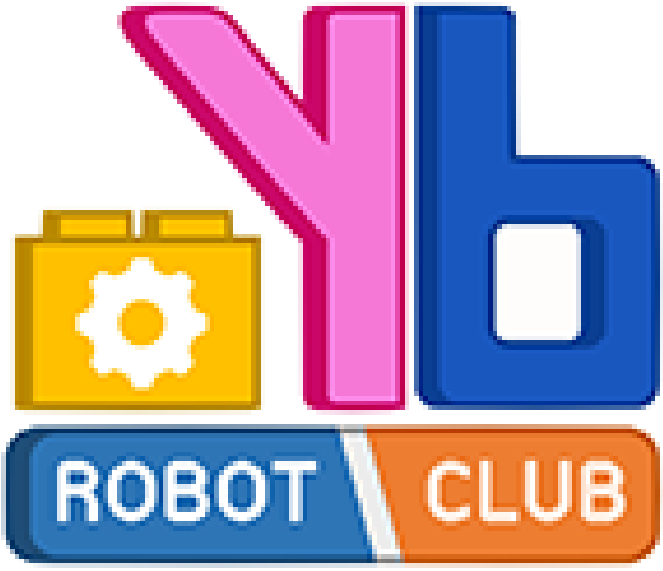


World Robot Olympiad Thailand 2025

Future Engineers Categories

Documentation

YB-NEAB

Content

1. About Our Team
2. Engineering Consideration
3. Chassis Design and Robot Photos
4. Hardware Information
5. ROS 2 Humble Explanation

1. About Our Team

Our team consists of three members; each assigned a different role:

1. **Vichaiwat Koonsap** – I am responsible for gathering data for our team members and taking detailed notes, which are then compiled into this documentation. I'm the on the **middle**.
2. **Norrapat Kesthong** – Norrapat is responsible for designing and building our robot. He's on the **right** side.
3. **Vorawet Narkglom** – Vorawet is the programmer who coded the robot. He stands on the **left** side.

2. Engineering Consideration

During the design process of our robot, we faced several engineering challenges, particularly in translating real-world measurements into our 3D modeling software. After printing, we frequently encountered issues with small features—for example, a M3 screw thread could not be printed accurately caused by 3d printer's nozzle size, resulting in undersized holes. This showed the importance of accounting for 3D printer limitations during the design stage. Paying close attention to these details helps minimize unnecessary redesigns and reprints.

Another key learning was the importance of incorporating a differential system. At first, we placed gears of equal size next to each other, but this setup proved ineffective. By integrating a differential, the wheels were able to rotate at different speeds—an essential factor for smooth turning and overall robot performance.

The robot chassis was modeled in Fusion 360 and manufactured using FDM 3D printing. This approach provided durability, lightweight construction, and effective protection for the onboard hardware. The design was fully customized by our mechanic to meet the specific needs of our robot.

Our robot aim and objective

1. Innovation and Creativity:

-Encourage participants to design and build innovative robots that can perform complex tasks.

-Promote creative thinking and the development of unique solutions to the given challenges.

2. Engineering Skills:

-Enhance participant's engineering skills through the design, construction, and programming of advanced robots.

-Foster an understanding of mechanical design, electronics, and software integration in robotics.

3. Problem-Solving:

-Challenge participants to solve real-world problems using robotics, requiring analytical thinking and strategic planning.

-Develop participants' ability to troubleshoot and optimize their robotic systems.

4. Collaboration and Teamwork:

-Promote collaboration and effective teamwork among participants.

-Encourage communication and cooperation within teams to achieve common goals.

5. Application of STEM Knowledge:

-Apply theoretical STEM knowledge to practical scenarios, reinforcing learning through hands-on experience.

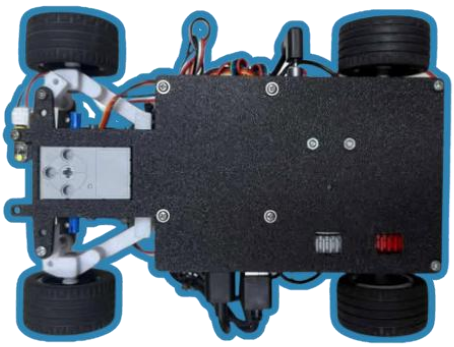
-Integrate multiple STEM disciplines to develop comprehensive robotic solutions.

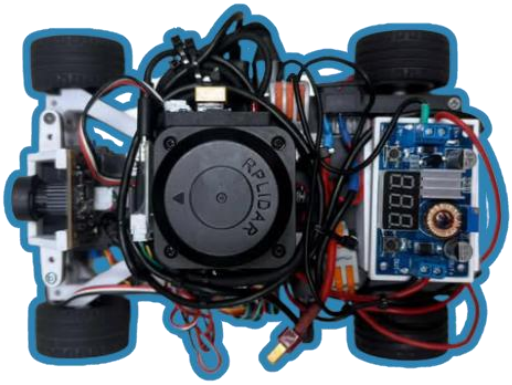
6. Presentation and Documentation:

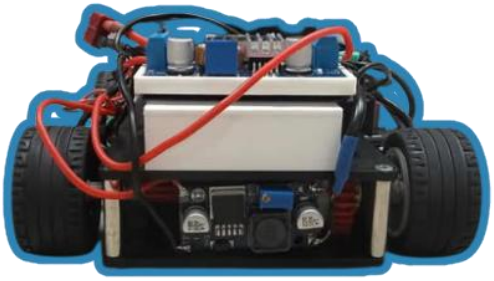
-Develop participants' skills in presenting their projects and solutions effectively.

