UNIVERSIDAD NACIONAL DE MOQUEGUA
ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
FACULTAD DE ARQUITECTURA E INGENIERÍA

Proyecto Final

Robótica II

Carrito Bombero Aplicando Teleoperación Híbrida

Docente:
Mgr. Yessica Rosas Cuevas

Estudiantes:
Gianfranco Flores Vasquez
Leopoldo Contreras Melendrez
Gustavo Huailla Ramos

Fecha de entrega: 09 de Julio del 2025

Índice de contenido

1. Problemática **4**

2. Solución propuesta **4**

3. Objetivo **4**
3.1. Objetivo general . . . 4
3.2. Objetivos específicos . . . 4

4. Justificación **4**

5. Recursos Necesarios **5**

6. Estado del Arte **5**

7. Marco Teórico **9**
7.1. Robótica Móvil . . . 9
7.2. Modelo cinemático de un robot móvil diferencial . . . 10
7.3. Detección de Incendios . . . 12
7.3.1. Sistemas de Detección de Incendios . . . 12
7.3.2. Sistema de Extinción Automatizado . . . 13
7.4. Hardware . . . 13
7.4.1. Microcontrolador Arduino UNO . . . 13
7.5. Comunicación inalámbrica . . . 14
7.5.1. Módulo Bluetooth HC-06 . . . 14
7.5.2. Teleoperación híbrida . . . 16
7.6. Sensado de proximidad en robótica móvi . . . 18
7.6.1. Sensor ultrasónico HC-SR04 . . . 18
7.7. Aplicaciones móviles con MIT App Inventor . . . 19

8. Diseño de Propuesta **20**
8.1. Requerimientos funcionales . . . 20
8.2. Requerimientos no funcionales . . . 21
8.3. Arquitectura del sistema . . . 21
8.4. Arquitectura de Comunicacion . . . 22
8.5. Diagrama del Circuito Electrónico . . . 23
8.6. Implementación y Desarrollo . . . 25
8.6.1. Montaje físico del prototipo . . . 25
8.6.2. Desarrollo de la Aplicación en MIT App Inventor . . . 28
8.6.3. Programación del Prototipo . . . 30

9. Pruebas y Resultados **35**
9.1. Pruebas . . . 35
9.2. Resultados . . . 36

10. Conclusiones **38**

11.Trabajos Futuros **38**

12.Referencias bibliográficas **40**

Índice de figuras

1. Robot Diferencial 10
2. Sistemas de deteccion de incendios 12
3. Pines del microcontrolador Arduino UNO 14
4. Pines del Modulo Bluetooth HM-10 15
5. Pines del sensor ultrasonico hc-sr04 18
6. Componentes de la interfaz de APP Inventor 19
7. Diagrama de circuito Electrico 23
8. Prototipo - Vista Lateral 1 26
9. Prototipo - Vista Superior 27
10. Prototipo - Vista Lateral 2 27
11. Prototipo - Vista Frontal 28
12. Interfaz de la aplicacion 29
13. Prueba App n°1 37
14. Prueba App n°2 38

1. Problemática

Cuando el incendio ocurre en lugares de difícil acceso o sin presencia humana, la respuesta puede tardar demasiado, permitiendo que el fuego se propague. Aunque existen sistemas automáticos de extinción (como rociadores), estos suelen ser costosos o inaccesibles para familias, pequeños negocios o instituciones educativas. Por ello, es necesario encontrar soluciones tecnológicas accesibles, seguras y funcionales que permitan actuar de forma rápida, remota y automática ante la presencia de fuego.

2. Solución propuesta

Como alternativa a esta problemática real, se plantea el desarrollo de un prototipo funcional de robot bombero móvil, que simula una intervención temprana frente a incendios de baja escala. Este prototipo combina sensores, actuadores y control remoto para detectar flamas y activar un sistema de aspersión de agua.

El robot está basado en una configuración de robot diferencial, controlada por un **Arduino UNO** y dirigida mediante teleoperación híbrida. Utiliza **tres sensores de flama** para detectar fuego en el entorno. Al identificar una fuente de calor, activa una **bomba de agua** mediante un **relé**, lanzando agua en la dirección del fuego con ayuda de un **servomotor SG90**. También incluye un **sensor ultrasónico** para evitar colisiones con obstáculos.

3. Objetivo

3.1. Objetivo general

Desarrollar un prototipo funcional de robot bombero con teleoperación híbrida, capaz de detectar y extinguir focos de incendio mediante sensores de flama y un sistema de aspersión de agua automatizado.

3.2. Objetivos específicos

1. Diseñar la estructura móvil del robot con configuración diferencial.
2. Integrar los sensores de flama al sistema de control mediante Arduino UNO.
3. Implementar el sistema de aspersión con bomba, relé y servomotor SG90.
4. Desarrollar la aplicación Android para la teleoperación por Bluetooth.

4. Justificación

Este proyecto surge como una respuesta tecnológica y educativa frente a una problemática real la falta de intervención rápida en los primeros minutos de un incendio. Al tratarse de un prototipo, su principal

función es demostrar que es posible construir un sistema autónomo y remoto de detección y respuesta frente al fuego utilizando componentes electrónicos de bajo costo.

Diversos estudios han demostrado la viabilidad del uso de sensores de flama, módulos de comunicación inalámbrica y microcontroladores para la construcción de robots bombero de bajo costo. Por ejemplo, en el artículo Fire-Extinguishing Robot Design by Using Arduino (2020), se describe un prototipo que utiliza sensores de flama para detectar fuego, una bomba de agua activada por un relé y un servomotor para orientar la dirección del chorro de agua, todo controlado por un Arduino UNO y un módulo Bluetooth HC-05. Este diseño es muy similar al propuesto en nuestro proyecto, lo cual valida la efectividad de los componentes seleccionados y respalda técnicamente la funcionalidad del prototipo.

5. Recursos Necesarios

- 01 Estructura de robot 2WD (chasis de acrílico + motores + ruedas + interruptor + porta baterías)
- 01 Arduino UNO
- 01 Placa de expansión
- 01 Módulo Bluetooth HM-10
- 01 Driver L298N
- 03 Sensores de flama YG1006
- 01 Servomotor SG90
- 01 Bomba de agua 12v
- 01 Relé de 1 canal
- 02 Pilas 18650
- Cables
- 01 Sensor ultrasónico

6. Estado del Arte

Con el objetivo de sustentar el presente proyecto en antecedentes relevantes, se realizó una revisión documental de 12 trabajos entre artículos científicos, tesis y proyectos similares enfocados en el desarrollo de robots móviles para la detección y extinción de incendios. Esta revisión permitió identificar las tecnologías más utilizadas, los componentes electrónicos comunes, los enfoques de control implementados, así como los principales logros y limitaciones de cada propuesta.

A continuación, se presenta una tabla comparativa que resume los aspectos más importantes de cada uno de estos trabajos, permitiendo visualizar de manera clara las similitudes, diferencias y aportes que

distinguen el presente proyecto. La tabla incluye los siguientes campos: título, componentes principales, aporte del trabajo, resultados obtenidos y limitaciones identificadas.

Título	**Componentes principales**	**Aporte**	**Resultados**	**Limitaciones**
Robot automático de búsqueda y extinción de incendios uso de sensores basados en llama y ultrasonidos arduino uno (2024)	Arduino Uno, controlador de motor L298N, sensor de llama, sensor ultrasónico, motor de CC, motor de ventilador L9110 (Wemos D1 R1)	Diseñar un robot que pueda ayudar a los humanos a encontrar y extinguir incendios rápidamente y así prevenirlos.	Cuantos más obstáculos detecte, más tiempo tardará el robot en encontrarlo (1m - 2.7min), Cuanto más cerca esté el robot del foco, más rápido se extinguirá el incendio. (3cm - 6seg)	Solo cuenta con un sensor ultrasónico, No cuenta con cámara para ver condiciones del entorno
Robot detector y extintor de incendios basado en Arduino (2020)	Arduino Uno, sensor ultrasónico, sensor de llama, tanque de agua, Circuito controlador, Motores CC, driver L293D, Receptor IR, microcontrolador Atmega 328p	Mejorar la eficiencia de los bomberos y evitar arriesgar vidas humanas. mecanismo de rociado	Prototipo funcional para implementar el robot de extinción de incendios	Falta de cámara inalámbrica (imágenes y video para visualizar), falta de un controlador para la bomba de agua (consumo de corriente)
Diseño de sistemas automáticos de detección de incendios y Dispositivos de extinción con Arduino (2023)	Arduino Mega 2560, detector de llamas infrarrojo, sensor ultrasónico HC-SR04, rociado de agua con motor CC	Mitigación de Riesgos Humanos, Diseño y Prueba de un Prototipo Funcional, Base para Proyectos Futuros Desarrollados, Identificación y Prevención de Incendios	Detección de Incendios, Evitación de obstáculos, Extinción de Incendios, Diseño Integrado, Movimiento Automático	Cobertura de detección, Rociado unidireccional, Falta de Localización/-Navegación Avanzada, Entorno de Operación Limitado (pasillos)

Tabla 1
Tabla comparativa de artículos sobre robots de extinción de incendios

Título	**Componentes principales**	**Aporte**	**Resultados**	**Limitaciones**
Robot bombero de control inalámbrico y autónomo. (2021)	Motorreductor de CC (12 V), Bomba de CC (12 V), Pantalla LCD (20 x 4), Servomotor, sensor ultrasónico, Módulo Bluetooth (HC05), Detector de fuego, ATmega 16, ATmega 8, Conductor de motor, Cámara FPV, Transmisor FPV, receptor FPV, Batería de lipo (11 v), Regulador de voltaje (7805)	Reducir masivamente, incluso eliminar, cualquier peligro que un ser humano pueda enfrentar al combatir incendios y participar en operaciones de búsqueda y rescate.	El control inalámbrico y el modo autónomo permite al robot detectar y extinguir incendios eficientemente.	El alcance del sensor ultrasónico es de 10-15 pies, lo cual podría ser limitado en áreas grandes.
Robot de extinción de incendios basado en Arduino. (2021)	Arduino UNO, sensor ultrasónico, sensor de llama, Sensor de temperatura (LM35), Sensor de gas (MQ3), Relé, MCU del nodo, Tanque de agua, Control remoto inalámbrico, dispositivo inalámbrico Android, cámara con WI-FI	Agrega nuevas características que hacen que sea más realista reconocer la gravedad del incendio y la forma de los gases presentes, lo que es fundamental para evitar una mayor propagación del incendio.	El sistema desarrollado agrega características que hacen más realista reconocer la gravedad del incendio y la forma de los gases presentes. El uso de sensores de temperatura y gas, junto con el sensor de llama, permite una mejor evaluación de la situación del incendio.	Enfatiza las nuevas características añadidas y su potencial, sugiriendo que el enfoque está en las mejoras respecto a otros sistemas.
Desarrollo e implementación de Robot extintor de incendios de modo dual basado en un microcontrolador Arduino (2017)	Arduino UNO, detector de llama LM393 (2 largo y 1 corto alcance), módulo Bluetooth, detector de obstáculos HC-SR04, cámara de vigilancia, driver L293D, motores CC, Bomba de agua, Tanque de agua	Robot de extinción de modo dual que trabaja tanto en modo automático como en modo manual.	La cámara de vigilancia instalada en el robot permite supervisar su movimiento en modo manual y obtener vigilancia en tiempo real en modo automático.	Para dar mayor precisión sería de agregar sensores de temperatura y sensores de gas para acercarlo más a la realidad.

