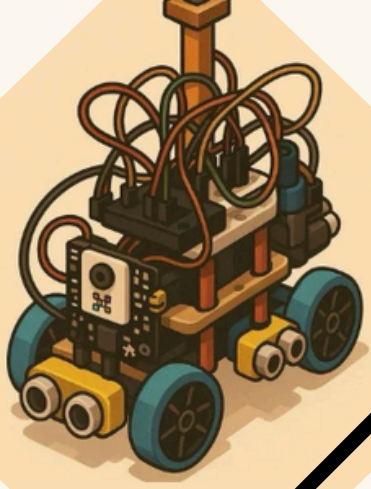


Team Apollo

WRO FUTURE ENGINEERS
2025
DOCUMENTATION

APOLLO

WRO®
WORLD ROBOT OLYMPIAD™

FUTURE
ENGINEERS

A SELF-DRIVING
CAR CHALLENGE

AGE GROUPS:
14-22

WRO® 2025
SELF-DRIVING CARS

WRO international premium partner

aramco

WRO international gold partners

fischertechnik

ROBOROBO
LEARNING FUTURE

ABSTRACT

OBJECTIVE

This documentation outlines the full behavior of the robot in two competition rounds: Open Challenge Round and Object Avoidance Challenge Round. The robot navigates the arena using a combination of MPU6050 gyroscope, ultrasonic sensors, and a Pixy2 vision system, while steering is controlled through a PID-based system. Each round is structured to test decision-making, directional control, object interaction, and path planning.

METHODS

To achieve intelligent autonomous navigation, we integrated vision-based recognition using the HuskyLens, combined with orientation and motion feedback from the MPU6050 IMU and distance sensing through ultrasonic sensors. Customized programming logic interprets this data in real-time to enable precise turns, controlled acceleration, and accurate stopping. Our system also employs energy-efficient routines to manage power consumption effectively. Rigorous pre-run diagnostics and sensor calibration ensure the robot's readiness and consistency throughout each challenge.

KEY ENGINEERING FACTORS

This project emphasizes a multi-sensor fusion approach for real-time environmental awareness and obstacle navigation. The integration of machine vision with motion and proximity sensors enables the robot to react swiftly and intelligently. The vehicle design supports optimal sensor placement and stability, while modular programming ensures flexibility for quick adaptations during testing. Our decision-making algorithms were developed to handle dynamic conditions with minimal delay, contributing to the robot’s robustness and reliability in competition scenarios.

CONCLUSION

This document showcases a holistic engineering approach that leverages sensor integration, vision processing, and adaptive coding to build a high-performance self-driving vehicle. Our participation in the WRO Future Engineers 2025 competition reflects a commitment to innovation, precision, and continuous improvement, with a strong focus on real-world applications of autonomous systems and robotics.

CONTENT

1. About us 1

2. Robot Photos 2

3.Robot’s Chassis 3

4. Electronic Components 5

5. Obstacle Management 10

- Round 1: Open Challenge Round
- Round 2: Object Avoidance Challenge Round

6. Source Code 16

7. Video files 18

1.ABOUT US

We're Pranav, Mifzal, and Arnav — three minds, one mission. Forged through high-stakes arenas like FLL, FTC, and FGC, we've not only competed but conquered, clinching awards and mastering the art of robotics along the way.

With a track record of innovation, precision, and teamwork, we bring a well-rounded arsenal of technical expertise and creative problem-solving. From building robust bots to devising efficient strategies, our journey has been one of constant evolution.

We don't just chase challenges — we transform them into opportunities to learn, lead, and leave a legacy. WRO is more than just another chapter; it's our next leap forward.

Team History

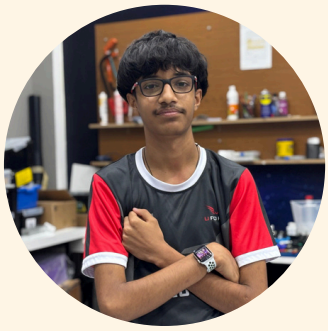
Team Apollo has established itself as a powerhouse in the WRO Future Engineers category, securing the National Winner title consecutively in 2021 and 2022. Representing the UAE on the global stage, the team showcased exceptional engineering and innovation skills.

In 2021, Team Apollo competed in the WRO International (Online – Germany) and achieved an impressive 7th place worldwide. Building on that momentum, the team returned stronger in 2022, earning a 6th place finish and was honored with the prestigious Silver Category Award.

As the defending national champions, Team Apollo continues to set benchmarks in robotic excellence and is determined to lead the way once again.

Our Team

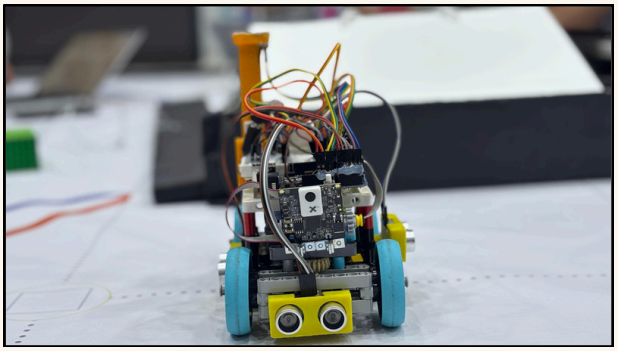
Pranav Nakkeeran is a Grade 12 student at Delhi Private School, Sharjah, and a passionate robotics enthusiast. He has represented the UAE in global competitions like FLL, FTC, and FGC, and is driven by a love for innovation and real-world problem solving.