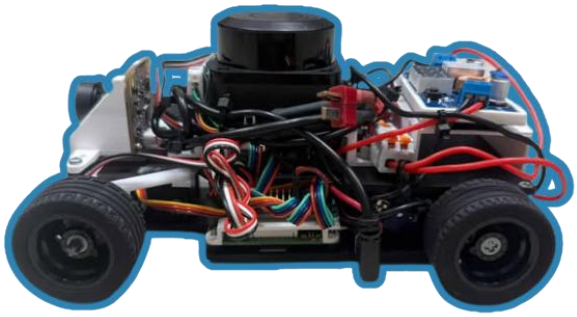
-Emphasize the importance of thorough documentation, including the design process, programming, and testing.

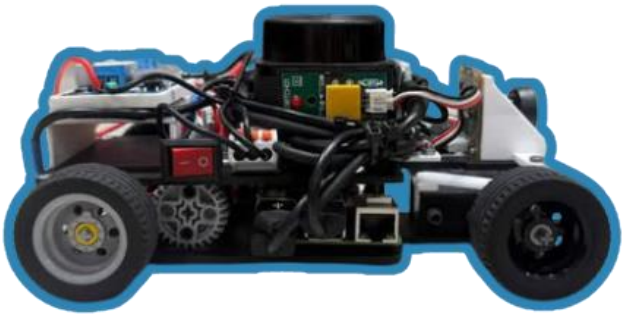
Robot Photos











3. Chassis Design

3.1 3D Printer and Filament

Bambu Lab P1S Combo

This is our 3D printer, which we use to produce both prototypes and the final robot chassis. It is capable of high-speed printing without compromising quality or risking failure. The printer features a fully enclosed chamber, allowing safe printing of high-temperature or hazardous materials such as ABS and ASA. It is one of Bambu Lab's best-selling models.

Additionally, it comes with an Automated Material System (AMS), enabling the printer to handle up to four different materials—whether the same or different types of plastic. The AMS also includes an Auto Reload function, allowing us to print overnight without worrying about filament running out.

Specification

Feature	Details
Model	P1S Combo
Build Volume	256 × 256 × 256 mm³
Print Speed	Up to 500 mm/s
Acceleration	Up to 20,000 mm/s²
Extruder Type	Direct Drive
Nozzle Type	Stainless Steel (upgradeable to hardened steel)
Hotend Max Temp	280°C (supports high-temp filaments with upgraded nozzle)
Heatbed Max Temp	100°C
Enclosure	Fully enclosed with carbon filter and fans
Cooling System	Auxiliary part cooling fan, chamber regulator fan, control board fan
AMS Compatibility	Yes (supports up to 16-color multi-material printing)
Connectivity	Wi-Fi, USB, Bambu-Bus
Display	2.7-inch 192×64 screen
Software	Bambu Studio (supports Cura, PrusaSlicer, SuperSlicer)
Camera	Built-in for remote monitoring and time-lapse
Dimensions (WxDxH)	389 × 389 × 458 mm³
Weight	9.5 kg
Power Input	100–240 VAC, 50/60 Hz
Max Power Consumption	1000 W @ 220V, 350 W @ 110V

Polylactic Acid or PLA

PLA is the most common filament used in 3D printing. It is easy to print, produces minimal stringing, and requires relatively low printing temperatures. However, it also has some drawbacks: it cannot withstand high heat and is more brittle compared to other materials. Because of this, we use PLA for thin parts that need to remain lightweight while still providing support without bending.

For our robot, we selected **Sunlu PLA+ 2.0**, an enhanced version of standard PLA. This material is stronger, lighter, and more flexible, giving us the benefits of PLA while improving durability and performance.

Polyethylene Terephthalate Glycol or PETG

PETG is another popular 3D printing filament, valued for its flexibility, strength, chemical resistance, and ability to withstand higher temperatures than PLA. Its main drawbacks are increased stringing and a slightly rougher surface finish.
Because PETG is highly impact-resistant, we use it for components that are more likely to experience collisions, ensuring the robot remains intact even after repeated impacts.

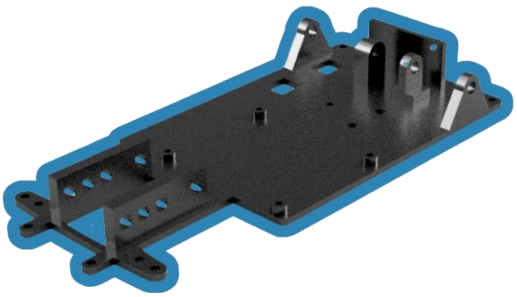
For our robot, we selected **Polymaker PolyLite PETG**. Polymaker is a trusted brand known for its consistency and reliability, with minimal defects. This PETG is particularly user-friendly, offering printability close to that of PLA while maintaining the toughness needed for demanding parts.

3.2 Printed Robot Chassis

All the 3D printable models can be found here.

