TEAM PARADUCKS

FUTURE ENGINEERS 2024

WHO ARE WE?

Team Name:
Paraducks

Category:
Future Engineers

School:
Grade 10, Dhirubhai Ambani International School, Mumbai

Members:
Aryan Pai
Manish Ganeshan
Micquel Devlaliwalla

CONTENTS

MOBILITY MANAGEMENT

FUTURE ENGINEERS 2024

MOTORS

To adhere to the stipulated limitations of **1 DC motor** and **1 servo motor** for our robot, we decided the following:

- **Rear-Wheel Drive:** We connected our DC motor to a bevel gear to power the entire rear axle with the singular motor, powered by the motor driver in turn directly sourcing power from the battery.
- **Servo Steering:** We used a servo to turn the front axle to control robot steering, the wheels remaining uncontrolled to move with the rear axle. The robot dimensions allow a maximum turning range of 180°. The axle is a disjoint shaft to avoid wheels slipping,

12V brushed DC motor: Affordable, simple, reliable, power efficient, consistent torque and speed, adequate to move entire robot at moderate speeds.

Differential gearbox: Allows wheels to rotate at different speeds

Cytron MD10C R3 Motor Driver: High continuous current supply.

RKI-1203 digital servo: Adequate torque for robot weight and accurate, fast turning, space and power efficient.

ENGINEERING FACTOR

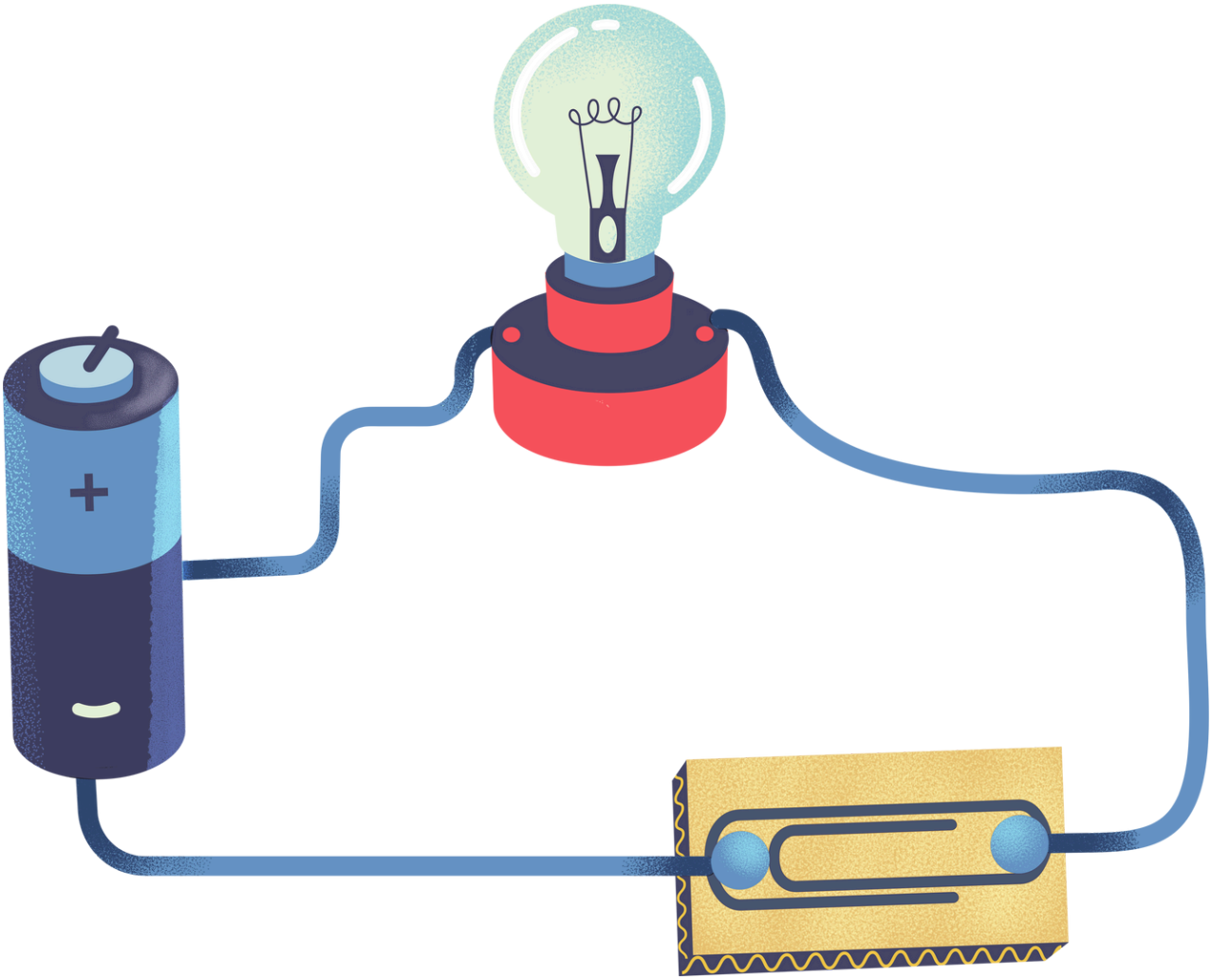
The design needed to be agile and have a very small radius of turn. This was achieved by using a **smaller wheelbase** and a stacked **CNC made Polycarbonate sheets design** to accommodate displaced electronic components.

