MANCHESTER 1824
The University of Manchester

Final Design Review

MSc Robotics AERO62520 Robotic Systems Design Project

Year of submission
2025

Group Number
Group 5

School of Engineering

Contents

1 Introduction

1.1 Summary of the problem statement and objectives

This document serves as the final design review for Team 5's project, which aims to

> **"Develop a robot which can autonomously retrieve coloured objects from the environment and place them in matching storage bins located at the starting point."**

The system integrates a Leo Rover with a PincherX 150 manipulator, using advanced sensors and control mechanisms to operate efficiently and safely within a pre-defined time limit.

The primary objectives align with the Design Requirements Analysis (DRA) outlined in Appendix A of the Preliminary Design Review (PDR). Key goals include the implementation of autonomous navigation, object detection and classification based on computer vision, and robust collision avoidance strategies to interact with static obstacles. In addition, the system will incorporate critical safety features, such as an emergency stop function, to ensure operational reliability.

This summary reflects Team 5's systematic approach to transform theoretical frameworks into a deployable robotic solution, focusing on alignment with stakeholder requirements and iterative validation through the PDR process.

1.2 Summary of how the team addressed feedback from the Preliminary Design Review

- **System Architecture:**
    1. The system block diagram was repositioned to appear before descriptions, improving readability.
- **Electrical Design:**
    1. The Intel NUC was formally added to the list of electrical components.
    2. The power budget analysis has been updated to address battery limitations identified in the PDR. An external power supply is now used for testing and stationary operations, reducing battery strain and ensuring stability.
- **Mechanical Design:**
    1. Annotations were added to the 3D CAD model.
    2. The 3D CAD model was modified according to our new operation process.
    3. The team decided to place the lidar at the front of the robot to allow rapid response during navigation. This ensures efficient and timely obstacle avoidance, particularly as the robot primarily moves forward. After navigation simulation, the lidar fully meets the needs of this project without interference from other sensors or components. It successfully detects obstacles and walls and determines accurate distances.

4. Acrylic and PLA were selected for manufacturing. Acrylic has good strength performance and durability, making it a long-lasting material. PLA was chosen for the support bin because it is biodegradable and made from renewable resources. It is also easy to manufacture via 3D printing. After strength analysis, PLA was confirmed to be suitable and strong enough to support the payload sled.

- **Software Design:**

1. The flowcharts were refined to improve clarity and better reflect the updated system architecture.
2. In addition to the original RQT graph, new modular RQT graphs were introduced for individual functional components, improving readability and making it easier to analyze interactions between specific modules.
3. The GitHub repository was updated with corresponding changes in structure and documentation. Further refinements, including code updates, will be added as development progresses.

- **Design Requirements Analysis:**

1. Requirements have now been explicitly cross-referenced with the Design Requirements Analysis (DRA) for improved traceability.
2. Additional quantifiable evidence has been incorporated where feasible to support the fulfillment of each requirement.
3. Test descriptions have been expanded to specify the number of iterations required to achieve statistical confidence, including clear success criteria (e.g., passing the test a certain number of times).

- **Project Plan:**

1. Milestones and deliverables have been explicitly incorporated to provide clear progress tracking.
2. Workload allocation has been elaborated, detailing specific responsibilities for each team member to ensure balanced and efficient task distribution.

1.3 Summary of any modifications made since the Preliminary Design Review

Since the Preliminary Design Review, the team has implemented several critical modifications to improve overall system performance, including mechanical design, perception, manipulation, and navigation strategies. The mechanical design was significantly revised with the removal of the storage bin. Instead of temporarily storing objects, the robot now continuously holds the object until it reaches the goal position, where it directly deposits it into the designated box. This change simplifies the manipulation process and reduces unnecessary mechanical complexity.

Further refinements were made to the payload sled design, with Acrylic chosen as the primary material due to its superior strength and durability. Given the need to support the weight of the manipulator and additional sensors, the base and top plates were carefully selected to provide sufficient

structural integrity. These plates were laser cut and tapped to ensure precision in assembly. Additionally, the support I-beams were manufactured using PLA via 3D printing, a sustainable material that provides adequate strength while maintaining lightweight properties. These modifications improve both the manufacturability and structural robustness of the system, ensuring that it can reliably support the robotic manipulator and other onboard components.

To address power limitations, the team implemented an external power supply for testing and stationary use. This setup directly powers high-demand components such as the Intel NUC and PincherX 150 Manipulator, reducing strain on the battery and ensuring system stability during prolonged operations.

The object detection node has been enhanced to enable real-time color detection using OpenCV and depth data extraction from the Intel RealSense Camera D435i. To improve measurement accuracy, a smoothing filter has been implemented to stabilize depth readings. Upon processing, the node publishes the detected object's 3D position to a ROS2 topic, facilitating seamless integration of the camera with the robotic manipulator and the Leo Rover platform. Color detection plays a pivotal role in this system, and the algorithm has been refined for higher precision while effectively filtering out noise. These improvements contribute to more robust system integration, enhanced performance, and increased efficiency in robotic operations.

For manipulator control, the system was optimized for precise and reliable object handling. The manipulator's basic functionality is managed through Python launch files, which process geometric pose messages via the joint state topic and adjust its position based on the detected object's coordinates. The object detection node provides real-time 3D position data, which is converted into joint commands using an inverse kinematics (IK) solver to ensure accurate grasping. Additionally, the grasping strategy was refined to maintain a secure hold until the object is successfully placed. These improvements increase the manipulator's precision, adaptability, and overall efficiency, making object retrieval more robust and autonomous.

For navigation and exploration, the team implemented SLAM Toolbox to construct a real-time map of dynamic environments while simultaneously performing localization. To enhance position accuracy and smoothness, an Extended Kalman Filter (EKF) was integrated to fuse data from odometry, IMU, and LiDAR. For path planning, the A* algorithm was adopted for global path planning due to its efficiency in finding the shortest path, while the Timed Elastic Band (TEB) method was selected for local path planning to ensure real-time trajectory optimization and dynamic obstacle avoidance. Additionally, Frontier Exploration was used to identify the boundaries of unexplored regions, guiding the robot to these areas for comprehensive environmental exploration. These improvements significantly boost the system's ability to navigate unknown spaces autonomously and efficiently.

In summary, the team's efforts have led to substantial improvements in both hardware and software, resulting in a more reliable and efficient robotic system. These modifications enhance the robot's ability to handle objects, navigate environments, and perform tasks autonomously, making the system better equipped for reliable and precise object retrieval in dynamic indoor environments.

2 Sustainability Checklist

This 10-point checklist is derived from a draft of the BS8622 Guide to Robot Sustainability [1].

1. **Materials**
   This project incorporates various materials, including metals for the LeoRover and manipulator, plastic components for the payload sled, and electronic parts for sensors and cameras. In today's world, sustainability is a key consideration in technology. Since the payload sled is the only component we can fully control, we use PLA and acrylic, both recyclable plastics, to reduce environmental impact.

2. **Software**
   The object detection, navigation, mapping, and grasping nodes are designed to achieve their goals with minimal software overhead. We eliminate unused communications and reduce the computational load to enhance efficiency and sustainability.

3. **Energy**
   The robot is powered by a high-capacity Li-Ion 3S battery, which is rechargeable and supports hundreds of charge cycles, contributing to energy efficiency and sustainability.

4. **Waste**
   Essentially, no waste is generated during the operation of the robot.

5. **Emissions**
   The direct emissions from this project are minimal. Additionally, the carbon footprint of the materials used in manufacturing is low. PLA, a bio-based plastic, is derived from renewable resources, emitting few greenhouse gases during production.

6. **Communications**
   During the robot design phase, we carefully assessed data transmission requirements, ensuring that only essential data is transmitted. Removing unnecessary data transmission improves efficiency, power consumption, and storage management, ultimately enhancing sustainability.

7. **Modularity**
   Our design emphasizes modularity, with tasks divided into separate modules, such as mapping and navigation, object detection, and grasping. This approach simplifies troubleshooting and maintenance, making it easier to fix or upgrade individual parts in the future.

8. **Location/Placement**
   Our robot operates in a lab environment, meaning no transportation is required. If needed, transportation can be done by walking or public transport, minimizing the project's carbon footprint.

9. **Maintenance**
   The team will check software and hardware weekly during lab sessions, including component integration, sensor performance, and robot movement testing. Each module must complete

a simulation before real operation. We use Rviz for manipulator simulation, run the test node for object detection, and use Gazebo for robot mapping and navigation simulation.

10. **Repurposing**
    Components such as LeoRover, manipulator, RPLidar, depth camera, and NUC can be reused in future projects. At the end of our project, the team will disassemble the robot. The payload sled is made of recyclable plastics, ensuring sustainability.

3 Cyber Security Considerations

To ensure the safe and reliable operation of an autonomous robot retrieving colored objects and placing them in matching storage bins, the following cyber security measures are considered:

Data Protection: Encrypt communication between the robot, sensors, and control systems to prevent unauthorized access.

Access Control: Use authentication methods like passwords or multi-factor authentication (MFA) to restrict unauthorized control.

Secure Communication: Protect data exchanges with VPNs, firewalls, and encrypted channels to prevent cyber threats.

Regular Security Checks: Perform periodic security audits and updates to identify and fix vulnerabilities.

Software Updates: Keep the robot's software and firmware up to date to address security risks.

Tamper Detection: Implement monitoring systems to detect unauthorized physical or network access attempts.

By following these measures, the robot can function securely and effectively while minimizing cyber risks.

4 System

4.1 System block-diagram

The system block-diagram below illustrates the interconnection of all the components in the robotic system. Each component is organized into specific categories: computing, sensors, peripherals and power. The color-coded legend in the top-left corner provides a visual guide to distinguish these categories and identify the functions of each component.

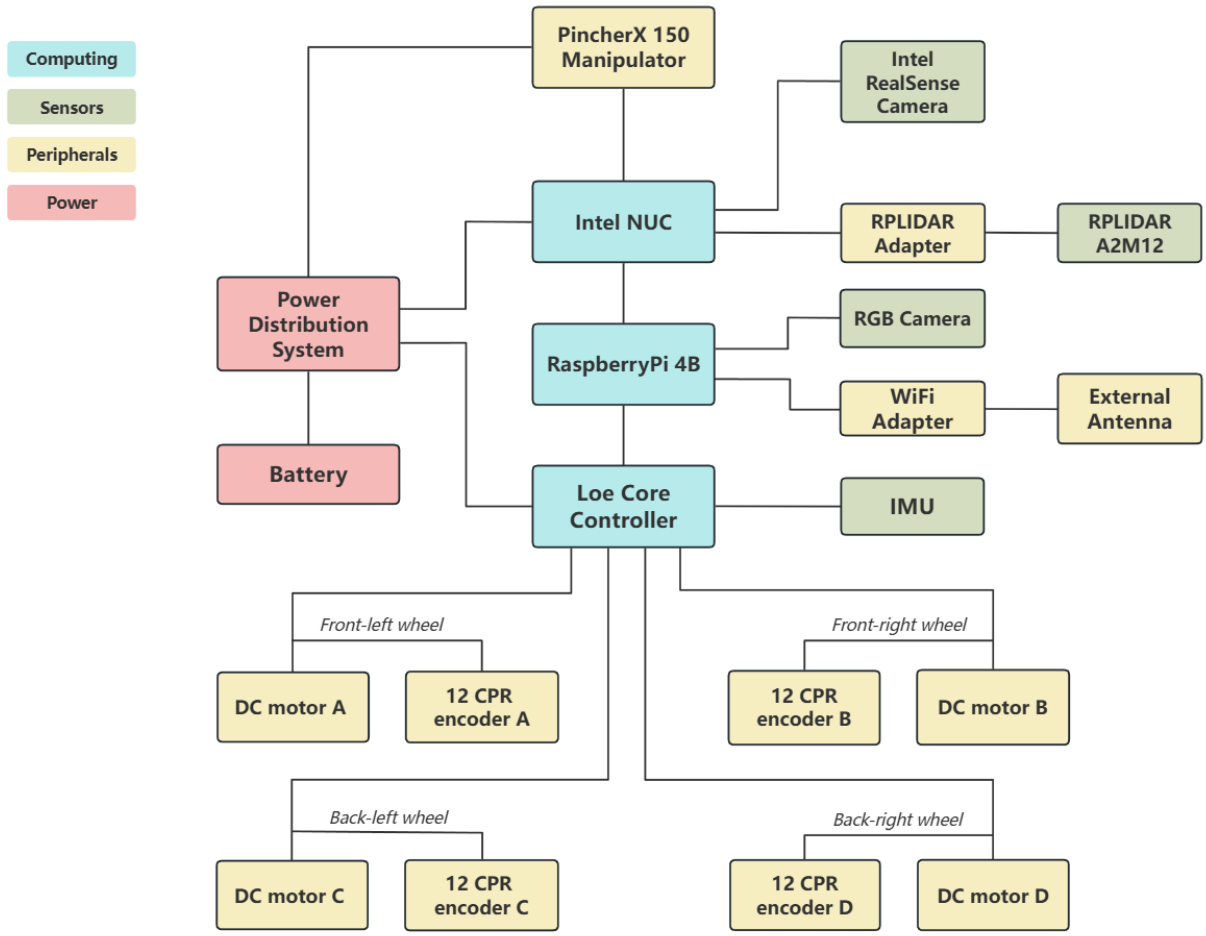


Fig. 1. System block-diagram

4.2 Components

- **Battery**: Provides the primary energy source required for the entire robotic system to operate autonomously, ensuring uninterrupted functionality during tasks.
- **Power Distribution System (PDS)**: Regulates and distributes power from the battery to various subsystems with precision, ensuring each component receives the necessary voltage and current for stable operation.
- **LEO Core Controller**: Serves as the central hub for motor control and sensor integration, enabling real-time feedback from the motors and encoders for precise movement and navigation.
- **DC Motors with Encoders**: Propel the robot's wheels while providing positional feedback, allowing accurate control of speed and direction.
- **Raspberry Pi 4B**: Functions as a key processing unit for sensor data acquisition and communication, bridging data exchange with the Intel NUC and managing wheel velocity inputs from the LEO Core.