Lower Base

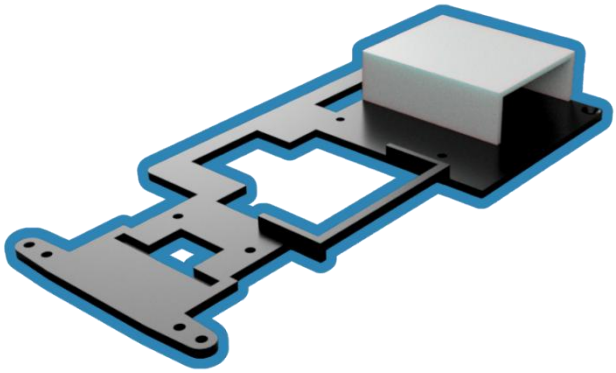
The lower base serves as the foundation of our robot, supporting nearly all essential components: the controller, motors, steering servo, light sensors, buck converter, and differential.



When designing this part, precise measurements were critical to ensure a proper fit for each component. In our robot, we uses a **LEGO TECHNIC** parts like axle, differential, and gear to ensure that those part would not break during our sessions.

To secure the LEGO axle, we designed support poles at the back of the base to hold it firmly in place. With this design, it allows the robot to reach its full potential.

Upper Base

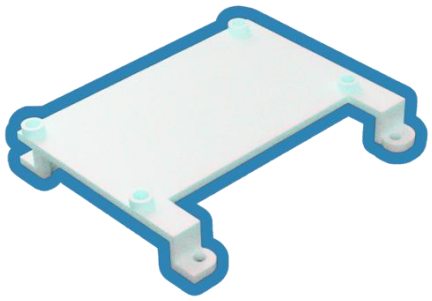


The upper base is positioned 3 cm above the lower base and is designed to hold the remaining components while providing additional stability to the robot.

To ensure a clean and seamless build, we applied the **counterbore technique** so that screws do not interfere with the lower section. This approach allowed us to hide the screws neatly, giving the chassis a screwless appearance.

Additionally, the upper base includes a **battery slot** at the back, ensuring secure placement and easy accessibility during operation.

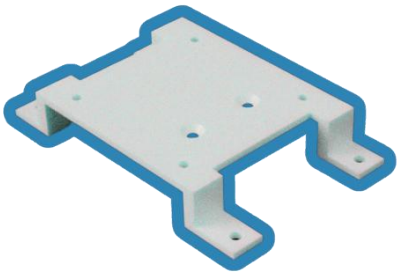
Arduino Nano Riser



Since we are using the Arduino Nano, we needed to find an ideal place to place it. The perfect location is right on top of the Raspberry Pi, allowing both boards to stay compact and organized. To achieve this, we designed the setup to use the same M2.5 screws that secure the Raspberry Pi. We also carefully accounted for the position of the GPIO pins and all necessary wiring, ensuring that everything fits neatly without interfering with connections.

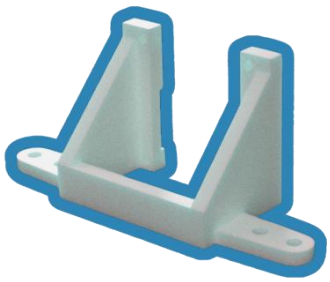
LiDAR Holder

The LiDAR holder is mounted on top of the upper base to keep the LiDAR elevated above other components. This prevents the sensor from mistakenly detecting parts of the robot itself.



The holder also included a **slot for the gyro sensor**, positioned directly beneath the LiDAR. This compact arrangement minimizes the overall robot size while ensuring that the LiDAR remains within the **10 cm height limit** to avoid detecting over the wall.

Camera mount



This component serves as a **stand for the camera** and includes protective covers for both the front and top. The design serves two key purposes:

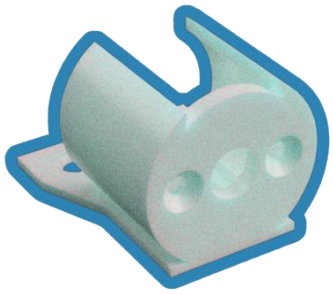
Protection – In case of collisions with walls, the front cover acts as a guard to protect the camera lens from damage.

Vision Control – The top cover blocks external light and distractions, ensuring the camera only captures the track area.

This combination enhances both the durability and reliability of the vision system during sessions.

Motor Bracket

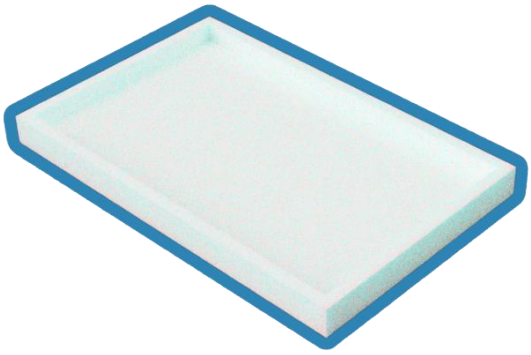
It holds the motor with the lower base. This design offer us a small, effective, and strong bracket. With 3 screw holes for attaching it to the base and 2 screw holes for attaching it to motor. This part also uses the Counterbore technique.



Gear Adapter

The gear we used was from **LEGO Technic**. Meaning that we can't attach it directly to the motor. We designed this part to place it on motor's rotating axle. Then, we added two LEGO pins to attach it to the gear.

