



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



Robot Basurero Autónomo con Navegación a Punto de Acopio

Pongo Calderón, René

Quispe Llanque, Gabriel Omar

Argote Huancapaza, Juan Julio

Romucho Sullcahuaman, Paola Celeste

Curso: Robótica II

Docente: Yessica Rosas Cuevas

22/07/2025

ÍNDICE

- **Planteamiento del Problema y Solución**
- **Arquitectura del Sistema**
- **Diseño y metodología**
- **Resultados**
- **Demostración**
- **Conclusiones**

PROBLEMA Y JUSTIFICACIÓN

Problema

El modelo tradicional (**limpieza programada**) de gestión de residuos en espacios concurridos, como universidades, **es ineficiente** al depender de **rutas y horarios fijas**, provocando desbordes o vaciados innecesarios.

Esta rigidez genera **problemas de higiene y desperdicio de recursos**. Se requiere un sistema inteligente que se adapte a la demanda real mediante tecnologías como IoT.



Justificación

Este proyecto propone una solución innovadora en la gestión de residuos, basada en:

- **Optimización de Procesos:** Se implementa un modelo de "limpieza por demanda" que optimiza el tiempo del personal y elimina las recolecciones innecesarias.
- **Mitigación de Riesgos Sanitarios:** Se mejora la higiene al eliminar los desbordamientos y permitir un depósito de residuos sin contacto.
- **Innovación y Escalabilidad:** El proyecto sirve como una prueba de concepto para un nuevo modelo logístico en la gestión de residuos, aplicable a futuros edificios y ciudades inteligentes.

OBJETIVOS

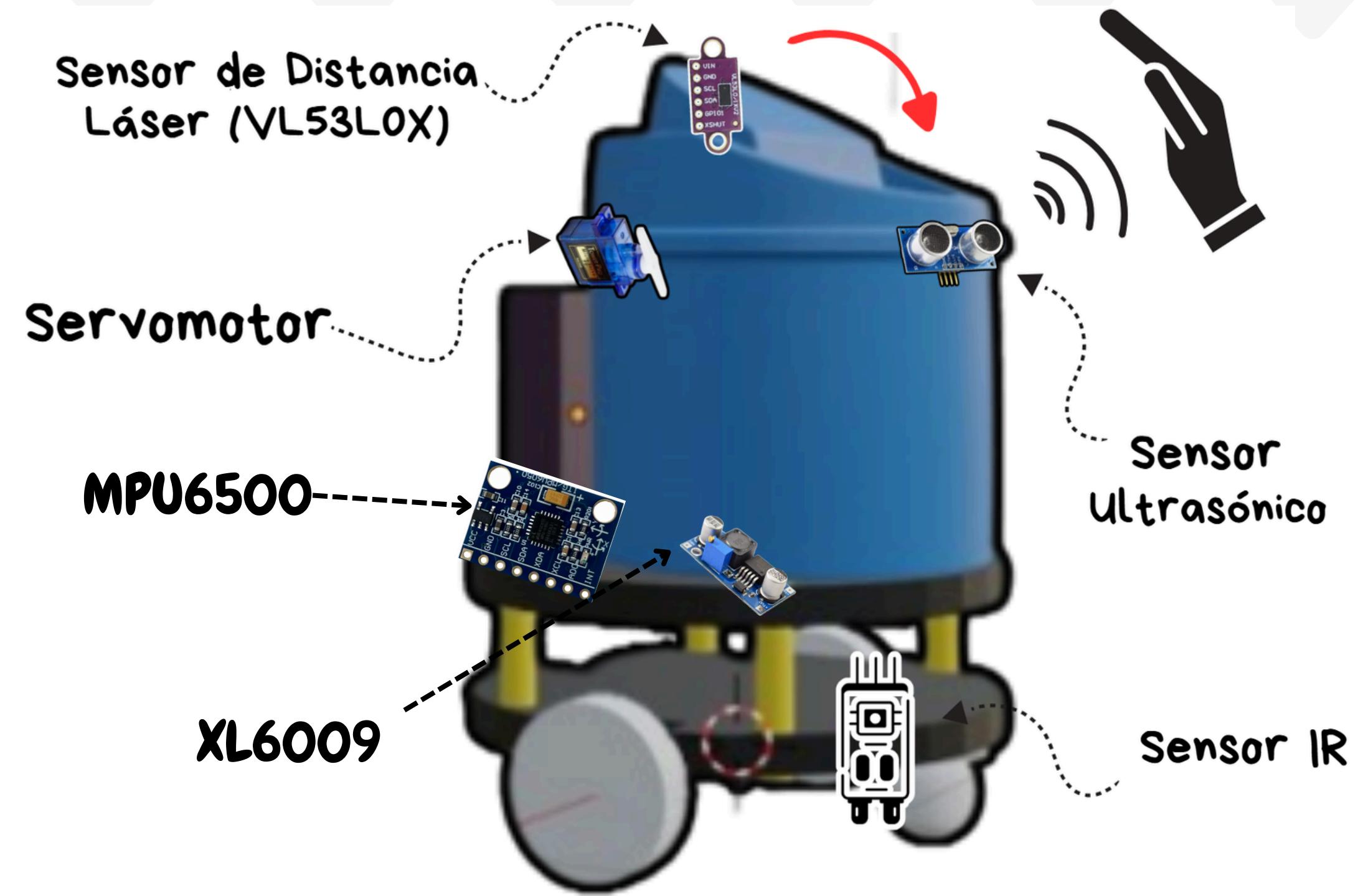
Objetivo General

Diseñar y construir un robot móvil automatizado que funcione como contenedor de residuos, capaz de enviar información de llenado a un dashboard web en tiempo real y desplazarse de forma guiada hacia un punto de acopio cuando esté lleno.

Objetivo Específico

- Automatizar la apertura y cierre de la tapa del contenedor mediante un sensor de proximidad y un servomotor.
- Monitorear el nivel de llenado del contenedor utilizando un sensor laser y mediante un dashboard web en tiempo real a través de Firebase.
- Implementar un controlador Proporcional-Integral-Derivativo para el robot basurero.
- Elaborar la documentación técnica del proyecto, incluyendo diagramas, código fuente comentado y el presente informe.

SOLUCIÓN PROPUESTA



- Para abordar el problema se ha diseñado el sistema inteligente “**Robot Basurero**”, un robot recolector autónomo de residuos que integra sensores de llenado, apertura automática de tapa, navegación basada en seguimiento de línea y monitoreo en tiempo real mediante una plataforma web conectada a Firebase.
- Este sistema invierte el modelo tradicional de recolección: *En lugar de que el personal vaya a los contenedores, el contenedor lleno se desplaza autónomamente hacia el personal (limpieza por demanda)*. Así, se reduce el contacto físico, se mejora la eficiencia operativa y se optimiza el uso del tiempo del personal de limpieza.

ALCANCE

Funcionalidades Incluidas (ALCANCE):

Para definir claramente los límites del prototipo, se especifica el siguiente alcance:

- Apertura y cierre automático de la tapa.
- Monitoreo del nivel de llenado y visualización en un dashboard online.
- Navegación autónoma desde su estación hasta un punto de acopio predefinido cuando está lleno.
- Espera de 20s en el punto de acopio para el vacío de residuos.
- Navegación autónoma de regreso a su estación una vez confirmada la descarga.



