MANCHESTER 1824
The University of Manchester

Final Design Review

Final Design Review of the robotic system design project submitted to
The University of Manchester for the degree of
Master of Science in Robotics
in the Faculty of Science and Engineering

Year of submission
2024

Student ID
Team 7
11429985
10705876
11428365
11350007

School of Engineering

Contents

Declaration of originality

I hereby confirm that this dissertation is my own original work unless referenced clearly to the contrary, and that no portion of the work referred to in the dissertation has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Intellectual property statement

i The author of this thesis (including any appendices and/or schedules to this thesis) owns certain copyright or related rights in it (the "Copyright") and s/he has given The University of Manchester certain rights to use such Copyright, including for administrative purposes.

ii Copies of this thesis, either in full or in extracts and whether in hard or electronic copy, may be made *only* in accordance with the Copyright, Designs and Patents Act 1988 (as amended) and regulations issued under it or, where appropriate, in accordance with licensing agreements which the University has from time to time. This page must form part of any such copies made.

iii The ownership of certain Copyright, patents, designs, trademarks and other intellectual property (the "Intellectual Property") and any reproductions of copyright works in the thesis, for example graphs and tables ("Reproductions"), which may be described in this thesis, may not be owned by the author and may be owned by third parties. Such Intellectual Property and Reproductions cannot and must not be made available for use without the prior written permission of the owner(s) of the relevant Intellectual Property and/or Reproductions.

iv Further information on the conditions under which disclosure, publication and commercialisation of this thesis, the Copyright and any Intellectual Property and/or Reproductions described in it may take place is available in the University IP Policy (see `http://documents.manchester.ac.uk/DocuInfo.aspx?DocID=24420`), in any relevant Dissertation restriction declarations deposited in the University Library, and The University Library's regulations (see `http://www.library.manchester.ac.uk/about/regulations/_files/Library-regulations.pdf`).

1 Introduction

1.1 Problem Statement

Design and develop an autonomous robot to retrieve a target object within an unmapped static room to enhance efficiency and safety in tasks requiring autonomous object retrieval.

1.2 Objectives

- Establish a Robust Simulation Framework: Develop proficiency in utilizing the Gazebo simulator and RViz visualization tools, employing the provided manipulator and Leo Rover files. Ensure a comprehensive understanding of the simulation environment by thoroughly reviewing associated README files.
- Integrate Sensor Data for Monitoring and Analysis: Connect various rover feeds, including camera, odometry, and map topics, to RViz for real-time monitoring and analysis. Emphasize the critical role of sensor data in both teleoperation and autonomous navigation.
- Implement SLAM Algorithms for Mapping: Initiate Simultaneous Localization and Mapping (SLAM) algorithms during teleoperation to map the unknown workspace. Analyze the impact of obstacles, evaluate different mapping algorithms, and conduct a comparative study to inform the selection of the most effective approach.
- Transition to Autonomous Navigation: Progress from teleoperation to autonomous navigation using a SLAM-based approach. Conduct extensive tests to fine-tune navigation stack parameters, considering the specific structure and sizing of the Leo Rover in relation to the unmapped static room.
- Facilitate Iterative Optimization: Engage in an iterative process of testing and optimizing various parameters within the navigation stack. Adapt code based on the unique features of the Leo Rover and the unmapped static room to continually enhance the performance of the autonomous navigation system.
- Explore and Modify Model Files for Versatility: Demonstrate flexibility and exploration by modifying model files to test scenarios beyond the customer's requirements. Ensure adaptability to unforeseen challenges and a proactive approach to refining the system.
- Validate Secondary Libraries for Additional Functionality: Incorporate additional components like the px150 robotic arm, depth camera, and LiDAR for autonomous traversal and object retrieval.
- Utilize Simulation for Baseline Verification: Leverage the simulation as a baseline to verify algorithms and approaches. Ensure the readiness of the system by validating its performance in a controlled environment closely resembling a similar unmapped static room.

- Holistic Preparation for Tasks: Address both primary (navigation) and secondary (object retrieval) tasks within the simulation environment. Conduct thorough tests, including path planning algorithms and various odometry source combinations, to ensure a comprehensive and well-prepared system.
- Document and Analyze Simulation Results: Maintain detailed documentation of simulation tests, results, and observations. Analyze the outcomes to inform further refinements, and utilize the simulation as a valuable tool for continuous improvement in preparation for real-world applications.

1.3 Summary of Addressing PDR Feedback

Based on PDR feedback, there are still many imperfections in the report that were realised. These issues are addressed with the following responses.

1. System overview
    - Component description: Component diagram is improved, including how components communicate with each other and physical methods, ensuring that interactions between components are clear and unambiguous.
2. Electrical system
    - Power: Volages and current have been addressed in the diagram and the battery's ability to provide the required power for the peripherals was discussed in more detail.
3. Mechanical design
    - Payload container: the size of the container was discussed and additional images of the robot were added in the mechanical design section.
4. Software systems
    - RQT graph: Enhanced the RQT graph for better readability through color-coding and breakdowns.
    - ROS/packages: Identified and documented additional ROS packages essential for object detection, SLAM and manipulator control.
    - Mission planning through software: Developed and included a preliminary decision tree for mission planning in the software design.
    - Git documentation: Improved Git documentation with a clear folder structure and user guide.
5. Design requirements analysis
    - Fulfilled requirements: Added applicable software to examine and analyse the results of stage experiments.

1.4 Summary of Modifications

This section provides a summary of the sections that have been modified in response to PDR feedback. Includes all significant changes made since the PDR, whether or not they stemmed directly from PDR feedback.

1. System overview: Illustrated how the different peripherals and sensors communicate with each other in the Component diagram making the component diagram clearer and more consistent with the actual implementation.
2. Electrical system: The provided voltages and current were discussed in more detail and the battery was found to be able to power the majority of the tested sensors, however, there is an under-voltage when trying to power the NUC which will be investigated further during testing.
3. Mechanical design: The primary design of the robot was changed from having the depth camera on the manipulator to having it fixed on the upper plate of the robot. the camera attachment for the manipulator will be used as a second configuration for the robot as discussed in the mechanical design section.
4. Software systems: Updated RQT diagrams by making them color-coated for greater clarity and readability. Added all the necessary dependencies for the ROS package and Decision Tree sections, updated GitHub content, and added a new repository for the project with detailed guidance notes. created a separate section that explains robot navigation.
5. Design requirements analysis: Evaluate and analyze experimental results using more possible software where applicable. Updated the verification matrix to reflect the current progress of the project.