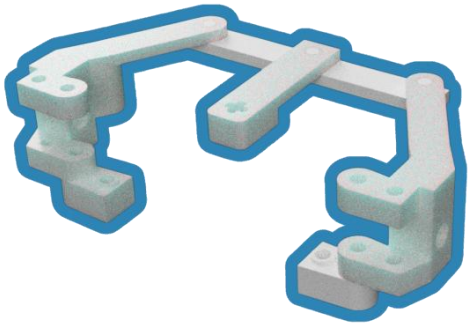
Step-Down Case



The step-down comes with a lot of pin under it. This was caused by soldering. This case was made to make sure that the bottom stays flat and avoid current leakage.

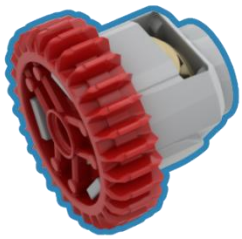
Steering Module

It consists of three main components: the servo link, main connector, and wheel bracket. When our mechanic designed this part, careful consideration was given to the steering angle. We applied the Ackermann Steering Principle, a system developed to achieve tighter and more efficient turns. With this method, the inner wheel turns at a sharper angle than the outer wheel—similar to the steering system used in Formula 1 cars. We also ensured that, during turns, the wheels would not collide with the robot's body.



3.3 LEGO Technic Parts

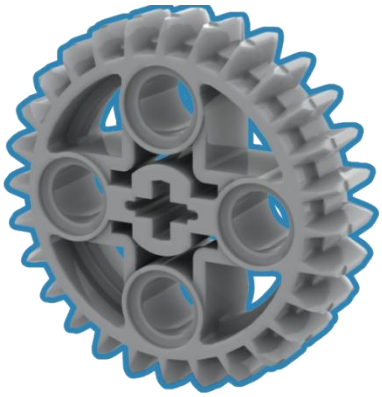
Differential



This differential includes a 28-tooth double bevel gear, identical to the one used with our motor. Inside, it contains five small gears that allow the opposite wheel to rotate freely. This mechanism ensures that both wheels rotate at the correct speed, even when the robot is turning.

Gear

We selected a 28-tooth double bevel gear to prevent the robot from losing speed or torque. It is made of ABS plastic, providing durability, and matches the size of the differential.

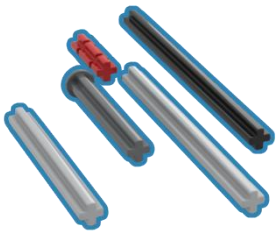


Pin

These pins are used to secure the servo to the robot. They act as a lock, ensuring the servo stays firmly in place.

Axle

Axles connect the wheels to the robot. We designed a 5.1 mm hole for the axle, allowing it to sit securely while still rotating freely.



Wheel: 43.2 * 22

The wheel is a crucial component of our robot and comes in various sizes and shapes. We selected this size for its ability to maintain grip on the surface during high-speed turns. Its 43.2 diameter also makes the robot easy to control.

Half Bush

This component creates space between the wheel and the robot, ensuring that the wheels can rotate freely without being obstructed during operation.

3.4 Additional

Screws

Screws come in various sizes. We designed our robot primarily around M3 screws, as they are the most common in robotics. In some cases, such as mounting the Raspberry Pi or LiDAR, M2.5 screws are required. For our build, we mainly use countersunk screws. To accommodate them, we chamfered the M3 screw holes by 1.8 mm, creating the perfect countersunk fit that allows the screws to sit flush and remain neatly hidden.

M3 Countersunk Screws

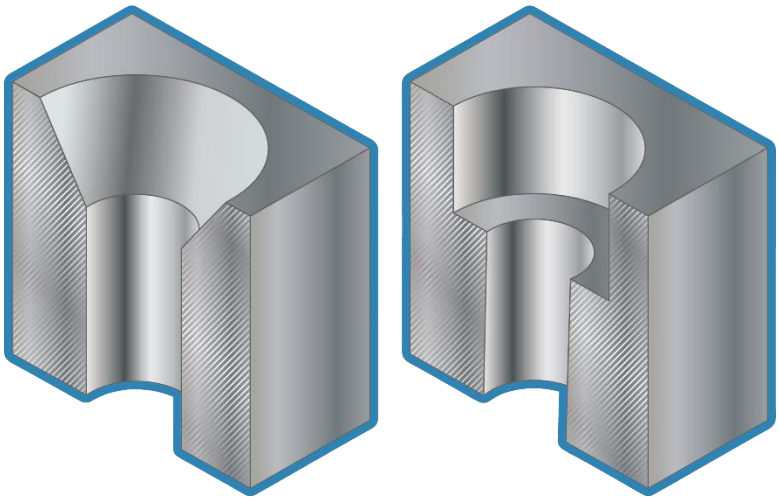
M2.5 Screws

PCB Support Post

The support posts connect the first and second floors of the robot. They enhance stability, ensuring that the LiDAR and camera remain steady during operation.

