadas en serie, activadas mediante un interruptor. A nivel driver en el ESP32 se utilizó el módulo de control de motores por modulación de pulsos (MCPWM) [46] que por medio de la configuración de sus unidades y del *duty cycle* [47] se puede controlar el sentido y velocidad de rotación de los motores. Los puentes L298N proporcionan también además una tensión de salida de 5 V, y que se utilizó para la alimentación del ESP32 conectando su pin Vin.

En el diagrama de la figura 3.7 puede apreciarse el conexionado lógico para el control de los motores con los componentes mencionados.

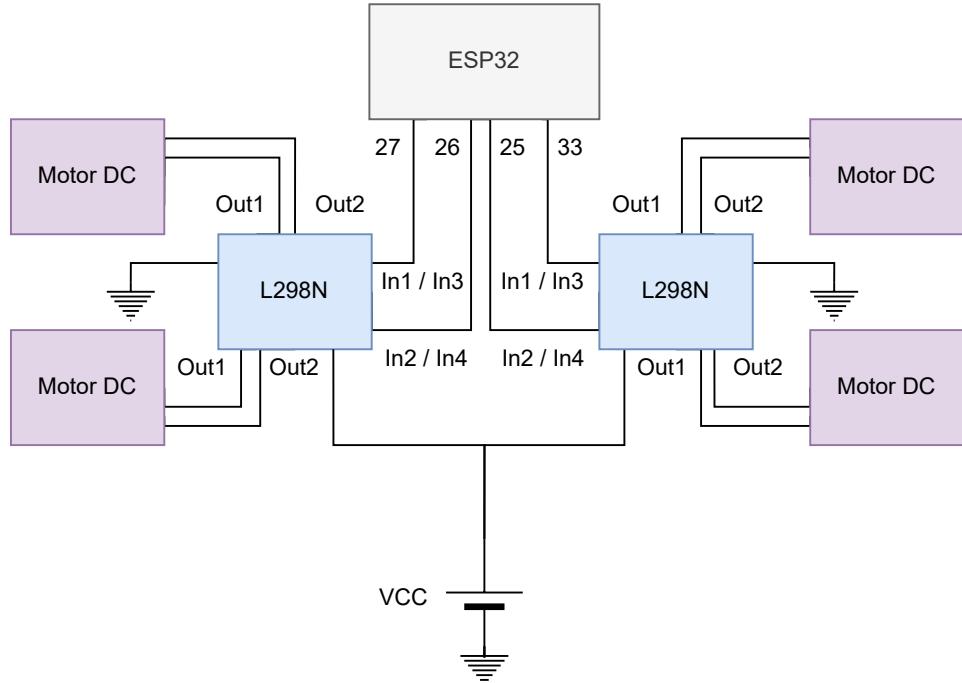


FIGURA 3.7. Conexión motores.

El desarrollo de este módulo se basó en el ejemplo provisto por Espressif [48]. El código fuente del prototipo realizado puede apreciarse en el siguiente enlace [49].

### 3.2.7. Control del display

Para la implementación del módulo de visualización de valores observados, se integró un display de dos líneas y dieciséis caracteres LCM1602A, por medio de un driver I2C que facilita su control. Al basarse en el protocolo I2C el display comparte las mismas líneas SCL y SDA que el sensor BMP280.

En el diagrama de la figura 3.8 puede apreciarse el conexionado lógico para el control del display.

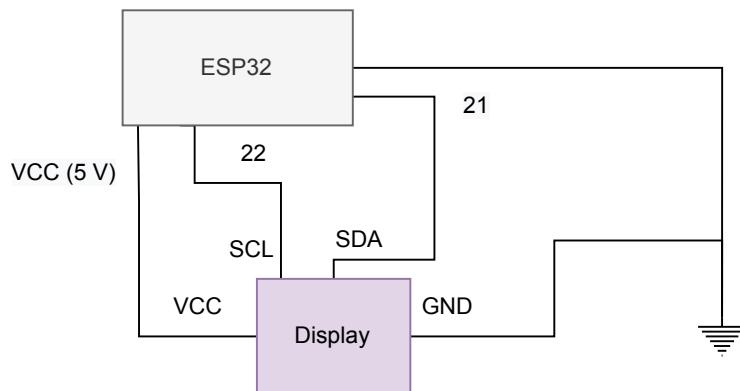


FIGURA 3.8. Conexión display.

El desarrollo de este módulo se basó en el ejemplo provisto en el enlace [50]. El código fuente del prototipo realizado puede apreciarse en el siguiente enlace [51].

### 3.3. Arquitectura de hardware

#### 3.3.1. Ensamblado final del producto v2.0

En las imágenes de la figura 3.9, se pueden apreciar las diferentes perspectivas del robot.



FIGURA 3.9. Hardware del robot.

En las figuras 3.10 y 3.11 se puede apreciar en mayor profundidad el hardware del robot y del joystick con sus subcomponentes.

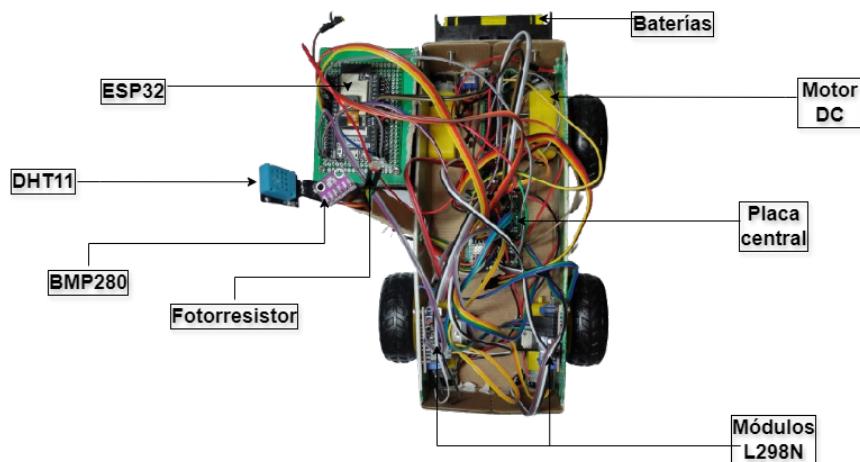


FIGURA 3.10. Detalles del hardware del robot.

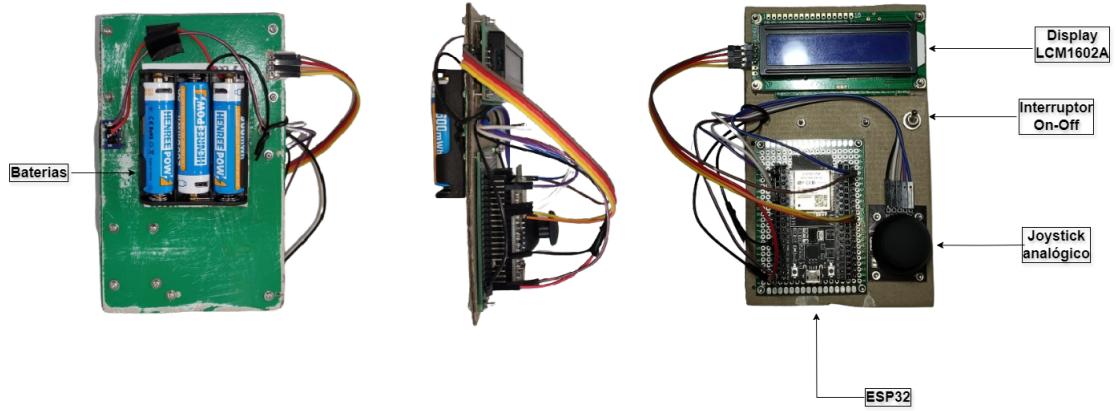


FIGURA 3.11. Detalles del hardware del joystick.

### 3.3.2. Conexionado lógico

En las figuras 3.12 y 3.13 se pueden apreciar los conexionados lógicos del robot y del joystick, respectivamente.