2 Sustainability Checklist

Sustainability revolves around achieving equilibrium between economic development, environmental care, and social equity [1]. There are several aspects to be considered when designing new products to achieve more sustainable solutions, some of which will be discussed below:

2.1 Materials

ABS Plastic was used as a 3D printing material for the sensors' mounts and supporting pillars as it offers several pathways for sustainable utilization. Firstly, its recyclability, as ABS can be reprocessed and reused multiple times without significant loss of its mechanical properties. Additionally, ABS-based products can have extended lifecycles, reducing the need for frequent replacements [2]. Lastly, selecting recycled or bio-based ABS variants contributes to a more sustainable approach in additive manufacturing [2].

Similarly, Acrylic sheets were used for the base plates to support the manipulator and the onboard PC as it provides several advantages. Firstly, its durability and structural rigidity make it suitable for long-lasting applications which reduces the need for frequent replacements and minimises waste. Secondly, acrylic is highly recyclable. When properly collected and processed, it can be reprocessed into new acrylic sheets or other products[3].

In the future, further exploration of sustainable and bio-based alternatives can be pursued to offer environmentally-friendly solutions.

2.2 Software

By refining software algorithms, it is possible to minimise computational overhead. Primarily, open-source packages were used and the developed code was also published as open-source to increase data accessibility. For object detection, an efficient training technique, with a mean average precision of 83.41%, was used, thereby reducing the needed computational power [4]. Additionally, the robot incorporates onboard processing for data analysis, reducing reliance on external servers, with all processed data stored locally. Pipelining is also implemented to carry out processes parallelly to minimize computation time. As for power management, we are implementing power-saving modes during idle periods or low activity will make the design more sustainable.

2.3 Energy & Waste

The main power source for the robot is the Leo Rover's rechargeable lithium-ion battery. When the robot starts its mission, the manipulator will be in its rest mode until the object is detected to minimise power consumption. Once the object is detected, the robot will grasp the target using the manipulator, store it in the payload container then return to its rest position while not in use. For a more sustainable design, Integrate energy-harvesting mechanisms such as a regenerative braking system to recover energy during operation can be considered in the future.

As the robot is fully electric, it is more environmentally friendly and produces minimal pollution (e.g. air and sound pollution) resulting in zero CO_2 emissions.

The robot uses of-the-shell components regulated under the Electromagnetic Compatibility Regulations 2016 [5] which in turn implements the EU Directive (2014/30/EU) on electromagnetic compatibility. This directive ensures that electrical and electronic equipment placed on the market meets essential requirements related to electromagnetic disturbance and immunity.

According to the study on smart EV charging, It would be more suitable to charge the battery during off-peak hours when electricity demand is lower. By doing so, it will reduce the contribution to peak-time strain on the grid and utilize cleaner energy sources [6].

2.4 Communications

As all the processing takes place on the robot onboard PC, only the control commands needed to start the robot, end the mission, and emergency stop will be sent to the robot. An online dashboard will be created to allow the user to monitor the robot's actions as well as observe the object detection model and the camera feed, in addition to a simple GUI to start and stop the feed. Additionally, ROS nodes are used to transmit and receive data across the system, and as the Intel NUC is connected to the Raspberry Pi using an ethernet connection, there is no need for data compression as the communications are fast enough for this application's purpose.

2.5 Modularity

This design places a strong emphasis on modularity as all the designed components were made to be swappable and can be reused in different parts of the robot. The camera mount, for example, was made so that it can either be used on the manipulator or as a fixed camera mount on the base plate. Many mounting holes were placed in the designed plates so that all the Leo rover's screws can be accessed without the need of dismantling the whole robot as discussed in section 5.

The robot's software is highly modular, it optimizes resource usage and reduces the environmental impact of software systems, especially in large-scale deployments. The ROS packages were first tested on a simulated robot and were easily repurposed onto the robot with minimal to no changes in the code. Modular software is designed and built with components that can be easily added, removed, or replaced without affecting the overall system. This promotes sustainability by enabling easier updates, maintenance, and scalability.

2.6 Location/Placement

The Robot is stored and tested indoors and onsite, therefore there is no need for transportation to an external location. However, due to the limited access to testing facilities which is set at a different part of the building, efficient time management is needed.

2.7 Maintenance

Regular inspections will be implemented to prevent mid-mission breakdowns. These inspections will include checking for wear, loose connections, and drift calibration. Additionally, a real-time diagnostic tool will be included on the dashboard to monitor performance and identify active ROS nodes to allow for easier error detection. As iterated earlier, modular software helps promote easier maintenance and this efficiently promotes sustainability.

2.8 Repurposing

At the end of this design project, the robot will be dismantled so it can be reused by the next MSc cohort. The end-of-life cycle of the designed components and the batteries in terms of recyclability and safe disposal will lay under the university's responsibility.

3 Cyber Security Considerations

1. Ensure encrypted communication: Encrypted communication between Leo Rover and the control system to prevent data theft during transmission.
2. Implement user authentication: Our robot only allows authorized users to control Leo Rover's functions such as movement, camera usage, and manipulator operations.
3. Regularly update code: The developers make sure to keep the Leo Rover's software updated to patch known security vulnerabilities and ensure a secure operating environment.
4. Utilize firewall protection: Employed a firewall to monitor and protect Leo Rover's network communications, safeguarding against unauthorized access and cyberattacks.
5. Plan for communication disruptions: Developed a plan to address disruptions in cyber communications, including procedures for quickly responding to user controls and restoring normal operations.
6. Maintain event logs: Logging all operational and security events to facilitate analysis and improve Leo Rover's cybersecurity posture over time.
7. Avoid default passwords: Encouraging the end users to set a strong, unique password and avoid default credentials to prevent unauthorized access to Leo Rover's systems.
8. Control data storage: Only store and control the data necessary for Leo Rover's operation and maintenance to minimize privacy risks associated with data handling tasks.
9. Enable secure remote access: Implementing secure remote access mechanisms, such as VPNs (Virtual Private Networks) or SSH (Secure Shell), to allow authorized personnel to remotely access Leo Rover's systems for maintenance and troubleshooting without compromising security.

4 System

This section provides a comprehensive breakdown of the Robot's components and the interaction between these components. Fig.1 shows the full assembled robot with all it's integrated peripherals.