3.5 Design References

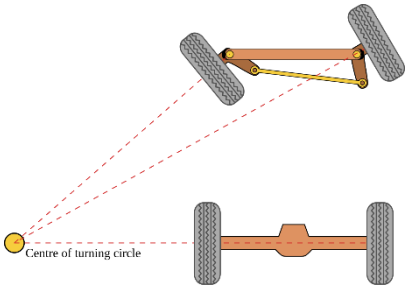
Counterbore Technique



Counterbore and countersunk are two techniques used to make screws sit flush with a surface. A countersunk hole has an angled top, allowing cone-shaped screws to fit neatly inside and create a smooth finish, while a counterbore hole has a flat, widened top so cylindrical screw heads can sit below the surface. In our robot, we use these methods to merge the screw into the structure by chamfering 1.8 mm into the screw hole or creating a 4 mm wide hole with a 2 mm depth. This not only saves space but also gives the robot a cleaner and more professional look.

Ackerman Steering

Ackermann steering is a method where the inner wheel turns sharper than the outer wheel when the robot turns. This matches the natural path of the wheels, preventing them from sliding against the ground. It makes the robot easier to control, gives it a tighter turning radius, and reduces friction—just like in real cars and F1 racing.



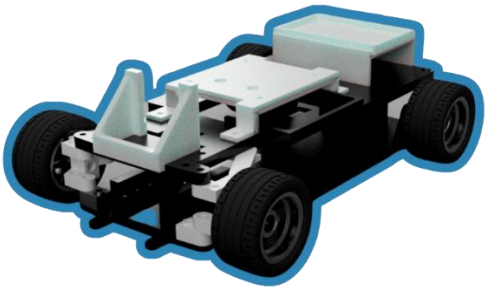
3.6 Printing Process

To print this robot, we first use a slicer to generate the **G-code for the 3D printer. In our case, we use **Bambu Studio**. A slicer lets us set up a **print profile**, which defines whether the part will be strong, lightweight, or have a smooth surface. Once the profile is set, we check for **overhangs** — parts of the model that **"float"** without support. While printers can handle small overhangs, larger ones require supports. Supports are temporary structures that hold the object during printing and can be removed afterward. There are two main types: **normal supports** and **tree supports**, with tree supports being the most popular because they use less filament and print faster. After preparing everything, we start the print.

Here's a timelapse of the printing process for some of our parts:

3.7 Full Chassis Assembly

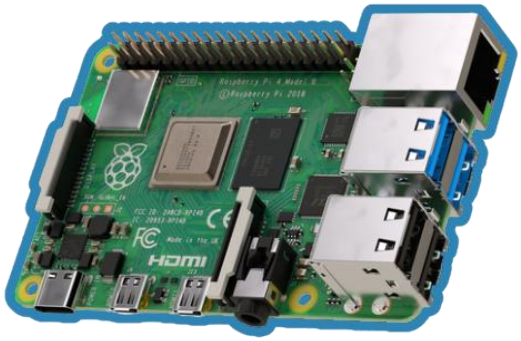
This is what our robot looks like with all the chassis part combined.



4. Hardware Information

4.1 Controller

Main controller: Raspberry Pi 4 Model B from Raspberry Pi



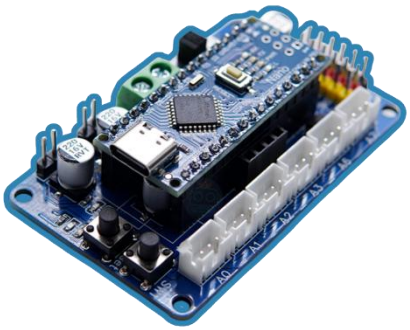
The Raspberry Pi 4 Model B was the first component we selected, and it serves as the most crucial part of the robot, acting as the brain of the system. Without it, the robot cannot process information or make decisions. The Raspberry Pi is widely used as a controller due to its versatility and accessibility, functioning like a mini computer with 4 USB ports, 40 GPIO pins, and dedicated slots for a camera and monitor. It can be powered via a USB-C connector with an operating voltage of 5 volts. Since we chose ROS2 as the robot's operating system, the Raspberry Pi 4 was the ideal choice to handle the required computation and communication tasks. And it can be connected with Wi-Fi or LAN cable for communicating with the programming computer.

Specification

Feature	Details
Processor	Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.8GHz
Memory Options	1GB, 2GB, 4GB or 8GB LPDDR4-3200 SDRAM
Wireless	2.4 GHz and 5.0 GHz IEEE 802.11ac, Bluetooth 5.0, BLE
Ethernet	Gigabit Ethernet
USB Ports	2 × USB 3.0, 2 × USB 2.0

Feature	Details
GPIO	40-pin header (fully backwards compatible)
Display Output	2 × micro-HDMI® (up to 4kp60), 2-lane MIPI DSI
Camera Interface	2-lane MIPI CSI port
Audio/Video	4-pole stereo audio and composite video port
Video Support	H.265 (4kp60 decode), H.264 (1080p60 decode, 1080p30 encode)
Graphics	OpenGL ES 3.1, Vulkan 1.0
Storage	Micro-SD card slot (OS and data storage)
Power Input	5V DC via USB-C (min 3A), 5V DC via GPIO (min 3A), Power over Ethernet (PoE with HAT)
Operating Temperature	0 – 50 °C ambient

Extension Board: Arduino Nano from Princebot



Using a Raspberry Pi comes with some limitations. Its GPIO pins only provide digital signals, meaning it cannot directly control motors or servos, and it also lacks built-in support for analog readings. To solve this, we use an extension board. The Arduino Nano is a small, fast, and flexible microcontroller, making it an ideal choice. By pairing the Nano with a custom PCB, we can easily drive motors, control servos, and read data from analog sensors. The PCB supports up to 10 volts of battery input, which makes it the perfect fit for our robot components.