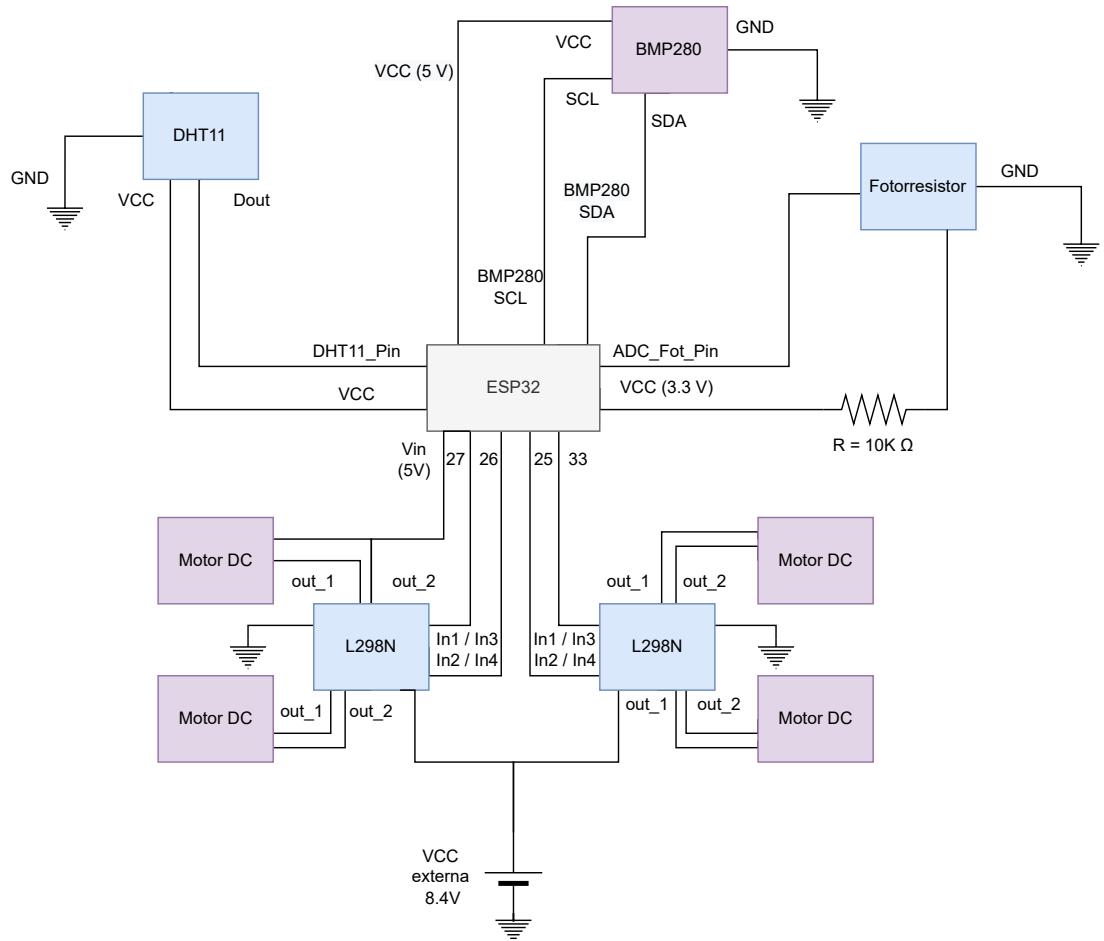


FIGURA 3.12. Conexionado del robot.

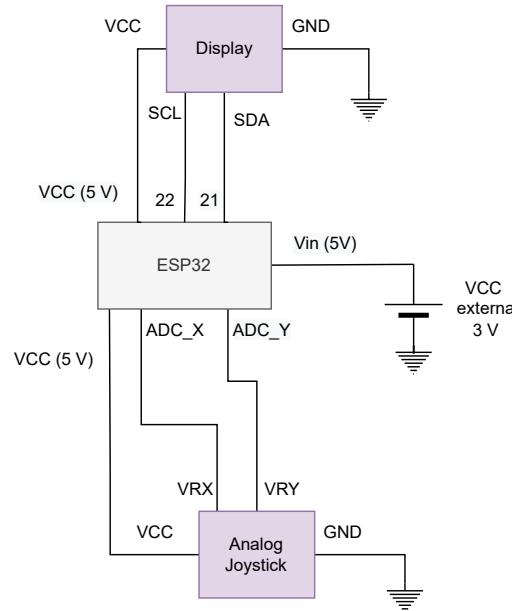


FIGURA 3.13. Conexión del joystick.

### 3.3.3. Conexión física

En la figura 3.14 se puede apreciar el conexiónado físico del robot.

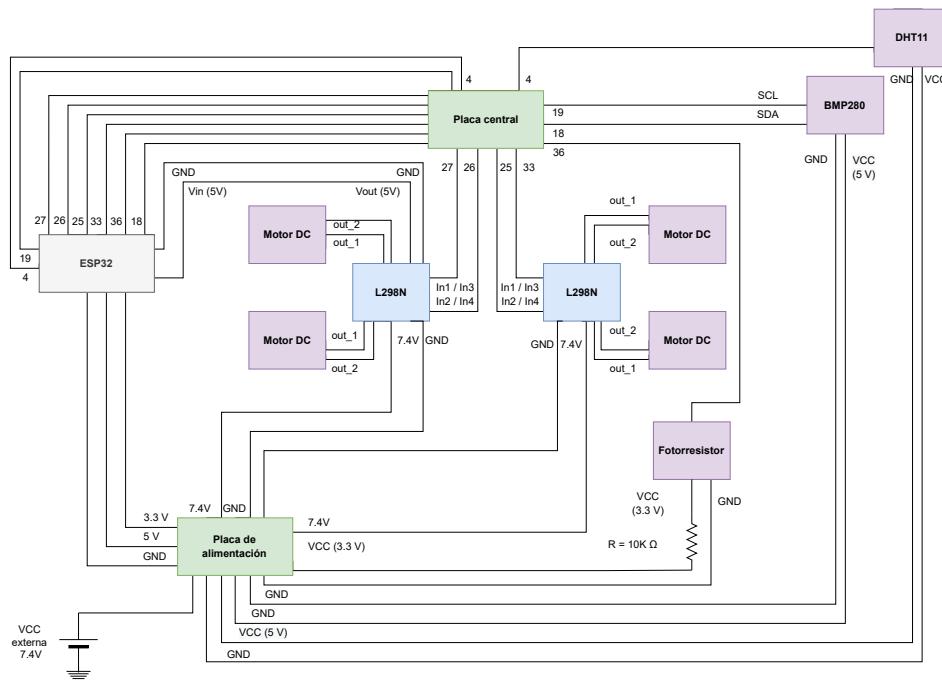


FIGURA 3.14. Conexión física del robot.

### 3.4. Plataforma de desarrollo y ciclo de CI/CD

Durante el ciclo de desarrollo, se utilizaron las herramientas descritas en el capítulo anterior, y para cada prototipo se creó una imagen Docker, extendiendo la de espressif/idf [52]. El conjunto de actividades del mismo fue el siguiente:

1. Codificar localmente en Ubuntu utilizando VSCode.
2. Construcción local en Ubuntu de imagen Docker, de acuerdo a la especificación de los siguientes pasos en el archivo `docker-compose.yml`:
  - a) Compilación del código, enlazado de bibliotecas y empaquetado de la aplicación.
  - b) Ejecución de los tests unitarios con *ceedling*.
  - c) Despliegue (flash) de la aplicación en el ESP32.
3. Versionado del código en el repositorio Github por medio de los comandos `git commit` y `git push`.
4. Construcción en el ambiente de CI/CD por medio de Google Cloud Build de acuerdo a la especificación de los siguientes pasos definidos en el archivo `cloudbuild.yml`:
  - a) Compilación del código, enlazado de bibliotecas y empaquetado de la aplicación.
  - b) Ejecución de los tests unitarios con *ceedling*.
  - c) Construcción de imagen docker.
  - d) *Tagging* y versionado de imagen docker en Google Artifact Registry.