Tabla 2
Comparativa de artículos sobre robots extintores de incendios (parte 2)

Título	**Componentes principales**	**Aporte**	**Resultados**	**Limitaciones**
Diseño y construcción de sistemas automáticos de extinción de incendios Robot de combate con notificación por SMS (2022)	Arduino Nano, módulo GSM Sim900A Shield, Módulo GPS Neo6m, Sensor térmico, Sensores IR, IC controlador de motor L293D, Relé, Motores de doble eje, Mini bomba de agua sumergible de CC, Adaptador	El robot automático de extinción de incendios con notificación por SMS.	El robot incorpora módulos GSM y GPS que envían un SMS al usuario para alertarle de un incendio e indicar su ubicación. Utiliza sensores térmicos e infrarrojos para detectar el incendio, se acerca a él y lo rocía con agua desde una distancia segura.	Los sistemas actuales de detección de humo son limitados: no identifican incendios pequeños ni su ubicación exacta, y su cobertura para extinción es insuficiente, lo que puede llevar a respuestas tardías y consecuencias graves.
Robot automático contra incendios con Arduino (2025)	Arduino UNO, sensores de llama, sensores ultrasónicos, servomotor, bomba de agua, Módulo controlador de motor (L298N), módulo de relé, Chasis y ruedas	Ofrecer un sistema de automatización de seguridad práctico y económico.	Los sensores de llama mostraron una buena respuesta en un rango de 60 a 80 cm. Las pruebas demostraron una extinción eficaz de llamas pequeñas de alcohol y velas.	No cuenta con cámara termográfica, Falta integración con IA, Navegación avanzada SLAM, Vigilancia remota IoT.
Diseño y Fabricación de Sistema Robótico Autónomo de Extinción de Incendios Equipado con Sensores sensibles para alarma y detección de incendios, mecanismo de comportamiento de evitación y Capacidad de mensajería SMS (2020)	Arduino Mega 2560, sensor de llama, sensor de humo, sensor ultrasónico HC-SR04, altavoz piezoeléctrico, LED RGB KYX016, Módem GSM SIM900A, Mini bomba sumergible 12v, motor CC y ruedas, Servo Digital SG90, driver L298N	El sistema diseñado es más rentable y debería emplearse en la lucha contra incendios para limitar el número de muertes y proteger edificios e infraestructura.	El tiempo de respuesta óptimo para la parada de movimiento es de 400 ms, lo que indica que el robot comenzará a extinguir. La alarma de detección de incendios y humo, así como la función de mensajería SMS, se activaron correctamente al detectar fuego y humo.	Se puede agregar cámara de vigilancia para analizar el entorno.

Tabla 3
Comparativa de artículos sobre robots extintores de incendios (parte 3)

Título	Componentes principales	Aporte	Resultados	Limitaciones
Implementación de un prototipo de sistema autónomo de detección y extinción de incendios. (2023)	Servomotor (SG90), microcontrolador ATMEGA328P, sensores de incendio, mini bomba sumergible de CC, módulo convertidor elevador, motores de CC, Algoritmo inteligente	Solución prometedora para la aplicación de extinción de incendios. Elimina la necesidad de intervención humana en situaciones peligrosas, haciéndolo ideal para escenarios que representan riesgos significativos para la vida humana.	El desarrollo de un algoritmo inteligente que discrimina correctamente entre incendios y condiciones no amenazantes o engañosas, sirviendo como validador para detectar falsas alarmas.	Se aborda la limitación de los sistemas tradicionales que requieren intervención humana y no pueden operar en entornos peligrosos.
Implementación de robot contra incendios tipo tanque rover para áreas cerradas basado en microcontrolador Arduino (2021)	Arduino UNO, sensor de llama, bomba de agua, servomotor de CC, módulo Bluetooth HC06, cámara WiFi, sensor infrarrojo pasivo (PIR), driver L9110, buzzer	Este robot extingue incendios en interiores difíciles de alcanzar, usando sensores de llama y PIR para detectar fuego y vida. Su cámara WiFi permite control remoto y un servomotor direcciona el agua, mostrando alta eficiencia en pruebas.	Dos sensores de llama y un sensor PIR se utilizaron para detectar la presencia de fuego y el movimiento de organismos vivos. El robot se controla de forma inalámbrica mediante el módulo Bluetooth HC-06 y la cámara Wi-Fi.	Se podría añadir un sensor de temperatura para tener mayor precisión al detectar el nivel de calor.
Robot automático de extinción de incendios con notificación. (2019)	Arduino Mega 2560, Escudo de motor Adafruit V2, Sensor ultrasónico HC-SR04, Sensor de llama, Módulo Bluetooth HC05, Tanque de agua, Bomba de agua, Motores	Envío de notificaciones mediante el módulo Bluetooth. Respuesta rápida y reducción de daños, operación en entornos peligrosos	Está equipado con tres sensores ultrasónicos para evitar obstáculos, lo que protege al robot y sus componentes internos. El robot se moverá aleatoriamente por la habitación. Detecta un incendio: el robot se acerca al foco y se detiene para rociar agua y notifica al usuario.	Lleva mucho tiempo encontrar la fuente de agua para extinguir el fuego, lo que puede provocar una rápida propagación del mismo y aumentar las muertes.

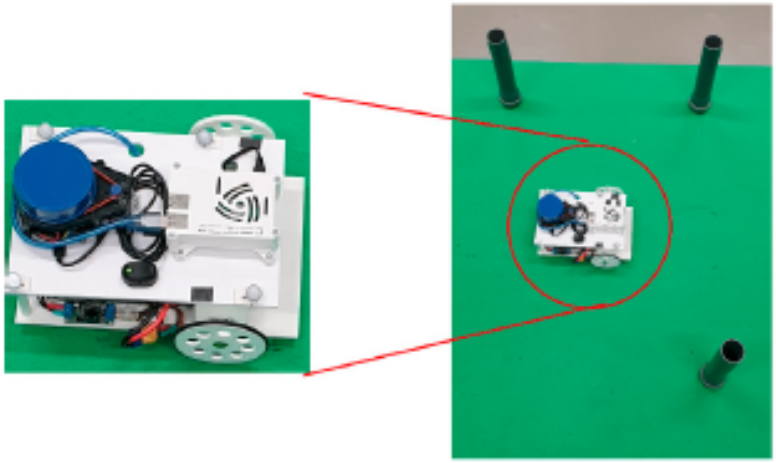
Tabla 4
Comparativa de artículos sobre robots extintores de incendios (parte 4)

7. Marco Teórico

7.1. Robótica Móvil

La robótica móvil se refiere a sistemas robóticos capaces de desplazarse en su entorno físico de forma autónoma o controlada. En nuestro caso, el *Carrito Bombero* es un robot móvil terrestre que emplea una configuración de tracción diferencial. Según Moreira et al. (2020), un robot móvil con tracción diferencial puede integrar controladores y sensores como LIDAR y odometría para mejorar su capacidad de navegación autónoma.

Imagen 1
Robot Diferencial



Este tipo de configuración es ideal para espacios cerrados o reducidos, como viviendas o aulas, donde la maniobrabilidad es crucial.

Elementos clave del sistema:

- **Arduino UNO:** Microcontrolador que actúa como el cerebro del sistema.
- **Driver L298N:** Permite controlar la dirección y velocidad de los motores.
- **Motores DC:** Impulsan las ruedas del robot.
- **Chasis 2WD:** Estructura básica con dos ruedas y un punto de apoyo.

7.2. Modelo cinemático de un robot móvil diferencial

El modelo cinemático permite describir el movimiento de un robot móvil en función de sus velocidades lineales y angulares, sin considerar las fuerzas que lo generan. En robots con tracción diferencial, como el prototipo del *Carrito Bombero*, este modelo es fundamental para comprender y predecir su desplazamiento en un entorno plano.

Descripción general del modelo: Un robot diferencial cuenta con dos ruedas motrices independientes ubicadas en los laterales del chasis. La combinación de sus velocidades angulares determina el comportamiento del robot: puede avanzar, retroceder, girar o describir trayectorias curvas.

La cinemática del robot se expresa mediante el siguiente conjunto de ecuaciones:

$$\dot{x} = \frac{r}{2}(\omega_R + \omega_L)\cos(\theta)$$
$$\dot{y} = \frac{r}{2}(\omega_R + \omega_L)\sin(\theta)$$
$$\dot{\theta} = \frac{r}{L}(\omega_R - \omega_L)$$

Donde:

- r es el radio de las ruedas.
- L es la distancia entre las ruedas izquierda y derecha.
- ω_R, ω_L son las velocidades angulares de las ruedas derecha e izquierda, respectivamente.
- (x, y) es la posición del robot en el plano.
- θ es la orientación del robot con respecto al eje horizontal.

Generación y significado del error cinemático En el contexto cinemático, el **error** es la diferencia entre el estado actual del robot y el estado objetivo o deseado. Se puede descomponer en:

- **Error de posición:** diferencia entre la ubicación real del robot (x, y) y un punto de referencia (x_d, y_d).
- **Error de orientación:** diferencia entre la orientación actual θ y la orientación deseada θ_d.

Estos errores se generan naturalmente cuando el robot intenta alcanzar una meta específica o seguir una trayectoria.

$$e_{\text{pos}} = \sqrt{(x_d - x)^2 + (y_d - y)^2} \qquad e_\theta = \theta_d - \theta$$

Minimizar estos errores es esencial para que el robot navegue de forma precisa y eficiente.

Valor deseado y trayectoria de referencia El **valor deseado** representa el objetivo cinemático del robot. Puede tratarse de:

- Un punto destino fijo en el entorno.
- Una trayectoria predeterminada que el robot debe seguir.
- Una orientación específica hacia un objeto o zona (por ejemplo, una fuente de fuego).

La comparación constante entre el estado actual y el estado deseado permite al sistema ajustar sus movimientos para lograr el comportamiento esperado.

Salida del modelo cinemático La **salida del modelo** se refiere a las velocidades lineales y angulares que produce el robot como resultado del movimiento de sus ruedas:

$$u(t) = \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \end{bmatrix}$$

Estas velocidades determinan cómo se desplaza el robot en el espacio y permiten evaluar si su comportamiento se ajusta a lo planificado.

7.3. Detección de Incendios

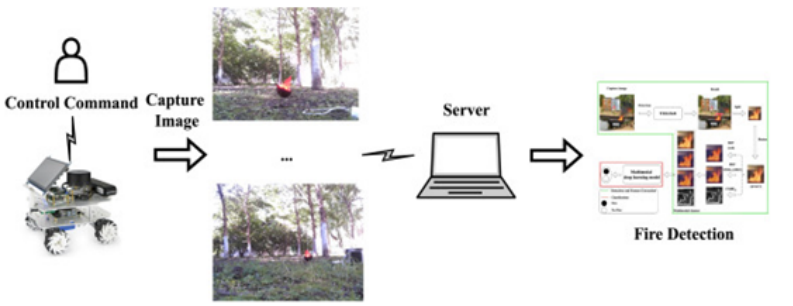
La detección temprana de fuego es crítica para evitar la propagación y minimizar daños. Nuestro prototipo implementa un sistema simple pero eficiente de detección basado en sensores de flama.