A **metal differential** is used to couple the rear disjoint shaft together, for better power delivery to both the rear wheels. This leads to close to **no slippage** while turning. The motor is mounted vertically with the shaft facing down.

The **stability** of the bot was another priority, taken care by **very low CG** due to the battery being placed below the deck.

The **retractable link** at the front of the bot gives us a considerably **larger mechanical footprint** than the actual size of the bot, which helps us with a **larger allotted parking area** for the robot. This link retracts when the time period starts even with a small flick of the Servo.

The camera is mounted on a very high position which gives us a **very large POV** for detection of blocks.



POWER AND SENSE MANAGEMENT

FUTURE ENGINEERS 2024

POWER

Power Supply: Range LiPo 11.1 V 1300mah 3 cell battery

The **motor driver** is directly connected and powered by the battery, which in turn powers the motor.

We use a 12V -> 5V buck converter module to step down voltage which is distributed by our custom PCB to all sensors and the following methods for the remaining components:

1. **Raspberry Pi 4B:** Connected and powered by the USB-A port in the buck converter to its USB-C power port
2. **Arduino MEGA:** Connected and powered from the RPi's USB-A port to its USB-B power port. Same connection used for serial communication between the two.

MICROCONTROLLERS

We initially began prototyping with an Arduino UNO, but found many flaws in it, primarily its image processing with a HuskyLens. We now use:

1. **Raspberry Pi 4B:** Primary microcontroller, where programs are stored and programmed in Python. Far better image processing and Arducam camera compatibility, GPIO ports are easy to use, display also attaches easier. Problems with serial communication.
2. **Arduino MEGA 2560 Rev3:** Secondary microcontroller. Used for its superior compatibility and libraries for serial communication with sensors: colour sensor and IMU.

Note: We also used an ESP32 DevKit for a while, but compatibility and connectivity issues forced us to replace it with the MEGA.

SENSORS

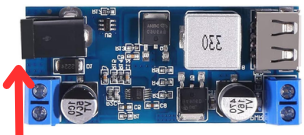
Name	Description	Use
Adafruit BNO085 9-DoF IMU	Has a gyroscope, accelerometer, magnetometer	Tracking heading for turning
TFMini Plus LiDAR distance sensor	Power and cost efficient, long 350m range	Finding distance in front and sides, wall avoidance
Adafruit TCS34725 RGB Color Sensor	Mounted under robot, has a white-yellow LED and IR filter	Detecting blue and orange corner ground stripes
Quadrature Encoder	4680 ticks per revolution, mounted on top of motor	Specific movement cases
Arducam IMX708 Camera Sensor	Uses Sony 12MP Autofocus sensor shooting HD color video	Detection of traffic signs and parking lot limitations