A continuación, se pueden apreciar capturas de pantallas de cada uno de los sistemas utilizados y los pasos ejecutados. En la imagen 3.15, se puede apreciar la salida por consola tras la ejecución de los tests unitarios y construcción de la imagen Docker de manera local.

```
Test 'test_adc_service.c'
-----
Running test_adc_service.out...

Test 'test_display_service.c'
-----
Running test_display_service.out...

Test 'test_joystick_service.c'
-----
Running test_joystick_service.out...

Test 'test_measuring_services.c'
-----
Running test_measuring_services.out...

Test 'test_motors_service.c'
-----
Running test_motors_service.out...

Test 'test_robot_position_state.c'
-----
Running test_robot_position_state.out...

Test 'test_wifi_service.c'
-----
Running test_wifi_service.out...

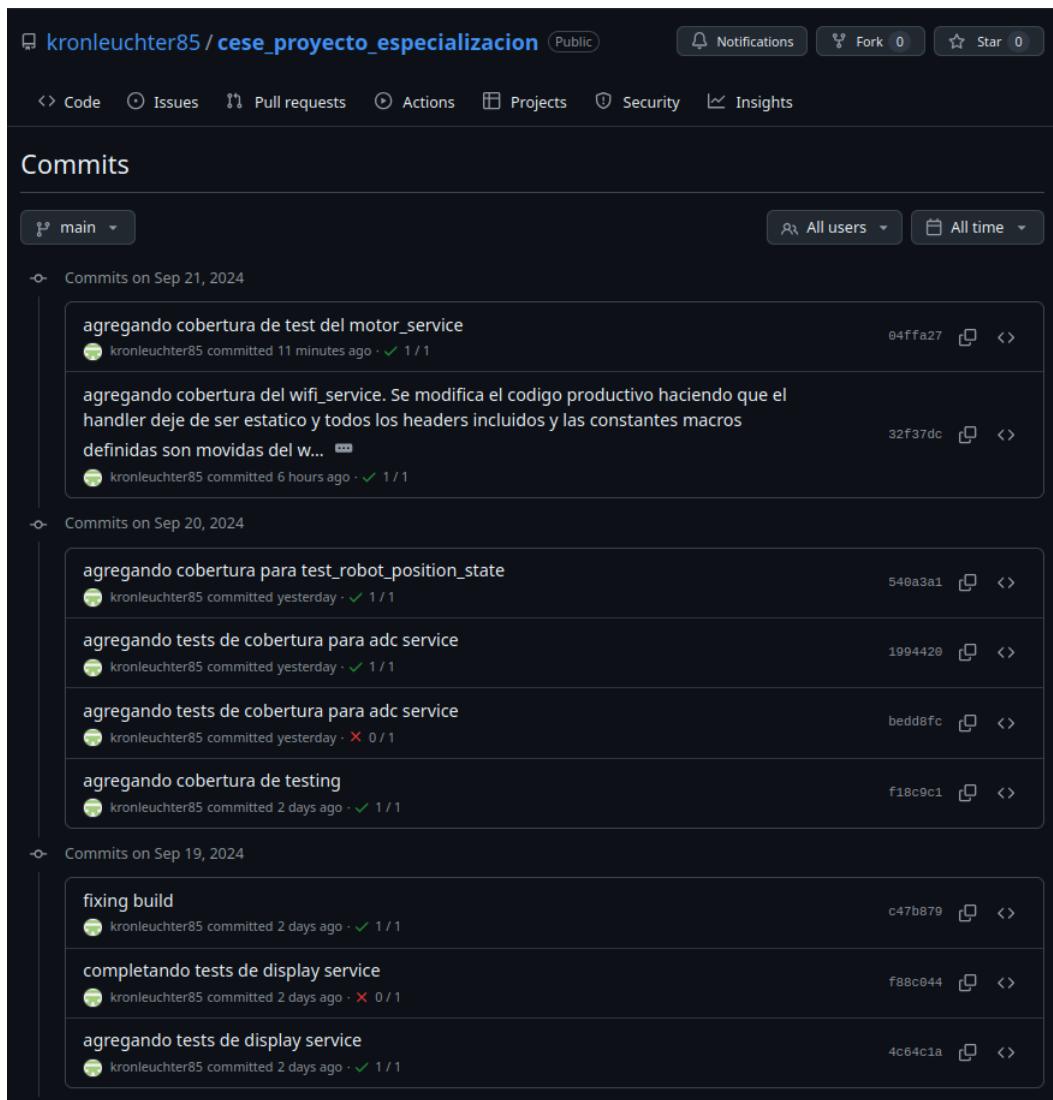
-----
TEST OUTPUT
-----
[test_motors_service.c]
- "initializing mcpwm gpio..."
- "Configuring Initial Parameters of mcpwm..."
- "initializing mcpwm gpio..."
- "Configuring Initial Parameters of mcpwm..."

[test_wifi_service.c]
- "wifi_init_softap finished. SSID:1 password:1 channel:1"
- "station 12:34:56:78:9A:BC leave, AID=1"
- "station 12:34:56:78:9A:BC leave, AID=1"

-----
OVERALL TEST SUMMARY
-----
TESTED: 39
PASSED: 39
FAILED: 0
IGNORED: 0
```

FIGURA 3.15. Ejecución de tests por consola.

Luego de realizar *commit* y *push* de los cambios locales se pueden apreciar en la imagen 3.16 el listado de las versiones en Github.

FIGURA 3.16. Listado de *commits* en Github.

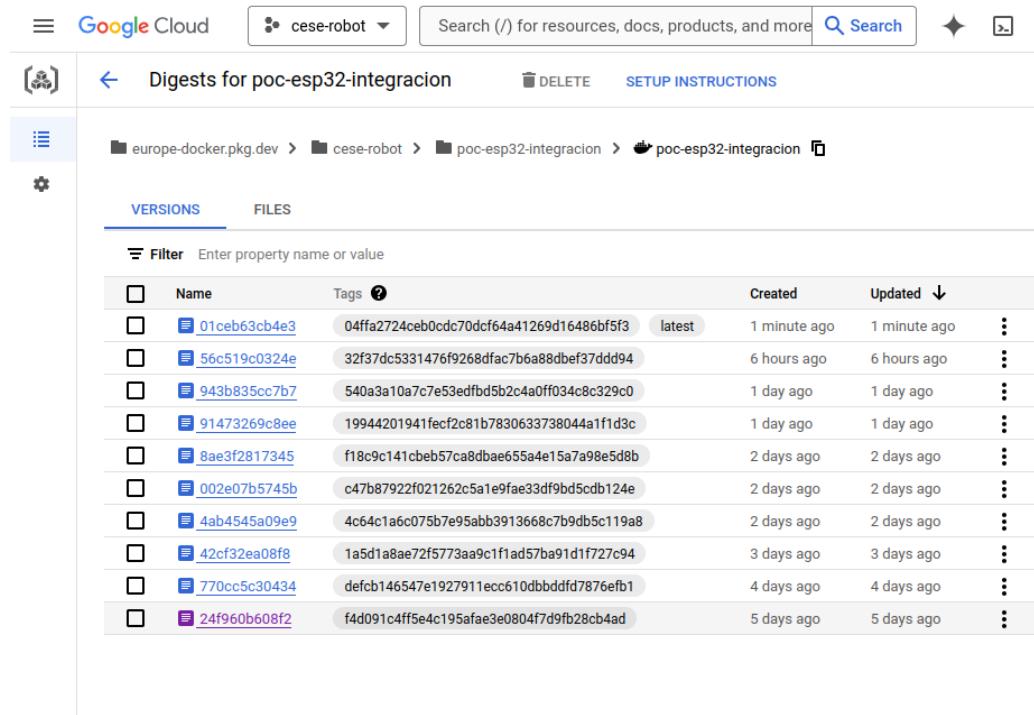
En la imagen 3.17 se pueden apreciar los diferentes builds disparados en Cloud Build referenciando los commits de Github.

The screenshot shows the Google Cloud Build history interface. At the top, there are navigation icons for Google Cloud, a dropdown for the project ('cese-robot'), a search bar ('Search (/) for resources, docs, products, and more'), and a search button ('Search'). Below the header, the title 'Build history' is displayed next to a 'STOP STREAMING BUILDS' button. A 'Region' dropdown is set to 'global (non-regional)'. A 'Filter' input field is present. The main area is a table listing build records:

	Status	Build	Source	Ref	Commit	Created	Duration
<input type="checkbox"/>	<span>✓</span>	<a href="#">8692aa76</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">04ffa27</a>	9/21/24, 11:40 AM	2 min 48 sec
<input type="checkbox"/>	<span>✓</span>	<a href="#">4fd6aed0</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">32f37dc</a>	9/21/24, 5:38 AM	3 min
<input type="checkbox"/>	<span>✓</span>	<a href="#">0b8a79d1</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">540a3a1</a>	9/20/24, 7:58 AM	2 min 40 sec
<input type="checkbox"/>	<span>✓</span>	<a href="#">fcd294ad</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">1994420</a>	9/20/24, 7:32 AM	3 min 13 sec
<input type="checkbox"/>	<span>○</span>	<a href="#">33b5ca8b</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">bedd8fc</a>	9/20/24, 7:22 AM	9 min 14 sec
<input type="checkbox"/>	<span>✓</span>	<a href="#">03a7fa23</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">f18c9c1</a>	9/19/24, 7:19 PM	2 min 18 sec
<input type="checkbox"/>	<span>✓</span>	<a href="#">bfc49475</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">c47b879</a>	9/19/24, 8:18 AM	3 min 13 sec
<input type="checkbox"/>	<span>!</span>	<a href="#">611ae05f</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">f88c044</a>	9/19/24, 8:13 AM	2 min 18 sec
<input type="checkbox"/>	<span>✓</span>	<a href="#">663b4c7f</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">4c64c1a</a>	9/19/24, 8:03 AM	2 min 49 sec
<input type="checkbox"/>	<span>○</span>	<a href="#">1a50ab16</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">4c64c1a</a>	9/19/24, 8:02 AM	32 sec
<input type="checkbox"/>	<span>✓</span>	<a href="#">dbc6a61d</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">1a5d1a8</a>	9/18/24, 11:01 AM	2 min 26 sec
<input type="checkbox"/>	<span>!</span>	<a href="#">c86acd89</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">7bf12f8</a>	9/18/24, 8:27 AM	5 sec
<input type="checkbox"/>	<span>!</span>	<a href="#">a74eb891</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">20c0354</a>	9/18/24, 8:25 AM	5 sec
<input type="checkbox"/>	<span>!</span>	<a href="#">167dc276</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">a10f80e</a>	9/18/24, 8:23 AM	5 sec
<input type="checkbox"/>	<span>!</span>	<a href="#">31e19ffa</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">1fa8152</a>	9/18/24, 5:58 AM	6 sec
<input type="checkbox"/>	<span>✓</span>	<a href="#">67d90dbf</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">defcb14</a>	9/17/24, 9:15 AM	2 min 16 sec
<input type="checkbox"/>	<span>✓</span>	<a href="#">c003ff38</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">f4d091c</a>	9/16/24, 4:48 PM	4 min 44 sec
<input type="checkbox"/>	<span>✓</span>	<a href="#">6770226a</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">d6d517c</a>	9/16/24, 4:45 PM	2 min
<input type="checkbox"/>	<span>✓</span>	<a href="#">635c84be</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">737be7c</a>	9/16/24, 4:37 PM	2 min 21 sec
<input type="checkbox"/>	<span>✓</span>	<a href="#">76586d13</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">ec6e7e7</a>	9/16/24, 2:29 PM	2 min 18 sec
<input type="checkbox"/>	<span>!</span>	<a href="#">cf5a5566</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">ef638d0</a>	9/16/24, 2:28 PM	6 sec
<input type="checkbox"/>	<span>!</span>	<a href="#">ef3b27aa</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">43f9df6</a>	9/16/24, 2:25 PM	7 sec
<input type="checkbox"/>	<span>✓</span>	<a href="#">b8d9f1a4</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">f3f6030</a>	9/16/24, 2:23 PM	5 sec
<input type="checkbox"/>	<span>!</span>	<a href="#">572346f7</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">9623f2d</a>	9/16/24, 2:20 PM	6 sec
<input type="checkbox"/>	<span>✓</span>	<a href="#">5c3b9a5f</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">3650450</a>	9/16/24, 2:09 PM	2 min 23 sec
<input type="checkbox"/>	<span>!</span>	<a href="#">62991d0f</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">34bdb75</a>	9/16/24, 2:08 PM	-
<input type="checkbox"/>	<span>!</span>	<a href="#">d8b523ef</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">8321b8d</a>	9/16/24, 2:05 PM	5 sec
<input type="checkbox"/>	<span>!</span>	<a href="#">45e6d6f1</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">8269577</a>	9/16/24, 2:03 PM	10 sec
<input type="checkbox"/>	<span>!</span>	<a href="#">434b04ba</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">39584cf</a>	9/16/24, 2:02 PM	5 sec
<input type="checkbox"/>	<span>✓</span>	<a href="#">36a99f94</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">2c3dd41</a>	9/16/24, 1:55 PM	2 min 7 sec
<input type="checkbox"/>	<span>!</span>	<a href="#">430a02cc</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">63770d4</a>	9/15/24, 5:34 PM	5 sec
<input type="checkbox"/>	<span>!</span>	<a href="#">e4624870</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">639a090</a>	9/15/24, 5:14 PM	5 sec
<input type="checkbox"/>	<span>!</span>	<a href="#">9726342e</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">430b9cb</a>	9/15/24, 4:58 PM	2 min 14 sec
<input type="checkbox"/>	<span>!</span>	<a href="#">1e8a33dd</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">3f35a65</a>	9/15/24, 4:50 PM	2 min 32 sec
<input type="checkbox"/>	<span>!</span>	<a href="#">e52428c2</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">a5a31f9</a>	9/15/24, 4:46 PM	2 min 12 sec
<input type="checkbox"/>	<span>!</span>	<a href="#">2c30803f</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">c13fd35</a>	9/15/24, 4:43 PM	1 min 55 sec
<input type="checkbox"/>	<span>!</span>	<a href="#">4adb3dfc</a>	<a href="#">kronleuchter85/cese_proyecto_especializacion</a>	main	<a href="#">b4d7a3a</a>	9/15/24, 4:42 PM	5 sec

FIGURA 3.17. Listado de *builds* en Google CloudBuild.

Finalmente en la imagen 3.18 se pueden apreciar las imágenes Docker versionadas y almacenadas en Artifact Registry.



The screenshot shows a Google Cloud interface for managing Docker images. At the top, there's a navigation bar with 'Google Cloud' and a dropdown for 'cese-robot'. A search bar is also at the top. Below the navigation, the title 'Digests for poc-esp32-integracion' is displayed, along with a 'DELETE' button and 'SETUP INSTRUCTIONS'. A breadcrumb trail shows the path: europe-docker.pkg.dev > cese-robot > poc-esp32-integracion > poc-esp32-integracion. There are tabs for 'VERSIONS' (which is selected) and 'FILES'. A filter bar allows entering a property name or value. The main area is a table listing 12 Docker image versions:

	Name	Tags	Created	Updated	⋮	
<input type="checkbox"/>	<a href="#">01ceb63cb4e3</a>	04ffa2724ceb0cdc70dcf64a41269d16486bf5f3	latest	1 minute ago	1 minute ago	⋮
<input type="checkbox"/>	<a href="#">56c519c0324e</a>	32f37dc5331476f9268dfac7b6a88dbef37ddd94		6 hours ago	6 hours ago	⋮
<input type="checkbox"/>	<a href="#">943b835cc7b7</a>	540a3a10a7c7e53edfb5b2c4a0ff034c8c329c0		1 day ago	1 day ago	⋮
<input type="checkbox"/>	<a href="#">91473269c8ee</a>	19944201941fecf2c81b7830633738044a1f1d3c		1 day ago	1 day ago	⋮
<input type="checkbox"/>	<a href="#">8ae3f2817345</a>	f18c9c141cbeb57ca8dbae655a4e15a7a98e5d8b		2 days ago	2 days ago	⋮
<input type="checkbox"/>	<a href="#">002e07b5745b</a>	c47b87922f021262c5a1e9fae33df9bd5cdb124e		2 days ago	2 days ago	⋮
<input type="checkbox"/>	<a href="#">4ab4545a09e9</a>	4c64c1a6c075b7e95abb3913668c7b9db5c119a8		2 days ago	2 days ago	⋮
<input type="checkbox"/>	<a href="#">42cf32ea08f8</a>	1a5d1a8ae72f5773aa9c1f1ad57ba91df727c94		3 days ago	3 days ago	⋮
<input type="checkbox"/>	<a href="#">770cc5c30434</a>	defcb146547e1927911ecc610dbbddfd7876efb1		4 days ago	4 days ago	⋮
<input type="checkbox"/>	<a href="#">24f960b608f2</a>	f4d091c4ff5e4c195afae3e0804f7d9fb28cb4ad		5 days ago	5 days ago	⋮

FIGURA 3.18. Listado de versiones de imágenes doker en Google ArtifactRegistry.

### 3.5. Reportes de ejecución y cobertura de testing unitario

A continuación se presentan los reportes de testing generados por la herramienta *ceedling* con el complemento *gcov* donde se puede apreciar el nivel de cobertura logrado para cada servicio.

En la imagen 3.19, se puede apreciar la salida por pantalla tras la ejecución local de *ceedling* con el plugin de cobertura, en donde se evidencia la cantidad de test cases.

```

-----
GCOV: OVERALL TEST SUMMARY
-----
TESTED: 39
PASSED: 39
FAILED: 0
IGNORED: 0

-----
GCOV: CODE COVERAGE SUMMARY
-----
adc_service.c Lines executed:100.00% of 17
adc_service.c No branches
adc_service.c Calls executed:100.00% of 13

display_service.c Lines executed:100.00% of 11
display_service.c No branches
display_service.c Calls executed:100.00% of 7

joystick_service.c Lines executed:91.67% of 24
joystick_service.c Branches executed:100.00% of 12
joystick_service.c Taken at least once:83.33% of 12
joystick_service.c Calls executed:100.00% of 2

measuring_services.c Lines executed:77.78% of 36
measuring_services.c Branches executed:100.00% of 10
measuring_services.c Taken at least once:80.00% of 10
measuring_services.c Calls executed:71.43% of 7

motors_service.c Lines executed:100.00% of 30
motors_service.c Branches executed:100.00% of 4
motors_service.c Taken at least once:75.00% of 4
motors_service.c Calls executed:100.00% of 15

robot_position_state.c Lines executed:81.48% of 27
robot_position_state.c Branches executed:100.00% of 24
robot_position_state.c Taken at least once:87.50% of 24
robot_position_state.c No calls

wifi_service.c Lines executed:100.00% of 19
wifi_service.c Branches executed:100.00% of 4
wifi_service.c Taken at least once:100.00% of 4
wifi_service.c Calls executed:100.00% of 15

```

FIGURA 3.19. Reportes de testing por consola.