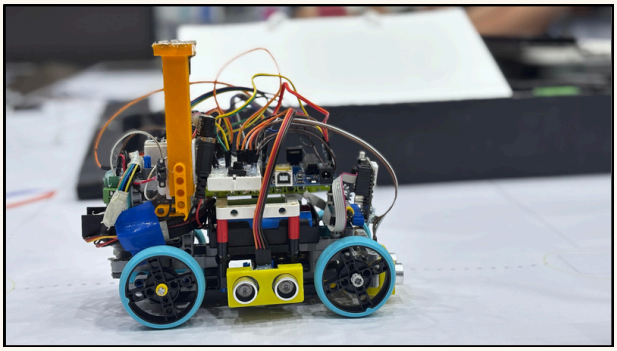


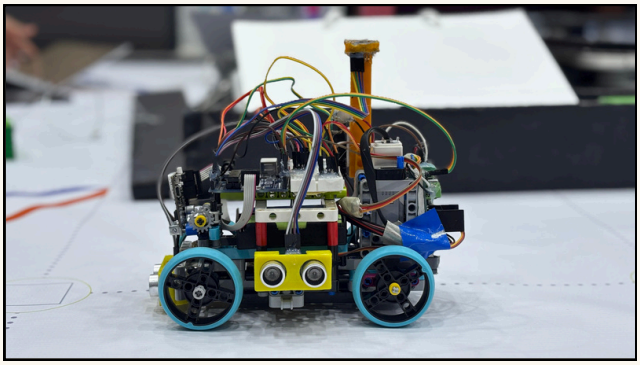
Aarnav Bhargava is a Grade 10 student at GEMS Modern Academy and a passionate robotics enthusiast. He has competed in and won several competitions including FLL and FTC. Keeps empowering youth through innovation and tech.

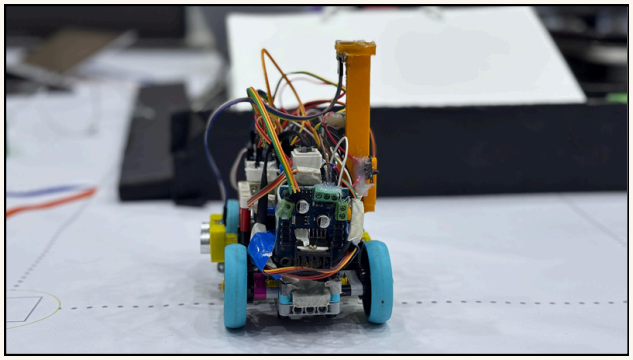
Mohamed Mifzal Maharoof is a Year 12 student at GEMS New Millennium School, Dubai, passionate about robotics, AI, and embedded systems. He has led and built projects using Arduino, Raspberry Pi, and ESP32, FTC, FLL, FGC and coaches FLL teams while exploring real-world tech solutions.

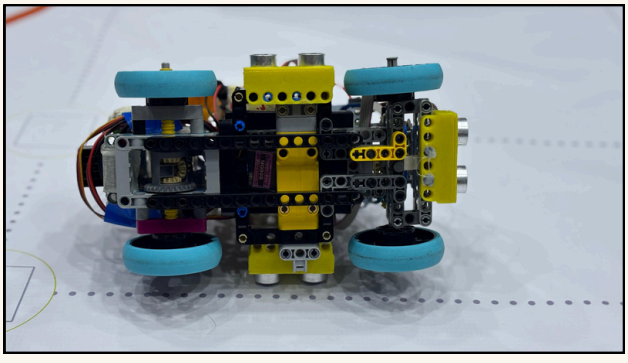
2. ROBOT PHOTOS

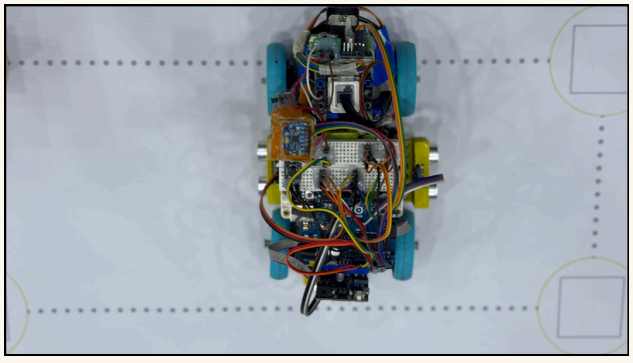




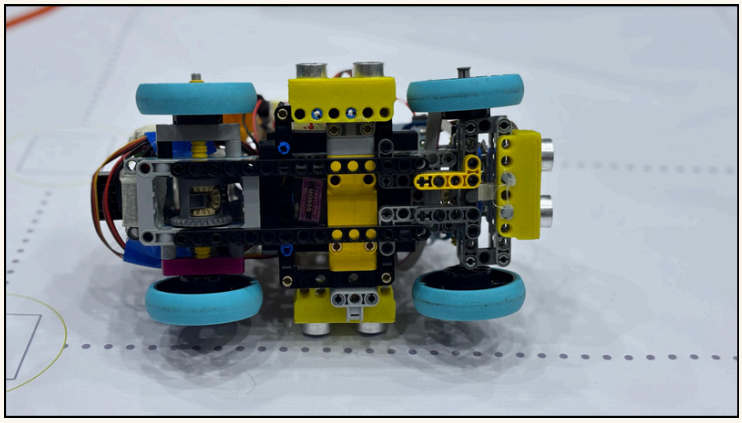








3. ROBOT'S CHASSIS



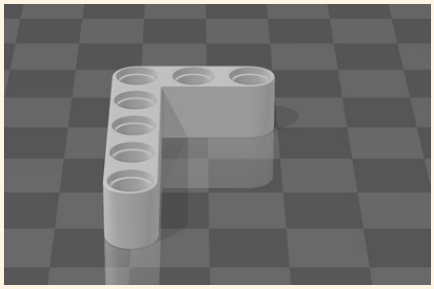
The robot's chassis was meticulously custom-designed using Fusion 360, enabling precise control over every aspect of its form and function to meet the specific needs of our WRO Future Engineers 2025 project. Using PLA filament, the frame was 3D printed in multiple interlocking parts to achieve a lightweight yet sturdy structure that balances durability with ease of manufacturing.