7.3.1. Sistemas de Detección de Incendios

Existen diversos tipos de sistemas de detección de incendios: térmicos, ópticos, de humo y de flama. En este proyecto se emplean sensores ópticos de flama KY-026, los cuales detectan la radiación infrarroja emitida por el fuego.

En cuanto a la detección, estudios recientes confirman que la combinación de sensores RGB, infrarrojos e incluso ultravioleta mejora la precisión y reduce las falsas alarmas frente a métodos tradicionales (Guo et al., 2024).

Imagen 2
Sistemas de deteccion de incendios



Características de los sensores KY-026:

- Detectan flamas en un rango aproximado de 0 a 80 cm.
- Alta sensibilidad al espectro infrarrojo típico de una llama (760 – 1100 nm).
- Incluyen un comparador que activa una señal digital al detectar fuego.

Disposición: Se colocan tres sensores de flama en la parte frontal del robot para ampliar el campo de visión y permitir una detección más precisa de la ubicación del incendio. Esto también facilita orientar correctamente la dirección del sistema de extinción.

7.3.2. Sistema de Extinción Automatizado

Una vez detectado el fuego, el sistema debe responder rápidamente para contenerlo. El desarrollo de robots inteligentes que integran navegación autónoma, sensores múltiples y sistemas automáticos de extinción ha demostrado ser una solución efectiva en entornos de alto riesgo. Estas plataformas pueden inspeccionar de forma autónoma, detectar incendios con precisión y activar mecanismos extintores sin necesidad de intervención humana directa (Xiang et al., 2022).

Componentes del sistema de extinción:

- **Bomba de agua 12V:** Actúa como el sistema principal para expulsar el agua.
- **Relé de 1 canal:** Funciona como un interruptor que permite activar la bomba desde el Arduino.
- **Servomotor SG90:** Dirige la boquilla del agua hacia la fuente del fuego, permitiendo precisión en la extinción.
- **Tanque de agua (opcional):** Contiene el líquido extintor que será expulsado por la bomba.

Funcionamiento:

1. Al detectar fuego, el Arduino activa automáticamente el relé, encendiendo la bomba.
2. Simultáneamente, el servomotor gira hacia la dirección del sensor que detectó el fuego.
3. Se lanza el chorro de agua hacia la llama hasta que se detecta su extinción o el usuario detenga la acción desde la aplicación.

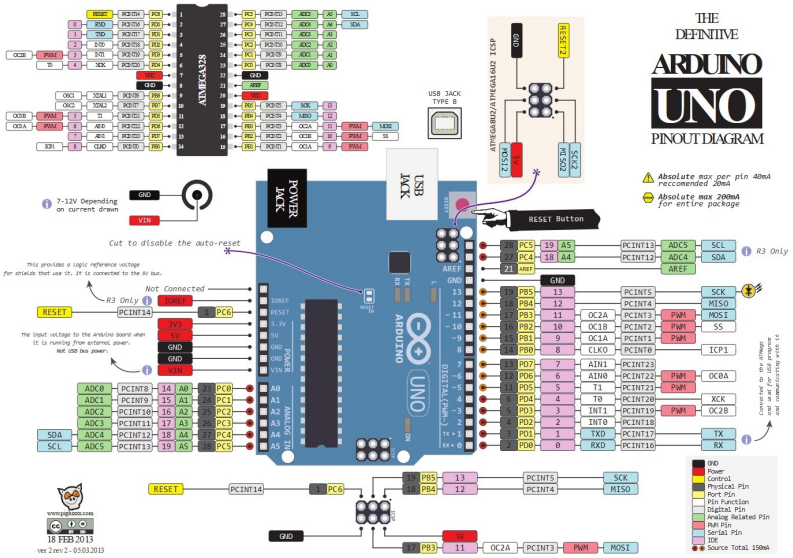
7.4. Hardware

7.4.1. Microcontrolador Arduino UNO

El Arduino UNO es una plataforma de desarrollo de hardware libre basada en el microcontrolador ATmega328P de 8 bits. Ha sido diseñada para facilitar el prototipado electrónico interactivo, permitiendo la lectura de entradas (como sensores) y el control de salidas (como motores o actuadores), mediante un lenguaje de programación basado en C/C++. La placa cuenta con 14 pines digitales de entrada/salida (6 con capacidad PWM), 6 entradas analógicas, un oscilador de 16 MHz, una interfaz USB para programación y alimentación, además de un conector de alimentación externa, un header ICSP y un botón de reinicio. Estas características hacen del Arduino UNO una opción robusta y ampliamente adoptada en entornos educativos, de investigación y prototipado industrial (Sunil, s.f.).

Imagen 3

Pines del microcontrolador Arduino UNO



El estudio de Prabowo e Irwanto (2023), basado en 1 122 artículos indexados en Scopus, demuestra que el Arduino UNO es la plataforma más utilizada para aplicaciones educativas, industriales y de automatización, debido a su equilibrio entre facilidad de uso, estabilidad y versatilidad. Su adopción en entornos académicos ha promovido el desarrollo de proyectos de robótica, control automático, monitoreo ambiental y sistemas embebidos en general.

Además, Rossi et al. (2020) lo aplicaron exitosamente como controlador en un vehículo autónomo con navegación basada en visión computacional, evidenciando que su capacidad de respuesta es adecuada incluso en entornos dinámicos y críticos como la robótica móvil.

7.5. Comunicación inalámbrica

7.5.1. Módulo Bluetooth HC-06

1. Descripción general

El HM-10 es un módulo de comunicación inalámbrica basado en Bluetooth 4.0 BLE (Bluetooth Low Energy). Está diseñado para operar en modo esclavo (slave) y permite la transmisión de datos de forma eficiente entre un microcontrolador (como Arduino UNO) y un dispositivo maestro (como un teléfono Android).

Aunque se trata de un módulo BLE, la conexión con el Arduino se realiza mediante una interfaz UART TTL, lo que lo hace compatible con la mayoría de placas de desarrollo. Gracias a su bajo consumo energético y su alcance confiable, el HM-10 es ideal para aplicaciones como control de robots, automatización del hogar, y monitoreo inalámbrico.

Protocolo de Comunicación Utilizado

El HM-10 utiliza el protocolo UART (Universal Asynchronous Receiver-Transmitter) para establecer la comunicación con el Arduino. Esta interfaz serial asíncrona transmite datos bit a bit a través de los pines TX y RX, sin requerir una señal de reloj compartida.

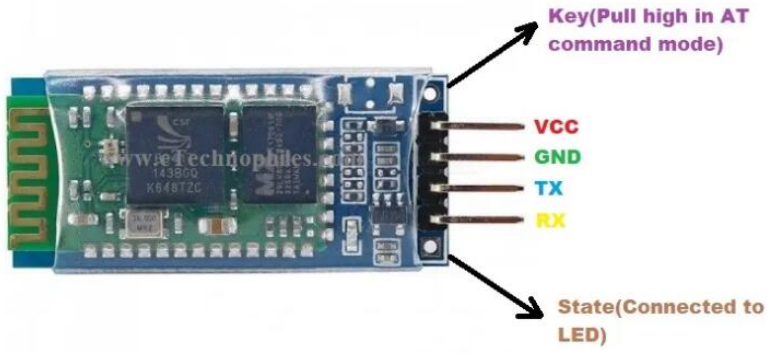
Aunque internamente el HM-10 utiliza tecnología BLE, para el Arduino se comporta como una conexión serial clásica, permitiendo recibir y enviar datos de forma sencilla mediante las funciones **Serial.read()** y **Serial.print()** en el entorno de programación de Arduino.

2. Configuración de pines

El módulo HM-10 generalmente incluye **cuatro pines funcionales**, los cuales se conectan directamente al microcontrolador:

Pin	Función
VCC	Alimentación (acepta de 3.3 V a 6 V en algunos módulos con regulador)
GND	Tierra común (referencia)
TXD	Transmisión de datos desde el módulo al microcontrolador (nivel 3.3 V)
RXD	Recepción de datos desde el microcontrolador al módulo (nivel 3.3 V)

Imagen 4
Pines del Modulo Bluetooth HM-10



Algunos modelos también incluyen un pin **STATE** que indica visualmente el estado de conexión (por ejemplo, con un LED integrado), y un pin **KEY** para ingresar al modo de comandos AT en modelos compatibles.

3. Configuración inicial del HM-10

Según el tutorial de RucksikaaR (2021), la configuración inicial del HC-06 es directa y requiere:

- Conectar **VCC** del módulo a 5 V del Arduino (si el módulo tiene regulador).

- Conectar **GND** a GND del Arduino.
- Conectar **TXD** del HM-10 al pin **RX** (pin 0) del Arduino.
- Conectar **RXD** del HM-10 al pin **TX** (pin 1) del Arduino mediante un **divisor de voltaje** (por ejemplo, usando resistencias de 1 kΩ y 2 kΩ).

Una vez conectado correctamente, el LED del módulo parpadeará rápidamente mientras espera emparejamiento. Desde el smartphone, puede emparejarse usando un PIN por defecto (como 1234 o 0000). Una vez emparejado, el módulo se comporta como una conexión serial inalámbrica transparente.

4. Funcionamiento y aplicaciones

Una vez energizado, el HM-10 comienza a emitir una señal BLE visible desde dispositivos Android compatibles. El emparejamiento se realiza mediante una aplicación BLE o mediante extensiones BLE en plataformas como MIT App Inventor. El módulo se conecta utilizando una dirección específica y un PIN por defecto **(como 000000 o 123456 dependiendo del firmware)**.

Después de emparejado, se establece un canal de comunicación bidireccional que permite al usuario controlar el robot y, opcionalmente, recibir datos de sensores.

Aplicaciones Comunes

- Control remoto de robots mediante BLE..
- Monitorización de sensores desde teléfonos móviles.
- Comunicación inalámbrica en proyectos con bajo consumo de energía.

7.5.2. Teleoperación híbrida

La **teleoperación híbrida** combina control remoto manual con modos de autonomía reactiva, alternando entre la operación directa del usuario y respuestas automatizadas del robot ante condiciones críticas. Este enfoque ofrece un balance entre control humano e intervención autónoma segura.

Aldhafeeri et al. (2024) presentan un esquema híbrido para teleoperación bilateral en robots manipuladores móviles holonómicos, usando un dispositivo háptico en tierra. Su sistema cambia automáticamente entre modos de navegación manual y manipulación automática cuando el efector entra en contacto con el entorno, sin sensores externos, logrando una transición con alta transparencia en menos de 0,3 s.

Pappas et al. (2020) introducen un enfoque de **control compartido (shared control)** para robots móviles en entornos peligrosos, combinando comandos remotos de un operador con un módulo autónomo de evasión de obstáculos (VFH+). Este método mejora significativamente la seguridad y eficiencia frente a la teleoperación pura.