Funcionalidades Excluidas (LIMITACIONES):

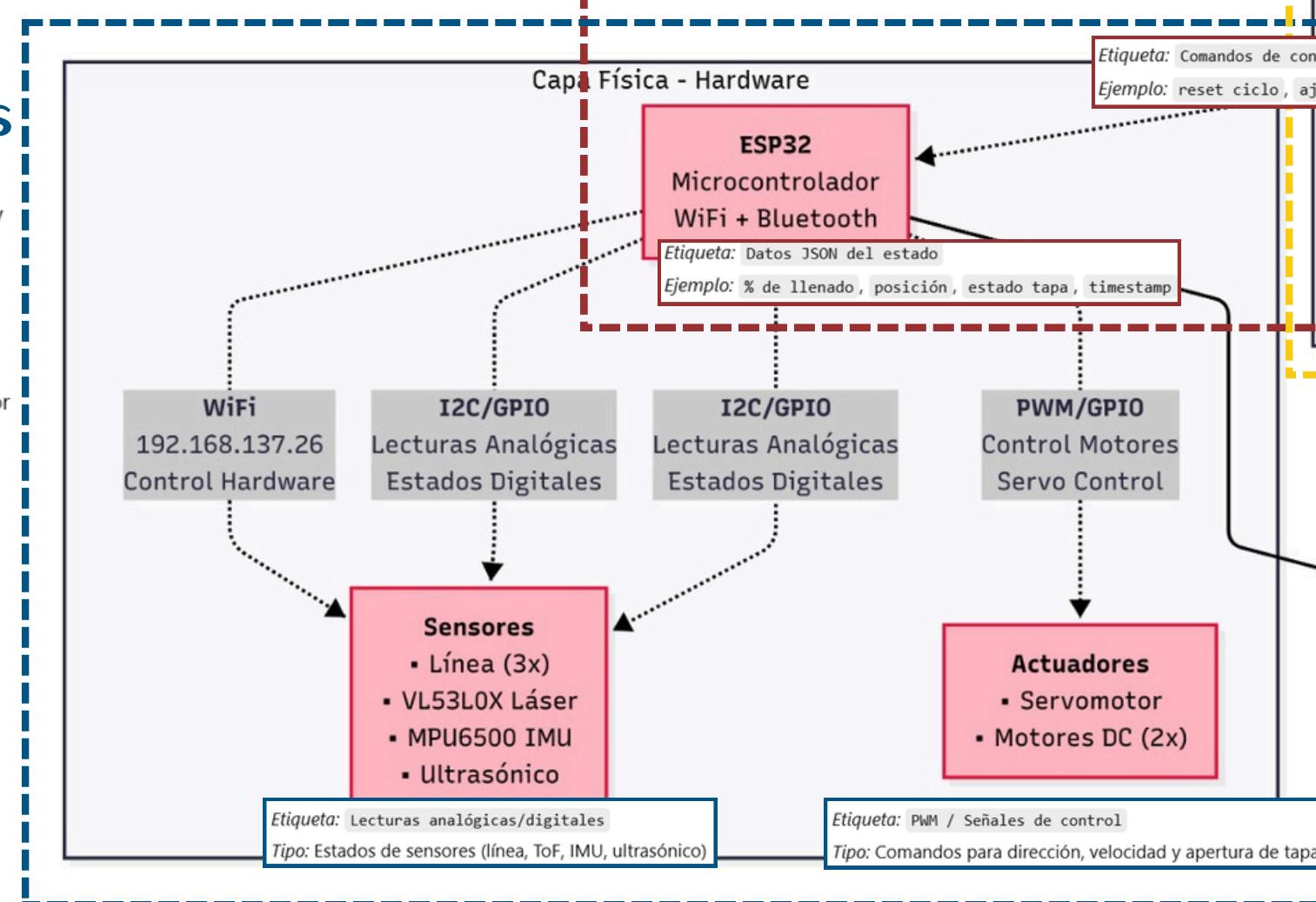


- El proyecto no incluirá un mecanismo de descarga o vaciado automático del contenedor. El proceso de vaciado será realizado manualmente por un operario una vez que el robot llegue al punto de acopio. La interacción termina cuando el operario descargue la basura, cada 20 sec se hará una comprobación si sigue detectando que está lleno el contenedor aumentará otros 20 sec de manera iterativa

ARQUITECTURA DEL SISTEMA

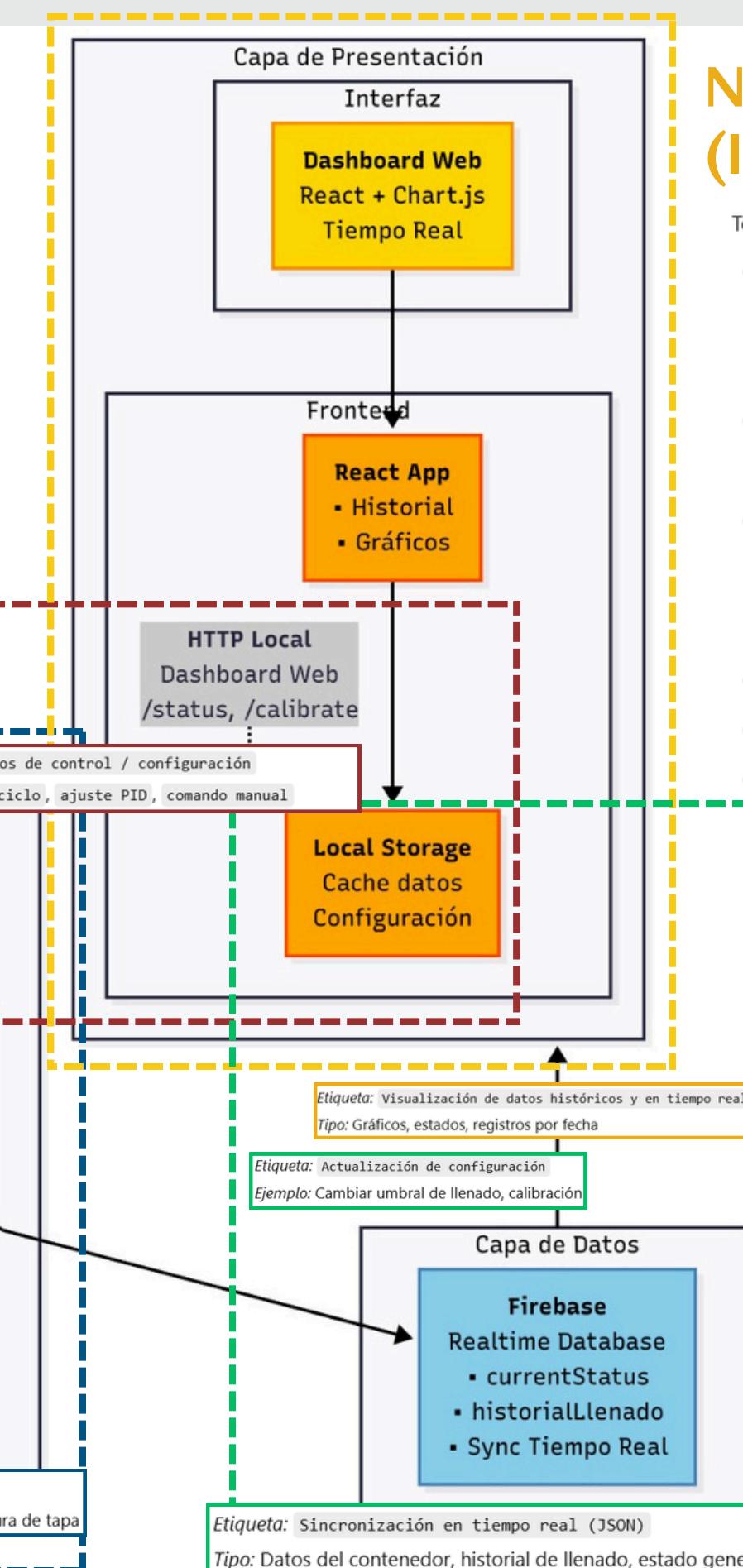
NIVEL 1. Dispositivos

- Centro del sistema: Microcontrolador **ESP32**, que recibe y procesa señales.
- Entradas:
 - Sensores de línea (3x) – Lectura digital.
 - VL53L0X (sensor láser) – Lectura analógica/digital por I2C.
 - MPU6050 (IMU) – Lectura de datos inerciales (aceleración y giro) por I2C.
 - Sensor ultrasónico – Lectura de distancia.
- Salidas:
 - Servo (tapa) – PWM.
 - Motores DC (2x) – PWM.
- Tipo de datos: Señales analógicas/digitales, pulsos PWM.



NIVEL 2. Comunicación

- WiFi (ESP32):
 - Transmite y recibe datos entre el microcontrolador y el sistema en la nube/local.
 - Comunicación mediante **HTTP** y **WebSocket**.
- Tipo de datos:
 - Mensajes en **JSON** para el estado del sistema, lectura de sensores, comandos y configuraciones.