The design incorporated dedicated compartments and mounting slots tailored for secure installation of all critical components, including the Arduino Uno, L298N motor driver, various sensors like the HuskyLens and MPU6050, as well as the 3-cell 18650 battery pack. To mount the ultrasonic sensors, custom 3D printed brackets were designed to hold them securely in forward and lateral positions, ensuring consistent alignment and stable distance measurements. Additionally, LEGO components were creatively integrated into the design to support and anchor certain modules, offering modular flexibility and structural reinforcement where needed. By engineering the chassis in-house, we optimized the overall layout for a low center of gravity, improving stability during navigation and sharp maneuvers.

Differential Gear

A differential gear is a mechanical component that allows two output shafts to rotate at different speeds while receiving power from a single input source. In robotics, especially in wheeled robots, it plays a crucial role in enabling smooth and efficient turning. When the robot makes a turn, the wheels on the inside and outside of the turn need to rotate at different speeds—this is where the differential gear becomes essential. By distributing torque appropriately between the wheels, it prevents slippage, reduces strain on the motors, and improves maneuverability. In our robot, the differential gear system allowed for stable and controlled steering, particularly during sharp turns or uneven terrain, enhancing both performance and reliability in dynamic environments.

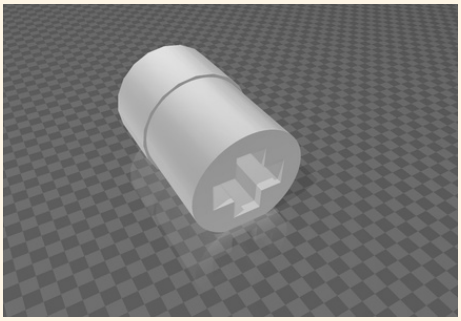
CUSTOM 3D PRINTED COMPONENTS



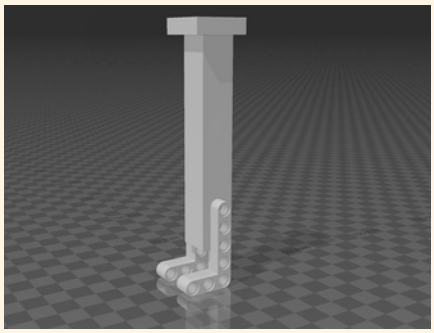
Chassis Holder

The Chassis Holder is a custom 3D-printed component that provided essential structural support to the robot's frame. It also served as a stable mounting base for key elements like the motors, sensors, and control boards, ensuring secure placement and optimal functionality.

The Servo to Axle Connector is a custom 3D-printed component designed to securely link the servo motor to the robot's wheel axle, enabling precise navigation and steering control.
4o



Servo to Axle Connector



MPU6050 Holder

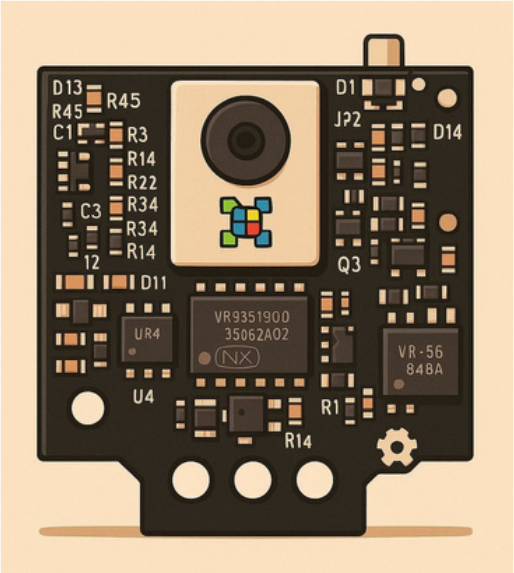
The MPU6050 Holder is a custom 3D-printed mount designed to elevate and isolate the gyro sensor, effectively minimizing vibration-induced interference and enhancing motion detection accuracy.

3D Printer:

The robot's chassis was 3D printed using the Creality K1 Max, a high-speed, high-precision FDM printer with a large 300 x 300 x 300 mm build volume. Its 600mm/s print speed, CoreXY structure, input shaping, and AI-assisted error detection ensured accurate, high-quality prints with minimal artifacts. Features like a heated bed, direct-drive extruder, and real-time monitoring enabled smooth PLA extrusion and reduced material wastage. These capabilities allowed us to efficiently prototype and finalize a durable, modular chassis suitable for competition-level performance.