Tabla comparativa de articulos mencionados

Autor	**Aldhafeeri et al. (2024)** *TAROS*	**Pappas et al. (2020)** *SSRR*
Título completo	*A New Hybrid Teleoperation Control Scheme for Holonomic Mobile Manipulator Robots Using a Ground-based Haptic Device*	*VFH+ based shared control for remotely operated mobile robots*
Tipo de robot	Robot manipulador móvil holonómico con interfaz háptica terrestre	Robot móvil teleoperado en entornos peligrosos, controlado con joypad
Enfoque	Teleoperación híbrida con conmutación automática basada en detección de contacto sin sensores externos	Control compartido: mezcla señales del operador y VFH+ para evadir obstáculos dinámicos
Modo de cambio de control	Automático al detectar contacto; retorno manual tras tarea	Fusión continua: el módulo VFH+ interviene sólo cuando es necesario durante teleoperación
Sensores requeridos	Utiliza un sensor háptico en el efector del brazo robotico	Utiliza sensores de proximidad para detección de obstáculos y VFH+
Interfaz de usuario	Dispositivo háptico que proporciona fuerzas artificiales para limitar workspace	Joypad estándar para teleoperación manual
Resultados principales	Transición con alta transparencia en $<0{,}3\,\mathrm{s}$. Reduce fuerzas irrelevantes comparadas con técnicas tipo “bubble” ($>50\,\%$)	En escenarios de desastre: mejora en seguridad y tiempo de completación frente a teleoperación pura
Ventajas destacadas	- Mantiene transparencia háptica - Reducción de cargas mentales - Conmutación rápida y fluida	- Seguridad mejorada - Eficiencia aumentada - Adaptable a entornos dinámicos
Limitaciones	Estudio realizado en simulación, requiere validación en hardware real	Evaluado en escenario experimental de respuesta a desastres; falta evaluación en manipulación compleja

Ventajas del enfoque híbrido

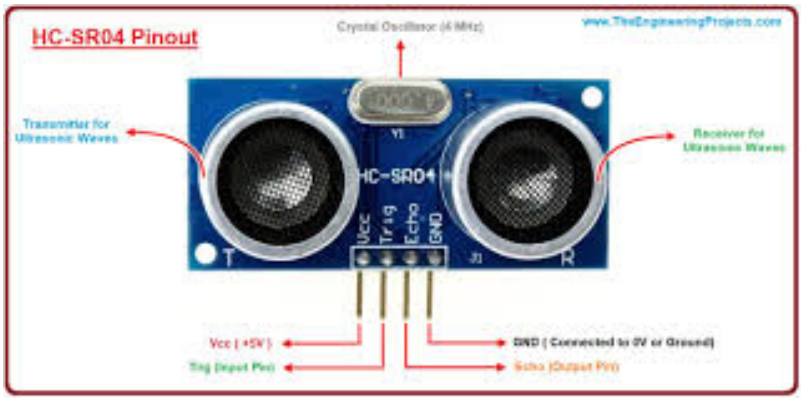
Ventaja	Justificación científica
Seguridad aumentada	La lógica automática responde rápidamente, sin esperar al operador.
Reducción de carga operativa	El operador solo actúa en situaciones no críticas.
Respuesta reactiva rápida	Conmutación instantánea entre modos sin interrupciones.
Adaptabilidad del sistema	Se puede modificar lógica de sensores sin cambiar interfaz Bluetooth.

7.6. Sensado de proximidad en robótica móvi

7.6.1. Sensor ultrasónico HC-SR04

El HC-SR04 es un sensor ultrasónico de medición de distancia que opera mediante la emisión de pulsos de ultrasonido a una frecuencia de 40 kHz y el posterior cálculo del tiempo que tarda el eco en regresar tras reflejarse en un objeto. El módulo cuenta con dos pines principales: TRIG, que dispara el pulso ultrasónico, y ECHO, que registra el tiempo de retorno. A partir del tiempo transcurrido, se puede calcular la distancia con una resolución de aproximadamente 3 mm, dentro de un rango operativo de 2 cm a 400 cm, bajo condiciones normales de temperatura y presión. Este principio de funcionamiento se basa en la velocidad del sonido en el aire y en una lógica de temporización digital estándar (ElecFreaks, 2013).

Imagen 5
Pines del sensor ultrasonico hc-sr04



Jabines y Perin (2022) desarrollaron un sistema autónomo de evasión de obstáculos usando un HC-SR04 y una placa Arduino Uno. Sus resultados demuestran que este sensor, debido a su bajo costo y fiabilidad, es una opción viable para sistemas robóticos móviles que requieren navegación segura en entornos variables.

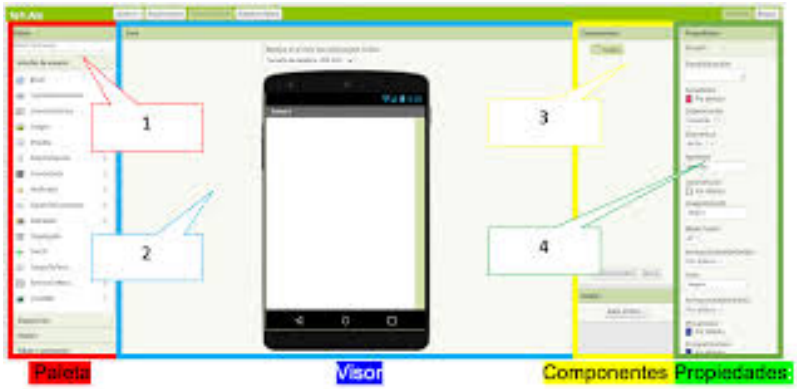
Por otro lado, Zapata (2025) calibró el HC-SR04 para medir la velocidad de un pequeño vehículo móvil. Sus experimentos arrojaron un error medio absoluto de solo 0.40 cm, evidenciando que el dispositivo puede

utilizarse también para estimaciones cinemáticas con fines de control de movimiento o posicionamiento en tiempo real.

7.7. Aplicaciones móviles con MIT App Inventor

MIT App Inventor es una plataforma visual y gratuita que permite *desarrollar rápidamente aplicaciones Android interactivas* sin necesidad de escribir código tradicional. Utiliza un editor de diseño de interfaz y un editor de bloques lógicos (tipo Blockly) ideal para proyectos educativos y de robótica.

Imagen 6
Componentes de la interfaz de APP Inventor



1. Interfaz gráfica para control Bluetooth

Se puede diseñar una aplicación con botones tipo `"Forward"`, `"Backward"`, `"Left"`, `"Right"`, `"Stop"`, y botones para `conectar` o `desconectar` el módulo HM-10.

En la interfaz (*Designer*), se añaden componentes visibles (botones, etiquetas, contenedores de disposición) y componentes no visibles como `BluetoothClient`, `Notificador` y `Clock`.

2. Lógica con bloques (editor Blocks)

Al presionar el botón `Connect`, la app muestra los dispositivos Bluetooth disponibles (list picker), permite seleccionar el HM-10 y realizar la conexión usando: `BluetoothClient.ConnectAddress`.

Cada botón direccional envía una letra específica (`"F"`, `"B"`, `"L"`, `"R"`, `"S"`) mediante: `BluetoothClient.SendText`.

El Arduino, conectado por UART al HM-10, interpreta estos caracteres para activar motores u otros actuadores.

3. Recepción de datos y feedback

La aplicación puede leer datos desde Arduino usando: `BluetoothClient.ReceiveText`, y mostrar valores en la pantalla (etiquetas o indicadores). Esto permite, por ejemplo, visualizar sensores o estado del sistema en tiempo real.

4. Arquitectura manual–automática en la app

Según Top & Gökbulut (2022), se desarrolló una app creada con App Inventor que permite al usuario elegir entre **modo manual** (control directo de movimiento vía botones) o **modo automático** (posición o velocidad predefinidas desde la app).

La elección se realiza en la pantalla principal y se comunica al Arduino vía Bluetooth con comandos especiales.

5. Ventajas principales

- **Facilidad de uso:** Su entorno gráfico permite crear aplicaciones mediante arrastrar y soltar componentes, lo cual es ideal para principiantes sin experiencia en programación.
- **Desarrollo rápido:** Permite probar las aplicaciones en tiempo real utilizando la app *AI2 Companion*, lo que acelera el proceso de prueba y depuración.
- **Accesible y gratuita:** Es una plataforma de código abierto y sin costo, lo que democratiza el desarrollo de aplicaciones móviles.
- **Versatilidad:** Permite integrar sensores, conectividad Bluetooth, bases de datos, GPS y elementos multimedia, siendo especialmente útil para proyectos robóticos como el *Carrito Bombero*.

6. Ejemplos prácticos

- El instructable *"Build a Bluetooth Robot W/ Arduino & MIT App Inventor"* muestra un robot móvil controlado completamente desde una app creada en App Inventor y conectada a Arduino mediante un HM-10.
- La experiencia *"My Droid Robot controlled by App Inventor"* narra el caso de un robot educativo controlado vía Bluetooth desde una app desarrollada en App Inventor. Marcelo Vila de Oliveira logró comunicar sensores y actuadores bidireccionalmente entre dispositivo y robot usando esta plataforma con Arduino y HM-10.

8. Diseño de Propuesta

8.1. Requerimientos funcionales

1. **El sistema debe detectar la presencia de fuego** mediante tres sensores de flama distribuidos al frente del robot.
2. **El robot debe activar automáticamente la bomba de agua** cuando se detecte fuego, con ayuda de un relé.
3. **El usuario debe poder controlar el movimiento del robot** (adelante, atrás, izquierda, derecha) desde una aplicación Android vía Bluetooth.

4. **El robot debe alternar entre dos modos de funcionamiento:** automático (responde al fuego por sí solo) y manual (controlado por el usuario).
5. **El sensor ultrasónico debe detectar obstáculos frontales** y enviar esta información al Arduino para evitar colisiones.

8.2. Requerimientos no funcionales

1. **Alcance de comunicación:** La conexión Bluetooth entre la aplicación Android y el robot debe mantenerse estable dentro de un rango mínimo de 10 metros en espacios abiertos.
2. **Autonomía operativa:** El sistema debe mantener un funcionamiento continuo mínimo de 20 minutos con dos baterías 18650 completamente cargadas.
3. **Facilidad de uso:** La interfaz de la aplicación Android debe ser sencilla e intuitiva, con botones claramente identificados para controlar el movimiento y la activación del sistema extintor.
4. **Compatibilidad de comunicación:**El módulo Bluetooth HM-10 debe utilizar el protocolo UART sobre Bluetooth 4.0 BLE, garantizando una conexión estable con dispositivos Android .

8.3. Arquitectura del sistema

La arquitectura del sistema del Carrito Bombero con teleoperación híbrida se organiza en tres capas principales: presentación, lógica y datos. Cada una cumple un rol específico en el funcionamiento del prototipo y se comunica de forma integrada para asegurar el control eficiente del robot ante la presencia de fuego.

Capa de presentación

Esta capa representa la interfaz gráfica de usuario, la cual fue desarrollada con **MIT App Inventor**, una plataforma de desarrollo visual que permite crear aplicaciones Android de manera rápida e intuitiva.

- La app permite al usuario enviar comandos de movimiento (adelante, atrás, izquierda, derecha, detenerse) y cambiar entre modo **manual o automático**.
- La interfaz se compone del *App Inventor Designer*, donde se agregan botones y elementos visuales, y del *editor de bloques (Blocks Editor)*, donde se define la lógica de comportamiento de cada botón.
- La comunicación con el hardware se realiza mediante **envío de datos seriales vía Bluetooth BLE** al módulo **HM-10**, conectado físicamente al Arduino UNO.