En la siguiente imagen 3.20 se pueden apreciar los detalles de la cobertura por cada servicio.

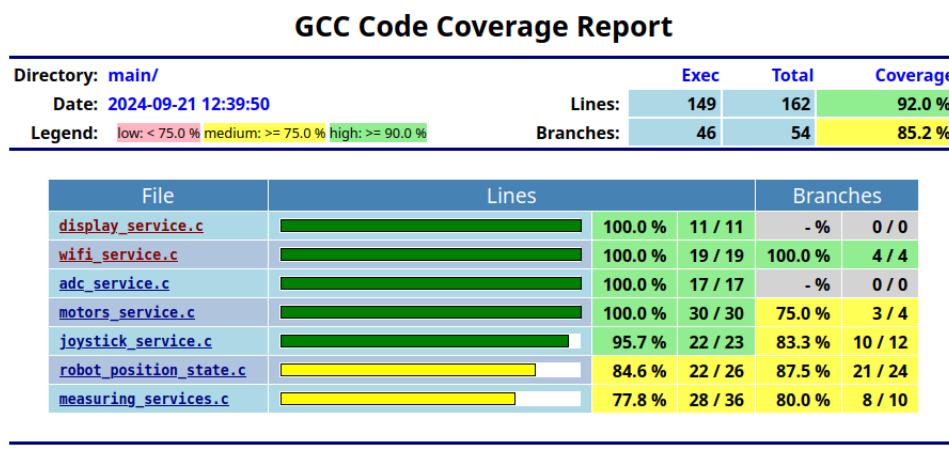


FIGURA 3.20. Reportes de testing web.

## Capítulo 4

# Ensayos y resultados

Esta sección presenta los diferentes prototipos realizados para determinar la viabilidad de cada una de las funcionalidades provistas, la metodología de desarrollo, testing y, finalmente, los entregables finales del trabajo.

### 4.1. Proceso de desarrollo y aseguramiento de calidad

Para el proceso de desarrollo se realizaron pruebas de concepto de las diferentes funcionalidades utilizando como materiales la bibliografía encontrada en Internet, las hojas de datos y los ejemplos de código provistos por el SDK y bibliotecas empleadas. Una vez logrado el objetivo funcional de cada componente, se optimizó y encapsuló cada módulo para ser integrado de manera individual a un prototipo integrador sin afectar el funcionamiento de cualquier otro módulo. De esta manera, se desarrolló un prototipo integrador como la sumatoria de todos los módulos de forma incremental, probándose por regresión que los módulos ya integrados previamente siguieran funcionando de forma óptima.

Una vez logrado el prototipo integrador, con todas las funcionalidades de la versión v1.0, se procedió a expandir el hardware para crear la versión v2.0. Para ello, se extrajeron los módulos de joystick y display, que posteriormente se agregarían al sistema embebido del joystick, y se incorporaron los módulos de conectividad UDP sobre Wi-Fi.

Tras lograr la versión v2.0 se repite el proceso de control de calidad de los diferentes módulos ya integrados.

En las siguientes secciones, se detallan las diferentes pruebas realizadas.

### 4.2. Verificación técnica de los diferentes módulos

Todos los módulos fueron probados mediante una inspección visual durante el proceso de pruebas de concepto.

#### 4.2.1. Verificación del módulo de joystick

Se verificó visualmente que los valores del joystick analógico puedan ser leídos apropiadamente, y que sean representativos y relevantes con la dirección del movimiento de la palanca sobre sus coordenadas X e Y.

#### 4.2.2. Verificación del módulo de control del display

Se verificó visualmente que el display mostraba los caracteres programados en la prueba de concepto con una intensidad de luz aceptable para poder leerlos apropiadamente.

#### 4.2.3. Verificación del módulo de control de motores

Se verificó visualmente que individualmente el motor pudiera girar en ambos sentidos. Luego, al implementarse los cuatro motores con sus ruedas, se probó que se puedan realizar los giros en todas las direcciones.

#### 4.2.4. Verificación del módulo de medición de temperatura y humedad

Se verificó visualmente que los valores obtenidos por el sensor DHT11 coincidieran con los esperados en relación a la temperatura en el interior del lugar de experimentación y que la humedad detectada se aproximara a los valores reportados por Google.

#### 4.2.5. Verificación del módulo de medición de presión atmosférica

Se verificó visualmente que el valor obtenido por el sensor BMP280 fuera cercano a lo esperado en relación al valor reportado por Google.

#### 4.2.6. Verificación del módulo de medición de luminosidad

Se verificó visualmente que los valores obtenidos del fotoresistor, una vez transformados a valores absolutos porcentuales, reflejaban el nivel de luminosidad ambiental del interior del lugar de experimentación.

#### 4.2.7. Verificación del módulo de comunicación UTP sobre Wi-Fi

Por medio de dos programas UDP, uno cliente y uno servidor, se probó el establecimiento de la comunicación UDP entre dos ESP32. Posteriormente, se incorporó el servicio de comunicaciones UDP en el robot, y desde el programa cliente se enviaban las acciones que representaban las direcciones del movimiento (FORWARD, BACKWARD, LEFT, RIGHT). Se observó visualmente cómo el robot giraba sus ruedas en función de los comandos enviados. Finalmente, se incorporó el módulo de comunicaciones en el joystick y activó el desplazamiento en cada una de sus direcciones. El robot se desplazó correctamente en respuesta a cada comando recibido.

### 4.3. Pruebas funcionales y validación del producto

El proceso de validación y pruebas del producto, se realizó comparando el resultado obtenido con los valores esperados en el alcance del proyecto.

Para la medición de temperatura, humedad y presión, se utilizó el anemómetro digital de la figura 2.17, mientras que para la validación de la medición de luminosidad ambiental se utilizó la observación visual.

Las pruebas realizadas fueron las siguientes:

- Prueba y validación del módulo de visualización de display.
- Prueba y validación del módulo de medición de temperatura y humedad.
- Prueba y validación del módulo de medición de presión atmosférica.
- Prueba y validación del módulo de medición de luminosidad ambiental.
- Prueba y validación del control y desplazamiento del robot.

En las siguientes secciones se presentan las diferentes pruebas funcionales realizadas sobre el producto.

#### 4.3.1. Prueba y validación del módulo de visualización de display

Se verificó el funcionamiento del display visualizando las lecturas de los valores censados y transmitidos por el robot. Se controló que:

- Las lecturas sean nítidas y entendibles.
- Las unidades de medida están presentes.
- Haya un detalle de lo que se está midiendo acompañando las lecturas y la unidad de medida.
- El nivel de luminosidad sea óptimo para permitir la lectura independiente-mente de la iluminación ambiental.
- Se presentan las lecturas de todos los valores observados.

A continuación, se pueden apreciar algunas fotografías tomadas durante el proceso de experimentación:

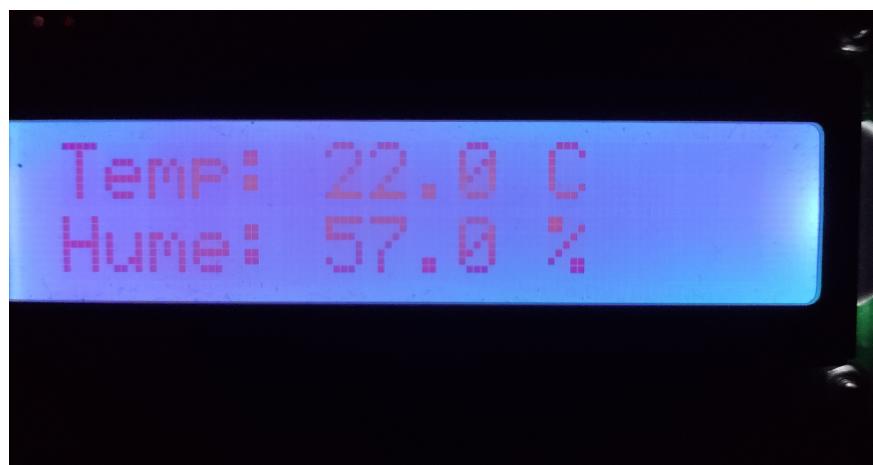


FIGURA 4.1. Visualización del display en la oscuridad.