4. ELECTRONIC COMPONENTS

Pixycam v2

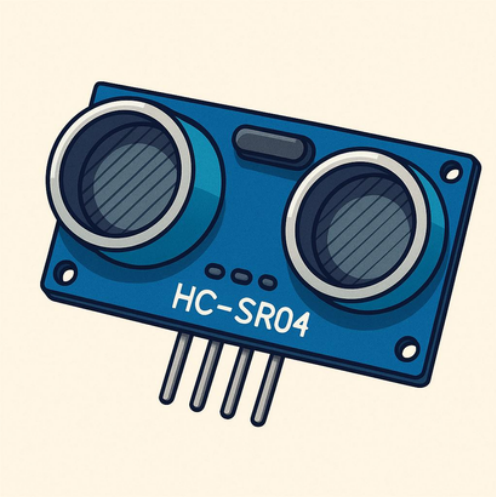


PixyCam is a compact, intelligent vision sensor designed for recognizing and tracking objects based on color signatures. It features a built-in image processor that eliminates the need for external computation, allowing it to detect and follow objects in real time.

In our robot, PixyCam is used during visual challenges to identify and react to colored markers on the field. These color signatures correspond to specific actions like turning, stopping, or path selection. The sensor sends coordinates, color signature IDs, and size data to the Arduino, which then determines the appropriate response.

We chose PixyCam because of its fast processing speed, ease of training, and ability to detect multiple color objects simultaneously. It offers a plug-and-play solution for computer vision in embedded systems, and it integrates cleanly with Arduino via UART or SPI.

This sensor plays a vital role in giving our robot vision-based decision-making, improving its interaction with dynamic environments and challenge-specific elements.

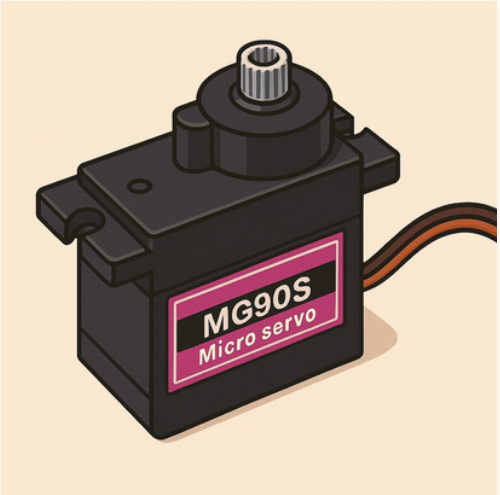


Ultrasonic Sensors (3x HC-SR04)

We chose the HC-SR04 ultrasonic sensor for our WRO Future Engineers 2025 robot due to its lightweight design, low cost, ease of interfacing with Arduino via digital I/O pins, and sufficient accuracy for our competition environment. The HC-SR04 measures distance by emitting high-frequency sound pulses and calculating the time taken for the echo to return, providing reliable proximity data. Our robot uses three HC-SR04 sensors—mounted at the front, left, and right—to ensure comprehensive spatial awareness.

The front sensor handles collision detection, allowing the robot to stop before hitting obstacles, while the side sensors support wall tracking and edge navigation by maintaining alignment with the track and guiding smart turning decisions. In software, we apply filtering techniques to smooth sensor noise and enhance measurement stability. When the front sensor detects an obstacle, the robot compares side distances and turns toward the clearer path, a strategy that also helps determine optimal navigation direction during autonomous laps.

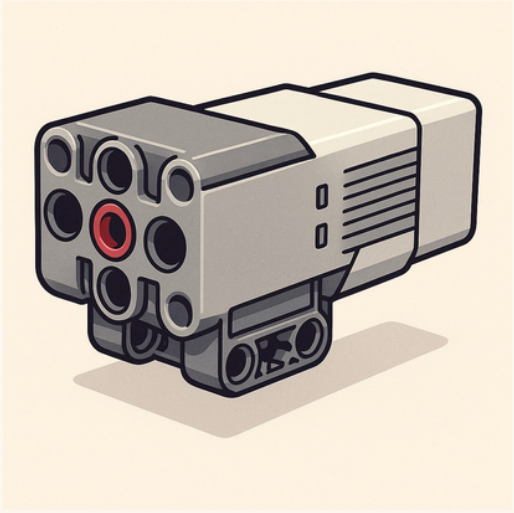
MG90s Servo Motor (180 Degrees)



We chose the MG90s micro servo motor for our WRO Future Engineers 2025 robot because of its combination of precision, durability, and responsiveness—key factors for effective autonomous steering. This high-torque servo offers 180° rotational movement, allowing the robot to execute sharp turns, U-turns, and directional corrections with ease. Its metal gear construction ensures long-term reliability and wear resistance, especially during repeated steering actions throughout multiple laps. Compact and lightweight, the MG90s fits seamlessly into our design without adding bulk, while its simple PWM control via Arduino allows for smooth integration into our control system. In lour build, the MG90s is mechanically linked to the front wheels through a bevel gear system, which ensures synchronized movement of both wheels for consistent and precise directional changes. This configuration enables the robot to respond rapidly to input from onboard sensors and vision systems, enhancing its agility and adaptability in both obstacle-rich areas and open-track sections. The servo's quick response time and positional accuracy make it ideal for real-time steering control, ultimately improving the overall stability, navigation precision, and performance of the autonomous vehicle.

LEGO EV3 Medium Servo Motor

IWe chose the LEGO EV3 Medium Motor (45503) for our WRO Future Engineers 2025 robot because it offers an ideal balance of high torque, compact design, and precise speed control through its built-in rotary encoder. This motor was used to drive the rear wheels via a differential gear system, ensuring efficient and smooth linear motion across a variety of field surfaces.