Fig. 1. Assembled Robot

4.1 System Components

1. Leo Rover:
    - Type: Base Rover
    - Function: Felicitates traversal in the unmapped environment
2. Battery Power Box:
    - Type: Lithium-ion battery pack
    - Function: Provides the electrical power required to operate the robotic arm and its components
    - Specifications: 11.1V DC, 8A
    - Capacity: 5000 mAh
3. Intel NUC Computing Module:
    - Type: Intel NUC mini PC
    - Function: Serves as the central processing unit (CPU) of the robotic arm, handling sensor data processing, control algorithm execution, and communication with external devices
    - Specifications: Intel Core i5 processor, 4GB RAM, 500GB SSD

4. RPLiDAR A2M12 LiDAR Sensor:
    - Type: 2D laser rangefinder (LiDAR) sensor
    - Function: Provides accurate distance measurements to surrounding objects, enabling obstacle avoidance and navigation
    - Specifications: 360-degree scanning range, 20cm-16m detection range
5. PX150 Robotic Manipulator:
    - Type: 5-axis robotic arm
    - Function: Executes manipulation tasks (object retrieval), including grasping, moving, and placing objects
    - Specifications: 5 degrees of freedom (DOF), 0.45m reach, 0.05kg payload
6. Intel Realsense Depth Camera:
    - Type: RGB-D stereo camera
    - Function: Captures both depth and RGB information, enhancing object recognition and manipulation capabilities
    - Specifications: Depth resolution of 1280x720, RGB resolution of 1920x1080

4.2 System Interactions

1. The battery power box provides power to all system components and is connected using a Powerbox-to-battery cable which powers the Leo Rover.
2. The Intel NUC computing module processes sensor data robot's IMU and RPLiDAR A2M12 LiDAR sensor, generates control commands and communicates with the PX150 robotic manipulator and Intel Realsense Depth Camera to identify and grasp a target object. This communication takes place utilizing USB Micro-B connected to the RPLidar A12M12 and PincherX Robot Arm and a USB-C connection to the Intel RealSense Depth Camera.
3. The RPLiDAR A2M12 LiDAR sensor provides distance measurements to the Intel NUC computing module for obstacle avoidance and autonomous navigation.
4. The PX150 robotic manipulator executes control commands received from the Intel NUC computing module to perform manipulation tasks to grasp a defined object.
5. The Intel Realsense Depth Camera provides depth and RGB information to the Intel NUC computing module for enhanced object recognition, manipulation (object retrieval task) and is also employed in SLAM.

4.3 System Block Diagram

As observed from Fig.2 the Leo Rover robotic arm system combines sensor data acquisition, data processing, control algorithm execution, and robotic manipulation to achieve its primary objective of performing grasping, moving, and placing objects in a controlled manner. The system's components work together seamlessly to enable precise and efficient object retrieval tasks.

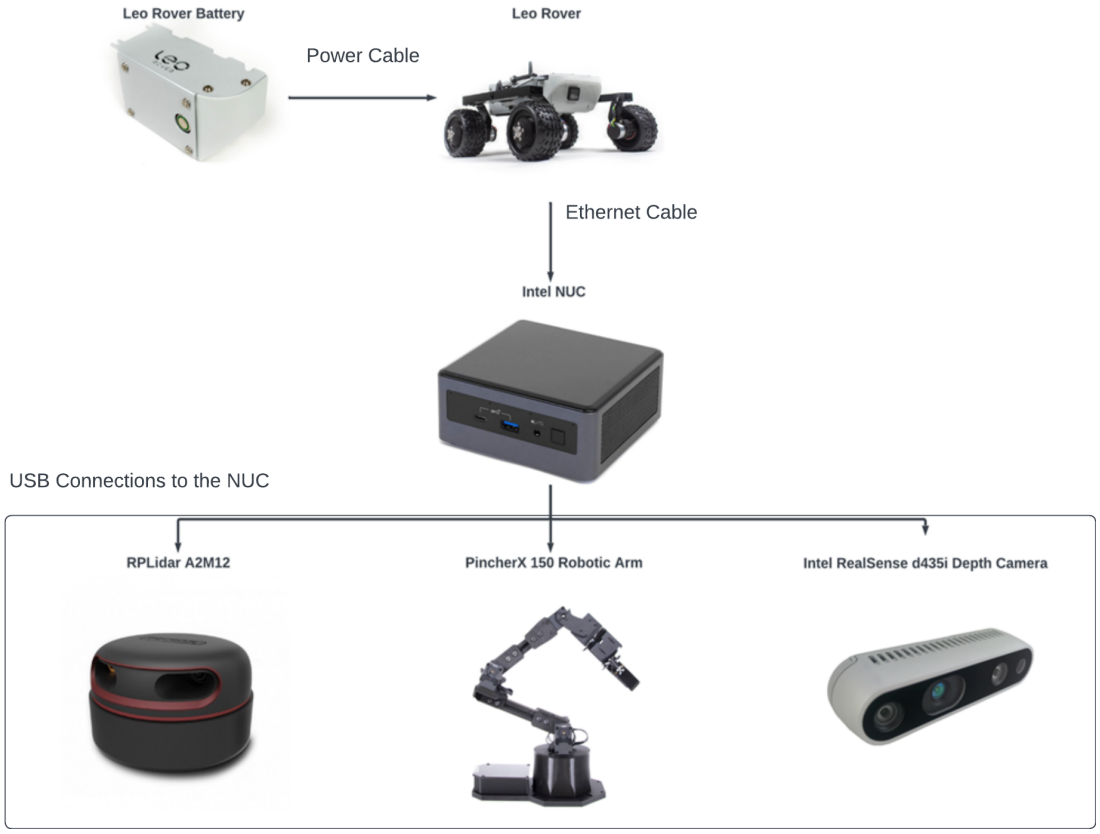


Fig. 2. System Block Diagram

5 Mechanical Design

5.1 3D CAD Model

The Robot payload sleds were designed such that the Manipulator is elevated above the LiDAR using four 3D-printed pillars. This will allow the LiDAR to have a clear front view while the back view will only be slightly obstructed by the pillars. As the two front pillars are closer to the LiDAR, their width was designed to be smaller than the back pillars to reduce the LiDAR's blind spots. Fig.3 below shows the main design configuration for the payload sleds.

