

# **Diario de Ingeniería**

**Nombre del proyecto:** ONLY

**Categoría:** Futuros Ingenieros – WRO.

**Integrantes del equipo y roles:**

- Lopez Guadalupe- Electrónica
- Masset Celina- Programación
- Piris Francisco- Diseño

**Mentor/Docente:**

- Almeida Claudio Emanuel

## Índice:

Presentación del Equipo.....	
Introducción.....	
Ejemplos reales de autos autónomos.....	
Investigación.....	
Diseño y planificación.....	
Bocetos.....	
Diseño 3D.....	
Imágenes del robot.....	
Plan de pruebas.....	
Cronograma.....	
Imágenes del equipo trabajando.....	
Construcción.....	
Programación.....	
Pruebas y Resultados:.....	
Reflexión.....	
Conclusión.....	
Código final y completo.....	

## Presentación del Equipo:



Guadalupe  
Lopez

Francisco  
Piris

Celina  
Masset

Somos un equipo de jóvenes de **Cerro Corá, Candelaria – Misiones**, orgullosos integrantes de la **Red Maker de Cerro Corá**, que decidieron dar el gran paso de participar en la **WRO 2025 – Futuros Ingenieros**, movidos por nuestra pasión por la robótica, la creatividad y el trabajo en equipo.

.Nuestro grupo está conformado por **Guadalupe López**, responsable de electrónica; **Celina Masset**, programadora y, **Francisco Piris**, encargado del diseño. Además, también contamos con la ayuda y motivación de nuestros facilitadores **Claudio Almeida** y **Verónica Fonzalida**, quienes nos guían en cada etapa del proyecto.

Este año presentamos a nuestro robot **ONLY**, que representa la unión de nuestros talentos, el esfuerzo compartido y las horas de aprendizaje que hemos invertido. Para nosotros, participar en la WRO significa mucho más que competir: es la oportunidad de aprender, conocer otros proyectos, inspirarnos y compartir experiencias entre pares, de diferentes lugares, demostrando que nuestra mayor fortaleza es la **pasión y amor por la robótica**, motor que nos impulsa a seguir creciendo y a soñar con un futuro mejor.

## Introducción:

**Breve descripción del desafío:** desarrollar un robot autónomo capaz de recorrer circuitos y superar los distintos retos planteados en el campo de prueba.

**Objetivo del proyecto:** desarrollar un vehículo autónomo capaz de desplazarse de manera segura y eficiente dentro de un área determinada, cumpliendo con los desafíos propuestos por la categoría.

**Importancia de la solución:** para nosotros, este proyecto tiene un valor muy especial porque nos permite **aplicar los conocimientos adquiridos en nuestra sede del Espacio Maker de Cerro Corá**, vinculando las matemáticas, la ciencia y la tecnología con un desafío real. A través de este proceso logramos **diseñar, construir y programar un robot autónomo**, capaz de desplazarse por sí mismo, esquivar obstáculos y reconocer distintos elementos dentro del campo de prueba.

Más allá de la competencia, creemos que nuestro robot representa una herramienta con **gran potencial en la vida real**. Un sistema autónomo con estas características podría, por ejemplo, utilizarse en **operaciones de rescate**, accediendo a zonas de difícil o peligroso acceso para las personas. Asimismo, este tipo de desarrollo puede aportar a la **movilidad segura**, aplicándose en vehículos autónomos que necesitan detectar y evitar colisiones, tal como lo hacen actualmente los autos Tesla.

Este proyecto no solo nos motiva a competir, sino también a **imaginar aplicaciones concretas que mejoren la vida de las personas**, reforzando nuestra convicción de que la robótica y la innovación son claves para construir un futuro mejor.

**Planteo del Problema:** nuestro robot tiene que resolver varias dificultades. Una de ellas es **moverse de manera autónoma**. También debe **detectar obstáculos** y esquivarlos a tiempo, evitando, cambiar el sentido de dirección. Otra dificultad es **reconocer las señales de tráfico** (rojo y verde) y decidir cómo actuar según cada caso. También, debe intentar aparcar en la zona marcada. Además, el robot tiene que lograr **mantenerse dentro de un espacio limitado** (el cuadrado) sin salirse y hacerlo de forma rápida y precisa. Todo esto es un reto porque tenemos que

**programar bien los sensores y ajustar el diseño mecánico** para que funcione en diferentes pruebas.

### **Ejemplos reales de autos autónomos:**

**Tesla:** algunos modelos tienen "Autopilot", que permite **mantenerse en el carril, frenar y acelerar solos**, usando cámaras y sensores. Además, prevenir potenciales accidentes de tránsito.

Enlace para ver Info:<https://www.findmyelectric.com/tesla-autopilot-ultimate-guide/>

**Waymo (de Google):** autos completamente autónomos que **transportan personas en algunas ciudades**, usando radares, cámaras y sensores láser.

Enlace para ver Info:<https://waymo.com/intl/es/>

**Zoox (de Amazon):** vehículos autónomos diseñados para **entregar productos sin conductor**.

Enlace para ver Info:<https://www.infobae.com/america/tecno/2020/12/14/amazon-presento-zoox-un-vehiculo-sin-conductor-que-functionara-como-taxi-robot/>

Robots de delivery:

Enlace para ver :

[https://www.youtube.com/watch?v=Oj5b3q\\_iDog](https://www.youtube.com/watch?v=Oj5b3q_iDog)

### **Investigación:**

Para desarrollar nuestro robot ONLY realizamos una investigación previa sobre los distintos componentes necesarios y sus posibles aplicaciones. Estudiamos el **módulo controlador TB6612FNG**, encargado de manejar los motores de corriente continua; analizamos el uso del **servo motor**, que nos permitió controlar la dirección

con precisión; y exploramos diferentes tipos de sensores, siendo los **ultrasónicos** los más aptos para medir distancias y detectar obstáculos en tiempo real. También investigamos el **sensor de color**, fundamental para que el robot pueda reconocer señales de tráfico (rojo y verde) y tomar decisiones en base a ellas.

