WORLD ROBOT OLYMPIAD™

WRO®

VENEZUELA

DIARIO DE INGENIERÍA: FUTUROS INGENIEROS ELECTROTECH

Integrantes:

Aguilar, Andrea

Contreras, Gabriel

Urdaneta, Gerardo

Caracas, julio de 2024

Diario de Ingeniería

1. Gestión de Movilidad

La categoría de “Futuros Ingenieros” presenta un desafío que consiste en diseñar un vehículo robótico capaz de conducir de manera autónoma en un circuito que cambia aleatoriamente en cada ronda de la competencia.

Para la movilidad del vehículo por la pista se seleccionaron un motor DC y un servo-motor del Kit “Nezha Inventor’s Kit for micro:bit”. Conectados al módulo Nezha y utilizados, el motor DC para la conducción, y el servo-motor para la dirección. Siendo la función del servo-motor indispensable para llevar a cabo el reto, se realizó un sistema de engranajes y tornillos sin fin que permiten girar las ruedas hacia la derecha o izquierda según lo detectado por los sensores y ejecutado por el servo-motor. El módulo Nezha incluye una batería de litio de 900mAh que permite alimentar la placa micro:bit y representa la fuente de energía para el movimiento de los motores conectados a él.

2. Gestión de Energía

El vehículo cuenta con el módulo Nezha como fuente de alimentación; éste incluye una batería de litio de 900mAh que permite alimentar la placa micro:bit y conectar todos los componentes simultáneamente. El módulo posee cuatro entradas para motores DC y servo-motores, así como para los diversos sensores que ofrece.

Para la ejecución del reto se seleccionaron 3 sensores de ultrasonido y una Cámara “Smart AI Lens” de ELECFREAKS. Los sensores de ultrasonido son programados para detectar cuerpos a distancias específicas, por lo que en el reto permiten detectar la distancia entre el vehículo y las barreras para realizar los giros por la pista o evitar chocar con ellas. Por otra parte, la cámara permite detectar objetos, rostros, números, figuras, y colores, más detalladamente, por lo que fue programada para detectar las señales de tráfico colocadas en la pista, para que el vehículo las esquive según corresponda.

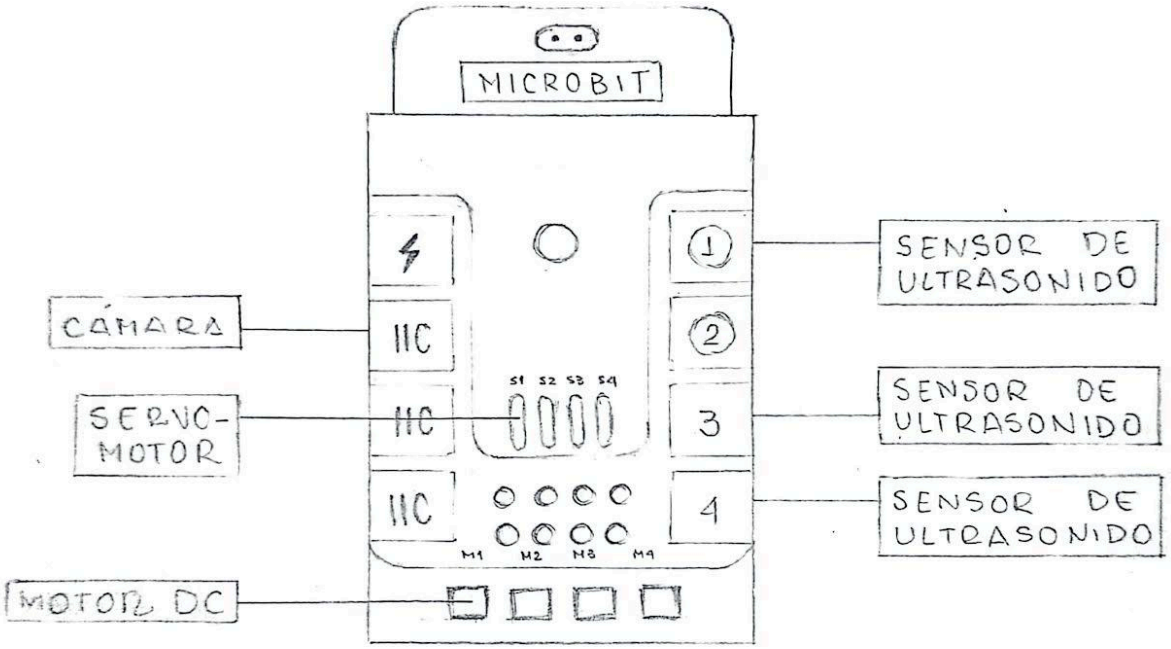

Imagen 1. Diagrama de cableado.

3. Gestión de Obstáculo

La estrategia del vehículo para superar los obstáculos consiste en la detección del color de las señales de tráfico con la cámara, para así tomar la decisión lógica de esquivarla por el lado izquierdo o derecho, ejecutando los giros mediante el servo-motor.