CIRCUIT DIAGRAM

Battery

Switch

12V



Buck Converter

12V

5V

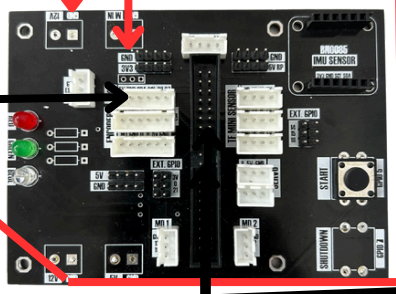
Custom PCB

12V

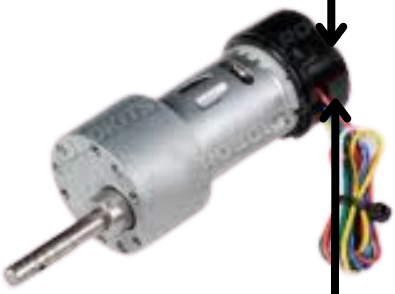
5V

Motor Driver

12V



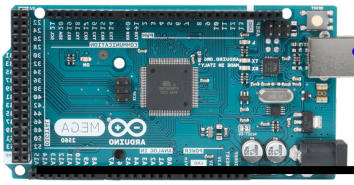
Raspberry Pi 4B



DC Motor

Servo

Camera



Arduino MEGA

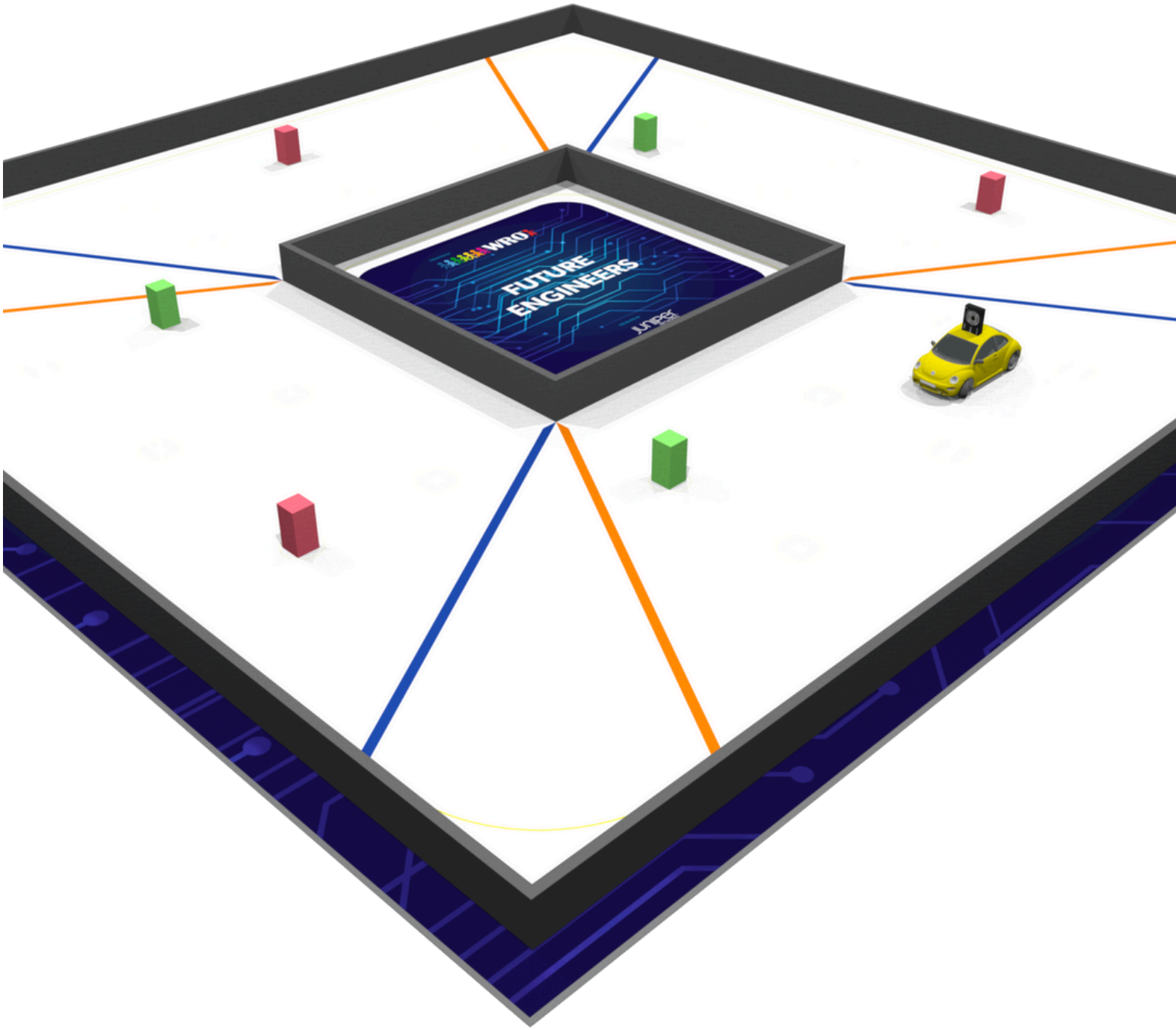
IMU

Distance sensor

Key:

Red: Power
Black: Data
Orange: USB-C
Blue: USB-B

FUTURE ENGINEERS 2024



OBSTACLE MANAGEMENT

FUTURE ENGINEERS 2024

PROGRAMMING

Key Functions:

TFMini Data: Calculates **checksum** of 9-bit data packages sent by TFMini, accepts value if correct and converts to cm.

Set Angle: Uses PWM **(pulse-width modulation)** to set angle, converts degree input to pulse-width

Button Toggle: Compares previous and current state of the button **(on/off)** to see if it is pressed to start or stop the robot.

Running Encoder: Establishes **UART** communication with **Arduino MEGA;** reads, decodes and strips lines of data received

PID controller: Used to decrease error between desired and servo position

Programming Language:

Python

Image Processing:

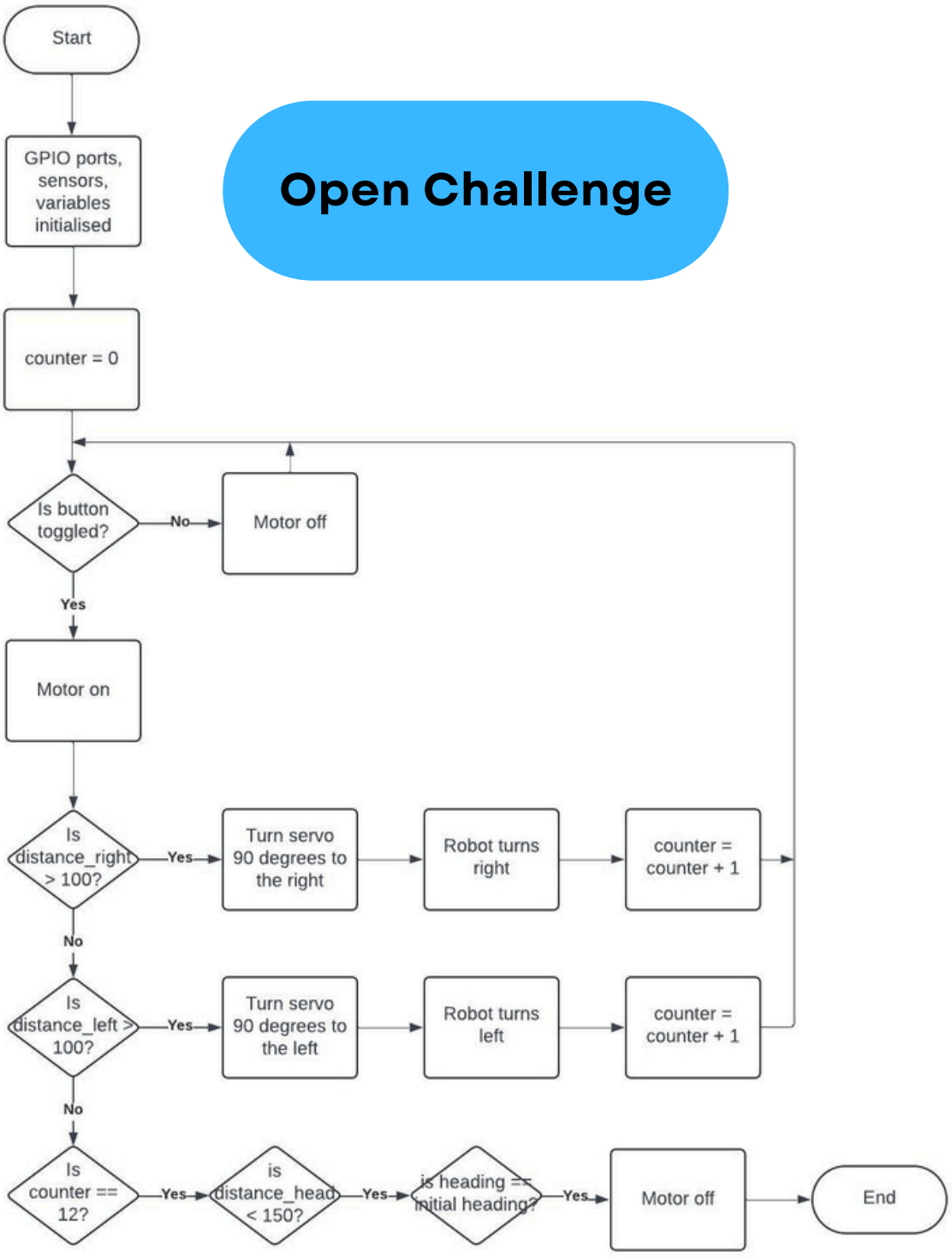
Establishes camera feed in **OpenCV**
Creates and denoises masks within **HSV colour bounds**
Finds and draws contours of same colour on frame
Checks for **white surrounding** detected object to prove its presence in the field
Draws a rectangle around largest area of contours (detected objects)

