

WRO

Futuros ingenieros 2025

Diario de ingeniería

Nombre del Equipo: Encore

Miembros del equipo:

José Heráldez

Abel Herrera

Pablo González

Entrenador:

Octavio Herrera

Tabla de Contenidos

1	Resumen ejecutivo.....	3
2	Introducción	4
2.1	Objetivos:.....	4
3	Proceso de diseño y desarrollo	4
3.1	Material utilizado.....	5
3.2	Configuración general	9
3.2.1	Fotos del robot	9
3.2.2	Otros detalles generales	12
3.3	Descripción del chasis.....	13
3.4	Sistema de dirección	14
3.4.1	Proceso de Diseño e Implementación.....	15
3.4.2	Análisis de Componentes del Sistema	15
3.4.3	Justificación y Ventajas del Diseño	16
3.5	Sistema de propulsión.....	17
3.5.1	Diseño inicial y problemas de torque	17
3.5.2	Iteración y solución	17
3.5.3	Montaje del motor y estructura de soporte.....	18
3.5.4	diseño del eje y estabilidad.....	18
3.6	Diseño eléctrico	19
3.6.1	Controladores.....	22
3.6.2	Sensores	22
3.6.3	Otros	23
3.7	Gestión de energía	23
3.8	Manejo de Obstáculos y Fusión de Sensores.....	23
4	Diseño del código/programación	25
4.1	Código del Arduino Nano	25
4.2	Código de la Raspberry Pi 5	26
5	Resultados y demostración	27
6	Conclusiones.....	27
7	Aspectos por mejorar	28

1 Resumen ejecutivo

El presente documento detalla el proceso de diseño iterativo y la arquitectura del robot "Encore", un vehículo autónomo desarrollado para la competencia WRO Futuros Ingenieros 2025. El proyecto demuestra la aplicación de principios de ingeniería para resolver desafíos del mundo real, partiendo de un chasis prefabricado y evolucionando hacia un sistema altamente personalizado.

La arquitectura del robot se fundamenta en un sistema de control distribuido, que combina la potencia de una Raspberry Pi 5 para la visión por computadora y la toma de decisiones, con la fiabilidad en tiempo real de un Arduino Nano que gestiona todos los sensores y actuadores de bajo nivel. Esta separación de tareas fue una mejora crítica implementada para garantizar la estabilidad y precisión del control.

A lo largo del desarrollo, se superaron obstáculos significativos, incluyendo:

- Una optimización del chasis que redujo el peso total en un 28% al sustituir el metal por acrílico.
- La resolución de un déficit de torque en la propulsión mediante la inversión de la relación de engranajes a una de reducción 7:9.
- El rediseño del sistema de alimentación para crear circuitos de potencia independientes, solucionando problemas de reinicios por bajo voltaje.

El sistema de navegación actual fusiona datos de una cámara y un sensor ultrasónico para ejecutar maniobras de evasión de obstáculos, controlados por una lógica de software maestro-esclavo. Este diario de ingeniería documenta cada una de estas decisiones, presentando un vehículo que no solo cumple con los requisitos de la competencia, sino que también representa una solución robusta a complejos desafíos de ingeniería.

2 Introducción

Esta documentación (Diario de ingeniería) le presentará el desarrollo del proyecto Encore, detallando las secciones listadas en las reglas de la WRO-Futuros ingenieros.


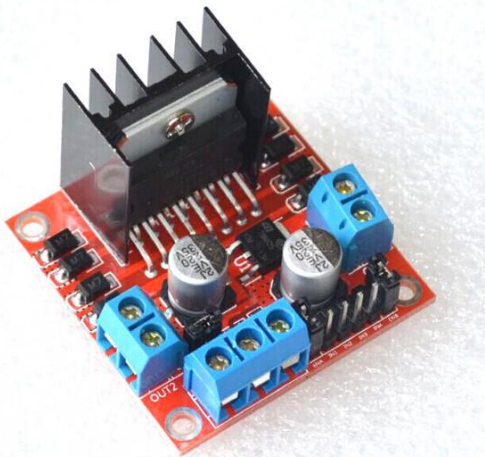
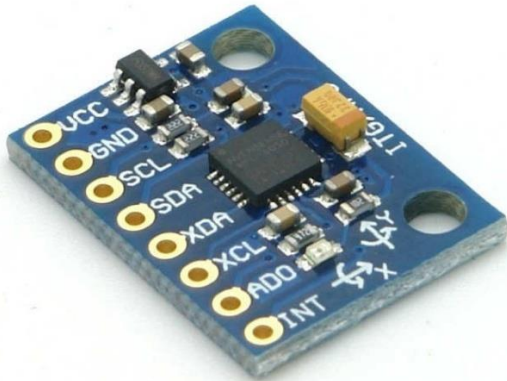
2.1 Objetivos:

- **Objetivo general:** Desarrollar un Sistema de Navegación Autónoma Fiable: Diseñar, construir y programar un vehículo capaz de completar de manera consistente y repetida el circuito oficial, utilizando una fusión de datos de la cámara y sensores de proximidad para la localización y el seguimiento de la trayectoria.





3 Proceso de diseño y desarrollo




3.1 Material utilizado

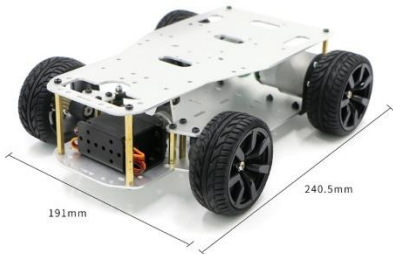
A continuación, se presenta una lista con todos los materiales utilizados en el proyecto.

<u>Material</u>	<u>Descripción</u>	<u>Imagen</u>
Raspberry Pi 5 8GB	El cerebro del robot; un potente microordenador que ejecuta el código y procesa la información.	
L298N	Un controlador de motores que permite controlar la dirección y velocidad de los motores DC.	
MPU6050	Un sensor de movimiento (acelerómetro y giroscopio) para medir la orientación y la inclinación del robot.	

<p>HC-SR04</p>	<p>Un sensor ultrasónico que mide la distancia a los objetos para la detección de obstáculos.</p>	 <p>The image shows an HC-SR04 ultrasonic sensor module. It is a blue printed circuit board (PCB) with two large, circular, silver-colored mesh speakers on either side of a central ultrasonic transducer. The text 'HC-SR04' is printed on the board. Four pins are visible at the bottom, labeled 'vcc', 'trig', 'echo', and 'gnd'.</p>
<p>webcam genérica</p>	<p>La cámara del robot, utilizada para la visión por computadora (detectar colores, formas, etc.).</p>	 <p>The image shows a generic webcam with a black plastic housing and a silver-colored lens. It has a small, adjustable stand at the bottom.</p>
<p>Servomotor genérico</p>	<p>Un motor para movimientos precisos y controlados, usado para la dirección.</p>	 <p>The image shows a generic servo motor. It is a black plastic housing with a gold-colored potentiometer on top. The text 'Power' is visible on the side. Three wires (red, white, and black) are connected to the bottom. The URL 'www.pololu.com' is visible at the bottom of the image.</p>
<p>Protoboard</p>	<p>Placa para construir los circuitos sin la necesidad de soldar.</p>	 <p>The image shows a standard protoboard, which is a white plastic board with a grid of holes for electronic components. It has two rows of pins on the left and right sides, labeled '1' through '20'.</p>

<p>Motor DC (metal)</p>	<p>El motor principal con engranajes de metal, encargado de la propulsión y el movimiento de las ruedas.</p>	
<p>Powerbank Romoss (10000mAh)</p>	<p>Fuente de poder exclusiva para la Raspberry Pi 5.</p>	
<p>Batería 9v</p>	<p>Fuente de poder exclusiva para la parte mecánica y sensores.</p>	
<p>Cables Dupont</p>	<p>Interconexiones entre los módulos.</p>	