- **Integrated IMU**: Provides real-time information on the robot's orientation, acceleration, and angular velocity, supporting motion control and navigation accuracy.
- **RGB Camera**: Captures high-resolution visual data for object recognition and scene analysis, contributing to navigation and perception capabilities.
- **WiFi Adapter with External Antenna**: Enables wireless connectivity for the robot, facilitating data transmission and remote control during testing and operation.
- **Intel NUC**: Acts as a high-performance computing platform for intensive tasks such as SLAM (Simultaneous Localization and Mapping) and advanced navigation algorithms.
- **RPLIDAR A2M12**: Performs 360-degree environmental scanning to generate precise 2D maps of the surroundings, essential for path planning and obstacle avoidance.
- **RPLIDAR USB Adapter**: Provides a seamless interface between the RPLIDAR and the Intel NUC, ensuring efficient data transfer and compatibility.
- **Intel RealSense Depth Camera**: Captures detailed depth and 3D data of the environment, enhancing the robot's ability to perform object detection and spatial analysis.
- **PincherX 150 Manipulator**: A robotic arm mounted on the LoeRover, capable of executing manipulation tasks such as grabbing and moving objects when detected.

5 Mechanical Design

5.1 CAD model

Figure 2 illustrate 3D models of the LeoRover robot equipped with a RealSense depth camera, LiDAR, NUC and a manipulator. After carefully analyzing the design requirements and ensuring that all sensors can operate without obstruction, we developed a two-platform design:

1. Base plate: Serves as the mounting area for the LiDAR and NUC.

2. Top Plate: The robotic arm, depth camera.

The two plates are connected using six I beams. All components are securely fixed with screws to maintain the alignment and stability of the plates. To optimize mapping and object detection, the depth camera and LiDAR are positioned at the front of the robot. Adequate clearance is provided for all components to facilitate proper wiring and operation.

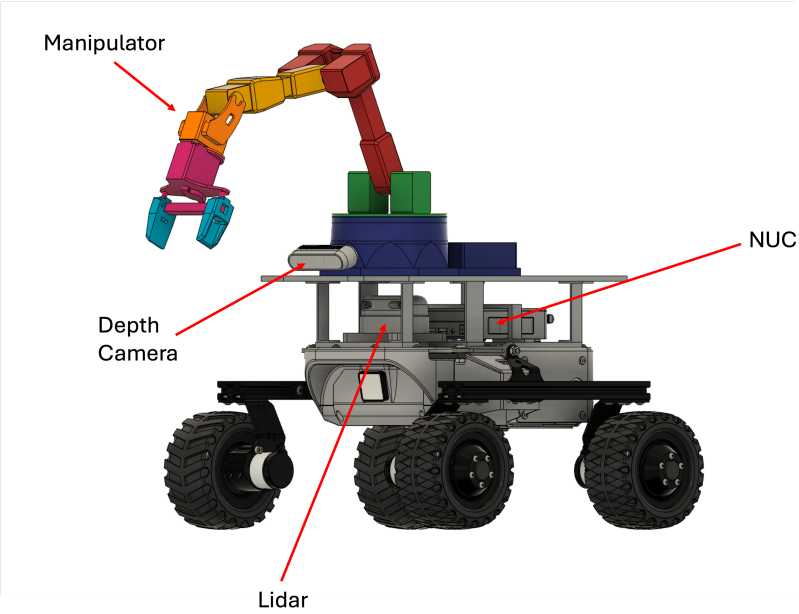


Fig. 2. Isometric View CAD

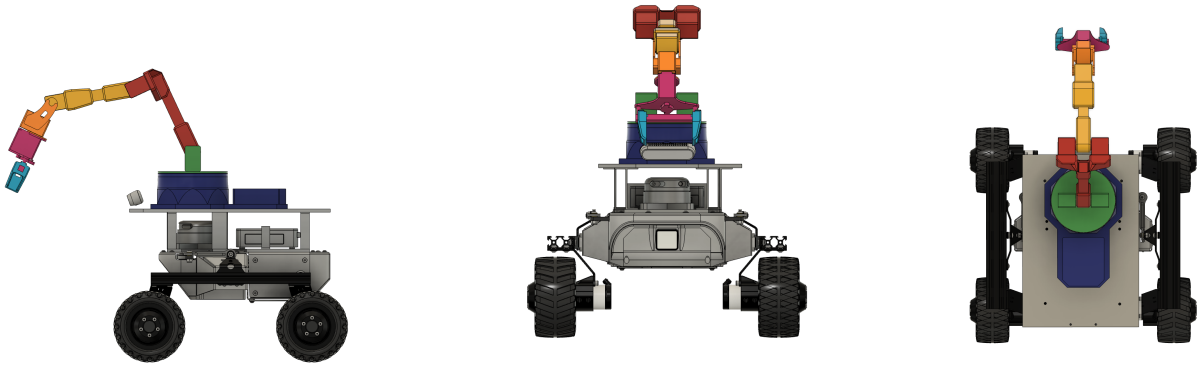


Fig. 3. Three views of CAD model

5.2 Payload Sled Design

The primary challenge was broken down into three key tasks: mapping and navigation, object detection, and manipulation. A critical point of our design approach was to eliminate potential sensor interference, ensuring unobstructed functionality and precise data collection. In order to do this, the lidar and depth camera are located at the very front of the payload where interference from other components can be minimized.

The RPLidar A2M12 is a high-performance 360-degree 2D laser scanner, ideally suited for obstacle detection. Its performance improves when it is positioned closer to the ground, as it operates more effectively at lower levels. Therefore, the LiDAR was strategically mounted on the base plate, bringing it closer to the surface for enhanced detection capabilities. Additionally, positioning it at the front of the robot allows for rapid response during navigation, especially as the robot primarily moves forward, ensuring efficient and timely obstacle avoidance.

The depth camera requires an unobstructed line of sight to accurately identify objects. Mounting it on the top plate not only prevents interference with the LiDAR but also provides a broader field of view. In addition, reducing the distance between the camera and the manipulator minimizes the coordinate transformation, resulting in improved accuracy and reduced error.

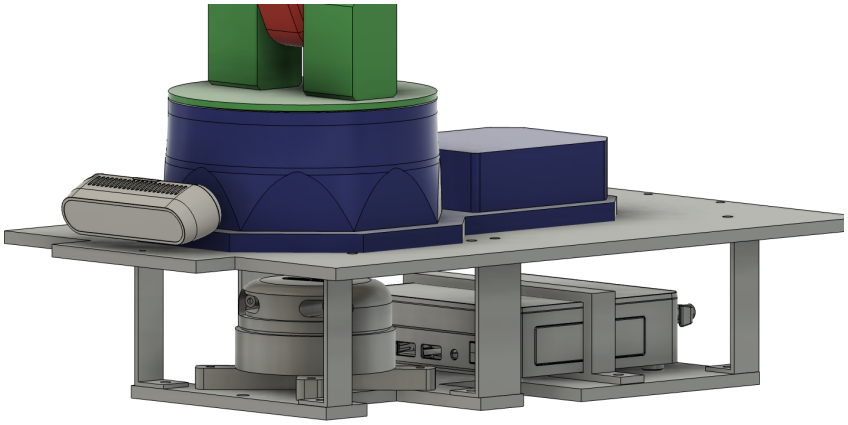


Fig. 4. Payload Sled

5.3 Strength

The components are connected using I-shaped beams, chosen for their structural advantages. The I beam design provides high bending stiffness in the vertical direction, making it ideal for supporting vertical loads. While plastic or acrylic I-beams are less commonly used compared to steel ones, the simulation results shown below demonstrate their suitability for this application.

Since the base plate is fixed to the LeoRover robot, the simulation focuses primarily on the I-beams, which bear the majority of the manipulator's weight. The depth camera, weighing only 72g compared to the manipulator's 770g, exerts negligible force and is therefore excluded from consideration. The simulation results indicate a maximum stress of approximately 0.108 MPa, well below the material's yield strength, ensuring that no strain under these conditions.

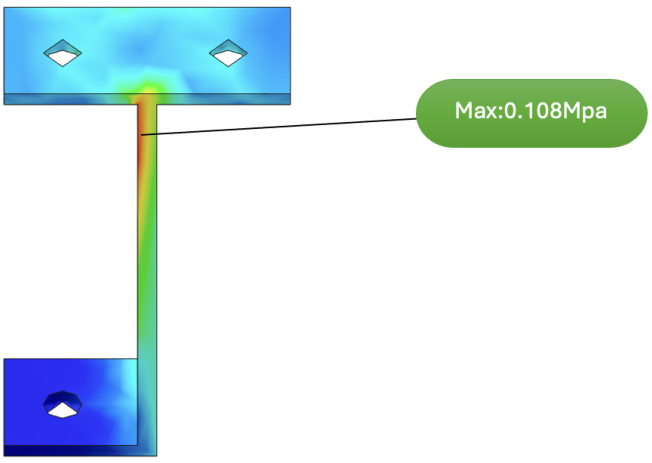


Fig. 5. Payload Sled

5.4 Weight Restriction

The LeoRover has a maximum payload capacity of 5 kg. According to Table 1, the total weight of the components is approximately 3.0 kg. Therefore, the combined weight of the payload sled and the objects temporarily stored in the robot must not exceed 2.0 kg. The weight of the payload is adjustable by modifying the size of the plate and selecting lighter material as needed.

Component	Weight (kg)
Manipulator	0.7
Lidar	0.2
NUC	0.4
Depth camera	0.1
LeoRover	1.6
Total	3.0

Table 1. Component Weight

5.5 Design Files

5.5.1 Overall

The overall design points out all components and the size of the complete robot.

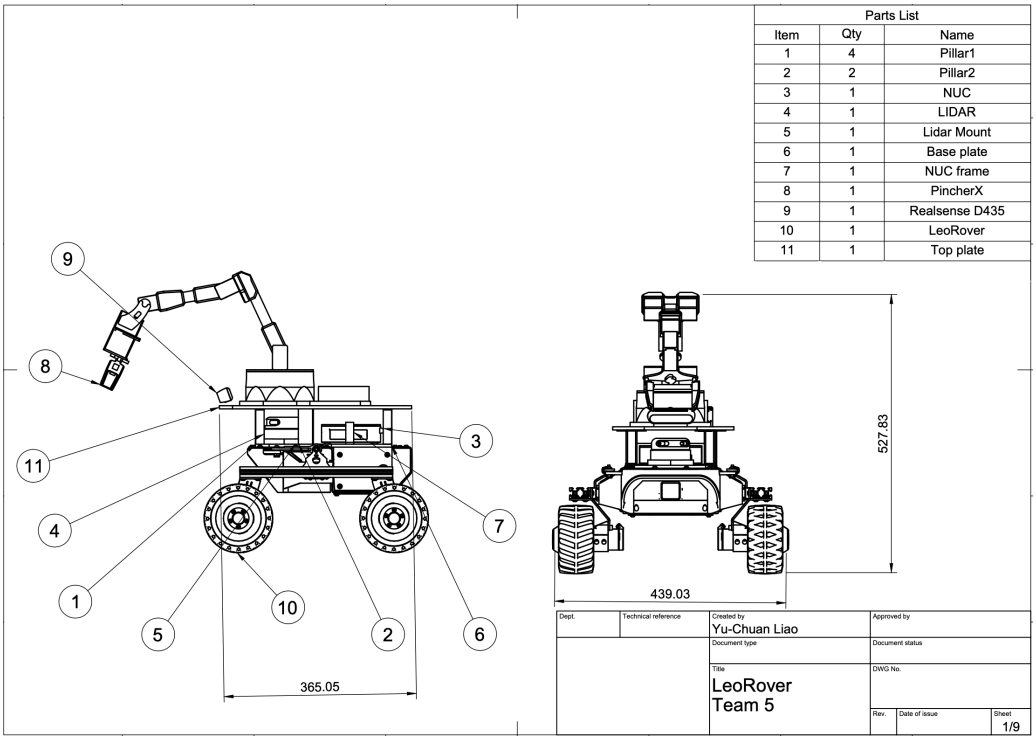
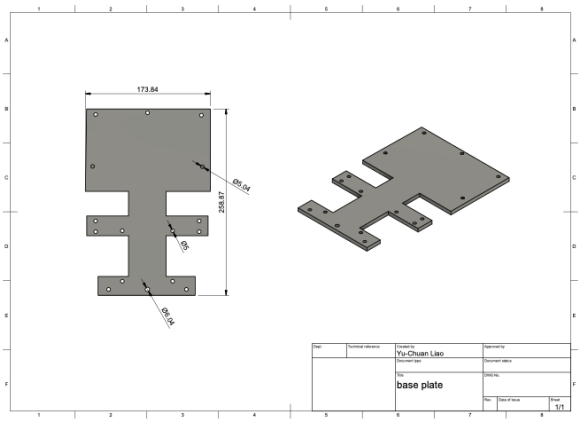


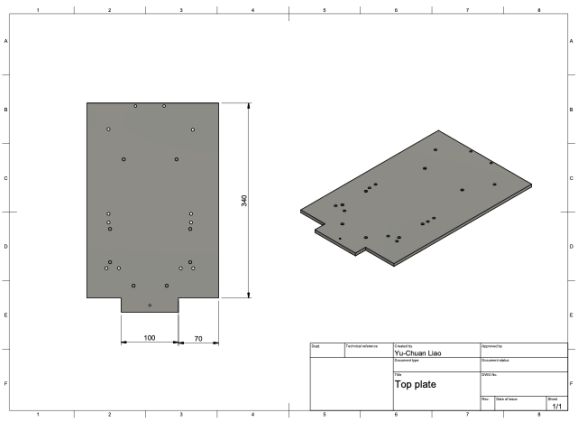
Fig. 6. Overall Drawing

5.5.2 Base plate and Top plate

The base plate and top plate drawing includes the dimensions and area, as well as the sizes of the screws.



(a) Base Plate Drawing



(b) Top Plate Drawing

5.5.3 Pillar

The drawing indicate the width and length of both type of pillars which are used to connect two plates and the size of screws.

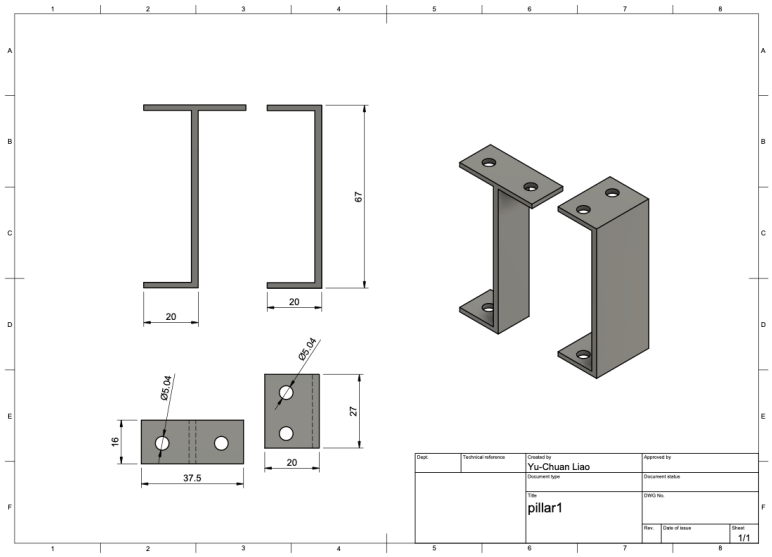


Fig. 8. Pillar Drawing

5.5.4 Lidar Module

The LiDAR is securely mounted on the base plate using an X-shaped bracket with two sets of screws. One set is used to attach the LiDAR to the bracket, while the other set fastens the bracket to the base plate.