Capa de lógica

En esta capa se lleva a cabo la **interacción entre el microcontrolador Arduino UNO y el módulo Bluetooth HM-10**, utilizando el protocolo de comunicación **UART (Universal Asynchronous Receiver-Transmitter)**.

- El HM-10 se conecta a los pines RX y TX del Arduino UNO, estableciendo un canal de comunicación serial de baja energía.
- El robot recibe las instrucciones del usuario (por ejemplo, ”F”, ”B”, ”L”, ”R”, ”S”) desde la aplicación móvil, interpretadas por el código programado en Arduino.
- También se gestiona la lógica de decisión cuando el robot está en **modo automático**, permitiendo la detección de fuego y activación del sistema extintor de forma autónoma.

Capa de datos

Esta capa está relacionada con los componentes de bajo nivel que ejecutan las acciones **físicas** del sistema.

- El **módulo Bluetooth HM-10**, basado en el chip CC2541, permite la recepción de comandos en formato ASCII utilizando comandos AT preconfigurados.
- El **Arduino UNO** procesa las instrucciones recibidas y controla directamente los actuadores:
  - Motores (a través del driver L298N) para movimiento.
  - Bomba de agua (a través de un relé).
  - Servomotor SG90 para dirigir la boquilla extintora.
- Esta capa también puede enviar información de retorno hacia la app, como estado del sistema o sensores activos, permitiendo retroalimentación en tiempo real.

Flujo general de funcionamiento:

1. El usuario interactúa con la app móvil y selecciona comandos.
2. El módulo **HM-10** transmite los comandos mediante UART al **Arduino UNO**.
3. El Arduino interpreta y ejecuta las instrucciones, activando motores o actuadores.
4. En **modo automático**, el sistema responde sin intervención directa del usuario.

8.4. Arquitectura de Comunicacion

La comunicación estará basada en un esquema sencillo que utiliza Bluetooth BLE 4.0 mediante el módulo HM-10, facilitando la conexión entre el robot y una aplicación móvil en Android.

Componentes Involucrados

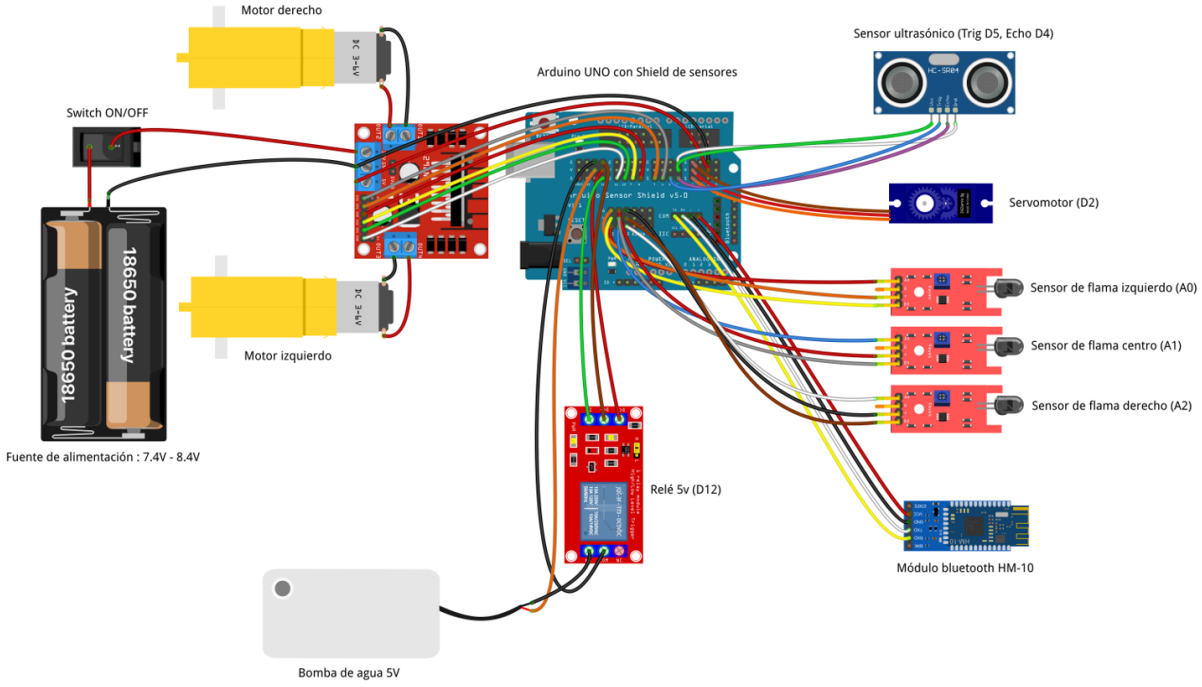
- **Módulo Bluetooth HM-10:**
  Es un pequeño dispositivo que permite al robot recibir instrucciones desde el teléfono móvil por medio de Bluetooth de baja energía (BLE). Funciona como un receptor que escucha los comandos enviados por la aplicación móvil y los transmite al microcontrolador (Arduino UNO) mediante la interfaz UART (TX y RX).

- **Microcontrolador (Arduino UNO):**
  Es el *cerebro* del robot. Recibe las órdenes que le manda el celular a través del módulo Bluetooth y se encarga de hacer que el robot actúe, como mover los motores.
- **Aplicación móvil en Android :**
  Es una app sencilla que sirve para controlar el robot desde el celular.
- **Protocolo de comunicación UART:**
  Aunque el módulo HM-10 utiliza tecnología Bluetooth 4.0 BLE, la comunicación con el microcontrolador se realiza mediante el protocolo UART (Universal Asynchronous Receiver-Transmitter). Este protocolo permite la transmisión de datos en serie, utilizando los pines TX y RX. Cada paquete de datos contiene bits de inicio, datos y parada, lo que asegura una sincronización confiable entre el HM-10 y el Arduino, siguiendo un formato que incluye bits de inicio, datos y parada para garantizar que ambos dispositivos se comprendan correctamente (Gong, Guo, Sun, 2023).

8.5. Diagrama del Circuito Electrónico

El siguiente diagrama muestra la **conexión electrónica completa** del prototipo del **Carrito Bombero con teleoperación híbrida**, implementado mediante un **Arduino UNO**, un **Shield de sensores v5.0** y diversos componentes como motores, sensores, actuadores y el módulo Bluetooth HM-10.

Imagen 7
Diagrama de circuito Electrico



Este diseño integra todos los módulos necesarios para cumplir con las funcionalidades tanto manuales como automáticas del sistema, permitiendo detectar fuego, moverse en diferentes direcciones, evitar obstáculos

y activar la bomba de agua.

Descripción de conexiones principales:

- **Fuente de alimentación:**
  - Dos baterías 18650 conectadas en serie (7.4 V–8.4 V) alimentan todo el sistema.
  - Conectadas a través de un **switch ON/OFF** para facilitar el encendido del robot.
- **Módulo de potencia (Driver L298N):**
  - Controla los **motores izquierdo y derecho** de tracción diferencial.
  - Se conecta directamente al Shield de sensores (conexiones a IN1, IN2, IN3, IN4 y ENA/ENB).
  - La alimentación del driver proviene de las baterías.
- **Sensor ultrasónico (HC-SR04):**
  - Utilizado para **detección de obstáculos** en modo automático.
  - Conectado a los pines `Trig (D5)` y `Echo (D4)`.
- **Sensores de flama (3 unidades YG1006):**
  - Detectan la presencia de fuego en el entorno del robot.
  - Conectados a las entradas `A0`, `A1` y `A2` del Arduino (izquierda, centro y derecha, respectivamente).
- **Servomotor SG90:**
  - Utilizado para **dirigir el chorro de agua** hacia la fuente de fuego.
  - Conectado al pin `D2` del Arduino.
- **Módulo relé de 5V:**
  - Controla el encendido de la **bomba de agua** de forma segura.
  - El relé se activa desde el pin `D12` del Arduino.
- **Módulo Bluetooth HM-10:**
  - Permite la comunicación inalámbrica entre el robot y la aplicación Android.
  - Conectado a través de `TX` y `RX` al Shield de sensores (niveles de 3.3 V).

Consideraciones adicionales:

- El uso del **Shield de sensores v5.0** facilita la organización de conexiones y permite una rápida prototipación.

- Todas las conexiones de señal, alimentación y tierra están claramente separadas para evitar interferencias eléctricas.
- La **bomba de agua de 5 V** se activa únicamente cuando se detecta fuego, garantizando eficiencia energética y respuesta automática.

8.6. Implementación y Desarrollo

8.6.1. Montaje físico del prototipo

El montaje físico del **Carrito Bombero con teleoperación híbrida** se llevó a cabo sobre un chasis acrílico de dos ruedas motrices (2WD). A continuación, se describe detalladamente la disposición física de los elementos y sus conexiones:

Estructura general del robot

El chasis acrílico cuenta con:

- **Dos ruedas laterales** impulsadas por motores DC de 3–6 V, ubicadas en los costados traseros.
- Una **rueda loca metálica** en la parte trasera para equilibrar el movimiento y permitir giros suaves.
- La parte superior del chasis fue empleada para fijar los principales módulos electrónicos y de control.

Distribución de componentes

Disposicion de los componentes:

- **Arduino UNO** montado encima del driver L298N junto con un **Sensor Shield v5.0**, el cual organiza eficientemente todas las conexiones eléctricas.
- **Driver L298N**, ubicado en parte trasera debajo del arduino y atras del contenedor de agua, controla el giro y velocidad de los motores DC conectados a cada rueda.
- **Relé de 5V**, ubicado en la parte media derecha del robot al costado del contenedor de agua, encargado de activar la bomba de agua.
- **Módulo Bluetooth HM-10**, montado de manera visible en la parte superior.
- **Dos baterías 18650** colocadas de forma segura en su portapilas , con **interruptor ON/OFF** para control de energía ubicadas al frente del contenedor de agua.
- **Bomba de agua 5V**, ubicada en el contenedor de agua, conectada mediante una manguera a la boquilla frontal.
- **Servomotor SG90**, sujeto al frente del chasis, encargado de mover la boquilla extintora horizontalmente.
- **Sensor ultrasónico HC-SR04**, montado junto al servomotor, centrado para detectar obstáculos al frente.

- **Tres sensores de flama** ubicados en línea en la parte frontal: uno izquierdo, uno central y uno derecho, que amplían el campo de visión para la detección de fuego.

Organización y cableado

- El cableado del sistema se realizó utilizando **jumpers tipo Dupont**, conectados al Shield para facilitar el montaje .
- La manguera transparente de salida de agua fue fijada sobre la estructura frontal con cintillos, alineada con el servomotor para facilitar el giro automático.
- El módulo Bluetooth se posicionó estratégicamente hacia el exterior para mejorar la recepción de la señal desde la aplicación Android.