<p>tornillería</p>	<p>Sujeción del chasis y parte del motor.</p>	
<p>Láminas de acrílico</p>	<p>Bases para el chasis, reemplazaron a las láminas de metal pesadas.</p>	
<p>Piezas LEGO-Technic</p>	<p>Sujetan el motor, conforman el eje principal y sus engranajes.</p>	

Chasis Akerman Prefabricado	Utilizado como base para la construcción del robot.	
-----------------------------	---	--

3.2 Configuración general

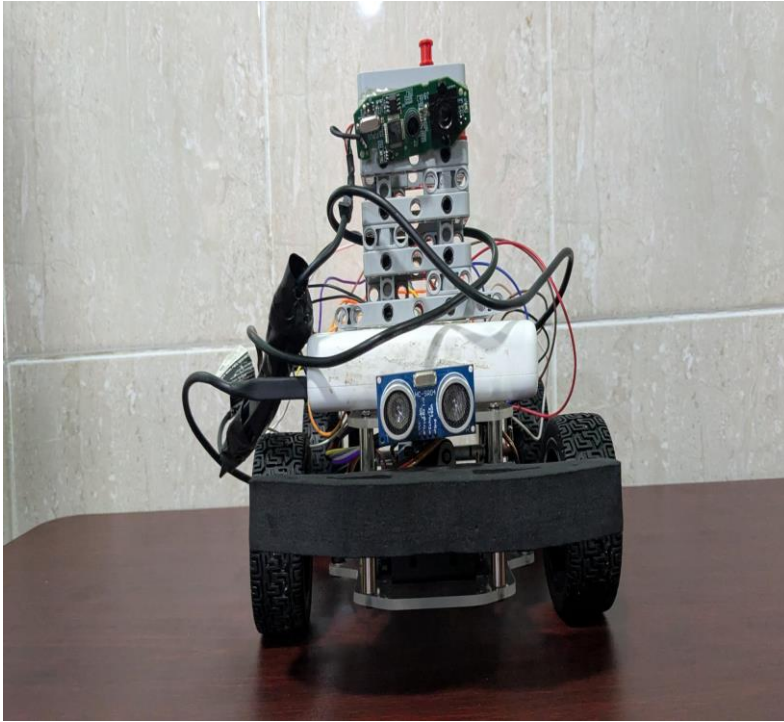
El diseño del robot parte de un kit Akerman desmontado, utilizado con el objetivo de no sobrecargar al equipo en diseñar una estructura de cero y poder evidenciar avances rápido, la elección de un chasis prefabricado fue buena para tener una base sólida para trabajar, pero que trajo múltiples problemas como: la necesidad de cambiar el sistema de propulsión, problemas de peso excesivo, complejidad a la hora de mejorar el sistema Akerman de dirección.



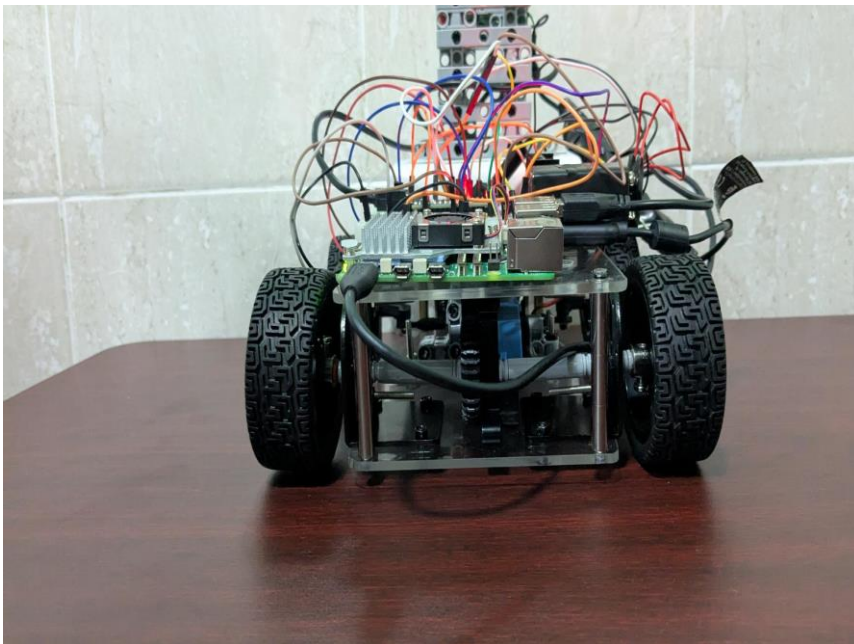
3.2.1 Fotos del robot

Se presentan fotos del robot desde cada ángulo.

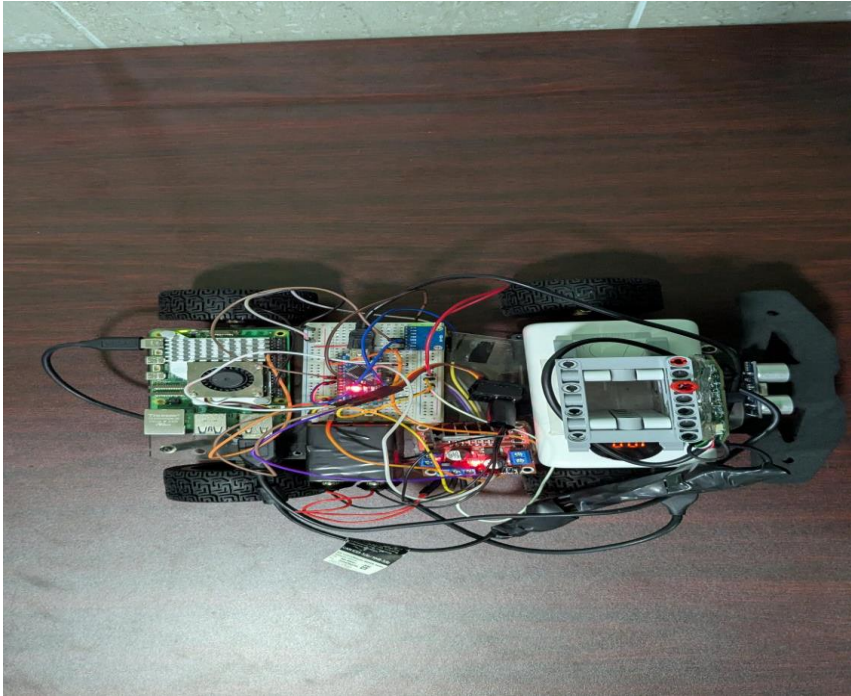
- Frontal:



- Trasera:



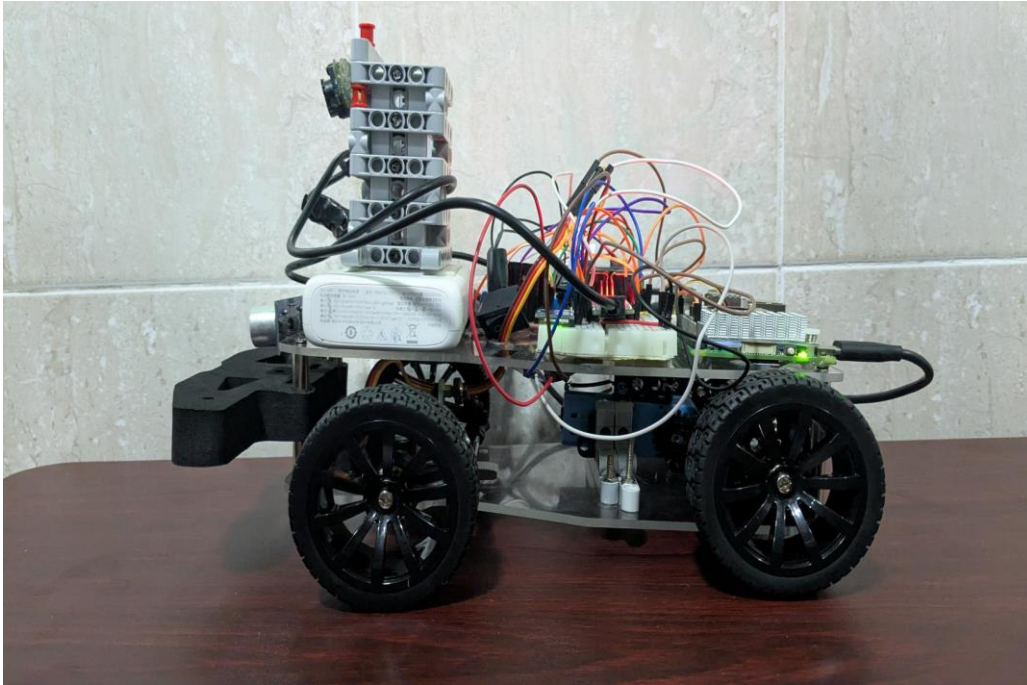
- Superior:



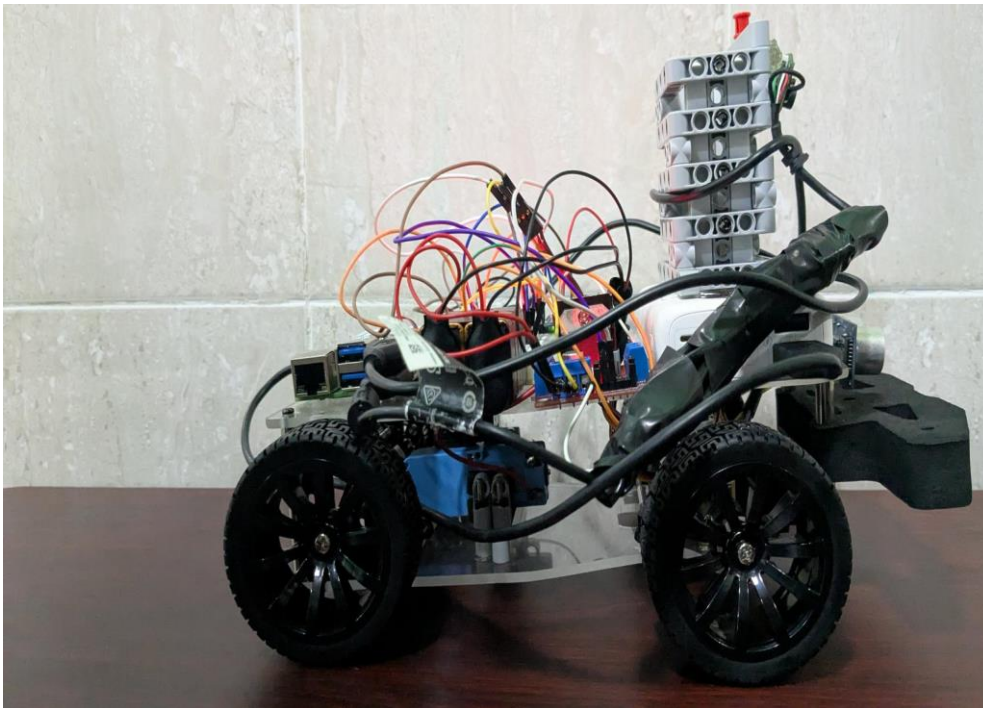
- Inferior:



- Lateral 1:



- Lateral 2:



3.2.2 Otros detalles generales

- Propulsión: Motor DC fijado con piezas de LEGO Technic.
- Dirección: Sistema Akerman Mejorado.
- Dimensiones: Largo: 27.5 cm, 18.5 cm de ancho y 19.5 de alto.

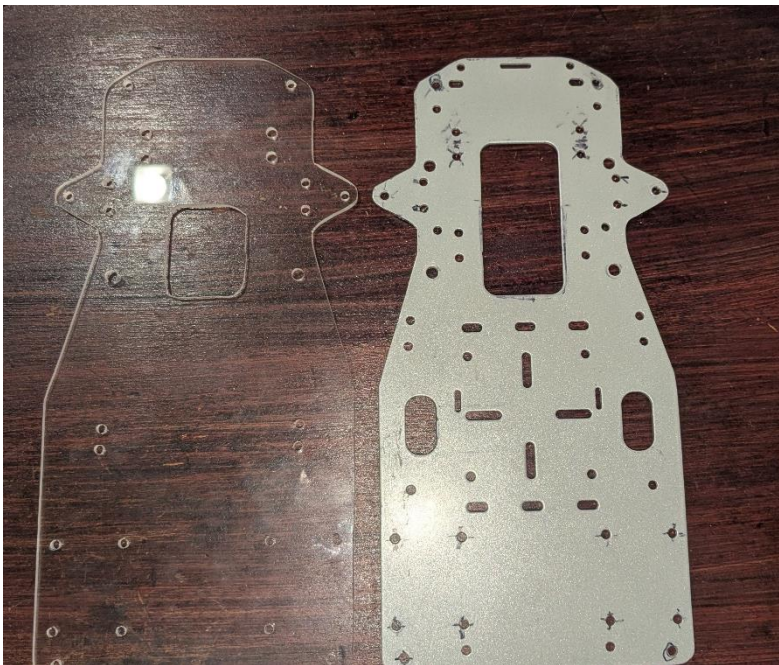
- Energía: Una powerbank de 10000mAh y dos baterías de 9v.
- Manejo de circuito: Protoboard con cables Dupont.

3.3 Descripción del chasis

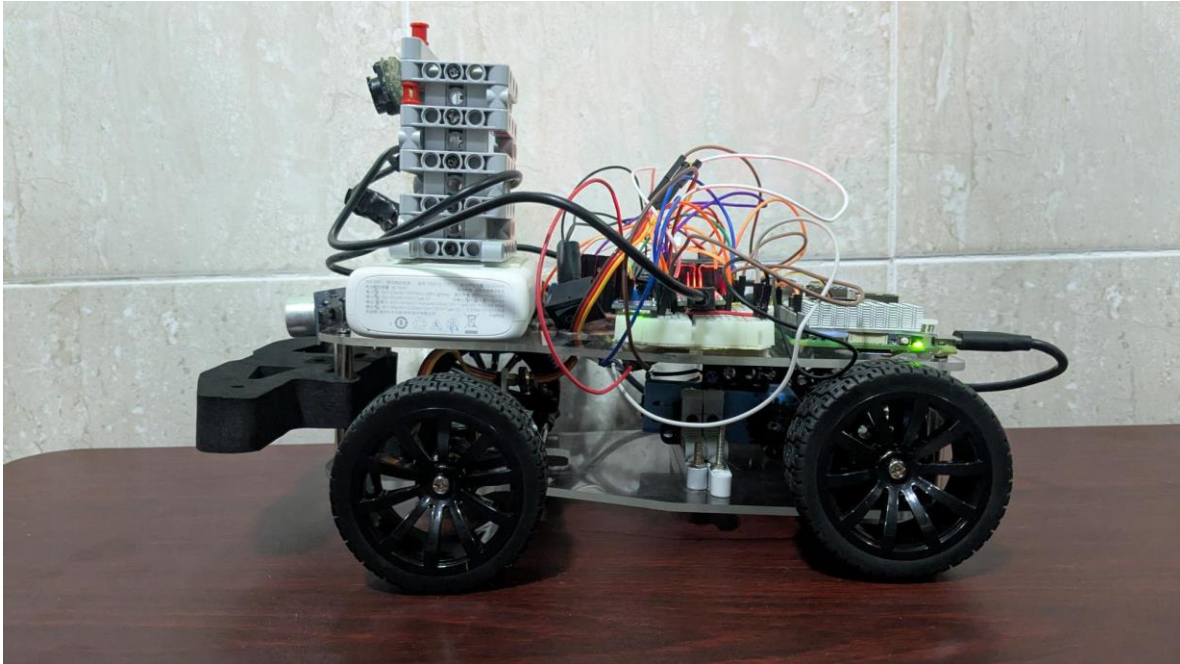
Inicialmente, el prototipo se construyó con láminas de metal por su alta resistencia (También eran las que incluía el kit). Sin embargo, las pruebas revelaron que el metal aumentaba excesivamente el peso total del robot (1700 g), lo que limitaba su agilidad y sobrecargaba los motores.