NIVEL 4. Aplicación (Interacción)

Todo lo relacionado con el usuario y visualización:

- Dashboard Web (React + Chart.js)
 - Visualización en tiempo real
 - Gráficos históricos
- React App
 - Historial, configuración, gráficos
- Local Storage
 - Almacena configuraciones y caché

Tipo de datos que fluyen:

- JSON para renderizar datos
- txt/config en el Local Storage
- Interfaz gráfica con gráficos e indicadores

Parte encargada de procesar y almacenar información:

- Firebase Realtime Database
 - currentStatus
 - historialLlenado
 - sincronización en tiempo real

Tipo de datos que fluyen:

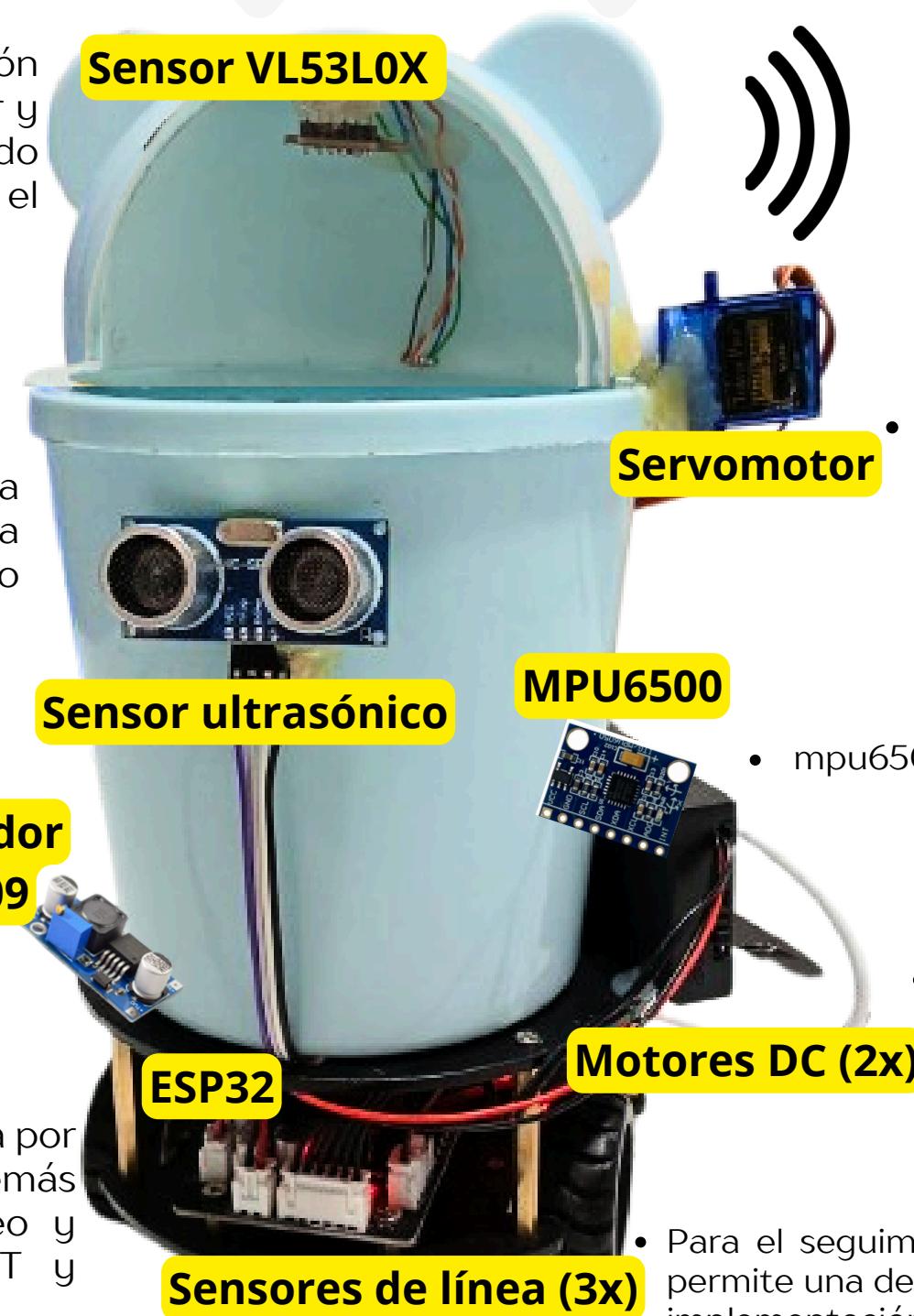
- JSON estructurado, sincronizado automáticamente
- Marca de tiempo, porcentaje de llenado, estados del robot

NIVEL 3. Ejecución de servicios

DISEÑO DE HARDWARE

Prototipo

- El sensor VL53L0X se seleccionó por su alta precisión en la medición de distancias cortas. Mide la distancia entre el tope del contenedor y los residuos acumulados, lo que permite determinar el nivel de llenado y activar automáticamente el desplazamiento del robot hacia el centro de acopio cuando se alcanza un umbral predefinido.



- El sensor ultrasónico fue incorporado para detectar la presencia del usuario frente al robot, activando así la apertura automática de la tapa. Esto asegura la interacción higiénica, sin contacto físico, y mejora la experiencia del usuario.

- asegurar un suministro de voltaje estable al sistema, permitiendo que los componentes electrónicos funcionen correctamente incluso cuando la fuente de energía presenta variaciones. Esto garantiza la operatividad continua del robot.

- Se utilizó un microcontrolador ESP32 como núcleo del sistema por su capacidad de integrar múltiples sensores y actuadores, además de su conectividad WiFi integrada que permite monitoreo y control remoto en tiempo real, ideal para sistemas IoT y automatización.



- Se utilizó un servomotor para automatizar la apertura y cierre de la tapa del contenedor, ya que permite un control preciso del ángulo de movimiento. En conjunto con el sensor ultrasonico, se garantiza una interacción fluida y sin contacto con el usuario, mejorando la higiene.

- mpu6500:Sensor MEMS de 6 ejes que combina un acelerómetro y un giroscopio de 3 ejes cada uno

- Para ejecutar el movimiento autónomo del robot, se utilizaron dos motores DC, que permiten una locomoción controlada y adaptada al entorno. Además, se integró un servomotor para accionar la apertura y cierre de la tapa del contenedor, brindando una solución mecánica eficiente, silenciosa y confiable.

- Para el seguimiento de la ruta, se emplearon tres sensores infrarrojos de línea, lo cual permite una detección precisa de desviaciones en el trayecto. Esta configuración facilita la implementación de un control PID, que garantiza un desplazamiento suave, estable y con alta precisión, evitando oscilaciones indeseadas durante la navegación punto a punto.

DISEÑO SOFTWARE Y CONTROL

Modelo Cinemático

$$v_L = V - \frac{L \cdot \omega}{2}$$

$$v_R = V + \frac{L \cdot \omega}{2}$$

Donde:

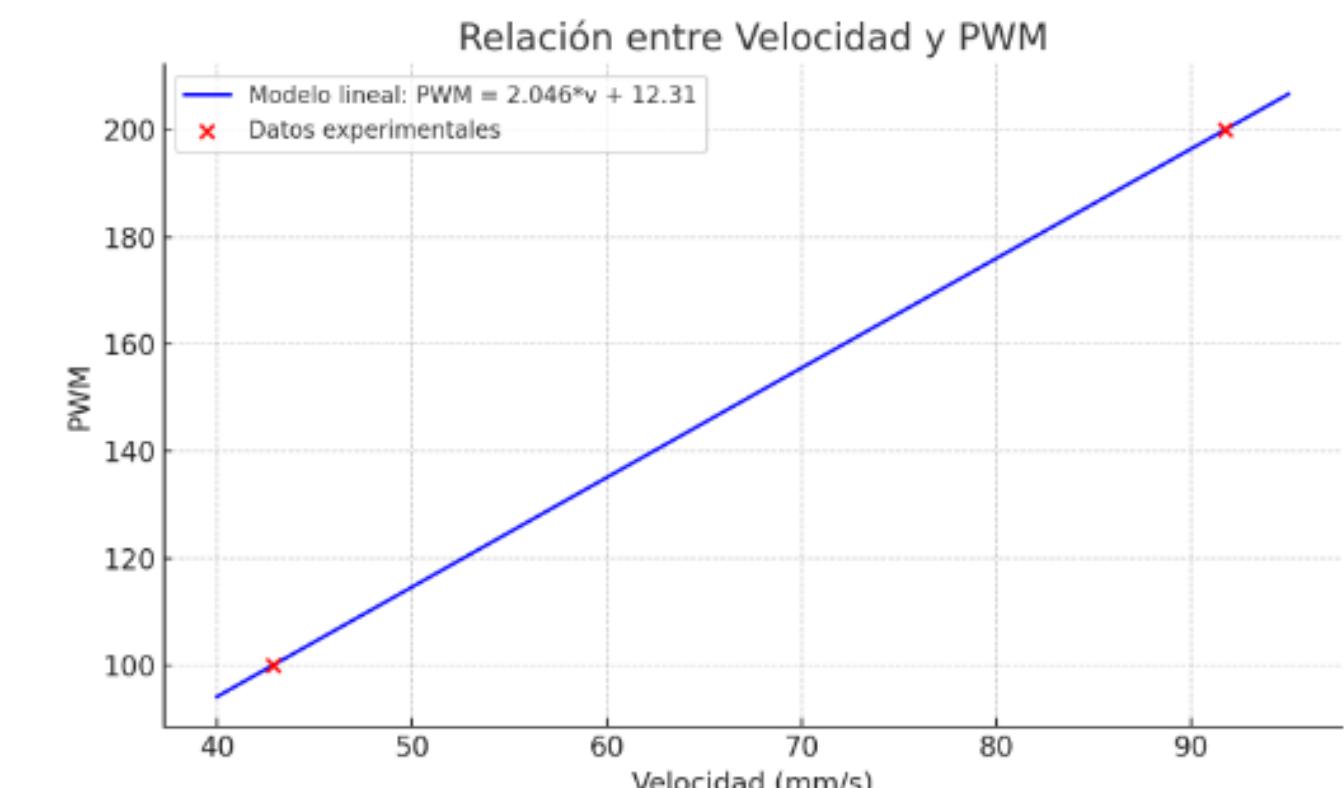
- V es la velocidad lineal base del robot, definida en el código como `BASE_LINEAR_VELOCITY`.
- ω es la velocidad angular correctiva, calculada por el controlador PID.
- L es la distancia entre las dos ruedas motrices (`DISTANCE_BETWEEN_WHEELS`).

```
float vL = BASE_LINEAR_VELOCITY - (DISTANCE_BETWEEN_WHEELS / 2.0) * omega;
float vR = BASE_LINEAR_VELOCITY + (DISTANCE_BETWEEN_WHEELS / 2.0) * omega;
```