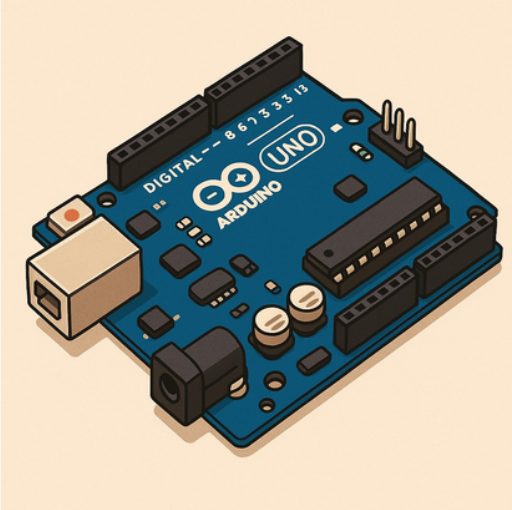
Its ability to maintain consistent torque and RPM under load makes it highly reliable for tasks requiring stable performance, such as straight-line driving, PID-based speed regulation, and maneuvering through tight corners. The motor's robust housing ensures long-term durability, especially during repeated use in dynamic and demanding competition environments. We integrated the EV3 Medium Motor with an Arduino using an L298N motor driver, enabling PWM-based control and real-time speed adjustments based on sensor feedback from the ultrasonic sensors and the MPU6050 gyroscope. Its compatibility with our drivetrain setup, combined with its power efficiency and mechanical resilience, made it the optimal choice for achieving reliable autonomous navigation and responsive obstacle-avoidance throughout the course.

L298N Motor Driver

We selected the L298N motor driver for our WRO Future Engineers 2025 robot due to its dual H-bridge configuration, which enables independent control of two DC motors—perfect for managing the LEGO EV3 Medium Motors in our differential drive system. It acts as a reliable bridge between the Arduino and the motors, supporting PWM-based speed control and directional reversal through simple digital logic signals. This flexibility allows the robot to perform real-time speed adjustments, smooth turns, and PID-based corrections, all driven by sensor inputs. The L298N's integrated heat sink provides thermal protection during extended operation, and its 2A per channel current capacity ensures stable performance even under heavy motor loads.

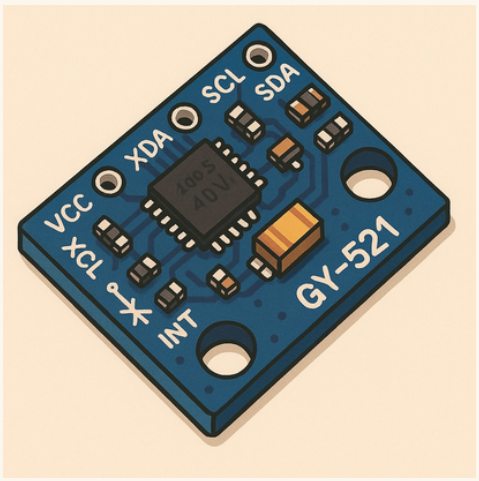
Readily available and cost-effective, the module's robust build made it an ideal choice for our design, offering the reliability, power handling, and versatility required for consistent motion control in both linear driving and obstacle negotiation throughout the competition.

Arduino Uno (ATmega328P)



We chose the Arduino Uno as the central microcontroller for our WRO Future Engineers 2025 robot due to its wide compatibility, real-time performance, and robust hardware architecture. Featuring an ATmega328P chip with 14 digital I/O pins (including 6 PWM outputs), 6 analog inputs, and USB programmability, the Uno seamlessly integrates with all our components—such as the HC-SR04 ultrasonic sensors, MPU6050 IMU, HuskyLens vision sensor, and the L298N motor driver. Its stable power management via the Vin pin allows it to be directly powered by our 11.1V battery pack, while a shared ground across all modules ensures reliable signal integrity. The Uno's support for C++ and a vast collection of open-source libraries made it easy to implement complex control logic, including obstacle avoidance, PID-based driving, and sensor fusion. Acting as the robot's "brain," it processes real-time sensor data, executes decision-making algorithms, and orchestrates motor and servo actions, making it a reliable and efficient controller for our autonomous navigation system.

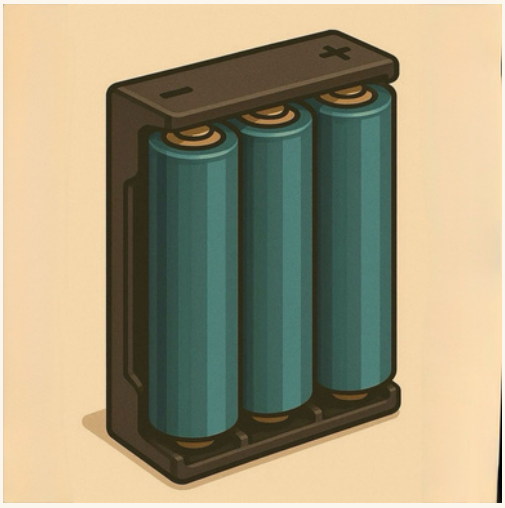
GY-521 (MPU6050) – Accelerometer and Gyroscope



We chose the MPU6050, integrated on the GY-521 module, for its compact size, 6-axis motion sensing, and I2C communication, which allowed seamless integration with the Arduino Uno. Combining a 3-axis gyroscope and a 3-axis accelerometer, the MPU6050 provided reliable real-time motion and orientation data critical for maintaining directional stability in our WRO Future Engineers 2025 robot. We used the gyroscope's yaw data to monitor angular drift, enabling accurate course correction via a PID feedback loop.

When the robot veered off its intended heading, the Arduino adjusted motor outputs to realign it. The accelerometer supported this by verifying orientation changes during rapid turns or terrain-induced shifts, enhancing the system's overall accuracy. This dual-sensor feedback ensured smooth navigation, consistent path-following, and precise turning performance. Its stable output and ease of integration made the MPU6050 an essential component for real-time motion correction and robust navigation in dynamic competition environments.