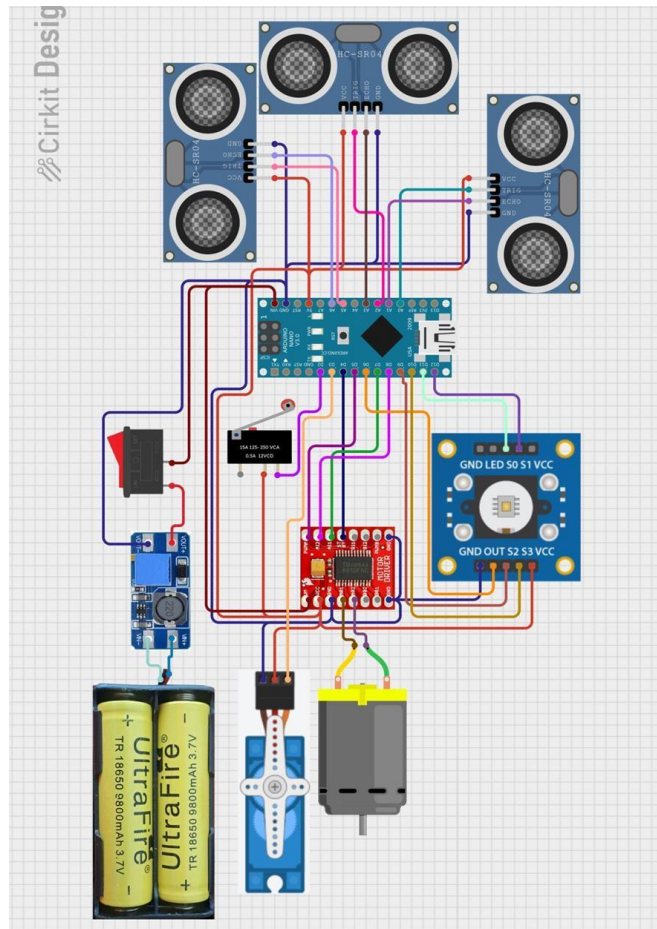
En cuanto a la **programación**, profundizamos en el uso de **Arduino** utilizando diversas **librerías** para controlar **el servomotor ,el motor DC y utilizar los sensores de distancia y sensor de color**, aprendiendo a integrar sensores y actuadores en un solo código que permitiera al robot moverse de manera autónoma, reaccionar ante estímulos y optimizar su tiempo de respuesta. Para la gestión del proyecto, utilizamos un **repositorio en GitHub**, lo que nos permitió almacenar el código, documentación, fotos y archivos .stl de forma segura, además de poder trabajar de manera colaborativa y llevar un registro de las distintas versiones del proyecto.

Asimismo, investigamos la **impresión 3D en PLA** para el diseño, ya que la misma brinda resistencia, lo cual es ideal en prototipos de robótica. También revisamos experiencias de robots autónomos (resuelve laberintos) en competencias anteriores y ejemplos de la industria, como los vehículos Tesla, para comprender cómo se aplican estas tecnologías en la vida real.

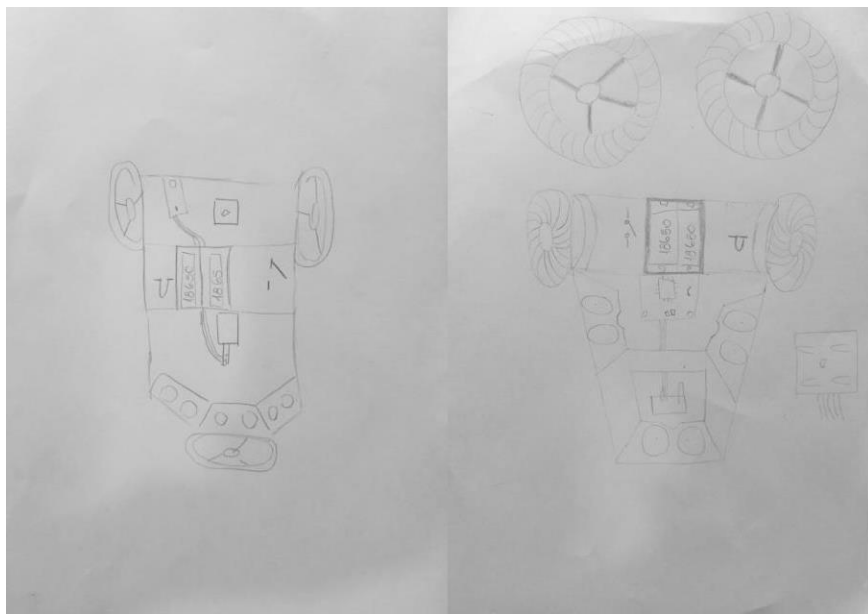
Todo este proceso de investigación nos permitió tomar decisiones fundamentadas en el diseño, integrar correctamente hardware y software, y fortalecer nuestros conocimientos en ciencia, tecnología e ingeniería.