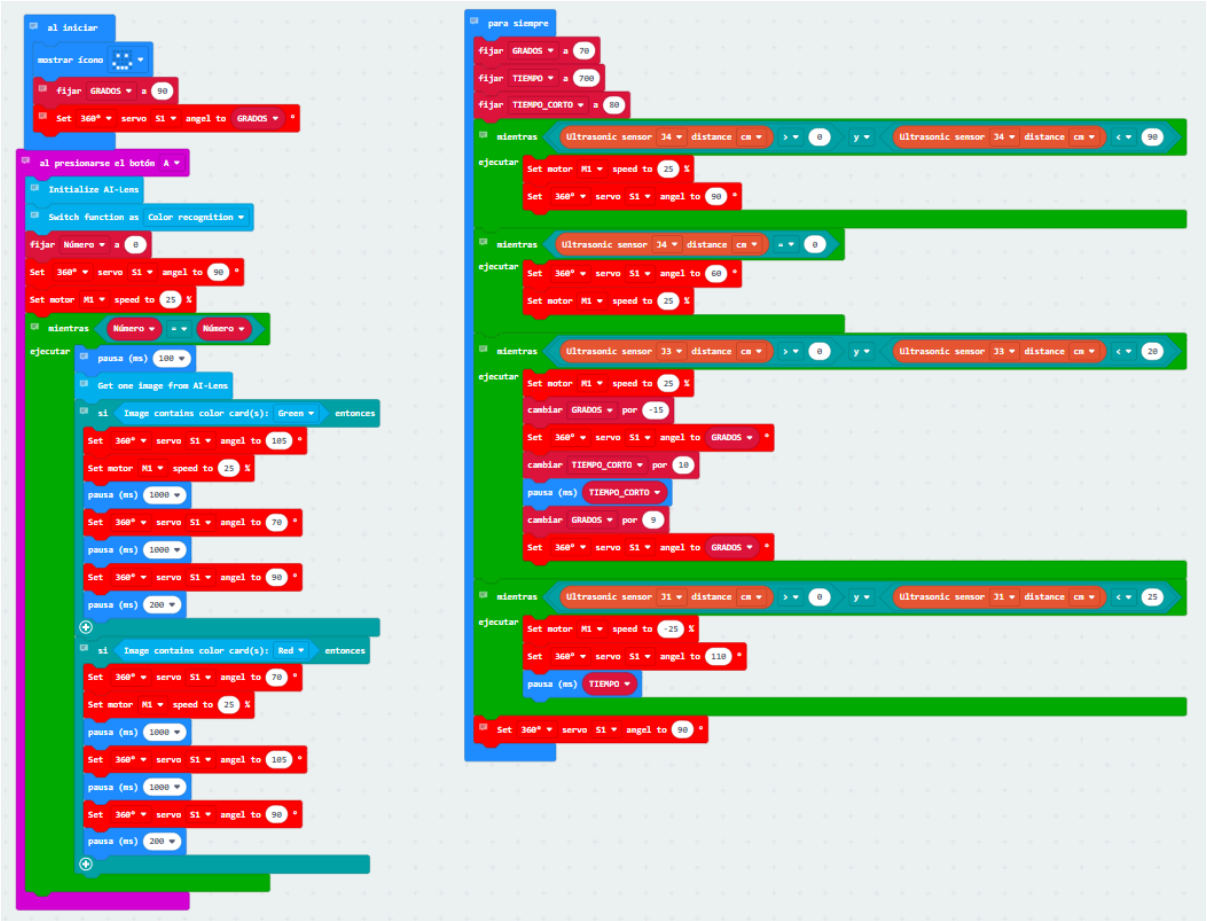

Imagen 2. Código en bloques.

Código en Python:

```
# Segmento de código para el reconocimiento de las señales de tráfico. Inicia al presionarse el botón A de la tarjeta micro:bit.
#
# Al ejecutarse, inicializa la cámara y la configura, así como determina el valor 0 para la variable "Número". Además, fija el ángulo del servo-motor a 90° y enciende el motor DC con una potencia del 25%.

def on_button_pressed_a():
    global Número
    # Inicializa la cámara para su uso.
    PlanetX_AILens.init_module()
    # Determina la función de reconocimiento de la cámara para: Detección de color.
    PlanetX_AILens.switchfunc(PlanetX_AILens.FuncList.COLOR)
    Número = 0
    neZha.set_servo_angel(neZha.ServoTypeList._360, neZha.ServoList.S1, 90)
    neZha.set_motor_speed(neZha.MotorList.M1, 25)
```

```
    # Bucle que confirma la igualdad entre las dos variables para ejecutar el
código.
    while Número == Número:
        # Realiza una pausa de 0.1 segundos antes de cada repetición.
        basic.pause(100)
        # Analiza la imagen detectada por la cámara.
        PlanetX_AILens.camera_image()
         # Indica que cuando la cámara detecte el color verde, se fijará el ángulo
del servo-motor a 105° y la potencia del motor DC a 25%. Luego de 1 segundo, fijará
el ángulo del servo-motor a 70°. Y luego de 1 segundo, fijará el ángulo a 90°; para
posteriormente realizar una pausa de 0,2 segundos.
        #
        #
         # Esto permitirá al vehículo esquivar la señal de tráfico verde por el lado
izquierdo.
        if PlanetX_AILens.color_check(PlanetX_AILens.ColorLs.GREEN):
                neZha.set_servo_angel(neZha.ServoTypeList._360, neZha.ServoList.S1,
105)
            neZha.set_motor_speed(neZha.MotorList.M1, 25)
            basic.pause(1000)
            neZha.set_servo_angel(neZha.ServoTypeList._360, neZha.ServoList.S1, 70)
            basic.pause(1000)
            neZha.set_servo_angel(neZha.ServoTypeList._360, neZha.ServoList.S1, 90)
            basic.pause(200)
          # Indica que cuando la cámara detecte el color rojo, se fijará el ángulo
del servo-motor a 70° y la potencia del motor DC a 25%. Luego de 1 segundo, fijará
el ángulo del servo-motor a 105°. Y luego de 1 segundo, fijará el ángulo a 90°;
para posteriormente realizar una pausa de 0,2 segundos.
        #
         # Esto permitirá al vehículo esquivar la señal de tráfico roja por el lado
derecho.
        if PlanetX_AILens.color_check(PlanetX_AILens.ColorLs.RED):
            neZha.set_servo_angel(neZha.ServoTypeList._360, neZha.ServoList.S1, 70)
            neZha.set_motor_speed(neZha.MotorList.M1, 25)
            basic.pause(1000)
                neZha.set_servo_angel(neZha.ServoTypeList._360, neZha.ServoList.S1,
105)
            basic.pause(1000)
            neZha.set_servo_angel(neZha.ServoTypeList._360, neZha.ServoList.S1, 90)
            basic.pause(200)
input.on_button_pressed(Button.A, on_button_pressed_a)

# Acciones que se ejecutan al encender el vehículo. Comienza mostrando una imagen
en la tarjeta micro:bit, que permite identificar el programa.
TIEMPO_CORTO = 0
TIEMPO = 0
Número = 0
basic.show_icon(IconNames.HAPPY)
# Se fija el valor de la variable "GRADOS" a 90. Para que al iniciar el programa,
el robot verifique el ángulo en que se encuentra el servo-motor, y lo coloque a
90°.
```

```
GRADOS = 90
# Coloca el servo-motor a los grados que indica la variable.
neZha.set_servo_angel(neZha.ServoTypeList._360, neZha.ServoList.S1, GRADOS)