Counter

Increments after every corner to identify lap and lane

FLOWCHART

Open Challenge



FUTURE ENGINEERS 2024

FLOWCHART

Obstacle Challenge

FUTURE ENGINEERS 2024

PSEUDOCODE

Open Challenge

Setup
Set up GPIO pins:
Set pin 14 as output
Set pin 5 as input with pull-up resistor
Reset system:
Output low on pin 14 for 1 second
Output high on pin 14 for 1 second

getTFminiData()
Sleep for 0.01 seconds
Read data from serial connections:
RX_Head
RX_Left
RX_Right
Parse data and calculate distances:
distance_head
distance_left
distance_right

correctAngle()
Calculate error between current heading and set point
Calculate correction using PID algorithm
Apply correction to servo

servoDrive(distance, block, pwm, counts, head)
Initialize servo and serial connections
Set up PWM frequency and duty cycle
Read button state and toggle motor power
Read encoder data and update heading angle
Correct angle using correctAngle() function
Update motor power and direction based on heading angle and distance sensors

runEncoder(counts, head)
Read encoder data from serial connection
Update encoder count and heading angle

Main Loop
Create multiprocessing objects for servo drive and encoder processes
Start servo drive and encoder processes
Run until interrupted

Cleanup
Stop servo drive and encoder processes
Close serial connections and GPIO pins
Clean up IMU and GPIO resources

PSEUDOCODE

Obstacle Challenge

Setup
Set up GPIO pins:
Set pin 14 as output
Set pin 5 as input with pull-up resistor
Reset system:
Output low on pin 14 for 1 second
Output high on pin 14 for 1 second

Live_Feed(): Image Processing (defined above)