Imagen 8
Prototipo - Vista Lateral 1

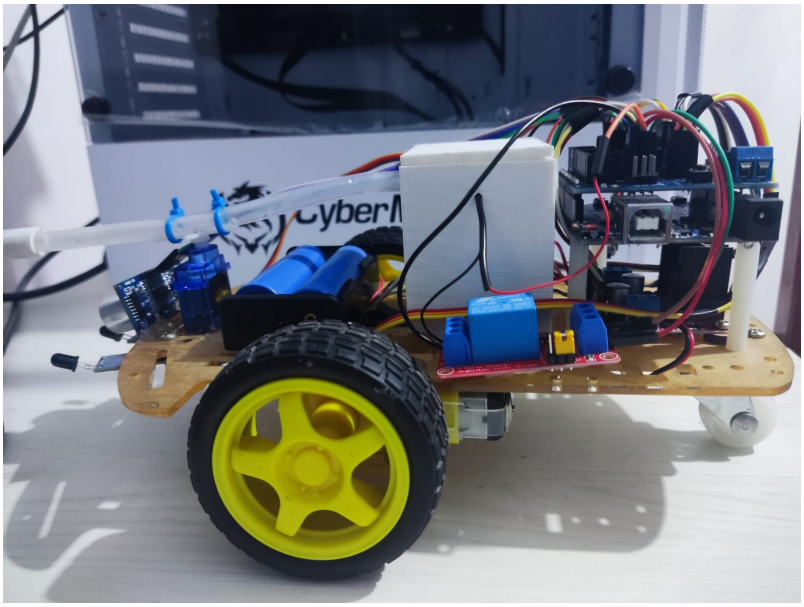


Imagen 9
Prototipo - Vista Superior

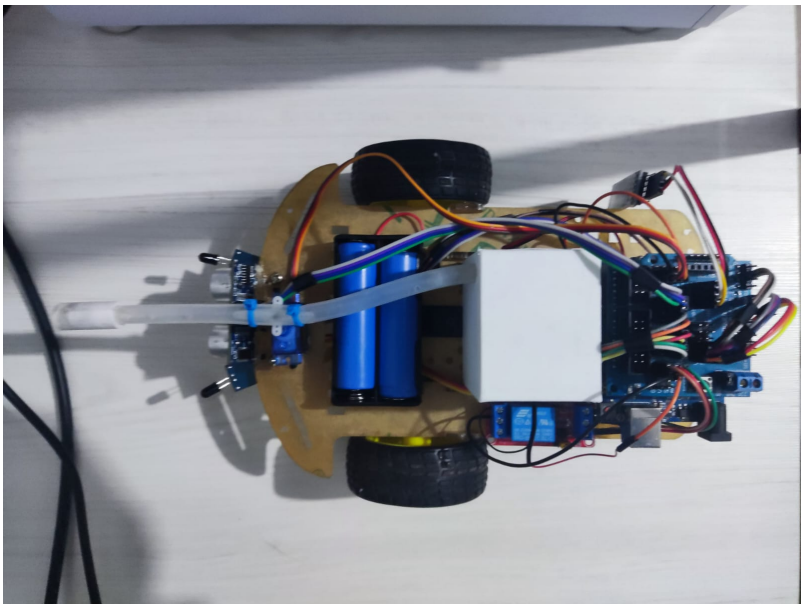


Imagen 10
Prototipo - Vista Lateral 2

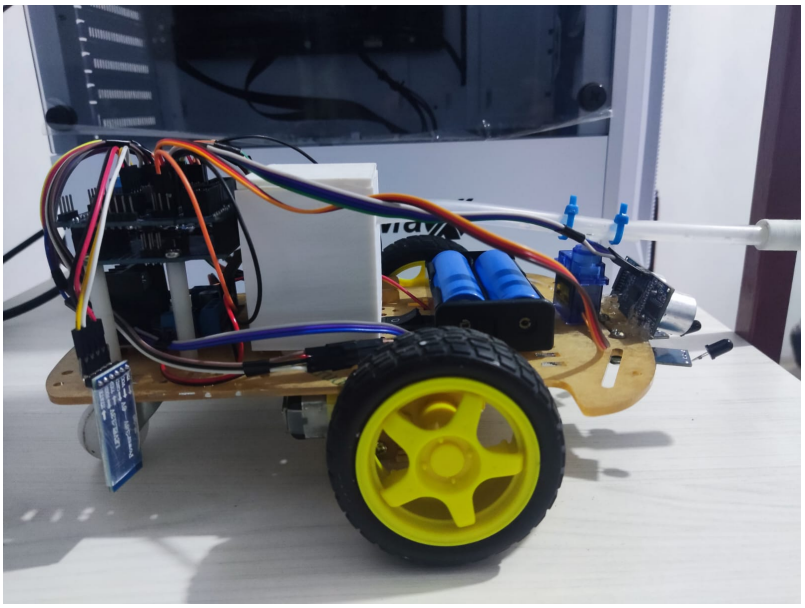
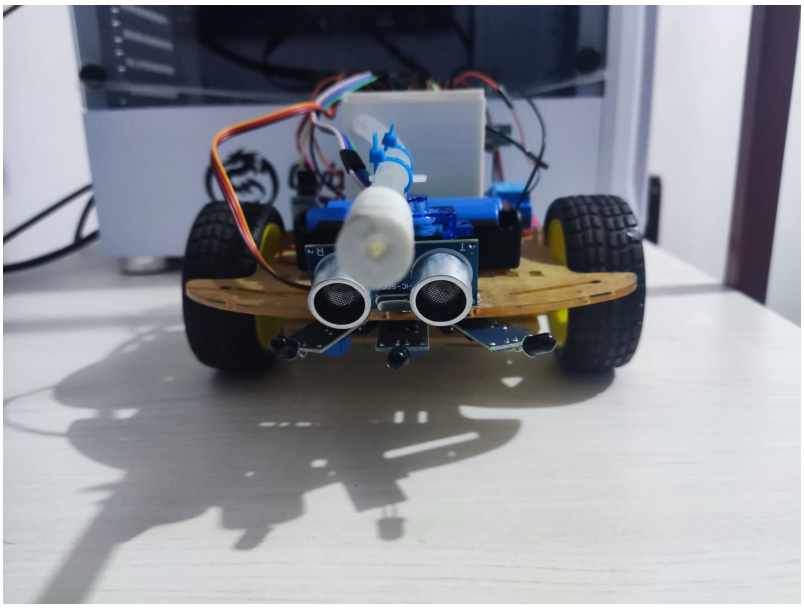


Imagen 11
Prototipo - Vista Frontal



8.6.2. Desarrollo de la Aplicación en MIT App Inventor

Para permitir el control remoto del prototipo robótico y la visualización de datos en tiempo real, se desarrolló una aplicación móvil utilizando la plataforma **MIT App Inventor**, compatible con dispositivos Android. Esta app se conecta vía **Bluetooth BLE** al módulo **HM-10**, lo que permite una comunicación inalámbrica eficiente y de bajo consumo.

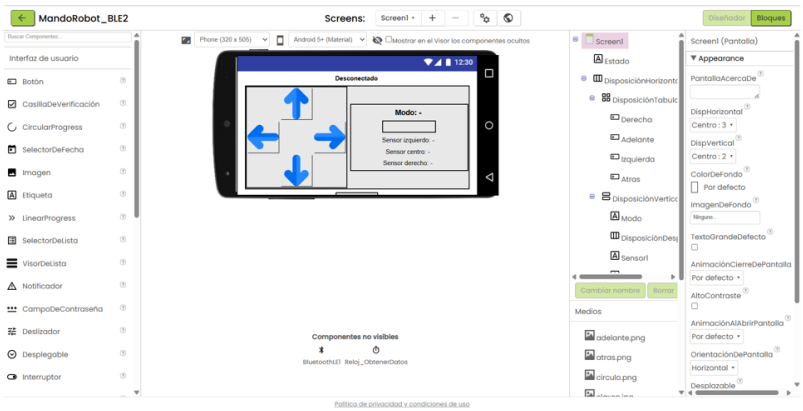
Interfaz de usuario

La interfaz de la aplicación se diseñó de forma intuitiva, dividiendo la pantalla en dos secciones principales:

- **Control de dirección (modo manual):** A la izquierda, se ubican los botones de flechas que permiten enviar comandos al robot para moverse en las direcciones: adelante, atrás, izquierda y derecha. Estos botones activan el movimiento al presionarlos y lo detienen al soltarlos.
- **Visualización de sensores y estado:** A la derecha, se muestra el modo actual de operación del carrito (manual o automático) y los valores de los sensores de flama izquierdo, centro y derecho, permitiendo así un monitoreo constante de su entorno.

En la parte superior se incluye una etiqueta que indica si el dispositivo está **conectado** o **desconectado** al módulo HM-10.

Imagen 12
Interfaz de la aplicacion



Conexión vía Bluetooth BLE

Se utilizó el componente `BluetoothLE1` para gestionar la comunicación con el módulo Bluetooth BLE HM-10. Al iniciar la aplicación o al presionar el botón "Conectar", se realiza un escaneo de dispositivos cercanos. Una vez encontrado el módulo HM-10, se establece la conexión mediante su dirección MAC.

El estado de conexión se refleja dinámicamente en la app:

- Si se conecta correctamente, el estado cambia a **"Conectado"**, y se habilita el botón "Desconectar".
- Si se pierde la conexión, se actualiza a **"Desconectado"**, volviendo a habilitar el botón "Conectar".

Envío de comandos de control

La app implementa un sistema de control manual que permite enviar comandos específicos a través del BLE:

Acción del usuario	Comando enviado
Adelante (presionar)	A
Atrás (presionar)	S
Izquierda (presionar)	I
Derecha (presionar)	D
Cualquier botón (soltar)	T

Tabla 5
Comandos enviados por la app al robot

Estos comandos son interpretados por el Arduino para mover el robot en la dirección deseada y detenerlo cuando se suelta el botón.

Cada comando se transmite a través del método `.WriteStrings` utilizando los identificadores `serviceUuid` y `characteristicUuid` correspondientes al protocolo BLE del módulo HM-10.

Recepción y visualización de datos

Para la lectura de información enviada por el Arduino, se implementó un temporizador (`Reloj_ObtenerDatos`) que consulta periódicamente si el dispositivo está conectado. Si es así, se ejecuta la función `.ReadStrings`, la cual recibe datos en el siguiente formato:

`sensorIzq,sensorCentro,sensorDer,modo`

Estos datos se separan por comas y se asignan a las etiquetas correspondientes en la interfaz. Adicionalmente, se interpreta el modo (`a` para automático y `m` para manual), cambiando su representación visual y el color de fondo:

- **Automático:** Color Verde, texto "Automático".
- **Manual:** Color amarillo, texto "Manual".

8.6.3. Programación del Prototipo

El funcionamiento del **Carrito Bombero con teleoperación híbrida** fue implementado mediante un programa desarrollado en el entorno **Arduino IDE**, utilizando lenguaje C++. El código controla todos los módulos integrados: sensores de flama, sensor ultrasónico, bomba de agua, servomotor, motores de tracción y comunicación Bluetooth BLE (HM-10).