# Acciones que se repiten en bucle desde que se enciende el vehículo. Inicialmente se declaran las variables "GRADOS", "TIEMPO", y "TIEMPO_CORTO"

def on_forever():
    global GRADOS, TIEMPO, TIEMPO_CORTO
    GRADOS = 70
    TIEMPO = 700
    TIEMPO_CORTO = 80
    # Indica que mientras el sensor de ultrasonido conectado al puerto J4 detecte una distancia mayor de 0cm y menor de 90cm, encienda el motor DC a una potencia de 25% y fije el ángulo del servo-motor a 90°.
    while PlanetX_Basic.ultrasound_sensor(PlanetX_Basic.DigitalRJPin.J4,
                    PlanetX_Basic.Distance_Unit_List.DISTANCE_UNIT_CM) > 0 and PlanetX_Basic.ultrasound_sensor(PlanetX_Basic.DigitalRJPin.J4,
        PlanetX_Basic.Distance_Unit_List.DISTANCE_UNIT_CM) < 90:
        neZha.set_motor_speed(neZha.MotorList.M1, 25)
        neZha.set_servo_angel(neZha.ServoTypeList._360, neZha.ServoList.S1, 90)
    # Indica que mientras el sensor de ultrasonido conectado al puerto J4 detecte una distancia igual a 0cm, encienda el motor DC a una potencia de 25% y fije el ángulo del servo-motor a 60°. Esto le permite realizar los giros en la pista.
    #
    # El vehículo detecta la distancia de la barrera interna de la pista y se mantiene conduciendo de forma recta. Una vez que ya no la detecta, comienza a girar hasta detectarla nuevamente.
    while PlanetX_Basic.ultrasound_sensor(PlanetX_Basic.DigitalRJPin.J4,
        PlanetX_Basic.Distance_Unit_List.DISTANCE_UNIT_CM) == 0:
        neZha.set_servo_angel(neZha.ServoTypeList._360, neZha.ServoList.S1, 60)
        neZha.set_motor_speed(neZha.MotorList.M1, 25)
    # Indica que mientras el sensor de ultrasonido conectado al puerto J3 detecte una distancia mayor de 0cm y menor de 20cm, encienda el motor DC a una potencia de 25% y reste 15° al ángulo del servo-motor. Luego de 0,08 segundos, añade 9° al ángulo del vehículo.
    #
    # Esta función permite que cuando el vehículo detecte una barrera a menos de 20cm por su lado izquierdo, no choque con ella.
    while PlanetX_Basic.ultrasound_sensor(PlanetX_Basic.DigitalRJPin.J3,
                    PlanetX_Basic.Distance_Unit_List.DISTANCE_UNIT_CM) > 0 and PlanetX_Basic.ultrasound_sensor(PlanetX_Basic.DigitalRJPin.J3,
        PlanetX_Basic.Distance_Unit_List.DISTANCE_UNIT_CM) < 20:
        neZha.set_motor_speed(neZha.MotorList.M1, 25)
        GRADOS += -15
        neZha.set_servo_angel(neZha.ServoTypeList._360, neZha.ServoList.S1, GRADOS)
        TIEMPO_CORTO += 10
        basic.pause(TIEMPO_CORTO)
        GRADOS += 9
        neZha.set_servo_angel(neZha.ServoTypeList._360, neZha.ServoList.S1, GRADOS)
```

```
    # Indica que mientras el sensor de ultrasonido conectado al puerto J1 detecte una distancia mayor de 0cm y menor de 25cm, que encienda el motor DC a una potencia de -25% para que el vehículo retroceda. Y fije el ángulo del servo-motor a 110°, por un tiempo de 0,7 segundos.
    #
    # De esta manera, si el vehículo se dirige de frente hacia una barrera, es capaz de retroceder y rectificar su trayectoria.
    while PlanetX_Basic.ultrasound_sensor(PlanetX_Basic.DigitalRJPin.J1,
                    PlanetX_Basic.Distance_Unit_List.DISTANCE_UNIT_CM)  >  0  and PlanetX_Basic.ultrasound_sensor(PlanetX_Basic.DigitalRJPin.J1,
        PlanetX_Basic.Distance_Unit_List.DISTANCE_UNIT_CM) < 25:
        neZha.set_motor_speed(neZha.MotorList.M1, -25)
        neZha.set_servo_angel(neZha.ServoTypeList._360, neZha.ServoList.S1, 110)
        basic.pause(TIEMPO)
    # Indica que el ángulo del servo-motor sea fijado a 90°.
    neZha.set_servo_angel(neZha.ServoTypeList._360, neZha.ServoList.S1, 90)
basic.forever(on_forever)
```