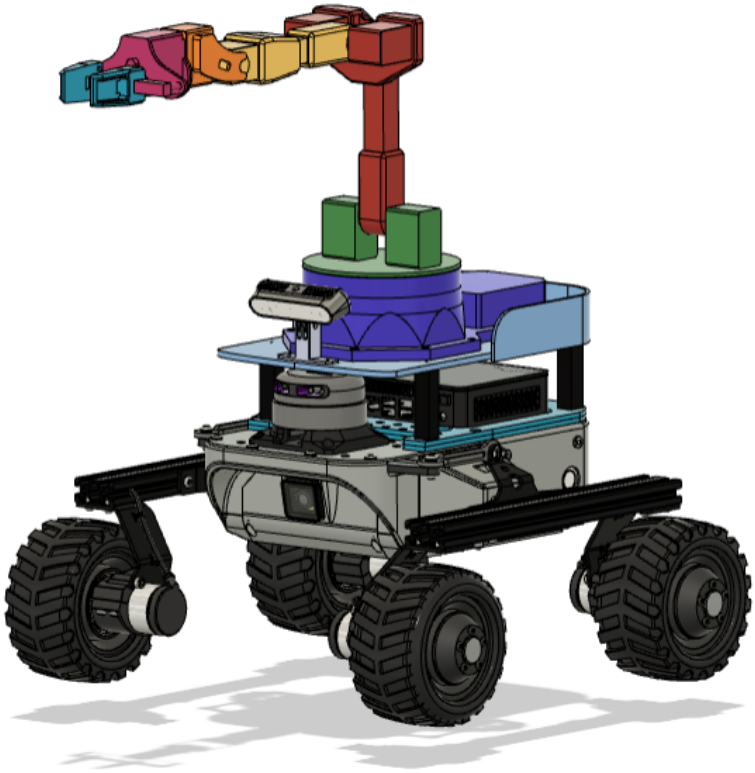


Fig. 3. CAD model of Robot with the payload sleds.

The base platform consists of two separate plates that will be joined together after laser-cutting to make the manufacturing process easier. The first plate will be used to make the surface of the robot leveled as well as to secure the four pillars using 2 mm screws. The second layer has four slots for the pillar to rest in to provide stability for the structure and prevent the pillars from moving or shifting while the robot is in operation. Fig.4 below shows the different components of the base plate.

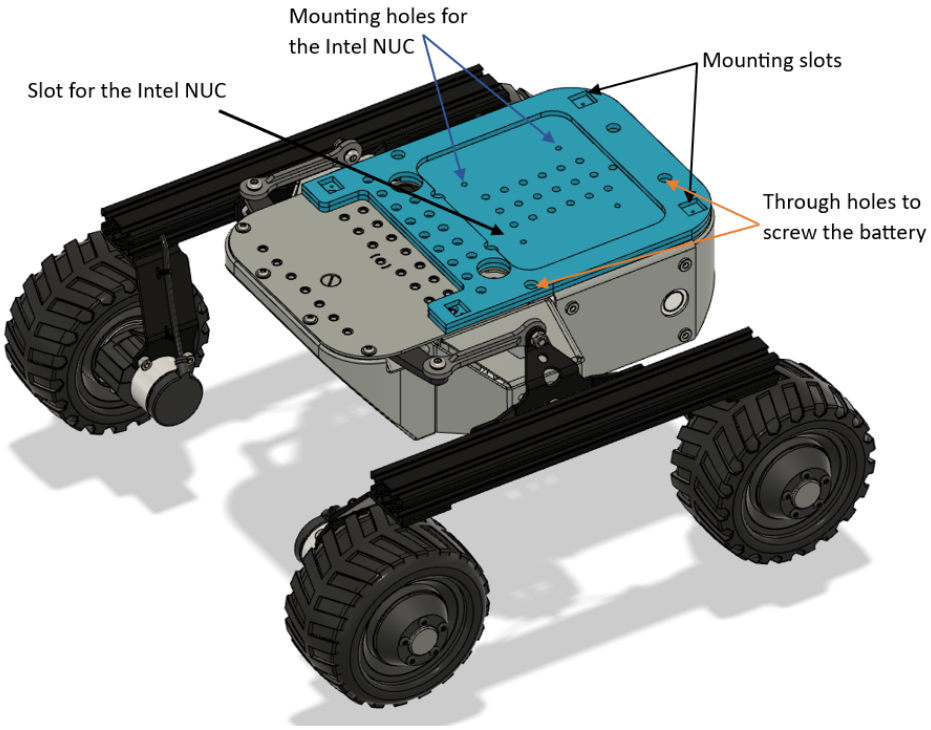


Fig. 4. CAD model of the base plate mounted on the robot.

A second platform is placed on top of the pillars for the Manipulator to rest on, it has a 40 mm edge that can be attached to act as a storage for the blocks while keeping the whole design symmetrical. The edge was also laser-cut from a piece of acrylic and bent using heat to match the outer profile of the upper platform. This configuration formed two storage areas, each measuring 40 mm x 40 mm x 90 mm, positioned on either side of the manipulator, as illustrated in Fig.5 below.

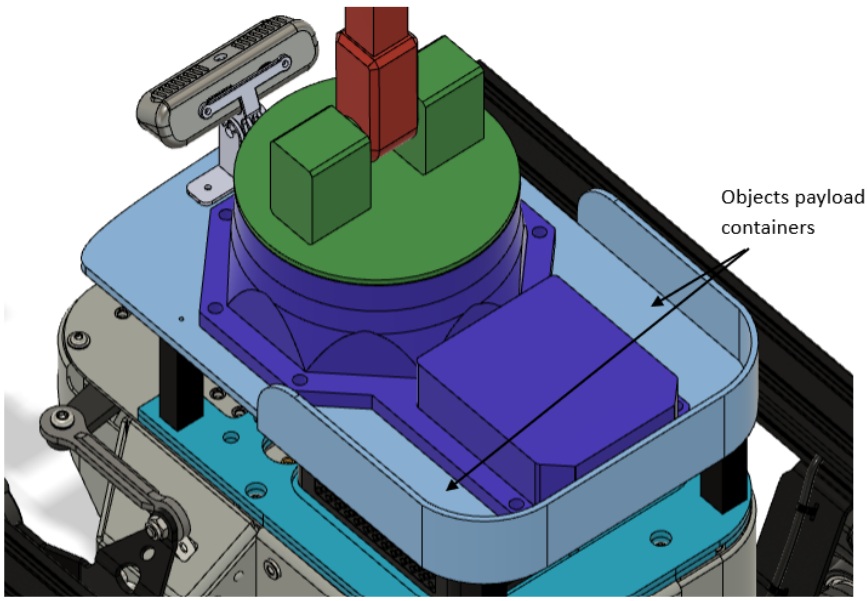


Fig. 5. CAD model of the upper plate showing the payload container

The main aim was to make the payload sled modular and repairable, and therefore, two camera mountings for the real-sense depth camera were designed. The first camera mount, which is the starting approach, is designed to be fixed on the second plate as shown in Fig. 6. This will make the transformation between the camera frame and the manipulator frame simpler and will enable the depth camera to detect the end effector of the manipulator and the target object at all times during operation. The camera mount also has marks of different angles so that the camera tilting angle can be changed when needed to allow for testing and improvement (with different camera view angles).