PWM Aplicado	Tiempo (s)	Distancia (mm)	Velocidad (mm/s)
100	14.00	600	42.86
200	6.54	600	91.74

```
int pwmLeft = constrain((int)(vL * VEL_TO_PWM_SLOPE + VEL_TO_PWM_INTERCEPT), 0, MAX_SPEED);
int pwmRight = constrain((int)(vR * VEL_TO_PWM_SLOPE + VEL_TO_PWM_INTERCEPT), 0, MAX_SPEED);
```

$$\text{PWM}_{\text{calculado}} = (2,046 \cdot \text{Velocidad}_{\text{deseada}}) + 12,31$$



DISEÑO SOFTWARE Y CONTROL

Control PID

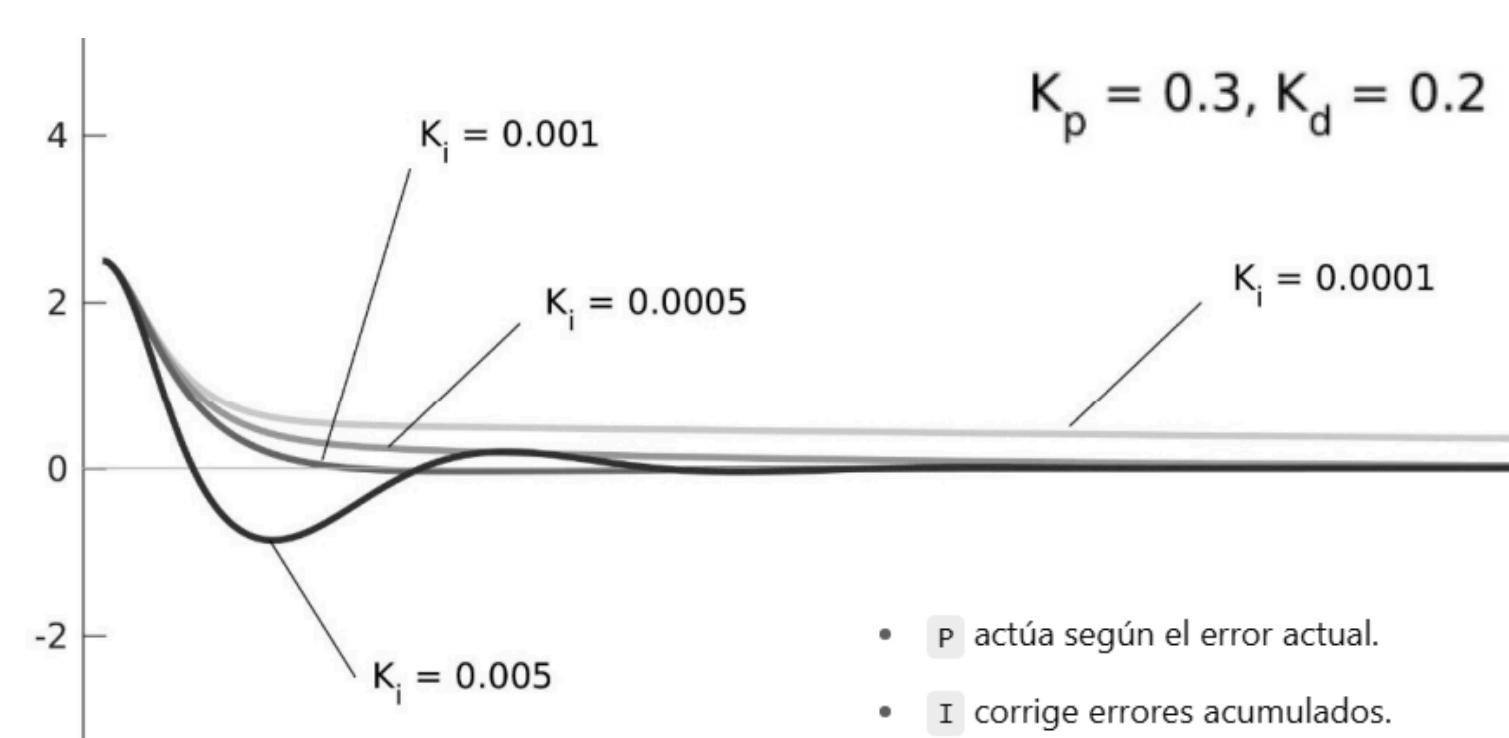
Se utilizó un controlador PID (Proporcional–Integral–Derivativo) para la navegación autónoma del robot, permitiendo el seguimiento de línea con corrección de trayectoria suave y precisa.

El PID se implementó en tiempo real para ajustar la velocidad de los motores y corregir desviaciones al seguir la línea negra.

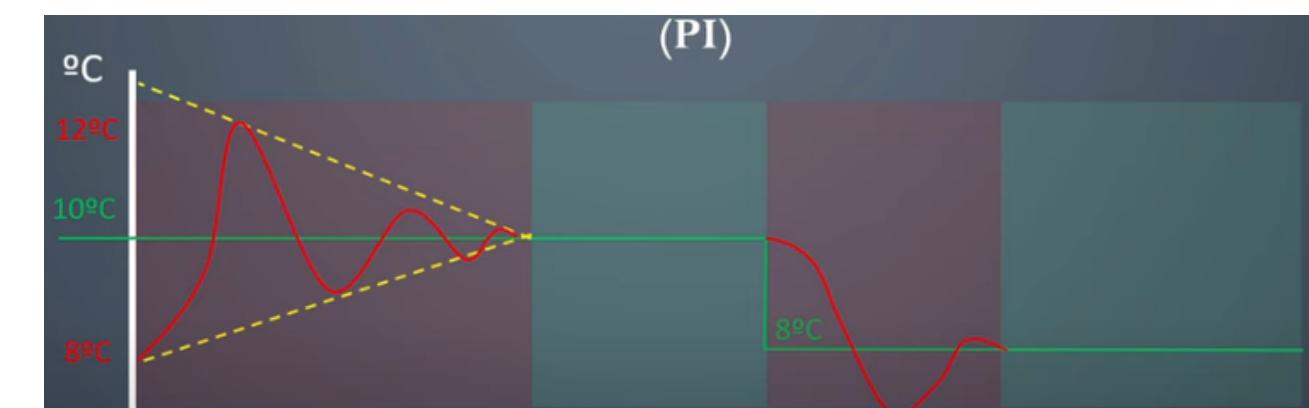
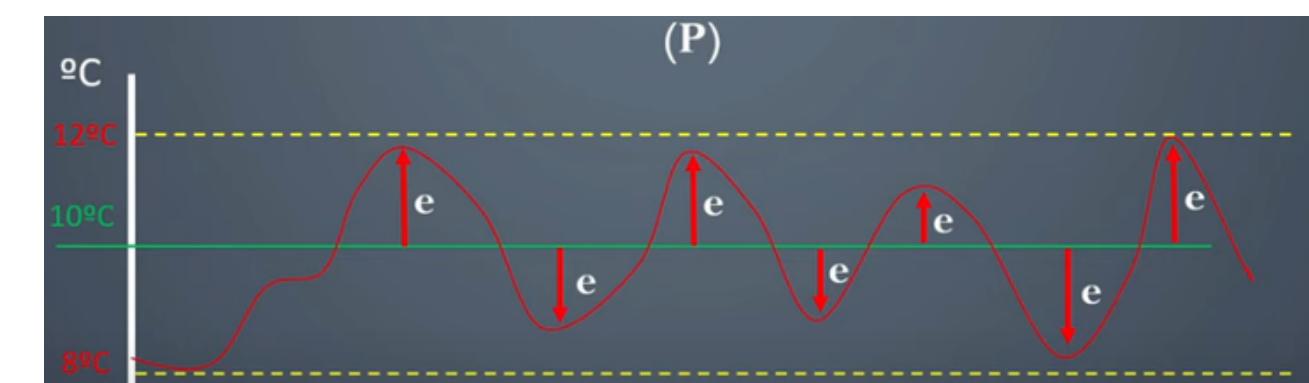
$$\omega = (K_p \cdot \text{error}) + (K_i \cdot \sum \text{error}) + (K_d \cdot (\text{error}_{\text{actual}} - \text{error}_{\text{anterior}}))$$