Código en JavaScript:

```
//  Segmento de código para el reconocimiento de las señales de tráfico. Inicia al presionarse el botón A de la tarjeta micro:bit.
//
//  Al ejecutarse, inicializa la cámara y la configura, así como determina el valor 0 para la variable "Número". Además, fija el ángulo del servo-motor a 90° y enciende el motor DC con una potencia del 25%.
input.onButtonPressed(Button.A, function on_button_pressed_a() {

    //  Inicializa la cámara para su uso.
    PlanetX_AILens.initModule()
    //  Determina la función de reconocimiento de la cámara para: Detección de color.
    PlanetX_AILens.switchfunc(PlanetX_AILens.FuncList.Color)
    Número = 0
    neZha.setServoAngel(neZha.ServoTypeList._360, neZha.ServoList.S1, 90)
    neZha.setMotorSpeed(neZha.MotorList.M1, 25)
    //  Bucle que confirma la igualdad entre las dos variables para ejecutar el código.
    while (Número == Número) {
        //  Realiza una pausa de 0.1 segundos antes de cada repetición.
        basic.pause(100)
        //  Analiza la imagen detectada por la cámara.
        PlanetX_AILens.cameraImage()
        //  Indica que cuando la cámara detecte el color verde, se fijará el ángulo del servo-motor a 105° y la potencia del motor DC a 25%. Luego de 1 segundo, fijará el ángulo del servo-motor a 70°. Y luego de 1 segundo, fijará el ángulo a 90°; para posteriormente realizar una pausa de 0,2 segundos.
```

```
        //
        //
          //  Esto permitirá al vehículo esquivar la señal de tráfico verde por el lado izquierdo.
        if (PlanetX_AILens.colorCheck(PlanetX_AILens.ColorLs.green)) {
            neZha.setServoAngel(neZha.ServoTypeList._360, neZha.ServoList.S1, 105)
            neZha.setMotorSpeed(neZha.MotorList.M1, 25)
            basic.pause(1000)
            neZha.setServoAngel(neZha.ServoTypeList._360, neZha.ServoList.S1, 70)
            basic.pause(1000)
            neZha.setServoAngel(neZha.ServoTypeList._360, neZha.ServoList.S1, 90)
            basic.pause(200)
        }

         //  Indica que cuando la cámara detecte el color rojo, se fijará el ángulo del servo-motor a 70° y la potencia del motor DC a 25%. Luego de 1 segundo, fijará el ángulo del servo-motor a 105°. Y luego de 1 segundo, fijará el ángulo a 90°; para posteriormente realizar una pausa de 0,2 segundos.
        //
          //  Esto permitirá al vehículo esquivar la señal de tráfico roja por el lado derecho.
        if (PlanetX_AILens.colorCheck(PlanetX_AILens.ColorLs.red)) {
            neZha.setServoAngel(neZha.ServoTypeList._360, neZha.ServoList.S1, 70)
            neZha.setMotorSpeed(neZha.MotorList.M1, 25)
            basic.pause(1000)
            neZha.setServoAngel(neZha.ServoTypeList._360, neZha.ServoList.S1, 105)
            basic.pause(1000)
            neZha.setServoAngel(neZha.ServoTypeList._360, neZha.ServoList.S1, 90)
            basic.pause(200)
        }

    }
})
//  Acciones que se ejecutan al encender el vehículo. Comienza mostrando una imagen en la tarjeta micro:bit, que permite identificar el programa.
let TIEMPO_CORTO = 0
let TIEMPO = 0
let Número = 0
basic.showIcon(IconNames.Happy)
//  Se fija el valor de la variable "GRADOS" a 90. Para que al iniciar el programa, el robot verifique el ángulo en que se encuentra el servo-motor, y lo coloque a 90°.
let GRADOS = 90
//  Coloca el servo-motor a los grados que indica la variable.
neZha.setServoAngel(neZha.ServoTypeList._360, neZha.ServoList.S1, GRADOS)
//   Acciones que se repiten en bucle desde que se enciende el vehículo. Inicialmente se declaran las variables "GRADOS", "TIEMPO", y "TIEMPO_CORTO"
basic.forever(function on_forever() {

    GRADOS = 70
    TIEMPO = 700
```

```
    TIEMPO_CORTO = 80
    //  Indica que mientras el sensor de ultrasonido conectado al puerto J4 detecte una distancia mayor de 0cm y menor de 90cm, encienda el motor DC a una potencia de 25% y fije el ángulo del servo-motor a 90°.
            while (PlanetX_Basic.ultrasoundSensor(PlanetX_Basic.DigitalRJPin.J4, PlanetX_Basic.Distance_Unit_List.Distance_Unit_cm) > 0 && PlanetX_Basic.ultrasoundSensor(PlanetX_Basic.DigitalRJPin.J4, PlanetX_Basic.Distance_Unit_List.Distance_Unit_cm) < 90) {
        neZha.setMotorSpeed(neZha.MotorList.M1, 25)
        neZha.setServoAngel(neZha.ServoTypeList._360, neZha.ServoList.S1, 90)
    }
    //  Indica que mientras el sensor de ultrasonido conectado al puerto J4 detecte una distancia igual a 0cm, encienda el motor DC a una potencia de 25% y fije el ángulo del servo-motor a 60°. Esto le permite realizar los giros en la pista.
    //
     //  El vehículo detecta la distancia de la barrera interna de la pista y se mantiene conduciendo de forma recta. Una vez que ya no la detecta, comienza a girar hasta detectarla nuevamente.
            while (PlanetX_Basic.ultrasoundSensor(PlanetX_Basic.DigitalRJPin.J4, PlanetX_Basic.Distance_Unit_List.Distance_Unit_cm) == 0) {
        neZha.setServoAngel(neZha.ServoTypeList._360, neZha.ServoList.S1, 60)
        neZha.setMotorSpeed(neZha.MotorList.M1, 25)
    }
    //  Indica que mientras el sensor de ultrasonido conectado al puerto J3 detecte una distancia mayor de 0cm y menor de 20cm, encienda el motor DC a una potencia de 25% y reste 15° al ángulo del servo-motor. Luego de 0,08 segundos, añade 9° al ángulo del vehículo.
    //
     //  Esta función permite que cuando el vehículo detecte una barrera a menos de 20cm por su lado izquierdo, no choque con ella.
            while (PlanetX_Basic.ultrasoundSensor(PlanetX_Basic.DigitalRJPin.J3, PlanetX_Basic.Distance_Unit_List.Distance_Unit_cm) > 0 && PlanetX_Basic.ultrasoundSensor(PlanetX_Basic.DigitalRJPin.J3, PlanetX_Basic.Distance_Unit_List.Distance_Unit_cm) < 20) {
        neZha.setMotorSpeed(neZha.MotorList.M1, 25)
        GRADOS += -15
        neZha.setServoAngel(neZha.ServoTypeList._360, neZha.ServoList.S1, GRADOS)
        TIEMPO_CORTO += 10
        basic.pause(TIEMPO_CORTO)
        GRADOS += 9
        neZha.setServoAngel(neZha.ServoTypeList._360, neZha.ServoList.S1, GRADOS)
    }
    //  Indica que mientras el sensor de ultrasonido conectado al puerto J1 detecte una distancia mayor de 0cm y menor de 25cm, que encienda el motor DC a una potencia de -25% para que el vehículo retroceda. Y fije el ángulo del servo-motor a 110°, por un tiempo de 0,7 segundos.
    //
     //  De esta manera, si el vehículo se dirige de frente hacia una barrera, es capaz de retroceder y rectificar su trayectoria.
            while (PlanetX_Basic.ultrasoundSensor(PlanetX_Basic.DigitalRJPin.J1, PlanetX_Basic.Distance_Unit_List.Distance_Unit_cm) > 0 &&
```

```
PlanetX_Basic.ultrasoundSensor(PlanetX_Basic.DigitalRJPin.J1,
PlanetX_Basic.Distance_Unit_List.Distance_Unit_cm) < 25) {
        neZha.setMotorSpeed(neZha.MotorList.M1, -25)
        neZha.setServoAngel(neZha.ServoTypeList._360, neZha.ServoList.S1, 110)
        basic.pause(TIEMPO)
    }
    //  Indica que el ángulo del servo-motor sea fijado a 90°.
    neZha.setServoAngel(neZha.ServoTypeList._360, neZha.ServoList.S1, 90)
})
```