Specification

Feature	Details
Processor	Atmega328 (Arduino Nano)
Input Voltage	6–10 V (with reverse polarity protection)
Motor Driver	2 × modules (6–10 V, 1.5 A each)
Sensor Ports	6 × Analog (JST connectors)
Servo Ports	6 × Digital (servo compatible)
Buzzer	1 × Speaker
Switches	1 × Switch (D12), 1 × Reset switch
Dimensions	64 mm (W) × 45 mm (H)

4.2 Sensors and visibility

LiDAR: RPLIDAR C1 from SLAMTEC

This is a LiDAR, a device that emitted light toward the surface and then measure the time it take to bounce back. Most of the car uses it for parking. This part plays a crucial role in this competition, by measuring the distance between the inner and outer wall then divide by 2 so that the robot would stay in the perfect middle. And it also plays a significant role in parking sequence due to it's ability to view 360 degree. We can connect the LiDAR with the USB port on Raspberry Pi, making it very user friendly.

Specification

Feature	Details
Distance Range	White object: 0.05–12 m (70% reflectivity) Black object: 0.05–6 m (10% reflectivity)
Sample Rate	5 kHz
Scanning Frequency	8–12 Hz (10 Hz typical)
Angular Resolution	0.72° (typical)
Scan Field Flatness	0°–1.5°
Communication Interface	TTL UART
Communication Speed	460800 baud
Accuracy	±30 mm
Resolution	15 mm

Feature	Details
Degree of Protection	IP54
Ambient Light Limit	40,000 lux
Weight	110 g
Working Temperature	-10 °C to +40 °C
Storage Temperature	-20 °C to +60 °C

Camera: HBV-1716WA

This component is very important for avoiding obstacle. It can detect red and green obstacle from distance to avoid crashing into it. It can co-operate with the LiDAR to ensure the maximum efficiency when the robot turns to avoid the block. This camera doesn't have it's own micro-controller meaning that it can achive it's full **frame per sec** performance. And it can also be connect to Raspberry Pi through USB port.

Specification

Feature	Details
Model	HBV-1716WA
Highest Resolution	1920 × 1080
Field of View	140°
Chip	OV2710 (1/2.7")
Frame Per Second	30 FPS
Weight	Approx. 32 g / 1.1 oz
Outer Size	Approx. 38 × 38 × 25 mm / 1.5 × 1.5 × 1.0 in
Inner Size	Approx. 32 × 32 × 25 mm / 1.3 × 1.3 × 1.0 in

Gyro Sensor: SEN-0253 from DFRobot

Gyro sensor is a component that enables a robot to determine its orientation and turn in the appropriate direction. The gyro sensors detect 3 axis: yaw, pitch, and roll. We chose this gyro sensor specifically because of how effective it is. It collaborate with LiDAR and camera to ensure that the robot would not steer of it's course when it's turning to avoid the obstacle. And it also helps when the robot initiate the parking. It helps the robot make sure that it is turning toward the right direction. The gyro uses I2C wiring. The I2C features 4 different wire: SDA, SCL, 5 volts, and Ground. The wires go onto the GPIO pins on Raspberry Pi.

Specification

Feature	Details
Operating Voltage	3.3 V – 5 V DC
Operating Current	5 mA
Interface	Gravity-I2C
Operating Temperature	-40 ~ +80 °C
Product Dimension	32 × 27 mm / 1.26 × 1.06”

BNO055 Accelerometer

Feature	Details
Acceleration Ranges	±2 g / ±4 g / ±8 g / ±16 g
Low-pass Filter Bandwidth	1 kHz ~ <8 Hz

Feature	Details
Operation Modes	Normal, Suspend, Low Power, Standby, Deep Suspend
Interrupt Control	Motion-triggered interrupt signal

BNO055 Gyroscope

Feature	Details
Ranges	±125°/s ~ 2000°/s (switchable)
Low-pass Filter Bandwidth	523 Hz ~ 12 Hz
Operation Modes	Normal, Fast Power Up, Deep Suspend, Suspend, Advanced Power Save
Interrupt Control	Motion-triggered interrupt signal

BNO055 Geomagnetic

Feature	Details
Magnetic Field Range	±1300 µT (x-, y-axis), ±2500 µT (z-axis)
Magnetic Field Resolution	~0.3
Operating Modes	Low Power, Regular, Enhanced Regular, High Accuracy
Power Modes	Normal, Sleep, Suspend, Force

BMP280 Digital Pressure Sensor

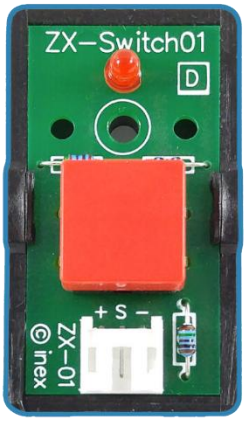
Feature	Details
Pressure Range	300 ~ 1100 hPa
Relative Accuracy	±0.12 hPa (~ ±1 m)
Absolute Accuracy	±1 hPa (~ ±8.33 m)
Temperature Range	0 °C ~ 65 °C
Temperature Resolution	0.01 °C

Reflected Light Sensors: TC-01 from Princebot

Reflected Light Sensors come in various color. Each color can be used for different surface color. This track has 3 different colors: red, blue, and white. We tried using a single red or blue light, but it doesn't work. So at first we come up with using 2 sensors at the same time. And in the end, we settle on using a single **White light** color. White has an ability to amazingly seperate the red, blue, and white. It's purpose is to count and detect lines. This sensor uses **JST** head as a connector. Since our extension board also uses JSTs, we can connect the sensor straight to it.