Para solucionar esto, se decidió migrar a láminas de acrílico cortadas por el equipo, lo que resultó en dos ventajas clave:

- **Reducción de Peso:** El peso de las placas del chasis se redujo de 520 g (2 placas de 260 g) a solo **140 g** (2 placas de 70 g). Esto fue un factor crucial para disminuir el peso total del robot a **1217 g**.
- **Simplificación del Diseño:** Al cortar el acrílico, se incluyeron únicamente las perforaciones necesarias, optimizando y limpiando el diseño final en comparación con las placas de metal genéricas.



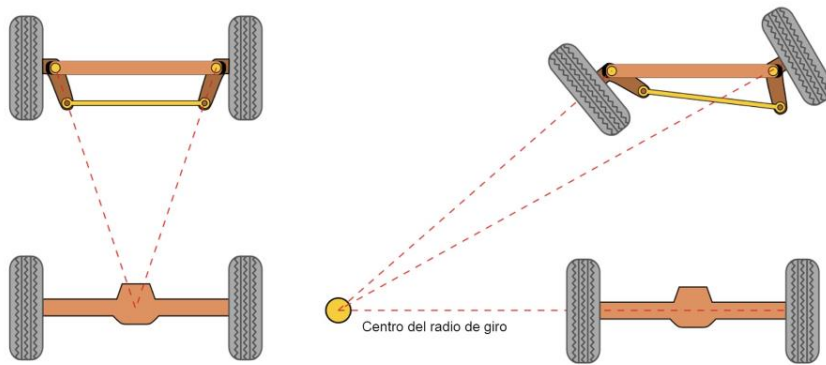
El hecho de utilizar dos niveles no es mero capricho, es una decisión clave para tener más superficie de trabajo, separando las áreas clave del robot: electrónica en la parte superior y componentes mecánicos abajo.



3.4 Sistema de dirección

La capacidad de un vehículo autónomo para navegar con precisión su entorno depende críticamente de la efectividad de su sistema de dirección. Para nuestro proyecto, hemos implementado una **geometría de dirección Ackerman**, un diseño mecánicamente sofisticado que se inspira directamente en su aplicación universal en vehículos reales. La elección de este sistema, en lugar de alternativas más simples como la dirección por pivote central, se fundamenta en su capacidad para ejecutar giros cinemáticamente correctos, minimizando el deslizamiento lateral de las ruedas y reduciendo el estrés mecánico sobre el chasis.

El principio Ackerman asegura que, durante una curva, la rueda interior gire con un ángulo mayor que la exterior. Esta diferencia de ángulos permite que ambas ruedas tracen arcos concéntricos con un centro de giro común, lo que resulta en un movimiento más estable, predecible y eficiente.



3.4.1 Proceso de Diseño e Implementación

La construcción de este mecanismo representó un desafío de diseño considerable, ya que no contábamos con un kit predefinido o instrucciones. El proceso fue iterativo y se dividió en varias fases:

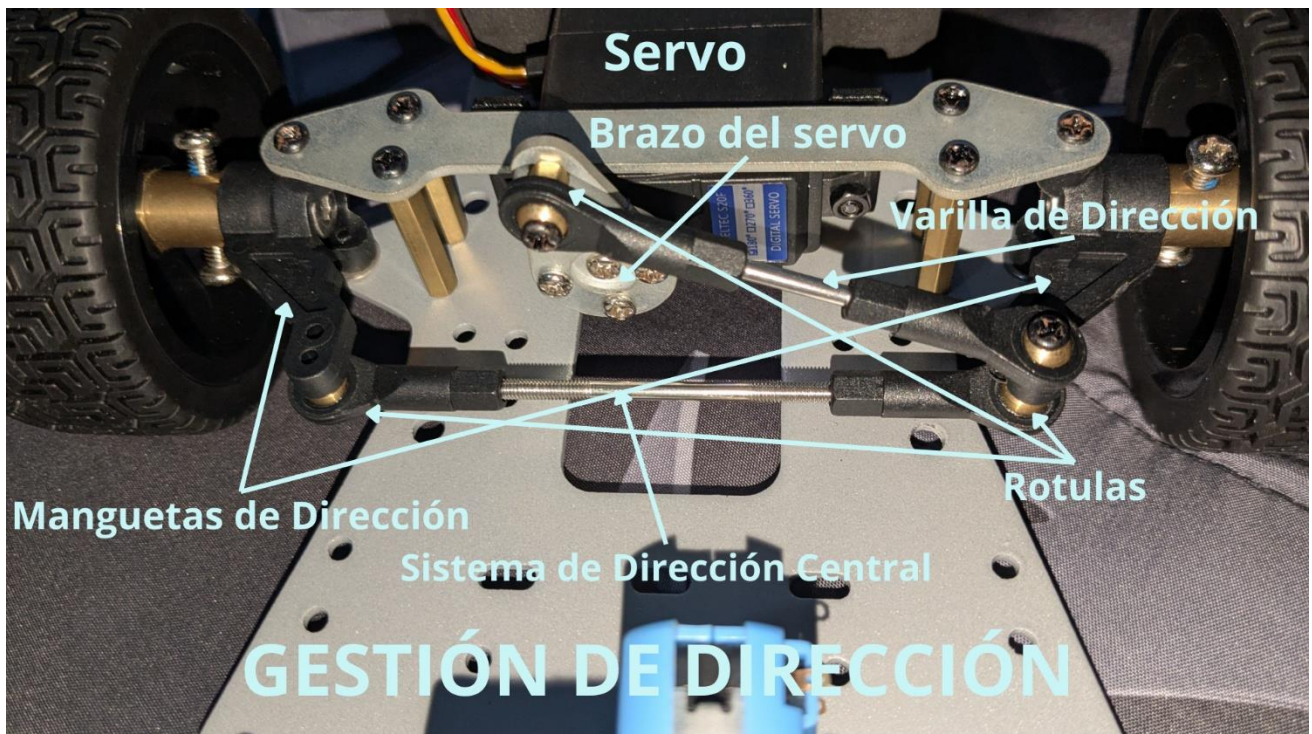
1. **Investigación y Conceptualización:** La primera etapa consistió en un estudio detallado de diagramas técnicos y principios de la geometría Ackerman. Analizamos cómo la disposición trapezoidal de las varillas de dirección lograba la diferencia de ángulos necesaria.
2. **Prototipado y Fabricación:** A partir de los conceptos estudiados, procedimos a diseñar y fabricar los componentes. Esto implicó la creación de soportes a medida y la selección de varillas de longitudes específicas para construir el sistema desde cero.
3. **Calibración y Ajuste:** La fase final fue un meticuloso proceso de ajuste de la longitud de las varillas y los puntos de pivote para aproximarnos lo más posible a la geometría ideal, asegurando un funcionamiento suave y preciso.

3.4.2 Análisis de Componentes del Sistema

Nuestro mecanismo de dirección es un sistema integrado donde cada componente cumple una función específica para traducir el comando electrónico del servo en un movimiento mecánico coordinado.

- **Actuador Principal (Servomotor):** El origen del movimiento es un **servomotor de alto torque**. La selección de un servo potente fue crucial para superar la inercia y la fricción estática del sistema, así como para mantener el ángulo de dirección de manera firme contra las fuerzas dinámicas que actúan sobre las ruedas mientras el vehículo está en movimiento.