## Diseño y Planificación

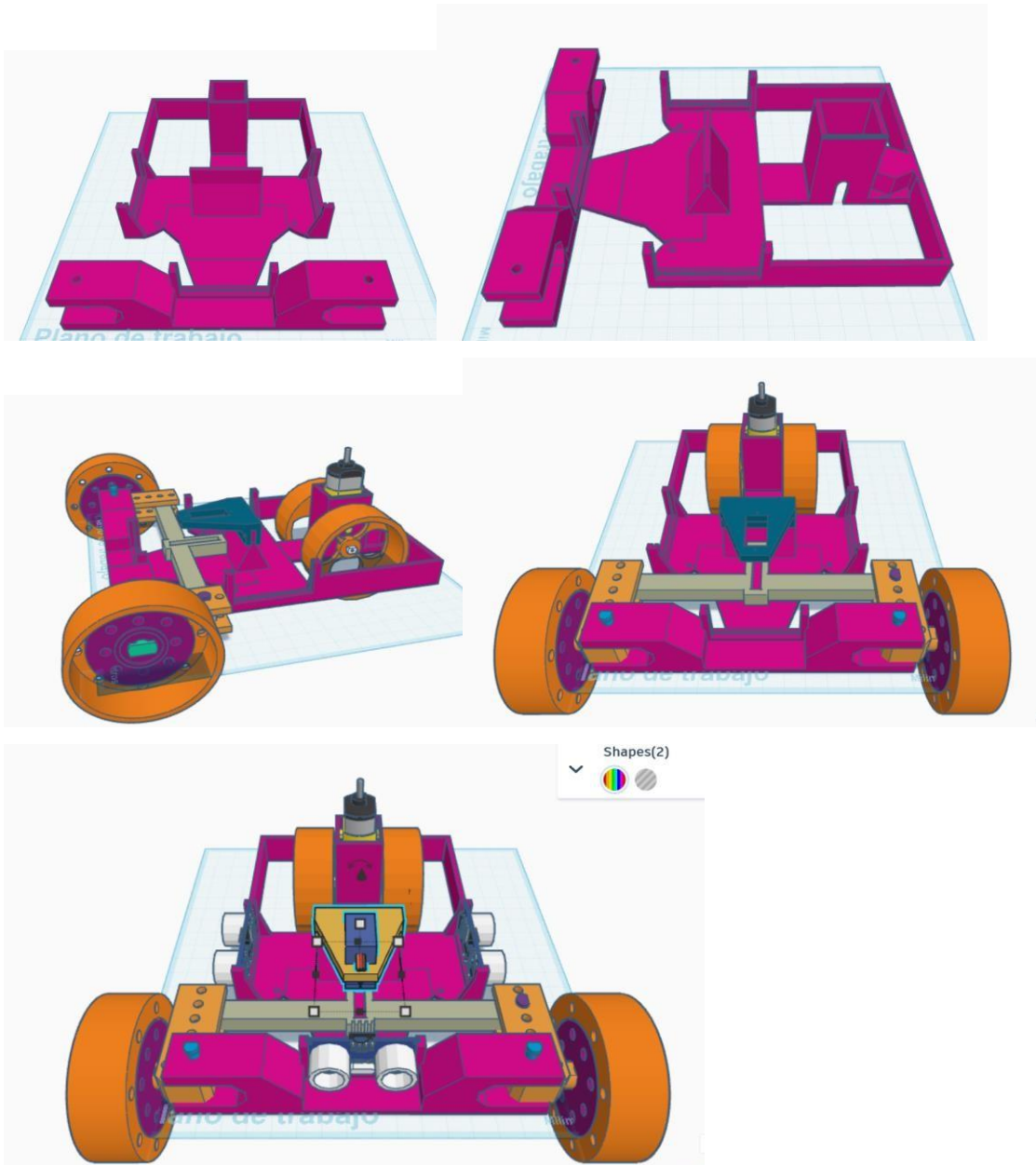
### Diseño del circuito electrónico:



### Bocetos:



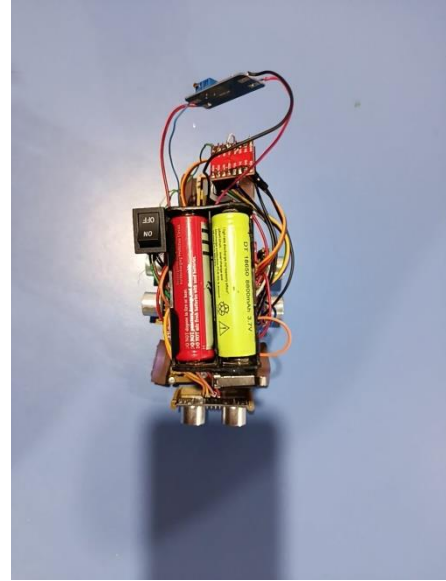
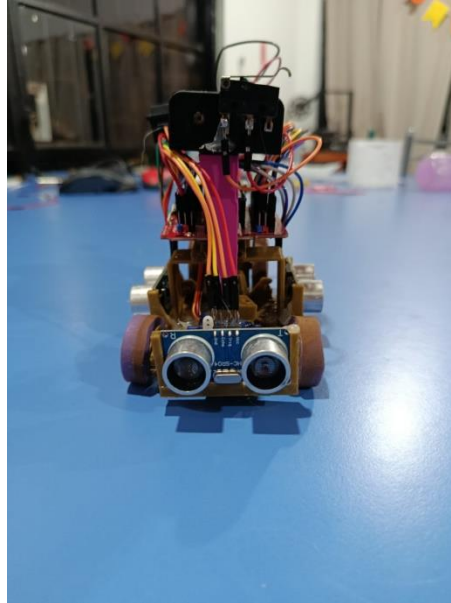
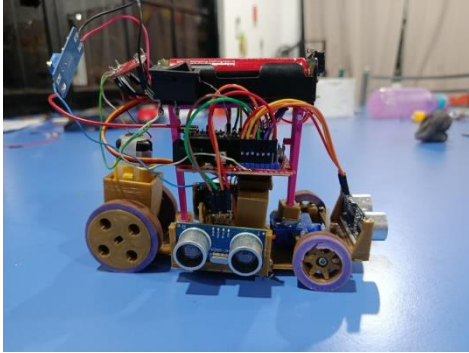
## Diseño 3D:










Red Maker-Cerro Corá

**Tomas final del robot:**



**Plan de pruebas (qué vamos a probar primero, cómo medir resultados):**

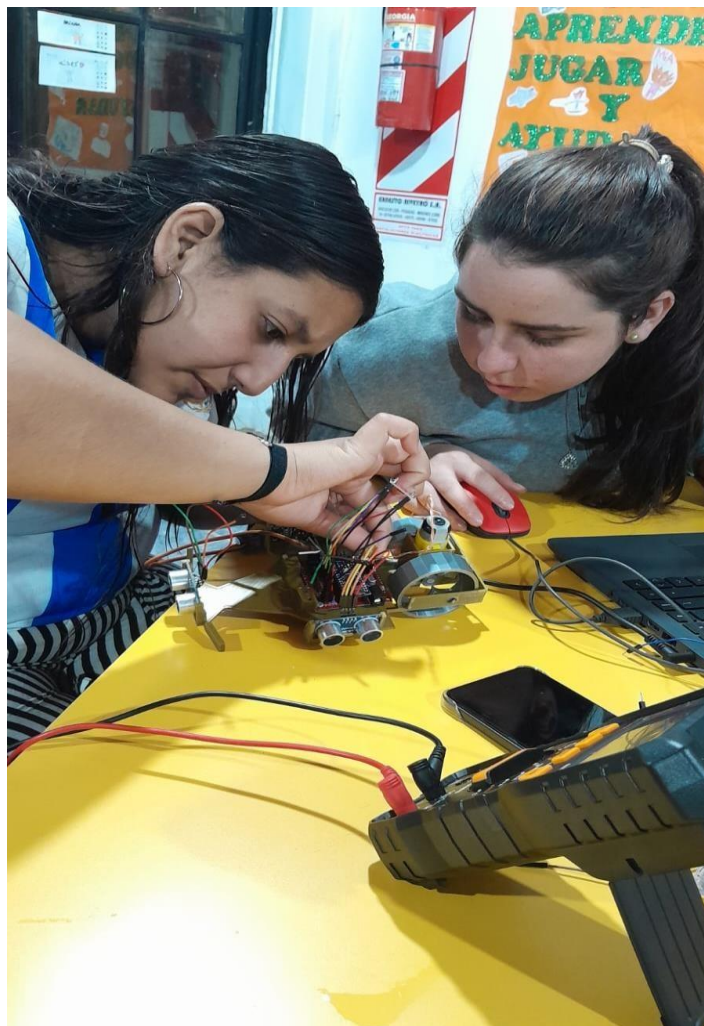
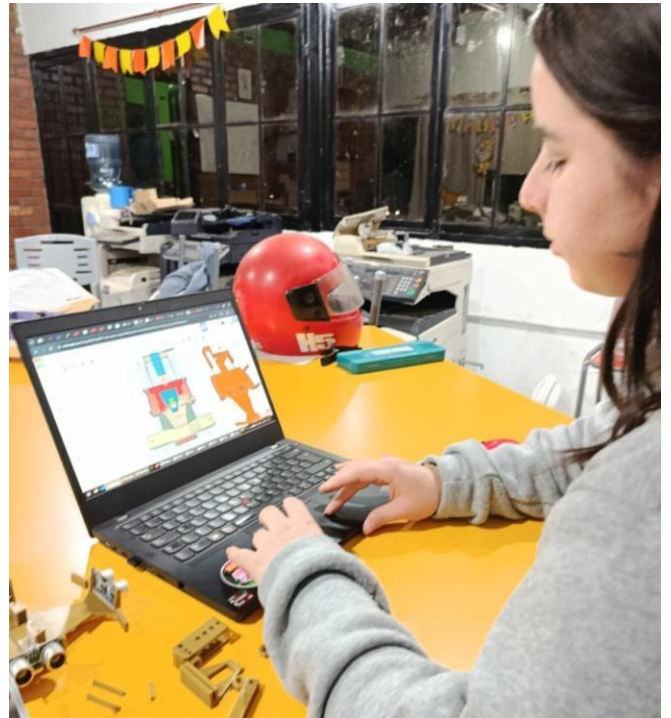
Etapa	Objetivo	Qué se prueba	Criterio de éxito	Resultado
Hardware	Verificar funcionamiento individual	Encendido, servo 0°–180°, sensores ultrasónicos, motor	Servo responde, sensores con error < $\pm 2$ cm, motor avanza y frena	
Movimiento básico	Validar avance y giros	Avance recto 1 m, giro izquierda (155°), giro derecha (30°)	Desvío < 10 cm en 1 m recto, giros con $\pm 15^\circ$ de error	
Obstáculos	Reacción autónoma ante objetos	Objeto al frente, izquierda y derecha	Tiempo reacción < 0,5 s, sin choques	
Señales tráfico	Interpretación de señales	Roja = frenar, Verde = avanzar	Se detiene < 10 cm antes de rojo, arranca en < 2 s con verde	
Circuito completo	Integrar todo en condiciones reales	Recorrido en tapete 3000x3000, mantenerse en área, aparcamiento	No salir del área, recorrido < 3 min, aparcar 3/5 intentos	

**Cronograma (semanas de trabajo y metas):**

Semana	Metas principales	Actividades realizadas
<b>04/08 – 10/08</b> Semana 1	Organización inicial	<ul style="list-style-type: none"> <li>• Formación del equipo y roles (Guadalupe: electrónica, Celina: programación, Francisco: diseño).</li> <li>• Elección del nombre del robot: <b>ONLY</b>.</li> <li>• Primeros encuentros en el Espacio Maker de Cerro Corá.</li> <li>• Lectura y análisis del reglamento.</li> </ul>
<b>11/08 – 17/08</b> Semana 2	Investigación y diseño	<ul style="list-style-type: none"> <li>• Estudio de sensores ultrasónicos, servo y TB6612FNG.</li> <li>• Revisión de materiales permitidos (LEGO, 3D, PLA).</li> <li>• Bocetos iniciales en papel y software.</li> <li>• Ensamblado básico del chasis.</li> </ul>
<b>18/08 – 24/08</b> Semana 3	Montaje y pruebas físicas	<ul style="list-style-type: none"> <li>• Instalación de sensores (izq., frente, der.).</li> <li>• Conexión de servo y motor.</li> <li>• Pruebas iniciales: avanzar, frenar y girar.</li> <li>• Recolección de datos para calibración.</li> </ul>
<b>25/08 – 31/08</b>	Programación principal	<ul style="list-style-type: none"> <li>• Desarrollo de la lógica</li> </ul>

Semana 4		<p>de movimiento:</p> <ul style="list-style-type: none"><li>– Avanzar recto.</li><li>– Frenar ante obstáculos.</li><li>– Giros (30°, 90°, 155°).</li><li>– Reconocimiento de señales (rojo = frenar, verde = avanzar).</li></ul> <ul style="list-style-type: none"><li>• Pruebas en campo reducido.</li><li>• Depuración de errores.</li></ul>
<b>01/09 – 05/09</b> Semana 5	Integración y documentación	<ul style="list-style-type: none"><li>• Integración de sensores + servo + motor + código.</li><li>• Optimización de velocidad y precisión.</li><li>• Ensayos en el tapete de prueba (3000 x 3000 mm).</li><li>• Correcciones con apoyo de Claudio Almeida y Verónica Fonzalida.</li><li>• Documentación final (equipo, objetivos, importancia).</li></ul>