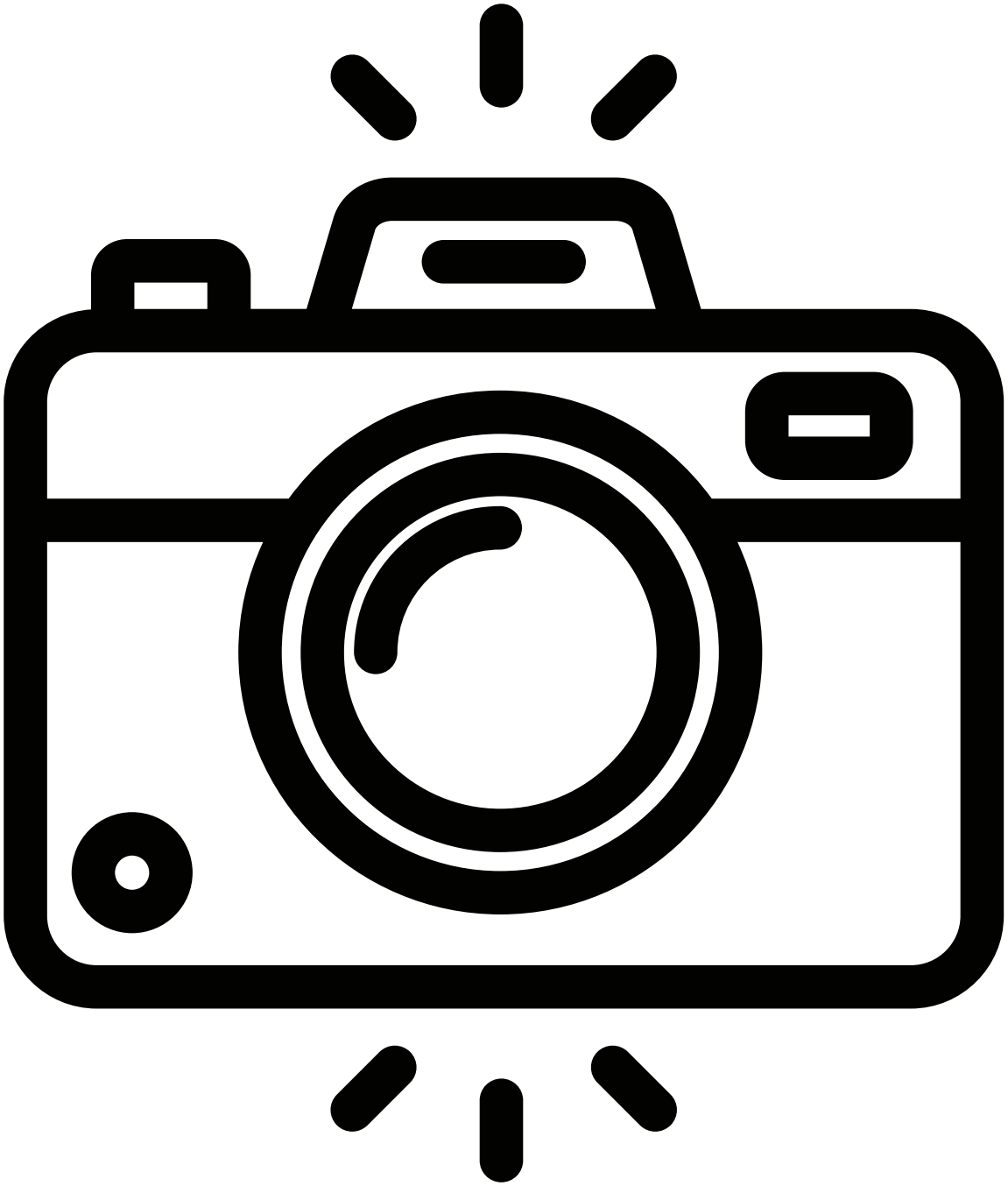
correctPosition():
Calculate error and correction for position based on color detection and distance sensors
Update set point based on lane detection and color detection
Apply corrections for pink wall detection

correctAngle():
Same as Open Challenge

servoDrive():
Initialize servo and serial connections
Set up PWM frequency and duty cycle
Read button state and toggle motor power
Read encoder data and update heading angle
Correct angle using correctAngle() function
Update motor power and direction based on heading angle and distance sensors
Implement parking logic and path correction
runEncoder(): Running Encoder (defined above)

Main Loop
Create multiprocessing objects for servo drive and encoder processes
Start servo drive and encoder processes
Run until interrupted

Cleanup
Stop servo drive and encoder processes
Close serial connections and GPIO pins
Clean up IMU and GPIO resources