- **La Tirantería de Dirección (Steering Linkage):** Este es el corazón del mecanismo Ackerman y se compone de varios elementos interconectados:
 - **Brazo del Servo y Varilla de Empuje:** El servo transfiere su rotación a través de un brazo (servo horn) a una varilla de empuje principal.
 - **Placa de Dirección:** Esta varilla actúa sobre una placa de dirección central (en nuestro caso, una rueda modificada para este propósito), que pivota y convierte el movimiento lineal de la varilla de empuje en el movimiento lateral necesario para las dos ruedas.
 - **Varillas de Acoplamiento (Tie Rods):** Desde la placa de dirección, dos varillas de acoplamiento se conectan a los soportes de cada rueda. La longitud y el ángulo de estas varillas son los parámetros críticos que definen la geometría Ackerman.
 - **Soportes de Rueda Pivotantes (Steering Knuckles):** Para cada rueda directriz, se diseñaron soportes robustos que permiten un pivote suave sobre un eje vertical. Estos soportes no solo anclan la rueda al chasis, sino que también sirven como punto de conexión para las varillas de acoplamiento, asegurando la rigidez estructural y la precisión del giro.



3.4.3 Justificación y Ventajas del Diseño

La implementación de la geometría Ackerman, aunque compleja, proporciona beneficios tangibles que justifican el esfuerzo de ingeniería.

- **Precisión de Maniobra:** Al eliminar casi por completo el deslizamiento lateral de las ruedas, el vehículo sigue trayectorias mucho más predecibles y exactas.
- **Estabilidad Dinámica:** El giro suave y coordinado mejora la estabilidad general del robot, especialmente al entrar y salir de curvas a una velocidad considerable.
- **Eficiencia Energética:** Al reducir la fricción por arrastre de las ruedas, el sistema de propulsión trabaja de manera más eficiente, ya que no necesita vencer una resistencia innecesaria, lo que se traduce en un menor consumo de energía.
- **Durabilidad Mecánica:** Un movimiento geométricamente correcto reduce las tensiones y el desgaste en los componentes del chasis, los ejes y las propias ruedas, aumentando la vida útil y la fiabilidad del sistema.

3.5 Sistema de propulsión

El sistema de propulsión es responsable de convertir la energía eléctrica del motor en movimiento mecánico. Nuestro diseño evolucionó a través de pruebas prácticas para encontrar el equilibrio adecuado entre velocidad y torque, utilizando componentes LEGO Technic y tornillería para asegurar la rigidez del conjunto.

3.5.1 Diseño inicial y problemas de torque

Inicialmente, nuestro diseño priorizaba la velocidad máxima teórica. Para lograrlo, implementamos una configuración de engranajes de sobremarcha (**overdrive**).

- **Configuración Inicial:** El engranaje motriz (conectado al motor) tenía 36 dientes, y el engranaje conducido (en el eje de las ruedas) tenía 28 dientes, resultando en una relación de 9:7.
- **Observaciones:** Durante las pruebas iniciales, el vehículo fue incapaz de romper la inercia estática por sí solo. Requería un empuje manual para comenzar a moverse y asistencia constante para mantener el avance.
- **Diagnóstico:** La configuración de sobremarcha multiplicaba la velocidad, pero reducía el torque disponible en las ruedas a un nivel insuficiente para vencer la fricción y el peso del robot.

3.5.2 Iteración y solución

Para corregir este déficit de torque y permitir la operación autónoma, se tomó la decisión de invertir la relación de transmisión, pasando a una configuración de reducción siendo esta una solución instantánea y permanente frente a incrementar el amperaje.

- **Configuración Actual:** Se invirtieron los engranajes. El engranaje motriz (motor) ahora tiene 28 dientes, y el engranaje conducido (eje) tiene **36** dientes.
- **Relación de Transmisión 7:9:** Esta nueva configuración resulta en una relación de 7:9.
- **Resultado:** Esta relación multiplica el torque del motor a expensas de la velocidad máxima. Este cambio permitió que el robot tuviera la fuerza necesaria para arrancar desde el reposo y moverse de manera autónoma y fiable, restando velocidad, pero era un precio que pagar aceptable.

3.5.3 Montaje del motor y estructura de soporte

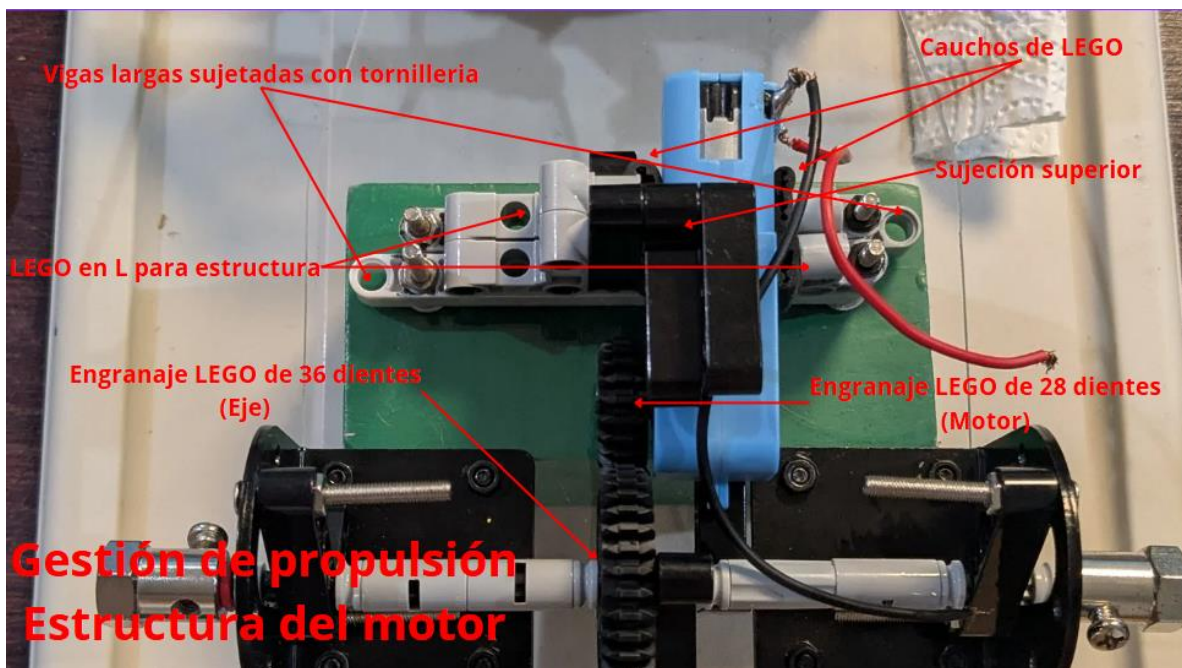
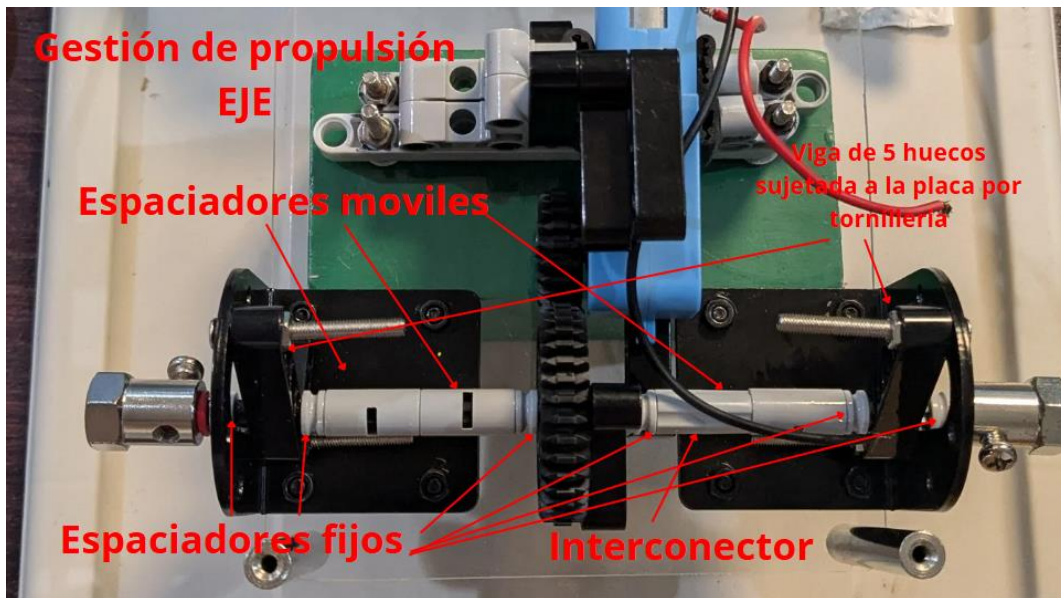
Independientemente de la relación de transmisión, la estabilidad del motor es crucial, debido a eso se eligió fijar el motor haciendo presión con gomas de LEGO.