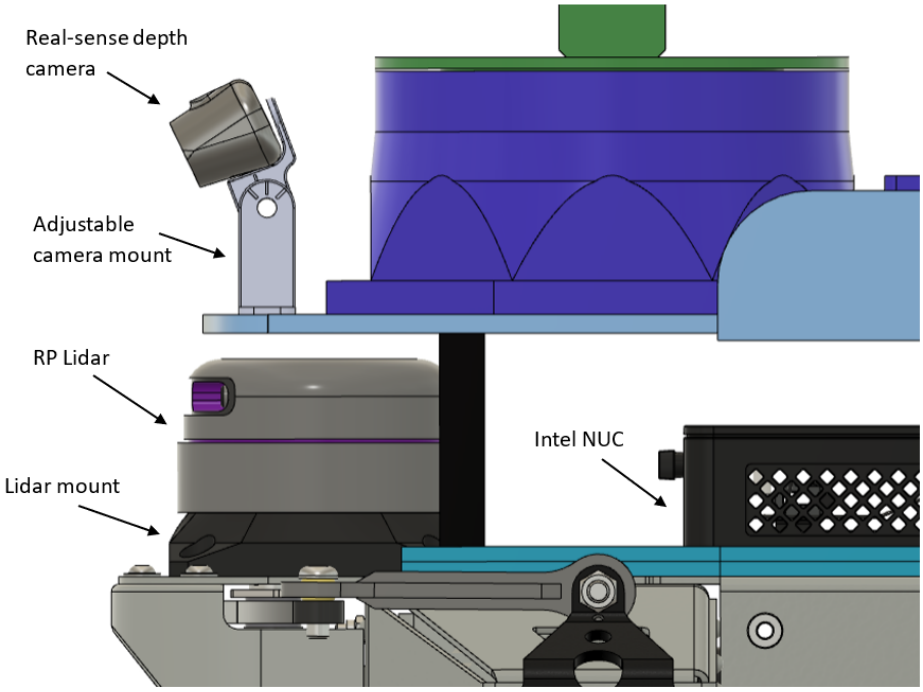


Fig. 6. CAD model of the robot with the depth camera fixed mount.

The second configuration has the camera mount attached to the Manipulator as shown in Fig. 7 below. This design allows the camera to move freely using the Manipulator so that the camera can scan the environment and the target object can be detected without the need for the robot to turn. Additionally, the top part of the camera holder can be repurposed and used in either the Manipulator or fixing it on the platform.

The task of object detection and grasping using the second configuration, shown in Fig.8 was found to be more challenging. This complexity arises from the depth camera's minimum range of 28 cm. In this setup, the robot must first detect the object and then blindly attempt to grasp it based solely on the pose provided by the camera, before it approaches too closely to the target. Notably, once the robot holds the object, it loses the ability to detect its position. As a result, the first configuration emerged as a preferable initial option for simpler and more reliable object handling.

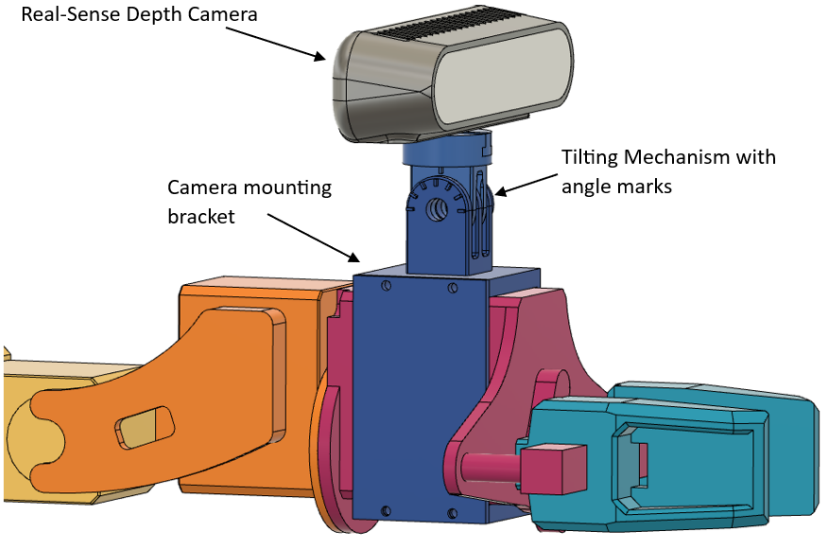


Fig. 7. Depth camera mount attachment for the Manipulator.

Lastly, the Intel NUC was mounted on the base platform to make the design compact. After assembling and testing the designed payload sleds, it was found that the NUC placement does not abstract the lidar field of view, therefore the designed backup case for the NUC can be repurposed as an additional payload container.

The LiDAR mount was imported from the Leo Rover website under the name of (07010 RPLiDAR A2M8 adapter)[7] and 3D printed in ABS plastic.