5.5.5 NUC Module

The frame is design to securely fix the NUC on the base plate through the force of the screw.

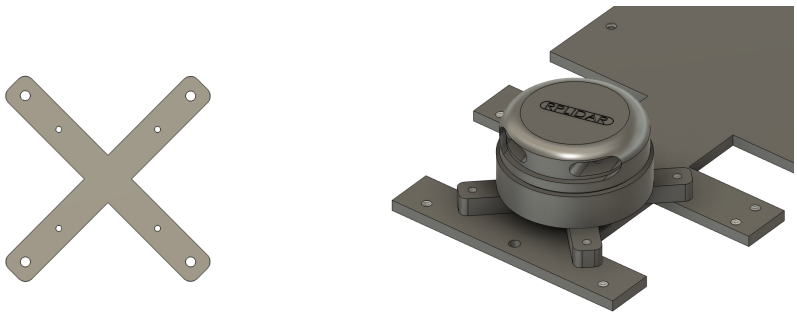


Fig. 9. Lidar Module

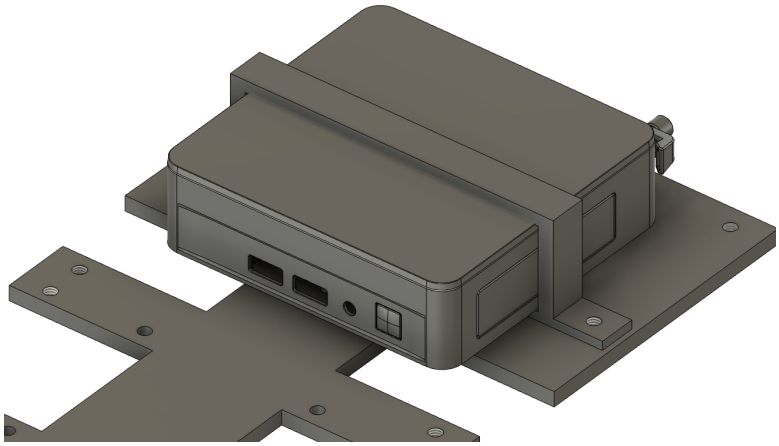


Fig. 10. NUC module

5.5.6 Depth Camera Module

The depth camera is mounted on the front side of the top plate and secured using a component featuring a spherical core structure. This spherical core allows free rotation in all directions, enabling flexible and detailed angle adjustments.

5.6 Manufacturability

Manufacturability is a primary goal of the design. Both laser cutting and 3D printing are excellent options for the payload sled design. Laser cutting offers higher precision, consistency, and faster production. On the other hand, 3D printing takes more time and provides lower precision but is ideal for creating custom prototypes. In conclusion, both methods are suitable for the current requirements, and the final decision may depend on the material chosen after confirmation.

The design is suitable for the goals of mapping and navigation, object detection, and manipulation. As for objective dropping to the final bin, it should control the position it drop down to a 20 centermeters cube with 15 centermeters circular area at the center of the cube.

6 Electrical Design

6.1 Power connection diagram

The power connection diagram illustrates the distribution of electrical power from the battery through the Power Distribution System (PDS) to the various components of the robot. The key details of the power connections are as follows:

- **Battery:** The core energy source of the LeoRover is an 11.1V, 5800mAh Li-ion battery capable of delivering up to 8A of current.
- **Power Distribution System (PDS):** Regulates the 12V input from the battery and distributes power to connected components, including the Intel NUC (12V, 3A), Loe Core Controller (12V, 5A), and PincherX 150 Manipulator (12V, 5A).
- **Intel NUC:** Powered by the PDS with 12V, 10A, it serves as the main processing unit, handling computationally intensive tasks such as SLAM and advanced navigation.
- **Loe Core Controller:** Powered by the Power Distribution System (PDS) with 12V, 5A through its dedicated PWR port. It regulates power to the components, converting 12V to 5V for the Raspberry Pi, 3.3V for the IMU, and 12V for the motors.
- **Raspberry Pi 4B:** Powered via the Loe Core Controller with 5V, 3A through a UART connection, it serves as a central hub for peripherals such as the WiFi Adapter and RGB Camera.
- **PincherX 150 Manipulator:** Receives 12V, 5A directly from the PDS for robotic arm operations, including object manipulation tasks.
- **Intel RealSense Camera:** Receives 5V, 0.7A through a USB connection from the Intel NUC, enabling 3D depth perception and object detection.
- **RPLIDAR A2M12:** Powered via the RPLIDAR Adapter, which requires 5V, 0.6A through the NUC.
- **RGB Camera:** Powered by the Raspberry Pi through the CSI interface, it operates on 3.3V, 0.3A, capturing visual data for navigation and object detection.
- **WiFi Adapter with External Antenna:** Powered by the Raspberry Pi with 5V, 1.2A through a USB connection, it facilitates wireless communication for remote monitoring and control.
- **IMU:** It draws 3.3V, 6mA from the Loe Core Controller via UART, delivering real-time orientation and acceleration data.
- **DC Motors with Encoders:** Four motors, each powered by 12V, 0.55A from the Loe Core Controller, drive the robot's movement and ensure precise control through positional feedback.

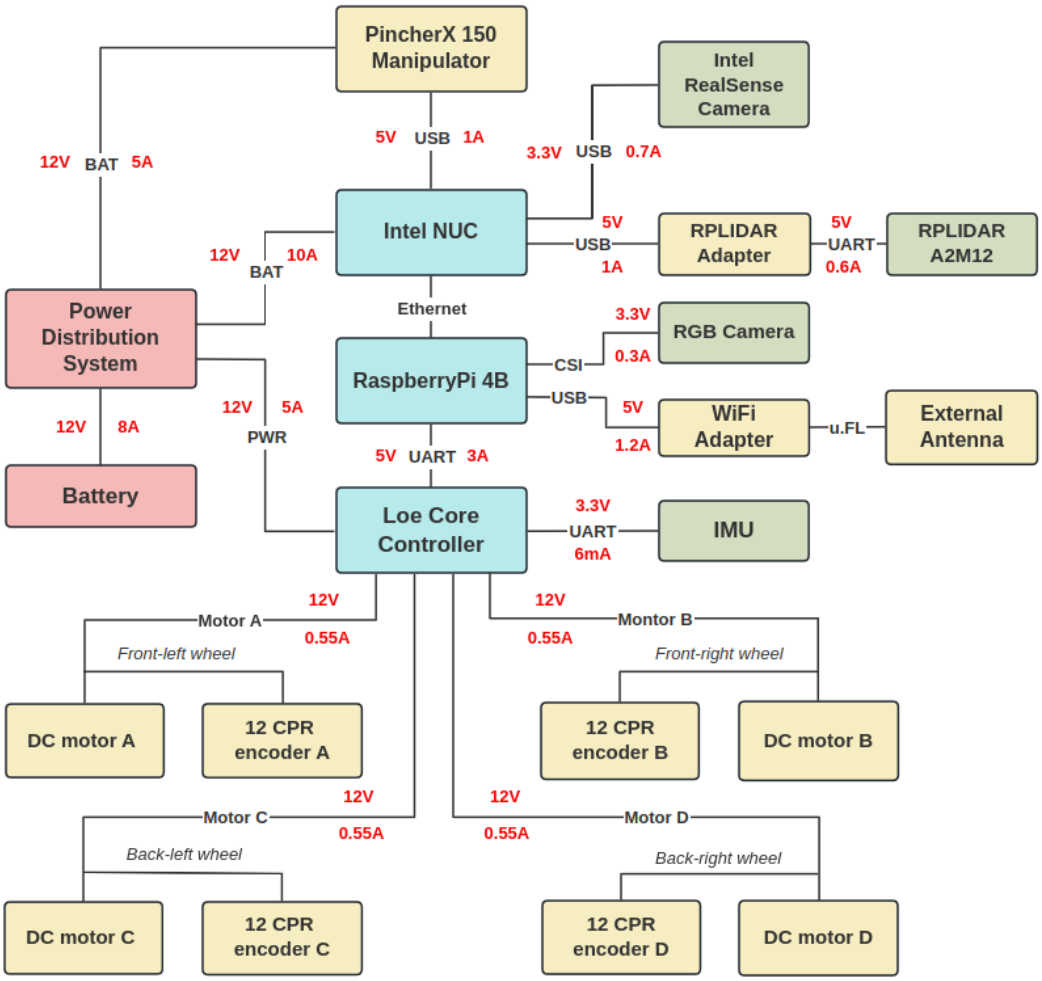


Fig. 11. Power connection diagram

6.2 Power Budget Analysis

The power budget analysis indicates that the current 12V, 8A battery is insufficient to power all components simultaneously. With a total capacity of 96W, the combined demands of the Intel NUC (12V, 10A), PincherX 150 Manipulator (12V, 5A), and Loe Core Controller along with its peripherals (12V, 5A) exceed the battery's maximum output. This results in a total current requirement of over 20A, which is far beyond the battery's capability.

To address this issue, the following specific adjustments are recommended:

- **Adding Batteries in Parallel**: Connect an additional 12V battery in parallel with the existing one to increase the available current capacity. This approach will maintain the 12V output while ensuring that the combined current supply meets the system's requirements. With this modification, the system will have more than enough power to support the Intel NUC, the manipulator, and all peripherals simultaneously without overloading the battery.
- **Power Management Optimization**: In addition to adding batteries, implement a power management strategy to optimize energy distribution. This could involve prioritizing critical com-

ponents like the Intel NUC and manipulator during high-demand operations while limiting or temporarily shutting down non-essential components when necessary.

- **Supplementing with an External Power Supply**: For testing or stationary use, an additional external power supply can be used to directly power high-demand components such as the Intel NUC or manipulator, alleviating the strain on the battery and ensuring stability during prolonged operations.

Without these changes, the system will experience power shortages, leading to instability or operational failures. The proposed modifications will ensure reliable performance and meet the power demands of all components, enabling stable and continuous operation.

7 Software Design

7.1 Software Function Overview

Based on the analysis of Design Requirements Analysis, we have ultimately defined the core functional modules of the robot software as follows. The RQT diagram and sub-module design sections are structured accordingly. The system mainly consists of six modules: Perception Module, Navigation Module, Grasping Module, User Interface Module, Communication Module, System Management Module, which are primarily responsible for implementing the robot's environmental perception function, path planning and obstacle avoidance function, object grasping and placement function, user interaction and visualization function, data communication function, and system state management function.

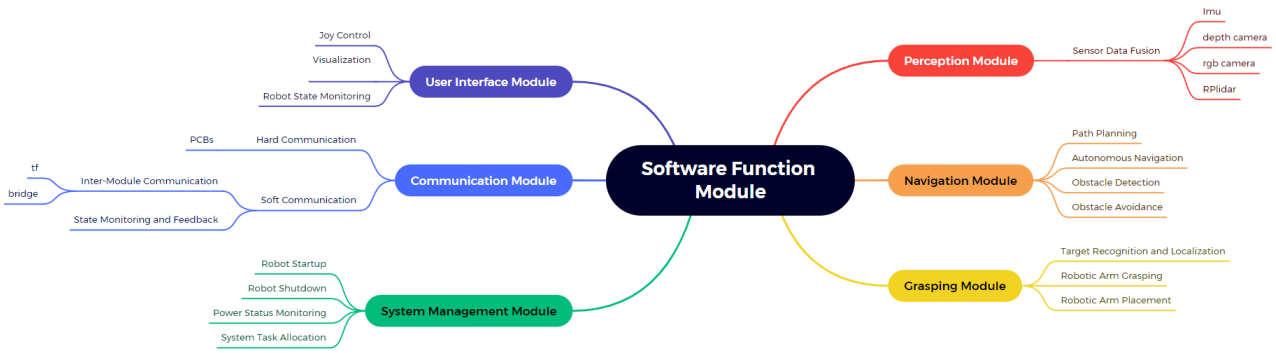


Fig. 12. Overview Diagram of Software Functional Modules

7.2 RQT graph

Consistent with the system diagram introduced in Chapter 7.1, we have ultimately designed and generated a comprehensive RQT (Robotics Query Tool) node graph (as shown in Figure 13). Different colors correspond to the different modules described in Chapter 7.1, visually presenting the

implementation flow of various module nodes and the topic communication status. This diagram outlines the RQT of the core nodes for each module.