- **Fijación al Chasis:** La estructura que soporta el motor se ancló firmemente al chasis utilizando vigas largas de LEGO Technic y cuatro tornillos pasantes. Esto previene la flexión durante la operación.
- **Sujeción del Motor:** El motor se fija a estas vigas mediante piezas LEGO en forma de "L". Para maximizar la estabilidad y absorber las vibraciones, se añadieron cauchos de LEGO en los puntos de contacto, asegurando que el motor no se desplace.

3.5.4 diseño del eje y estabilidad

La estabilidad del eje trasero es fundamental para una transmisión de potencia eficiente debido a eso, elegimos realizar y fortalecer nuestro eje trasero con los recursos que teníamos, que en este caso eran las piezas de LEGO Technic.

- **Construcción del Eje:** Se utiliza un eje en cruz de LEGO. Para conectar el engranaje central de 36 dientes, se emplean dos sub-ejes unidos por un interconector, asegurando que la fuerza se transmita de manera centrada.
- **Reducción de Holgura:** Se colocaron espaciadores fijos y móviles a lo largo del eje para minimizar la vibración y el desplazamiento lateral “juego” de las ruedas.
- **Ajuste de Altura:** Para elevar la altura libre al suelo del chasis, el soporte del eje se montó sobre una viga transversal de 5 huecos, se puede notar que el eje trasero esta más bajo que el delantero, pero no tiene ningún efecto sobre el robot.



3.6 Diseño eléctrico

El diseño eléctrico ha evolucionado hacia una **arquitectura de doble procesador** para optimizar el rendimiento y la fiabilidad. El sistema ahora combina la potencia de procesamiento de una **Raspberry Pi 5** con la precisión en tiempo real de un microcontrolador **Arduino Nano**.

Esta arquitectura distribuida permite una división de tareas eficiente:

- **Raspberry Pi 5 (Cerebro Principal):** Se dedica exclusivamente a las tareas de alta carga computacional, como el procesamiento de imágenes de la webcam para la visión por computadora y la toma de decisiones estratégicas.
- **Arduino Nano (Co-procesador de Tiempo Real):** Funciona como un controlador de bajo nivel. Su función es gestionar todas las interacciones de hardware que requieren una temporización precisa, liberando a la Raspberry Pi de esta carga.

Esta separación garantiza que las señales de control para los motores y servos sean estables y sin el "jitter" (temblor) que puede ocurrir en un sistema operativo no determinista como el de la Pi, mejorando drásticamente la precisión de los movimientos del robot.

Diagrama

de

circuito

gráfico:

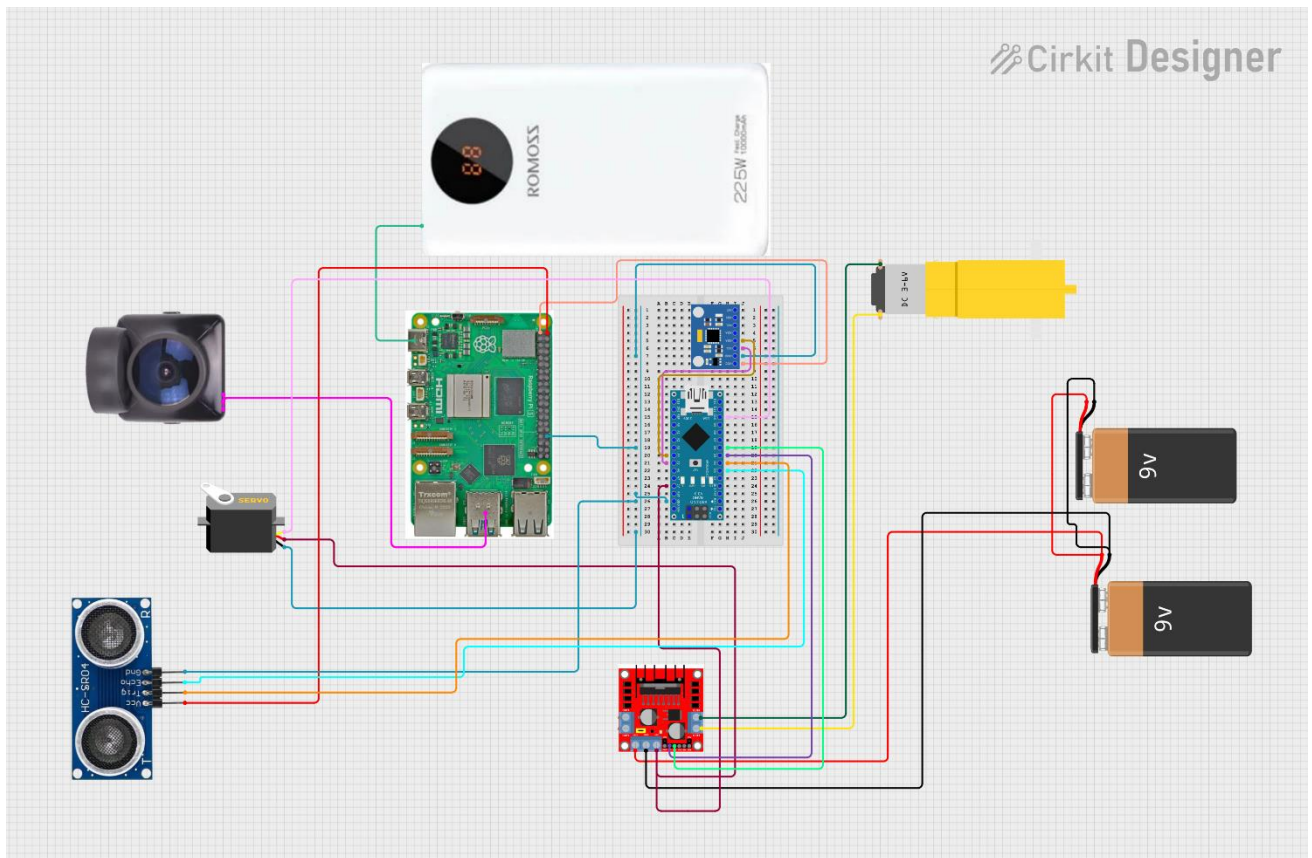
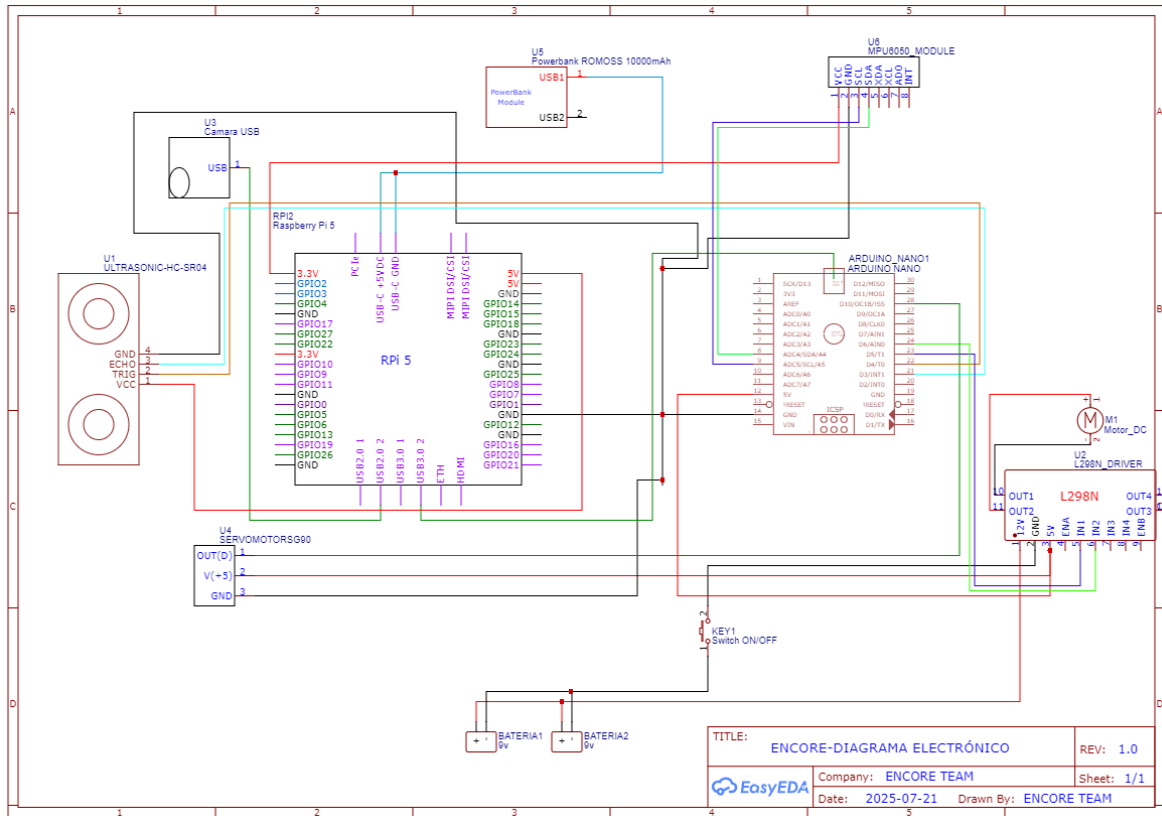