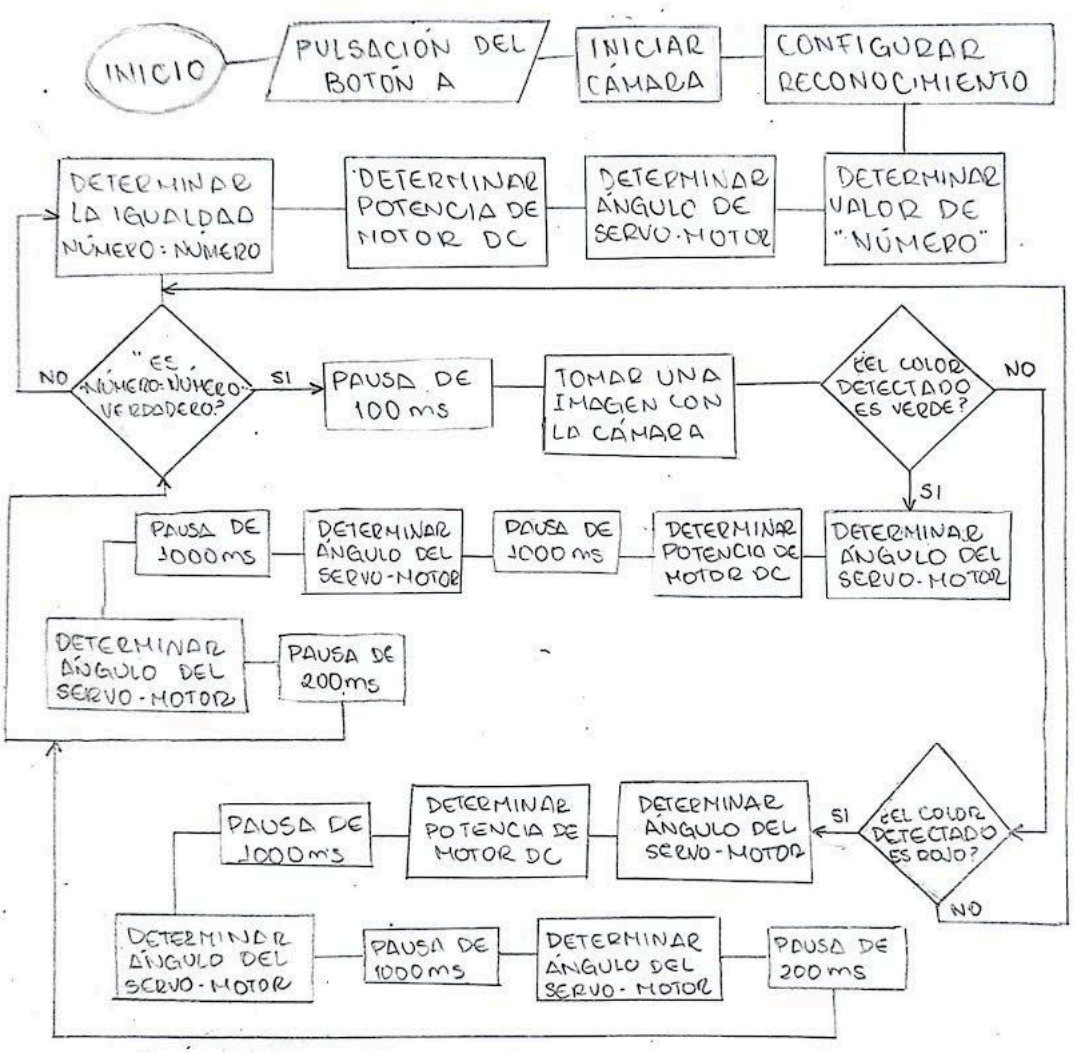

Imagen 3. Diagrama de flujo.

4. Fotos - Equipo y Vehículo

Imagen 4. Foto del equipo con el vehículo.

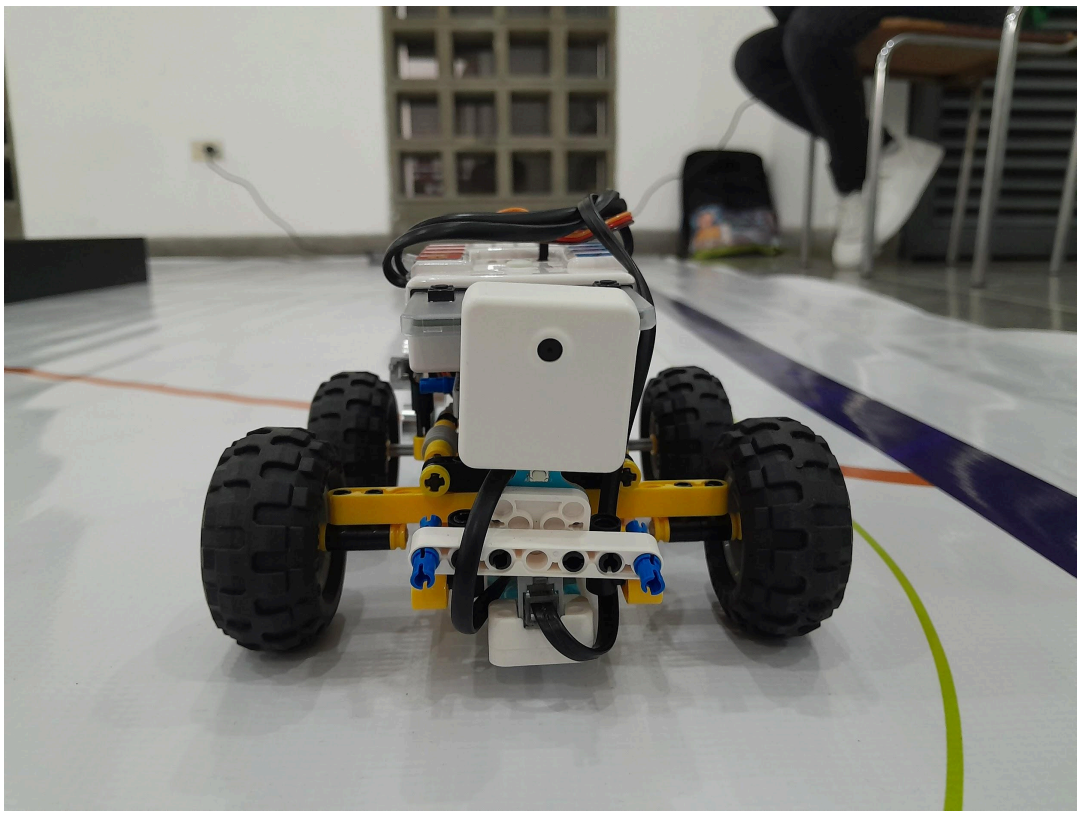

Imagen 5. Imagen frontal del vehículo.