Lectura de sensores	Interpretación	Error
I: <input checked="" type="radio"/> C: <input type="radio"/> D: <input type="radio"/>	Línea muy a la izquierda	-2
I: <input checked="" type="radio"/> C: <input checked="" type="radio"/> D: <input type="radio"/>	Línea a la izquierda	-1
I: <input type="radio"/> C: <input checked="" type="radio"/> D: <input type="radio"/>	Línea al centro	0
I: <input type="radio"/> C: <input checked="" type="radio"/> D: <input checked="" type="radio"/>	Línea a la derecha	1
I: <input type="radio"/> C: <input type="radio"/> D: <input checked="" type="radio"/>	Línea muy a la derecha	2
Otro caso	Incertidumbre → mantiene	lastError

```
void seguirLineaPID() {  
    if (!enMovimiento) {  
        stopMotors();  
        return;  
    }  
  
    integral += error;  
  
    int derivative = error - lastError;  
  
    float omega = Kp * error + Ki * integral + Kd * derivative;  
  
    lastError = error;
```



- P actúa según el error actual.
- I corrige errores acumulados.
- D reacciona a cambios rápidos.



DISEÑO SOFTWARE Y CONTROL

Diagrama de secuencia

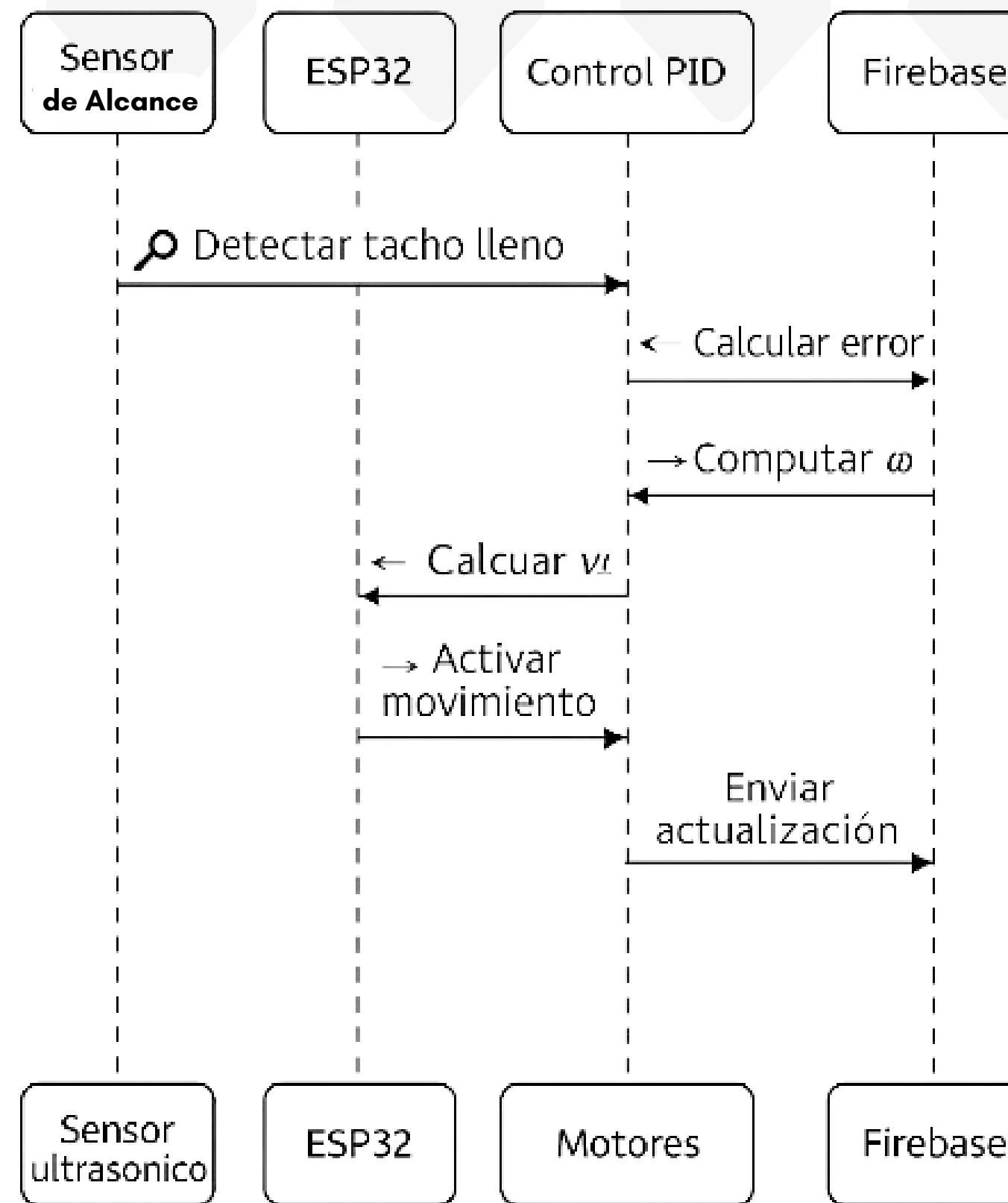
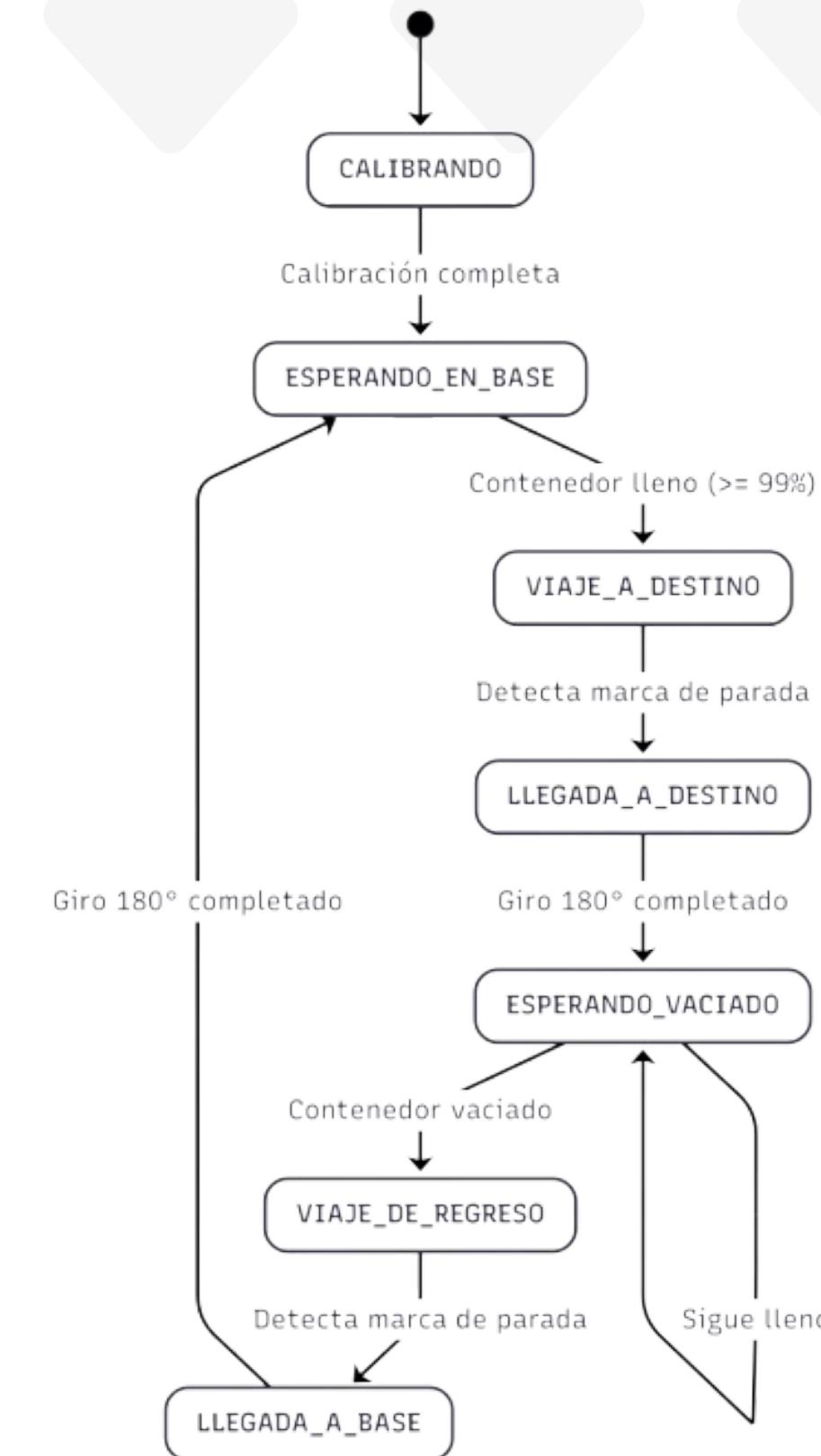