Power Supply (3x 18650 Battery Pack)



The power supply system of our WRO Future Engineers 2025 robot is built around a 3-cell 18650 lithium-ion battery pack, connected in series to deliver a nominal voltage of 11.1V (3.7V per cell × 3). This configuration offers a reliable balance between energy density, weight, and sustained current output, making it ideal for robotics applications that require continuous operation of multiple components.

The Arduino Uno is powered directly via its Vin pin, which accepts input voltages between 9V and 12V. Supplying 11.1V ensures stable onboard voltage regulation, which in turn powers the logic-level circuitry and sensor modules such as the MPU6050, HC-SR04, and HuskyLens.

The L298N motor driver also draws from the same 11.1V source to power the LEGO EV3 Medium Motors, which require consistent voltage and current for reliable performance under load. The L298N's built-in voltage regulation and heat dissipation features make it suitable for handling the motor's demands without risk of overheating.

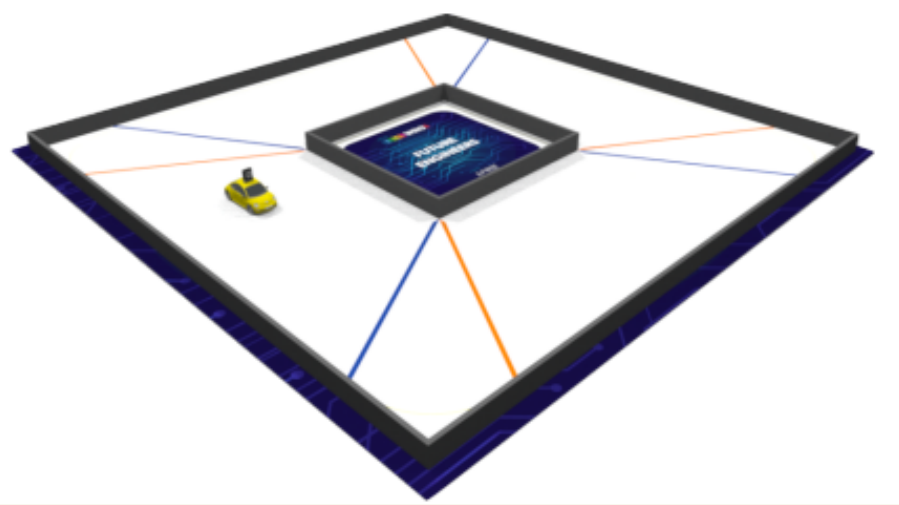
To ensure electrical integrity and signal stability, a common ground (GND) is established across all connected components — including the Arduino, sensors, motor driver, and servos. This shared reference point eliminates potential voltage mismatches that could interfere with communication and control signals.

Additionally, the 18650 cells provide high capacity (typically 2000–3000mAh each), offering extended runtime for prolonged autonomous tasks. The power setup was tested under various load conditions and found to deliver sufficient current for all modules to operate simultaneously without experiencing voltage sag, brownouts, or overheating — ensuring smooth and uninterrupted robot operation throughout the competition.

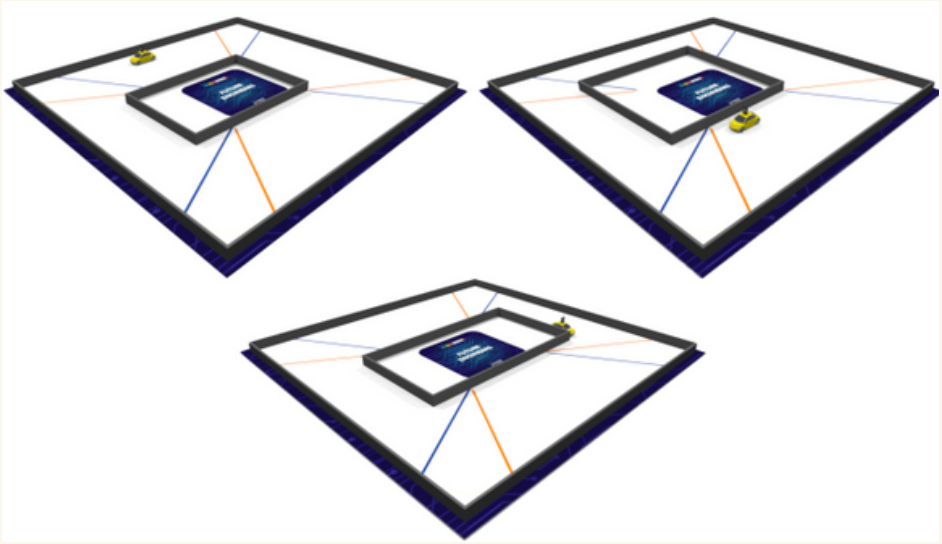
5. OBSTACLE MANAGEMENT

Round 1: Open Challenge Round

During Open Challenge rounds, the racetrack will have no traffic signs.



The distance between the track borders could be either 1000 mm or 600 mm (+/- 100 mm for the International Final).



1. Initialization and Direction Decision

Upon power-up:

The robot activates the gyroscope and begins tracking angular orientation.

Using its left and right ultrasonic sensors, the robot identifies which side of the arena is open:

If the right sensor detects a distance greater than 80 cm, the robot infers a clockwise route.

If the left sensor sees more than 80 cm, it chooses a counter-clockwise route.

This decision is made only once at the start and sets the rotational direction for the entire round.

2. Edge Detection and Lap Counting

The robot counts each 90-degree corner as one “edge”.