**Imágenes del equipo trabajando:**



## **Construcción:**

### **Herramientas usadas:**

- Cautin de estaño
- Destornilladores
- Pinza y alicate
- Calibre
- Taladro y mechas
- Cinta métrica
- Pistola de silicona
- Multímetro
- Impresora 3d
- Trincheta

### **Materiales usados:**

- Gomas de caucho
- PLA
- Pegamento (La gotita)
- Silicona
- Estaño

### **Componentes usados:**

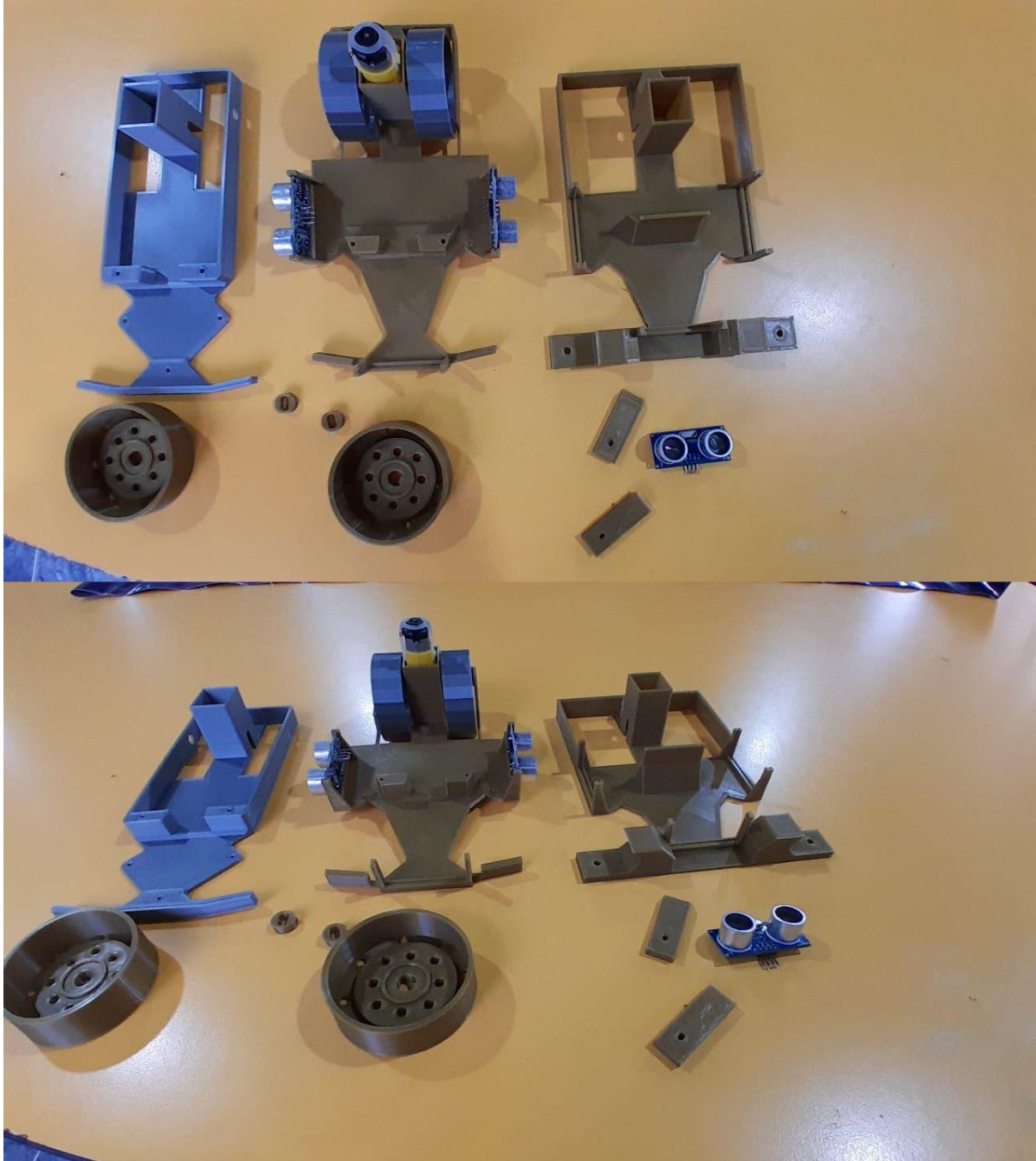
- Tres sensores de distancia
- Un sensor de color (TCS3200)
- Un final de carrera
- Un switch interruptor
- Un step UP(regulador-elevador de voltaje)
- Un Porta Baterías 18650
- Dos baterías 18650
- Un Driver TB5612 FNG
- Un motor DC
- Un servomotor
- Un arduino nano