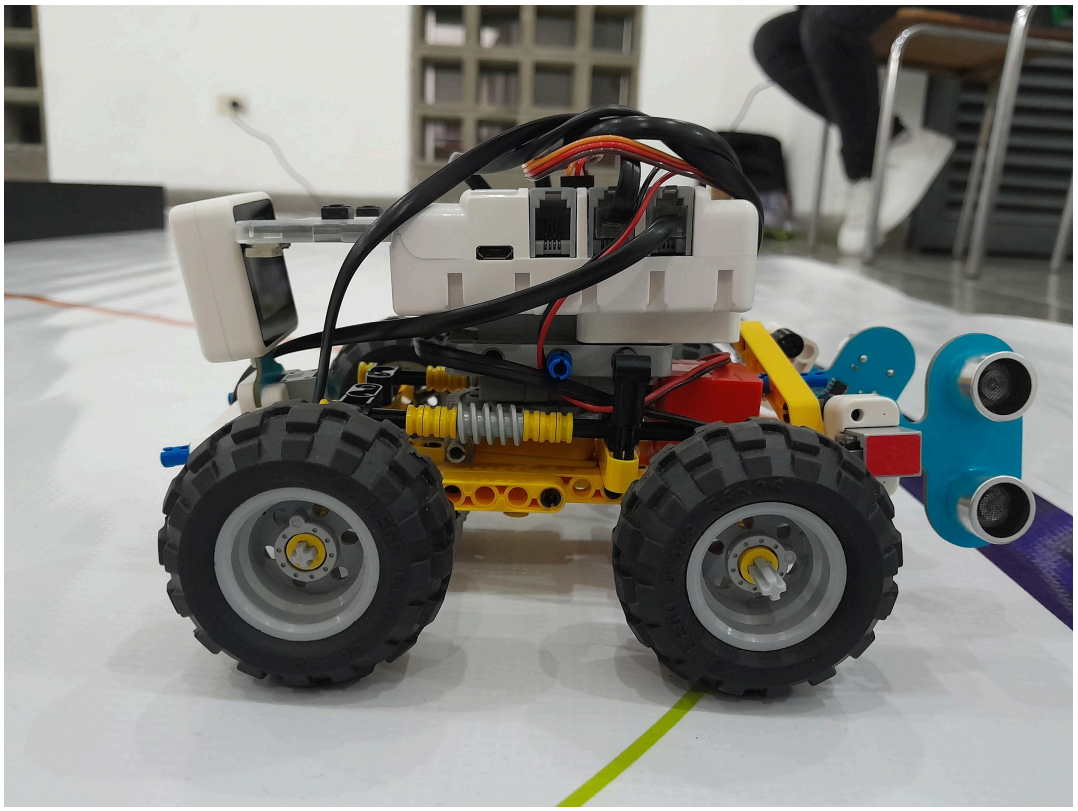

Imagen 6. Imagen del lateral izquierdo del vehículo.