En la siguiente sección pueden encontrarse los videos en los que se puede apreciar el funcionamiento del display durante el día [53] y en la oscuridad [54].

#### 4.3.2. Prueba y validación del módulo de medición de temperatura y humedad

Se compararon los valores medidos por el módulo de medición de temperatura y humedad basado en el sensor DHT11 con los obtenidos a través de un dispositivo

de medición de temperatura y humedad. Se realizó la medición en diferentes contextos:

- En el interior de una casa.
- En el exterior, durante el día.

En la siguiente tabla se pueden apreciar los resultados.

TABLA 4.1. Resultados de mediciones de temperatura y humedad

Contexto	Temp. Robot	Temp. Ref.	Hume. Robot	Hume. Ref.
Interior	22,0	23,6	44,0 - 45,0	58,5
Exterior (día)	17,0	14,0	47,0 - 53,0	62,9 - 64,0
Exterior (noche)	-	-	-	-

A continuación, se pueden apreciar algunas fotografías tomadas durante el proceso de experimentación:

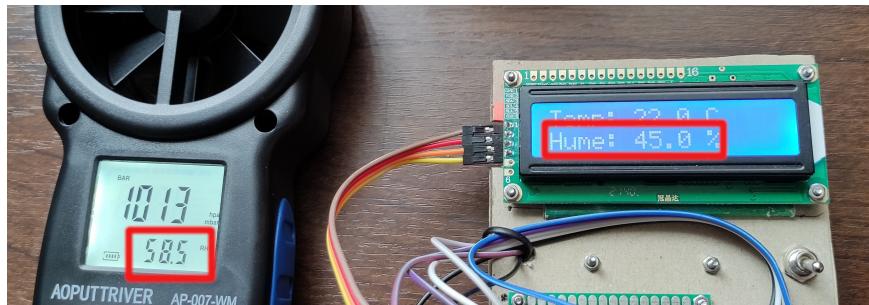


FIGURA 4.2. Medición de humedad en el interior.

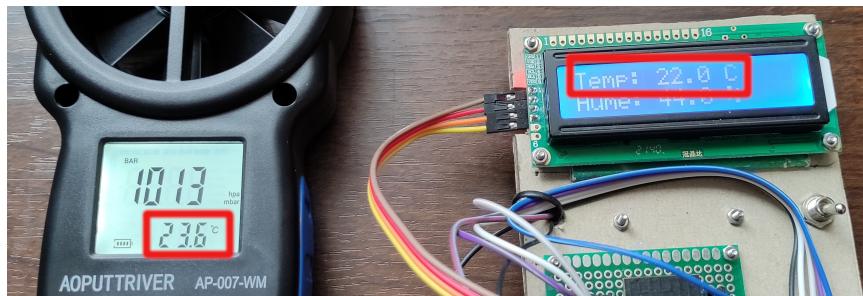


FIGURA 4.3. Medición de temperatura en el interior.



FIGURA 4.4. Medición de humedad en el exterior.

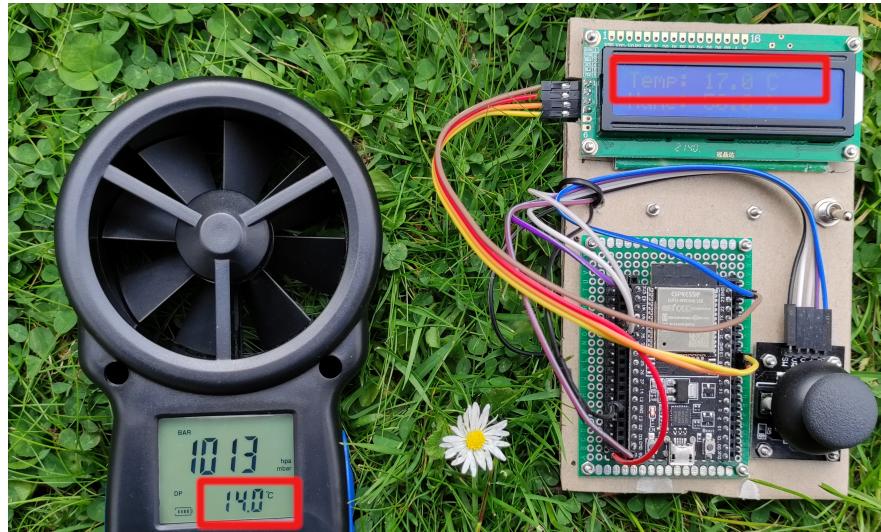


FIGURA 4.5. Medición de temperatura en el exterior.

En la siguiente sección puede encontrarse el video que muestra el funcionamiento del módulo de mediciones, en el que se aprecia el funcionamiento del módulo de medición de temperatura y humedad [53].

#### 4.3.3. Prueba y validación del módulo de medición de presión atmosférica

Se compararon los valores medidos por el módulo de medición de presión basado en el sensor BMP280 con los obtenidos a través de un dispositivo manómetro digital. Las mediciones se realizaron en el interior de la vivienda en dos días distintos.

En la siguiente tabla pueden apreciarse los resultados obtenidos:

TABLA 4.2. Resultados de mediciones de presión ambiental.

Contexto	Presión. Robot	Presión. Ref.
Día 1	1013	1018,9
Día 2	1003,9	998

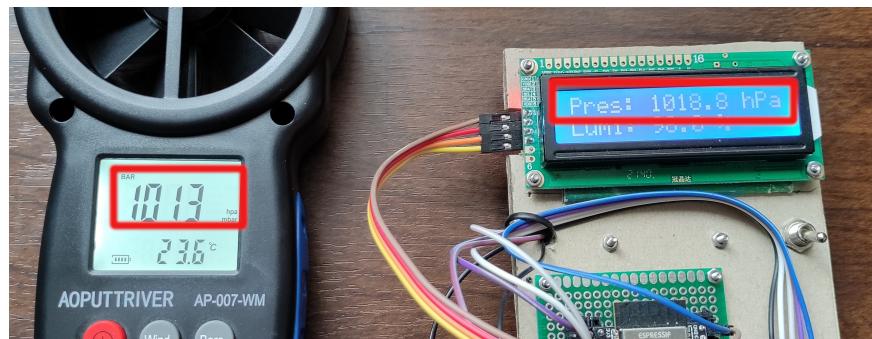


FIGURA 4.6. Medición de presión atmosférica en el interior.

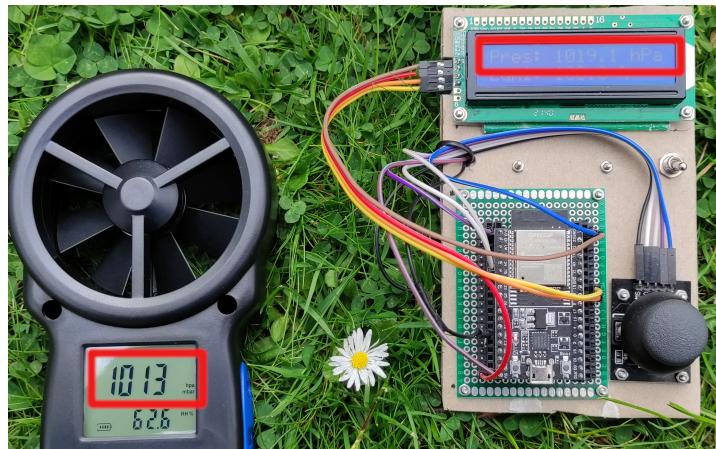


FIGURA 4.7. Medición de presión atmosférica en el exterior.

En la siguiente sección puede encontrarse el video que muestra el funcionamiento del módulo de mediciones, en el que se aprecia el funcionamiento del módulo de medición de presión atmosférica [53].

#### 4.3.4. Prueba y validación del módulo de medición de luminosidad ambiental

Se compararon los valores medidos por el módulo de medición de luminosidad basado en un fotorresistor percibidos por el ojo humano sin utilizar ningún dispositivo de medición. Se realizó la medición en diferentes escenarios:

- En exteriores durante el día con luz ambiental.
- En interiores con luz ambiental.
- En interiores a oscuras.

Los resultados mostraron que los valores porcentuales indicados por el módulo de medición de luminosidad son consistentes con los niveles de luz detectados por el ojo humano. En las figuras 4.8, 4.9 y 4.10 pueden apreciarse los resultados de las mediciones.