Red Maker-Cerro Corá

- Un shield de expansión para arduino nano
- Cable USB para arduino nano

**Cambios realizados sobre el diseño inicial:**



## Programación:

```
#include <Servo.h>

// ===== TB6612FNG - Canal A =====

const int PIN_STBY = 4;

const int PIN_PWMA = 5;    // PWM motor

const int PIN_AIN1 = 7;

const int PIN_AIN2 = 8;

int velocidad = 135;  // 0-255

// ===== Servo dirección =====

Servo direccion;

const int PIN_SERVO = 3;    // servo conectado a D3

int posicionServo = 90;    // inicial

// ===== Sensores ultrasónicos =====

const int trigIzq = A0, echoIzq = A1;

const int trigFront = A2, echoFront = A3;

const int trigDer = A4, echoDer = A5;

long distanciaIzq, distanciaFront, distanciaDer;

const int UMBRAL = 20; // cm
```



```
// ===== Botón en pin 13 =====

const int PIN_BOTON = 13;

bool robotActivo = false;    // bandera: inicia apagado


// ===== Funciones Motor =====

void motorAvanzar(int vel) {

    digitalWrite(PIN_AIN1, HIGH);

    digitalWrite(PIN_AIN2, LOW);

    analogWrite(PIN_PWMA, constrain(vel, 0, 135));

}

void motorDetener() {

    analogWrite(PIN_PWMA, 0);

}

void motorStandby(bool activo) {

    digitalWrite(PIN_STBY, activo ? HIGH : LOW);

}


// ===== Medir distancia =====

long medirDistancia(int trig, int echo) {

    digitalWrite(trig, LOW);

    delayMicroseconds(2);

    digitalWrite(trig, HIGH);
```

```
    delayMicroseconds(10);

    digitalWrite(trig, LOW);

    long duracion = pulseIn(echo, HIGH, 20000); // timeout 20ms

    long distancia = duracion * 0.034 / 2; // en cm

    return distancia == 0 ? 999 : distancia; // si no lee, devuelve
    "muy lejos"
}

// ===== Setup =====

void setup() {

    pinMode(PIN_STBY, OUTPUT);

    pinMode(PIN_PWMA, OUTPUT);

    pinMode(PIN_AIN1, OUTPUT);

    pinMode(PIN_AIN2, OUTPUT);

    pinMode(trigIzq, OUTPUT);

    pinMode(echoIzq, INPUT);

    pinMode(trigFront, OUTPUT);

    pinMode(echoFront, INPUT);

    pinMode(trigDer, OUTPUT);

    pinMode(echoDer, INPUT);

    pinMode(PIN_BOTON, INPUT); // Pulsador a 5V con resistencia pull-
down
```

```
    direccion.attach(PIN_SERVO);

    direccion.write(posicionServo); // posición inicial recto

    motorStandby(true);

    motorDetener();

    Serial.begin(9600);

}

// ===== Función principal del robot =====

void ejecutarRobot() {

    // Leer distancias

    distanciaFront = medirDistancia(trigFront, echoFront);

    distanciaIzq    = medirDistancia(trigIzq, echoIzq);

    distanciaDer    = medirDistancia(trigDer, echoDer);

    // Mostrar en monitor serial

    Serial.print("Izq: "); Serial.print(distanciaIzq);

    Serial.print(" cm | Front: "); Serial.print(distanciaFront);

    Serial.print(" cm | Der: "); Serial.print(distanciaDer);

    Serial.print(" cm | Servo: "); Serial.println(posicionServo);

    // Lógica de decisión

    if (distanciaFront > UMBRAL) {
```

```
// Avanzar recto

posicionServo = 90;

direccion.write(posicionServo);

motorAvanzar(velocidad);

}

else if (distanciaIzq > UMBRAL) {

    // Girar izquierda

    posicionServo = 155;

    direccion.write(posicionServo);

    motorAvanzar(velocidad);

    delay(500);

    distanciaFront = medirDistancia(trigFront, echoFront);

    if (distanciaFront > UMBRAL) {

        posicionServo = 90;

        direccion.write(posicionServo);

        motorAvanzar(velocidad);

    }

}

else if (distanciaDer > UMBRAL) {

    // Girar derecha

    posicionServo = 30;

    direccion.write(posicionServo);

    motorAvanzar(velocidad);

    delay(500);
```

```
    distanciaFront = medirDistancia(trigFront, echoFront);

    if (distanciaFront > UMBRAL) {

        posicionServo = 90;

        direccion.write(posicionServo);

        motorAvanzar(velocidad);

    }

}

else {

    // Todo bloqueado → detenerse

    motorDetener();

}

delay(100); // pequeña pausa para estabilidad

}

// ===== Loop =====

void loop() {

    if (!robotActivo && digitalRead(PIN_BOTON) == HIGH) {

        robotActivo = true; // primera vez que se aprieta, se activa

        Serial.println("Robot iniciado!");

        delay(300); // pequeña pausa para evitar rebote del botón

    }

    if (robotActivo) {
```

```
    ejecutarRobot();  
  
    } else {  
  
        motorDetener();  
  
    }  
  
}
```

### Cómo funciona el programa del robot ONLY:

Cuando encendemos el robot, **no hace nada** hasta que se aprieta el botón de inicio (final de carrera) el **botón en el pin 13**.

En ese momento, queda marcado como *“iniciado”* y empieza a funcionar.

#### 1. Avanzando normal

- El robot va leyendo todo el tiempo la distancia con el sensor frontal y los dos laterales.
- Si adelante no hay nada cerca (más lejos que 20 cm), el servo se pone en **90° (recto)** y el motor avanza derecho.

#### 2. Cuando se encuentra con un obstáculo de frente

- Antes de mirar a la izquierda o a la derecha, le pregunta al **sensor de color**:

- Si el sensor de color ve **rojo** → el servo gira a **155°** (como doblar a la derecha).
- Si el sensor ve **verde** → el servo gira a **30°** (como doblar a la izquierda).
- En ambos casos, avanza un poquito, después vuelve a servo **90°** (recto) y sigue derecho.
- Si el sensor de color **no reconoce nada** (ni rojo ni verde), recién ahí hace lo de siempre:
  - Pregunta si hay espacio a la izquierda → si lo hay, dobla a la izquierda.
  - Si no, prueba a la derecha → si lo hay, dobla a la derecha.
  - Si tampoco puede, se queda quieto.

### Pruebas y Resultados:

**Tabla con pruebas realizadas (fecha, objetivo, qué pasó, problemas encontrados, mejoras):**

Fecha	Objetivo	¿Qué pasó?	Problemas encontrados	Mejoras realizadas
07/08/2025	Ensamblar estructura inicial	Se montaron las piezas impresas y	Algunas piezas de	Ajustar diseño antes de imprimir /

		motores'	impresión eran más grandes de lo esperado → hubo que lijar, recortar y perforar	prever margen de tolerancia
12/08/2025	Ajustar servo de dirección	El servo respondió pero no lograba el ángulo adecuado	Hubo que recalibrar grados del servomotor para que los giros sean más precisos	Programar ángulos correctos y fijar servo con más precisión
19/08/2025	Programar movimiento básico	Robot avanzó y giró de forma limitada	Dificultad para encontrar la librería correcta para controlar motores	Buscar librerías oficiales y buscar documentaci ón de los componentes (Data sheet)

### Reflexión:

**¿Qué aprendimos? (trabajo en equipo, ensayo y error, programación, perseverancia).**

Durante el desarrollo del proyecto **ONLY**, aprendimos a trabajar en equipo, distribuyendo roles y coordinando nuestras tareas mientras valorábamos las habilidades de cada integrante. Mejoramos nuestros conocimientos en programación, electrónica y diseño, integrando sensores, actuadores, y utilizando impresión 3D con PLA. A través del ensayo y error, comprendimos la importancia de probar, analizar errores y realizar mejoras continuas, desarrollando perseverancia y resiliencia ante los desafíos técnicos. Además, logramos conectar los conocimientos



de matemáticas y tecnología con un problema real, aplicando la teoría en soluciones concretas. Este proyecto fortaleció nuestra capacidad de resolver problemas, pensar creativamente y trabajar de manera autónoma, preparándonos para futuros desafíos en robótica e ingeniería.