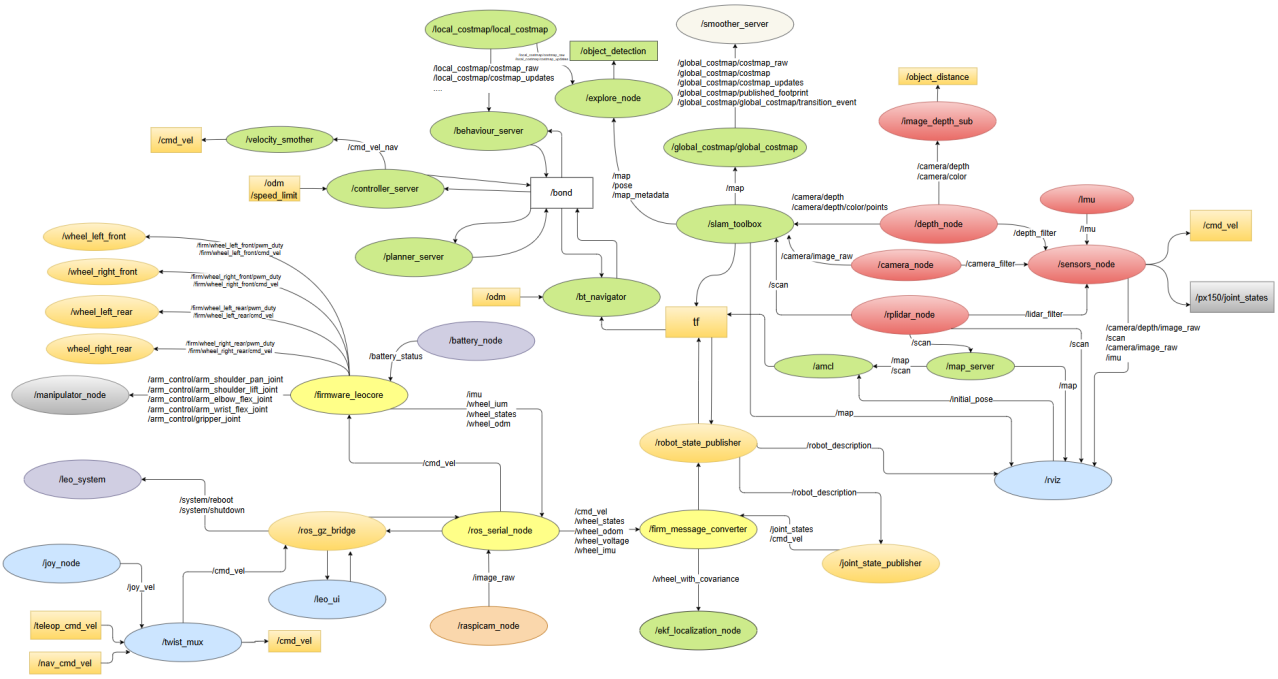


Fig. 13. RQT Graph Corresponding to System Block Diagram

Color	Module	Description
	Communication Module	Direct interaction with hardware via PCBs, converting hardware information into software information through interfaces.
	Communication Module	Data exchange between modules is achieved through the ROS framework.
	Grasping Module	Perform tasks such as target recognition, grasping, and placement.
	User Interface Module	Provide interfaces for user-robot interaction, including remote control and visualization functions.
	Perception Module	Collect sensor data, perform data fusion, and provide raw data for target detection.
	Navigation Module	Implement path planning, localization, navigation, obstacle avoidance, and target detection.
	System Management Module	Manage the system's operational states, including startup, shutdown, restart, and power management.

Fig. 14. RQT Module Division Color-Coded Reference Diagram

7.3 Modular Design

7.3.1 Perception Module Design

This module primarily corresponds to the red section of the RQT diagram. In our current configuration, the main sensors used include LiDAR (Laser Radar), IMU (Inertial Measurement Unit), and a

depth camera. Multi-sensor data fusion enhances the accuracy of robot localization and map construction.

In the RQT graph, the main nodes of the Perception Module include `/depth`, `/camera`, `/rplidar_node`, and `/imu`, while `/sensors_node` integrates these multiple nodes. The relevant topics include `/scan`, `/camera/depth/image_raw`, and `/imu`, providing reliable data support for subsequent mapping (SLAM), obstacle detection, and localization functionalities.

7.3.2 Navigation Module design

This module primarily corresponds to the green section of the RQT graph. We plan to use `slam_toolbox` to construct real-time maps of dynamic environments, enabling the exploration of unknown areas while performing local localization. Additionally, Extended Kalman Filter (EKF) will be employed to fuse odometry, IMU, and LiDAR data, enhancing the smoothness and accuracy of localization.

The target detection algorithm has not yet been determined. For Global Path Planning, we intend to use the `A*` algorithm, which is simple and efficient for finding the shortest path. For Local Path Planning, we plan to use `TEB`.

The core nodes include `/planner_server`, `/controller_server`, and `/bt_navigator` for path planning and navigation. `/slam_toolbox` is responsible for real-time SLAM and localization. Additionally, the `/velocity_smoother` node ensures smooth velocity control, and the `/explore_node` node handles target detection and obstacle avoidance during navigation.

Through communication topics such as `/odom`, `/speed_limit`, `/map`, and `/joint_states`, real-time updates to `/cmd_vel` and `/pose` are achieved, enabling autonomous navigation, obstacle avoidance, dynamic path adjustment, and target detection during navigation.

Implementation Flowchart: The final Navigation Module implementation flowchart is shown below (Figure 15). Figure 16 is a detailed implementation flowchart for Frontier Exploration. This flowchart primarily represents the navigation system process for path planning and dynamic obstacle detection, including sensor initialization, localization, frontier exploration, global path planning, local path planning, and target grasping.

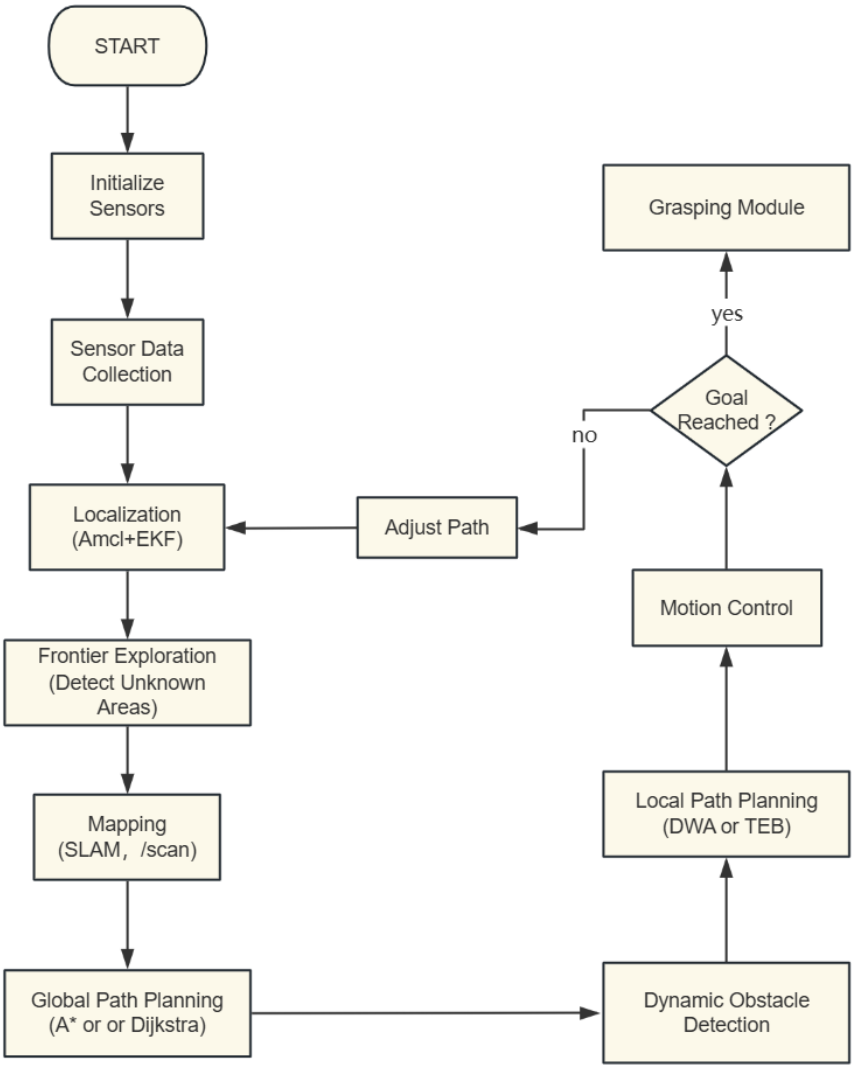


Fig. 15. Behavior Tree of the Navigation Module Implementation

Through Frontier Exploration, the boundaries of unexplored areas are identified, guiding the robot to navigate to uncharted regions, ultimately achieving global exploration of the environment. The flowchart is as follows:

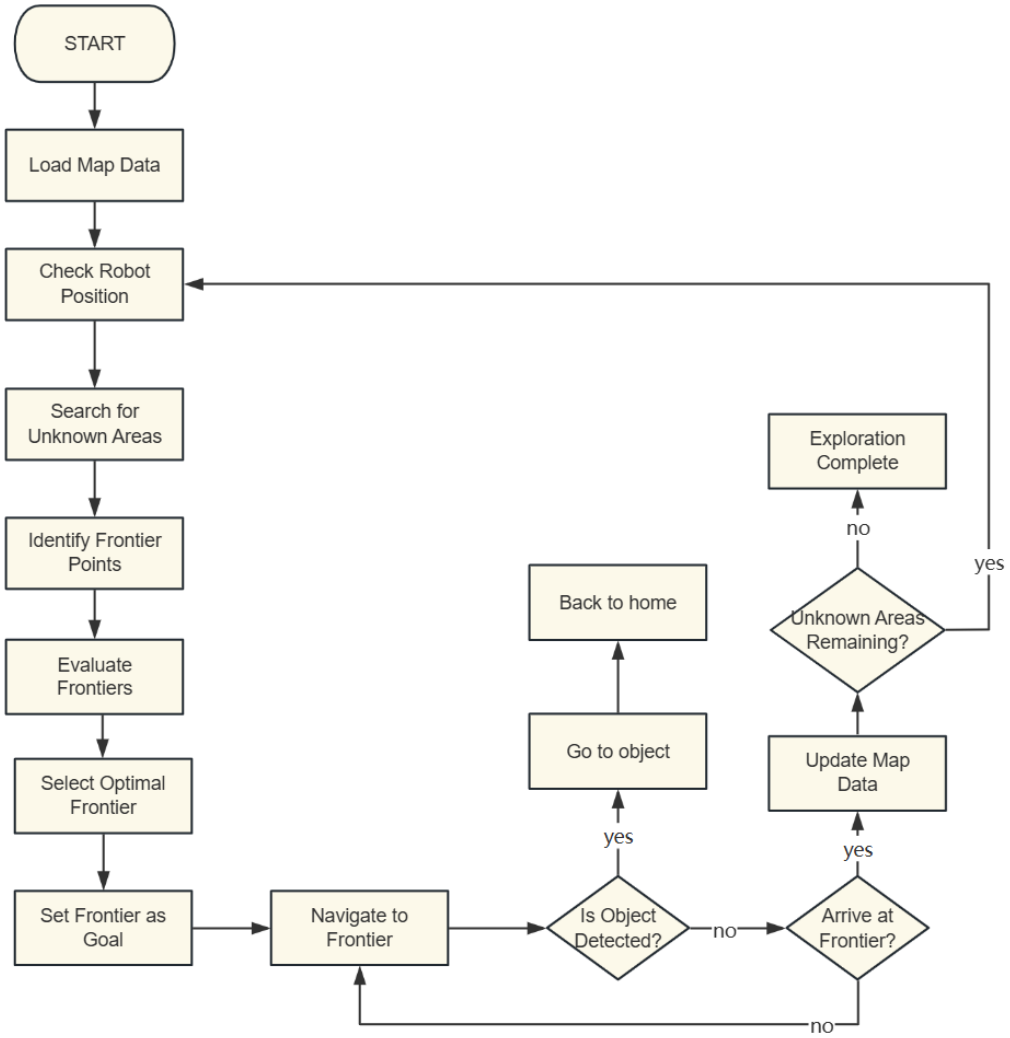


Fig. 16. Flowchart of Frontier Exploration Implementation

7.3.3 Grasping Module design

This module primarily corresponds to the gray section of the RQT graph. It is designed to implement the robotic arm's grasping and placement functions. By integrating the robotic arm's operational control, target detection via a depth camera, and grasping path planning, the system ensures that the robot can accurately identify, locate, and grasp target objects.

The core nodes include `/px150_control_node`, `/target_subscriber`, and `/flag_subscriber`, which are responsible for controlling the robotic arm's movements, gripper operations, and their coordination.

In future plans, we aim to further refine and optimize the target detection algorithm, explore the potential of machine learning to enhance detection efficiency, and investigate force feedback-based grasping algorithms to improve the robot's adaptability to complex objects.

This module utilizes topics such as `/joint_states`, `/camera/depth`, and `/scan` to acquire the robotic arm's state and 3D information about objects. Coordination with the navigation module is achieved

through `/cmd_vel`, ensuring that the grasping path avoids obstacles and completes grasping and placement tasks safely and efficiently.

Implementation Flowchart: The final Grasping Module implementation flowchart is shown below. The flowchart illustrates the robotic arm grasping process, including target detection, 3D position calculation, execution of the grasp, and robot movement.
Please refer to the Grasping Module RQT graph in the next chapter for a more detailed and clear explanation of each node in the RQT graph related to the grasping program.

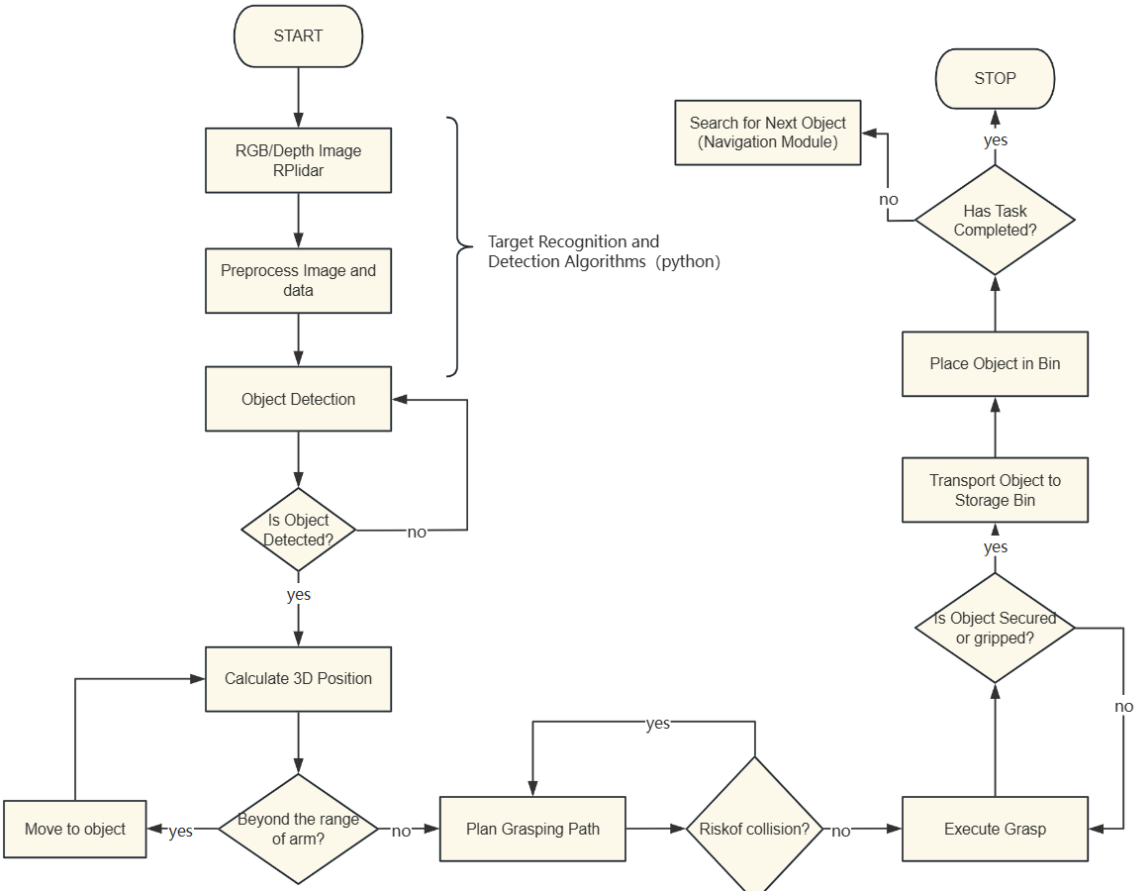


Fig. 17. Behavior Tree of the Grasping Module Implementation

As shown in the figure above, the general process of the robotic arm grasping module is as follows: After target detection, /color_object_position publishes the target coordinates to /px150_control_node, triggering the grasping task. Then, the robotic arm performs motion control via /px150/commands/* and continuously updates /px150/joint_states in real time. Once the operation is completed, a signal is published through /control/flag, notifying flag_subscriber_node to proceed to the next step, such as placing the object, resetting, or executing the next grasping task.