Each full lap includes four corners. A total of 12 edges indicates 3 laps completed.

At each edge, the robot executes a 90-degree turn using gyro data to maintain heading accuracy.

Steering corrections during straight motion are applied using PID logic, factoring in gyro errors and wall distances.

3. Wall Following and Anti-Collision

The robot maintains a:

30 cm buffer from the inner wall (for smooth line tracking).

Minimum of 20 cm from the outer wall (to prevent crashing).

Left and right ultrasonic sensors provide real-time distance data, used to calculate inner and outer wall error terms.

These values are included in the PID formula to dynamically adjust steering angles and stabilize path following.

4. Completion Behavior

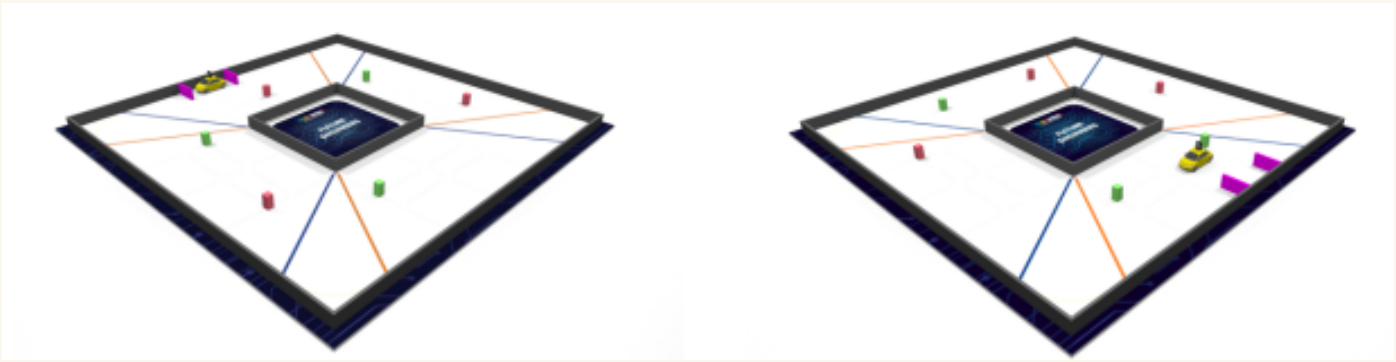
After detecting 12 edges (3 laps), the robot:

Performs no turnaround.

Stops moving after a short forward motion, marking the end of the Open Challenge round.

Round 2: Object Avoidance Challenge Round

During Obstacle Challenge rounds, the red and green pillars will be set up on the racetrack as the traffic signs. In addition, two boundaries will be placed and form a parking lot. The distance between the track borders will be always 1000 mm (+/- 10 mm for the International Final).



1. Initialization and Navigation

The robot begins similarly, identifying direction (clockwise or counter-clockwise) using ultrasonic readings.

It maintains its heading using gyro feedback and PID-based steering until it encounters either:

A corner to turn.

A colored cube detected by the Pixy2 camera.

2. Color-Based Obstacle Avoidance

The robot continuously scans using the Pixy2 camera for color signatures:

Red blocks (Signature 1): The robot gently steers right to pass beside the block.

Green blocks (Signature 2): The robot steers left to avoid it.

The X-coordinate of the detected block is used to determine offset from the robot's center and calculate a proportional correction.

The height of the block is used to estimate distance; only nearby objects are considered.

3. Steering Control – PID Implementation
The robot's servo steering is managed using:

$$\text{Correction} = Kp \cdot \text{gyroError} + Kw \cdot \text{innerWallError} + Ko \cdot \text{outerWallPenalty}$$

Gyro Error: Keeps the robot aligned with the intended direction.
Inner Wall Error: Adjusts for deviation from the ideal 30 cm distance.
Outer Wall Penalty: Activates when the robot gets closer than 20 cm to the outer wall.
The computed correction is applied to the servo, which has a constrained range (typically 60°–120°).

4. Edge Counting and Final Lap Behavior
Just like in Round 1, the robot counts edges to determine lap progress.
On reaching the third lap, the behavior diverges:
The robot does not perform a turnaround (no 180° pivot as in prior logic).
Instead, it transitions into a parking sequence.

5. Parallel Parking Execution
The robot begins searching for a suitable parking spot using camera and distance sensors.
Once identified:
It aligns itself beside the zone using steering adjustments.
Executes parallel parking by reversing into the space while maintaining orientation using gyro and servo feedback.
The motors are then stopped, concluding the challenge.

PID-Based Steering Control

The robot uses a PID (Proportional-Integral-Derivative) controller to calculate the steering angle needed to stay centered between the walls and maintain its heading. This is done using input from the MPU6050 gyroscope and left/right ultrasonic sensors.

What Is PID Control?

PID is a feedback mechanism used in robotics and automation to minimize error between a desired setpoint and the current state. It does this using three terms:

Proportional (P):
Reacts to the present error (how far off the robot is from its ideal direction).

In your case, this is based on the gyro angle deviation from the target heading.

Integral (I):
Reacts to the accumulated error over time, useful for correcting steady drift or bias.

You accumulate small heading errors here (e.g., if robot constantly drifts left).

Derivative (D):
Predicts future error by checking how fast the error is changing.