### **Dificultades superadas:**

Durante el desarrollo del proyecto ONLY, enfrentamos varias dificultades que logramos superar gracias al trabajo en equipo y la perseverancia. Tuvimos que aprender a integrar correctamente los sensores ultrasónicos, ajustar los ángulos del servo de dirección y calibrar los motores para que el robot se desplazará de manera precisa. Además, lidiamos con errores en la programación y retrasos en la reacción ante obstáculos, lo que nos obligó a probar distintas soluciones mediante ensayo y error. También superamos desafíos relacionados con la coordinación de nuestras tareas y la planificación de actividades semana a semana. Gracias a estas experiencias, logramos mejorar continuamente el rendimiento del robot y fortalecer nuestras habilidades técnicas y de colaboración.

### **Posibles mejoras futuras:**

Para el futuro, nuestro robot ONLY podría mejorarse incorporando sensores adicionales para una mayor precisión en la detección de obstáculos y señales de tráfico. También planeamos rediseñar el prototipo para reducir sus dimensiones, logrando un diseño más compacto y dinámico. Además, sería útil optimizar el diseño mecánico para mayor estabilidad y velocidad en recorridos largos, así como perfeccionar la programación para que el robot pueda tomar decisiones más rápidas y complejas en situaciones imprevistas. Otra mejora sería implementar sistemas de retroalimentación automática que permitan ajustes en tiempo real durante la ejecución de las pruebas. Otra mejora significativa sería la implementación de una cámara que nos permita reconocer las señales de tráfico a través de IA. Estas mejoras nos permitirían aumentar la autonomía, eficiencia y desempeño del robot en competencias futuras y en posibles aplicaciones prácticas.

## **Conclusión:**

### **Preparación para la competencia:**

Para la preparación de la competencia, organizamos nuestro trabajo semana a semana, distribuyendo tareas según los roles de cada integrante: Guadalupe en electrónica, Celina en programación y Francisco en diseño. Además de cumplir con nuestros roles, nos ayudamos mutuamente para resolver problemas y mejorar el rendimiento del robot. Realizamos pruebas constantes del robot, empezando por verificar el hardware, luego probando movimientos básicos, detección de obstáculos y reconocimiento de señales, hasta integrar todas las funciones en el circuito completo. Cada ensayo nos permitió identificar errores, corregirlos y mejorar tanto el diseño como el código. Contamos con la guía y motivación de nuestros facilitadores, Claudio Almeida y Verónica Fonzalida, quienes nos ayudaron a optimizar nuestro robot y a planificar las actividades. Esta preparación meticulosa nos dio confianza para enfrentar la competencia con nuestro robot **ONLY**, asegurando que pudiera cumplir los desafíos de manera autónoma y eficiente.

### **¿Qué esperamos lograr con el robot?**

Con este robot, esperamos mejorar nuestras habilidades en robótica, programación, diseño y trabajo en equipo, aplicando los conocimientos que adquirimos en la práctica. Queremos que el robot sea capaz de desplazarse de manera autónoma, esquivar obstáculos, reconocer señales de tráfico y completar los circuitos de prueba con precisión y eficiencia. Además, buscamos aprender a enfrentar desafíos reales mediante ensayo y error, fortalecer nuestra creatividad y perseverancia, y prepararnos para competencias futuras, demostrando nuestra pasión y compromiso por la robótica. A largo plazo, también aspiramos a que nuestro proyecto tenga aplicaciones prácticas, como contribuir a soluciones tecnológicas en movilidad o rescate.