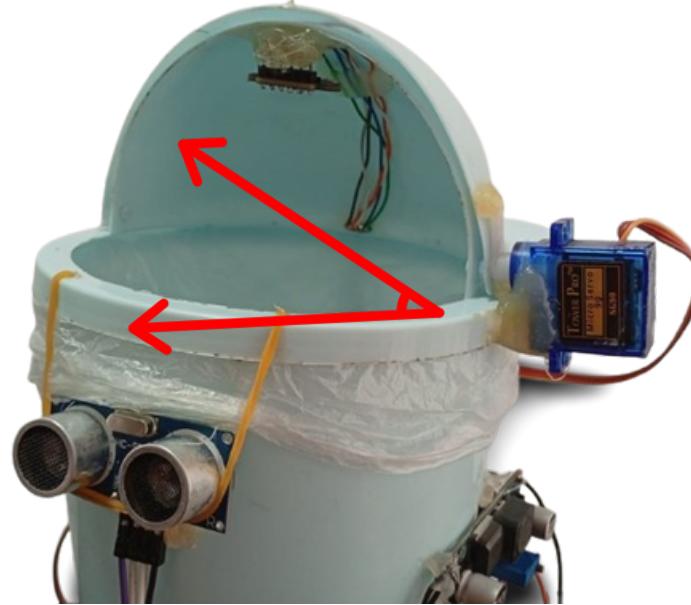
Diagrama de estados



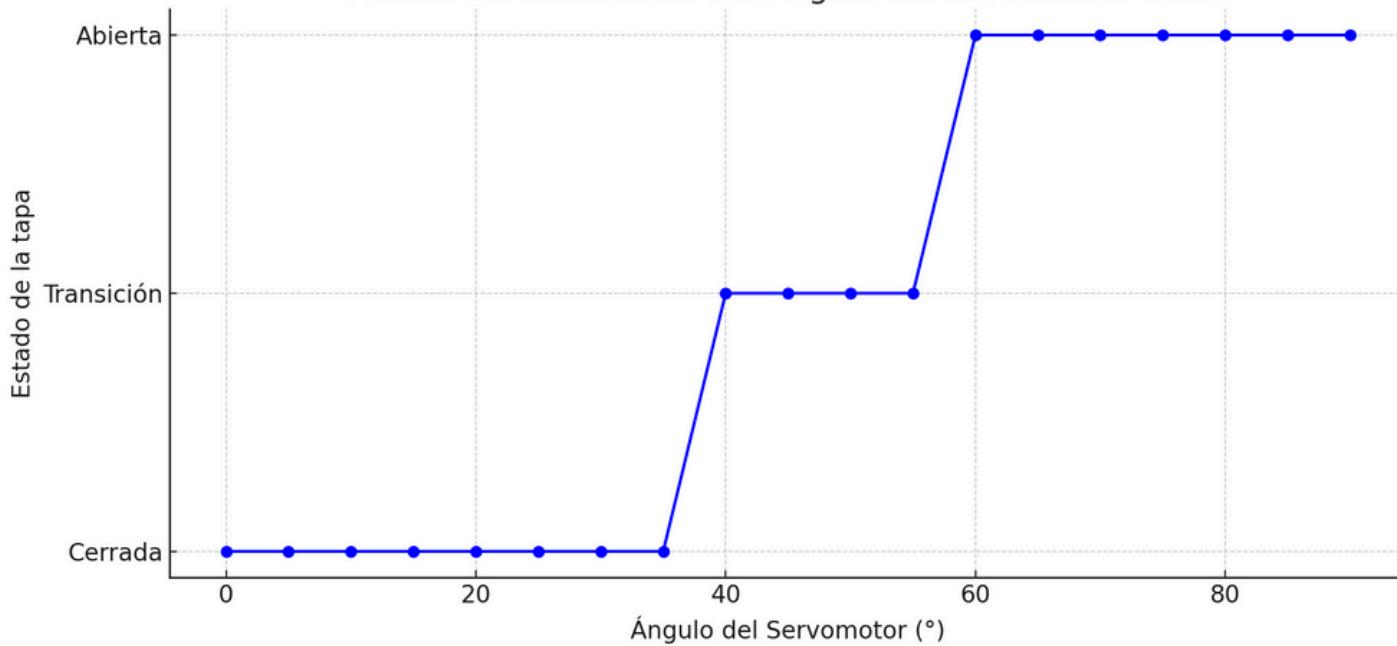
PRUEBAS

Validación del Sistema de Tapa Automática

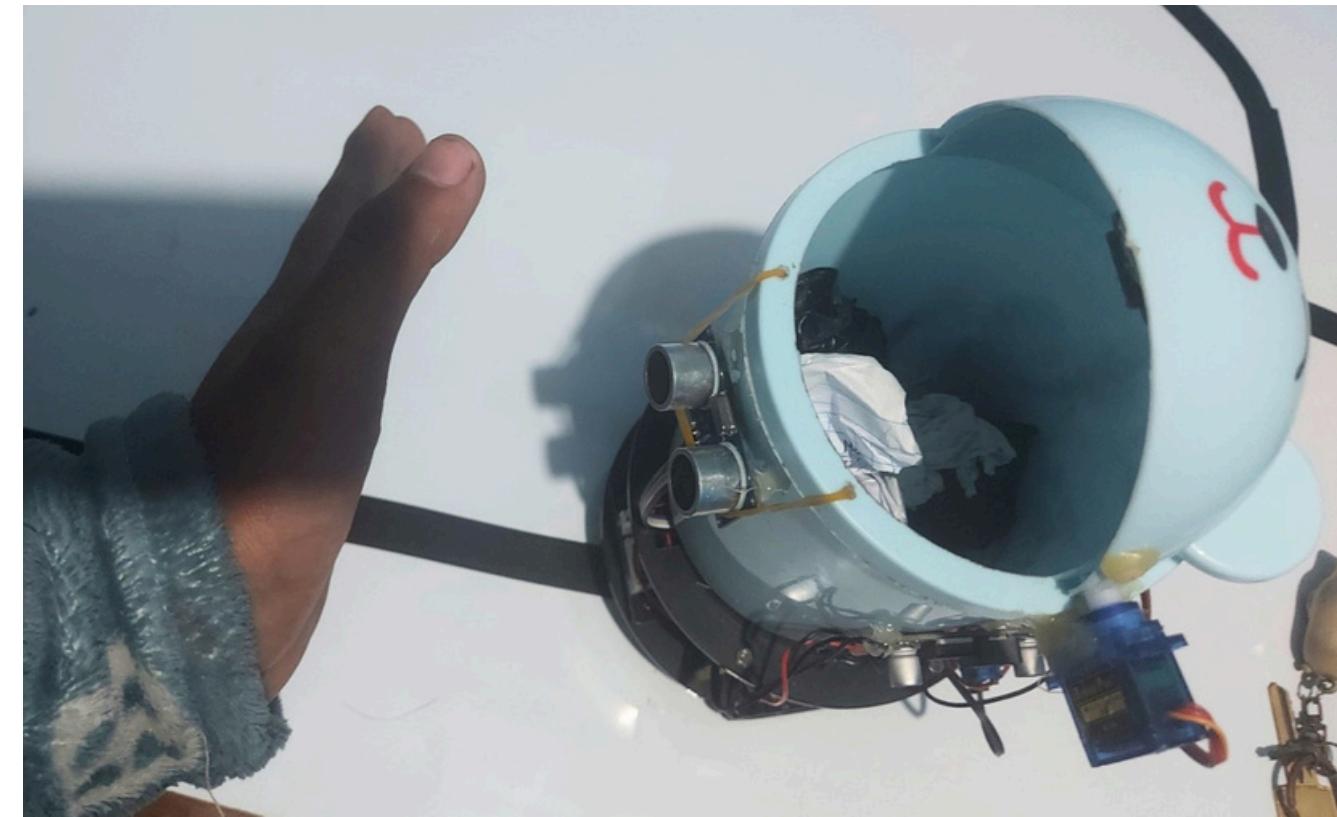
- Prueba de Calibración de Ángulos del Servomotor