El siguiente código implementa el comportamiento completo del *Carrito Bombero*, incluyendo control por Bluetooth, detección de fuego y activación automática del sistema extintor:

```
#include <Servo.h>

const int UMBRAL = 875;
const int UMBRAL_CERCA = 400;

const int ENA = 6, IN1 = 7, IN2 = 8, IN3 = 9, IN4 = 10, ENB = 11;
const int BOMBA = 12;
const int IZQ = A0, CENTRO = A1, DER = A2;
const int trigPin = 5, echoPin = 4;

Servo servo;
int sensor_izquierdo, sensor_centro, sensor_derecho;
String modo_actual = "bluetooth";

unsigned long t_anterior = 0;
const unsigned long intervalo = 1000;

void setup() {
  pinMode(ENA, OUTPUT); pinMode(IN1, OUTPUT); pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT); pinMode(IN4, OUTPUT); pinMode(ENB, OUTPUT);
```

```
21    pinMode(BOMBA, OUTPUT);
22    pinMode(IZQ, INPUT); pinMode(CENTRO, INPUT); pinMode(DER, INPUT);
23    servo.attach(2);
24    digitalWrite(BOMBA, LOW);
25    pinMode(trigPin, OUTPUT); pinMode(echoPin, INPUT);
26    Serial.begin(9600);
27  }
28
29  void loop() {
30    sensor_izquierdo = analogRead(IZQ);
31    sensor_centro = analogRead(CENTRO);
32    sensor_derecho = analogRead(DER);
33
34    if (sensor_izquierdo < UMBRAL || sensor_centro < UMBRAL || sensor_derecho < UMBRAL) {
35      modo_actual = "auto";
36      seguirFuego();
37    } else {
38      controlBluetooth();
39    }
40
41    if ((millis() - t_anterior) >= intervalo) {
42      enviarDatosBT(sensor_izquierdo, sensor_centro, sensor_derecho, modo_actual);
43      t_anterior = millis();
44    }
45    delay(50);
46  }
47
48  void adelante() {
49    digitalWrite(IN1, HIGH); digitalWrite(IN2, LOW);
50    digitalWrite(IN3, LOW);  digitalWrite(IN4, HIGH);
51    analogWrite(ENA, 140); analogWrite(ENB, 110);
52  }
53
54  void atras() {
55    digitalWrite(IN1, LOW);  digitalWrite(IN2, HIGH);
56    digitalWrite(IN3, HIGH); digitalWrite(IN4, LOW);
57    analogWrite(ENA, 140); analogWrite(ENB, 110);
58  }
59
60  void izquierda() {
61    digitalWrite(IN1, HIGH); digitalWrite(IN2, LOW);
62    digitalWrite(IN3, HIGH); digitalWrite(IN4, LOW);
63    analogWrite(ENA, 100); analogWrite(ENB, 100);
64  }
65
66  void derecha() {
67    digitalWrite(IN1, LOW);  digitalWrite(IN2, HIGH);
68    digitalWrite(IN3, LOW);  digitalWrite(IN4, HIGH);
69    analogWrite(ENA, 100); analogWrite(ENB, 100);
70  }
```

```
71
72  void detener() {
73    digitalWrite(IN1, LOW); digitalWrite(IN2, LOW);
74    digitalWrite(IN3, LOW); digitalWrite(IN4, LOW);
75    analogWrite(ENA, 0); analogWrite(ENB, 0);
76  }
77
78  void controlBluetooth () {
79    if (modo_actual != "bluetooth") {
80      detener();
81      modo_actual = "bluetooth";
82    }
83    if (Serial.available()) {
84      char comando = Serial.read();
85      switch (comando) {
86        case 'A': adelante(); break;
87        case 'S': atras(); break;
88        case 'I': izquierda(); break;
89        case 'D': derecha(); break;
90        case 'T': detener(); break;
91      }
92    }
93  }
94
95  void seguirFuego() {
96    int distancia = obtenerDistancia();
97    if (distancia > 0 && distancia < 20) { detener(); apagarFuego(); return; }
98
99    if (sensor_izquierdo < UMBRAL_CERCA || sensor_centro < UMBRAL_CERCA || sensor_derecho
          < UMBRAL_CERCA) {
100     detener(); apagarFuego(); return;
101   }
102
103   if ((sensor_izquierdo < UMBRAL && sensor_izquierdo > UMBRAL_CERCA) ||
104       (sensor_centro < UMBRAL && sensor_centro > UMBRAL_CERCA) ||
105       (sensor_derecho < UMBRAL && sensor_derecho > UMBRAL_CERCA)) {
106     int minValor = sensor_centro;
107     adelante();
108     if (sensor_izquierdo < minValor) { minValor = sensor_izquierdo; izquierda(); }
109     if (sensor_derecho < minValor) { minValor = sensor_derecho; derecha(); }
110   } else { detener(); }
111 }
112
113 void apagarFuego() {
114   digitalWrite(BOMBA, HIGH);
115   for (int i = 90; i < 110; i++) { servo.write(i); delay(10); }
116   for (int i = 110; i > 70; i--) { servo.write(i); delay(10); }
117   for (int i = 70; i < 90; i++) { servo.write(i); delay(10); }
118   digitalWrite(BOMBA, LOW);
119 }
```

```
120
121   int obtenerDistancia() {
122     long duration;
123     float distance;
124     digitalWrite(trigPin, LOW); delayMicroseconds(2);
125     digitalWrite(trigPin, HIGH); delayMicroseconds(10);
126     digitalWrite(trigPin, LOW);
127     duration = pulseIn(echoPin, HIGH, 30000);
128     distance = duration * 0.0343 / 2;
129     return distance;
130   }
131
132   void enviarDatosBT(int s1, int s2, int s3, String modo) {
133     String buffer = String(s1) + "," + String(s2) + "," + String(s3) + "," + ((modo == "
          auto") ? "a" : "m") + "\0";
134     Serial.print(buffer);
135   }
```

Estructura general del programa

El programa se divide en las siguientes secciones:

- **Inicialización (`setup()`)**: Configura los pines, inicializa el puerto serial y el servomotor, y deja apagada la bomba de agua al inicio.
- **Bucle principal (`loop()`)**: Evalúa si hay fuego (lectura de sensores) y decide si activar el modo automático. Si no hay fuego, entra en modo Bluetooth. También se encarga del envío periódico de datos hacia la aplicación móvil.
- **Funciones auxiliares**: Cada acción del robot está contenida en funciones específicas, facilitando la lectura del código y la reutilización.

Funciones clave del sistema

El siguiente cuadro resume las funciones implementadas en el código, junto con su propósito:

Función	**Descripción y propósito**
`setup()`	Inicializa pines, servo, sensor ultrasónico y Bluetooth. Es el punto de arranque del prototipo.
`loop()`	Ejecuta continuamente la lógica principal. Cambia de modo según detección de fuego y envía datos a la app.
`adelante()`, `atras()`, `izquierda()`, `derecha()`, `detener()`	Controlan los motores mediante el driver L298N. Son utilizados en ambos modos de operación.
`controlBluetooth()`	Interpreta los comandos (A, S, I, D, T) enviados por la app y ejecuta movimientos. Activa el modo "bluetooth" si no hay fuego.
`seguirFuego()`	Lógica de modo automático. Evalúa qué sensor detecta fuego y mueve el robot en esa dirección. Si el fuego está muy cerca, activa la bomba.
`apagarFuego()`	Activa la bomba de agua (vía relé) y realiza un barrido con el servomotor para cubrir la zona afectada.
`obtenerDistancia()`	Calcula la distancia mediante el sensor ultrasónico. Evita acercamientos bruscos al fuego.
`enviarDatosBT()`	Envía a la app móvil los valores de los sensores y el modo actual del robot (manual o automático).

Tabla 6
Funciones clave del sistema y su propósito

Modo automático

El prototipo evalúa continuamente los sensores de flama. Si alguno detecta fuego (valor analógico < 875), el sistema entra en **modo automático**. El robot se dirige hacia el sensor con mayor proximidad al fuego y, si se encuentra a menos de 20 cm o supera el umbral de cercanía, se detiene y activa la bomba para apagar el foco de incendio.

Durante esta fase, el sistema toma decisiones autónomas y reporta constantemente su estado a la app.

Modo manual (Bluetooth BLE)

Cuando no se detecta fuego, el sistema entra en **modo manual**. El usuario puede controlar el movimiento del robot desde la app desarrollada en MIT App Inventor, enviando comandos que son interpretados por la función `controlBluetooth()`.

Esto permite que el usuario explore manualmente el entorno, reposicione el robot o controle directamente la acción extintora si lo desea.

Comunicación y monitoreo

El robot envía datos cada 1 segundo al dispositivo Android a través del módulo **HM-10**. Los datos enviados incluyen:

- Valor de los tres sensores de flama.
- Modo actual (`a` para automático, `m` para manual).

Esto se realiza en la función `enviarDatosBT()` y es visualizado en tiempo real en la aplicación, permitiendo al usuario conocer el estado del sistema en todo momento.

9. Pruebas y Resultados

9.1. Pruebas

A continuación se describen las pruebas realizadas para validar cada uno de los objetivos específicos del proyecto:

Objetivo específico	Prueba realizada	Descripción de la prueba	Criterio de éxito
Diseñar la estructura móvil del robot con configuración diferencial	Prueba de movilidad en superficie plana y con obstáculos	Se puso a prueba la maniobrabilidad del robot en distintas condiciones del terreno, comprobando la tracción de las ruedas y la dirección controlada.	El robot logra desplazarse hacia adelante, atrás y girar sobre su eje sin perder estabilidad.
Integrar los sensores de flama al sistema de control con Arduino UNO	Simulación de incendio en varios ángulos	Se encendieron fuentes de calor controladas frente a los sensores izquierdo, central y derecho. Se evaluó si el robot los reconoce correctamente.	El sistema detecta el fuego desde distintas posiciones y activa el modo automático de seguimiento.
Implementar el sistema de aspersión con bomba, relé y servomotor SG90	Activación del sistema de extinción	Una vez detectado el fuego, se verifica si se detiene el robot, se activa la bomba y el servomotor realiza el barrido de agua.	El sistema se activa al detectar flama cercana, pulveriza agua y regresa a posición inicial.
Desarrollar la app Android para teleoperación por Bluetooth	Control remoto desde la app en MIT App Inventor	Se envían comandos desde un celular Android para mover el robot y se monitorean los datos de sensores desde la app y el robot debe de alternar de modo (automatico y manual).	El robot obedece correctamente los comandos (A, S, I, D, T) y transmite datos al celular y cambia de modo cuando detecta fuego de manera correcta.

Tabla 7
Pruebas realizadas según los objetivos específicos del proyecto

9.2. Resultados

Los resultados obtenidos a partir de las pruebas realizadas evidencian el cumplimiento exitoso de cada uno de los objetivos específicos planteados en el proyecto. A continuación, se detallan los logros alcanzados en cada fase, junto con evidencia visual que respalda el funcionamiento del prototipo:

- **Movilidad del robot:**
  El robot demostró un **desplazamiento estable y controlado** en superficies planas y ejecutando giros suaves gracias a su **configuración diferencial**. Se comprobó el control total de sus movimientos (adelante, atrás, izquierda, derecha y detener) tanto de forma automática como mediante comandos enviados por la app móvil.

  EVIDENCIA EN EL ENLACE DEL VIDEO:**Video demostrativo del Carrito Bombero (YouTube)**

- **Detección de flama:**
  Los tres sensores de flama (izquierdo, central y derecho) fueron capaces de detectar la presencia de fuego en diferentes posiciones. Esto permitió activar automáticamente el modo de seguimiento, **orientando el robot hacia la fuente de fuego**, lo cual fue verificado mediante el monitor serial y observación directa. EVIDENCIA EN EL ENLACE DEL VIDEO: **Video demostrativo del Carrito Bombero (YouTube)**

- **Sistema de aspersión:**
  Al detectar una flama cercana, el robot se detuvo y activó correctamente el sistema de extinción:

  - Se encendió la bomba de agua conectada mediante un **módulo relé**.
  - El servomotor SG90 realizó un **barrido horizontal de 180°**, simulando la dispersión del agua para apagar el fuego.
  - Una vez completado el ciclo, el sistema volvió a su estado inicial.