## Anexos (opcional):

### Código final y completo:

```
#include <Servo.h>

// ===== TB6612FNG - Canal A =====
const int PIN_STBY = 4;
const int PIN_PWMA = 5;    // PWM motor
const int PIN_AIN1 = 7;
const int PIN_AIN2 = 8;

int velocidad = 135;    // 0-255

// ===== Servo dirección =====
Servo direccion;
const int PIN_SERVO = 3;    // servo conectado a D3
int posicionServo = 90;    // inicial

// ===== Sensores ultrasónicos =====
const int trigIzq = A0, echoIzq = A1;
const int trigFront = A2, echoFront = A3;
const int trigDer = A4, echoDer = A5;

long distanciaIzq, distanciaFront, distanciaDer;
const int UMBRAL = 20;    // cm

// ===== Pulsador inicio =====
const int PIN_START = 13;
bool iniciado = false;

// ===== Pines TCS3200 =====
#define S0 11
#define S1 12
#define S2 9
#define S3 10
#define sensorOut 6
```

```
// Valores RGB de referencia
int rojoRef[3] = {238, 39, 55};
int verdeRef[3] = {68, 214, 44};
int tolerancia = 30;

#define N_CAL 10
#define N_HIST 5
int histR[N_HIST], histG[N_HIST], histB[N_HIST];
int idxHist = 0;

// ===== Funciones Motor =====
void motorAvanzar(int vel) {
    digitalWrite(PIN_AIN1, HIGH);
    digitalWrite(PIN_AIN2, LOW);
    analogWrite(PIN_PWMA, constrain(vel, 0, 135));
}

void motorDetener() {
    analogWrite(PIN_PWMA, 0);
}

void motorStandby(bool activo) {
    digitalWrite(PIN_STBY, activo ? HIGH : LOW);
}

// ===== Medir distancia =====
long medirDistancia(int trig, int echo) {
    digitalWrite(trig, LOW);
    delayMicroseconds(2);
    digitalWrite(trig, HIGH);
    delayMicroseconds(10);
    digitalWrite(trig, LOW);
    long duracion = pulseIn(echo, HIGH, 20000);
    long distancia = duracion * 0.034 / 2;
    return distancia == 0 ? 999 : distancia;
}

// ===== Funciones sensor de color =====
long readColorRaw(int s2, int s3){
    digitalWrite(S2, s2);
    digitalWrite(S3, s3);
    long total = 0;
```

```

    for(int i=0;i<5;i++){
        long lectura = pulseIn(sensorOut, LOW, 100000);
        if(lectura == 0) lectura = 10000;
        total += lectura;
    }
    return total/5;
}

int normalizarAuto(long lectura, long minVal, long maxVal){
    if(lectura < minVal) lectura = minVal;
    if(lectura > maxVal) lectura = maxVal;
    return map(lectura, maxVal, minVal, 0, 255);
}

bool dentroTolerancia(int valor, int referencia){
    return (valor >= referencia - tolerancia && valor <= referencia +
tolerancia);
}

String detectarColor(){
    // Calibración de luz
    long rawR[N_CAL], rawG[N_CAL], rawB[N_CAL];
    long minR=99999, maxR=0, minG=99999, maxG=0, minB=99999, maxB=0;

    for(int i=0;i<N_CAL;i++){
        rawR[i]=readColorRaw(LOW, LOW);
        rawG[i]=readColorRaw(HIGH, HIGH);
        rawB[i]=readColorRaw(LOW, HIGH);

        if(rawR[i]<minR) minR=rawR[i]; if(rawR[i]>maxR) maxR=rawR[i];
        if(rawG[i]<minG) minG=rawG[i]; if(rawG[i]>maxG) maxG=rawG[i];
        if(rawB[i]<minB) minB=rawB[i]; if(rawB[i]>maxB) maxB=rawB[i];
    }

    long sumR=0, sumG=0, sumB=0;
    for(int i=0;i<N_CAL;i++){
        sumR+=rawR[i]; sumG+=rawG[i]; sumB+=rawB[i];
    }

    int R = normalizarAuto(sumR/N_CAL, minR, maxR);
    int G = normalizarAuto(sumG/N_CAL, minG, maxG);
    int B = normalizarAuto(sumB/N_CAL, minB, maxB);

```

```
// Suavizado
histR[idxHist] = R;
histG[idxHist] = G;
histB[idxHist] = B;
idxHist = (idxHist + 1) % N_HIST;

long avgR=0, avgG=0, avgB=0;
for(int i=0;i<N_HIST;i++){
    avgR += histR[i];
    avgG += histG[i];
    avgB += histB[i];
}
avgR /= N_HIST; avgG /= N_HIST; avgB /= N_HIST;

if(dentroTolerancia(avgR, rojoRef[0]) &&
    dentroTolerancia(avgG, rojoRef[1]) &&
    dentroTolerancia(avgB, rojoRef[2])){
    return "ROJO";
}

else if(dentroTolerancia(avgR, verdeRef[0]) &&
        dentroTolerancia(avgG, verdeRef[1]) &&
        dentroTolerancia(avgB, verdeRef[2])){
    return "VERDE";
}

else return "DESCONOCIDO";
}

// ===== Setup =====
void setup() {
    pinMode(PIN_STBY, OUTPUT);
    pinMode(PIN_PWMA, OUTPUT);
    pinMode(PIN_AIN1, OUTPUT);
    pinMode(PIN_AIN2, OUTPUT);

    pinMode(trigIzq, OUTPUT);
    pinMode(echoIzq, INPUT);
    pinMode(trigFront, OUTPUT);
    pinMode(echoFront, INPUT);
    pinMode(trigDer, OUTPUT);
    pinMode(echoDer, INPUT);
}
```

```
pinMode(PIN_START, INPUT);

direccion.attach(PIN_SERVO);
direccion.write(posicionServo);

// Pines TCS3200
pinMode(S0, OUTPUT);
pinMode(S1, OUTPUT);
pinMode(S2, OUTPUT);
pinMode(S3, OUTPUT);
pinMode(sensorOut, INPUT);

digitalWrite(S0, HIGH);
digitalWrite(S1, LOW);

motorStandby(true);
motorDetener();

Serial.begin(9600);
}

// ===== Loop =====
void loop() {
  if (!iniciado) {
    if (digitalRead(PIN_START) == HIGH) {
      iniciado = true;
      Serial.println("Robot iniciado!");
    } else {
      return;
    }
  }

  // Leer distancias
  distanciaFront = medirDistancia(trigFront, echoFront);
  distanciaIzq    = medirDistancia(trigIzq, echoIzq);
  distanciaDer    = medirDistancia(trigDer, echoDer);

  Serial.print("Front: "); Serial.print(distanciaFront);
  Serial.print(" | Izq: "); Serial.print(distanciaIzq);
  Serial.print(" | Der: "); Serial.print(distanciaDer);

  if (distanciaFront > UMBRAL) {
```

```
// Avanzar recto
posicionServo = 90;
direccion.write(posicionServo);
motorAvanzar(velocidad);
}
else {
    // === Antes de mirar izquierda/derecha, ver el color ===
    String color = detectarColor();
    Serial.print(" | Color: "); Serial.println(color);

    if (color == "ROJO") {
        posicionServo = 155;
        direccion.write(posicionServo);
        motorAvanzar(velocidad);
        delay(500);
        posicionServo = 90;
        direccion.write(posicionServo);
    }
    else if (color == "VERDE") {
        posicionServo = 30;
        direccion.write(posicionServo);
        motorAvanzar(velocidad);
        delay(500);
        posicionServo = 90;
        direccion.write(posicionServo);
    }
    else {
        // Si no detecta color válido, usar lógica normal
        if (distanciaIzq > UMBRAL) {
            posicionServo = 155;
            direccion.write(posicionServo);
            motorAvanzar(velocidad);
            delay(500);
            posicionServo = 90;
            direccion.write(posicionServo);
        }
        else if (distanciaDer > UMBRAL) {
            posicionServo = 30;
            direccion.write(posicionServo);
            motorAvanzar(velocidad);
            delay(500);
            posicionServo = 90;
        }
    }
}
```



```
        direccion.write(posicionServo);  
    }  
    else {  
        motorDetener();  
    }  
}  
}  
  
delay(100);  
}
```