Prueba de Calibración del Ángulo del Servomotor SG90



- Prueba de Fiabilidad del Sensor de Proximidad



30 ensayos
Umbral 35 cm

PRUEBAS

Validación del Monitoreo de Llenado

- Prueba de Precisión del Sensor ToF

Nivel de llenado	Distancia real (cm)	Promedio sensor (cm)	Error (%)
20%	20	21.2	6.0%
40%	16	15.0	6.3%
60%	12	12.7	5.8%
80%	8	8.6	7.5%
100%	4	4.3	7.5%



- Prueba de Latencia del Dashboard:



Campo Visual	Descripción	Origen	Valores Posibles
emojiAlerta	Emoji ⚠ que aparece al alcanzar 100% de llenado	fillLevel	⚠ o vacío
colorPorcentaje	Color dinámico del porcentaje según el nivel	fillLevel	Verde, Amarillo claro, Amarillo oscuro, Rojo
modoCarroTexto	Muestra texto del modo actual del carro	enMovimiento	🚚 Autonomo, 🚧 Estacionario
estadoTapaTexto	Representación visual del estado de la tapa o "N/A" si está en movimiento	lidStatus, enMovimiento	🟢 Abierta, 🔴 Cerrada, N/A
formatoFecha	Fecha convertida legible a partir de timestamp	timestamp	DD/MM/AAAA, HH:MM
persona	Ver si la persona esta cerca según el personDetected	personDetected	🚫 No Detectada, 🤷 Detectada

PRUEBAS

Validación de la Navegación Autónoma

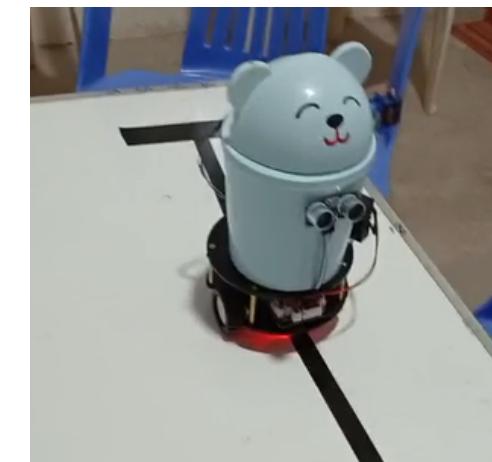
- Prueba de Calibración de Sensores de Línea



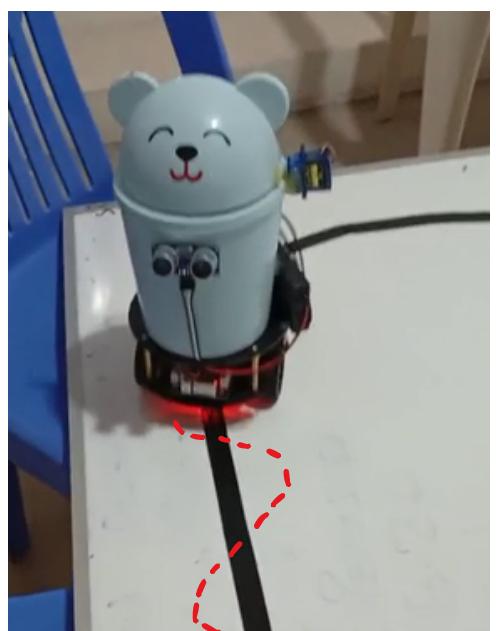
Lectura de sensores	Interpretación	Error
I:● C:○ D:○	Línea muy a la izquierda	-2
I:● C:● D:○	Línea a la izquierda	-1
I:○ C:● D:○	Línea al centro	0
I:○ C:● D:●	Línea a la derecha	1
I:○ C:○ D:●	Línea muy a la derecha	2
Otro caso	Incertidumbre → mantiene	lastError

- Prueba de Calibración Velocidad-PWM

Prueba	PWM Aplicado	Tiempo (s)	Distancia (mm)	Velocidad (mm/s)
1	100	14.00	600	42.86
2	200	6.54	600	91.74

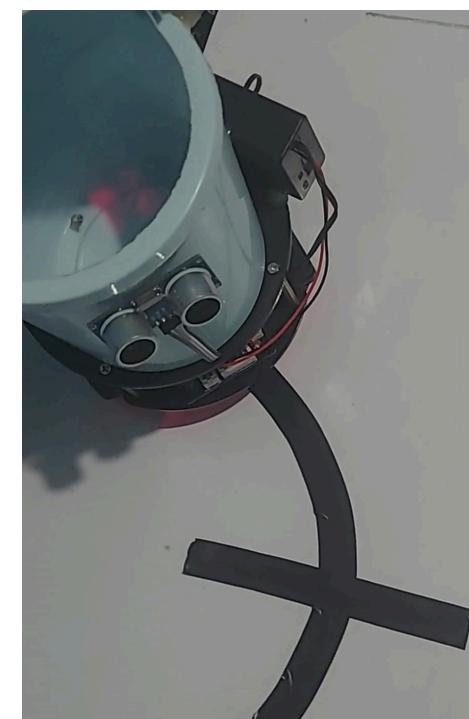


- Prueba de Sintonización del Controlador PID



Kp	Ki	Kd	Comportamiento Observado
1.0	0	0	Lento, se sale en curvas cerradas.
3.5	0	0	Reacciona rápido pero oscila violentamente (inestable).
2.0	0	4.0	Oscilaciones reducidas, pero aún sobrepasa la línea.
2.0	0.001	6.0	Óptimo: Seguimiento estable, suave y preciso.

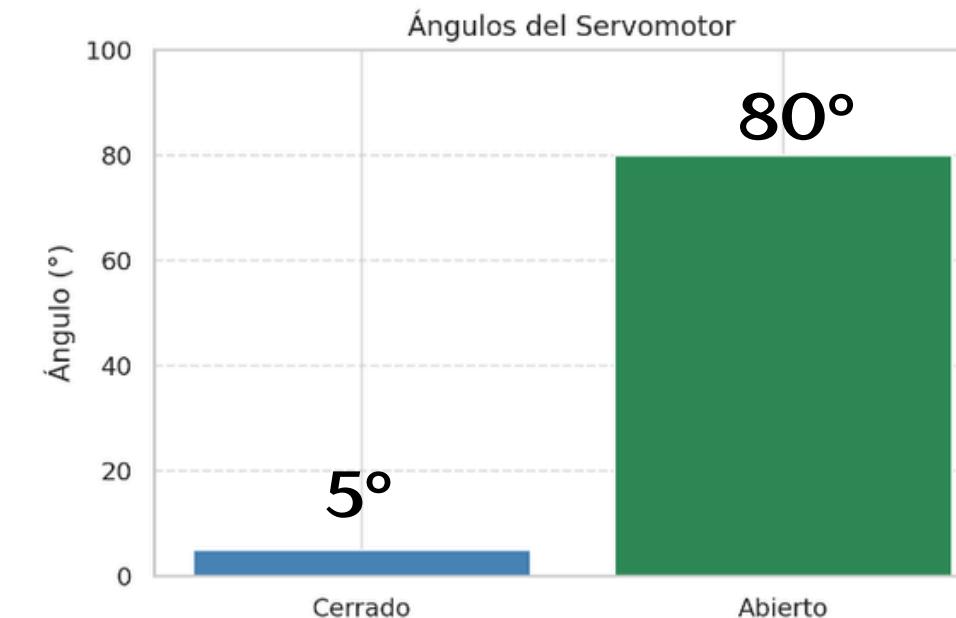
- Prueba de Precisión de Giro (IMU)