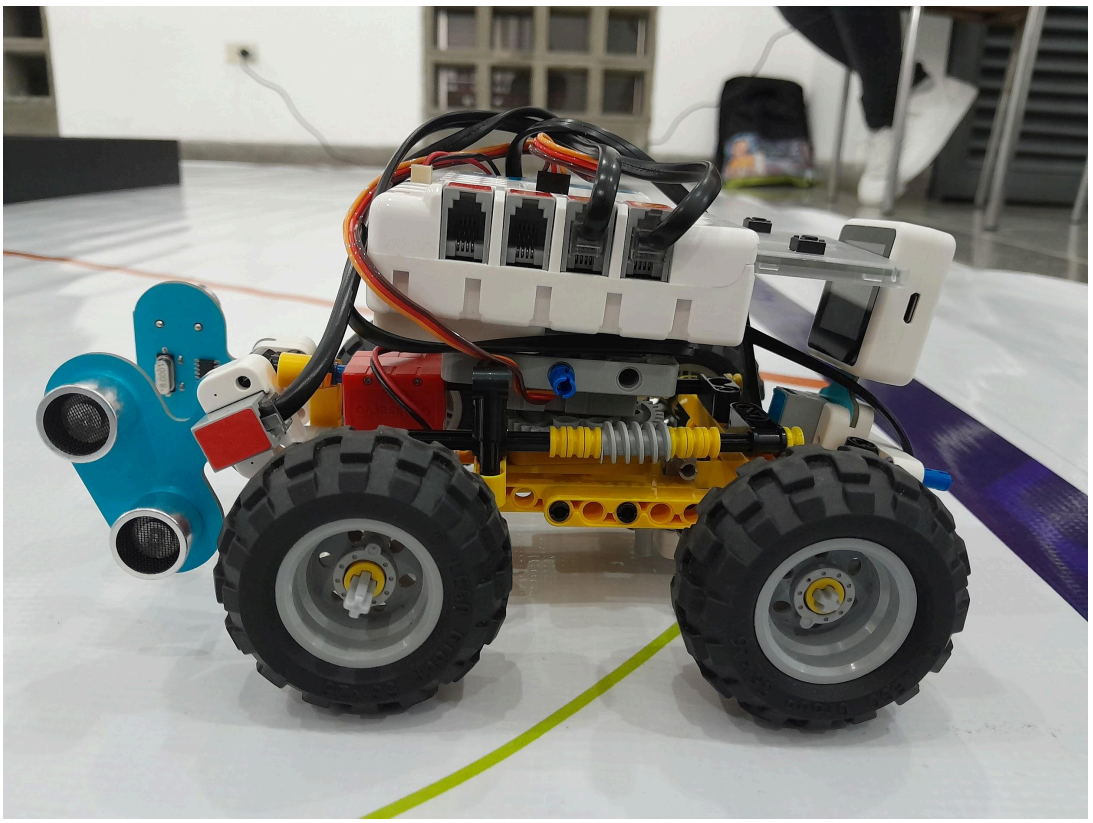

Imagen 7. Imagen del lateral derecho del vehículo

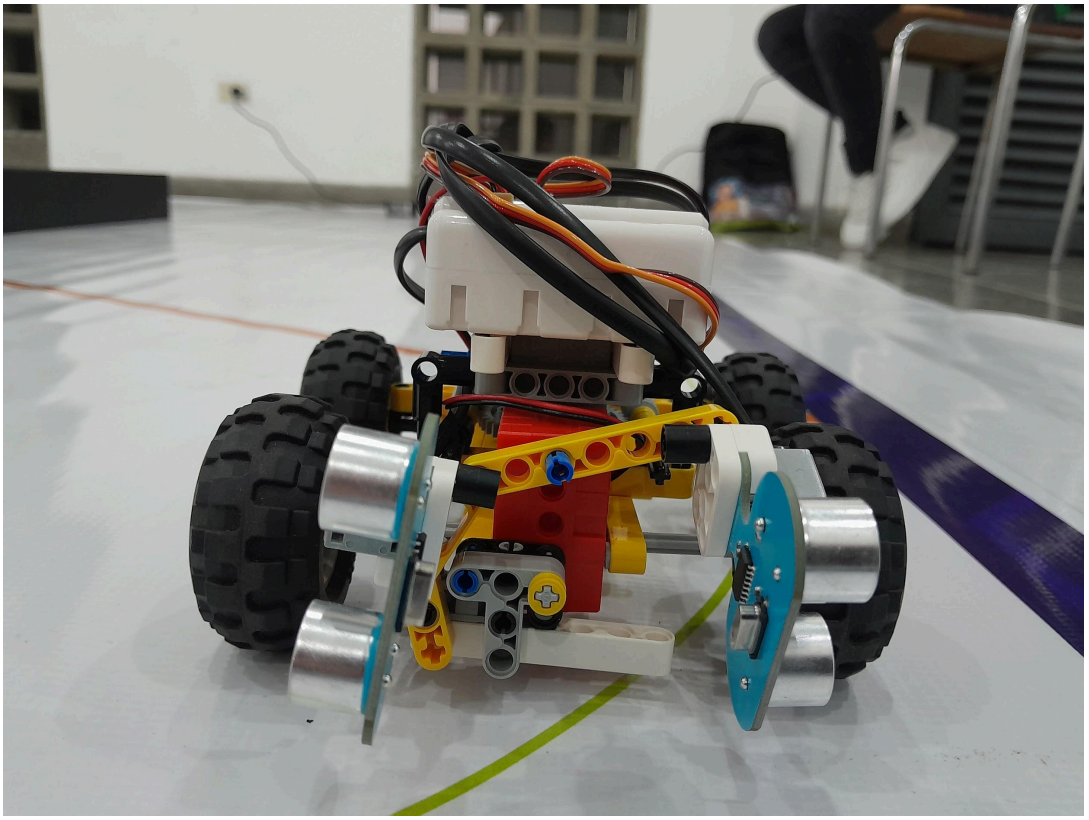

Imagen 8. Imagen de la parte trasera del vehículo

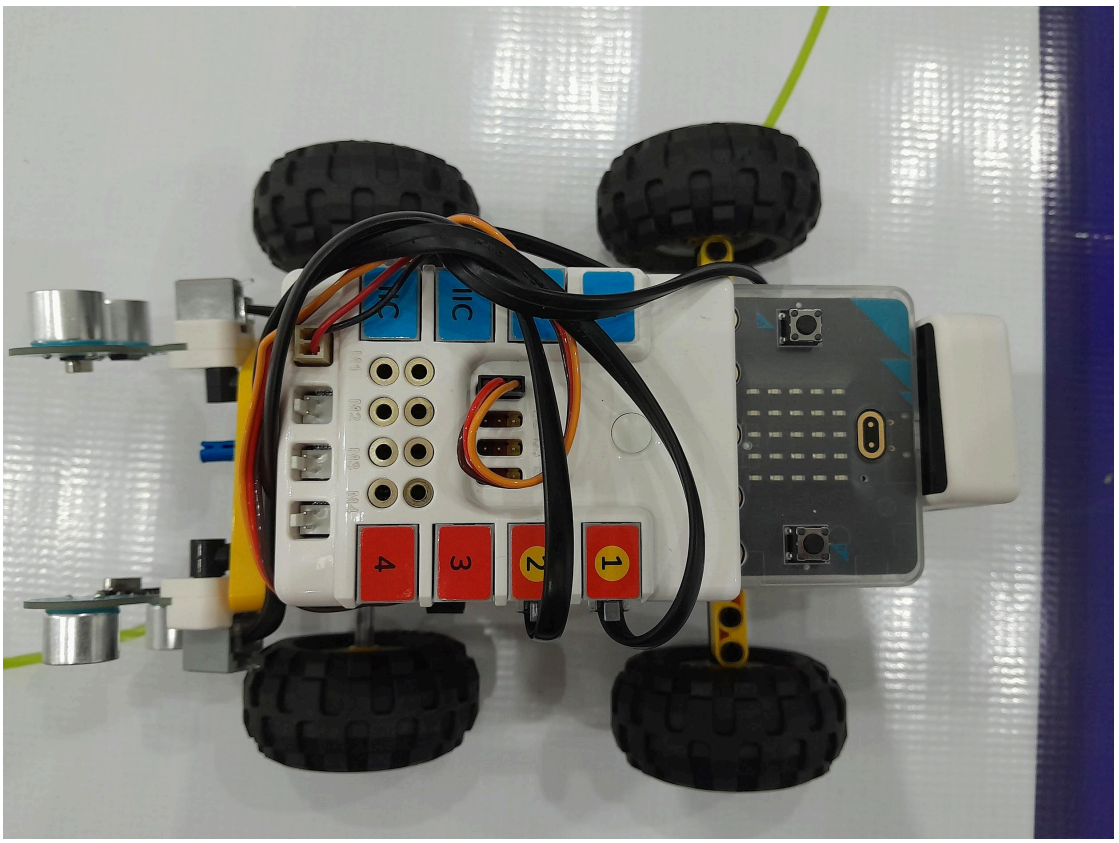

Imagen 9. Imagen de la parte superior del vehículo.

5. Videos de Rendimiento

Se realizó un video para demostrar el rendimiento del vehículo durante el desplazamiento en pista. Puede accederse a él a través del enlace: https://youtu.be/ANmnAtssnoo?si=y03vp8YUODDqMHdq

6. Utilización de GitHub

Se utilizó la plataforma GitHub para la gestión del proyecto y el control de versiones. Mediante los commits se registró con especificaciones los cambios realizados al código durante la programación del reto.

Puede accederse al repositorio del equipo ElectroTech a través del link: https://github.com/apguilar/ElectroTech-WRO-2024.git

Bitácora de construcción y programación

- 17 de junio de 2024

Se empieza el análisis y construcción del vehículo utilizando piezas del Kit “Nezha Inventor’s Kit for micro:bit”, comenzando por el mecanismo para ensamblar los motores a las ruedas.

- 18 de junio de 2024

Se diseña un mecanismo para las ruedas de dirección y el servo-motor, y se acoplan al robot. A partir de ahí se comienza a ensamblar el resto de las partes, añadiendo el motor DC para la conducción y sus ruedas. Se identificaron los sensores a utilizar y se ensamblaron al vehículo.

- 20 de junio de 2024

Se da inicio a la programación del vehículo en el programa MakeCode de Microsoft para micro:bit, para ello se instalan las extensiones de NeZha y Planet X con sus complementos.

A su vez, se cambian las ruedas utilizadas inicialmente en el vehículo por un inconveniente de funcionamiento.

- 21 de junio de 2024

Se realizan pruebas de funcionamiento en la pista al vehículo con el fin de calibrar y comprender el funcionamiento de los sensores. Se inicia la programación de los giros del vehículo utilizando un sensor de color para detectar las líneas de colores en las esquinas de la pista.

- 25 de junio de 2024

Se desarrollan 2 alternativas de código para realizar el circuito en sentido horario y antihorario, debido al patrón de líneas naranjas y azules con el que cuenta la pista,

dependiendo de la sucesividad de los colores. El resultado no fue el esperado, dado que el sensor de color no detectaba las líneas con presición.

- 26 de junio de 2024

El sensor de color fallaba al detectar el color naranja, pues ejecutaba las acciones programadas para la detección del color azul, y luego del giro las ruedas no regresaban a su posición inicial. Tras esto, se comenzó a estructurar el código de la cámara para la detección de las señales de tráfico.