Diagrama esquemático:



3.6.1 Controladores

Se ha implementado una arquitectura de controlador dual:

- **Raspberry Pi 5 8GB:** Actúa como el controlador de alto nivel, ejecutando el programa principal en Python y los algoritmos de visión artificial.
- **Arduino Nano:** Se ha añadido como un co-procesador dedicado. Ahora es el responsable directo de gestionar las señales de control del driver **L298N**, el **servomotor**, el sensor ultrasónico **HC-SR04** y el giroscopio **MPU6050**. Se comunica con la Raspberry Pi a través de una conexión serie por USB, recibiendo comandos de alto nivel y ejecutándolos con precisión.

3.6.2 Sensores

Para los sensores nos focalizamos en hacer trabajar a la par un módulo ultrasónico HC-SR04 junto a una cámara, utilizándolas para que el sensor ultrasónico determine la distancia y la cámara determine el color y posición del objeto. Esta integración fue bastante complicada, principalmente por el distinto tiempo de refresco de cada sensor, pues el ultrasónico le envía información mucho más lento que la cámara al microcontrolador.

3.6.3 Otros

- **Motor DC:** El motor DC utilizado fue seleccionado debido a su versatilidad, bajo perfil y bajo peso, siendo éstas clave para que pueda formar parte del chasis sin sumar tanto peso, para este motor decidimos conseguir motores que vinieran con engranajes de metal para que aumentara su resistencia, también por su relación reductora 1:90.
- **Protoboard:** Para realizar algunas conexiones para el sensor ultrasónico, resistencia y también para el giroscopio. Para el equipo era una manera fácil de poder tener todo conectado sin la necesidad de tener una PCB exclusiva.

3.7 Gestión de energía

La gestión de energía fue rediseñada para soportar la nueva arquitectura de doble procesador y garantizar la estabilidad del sistema, un desafío que surgió durante las pruebas.

Inicialmente, se intentó alimentar el Arduino Nano directamente desde un puerto USB de la Raspberry Pi. Sin embargo, las pruebas revelaron que la carga combinada del Arduino y sus periféricos (servo, sensores) provocaba que la demanda total de corriente superara los 3A de la power bank. Esto causaba caídas de voltaje críticas que reiniciaban la Raspberry Pi de forma inesperada.

La solución implementada fue crear dos circuitos de alimentación independientes pero referenciados:

- **Alimentación Principal (Raspberry Pi):** La **Powerbank Romoss de 10000mAh** se dedica exclusivamente a alimentar la Raspberry Pi 5, asegurando un suministro estable de 5V/3A.
- **Alimentación Secundaria (Periféricos):** Las **dos baterías de 9V en paralelo** alimentan el controlador de motores L298N. La salida regulada de 5V de este mismo controlador se utiliza para alimentar el Arduino Nano, proporcionándole una fuente de energía estable e independiente de la Pi.

Esta configuración distribuye la carga, previene los reinicios por bajo voltaje y demuestra una solución robusta a un problema crítico de alimentación que surgió durante el desarrollo.

3.8 Manejo de Obstáculos y Fusión de Sensores

La capacidad del robot para navegar el circuito de manera autónoma depende fundamentalmente de su habilidad para detectar, clasificar y esquivar obstáculos de forma fiable. Nuestro enfoque para este desafío se basa en la **fusión de sensores**, una técnica que combina los datos de múltiples sensores para obtener una percepción del entorno más completa y robusta de lo que un solo sensor podría lograr.

Esta estrategia nos permite compensar las debilidades inherentes de cada sensor, utilizando una cámara para la **clasificación cualitativa** (¿qué es y de qué color?) y un sensor ultrasónico para la **medición cuantitativa** (¿a qué distancia está?).

Arquitectura de Percepción

El sistema de percepción se divide entre la Raspberry Pi y el Arduino Nano, asignando cada tarea al procesador más adecuado para ella:

1. **Sensor de Visión (Webcam):** Conectada a la **Raspberry Pi**, la cámara actúa como el sistema de visión principal. Su función es la detección y clasificación de objetos en el campo de visión. Utilizando algoritmos de visión por computadora, el programa en la Pi puede identificar los contornos de un obstáculo y, crucialmente, determinar su color para tomar decisiones estratégicas.
2. **Sensor de Proximidad (HC-SR04):** Conectado al **Arduino Nano**, el sensor ultrasónico se encarga de una única tarea: medir la distancia al objeto más cercano de forma continua y precisa. El Arduino es ideal para esta función, ya que puede gestionar los pulsos de tiempo del sensor sin las interrupciones de un sistema operativo, garantizando mediciones consistentes.

Algoritmo de Decisión y Ejecución de Maniobra

El proceso para esquivar un obstáculo sigue una secuencia lógica y programada, que se ejecuta en un bucle constante:

1. **Detección y Clasificación (Pi):** El programa de visión de la Raspberry Pi analiza el flujo de video de la cámara. Cuando detecta un contorno que corresponde a un obstáculo, extrae su color predominante. Esta información es clave para determinar la dirección de la evasión (por ejemplo, un color asignado a la derecha y otro a la izquierda).
2. **Consulta de Distancia (Pi → Nano):** Simultáneamente, la Raspberry Pi solicita constantemente la distancia actual al Arduino a través de la comunicación serie. El Arduino responde inmediatamente con el último valor medido por el sensor HC-SR04.
3. **Disparo de la Maniobra:** La lógica en la Raspberry Pi evalúa continuamente la distancia recibida. Cuando esta distancia cruza un umbral de seguridad predefinido (ej. 15 cm), se activa el estado de "maniobra de evasión".
4. **Ejecución del Giro (Pi → Nano → Servo):** La Pi envía un comando específico al Arduino (ej. SERVO:60), indicando el ángulo de giro necesario basado en el color detectado previamente. Para mantener la estabilidad y el control, el ángulo de giro del servomotor se ha limitado a un máximo de **±60 grados**. El Arduino recibe este comando y genera la señal PWM precisa para mover el servo a la posición exacta.