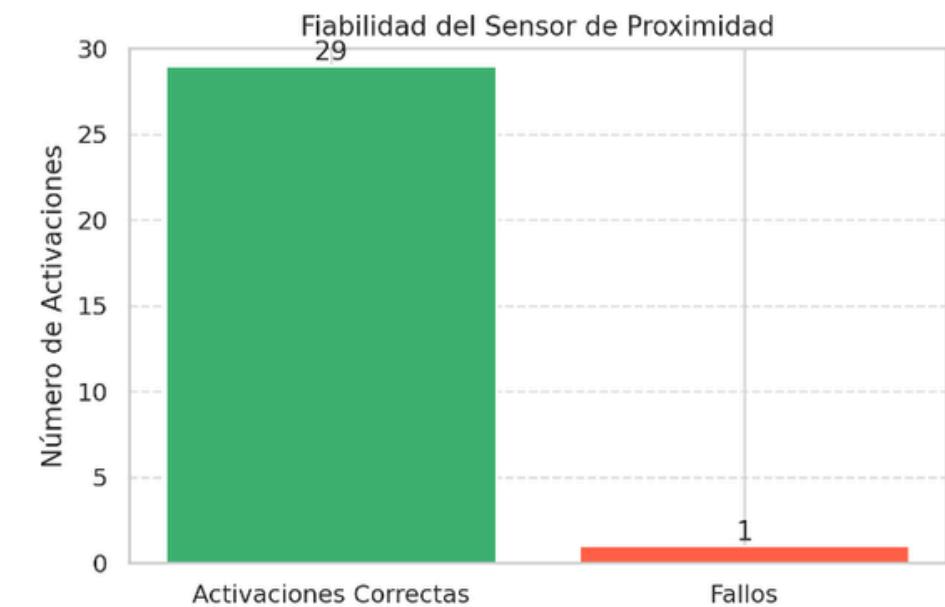
RESULTADOS

Calibración de Actuadores y Sensores:

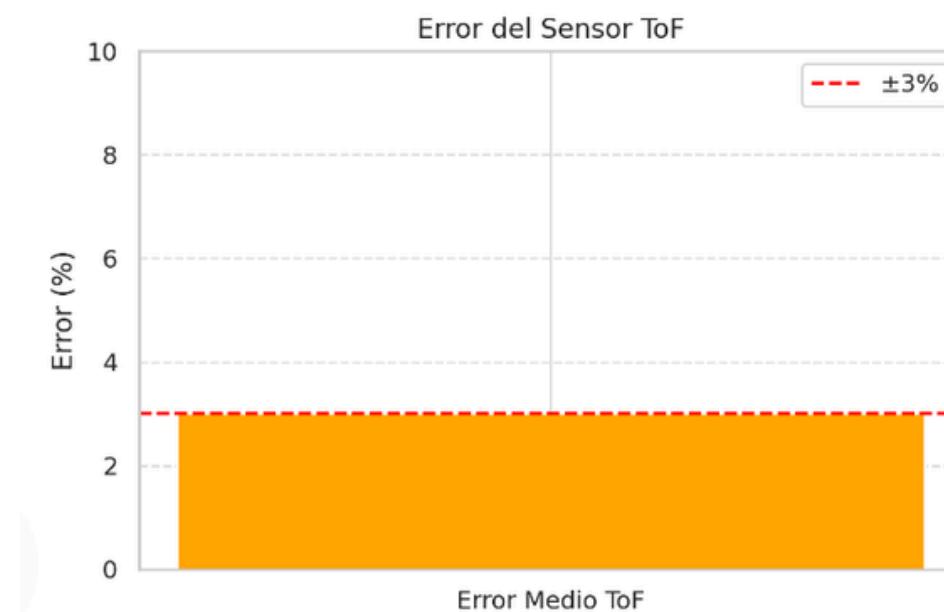
- Los ángulos óptimos para el servomotor se establecieron en 5° para cerrado y 80° para abierto, valores que se fijaron en el firmware como **SERVO_CERRADO** y **SERVO_ABIERTO**.



- La prueba de fiabilidad del sensor de proximidad arrojó una tasa de éxito del 96.7% (29 de 30 activaciones correctas). El único fallo ocurrió por una aproximación excesivamente rápida y oblicua.



- El sensor ToF para el nivel de llenado mostró un error medio de $\pm 3\%$, considerado altamente aceptable para esta aplicación. La latencia del dashboard fue consistentemente inferior a 3 o 4 segundos. Y la respuesta del sensor con el tacho es inmediata



RESULTADOS

Rendimiento del Control PID:

Las constantes finales que proporcionaron el mejor rendimiento (seguimiento suave, sin oscilaciones y rápida recuperación) fueron $K_p=2.0$, $K_i=0.001$, $K_d=6.0$

Kp	Ki	Kd	Comportamiento Observado
1.0	0	0	Lento, se sale en curvas cerradas.
3.5	0	0	Reacciona rápido pero oscila violentamente (inestable).
2.0	0	4.0	Oscilaciones reducidas, pero aún sobrepasa la línea.
2.0	0.001	6.0	Óptimo: Seguimiento estable, suave y preciso.

Con las constantes óptimas, se realizaron 25 pruebas de navegación completas en el circuito de 2 metros, obteniendo una tasa de éxito del 100%. La precisión del giro con la IMU MPU-6500 registró una desviación promedio de solo $\pm 4^\circ$ sobre el objetivo de 180° .



Iteración	Resultado	Observaciones
1-5	Fallo Parcial	El robot presentaba oscilaciones y se desviaba en curvas (fase de sintonización inicial).
6-30	Éxito	Seguimiento estable y preciso de la línea. El robot navegó fluidamente por toda la pista, incluyendo tramos rectos y curvos.
Tasa de éxito general (incluyendo sintonización): 83.3 %		
Tasa de éxito post-sintonización: 100 %		

DEMOSTRACIÓN



Para salir de la pantalla completa, pulsa **Esc**

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



Cerrar

Robot Basurero Autónomo con Navegación a Punto de Acopio

Pongo Calderón, René

Quispe Llanque, Gabriel Omar

Argote Huancapaza, Juan Julio

Romucho Sullcahuaman, Paola Celeste

Curso: Robótica II

Docente: Yessica Rosas Cuevas

22/07/2025

Usando:
Control PID
Cinematica
Dashboard Web

CONCLUSION

Conclusión

- Se logró construir un prototipo funcional que cumple con los objetivos planteados: seguimiento de línea, detección de llenado y retorno autónomo.
- Se validó satisfactoriamente el uso del control PID y del modelo cinemático diferencial, permitiendo una navegación precisa y eficiente.

TRABAJO FUTURO

1. Sistema de Gestión Energética Inteligente

- Actualmente, el prototipo depende de una batería fija. Usando sensores de voltaje y controladores inteligentes que regulen el uso de energía según la tarea del robot se podría optimizar el consumo energético y permitiría una mayor autonomía, sostenibilidad y menor necesidad de intervención humana.

2. Integración de Visión por Computador (IA)

- La navegación basada solo en sensores infrarrojos es limitada. Al incorporar visión artificial utilizando una cámara con procesamiento en el ESP32-CAM o con módulos de IA externos, el robot podría detectar obstáculos, clasificar residuos visualmente e incluso moverse libremente sin seguir una línea.

3. Clasificación Inteligente de Residuos

- Agregar capacidad de clasificación como la integración de sensores de peso, humedad, color o incluso brazos mecánicos simples, daría un salto en el impacto ecológico del proyecto, permitiendo separar residuos reciclables, orgánicos e inorgánicos desde el origen.



GRACIAS

Accede al
dashboard WEB



<https://robot-basurero-seguidor-linea-iot.vercel.app/>