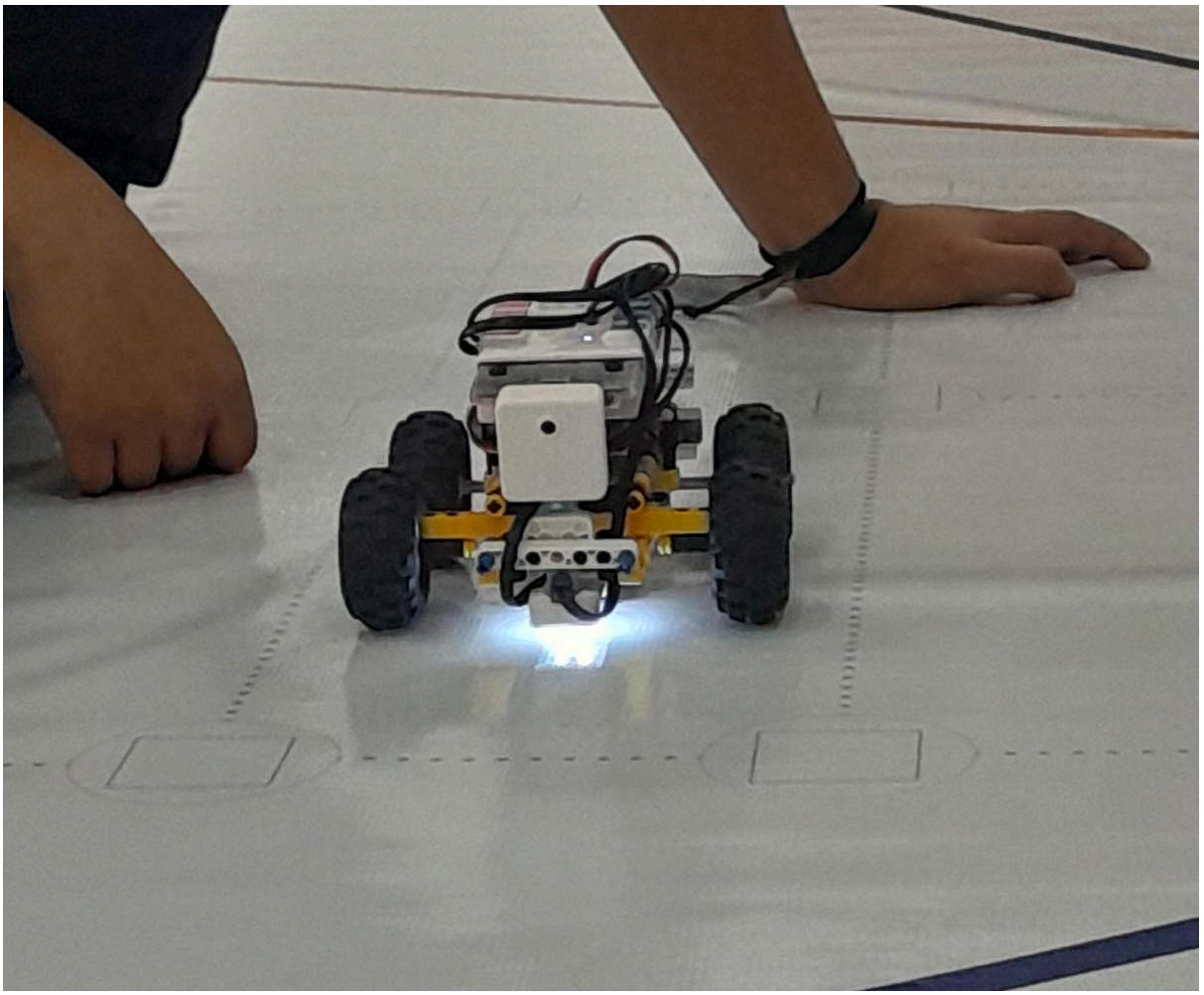

Imagen 10. Prueba del vehículo con el sensor de color.

- 27 de junio de 2024

Se continúa el desarrollo del código para la camara y el sensor de color. Por ineficacia se optó por abandonar el sensor de color y optimizar la programación de los sensores de ultrasonido. Se añade un nuevo sensor de ultrasonido y se coloca en la parte frontal del vehículo para evitar choques durante el desplazamiento.

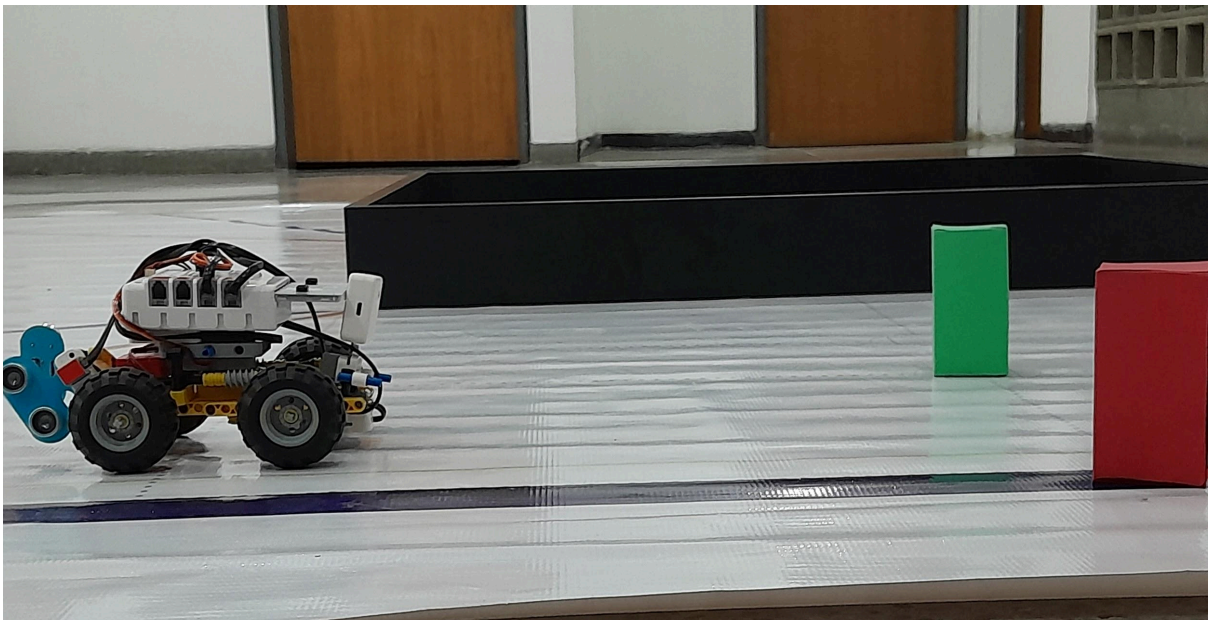

Imagen 11. Pruebas de la cámara para detección de señales de tráfico.

- 28 de junio de 2024

Se optimiza la programación de los sensores de ultrasonido para que el vehículo se desplace de forma continua por la pista.