Touch Sensor: ZX-Switch from INEX

This button gives us an easier way to start the robot. Since the controller board doesn't come with switches. So, we found this button that could be attached to the frame outside the board using bolt. It gives us the advantages, when we start the robot we can turn on the switch and then keep the robot on the ground so that it would have time to reset it's gyro and other components. Then we press the switch to start and stop the robot. This sensor required a digital reading. The input would be 0 and 1. 1 being pressed and 0 being nothing. We wired this straight into Raspberry Pi's GPIO pins.



4.3 Driving and Steering

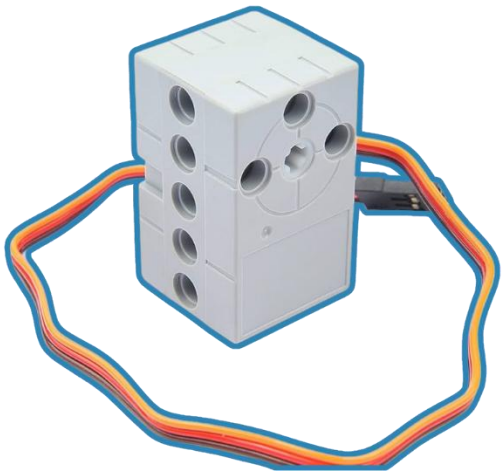
Motor: GM25-370 from Chihai Motor

This motor is usually used for a sumo robot due to it's speed and power. We then uses this with our robot because it can help us achive the speed we need to win this competition. And since our robot is very heavy due to the various components, this motor comes in real handy. We uses this motor with extension board to control the speed and the power it uses. The rotating axis is connected to a gear which then transfer the spinning to a differential then the wheels.

Specification

Feature	Details
Voltage	DC 7.4-12.0 V
Power	26 W
Unloaded Speed	1000 RPM @ 0.60 A
Maximum Power Point (1)	Load: 2.3 kg·cm (0.22 N·m) Speed: 780 RPM Power: 17 W Current: 2.5 A
Maximum Power Point (2)	Load: 5.0 kg·cm (0.5 N·m) Speed: 500 RPM Power: 25.6 W Current: 4.5 A
Plugging Parameter	2.3 kg·cm (0.21 N·m) @ 16 A
Gear Ratio	20:1

Servo: Geekservo 360 2KG from ELECFREAKS



We use this servo to steer the robot. This servo is compatible with LEGO, making it easy and convenient to build the robot by just putting studs in the hole on the side. We like how you can connect two axles to the dual outputs on this servo so you can power two wheels or gears or mount the servo securely inside articulated limbs and other contraptions. Additionally, the gears inside these servos will **slip** when the blocking load is too high instead of jamming, helping avoid damage to our servo and board. It can be connected to servo pin on the extension board.

Specification

Feature	Details
Rotation	360°
Start Voltage	2.5 V
Working Voltage	3.3 – 6.0 V
Rated Voltage	4.8 V
Rated Current	70 mA
Maximum Torque	1.6 ± 0.2 kg·cm (at 4.8 V)
Angle Rotation Speed	60° / 0.14 s

4.4 Power management

Battery: Firefox 3-cell 11.1v 1300mah from Firefox

The 11.1V battery enhances the robot's performance by providing higher voltage, which reduces current draw and minimizes power losses during our sessions. This battery ensures stable and continuous functionality without significant voltage drops, even under high current loads, a critical factor during intense practice sessions and competitions. The battery has 1300 mah which is enough to power all of our components. We also uses 2 step down for our robot which will be mentioned later on.

Specification

Feature	Details
Cells	3
Voltage	11.1 V
Capacity	1300 mAh 20C
Charging Current	Up to 5 × capacity (5C)
Connectors	Dean type, easily disconnectable

Stepdown: HW316E V6.0.1 and LM2596

Stepdown or Buck converter is very essential to power both Raspberry Pi and motor driver. Step-downs act like a power limiter for the battery. In our robot, we uses 2 different step-down due to the size.

HW316E V6.0.1

This is the stepdown we use to power our extension board. We set the limit voltage to 10 volts to achive the perfect power for our motor. This stepdown also comes with a display LED to indicate the battery level. It can show both input and the output limit we setted.

LM2596

We use this step down for Raspberry Pi. The output limit is set to 5 volts otherwise the Raspberry Pi would get short-circuit. It doesn't come with a screen so we need to use multimeter to measure the output although it isn't necessary since we can use the screen on another step down. The wire must be solder into the circuit. On the output end we use a Type-C wire and then connect to the controller.