7.3.4 Communication Module design

This module primarily corresponds to the orange and yellow sections of the RQT graph.

The orange section is responsible for enabling communication between the robot's internal modules through the ROS system, such as exchanging data in the form of topics and services. The core nodes include `/ros2_gz_bridge` and `/robot_state_publisher`.

The yellow section focuses on hardware communication, interacting with external devices (e.g., control consoles) via PCB boards, wireless communication interfaces, and other methods. The core nodes include `/firmware_lecore` and `/firm_message_converter`, which are used to receive and send data related to the chassis and sensors, ensuring detailed conversion between software and hardware.
Future work will focus on enhancing the stability and transmission speed of this module.

7.3.5 System Management Module design

This module primarily corresponds to the light purple section of the RQT graph, with some overlap with other modules.

We designed this module to be primarily responsible for the overall management and coordination of the robot system, including low-level hardware control, task scheduling, and status monitoring functionalities.

The core nodes include `/firmware_leo_core` for low-level hardware control, `/battery_node` for monitoring battery status, `/robot_state_publisher` for publishing robot state and posture information, and `/joint_state_publisher` for managing joint state data.
Through interaction with other modules, this module ensures the stable and reliable operation of the system.

7.3.6 User Interface Module design

This module primarily corresponds to the light blue section of the RQT graph. We designed this module to be primarily responsible for providing human-machine interaction functionalities, including remote control, status monitoring, and task configuration.

In the RQT graph, nodes related to user interaction can be observed. For example, `/joystick_node` is used to receive joystick input, enabling manual control of the robot's motion and tasks. `/teleop_cmd_vel` converts remote control input into velocity control commands, which are published to the `/cmd_vel` topic to control the robot's movement. `/rviz` provides a visualization interface for the user. `/leo_ui` helps users monitor the robot's real-time status information, such as position, speed, and battery level.

- Start or shut down the robot
- Visualize the robot's trajectory and other details
- Monitor the robot's battery level, speed, etc.
- Steering the robot

7.4 RQT Graph for Each Module

7.4.1 Perception module RQT graph

1) RPlidar

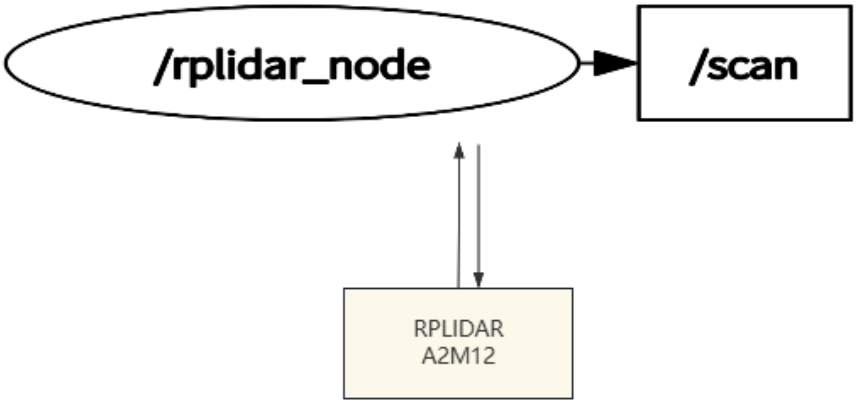


Fig. 18. Rqt Graph For LiDAR Sensor Data

The node /rplidar_node, located in the rplidar_ros package, serves as the ROS driver node for the RPLIDAR A2M12. It publishes the collected laser scan data to the /scan topic. Subsequent processes such as SLAM, obstacle avoidance, or path planning can subscribe to /scan to obtain real-time LiDAR data for map construction and navigation functions.

2) Imu

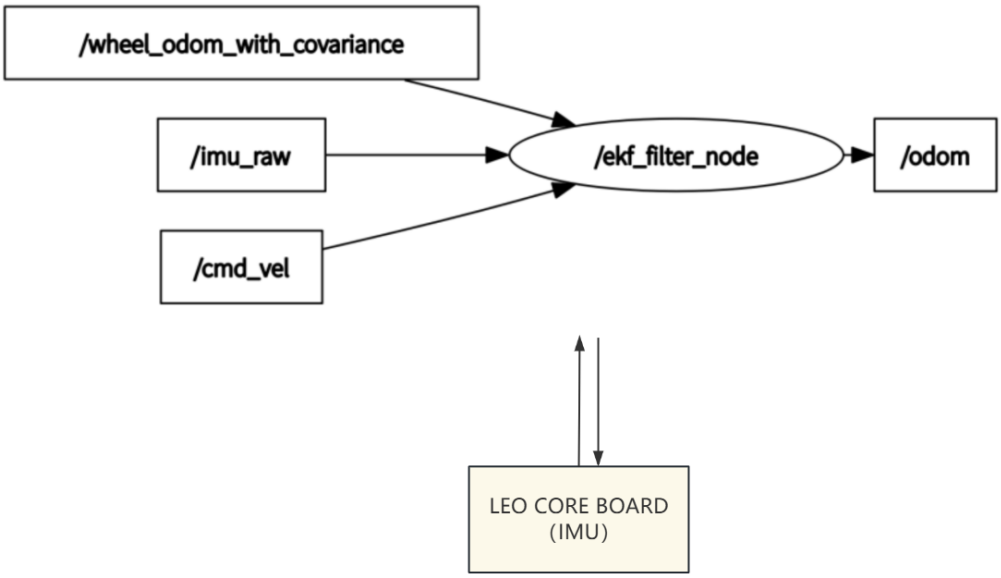


Fig. 19. RQT Graph for IMU Sensor Data and Sensor Data Fusion

IMU data is published by the sensor driver node to the /imu_raw topic, containing information such as acceleration and angular velocity. The /ekf_filter_node subscribes to this topic and fuses it with

/wheel_odom_with_covariance (wheel odometry) using EKF (Extended Kalman Filter) to improve the accuracy of the robot's pose estimation. Finally, it publishes the /odom topic, providing more stable localization information for SLAM and navigation.

3) Camera

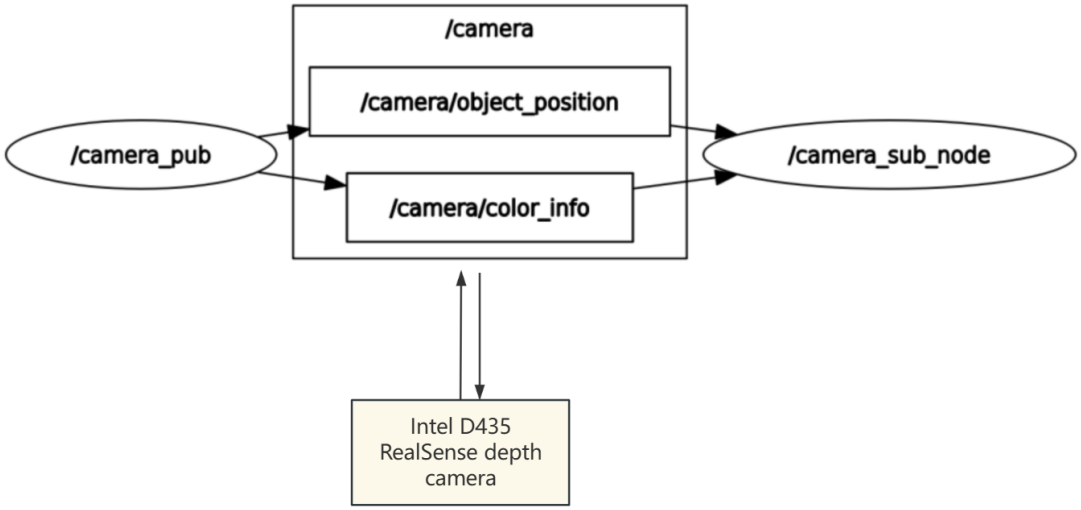


Fig. 20. RQT Graph for Camera Sensor Data

The /camera node, located in the realsense-ros package, serves as the RGB-D sensor node, capturing and publishing color information (/camera/color_info) and depth information (/camera/object_position). The /camera_sub_node subscribes to these topics to perform data fusion, target detection, or SLAM tasks, providing the robot with visual perception capabilities.

4) main_control

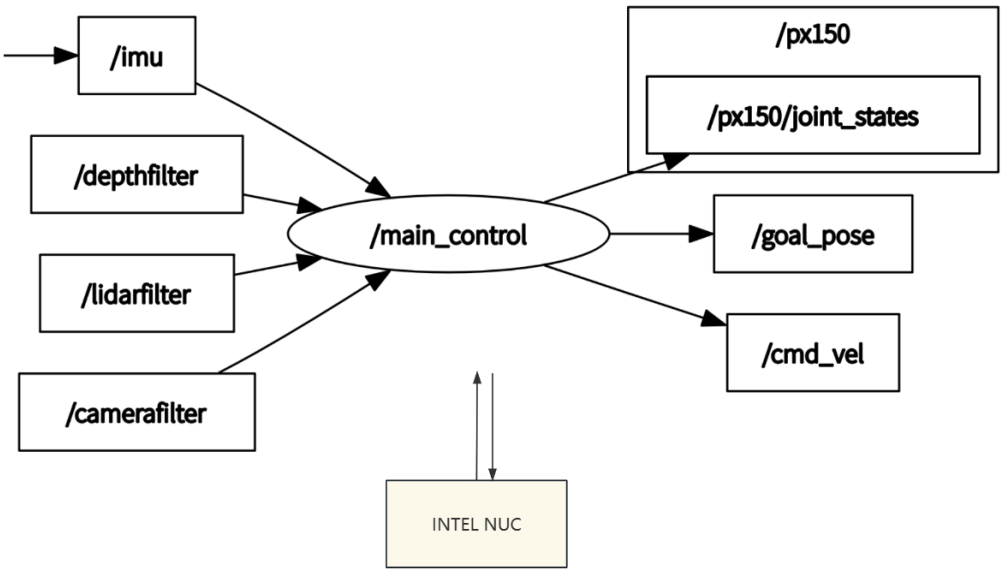


Fig. 21. RQT Graph For Core Control Module

The /main_control node serves as the core control node, receiving processed sensor data from /imu, /depth, /rplidar_node, and /camera for data processing and decision-making. It then publishes

target poses (/goal_pose), robot motion control commands (/cmd_vel), and PX150 robotic arm joint states (/px150/joint_states) to enable autonomous navigation and robotic arm operation. The Intel NUC computing platform is used for data processing and task execution.

7.4.2 Navigation Module RQT graph

1) Mapping and Localization

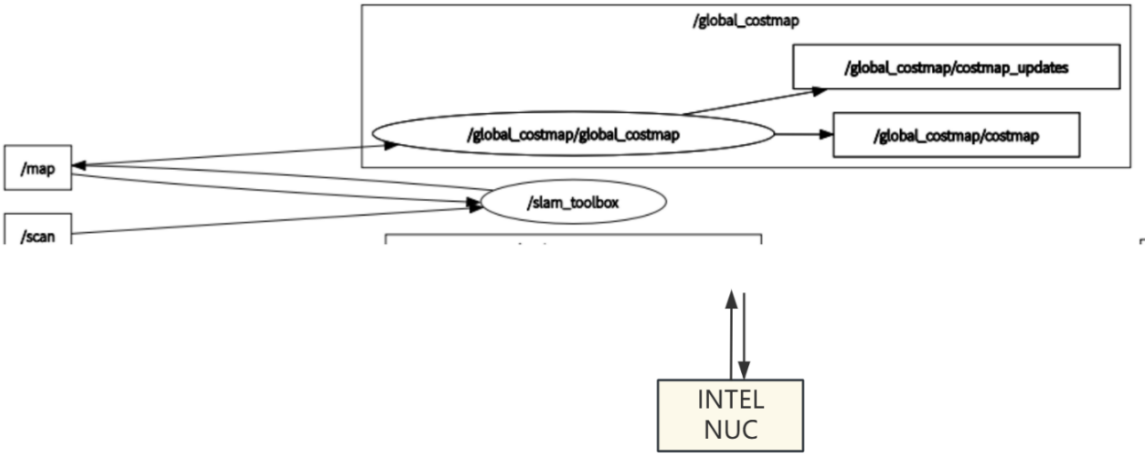


Fig. 22. RQT Graph For Mapping and Localization Module

The slam_toolbox package is used for Simultaneous Localization and Mapping (SLAM). The /slam_toolbox node subscribes to LiDAR data (/scan), wheel encoder and IMU data (/odom), and coordinate transformations (/tf) to fuse information and estimate the robot's pose (position and orientation) in real time. It utilizes LiDAR to capture static environmental features and continuously updates the map as the robot moves. Finally, it publishes the map in the form of a 2D occupancy grid (/map) for navigation and path planning.

2) Navigation

Navigation primarily consists of the following four components. Additionally, in real-world scenarios, we utilize Frontier Exploration for autonomous exploration of unknown areas, enabling navigation and local obstacle avoidance. For path planning, we ultimately use A* for global path planning and TEB for local path planning.