FIGURA 4.8. Medición de luminosidad ambiental en el exterior durante el día.



FIGURA 4.9. Medición de luminosidad ambiental en interiores durante el día.



FIGURA 4.10. Medición de luminosidad ambiental en interiores durante la noche.

En la siguiente sección puede encontrarse el video que muestra el funcionamiento del módulo de mediciones, en el que se aprecia el funcionamiento del módulo de medición de luminosidad ambiental [53].

#### 4.3.5. Prueba y validación del control y desplazamiento del robot

Se verificó el control del desplazamiento del robot de forma visual por medio de accionar el joystick en las diferentes coordenadas (X;Y) y se controló que:

- La dirección del movimiento del robot sea acorde al accionamiento del joystick.
- El tiempo de respuesta en el movimiento del robot y tras el accionar del joystick sea mínimo, permitiendo una buena experiencia de usuario.

En la siguiente sección pueden encontrarse los videos [55] y [56] evidenciando la demostración de este experimento.

## 4.4. Videos del producto durante el ensamblado y experimentación

En las siguientes subsecciones se listan los videos realizados durante el proceso de demostración del producto funcionando así como los grabados casualmente durante armado y prototipado del mismo.

#### 4.4.1. Videos demostrativos del producto final

Los experimentos realizados para evidenciar el cumplimiento con los requerimientos funcionales del producto son los siguientes:

- Demo - Hardware del producto [57].
- Demo - Comunicación Wi-Fi [58].
- Demo - Control de movimiento de las ruedas [55].
- Demo - Medición y visualización de parámetros ambientales [53].
- Demo - Control de desplazamiento en un circuito [56].
- Demo - Visualización del Display en la oscuridad [54].

#### 4.4.2. Videos durante el prototipado y ensamblado del robot

- Prototipado Robot v1 - Ensamblado (1) [59].
- Prototipado Robot v1 - Ensamblado (2) [60].
- Prototipado Robot v1 - Ensamblado (3) [61].
- Prototipado Robot v1 - Ensamblado (4) [62].
- Prototipado Robot v2 - Comunicación Joystick Robot (1) [63].
- Prototipado Robot v2 - Comunicación Joystick Robot (2) [64].
- Prototipado Desplazamiento (alimentación USB) [65].
- Prototipado Desplazamiento (alimentación por pilas) [66].

### 4.5. Documentación del producto

Se desarrolló la documentación del producto compuesta de los siguientes entregables

- Documentación técnica [67].
- Manual de usuario [68].

## Capítulo 5

# Conclusiones

Conclusiones del trabajo...

- ¿Cuál es el grado de cumplimiento de los requerimientos?
- ¿Cuán fielmente se pudo seguir la planificación original (cronograma incluido)?
- ¿Se manifestó algunos de los riesgos identificados en la planificación? ¿Fue efectivo el plan de mitigación? ¿Se debió aplicar alguna otra acción no contemplada previamente?
- Si se debieron hacer modificaciones a lo planificado ¿Cuáles fueron las causas y los efectos?
- ¿Qué técnicas resultaron útiles para el desarrollo del proyecto y cuáles no tanto?

### 5.1. Próximos pasos

Acá se indica cómo se podría continuar el trabajo más adelante.



# Bibliografía

- [1] Latam Mining. *Robots y minería: Gobierno argentino quiere implementarlos.* URL: <https://www.latam-mining.com/robots-y-mineria-gobierno-argentino-quiere-implementarlos/>.
- [2] Diario de Cuyo. *Gobierno pone la mira en el desarrollo de robots para la actividad minera.* URL: <https://www.diariodecuyo.com.ar/politica/Gobierno-pone-la-mira-en-el-desarrollo-de-robots-para-la-actividad-minera-20200202-0052.html>.
- [3] Universidad Nacional de San Juan. *Robots en la minería.* URL: [http://www.unsj.edu.ar/home/noticias\\_detalles/4810/1](http://www.unsj.edu.ar/home/noticias_detalles/4810/1).
- [4] Ing. Nelson Dario García Hurtado e Ing. Melvin Andrés González Pino. *Robot de exploración terrestre Geobot.* URL: [https://www.unipamplona.edu.co/unipamplona/portalIG/home\\_40/recursos/01\\_general/revista\\_1/09102011/v01\\_09.pdf](https://www.unipamplona.edu.co/unipamplona/portalIG/home_40/recursos/01_general/revista_1/09102011/v01_09.pdf).
- [5] Ing. Hernán L. Helguero Velásquez1 e Ing. Rubén Medinaceli Tórrez. *Robot Minero: Sistema Detector de Gases utilizando Sensores en Tiempo Real MIN – SIS 1.0 SDG-STR.* URL: [http://www.scielo.org.bo/scielo.php?script=sci\\_arttext&pid=S2519-53522020000100003](http://www.scielo.org.bo/scielo.php?script=sci_arttext&pid=S2519-53522020000100003).
- [6] Boston Dynamics. *Spot.* URL: <https://www.bostondynamics.com/products/spot>.
- [7] Waygate Technologies. *BIKE - An advanced crawler robot for remote visual inspection.* URL: <https://www.bakerhughes.com/waygate-technologies/robotic-inspection/bike>.
- [8] Espressif. *ESP32.* URL: <https://www.espressif.com/en/products/socs/esp32>.
- [9] Espressif. *ESP32-WROOM-32D Datasheet.* URL: [https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32d\\_esp32-wroom-32u\\_datasheet\\_en.pdf](https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32d_esp32-wroom-32u_datasheet_en.pdf).
- [10] Mouser. *DHT11 datasheet.* URL: <https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf>.
- [11] Bosch. *BMP280 datasheet.* URL: <https://cdn-shop.adafruit.com/datasheets/BST-BMP280-DS001-11.pdf>.
- [12] Handson Technology. *PS2 Joy Stick for Arduino/Raspberry.* URL: <http://www.handsontec.com/dataspecs/accessory/PS2-Joystick.pdf>.
- [13] Espressif. *Analog to Digital Converter (ADC).* URL: <https://docs.espressif.com/projects/esp-idf/en/v4.4/esp32/api-reference/peripherals/adc.html>.
- [14] Handson Technology. *I2C Serial Interface 1602 LCD Module.* URL: [http://www.handsontec.com/dataspecs/module/I2C\\_1602\\_LCD.pdf](http://www.handsontec.com/dataspecs/module/I2C_1602_LCD.pdf).
- [15] Adafruit. *DC Gearbox Motor - TT Motor -200RPM - 3 to 6VDC.* URL: [https://media.digikey.com/pdf/Data%20Sheets/Adafruit%20PDFs/3777\\_Web.pdf](https://media.digikey.com/pdf/Data%20Sheets/Adafruit%20PDFs/3777_Web.pdf).

- [16] Handson Technology. *L298N Driver Module Datasheet*. URL: <https://www.handsontec.com/datasheets/L298N%20Motor%20Driver.pdf>.
- [17] EEMB. *Li-Ion batteries 18650 3000 mAh*. URL: <http://www.kosmodrom.com.ua/pdf/LIR18650-3000mah.pdf>.
- [18] Farnell. *Li-Ion batteries AA 2600 mAh*. URL: <https://www.farnell.com/datasheets/3195148.pdf>.
- [19] AOPUTTRIVER. *Anemómetro digital AOPUTTRIVER AP-007-WM*. URL: <https://manuals.plus/m/a30ffaa3ac8fd2cf06cb5559777c3af166bc当地18d1ee49a56ea6f10c66dd4b.pdf>.
- [20] CMake. *CMake*. URL: <https://cmake.org/>.
- [21] CMake. *Ninja*. URL: <https://cmake.org/cmake/help/latest/generator/Ninja.html>.
- [22] Espressif. *ESP-IDF Programming Guide | Get Started*. URL: <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/>.
- [23] Readthedocs by Ruslan V. Uss. *ESP-IDF Components library*. URL: <https://esp-idf-lib.readthedocs.io/en/latest/>.
- [24] Espressif Programing Guide. *ESP-IDF Get Started*. URL: <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/index.html>.
- [25] Docker. *Docker*. URL: <https://docker.com/>.
- [26] ThrowTheSwitch. *ThrowTheSwitch - Ceedling*. URL: <https://github.com/ThrowTheSwitch/Ceedling>.
- [27] ThrowTheSwitch. *ThrowTheSwitch - CMock*. URL: <https://github.com/ThrowTheSwitch/CMock>.
- [28] ThrowTheSwitch. *ThrowTheSwitch - Unity Test*. URL: <https://github.com/ThrowTheSwitch/Unity>.
- [29] ThrowTheSwitch. *ThrowTheSwitch - Ceedling/GCov*. URL: <https://github.com/ThrowTheSwitch/Ceedling/blob/master/plugins/gcov/README.md>.
- [30] Github. *Github*. URL: <https://github.com/>.
- [31] Google Cloud Platform. *Google Cloud Build*. URL: <https://cloud.google.com/build>.
- [32] Google Cloud Platform. *Google Artifact Registry*. URL: <https://cloud.google.com/artifact-registry>.
- [33] Visualstudio. *Visualstudio Code*. URL: <https://code.visualstudio.com/>.
- [34] Ubuntu. *Ubuntu*. URL: <https://ubuntu.com/>.
- [35] Espressif. *ESP-IDF WiFi*. URL: <https://docs.espressif.com/projects/esp-idf/en/stable/esp32/api-guides/wifi.html>.
- [36] Espressif. *ESP-IDF - Wi-Fi SoftAP Example*. URL: [https://github.com/espressif/esp-idf/tree/v4.4/examples/wifi/getting\\_started/softAP](https://github.com/espressif/esp-idf/tree/v4.4/examples/wifi/getting_started/softAP).
- [37] Gonzalo Carreno. *POC ESP32-WiFi v4.4*. URL: [https://github.com/kronleuchter85/cese\\_proyecto\\_especializacion/tree/main/pocs/esp32-wifi-ap-v4.4](https://github.com/kronleuchter85/cese_proyecto_especializacion/tree/main/pocs/esp32-wifi-ap-v4.4).
- [38] Espressif. *ESP-IDF ADC1 Example*. URL: <https://github.com/espressif/esp-idf/tree/v4.0.3/examples/peripherals/adc>.
- [39] Gonzalo Carreno. *POC ESP32-joystick*. URL: [https://github.com/kronleuchter85/cese\\_proyecto\\_especializacion/tree/main/pocs/esp32-joystick](https://github.com/kronleuchter85/cese_proyecto_especializacion/tree/main/pocs/esp32-joystick).