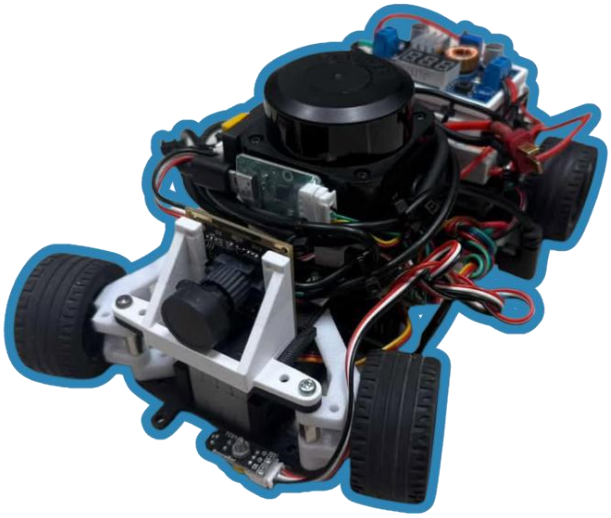
On-Off Switch

This switch is for cutting the power from the battery to the robot. The regulation states that before starting the robot, the power must be cut off. That's when this switch came in. To use this switch, we solder red wire (Positive pole) to the switch on 1 side for input. Then another solder red wire for output on the opposite side. You can put the black wire (Negative pole) straight into the step down. When the switch is turned on, the power from the battery will direct into stepdowns and then the Raspberry Pi and Driver board.

Wire Splitter

To power both the Raspberry Pi and Arduino Nano, we use a wire splitter to divide the battery connection into two wires per pole. For example, the positive pole is split into two separate wires, each directed to the positive input of its own step-down converter. The same is done for the negative pole. This ensures that both boards receive stable and isolated power from the same battery source.

Full Robot Assembly



This is the real photo of our robot. It consists of every component we listed.

5.ROS 2 Humble Explanation

In this competition, there are multiple approaches we could use, such as machine learning, ROS, or writing plain code directly on an Arduino board. Since our LiDAR only supports Raspberry Pi, we chose ROS because of its strong ability to handle navigation and decision-making.

There are two generations of ROS:

- **ROS 1 (2010–2025)** The original Robot Operating System. Its final release, ROS Noetic Ninjemys (2020), will be supported until 2025. After that, it will no longer receive updates, making it unsuitable for long-term projects or competitions.
- **ROS 2 (2017–present)**

The next generation of ROS, designed to overcome ROS 1's limitations. It adds real-time support, improved communication via DDS middleware, stronger security, and better support for multi-robot systems. ROS 2 is still actively developed with multiple distributions.

We selected ROS 2 Humble because it is stable, reliable, and has a large support community—perfect for our competition project.

What is ROS 2 Humble?

ROS 2 (Robot Operating System 2) is an open-source framework for building and running robot applications. It provides libraries, tools, and conventions to make developing complex robotic systems easier.

The Humble Hawksbill release, commonly called ROS 2 Humble, is a Long-Term Support (LTS) version released in May 2022, with updates and security patches maintained until May 2027.

Because it is stable and reliable, ROS 2 Humble is widely used in education, research, and industry—making it an ideal choice for long-term projects such as WRO competitions.

Why ROS 2 Humble?

The Future Engineers category challenges the competitors to design autonomous robots that can navigate, sense, and make decisions. ROS 2 is a strong fit because:

- Modular design – Breaks robot software into nodes that can handle tasks like motor control, camera vision, and sensor fusion separately. -Communication system – Uses topics, services, and actions for real-time data exchange between nodes, which is essential for autonomous navigation.
- Hardware flexibility – Works on Raspberry Pi, NVIDIA Jetson, or PCs, and supports many sensors and actuators.
- Simulation tools – Integrates with Gazebo and RViz to test algorithms before deploying on the physical robot.
- Community & libraries – Many open-source packages (SLAM, navigation, vision, etc.) are available, which saves development time.

Key Concepts of ROS 2 Humble

- Node – A single program that performs a specific task (e.g., controlling a motor, processing camera images).
- Topic – A channel where nodes publish and subscribe to share data (e.g., /camera/image_raw).
- Service – A request/response interaction (e.g., “start motor” or “get distance”).
- Action – Handles longer tasks that can provide feedback and be canceled (e.g., navigation goals).
- Package – A collection of nodes, configuration files, and launch files grouped as one project module.

Advantages of Using Humble in Competitions

- Stability (LTS) – Reliable platform during multi-year competition cycles.
- Wide hardware support – Works well with microcontrollers (via ROS 2 Micro-ROS), Raspberry Pi, and AI boards.
- Future-proof – We can reuse the same codebase and extend it each season.
- Industry alignment – Skills learned transfer directly to robotics and automation fields.

Typical ROS 2 Workflow for a WRO Robot

- Perception – Use a camera or sensors → ROS 2 node processes the data.
- Planning – Navigation stack or custom algorithm decides how to move.
- Control – Motor controller node executes wheel commands.
- Testing – Upload the written program to the robot.

Conclusion

ROS 2 Humble provides a structured, modular, and industry-standard software framework for building autonomous robots. For WRO Future Engineers, it helps teams develop reliable robots that can sense, think, and act both in simulation and in real matches.