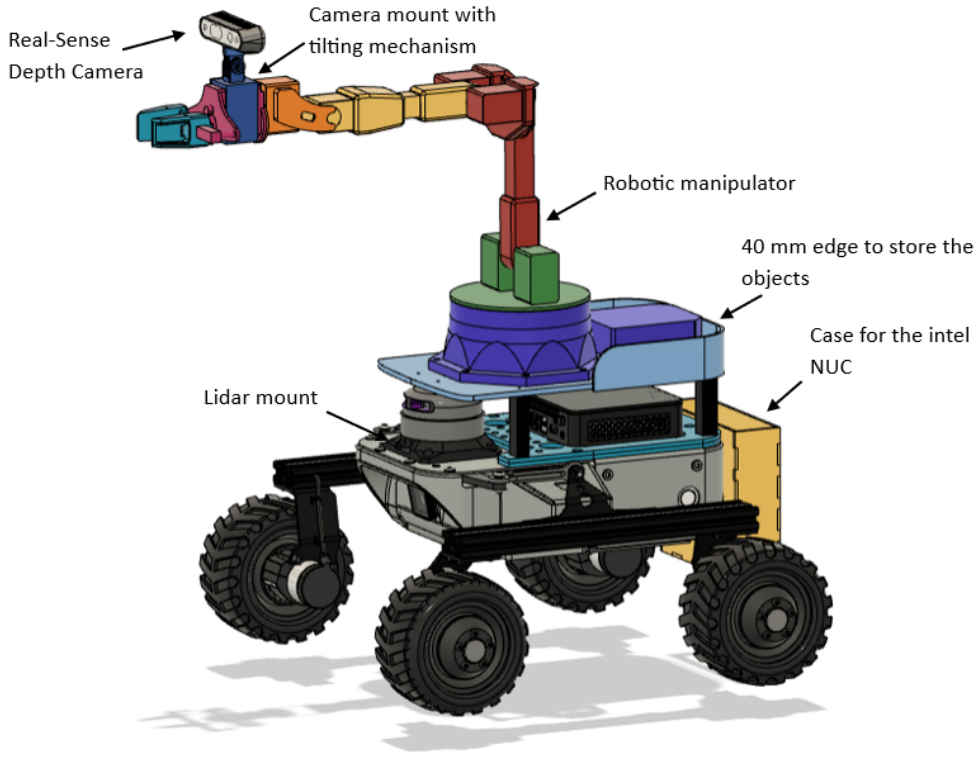
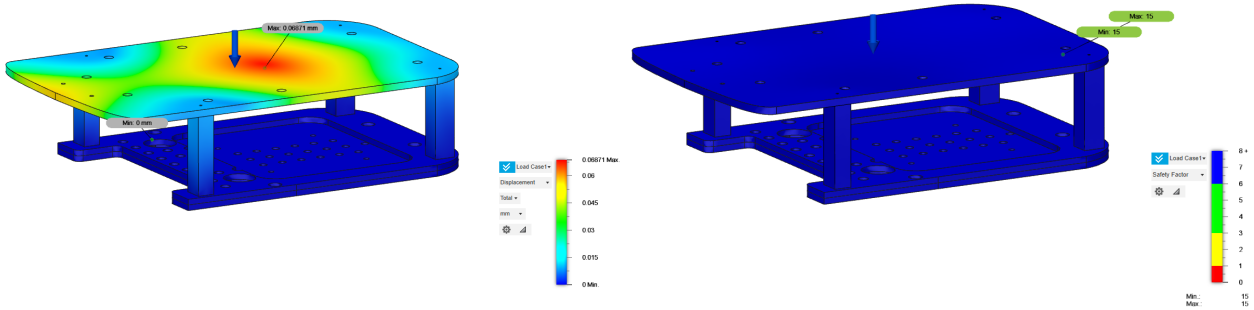


Fig. 8. The second configuration of the payload sleds.

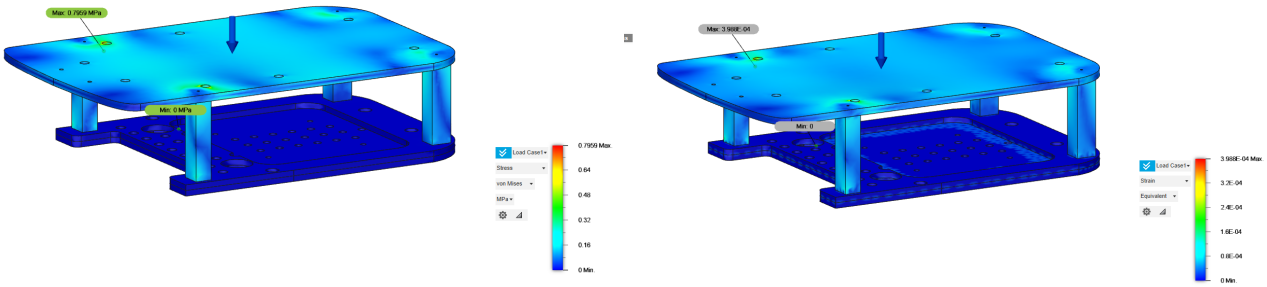
5.2 Stress Analysis

A stress analysis test was performed on the designed platform (payload sleds) to analyse how it might perform under real-life conditions and to predict any failures that might occur. Based on the manipulator and the depth camera weight in addition to the other designed components, a total of 1.85 kg or (18.142 N) was applied to the upper platform and simulated in Autodesk Fusion 360 CAD software as shown in Fig.9 and Fig.10 below. the martial of the plates was assigned as Acrylic whereas the pillars' martial was assigned as ABS plastic for 3D printing.



(a) Displacement **(b)** Safety factor

Fig. 9. Stress analysis results (displacement & safety factor)



(a) Stress **(b)** Strain

Fig. 10. Stress analysis results(Stress & Strain)

The stress analysis showed that the payload sled will be able to withstand the weight of the manipulator and the other sensors without any notable deformation, as Fig.9 above shows, the upper plate has a safety factor between 6 and 8 as well as a maximum displacement of 0.0687 mm.

The upper plate experiences a maximum stress of 0.7959 MPa and a maximum strain of 3.988 E-04 as shown in Fig.10.

5.3 Mechanical Drawing

The mechanical drawings of the Robot with the payload sleds as well as the base and upper platforms supported by the four pillars are shown in Fig.11 and Fig.12 respectively.