5. **Implementación del Temporizador de Seguridad:** Se identificó un problema crítico durante las pruebas: una vez que el robot comienza a girar, la cámara (montada al frente) deja de ver el obstáculo. Si el giro se detuviera en ese instante, la parte trasera del robot podría colisionar con el borde del obstáculo. Para resolver esto, se implementó un **temporizador de seguridad**. Después de que la cámara pierde de vista el objeto, el robot continúa girando durante un breve periodo de tiempo adicional (ej. 0.5 segundos). Este margen asegura que todo el chasis haya superado el obstáculo de forma segura antes de enderezar la trayectoria.

Este algoritmo, que combina la clasificación por color, la medición precisa de distancia y una lógica de temporización, permite al robot no solo reaccionar a los obstáculos, sino hacerlo de una manera inteligente y segura.

4 Diseño del código/programación

La arquitectura del software del robot se basa en un modelo de **control distribuido** que aprovecha las fortalezas complementarias de la Raspberry Pi 5 y el Arduino Nano. Ambos procesadores trabajan en conjunto, comunicándose a través del puerto serie (USB) para lograr una operación robusta y eficiente.

4.1 Código del Arduino Nano

El programa del Arduino, escrito en C++, está diseñado para ser un controlador de bajo nivel, fiable y determinista. Su única responsabilidad es interactuar directamente con el hardware, liberando a la Raspberry Pi de las tareas que requieren una temporización precisa.

- **Funciones Principales:**

- **Gestión de Actuadores:** Controla directamente el driver de motores L298N (regulando la velocidad del motor DC con señales PWM) y el servomotor de la dirección.
- **Adquisición de Datos de Sensores:** Realiza la lectura continua del sensor ultrasónico HC-SR04 y del giroscopio/acelerómetro MPU6050.

- **Flujo de Operación:**

- **Inicialización:** Al encenderse, el Arduino inicializa los pines y espera la orden "START" desde la Raspberry Pi por el puerto serie.
- **Calibración del Giroscopio:** Una vez recibe la orden de inicio, ejecuta una rutina de calibración de 2 segundos para el MPU6050. Durante este tiempo, toma múltiples lecturas en reposo para calcular el sesgo (drift) promedio del sensor. Este valor de sesgo se resta de

las lecturas posteriores para obtener una medición de la orientación mucho más precisa.

- **Bucle Principal (Loop):** En su ciclo principal, el Arduino realiza dos tareas de forma continua:
 - **Escuchar Comandos:** Revisa constantemente el puerto serie en busca de nuevos comandos de la Raspberry Pi (ej. MOTOR:150, SERVO:45). Al recibir un comando, lo interpreta y ejecuta la acción correspondiente de inmediato.
 - **Enviar Datos:** Lee los valores actuales de los sensores (distancia y orientación) y los envía de vuelta a la Raspberry Pi a través del puerto serie, para que el cerebro principal siempre tenga información actualizada del entorno.

4.2 Código de la Raspberry Pi 5

El programa de la Raspberry Pi, escrito en Python, actúa como el cerebro de alto nivel del robot. Se enfoca en las tareas que requieren una gran capacidad de procesamiento, como la visión por computadora y la lógica de decisión.

- **Funciones Principales:**

- **Visión por Computadora:** Utiliza la librería OpenCV para procesar en tiempo real el video de la webcam, identificar contornos, filtrar por color y determinar la posición de los objetos en la pista.
- **Lógica de Navegación:** Es el responsable de la estrategia general. Decide la dirección del robot basándose en el color de la pista y de los obstáculos.
- **Comunicación y Orquestación:** Inicia la secuencia de operación enviando el comando "START" al Arduino y luego gestiona el flujo de comandos y datos durante toda la operación.

- **Flujo de Operación:**

- **Inicio y Sincronización:** Al ejecutarse, establece la conexión serie con el Arduino y le envía la señal de "START". Luego, espera a que el Arduino complete su rutina de calibración.
- **Bucle Principal:**
 - **Análisis Visual:** Captura una imagen de la cámara, la procesa para identificar el color predominante de la pista (naranja o azul) y la presencia de obstáculos.
 - **Toma de Decisiones:** Basándose en los datos visuales y en la información de los sensores que recibe del Arduino, la Pi decide la acción a tomar (avanzar, girar, detenerse).

- **Envío de Comandos:** Envía el comando apropiado al Arduino para que este lo ejecute.
- **Condición de Finalización:** El programa monitorea el valor acumulado del giroscopio. Si este supera un umbral que equivale a aproximadamente tres vueltas completas (ej. valor absoluto > 1100), envía un comando de detención final al Arduino y termina el programa.
- **Calibración de Color:** El software incluye una utilidad que permite ajustar y guardar los rangos de color (HSV), facilitando la adaptación del robot a diferentes condiciones de iluminación de forma rápida.

5 Resultados y demostración

El rendimiento del robot fue evaluado en la competencia del 22 de julio, obteniendo los siguientes puntajes como línea base para nuestro proceso de mejora continua:

<i>Categoría</i>	<i>Puntaje Obtenido</i>
<i>Movilidad</i>	3/4
<i>Potencia y Manejo de Sensores</i>	3/4
<i>Manejo de Obstáculos</i>	1/4
<i>Fotos del Equipo y Vehículo</i>	3/4
<i>Video</i>	0/4
<i>Utilización de GitHub</i>	3/4
<i>Factores de Ingeniería</i>	4/4
<i>Impresión del Juez</i>	2/2

6 Conclusiones

El desarrollo del robot Encore ha sido un proceso iterativo de resolución de problemas de ingeniería. Los avances clave desde el inicio del proyecto incluyen una drástica reducción de peso del 28% mediante el cambio de materiales, la solución a un déficit crítico de torque en la propulsión, y la implementación de una sofisticada arquitectura de doble procesador para mejorar la fiabilidad del control.

Aunque el robot ya es capaz de navegar y reconocer obstáculos de forma autónoma, el sistema aún no está optimizado para completar el desafío de las tres vueltas de manera consistente debido a la complejidad de integrar todos los sistemas en tiempo real. Los esfuerzos actuales se centran en refinar la lógica de control y la fiabilidad mecánica para la próxima competencia del 7 de agosto, donde se espera demostrar una mejora sustancial en todas las áreas evaluadas.

7 Aspectos por mejorar

El vehículo presenta varios aspectos a mejorar:

- **Latencia del Bucle de Control:** Aunque se ha mejorado la reacción, existe un retraso entre la detección de la cámara y la acción del motor. Se trabajará en optimizar el código de Python y la comunicación serie para reducir esta latencia.
- **Fiabilidad de Conexiones:** El uso de un protoboard y cables Dupont es una fuente potencial de fallos por falsos contactos. El siguiente paso lógico es el diseño de una **PCB (Placa de Circuito Impreso)** o una protoboard soldada para crear conexiones permanentes y robustas.
- **Calibración y Pruebas de Software:** La complejidad del sistema dual requiere un protocolo de pruebas más riguroso para asegurar que la comunicación entre la Pi y el Arduino sea infalible bajo condiciones de competencia.
- **Optimización de Velocidad:** Una vez que la fiabilidad esté garantizada, se explorarán maneras de aumentar la velocidad de forma segura, posiblemente ajustando los parámetros de aceleración en el código para no comprometer el control.