- [40] Gonzalo Carreno. *POC ESP32-photoresistor*. URL: [https://github.com/kronleuchter85/cese\\_proyecto\\_especializacion/tree/main/pocs/esp32-photoresistor](https://github.com/kronleuchter85/cese_proyecto_especializacion/tree/main/pocs/esp32-photoresistor).
- [41] UncleRus. *ESP32 - Example for dht driver*. URL: <https://github.com/UncleRus/esp-idf-lib/tree/master/examples/dht/default>.
- [42] Gonzalo Carreno. *POC ESP32-DHT11*. URL: [https://github.com/kronleuchter85/cese\\_proyecto\\_especializacion/tree/main/pocs/esp32-dht11](https://github.com/kronleuchter85/cese_proyecto_especializacion/tree/main/pocs/esp32-dht11).
- [43] UncleRus. *ESP32 - Example for bmp280 driver*. URL: <https://github.com/UncleRus/esp-idf-lib/tree/master/examples/bmp280/default>.
- [44] Gonzalo Carreno. *POC ESP32-BMP280*. URL: [https://github.com/kronleuchter85/cese\\_proyecto\\_especializacion/tree/main/pocs/esp32-bmp280](https://github.com/kronleuchter85/cese_proyecto_especializacion/tree/main/pocs/esp32-bmp280).
- [45] Espressif. *L298N Dual H-Bridge Motor Driver*. URL: <https://www.handsontec.com/dataspecs/L298N%20Motor%20Driver.pdf>.
- [46] Espressif. *ESP-IDF Motor Control Pulse Width Modulator (MCPWM)*. URL: <https://docs.espressif.com/projects/esp-idf/en/stable/esp32/api-reference/peripherals/mcpwm.html>.
- [47] Espressif. *MCPWM*. URL: <https://docs.espressif.com/projects/esp-idf/en/v4.2/esp32/api-reference/peripherals/mcpwm.html>.
- [48] Espressif. *MCPWM brushed dc motor control Example*. URL: [https://github.com/espressif/esp-idf/tree/v4.2/examples/peripherals/mcpwm/mcpwm\\_brushed\\_dc\\_control](https://github.com/espressif/esp-idf/tree/v4.2/examples/peripherals/mcpwm/mcpwm_brushed_dc_control).
- [49] Gonzalo Carreno. *POC ESP32-motor-pwm*. URL: [https://github.com/kronleuchter85/cese\\_proyecto\\_especializacion/tree/main/pocs/esp32-motor-pwm](https://github.com/kronleuchter85/cese_proyecto_especializacion/tree/main/pocs/esp32-motor-pwm).
- [50] ESP32 Tutoriales. *ESP32 I2C LCD with ESP-IDF*. URL: <https://esp32tutorials.com/i2c-lcd-esp32-esp-idf/>.
- [51] Gonzalo Carreno. *POC ESP32-DHT11*. URL: [https://github.com/kronleuchter85/cese\\_proyecto\\_especializacion/tree/main/pocs/esp32-display](https://github.com/kronleuchter85/cese_proyecto_especializacion/tree/main/pocs/esp32-display).
- [52] Espressif. *Espressif Docker Image*. URL: <https://docs.espressif.com/projects/esp-idf/en/stable/esp32/api-guides/tools/idf-docker-image.html>.
- [53] Gonzalo Carreno. *Robot de exploracion ambiental - Demo Medicion y visualizacion de parametros ambientales*. URL: [https://youtu.be/BBOP3n8\\_gBg](https://youtu.be/BBOP3n8_gBg).
- [54] Gonzalo Carreno. *Robot de exploracion ambiental - Demo Visualizacion del Display en la oscuridad*. URL: <https://youtu.be/LwfYaotAi64>.
- [55] Gonzalo Carreno. *Robot de exploracion ambiental - Demo Control de movimiento de las ruedas*. URL: <https://youtu.be/FKXWx4Rqr7I>.
- [56] Gonzalo Carreno. *Robot de exploracion ambiental - Demo Control de despazamiento en un circuito*. URL: <https://youtu.be/sosSGwCTyaY>.
- [57] Gonzalo Carreno. *Robot de exploracion ambiental - Demo Hardware*. URL: <https://youtu.be/RNBnDawVJ6c>.
- [58] Gonzalo Carreno. *Robot de exploracion ambiental - Demo - Comunicacion WiFi*. URL: <https://youtu.be/CcBgyoKjLB0>.
- [59] Gonzalo Carreno. *Robot exploracion ambiental - Prototipado Ensamblado Robot v1 (1)*. URL: <https://youtu.be/tDXt1CsObWE>.

- [60] Gonzalo Carreno. *Robot exploracion ambiental - Prototipado Ensamblado Robot v1* (2). URL: <https://youtube.com/shorts/uGqJn2K0LbI>.
- [61] Gonzalo Carreno. *Robot exploracion ambiental - Prototipado Ensamblado Robot v1* (3). URL: <https://youtu.be/w9IOoE-d9Cw>.
- [62] Gonzalo Carreno. *Robot exploracion ambiental - Prototipado Ensamblado Robot v1* (4). URL: [https://youtu.be/obkJ-wM\\_wNU](https://youtu.be/obkJ-wM_wNU).
- [63] Gonzalo Carreno. *Robot de exploracion ambiental - Prototipado Comunicacion Joystick Robot* (1). URL: <https://youtu.be/SnRf6HSya88>.
- [64] Gonzalo Carreno. *Robot de exploracion ambiental - Prototipado Comunicacion Joystick Robot* (2). URL: <https://youtu.be/jiisheyu95w>.
- [65] Gonzalo Carreno. *Robot de exploracion ambiental - Prototipado Desplazamiento (alimentacion USB)*. URL: [https://youtu.be/\\_w8qdNWC-DQ](https://youtu.be/_w8qdNWC-DQ).
- [66] Gonzalo Carreno. *Robot de exploracion ambiental - Prototipado Desplazamiento (alimentacion por pilas)*. URL: <https://youtu.be/-MxMXKzztHU>.
- [67] Gonzalo Carreno. *Robot de exploracion ambiental - Documentacion Tecnica*. URL: [https://github.com/kronleuchter85/cese\\_proyecto\\_especializacion/blob/main/docs/Documentacion-Tecnica.pdf](https://github.com/kronleuchter85/cese_proyecto_especializacion/blob/main/docs/Documentacion-Tecnica.pdf).
- [68] Gonzalo Carreno. *Robot de exploracion ambiental - Manual de usuario*. URL: [https://github.com/kronleuchter85/cese\\_proyecto\\_especializacion/blob/main/docs/Manual-De-Usuario-vFinal.pdf](https://github.com/kronleuchter85/cese_proyecto_especializacion/blob/main/docs/Manual-De-Usuario-vFinal.pdf).