Helps dampen oscillations or sudden movements in heading.

$$\text{Correction} = K_p \cdot \text{gyroError} + K_w \cdot \text{innerWallError} + K_o \cdot \text{outerWallPenalty}$$

gyroError = target heading - current heading

innerWallError = ideal inner wall distance (30 cm) - actual inner distance

outerWallPenalty = activates if distance to outer wall is < 20 cm

These are weighted by the gains:

Kp: Proportional gain for heading (gyro)

Kw: Proportional gain for wall following

Ko: Outer wall crash avoidance correction

How PID Affects the Servo:

The total correction value is added to the center servo angle:

```
servoAngle = SERVO_CENTER + correction;
servoAngle = constrain(servoAngle, 60, 120);  // Safe servo bounds
steeringServo.write(servoAngle);
```

So:

If the robot drifts off its heading or wall distance, the servo turns slightly left or right to realign.

If it's aligned correctly, the servo remains around 90° (straight).

Color Detection Logic

The robot also uses Pixy2 to:
Detect red and green blocks
React based on X-position of the block:
Red block: steer right
Green block: steer left
The alignment is handled via:

```
servoAdjustment = KP_SERVO * (currentX - targetX);
myServo.write(SERVO_CENTER - servoAdjustment);
```

This block alignment temporarily overrides gyro-based PID, but once the object is passed, PID resumes for wall and heading control.

6. SOURCE CODE

Reliable Sensor Reading

```
float getFilteredDistance(NewPing &sonar) {
  const int samples = 3; // Sample count
  float sum = 0;
  for (int i = 0; i < samples; i++) {
    float dist = sonar.ping_cm();
    if (dist == 0) dist = MAX_DISTANCE; // Handle no response
    sum += dist;
    delay(5); // Allow sensor to settle
  }
  return sum / samples;
}
```

• Purpose: Averages multiple ultrasonic readings to reduce noise.
• Impact: Provides stable and trustworthy data used in wall-following and edge-detection decisions.
• Why filter? Ultrasonic sensors are prone to false or noisy readings. Using 3 samples reduces random fluctuations.

Precision Turns Using Gyroscope

```
void performTurn(bool clockwise) {
  mpu.update();
  float startAngle = mpu.getAngleZ();
  float turnTarget = startAngle + (clockwise ? -88 : 88); // Turn ±88 degrees
  analogWrite(MOTOR_ENB, 0);
  steeringServo.write(clockwise ? SERVO_MAX_RIGHT : SERVO_MAX_LEFT);

  while (abs(mpu.getAngleZ() - turnTarget) > 2) {
    mpu.update();
    analogWrite(MOTOR_ENB, SLOW_SPEED);
  }

  analogWrite(MOTOR_ENB, 0);
  steeringServo.write(SERVO_CENTER);
  targetAngle = turnTarget;
}
```

• Purpose: Turns the vehicle accurately by ~90°.
• Decision Factor: Chooses turn direction based on wall distances and uses MPU6050 to measure turn angle.
• Precision Control: Ensures consistent turns even after multiple laps (essential for edge counting).

Main Control Logic:

loop() Core Navigation & Correction

```
float gyroError = targetAngle - currentAngle;
int servoAngle = SERVO_CENTER;
if (directionSet && !isWaiting) {
  float innerDist = isClockwise ? leftDist : rightDist;
  float outerDist = isClockwise ? rightDist : leftDist;
  float wallError = TARGET_WALL_DISTANCE - innerDist;
  float outerError = outerDist < OUTER_WALL_THRESHOLD ? OUTER_WALL_THRESHOLD - c

  servoAngle += (int)(KP * gyroError + KW * wallError + KO * outerError);
} else {
  servoAngle += (int)(KP * gyroError);
}
servoAngle = constrain(servoAngle, 60, 120);
steeringServo.write(servoAngle);
```

• Multifactor Decision: Blends three correction mechanisms:
1. Gyro (MPU) to correct angular drift.
2. Inner Wall distance to maintain center path.
3. Outer Wall avoidance when too close to obstacles.
• Dynamic Steering: Adjusts the servo in real-time using a weighted formula.
• Smart Navigation: Enables adaptive behavior depending on lap direction (isClockwise) and surrounding environment.

Turn Initiation Based on Sensor Input:

```
if (!directionSet) {
  if (rightDist > SIDE_TURN_DISTANCE && rightDist < MAX_DISTANCE) {
    isClockwise = true;
    directionSet = true;
    edgeCount++;
    performTurn(isClockwise);
    isWaiting = true;
    waitStartTime = millis();
  } else if (leftDist > SIDE_TURN_DISTANCE && leftDist < MAX_DISTANCE) {
    isClockwise = false;
    directionSet = true;
    edgeCount++;
    performTurn(isClockwise);
    isWaiting = true;
    waitStartTime = millis();
  }
}
```

• Decision Maker: This is where the robot decides its lap direction based on which side wall disappears first.
• Sets Behavior: Once set, isClockwise changes all future logic.
• Edge Detection: First increment of edgeCount, starting lap tracking.

Edge Detection & Lap Completion:

```
float outerDist = isClockwise ? rightDist : leftDist;
if (outerDist > SIDE_TURN_DISTANCE && outerDist < MAX_DISTANCE) {
  edgeCount++;
  if (edgeCount >= TOTAL_EDGES) {
    performTurn(isClockwise); // Final turn
    delay(1000); // Move forward
    analogWrite(MOTOR_ENB, 0); // Stop
    isRunning = false;
    while (1); // Halt forever
  }
  performTurn(isClockwise);
  isWaiting = true;
  waitStartTime = millis();
}
```

- Edge Counting: Core mechanism to track progress across laps and corners.
- Laps Tracking: Uses a fixed TOTAL_EDGES = 13 to know when to stop.
- Autonomous Exit: Robot completes a routine and safely stops.

Round 1: Open Challenge round video

Round 2: Obstacle Challenge round video

Our Github