  Este comportamiento fue constante en todas las pruebas. EVIDENCIA EN EL ENLACE DEL VIDEO:**Video demostrativo del Carrito Bombero (YouTube)**

- **Aplicación Android:**
  La aplicación desarrollada en **MIT App Inventor** permitió enviar comandos vía Bluetooth al robot mediante los botones de control (”A”, ”S”, ”I”, ”D”, ”T”), con una interfaz simple y funcional. Además, la app recibió y mostró los valores de los sensores en tiempo real, lo que facilitó el monitoreo del estado del entorno del robot.

Imagen 13
Prueba App n°1

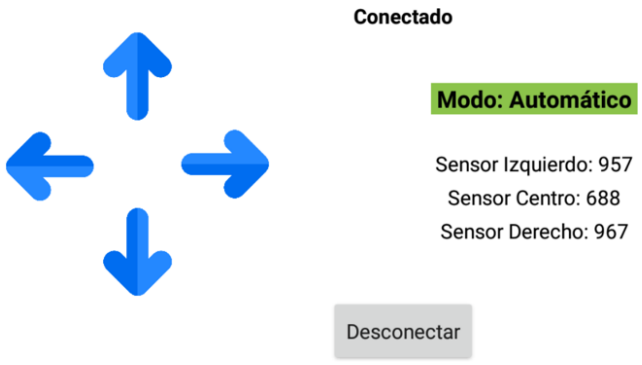
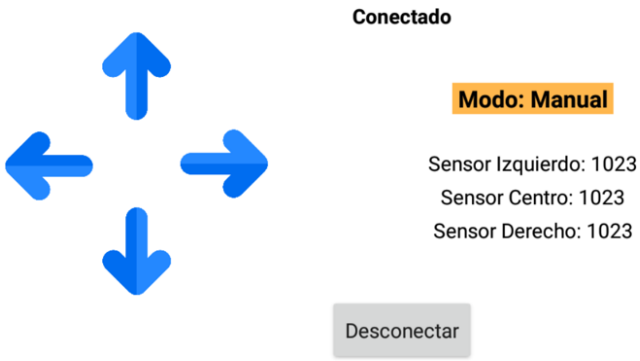


Imagen 14
Prueba App n°2



10. Conclusiones

1. El prototipo desarrollado logró integrar satisfactoriamente la teleoperación híbrida, combinando el modo automático de detección y extinción de fuego con el control manual mediante una aplicación Android, cumpliendo con el objetivo general del proyecto.
2. La estructura móvil con tracción diferencial proporcionó una buena estabilidad y maniobrabilidad al robot, permitiendo desplazarse con precisión hacia el foco de incendio en condiciones controladas.
3. La implementación de los sensores de flama junto con el sistema de aspersión (bomba, relé y servomotor SG90) demostró una respuesta eficaz al detectar y actuar sobre la presencia de fuego, cumpliendo con los objetivos funcionales del sistema.
4. La aplicación móvil desarrollada en MIT App Inventor permitió establecer una comunicación estable con el robot vía Bluetooth, facilitando el control remoto de los movimientos y validando su funcionalidad como herramienta de teleoperación.

11. Trabajos Futuros

A partir del desarrollo y evaluación del prototipo, se identifican posibles mejoras y líneas de trabajo que podrían implementarse en futuras versiones del proyecto:

1. **Integración de sensores de humo y temperatura:**
   Incorporar sensores adicionales permitiría mejorar la capacidad de detección en entornos con baja visibilidad o sin presencia directa de llama, aumentando así la precisión del sistema.
2. **Implementación de navegación autónoma mediante IA o algoritmos de mapeo:**
   Se podría desarrollar un sistema más avanzado que permita al robot explorar y localizar focos de

incendio sin intervención externa, utilizando técnicas como SLAM (Simultaneous Localization and Mapping).

3. **Comunicación inalámbrica de mayor alcance (Wi-Fi o GSM):**
   Sustituir el módulo Bluetooth por tecnologías de comunicación de mayor rango permitiría operar el robot desde ubicaciones remotas, aumentando su aplicabilidad en escenarios reales.

4. **Mejoras en la estructura y autonomía del sistema:**
   Rediseñar el chasis con materiales más resistentes, incorporar baterías de mayor capacidad y optimizar el consumo energético, mejoraría la durabilidad y el rendimiento general del prototipo.

12. Referencias bibliográficas

Uddin, M. M., Alam, M. M., Uddin, M. J., Ahmed, S., & Rahman, M. A. (2020). *Fire-Extinguishing Robot Design by Using Arduino*. International Journal of Scientific & Engineering Research (IJSER), *11*(7), 1366–1371. https://www.ijser.org/researchpaper/FIRE-EXTINGUISHING-ROBOT-DESIGN-BY-USING-ARDUINO.pdf

Raut, S., Kalokhe, P., Borse, S., & Borse, S. (2024). *Fire Fighter Robot Using IoT and Mobile Application*. International Journal of Research in Innovative Science and Engineering (IJRISE), *8*(2), 1–5.

Gong, J., Guo, W., & Sun, W. (2023). *UART communication protocol frame format explanation and application*. Recuperado de https://www.researchgate.net/publication/382389392_UART_communication_protocol_frame_format_explanation_and_application

RucksikaaR. (2021, 3 marzo). *Interfacing the HC-06 Bluetooth module with Arduino*. Project Hub, Arduino. Recuperado de https://projecthub.arduino.cc/RucksikaaR/interfacing-the-hc-06-bluetooth-module-with-arduino-94aabd

Rajguru Electronics. (s. f.). *HC-06 Core Bluetooth Module datasheet*. Recuperado de https://www.rajguruelectronics.com/Product/707/HC-06%20core%20bluetooth%20module.pdf

Aldhafeeri, B. H. A., Carrasco, J., Adorno, B. V., & Lopez, E. (2024). *A new hybrid teleoperation control scheme for holonomic mobile manipulator robots using a ground-based haptic device*. En *Towards Autonomous Robotic Systems (TAROS)* (Vol. 15051). https://doi.org/10.1007/978-3-031-72059-8_24

Patton, E. W., Tissenbaum, M., & Harunani, F. (2019). MIT App Inventor: Objectives, Design, and Development. En S. C. Kong & H. Abelson (Eds.), Computational Thinking Education (pp. 31–49). Springer. https://doi.org/10.1007/978-981-13-6528-7_3

Pappas, P., Chiou, M., Epsimos, G. T., Nikolaou, G., & Stolkin, R. (2020). *VFH+ based shared control for remotely operated mobile robots*. En L. Marques, M. Khonji y J. Dias (Eds.), *2020 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)* (pp. 366–373). IEEE. https://doi.org/10.1109/SSRR50563.2020.9292585

Huang, C., Zhang, C., & Zhao, Y. (2019). *Development of mobile robot with sensor fusion fire detection unit*. Procedia Manufacturing, 35, 1286–1291. https://doi.org/10.1016/j.ifacol.2018.11.324

Guo, Y., Guo, W., & Fan, Y. (2024). *High-performance fire detection framework based on feature fusion across multiple domains*. Intelligent Systems with Applications, 22, 200046.

https://doi.org/10.1016/j.jnlssr.2025.03.004

Xiang, F., Wang, Y., & Li, Z. (2022). *Design of intelligent fire fighting robot based on multi sensor fusion*. Journal of Intelligent & Robotic Systems, 105, 35. https://doi.org/10.1016/j.robot.2022.104122

Azeta, J., Ayoade, I., Nwakanma, C., & Akande, T. (2023). *Implementación de un prototipo de robot autónomo para la detección y extinción de incendios*. En Preprints. https://doi.org/10.20944/preprints202305.2010.v1

Deepthi, R., Reddy, S. D., Pouthri, R. A., Jyothi, P., & Jyothirmai, M. (2020). *Detector de incendios y extintor robot basado en Arduino*. Revista Internacional de Investigación Científica en Ciencia, Ingeniería y Tecnología, 195–198. https://doi.org/10.32628/ijsrset207354

Kiran, Prof. N. (2025). *Robot automático de extinción de incendios con Arduino*. Revista Internacional de Investigación Científica en Ingeniería y Gestión, 09(04), 1–9. https://ijsrem.com/download/automatic-fire-fighting-robot-using-arduino/?wpdmdl=49140&refresh=688947ed247211753827309

Kirubakaran, S., Rithanyaa, S. P., Thanavarsheni, S. P., & Vigneshkumar, E. (2021). *Arduino based firefighting Robot*. Journal of Physics: Conference Series, 1916(1), 012204. https://doi.org/10.1088/1742-6596/1916/1/012204

Mahfujul Islam, M. (2021). *Autonomous and wireless control fire fighter robot*. Automation Control and Intelligent Systems, 9(4), 97. https://doi.org/10.11648/j.acis.20210904.11

Nopriadi, N., & Fajrin, A. A. (2023). *Design of automatic fire detection and extinguishing devices using Arduino*. Sinkron, 8(1), 496–504. https://doi.org/10.33395/sinkron.v8i1.12047

Raju, J., Mohammed, S. S., Paul, J. V., John, G. A., & Nair, D. S. (2017). *Development and implementation of Arduino microcontroller based dual mode fire extinguishing robot*. 2017 IEEE International Conference on Intelligent Techniques in Control, Optimization and Signal Processing (INCOS). https://doi.org/10.1109/ITCOSP.2017.8303141

Resullar, L. J., Congreso, A. J. N., Comandante, F., Sarvida, F. P., & Buniel, G. G. (2020). *Design and fabrication of fire fighting autonomous robotic system equipped with sensitive sensors for fire alarm and detection, avoidance behaviour mechanism and SMS messaging capability*. Asian Journal of Basic Science & Research, 02(04), 21–51. https://doi.org/10.38177/ajbsr.2020.2404

Verma, A., Yadav, K., Malik, S., & Chaudhary, R. (2022). *Design and construction of automatic fire fighting robot with SMS notification*. En Advances in Transdisciplinary Engineering. IOS Press. https://ebooks.iospress.nl/doi/10.3233/ATDE220785

Jalani, J., Misman, D., Sadun, A. S., & Hong, L. C. (2019). *Robot automático de extinción de incendios con notificación*. Serie de Conferencias IOP: Ciencia e Ingeniería de Materiales, 637, 012002. https://iopscience.iop.org/article/10.1088/1757-899X/637/1/012002

Firmansyah, A., & Wisnuadji, T. W. (2024). *Robot automático de búsqueda y extinción de incendios uso de sensores basados en llama y ultrasonidos arduino uno*. SENAFTI - Seminario Nacional de Estudiantes de la Facultad de Tecnologías de la Información, 3(2), 1092–1099. https://senafti.budiluhur.ac.id/senafti/article/view/1517/778

Kiran, N., Sravani, T., Amrutha, B., Bhuvaneswari, K., Manohar, K., & Bharath Kumar, Y. (2025). *Robot automático contra incendios con Arduino*. https://ijsrem.com/download/automatic-fire-fighting-robot-using-arduino/?wpdmdl=49140&refresh=688947ed2472