- Local Path Planning and Motion Control

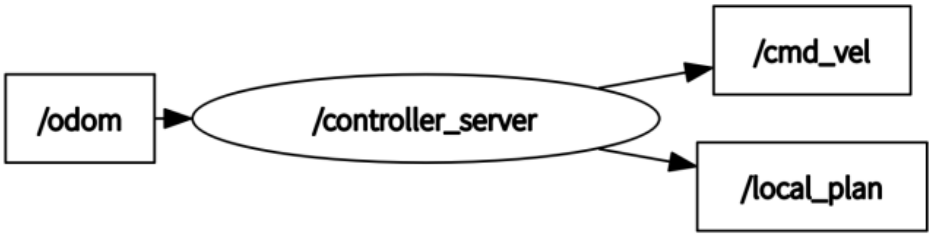


Fig. 23. RQT Graph Of Local Path Planning and Motion Control In The Navigation Module

The /controller_server node in the navigation_demos package is responsible for local path planning and motion control. This node subscribes to the /odom topic and calculates a locally feasible path (/local_plan) based on the results of global path planning. It then publishes velocity control commands (/cmd_vel) to ensure the robot can safely avoid obstacles and execute smooth motion control in dynamic environments.

- Global Path Planning

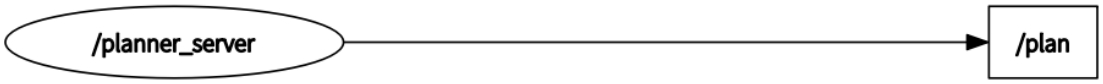


Fig. 24. RQT Graph Of Global Path Planning In The Navigation Module

The /planner_server node in the navigation_demos package is responsible for global path planning. It calculates the optimal travel route based on the robot's current pose, target location, and environment map. The planned path is then published via the /plan topic for use by the local path planning and control module, ensuring the robot can efficiently avoid obstacles and follow the predetermined route.

- Short-distance Obstacle Avoidance and Dynamic Path Adjustment

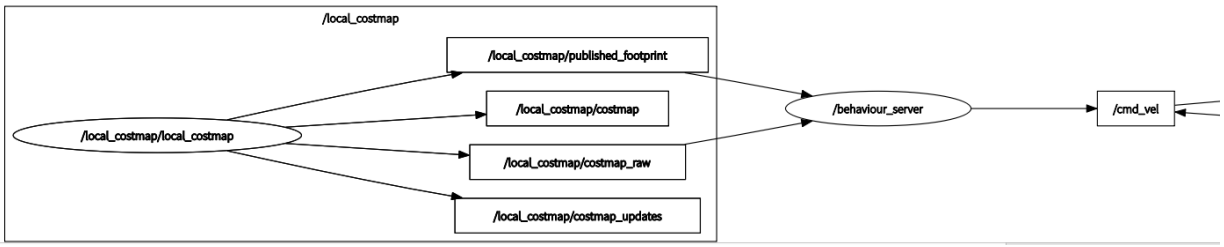


Fig. 25. RQT Graph For Short-distance Obstacle Avoidance and Dynamic Path Adjustment

The /behaviour_server node in the navigation_demos package is responsible for short-distance obstacle avoidance and dynamic path adjustment. It utilizes the /local_costmap/local_costmap node to construct a local environmental perception map. Obstacle information around the robot is continuously updated through the /local_costmap/costmap, /local_costmap/costmap_raw, and /local_costmap/costmap_updates topics. The /behaviour_server node subscribes to this data to generate appropriate obstacle avoidance strategies and controls the robot's safe navigation via the /cmd_vel topic.

- Behavior Tree Navigation

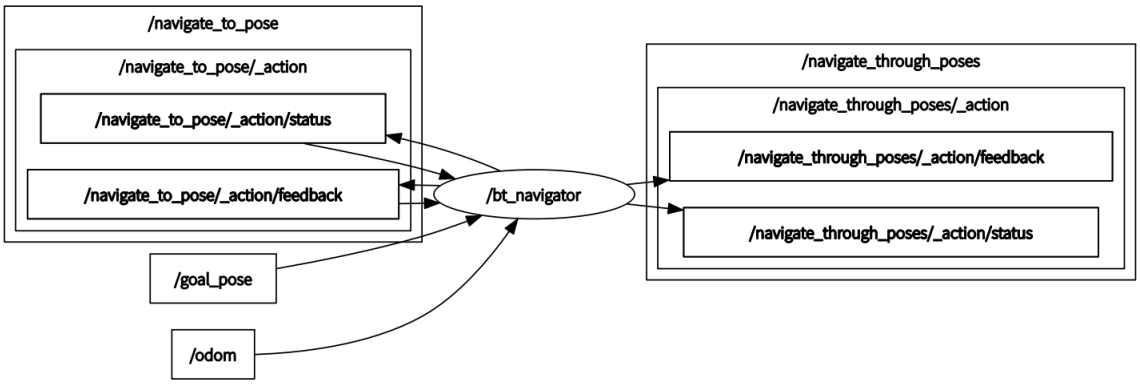


Fig. 26. RQT Graph Of Behavior Tree Navigation In The Navigation Module

The /bt_navigator node in the navigation_demos package manages single-point and multi-point navigation tasks. It dynamically optimizes the path by integrating target poses (/goal_pose) and odometry (/odom). Using Behavior Trees (BTs), it handles single-goal navigation (/navigate_to_pose) and waypoint sequence navigation (/navigate_through_poses). Additionally, it provides real-time feedback (_action/feedback) and task status (_action/status) to efficiently and flexibly execute autonomous navigation tasks.

3) Grasping Module RQT graph

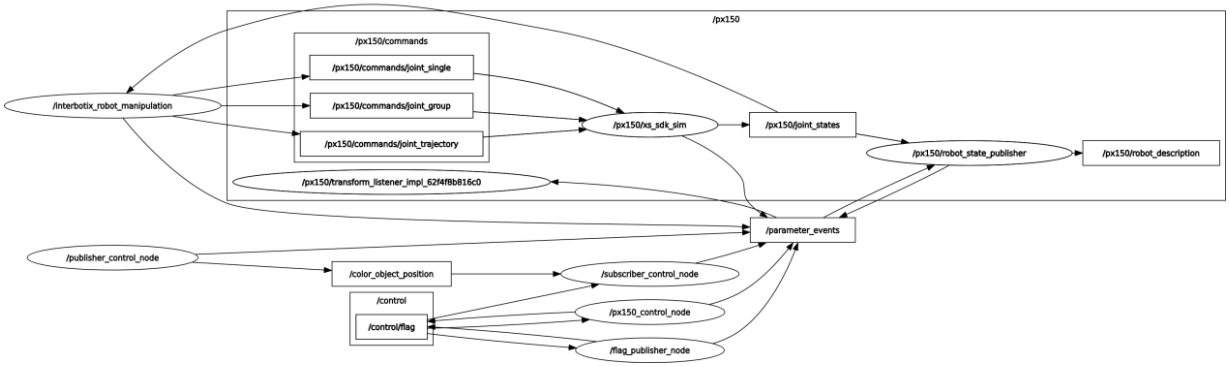


Fig. 27. RQT Graph For Grasping Module

- Robotic Arm Control
  We use /interbotix_robot_manipulation to send robotic arm commands, including single joint control (/px150/commands/joint_single), multi-joint control (/px150/commands/joint_group), and trajectory control (/px150/commands/joint_trajectory). Additionally, /px150/frame_from_listener listens to TF transformations, providing coordinate information. Meanwhile, /px150_control_node is used to execute grasping, placement, and signal feedback for the robotic arm.

- Robotic Arm State Management
  We use /px150/joint_states to publish the robotic arm joint states, while /px150/robot_state_publisher subscribes to these states and updates /px150/robot_description for robot model publishing.

- Target Detection and Control Signal
  We use /publisher_control_node to send target object coordinates (/color_object_position) to the robotic arm control node (/subscriber_control_node). At the same time, /control/flag

serves as a signal publisher, sent by flag_publisher_node and listened to by flag_subscriber_node, triggering the robotic arm to execute specific operations.

GitHub: https://github.com/CMX-9/Team5-Mobile_Grasping_Robot
Notice: For security reasons, the code has been encrypted. The password is:123123

8 Project Plan

The tasks for the second semester can be broadly categorized into three primary modules: **Mapping and Navigation, Object Detection**, and **Grasping**. Each task requires substantial preliminary preparation and subsequent debugging and optimization. Based on the project timeline, the following work plan has been formulated.

Task	Task Owner	Start Time	Duration	End Time	Milestones	Deliverables
Software Frame Programming	All	2025-01-27	12	2025-02-07	Completion of the robot's software framework, including ROS2 nodes, communication architecture, and basic control functions	Source code repository(GitHub)
Mapping&Navigation	Mingxiang Chen&Zhen Yang	2025-02-08	30	2025-03-09	The robot can autonomously generate a map and perform basic path planning and navigation	SLAM-generated map, navigation path test videos
Object Detection	Yu-Chuan Liao	2025-02-08	30	2025-03-09	Object detection model can accurately identify specific objects with an accuracy of over 90%	Object detection model (code), test dataset
Grasping	Yunxue Pan	2025-02-08	30	2025-03-09	The robot can recognize and grasp target objects with a success rate of at least 90%	Grasping demonstration video, grasp success rate analysis report
Software Function Integration	All	2025-03-10	14	2025-03-23	All functional modules are integrated, and the robot can execute a full task flow in both simulation and real environments	Integration test documentation, ROS2 launch files, Gazebo/real-world operation video
Integral robot assembly	All	2025-03-26	2	2025-03-27	Mechanical and electronic hardware assembly is completed, with all components installed and tested	Fully assembled robot
Testing	All	2025-03-31	10(excluding Easter Break)	2025-05-02	The robot is tested in various scenarios, performance is optimized, and all functions meet design requirements	Final test report (success rate, error analysis), optimized code documentation, demonstration videos

Fig. 28. Gantt Chart 1

9 Analysis

This section analyzes the mechanical design, electrical design, and software design based on the project goals, in accordance with the Functional and Performance Requirements. It explains how the preliminary design meets these requirements.

9.1 The robot shall have mobility capabilityA.3.1

Verification Success Criteria: The robot should be able to move forward, backward, and rotate on a dry, hard surface for at least 5 meters in each direction without deviation exceeding 5% from the intended path.

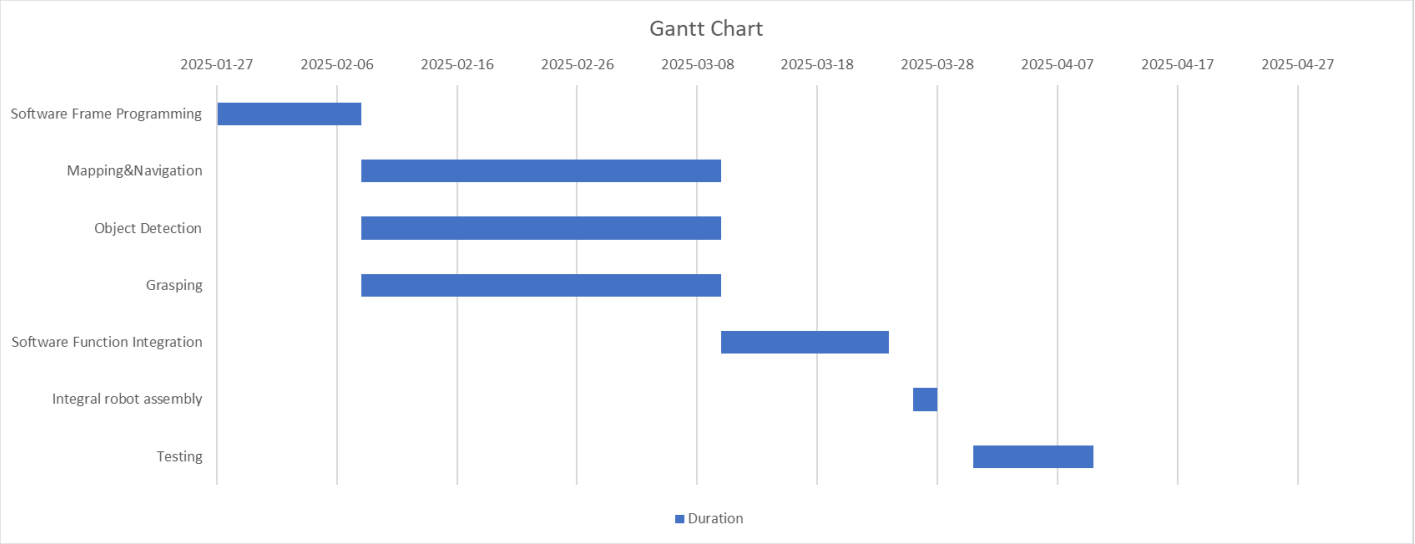


Fig. 29. Gantt Chart 2

Verification Method: Issue motion commands (forward, backward, and rotation in place) to the robot on a dry, hard surface. Each movement type will be tested at least 10 times to measure consistency, and deviations will be recorded.

Implementation Analysis: The LEO core controller serves as the central hub for motor control and sensor data processing, coordinating various modules to achieve precise motion control of the robot. The DC motors with encoders drive the robot's wheels and provide position feedback, ensuring precise control of speed and direction. The integrated IMU continuously monitors the robot's acceleration and angular velocity, supporting fine-tuned motion control. The WiFi adapter with an external antenna ensures a stable wireless connection for remote control and data transmission during testing and debugging.

9.2 The robot shall generate a map based on its surroundingsA.3.2

Verification Success Criteria: The map generated by the robot should match the actual environment with an accuracy of over 95%, verified through overlay comparison.

Verification Method: The robot will be placed in a controlled environment with known obstacles. The generated map will be compared with the actual layout using a similarity analysis. At least 5 test runs in different environments will be conducted to assess accuracy consistency.

Implementation Analysis: The RPLIDAR A2M12, along with the Intel RealSense depth camera, will be used to scan and capture the surrounding environment. These sensors provide essential data for mapping. The slam_toolbox will be employed to construct real-time maps of the dynamic environment, utilizing the LiDAR and depth camera data for precise localization and mapping. The SLAM algorithm processes the sensor information to build an accurate representation of the robot's surroundings, and continuous adjustments will be made based on the dynamic environment, ensuring an accurate map generation process.

9.3 The robot can autonomously navigate and perform obstacle avoidanceA.3.2A.3.3

Verification Success Criteria: The robot should successfully navigate around all obstacles and reach a predefined target in at least 9 out of 10 test runs.

Verification Method: Provide the robot with target coordinates in an environment containing obstacles. Track its movement and measure navigation success rate.