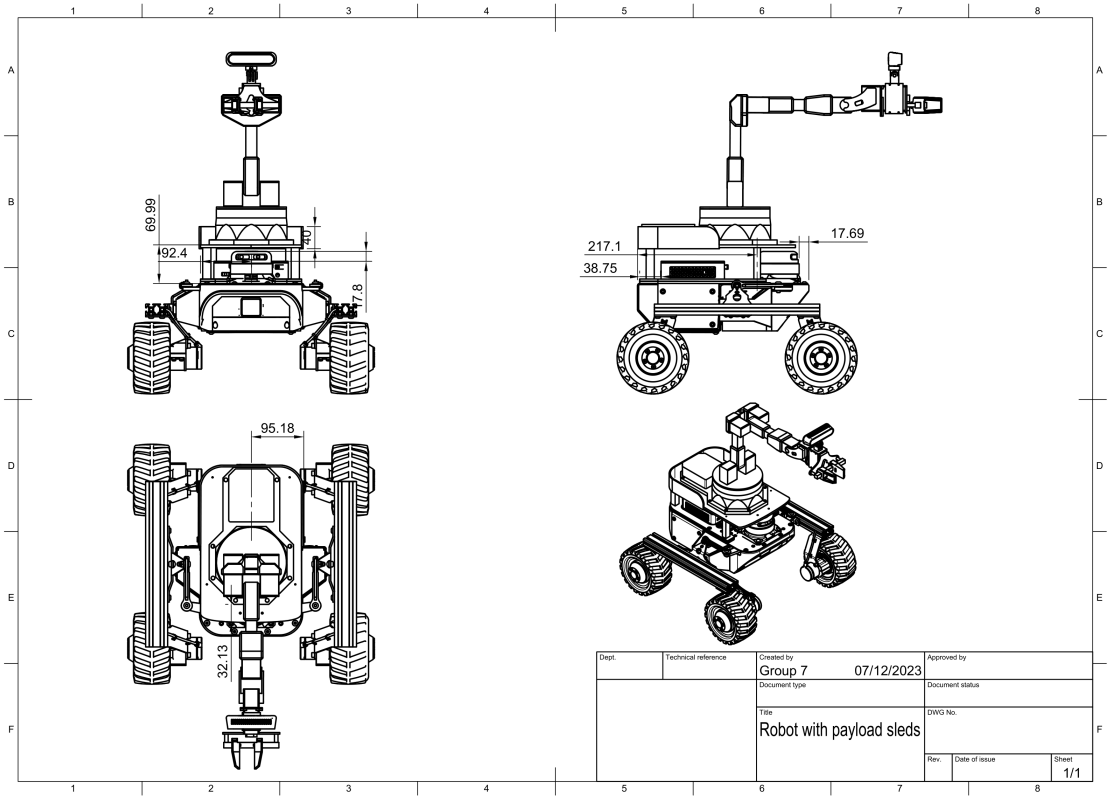


Fig. 11. Mechanical drawings of the Robot with the payload sleds

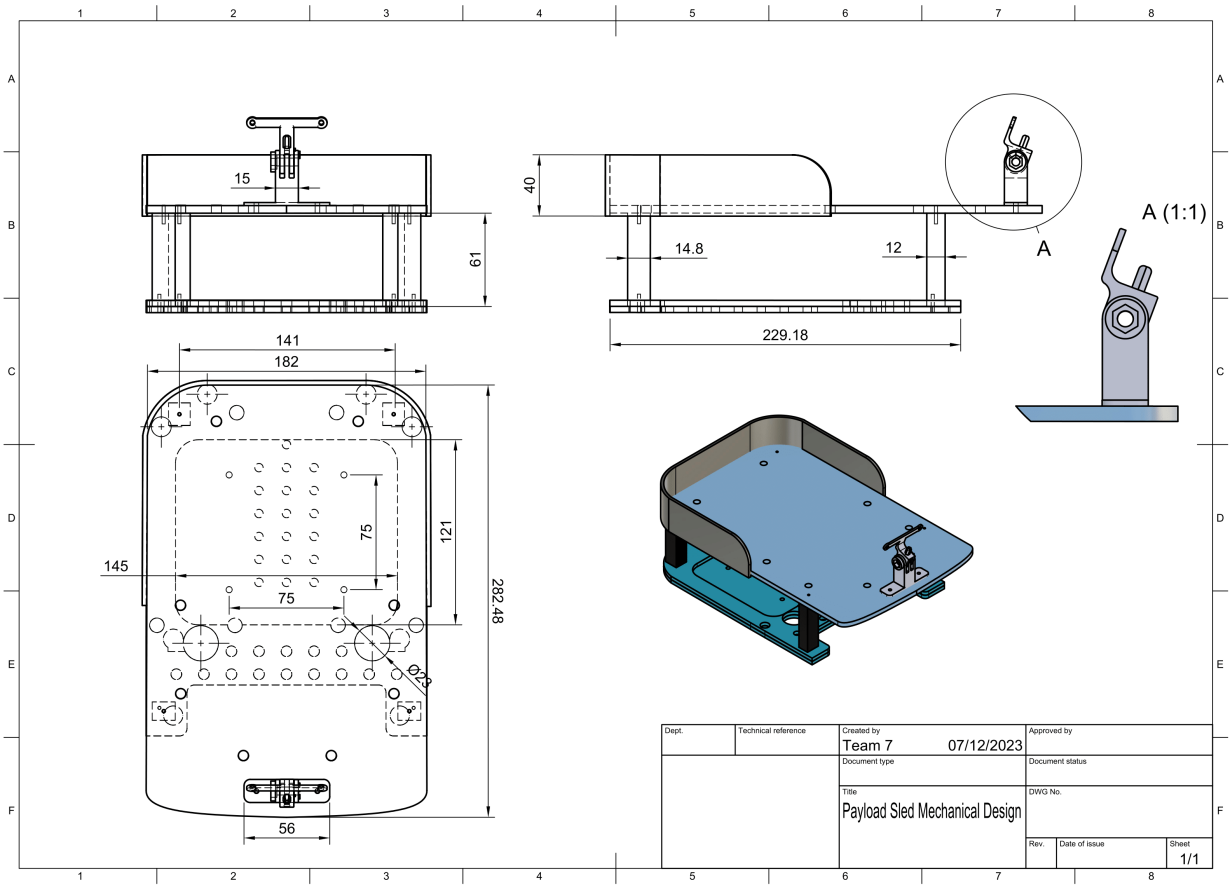


Fig. 12. Mechanical drawings of the payload sled platform

As for the mechanical drawings of the depth camera mount and the NUC Case, refer to Appendix C.

6 Electrical Design

The power connection diagram for the Leo Rover robotic system illustrates the distribution of electrical power across different components. It depicts the connections between the power sources, including the 11.1V Li-Ion battery, the PowerBox, and the various power-consuming devices, such as the Intel NUC, the Raspberry Pi, the robot arm, the LiDAR, the depth camera and the fisheye camera.

Power Sources:

1. 11.1V Li-Ion Battery: The primary power source for the Leo Rover is the 11.1V Li-Ion battery, which in turn supplies power to the PowerBox.
2. PowerBox: The PowerBox acts as the central power distribution unit, regulating the voltage and current from the 11.1V Li-Ion battery and providing power to the various components through either the 12V or 5V DC sockets.

Power Distribution and Data Transfer:

1. Intel NUC: The Intel NUC receives power from the Leo Rover's PowerBox. The NUC takes in data from the rover, the manipulator, the LiDAR and the depth camera.
2. Raspberry Pi 4B: The Raspberry Pi 4B draws power from the LeoCore Controller through its GPIO pins. An RJ45 cable (Ethernet) is connected to the Intel NUC for the Raspberry Pi to publish the robot topics to the Intel NUC.
3. Robot Arm: The PincherX 150 Robot Arm receives power from the PowerHub Board which is connected to the Leo rover's PowerBox. The ROBOTIS U2D2 Controller is used to send and receive data from the NUC and the arm.
4. LiDAR: The RPLiDAR A2M12 receives power from the Intel NUC, and it is connected through UART to a UART to USB adapter which in turn connects it to the Intel NUC. Data and power are transferred through this connection.
5. Fisheye Camera: The 5MPX Fisheye Camera is powered directly from the Raspberry Pi using a CSI/DSI cable, and this cable assists in sending camera data.
6. Robot Wheels: The Buhler BLDC Motors for the robot wheels receive power from the LeoCore Controller through 6-pin connections. The magnetic encoders attached to the motors provide feedback on their rotation.