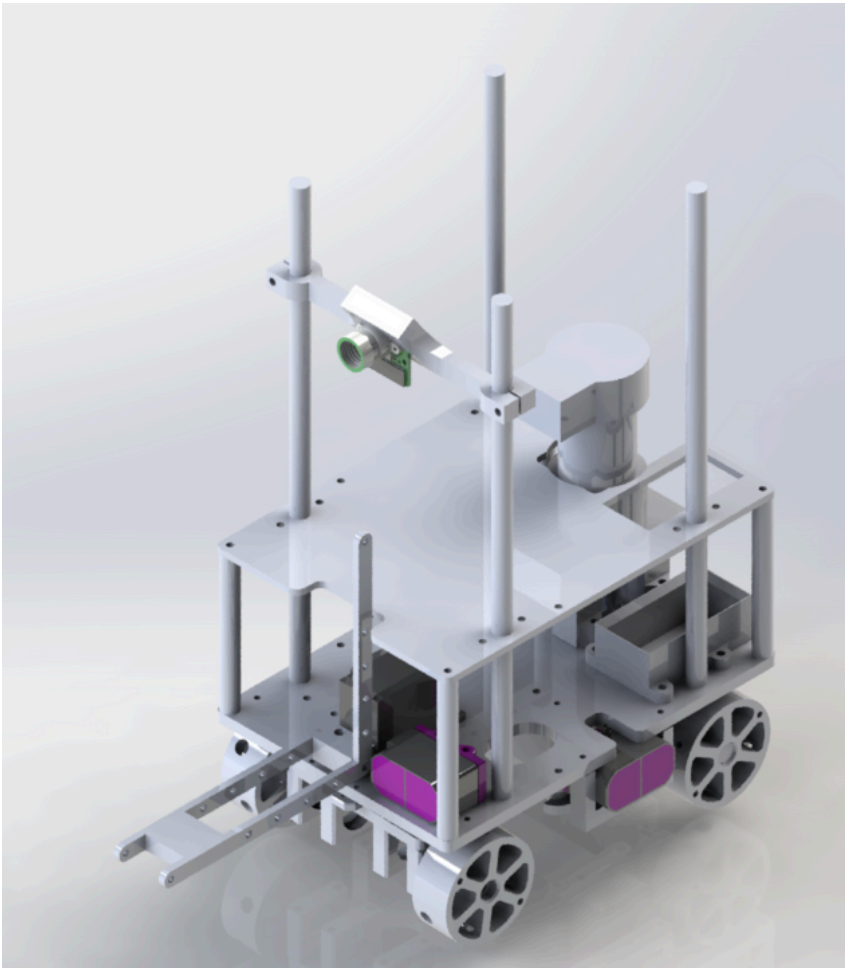


GALLERY

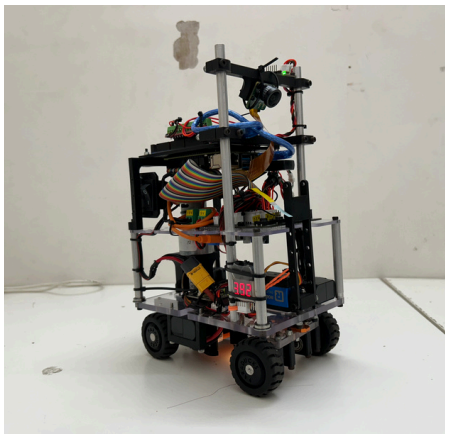
FUTURE ENGINEERS 2024

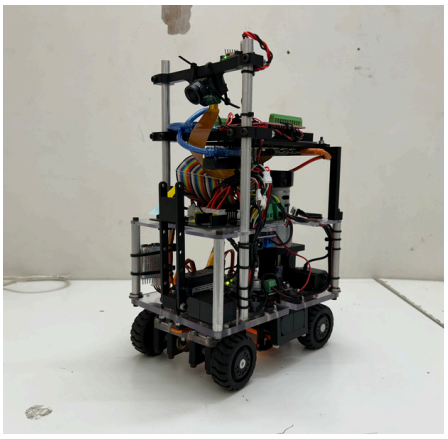
3D DESIGN

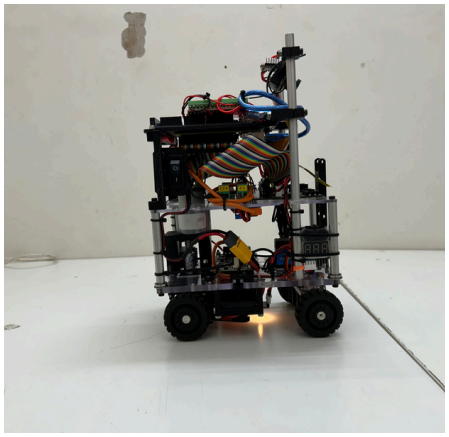
We designed our robot's online model using **Solidworks** as our software for **computer aided design (CAD)** which helped in a very high accuracy for sizing of components and creation of **custom 3D printed mounts**, helping us stay within the **stipulated robot limitations.**