Implementation Analysis: The robot's autonomous navigation and obstacle avoidance capabilities are achieved through a well-integrated system of sensors, algorithms, and computational resources. The IMU provides real-time data on the robot's direction, acceleration, and angular velocity, ensuring precise motion control. The Intel NUC serves as a high-performance platform for handling intensive tasks such as SLAM and advanced navigation algorithms. RPLIDAR A2M12 performs 360-degree environmental scanning, generating a detailed 2D map essential for path planning and obstacle detection, while the Intel RealSense depth camera captures 3D environmental data to enhance object detection and spatial analysis. To minimize sensor interference and maximize data accuracy, the LiDAR and depth camera are strategically positioned at the front of the robot's chassis. The A algorithm is employed for global path planning due to its efficiency in finding the shortest path, while TEB is used for local path planning to optimize navigation in dynamic environments. Key ROS nodes such as /planner_server, /controller_server, and /bt_navigator work together to ensure smooth, real-time navigation and obstacle avoidance, allowing the robot to autonomously reach its target.

9.4 The robot shall be able to detect objectsA.3.4

Verification Success Criteria: The robot shall be able to recognize objects based on color with a success rate exceeding 90% under fixed lighting conditions and color temperature. It shall also be able to match objects of the same color and display the quantity and color of recognized objects on the laptop screen in real time.

Verification Method:A set of colored objects with known properties will be placed in a controlled lighting environment. The robot's recognition system will be tested by capturing images and classifying objects based on color. At least 20 objects will be used to evaluate recognition accuracy. Additionally, the matching algorithm will be tested by presenting pairs of objects with identical and different colors to verify correct grouping. The laptop interface will be monitored to ensure real-time updates of object counts and colors.

Implementation Analysis: The robot's object detection system is built around the Intel RealSense D435 depth camera, which provides RGB and depth information for robust color-based object recognition. The camera captures high-resolution images, while the onboard IMU enhances spatial awareness and stability during movement. The vision module preprocesses images, extracts color features, and classifies objects using a predefined color model.

For color-based recognition, the RGB data from the D435 is processed using image segmentation techniques, such as HSV color space conversion and thresholding. The depth information helps in

filtering out background noise and ensuring objects are correctly identified. A clustering algorithm is used to match objects of the same color.

The detected objects' color and quantity are transmitted via a communication module to the laptop for real-time display. The controlled lighting conditions ensure consistency in color recognition, minimizing the impact of varying ambient light. The D435's depth data can also assist in object differentiation when colors are similar but spatial separation is required. The integration of the Intel RealSense D435 with onboard processing ensures efficient and accurate object detection and classification.

9.5 The robot shall be capable of grasping and releasing objectsA.3.5

Verification Success Criteria: The robotic arm must successfully grasp and transport objects with a drop probability of less than 5% over 30 trials.

Verification Method: The robotic arm will grasp and transport objects of different shapes and weights 30 times. The number of successful attempts will be recorded.

Implementation Analysis: The PincherX 150 robotic arm, with its high precision, multi-degree-of-freedom design, and flexibility, is well-suited for precise object grasping and placement tasks. Its core control system operates through software nodes such as /manipulator_node, /gripper_node, and /arm_controller, which are responsible for controlling the arm's motion, managing the gripper's operation, and ensuring coordination between these elements. Through these nodes, the robotic arm can precisely adjust its posture according to predefined paths and target positions, ensuring that objects are stably grasped and transported to the designated location. Additionally, the /joint_states and /camera topics provide real-time feedback on the arm's state and the 3D information of the object, enabling dynamic adjustments to prevent object drop during the grasping process and ensuring task completion with high efficiency. Furthermore, the PincherX 150's high payload capacity and durability allow it to maintain stability when handling objects of varying shapes and weights, ensuring reliable execution of tasks. Combining the advantages of hardware and software support, the PincherX 150 robotic arm can effectively and accurately complete object grasping and placement tasks.

9.6 The robot shall have a state tracking capabilityA.3.6

Verification Success Criteria: The robot's operational status should be continuously displayed on the terminal, with updates occurring at least every 500 ms.

Verification Method: Monitor terminal logs during robot operation to ensure real-time status updates. At least 10 test sessions will be conducted.

Implementation Analysis: The robot's state tracking capability is realized through the integration of hardware sensors, data processing algorithms, and the /leo_ui user interface, which allows real-time monitoring of the robot's status, including position, velocity, and battery level. The IMU pro-

vides data on the robot's orientation and movement, while wheel encoders track displacement and speed. Battery sensors continuously measure power levels. These data streams are collected and processed by the /leo_ui node, which visualizes the robot's state on the user interface. This enables users to monitor critical parameters in real time, ensuring that the robot's operational status is always available for adjustment or troubleshooting. The /leo_ui node plays a key role in integrating and displaying this information, ensuring the robot's state is accurately tracked and communicated.

9.7 Mechanical design requirements for the robotA.3.7

Verification Success Criteria: The robot's dimensions shall not exceed 500 mm x 500 mm x 500 mm, and structural stability shall be confirmed through stress testing.

Verification Method: Measure the final dimensions of the robot and conduct a structural strength analysis.

Implementation Analysis: The designed mechanical structure results in an overall robot size of 448.55 mm × 439.03 mm × 527.83 mm when the robotic arm is retracted. Based on the strength analysis in the mechanical design, it is determined that the structure will not undergo strain during movement, ensuring good stability.

9.8 The cost of the robot shall be controlledA.3.8

Verification Success Criteria: In subsequent designs, laser cutting and 3D printing shall not exceed the manufacturing budget.

Verification Method: Calculate the total cost and compare it against the budget limit.

Implementation Analysis: In addition to the already provided robot components, the subsequent costs will only involve the expenses for laser cutting or 3D printing to assemble the mechanical structure of various parts. This approach allows for effective cost control by minimizing the need for expensive or complex manufacturing processes, while still ensuring high precision and quality in the final assembly. The use of cost-effective manufacturing techniques like laser cutting and 3D printing offers flexibility in design and production, enabling rapid prototyping and modifications when necessary, which further enhances the overall cost-effectiveness of the project.

9.9 SafetyA.3.1

Verification Success Criteria: The maximum speed of the robot shall not exceed 0.5 m/s, and it shall be able to stop within 1 second after receiving a braking command.

Verification Method: Measure maximum speed and braking time over 10 trials to ensure compliance.

Implementation Analysis: The robot's maximum speed and braking functionality are achieved through an integrated approach across electrical, software, and mechanical design. The electrical design utilizes motor controllers with encoders for precise speed regulation and rapid braking via reverse polarity or dynamic braking. The software ensures smooth deceleration by implementing control algorithms, such as PID, to meet the 1-second stop requirement. Mechanically, the lightweight structure and differential drive system, along with optimized wheel design, enable efficient deceleration and ensure the robot can stop quickly within the specified limits.

9.10 ReliabilityA.3.7

Verification Success Criteria: The robot shall operate for at least one hour on a full charge under normal working conditions.

Verification Method: Fully charge the robot and record its operational duration over 3 test runs.

Implementation Analysis: According to the electrical design analysis, the batteries provided do not provide sufficient voltage. The Leo Rover is powered by its onboard battery supply, while an external power source is planned for the NUC and the manipulator.

10 Appendix

A Design Requirements Analysis

A.1 Stakeholder Engagement

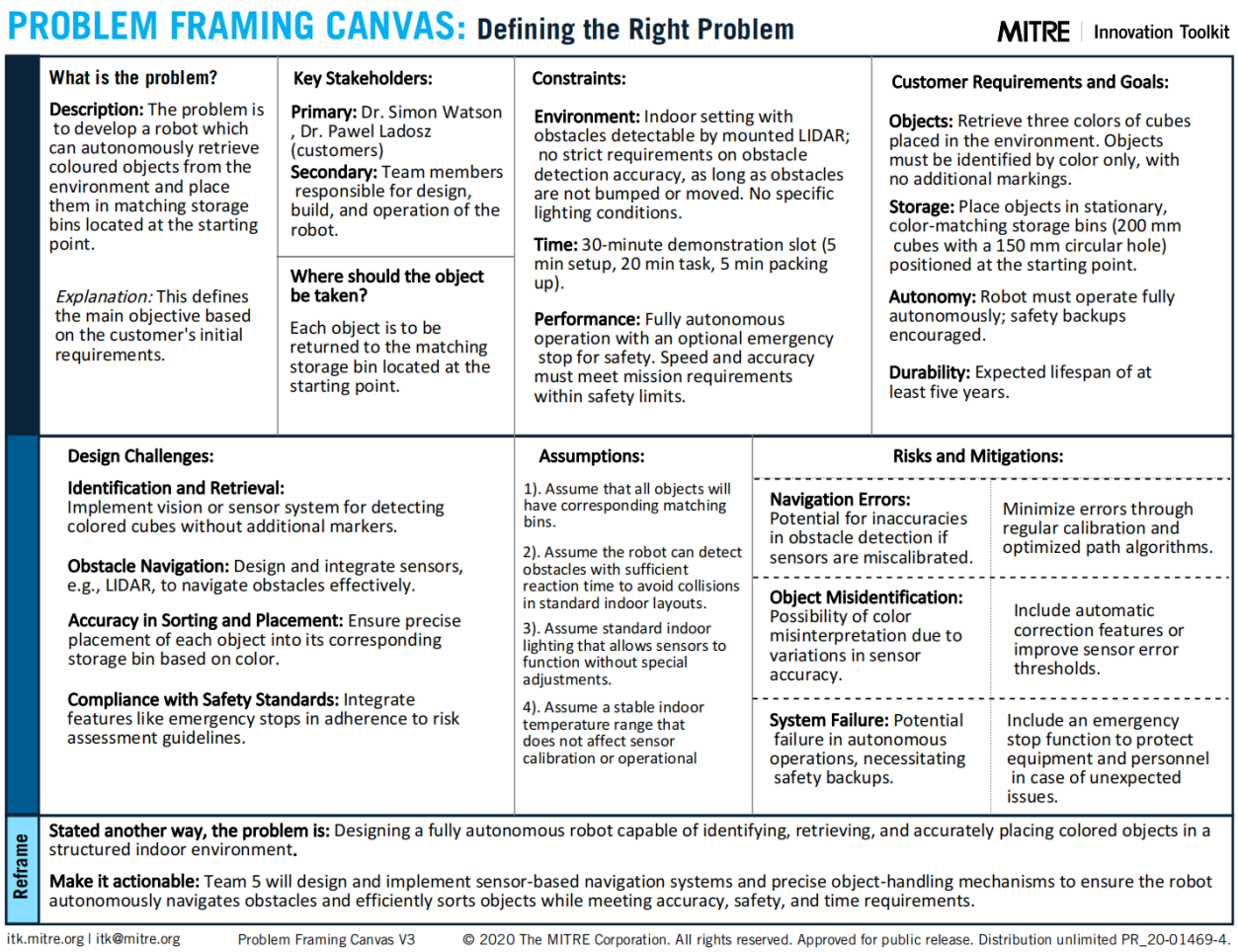
A.1.1 Introduction

In this project, Group 5 aims to understand and address the customer's requirements effectively. The initial statement provided by the customer is as follows:

> **"Develop a robot which can autonomously retrieve coloured objects from the environment and place them in matching storage bins located at the starting point."**

To further define the scope of the project, the team held two engagement sessions with the primary stakeholders, Dr. Simon Watson and Dr. Pawel Ladosz. The following sections summarize their responses to clarify key requirements and expectations for the robotic system design project.

A.1.2 Problem Framing Canvas



PROBLEM FRAMING CANVAS: Defining the Right Problem

MITRE | Innovation Toolkit

What is the problem?

Description: The problem is to develop a robot which can autonomously retrieve coloured objects from the environment and place them in matching storage bins located at the starting point.

Explanation: This defines the main objective based on the customer's initial requirements.

Key Stakeholders:

Primary: Dr. Simon Watson , Dr. Pawel Ladosz (customers)
Secondary: Team members responsible for design, build, and operation of the robot.

Where should the object be taken?

Each object is to be returned to the matching storage bin located at the starting point.

Constraints:

Environment: Indoor setting with obstacles detectable by mounted LIDAR; no strict requirements on obstacle detection accuracy, as long as obstacles are not bumped or moved. No specific lighting conditions.

Time: 30-minute demonstration slot (5 min setup, 20 min task, 5 min packing up).

Performance: Fully autonomous operation with an optional emergency stop for safety. Speed and accuracy must meet mission requirements within safety limits.

Customer Requirements and Goals:

Objects: Retrieve three colors of cubes placed in the environment. Objects must be identified by color only, with no additional markings.

Storage: Place objects in stationary, color-matching storage bins (200 mm cubes with a 150 mm circular hole) positioned at the starting point.

Autonomy: Robot must operate fully autonomously; safety backups encouraged.

Durability: Expected lifespan of at least five years.

Design Challenges:

Identification and Retrieval: Implement vision or sensor system for detecting colored cubes without additional markers.

Obstacle Navigation: Design and integrate sensors, e.g., LIDAR, to navigate obstacles effectively.

Accuracy in Sorting and Placement: Ensure precise placement of each object into its corresponding storage bin based on color.

Compliance with Safety Standards: Integrate features like emergency stops in adherence to risk assessment guidelines.

Assumptions:

1). Assume that all objects will have corresponding matching bins.

2). Assume the robot can detect obstacles with sufficient reaction time to avoid collisions in standard indoor layouts.

3). Assume standard indoor lighting that allows sensors to function without special adjustments.

4). Assume a stable indoor temperature range that does not affect sensor calibration or operational

Risks and Mitigations:

Navigation Errors: Potential for inaccuracies in obstacle detection if sensors are miscalibrated.	Minimize errors through regular calibration and optimized path algorithms.
Object Misidentification: Possibility of color misinterpretation due to variations in sensor accuracy.	Include automatic correction features or improve sensor error thresholds.
System Failure: Potential failure in autonomous operations, necessitating safety backups.	Include an emergency stop function to protect equipment and personnel in case of unexpected issues.

Reframe

Stated another way, the problem is: Designing a fully autonomous robot capable of identifying, retrieving, and accurately placing colored objects in a structured indoor environment.

Make it actionable: Team 5 will design and implement sensor-based navigation systems and precise object-handling mechanisms to ensure the robot autonomously navigates obstacles and efficiently sorts objects while meeting accuracy, safety, and time requirements.

itk.mitre.org | itk@mitre.org Problem Framing Canvas V3

Fig. 30. Problem Framing Canvas developed based on customer clarifications

A.2 Problem Statement

After discussion with the customers, Dr. Simon Watson and Dr. Pawel Ladosz, the requirements for the project were defined. The customer requested a fully autonomous robot equipped with sensors and a manipulator capable of retrieving specific objects from an unmapped environment and placing them into matching storage bins within a limited time. The environment will include several stationary obstacles.

This project aims to provide an opportunity to apply our robotics knowledge from lectures in a practical context while fostering teamwork skills, including communication, collaboration, and coordination.

A.2.1 Objectives

- Ensure the operation is safe, efficient, and fully autonomous.(3.1,3.2)

- Use a mobile robotic platform equipped with necessary hardware and sensors to achieve the project goals.
- Develop autonomous navigation capabilities, enabling the robot to map the environment and move without manual intervention.(3.2,3.3)
- Implement computer vision systems to detect, classify, and interact with objects accurately.(3.3,3.4)
- Ensure collision avoidance with both stationary obstacles in the environment.(3.2,3.4)
- Complete the task within a strict time limit of 20 minutes.
- Incorporate robust safety measures, such as an emergency stop function, to enhance operational safety. (3.1.4)

A.3 Functional and Performance Requirements

According to the above customer requirements, the robot to be designed shall be able to achieve autonomous navigation, avoiding obstacles, recognising objects and grasping and placing functions. The following is a detailed description of the function and performance.

A.3.1 The robot shall have mobility capability

3.1.1 The robot shall be capable of moving on dry, hard surfaces.

3.1.2 The robot shall operate in indoor environments.

3.1.3 According to the ISO/TS 15066 recommendations for safe speeds when interacting with humans, the maximum speed of the robot is specified to be no more than 0.5m/s. This speed significantly reduces the risk of injury from a collision, while still maintaining basic task execution efficiency

3.1.4 The robot shall have an braking function, enabling it to stop within 1 second to comply with safety standards.

A.3.2 The robot shall be able to autonomously plan its path

3.2.1 The robot shall be capable of generating a map of its surroundings in real-time during operation.

3.2.2 The robot shall be able to autonomously plan navigation paths based on the generated map.

A.3.3 The robot shall be able to avoid obstacles

3.3.1 The probability of the robot successfully navigating the path in the absence of moving obstacles shall be at least 90%.

3.3.2 The robot shall be capable of detecting obstacles within a one-meter diameter range, including those that may be lower than the robot's frame, with a false detection rate of less than 5%.

3.3.3 The obstacles shall be fixed static objects.

A.3.4 The robot shall be able to detect objects

3.4.1 Under fixed lighting conditions and color temperature, the robot shall be able to recognize objects based on color, with a success rate exceeding 90%.

3.4.2 The robot shall be able to match recognized objects of the same color.

3.4.3 The robot shall be able to display the quantity and color of recognized objects on the laptop screen in real time.

A.3.5 The robot shall be capable of grasping and storing objects

3.5.1 The robotic arm shall be able to move to the target position in 10 seconds to ensure efficient execution of the task and meet the overall time constraints of the operation.

3.5.2 The robotic arm shall be capable of grasping polygonal objects, with the ability to grasp items weighing less than 50 grams at a time.

3.5.3 The robot shall be able to place the grasped objects at designated locations, with a drop rate of less than 5% during the process.

A.3.6 The robot shall have a state tracking capability

3.6.1 The robot shall provide real-time feedback on its operational status: gripping, recognizing, and avoiding obstacles.

A.3.7 Mechanical design requirements for the robot

3.7.1 The dimensions of the robot shall not exceed 500 mm x 500 mm x 500 mm.

3.7.2 The robot's lifespan shall be at least 5 years to meet the client's specified durability requirement, ensuring reliable performance and long-term value throughout its intended operational period.

3.7.3 The robot shall be able to operate for at least 1 hour when fully charged.

A.3.8 The cost of the robot shall be controlled

3.8.1 In subsequent designs, laser cutting and 3D printing shall not exceed the manufacturing budget.

A.4 Requirements Verification Matrix

After identifying all the specific requirements, it is necessary to identify how to validate that the requirements are realised. The following verification matrix is used to document the validation process. The verification matrix contains test items, test methods and testers. The final test results are recorded in a tabular form. The verification matrix is shown in the table 1 below.

Table 2. Requirements Verification Matrix Table

Requirement No.[A]	Section[B]	Shall Statement[C]	Verification Success Criteria[D]	Verification Method[E]	Responsible Party[F]	Results[G]
P-1	3.1.1	The robot shall be capable of moving on dry, hard surfaces.	The robot can move normally on a dry, hard indoor surface.	Test the robot's movement performance on dry, hard indoor surfaces.	Yunxue	
P-2	3.1.2	The robot shall be able to move forward and backward and to turn.	A. The robot can move forward and backward. B. The robot can rotate.	Test and verify in a laboratory environment, observing and recording the robot's movement performance.	YunXue	
P-3	3.1.3	The robot's movement speed shall comply with safety standards.	The maximum speed does not exceed 0.5 meters	Run the robot continuously for at least half an hour in a laboratory environment, recording the maximum speed during the test.	YunXue	

Requirement No.[A]	Section[B]	Shall Statement[C]	Verification Success Criteria[D]	Verification Method[E]	Responsible Party[F]	Results[G]
P-4	3.1.4	The robot shall have an emergency braking function, enabling it to stop.	Successful stopping.	Test if the robot can stop.	YunXue	
P-5	3.1.4	The robot shall can brake successfully in less than 1 second	The time from issuing the stop command to the robot completely stopping does not exceed 1 second.	Test whether the robot's average braking time exceeds 1 second.	YunXue	
P-6	3.2.1	The robot shall be capable of generating a map of its surroundings in real-time during operation.	Real-time map navigation updates.	Test, observe, and record whether the map remains active without any crashes or black screens during a 60-minute period.	MingXiang	
P-7	3.2.2	The robot shall be able to autonomously plan navigation paths based on the generated map.	The robot moves according to the planned path.	Test, observe and compare the robot's actual movement path with the planned path.	MingXiang	
P-8	3.3.1	The probability of the robot navigating the path without moving obstacles shall be greater than 90 percent.	Obstacle avoidance success rate reaches 90 per cent or higher.	Test the robot in different obstacle scenarios, recording the number of successful and failed obstacle avoidance attempts.	MingXiang	

Requirement No.[A]	Section[B]	Shall Statement[C]	Verification Success Criteria[D]	Verification Method[E]	Responsible Party[F]	Results[G]
P-9	3.3.2	The robot shall be capable of detecting obstacles within a one-meter diameter range, including those that may be lower than the robot's frame, with a false detection rate of less than 5 per cent.	A. The robot can detect all obstacles within a 1-meter diameter range. B. The false detection rate is lower than 5 per cent.	Test the robot in various one-meter obstacle scenarios, recording the number of false detections and missed obstacles.	MingXiang	
P-10	3.4.1	In conditions of fixed lighting and color temperature, the robot shall be able to recognize objects based on color.	Target recognition success rate reaches 90 percent or higher.	Use cubes of different colors, recording the number of successful color recognitions and calculating the recognition success rate.	MingXiang	
P-11	3.4.2	The robot shall be able to match recognized objects of the same color.	The robot can correctly match colors.	Conduct multiple experiments with objects of different colors to observe whether the robot can successfully match them.	Zhen Yang	
P-12	3.4.3	The robot shall be able to display the quantity and color of recognized objects on the laptop screen in real-time	The color and quantity of recognized objects are continuously printed on the terminal.	During robot testing, record the terminal status information.	Zhen Yang	

Requirement No.[A]	Section[B]	Shall Statement[C]	Verification Success Criteria[D]	Verification Method[E]	Responsible Party[F]	Results[G]
P-13	3.5.1	The robotic arm shall be able to move to the target position within 10 seconds	After recognizing the target, the robotic arm moves to the target coordinates within 10 seconds.	During the robot's target recognition and grabbing tests, record the time it takes for the robotic arm to move and reach the target position.	Zhen Yang	
P-14	3.5.2	The robotic arm shall be capable of grasping polygonal objects, with the ability to grasp items weighing less than 50 grams at one time.	The success rate of grabbing a multi-sided cube is greater than 90 percent	Test the robot by grabbing cubes of different shapes, recording the success rate of the grab.	Zhen Yang	
P-15	3.5.3	The robot shall be able to place the grasped objects at designated locations, with a drop rate of less than 5 per cent during the process	The drop rate when the robot places an object is less than 5 per cent	After grabbing cubes of different shapes, place them in the designated position, and record the placement success rate.	Zhen Yang	
P-16	3.6.1	The robot shall provide real-time feedback on its operational status: gripping, recognizing, and avoiding obstacles	The robot's operational status is continuously printed on the terminal.	During robot testing, record the terminal status information.	Yu-Chuan	

Requirement No.[A]	Section[B]	Shall Statement[C]	Verification Success Criteria[D]	Verification Method[E]	Responsible Party[F]	Results[G]
P-18	3.7.1	The dimensions of the robot shall not exceed 500 mm x 500 mm x 500 mm	The robot's size is less than 500 mm x 500 mm x 500 mm.	Measure the robot's length, width, and height.	Yu-Chuan	
P-19	3.7.3	The robot shall be able to operate for at least one hour when fully charged	The robot can operate for more than one hour on a full charge.	Test whether the robot can continuously run for more than one hour in the laboratory environment with a full charge.	Yu-Chuan	
P-20	3.8.1	In subsequent designs, laser cutting and 3D printing shall not exceed the manufacturing budget.	The cost does not exceed the minimum cost threshold.	Calculate the total cost.	Yu-Chuan	

A. Unique identifier for each Robotic System requirement.
B. Section number each unique Robotic System Requirement is contained within.
C. Text (within reason) of the Robotic System requirement, i.e., the "shall".
D. Success criteria for the Robotic System requirement.
E. Verification method for the Robotic System requirement (analysis, inspection, demonstration, or test).
F. Responsible person for performing the verification.
G. Indicate documents that contain the objective evidence that the requirement was satisfied.

B Updated Workplace Charter

Our Purpose: This Workplace Charter sets out the principles, rules, and conflict management processes that Team 5 will follow throughout the Robotic Systems Design Project. The Charter aims to ensure that all members contribute effectively while maintaining a respectful and inclusive environment.

Our Principles and Commitments:

- **Equity:**We are dedicated to ensuring that each team member is treated fairly and provided with the same opportunities for success.
- **Diversity:**Our team values the variety of backgrounds, experiences, and perspectives that each member brings. We encourage all team members to contribute their unique viewpoints, enriching the overall project.
- **Inclusion:** Open dialogue will be encouraged so that everyone's opinions are acknowledged and respected, working collaboratively towards common goals.
- **Accessibility:**We are committed to ensuring that all team resources, meetings, and activities are accessible to every member.

Team Rules:

- All communication within the team will be conducted in English to ensure clarity and consistency.
- Standard working hours will be 9:00 AM to 6:00 PM on workdays. Work on weekends will only occur under special circumstances, with prior team agreement.
- The team will hold a regular weekly meeting every Wednesday from 3:00 PM to 4:00 PM to review progress, discuss tasks, develop a work plan, and resolve any issues. Attendance at all team meetings is required. If a member cannot attend, they should notify the team at least 24 hours in advance.
- All team members must adhere to agreed deadlines to maintain project momentum.
- Teams will be the primary platform for formal discussions, document sharing, and project updates. WhatsApp and WeChat will be used for quick and informal exchanges.
- The team will actively seek input from each team member throughout all stages of the project to ensure that everyone's thoughts and perspectives are considered.
- All major decisions will be made through a voting process, with a simple majority ruling. In the event of a tie, the team leader will have the final say.
- Respect for the politics, religion, culture and beliefs of each member. Discrimination is strictly prohibited.
- If someone break the rules, The first step is for the group leader to communicate privately with the group members. If the group member has a negative attitude and is extremely uncooperative with the group, seek help from the professor.
- The team will regularly assess and update the EDIA policies to ensure that our efforts remain effective, inclusive, and aligned with the project goals and values.

Daily Activity Conflict Avoidance:

In every meeting, each team member's tasks and responsibilities will be clearly defined, and the meeting minutes will be taken on a rotating basis to ensure all team members share responsibility for documentation and accountability. Regularly check the work progress of team members and supervise each other. A friendly and supportive communication style will be encouraged, promoting a positive atmosphere where issues can be discussed openly and resolved quickly before they escalate.

Conflict Resolution Flowchart:

When a conflict occurs, a judgement is first made about the type of conflict. Our team categorizes three types of conflict: communication and misunderstanding, work issues and, most seriously, issues such as discrimination and bullying. Different solutions apply to different types of conflicts. The process is illustrated in the figure below:

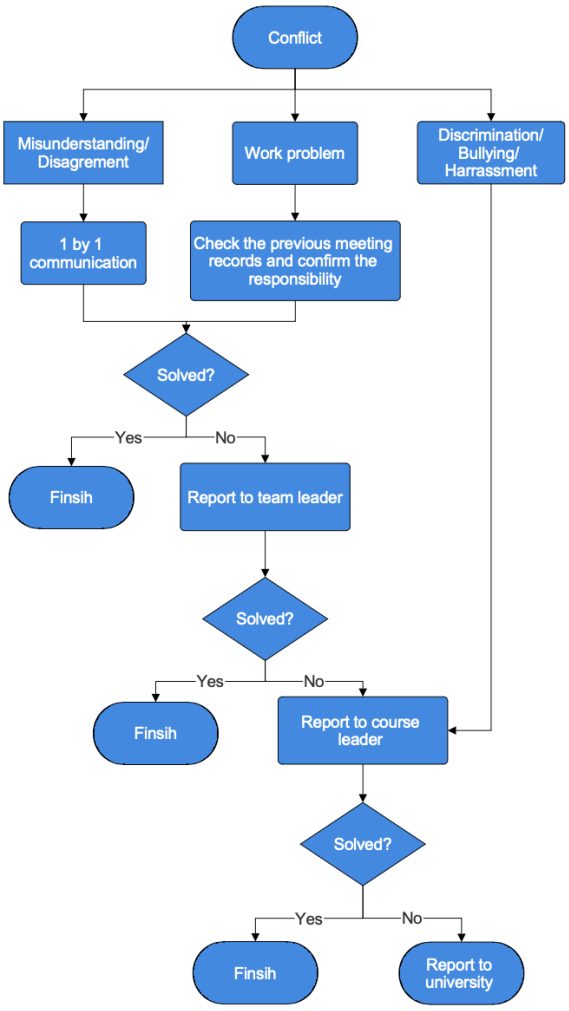


Fig. 31. Conflict Resolution Flowchart

References

[1] BS8622 Guide to Robot Sustainability, `https://www.bsigroup.com`