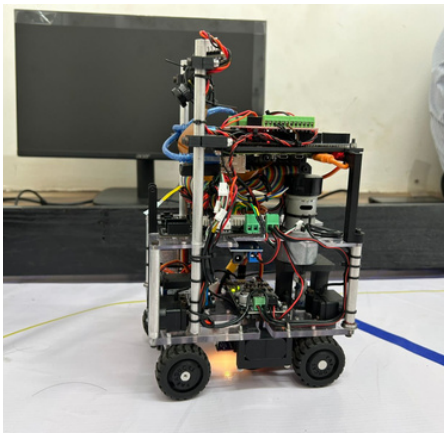


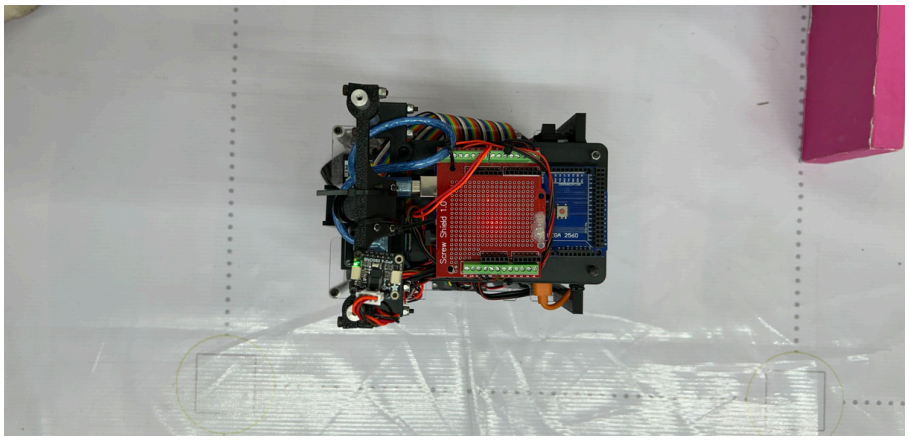
OUR ROBOT



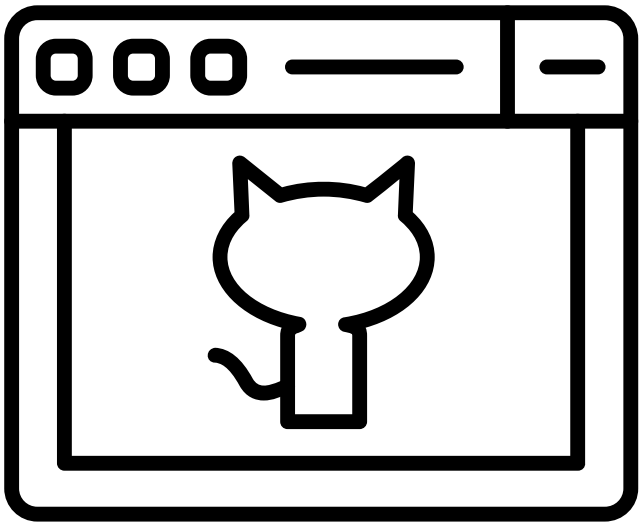


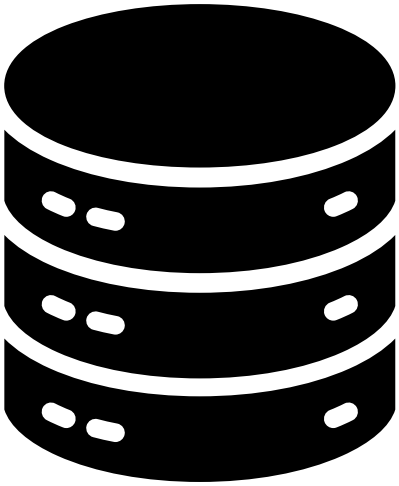


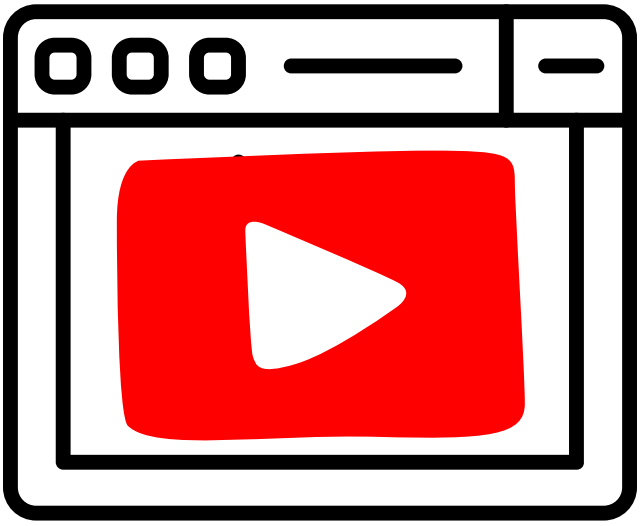




FUTURE ENGINEERS 2024







RESOURCES

FUTURE ENGINEERS 2024

GITHUB REPOSITORY

Scan to access our GitHub repository.

FUTURE ENGINEERS 2024

YOUTUBE CHANNEL

Scan to access our YouTube channel for robot videos.

FUTURE ENGINEERS 2024