Overall Power Management:

The Leo Rover's power connection diagram as observed in Fig.13 effectively distributes electrical power across the various components while maintaining proper voltage levels and ensuring efficient operation. The PowerBox plays a crucial role in regulating power from the main battery and providing clean power to the system's critical components. The battery has successfully been able to power all the sensors, however there are certain limitations in the integration of the intel NUC to be powered from the robot since there is a voltage drop from 12V to 10.7V when a direct supply is drawn from the battery. Alternate power sources are currently being explored to help mitigate this shortcoming.

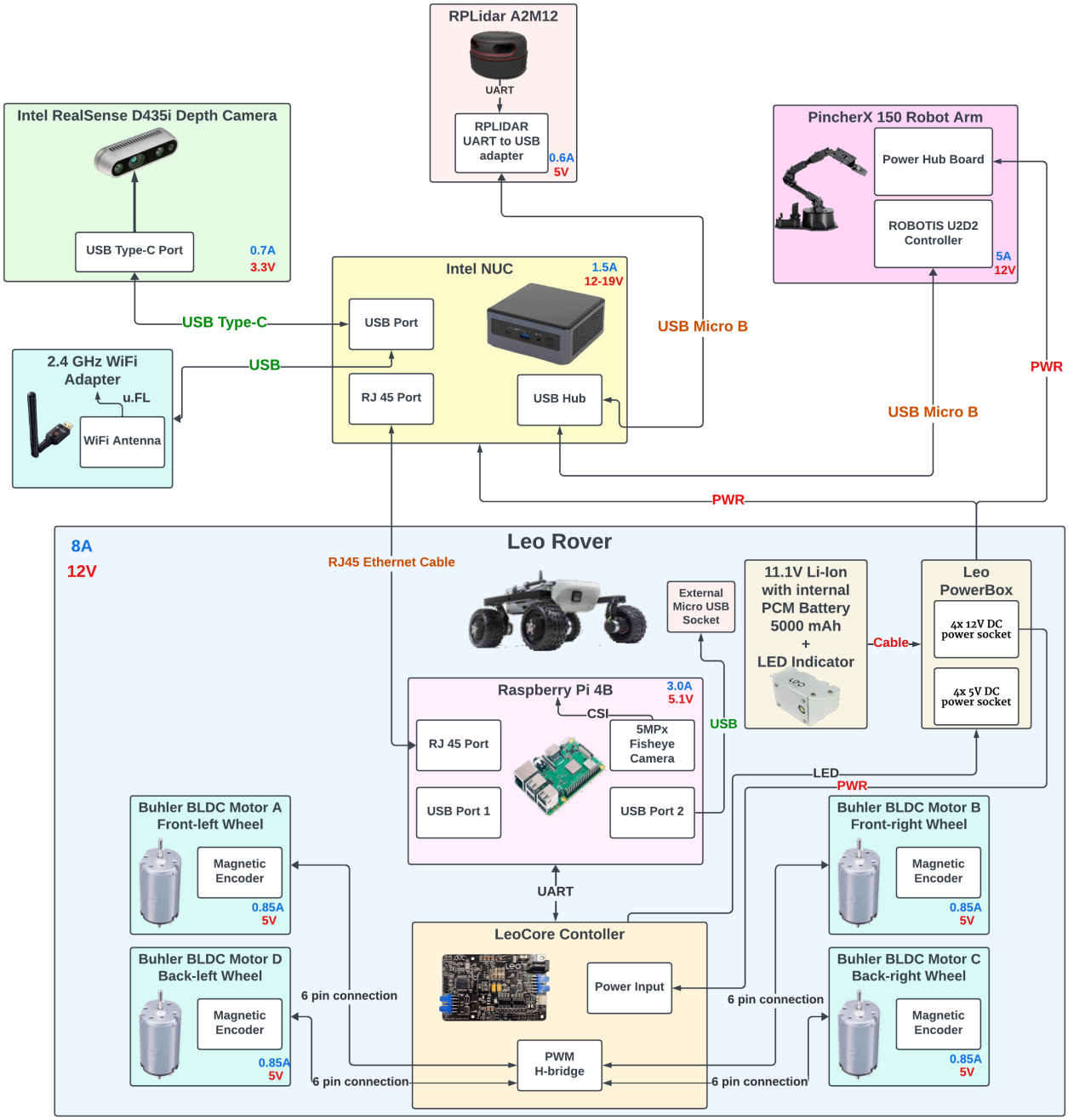


Fig. 13. Power Connection Diagram

7 Software Design

7.1 RQT Graph

In the Power Connection Diagram depicted in Fig.13, various connections have been established from the Intel NUC to different components of the robotic system, including the rover, manipulator, LiDAR, and depth camera. The communication and information exchange between these components are visualized in more detail through the representations in Fig.17 and Fig.18.

Fig.17 provides a comprehensive overview of the topics published by these connections to the NUC. Each connection corresponds to specific data streams, or ”topics,” that convey information about the state and operation of the respective components. The breakdown of topics is as follows:

- Rover topics: Under the **/firmware** node, the rover publishes information related to wheel states (**/firmware/wheel_states**), wheel odometry (**/firmware/wheel_odom**), inertial measurement unit (IMU) data (**/firmware/imu**), and joint states (**/joint_states**).
- RPLiDAR topic: The LiDAR component publishes scan data (**/scan**) through the **/rpLiDAR_node**.
- Manipulator topics: The manipulator component, represented by the **/px150** node, publishes joint states (**/px150/joint_states**) and robot description (**/px150/robot_description**).
- Depth Camera topics: The depth camera component, represented by the **/camera** node, publishes various image-related topics, including raw images (**/camera/image_raw**), monochrome images (**/camera/image_mono**), color images (**/camera/image_colour**), and additional information about the images (**/camera/image_info**).

Fig.18 further aids in understanding the relationship between these peripherals, the rover, and the NUC. The visualization highlights the distribution of topics on the **/param_events** parameter server. The left half of **/param_events** is covered with topics from the rover, while the right half is covered with topics from the manipulator, LiDAR, and depth camera. This spatial arrangement emphasizes the distinct data streams originating from each component and their contributions to the overall system.

In summary, the combination of Fig.17 and Fig.18 provides a detailed and organized representation of the communication patterns and data flow within the robotic system, enhancing the understanding of how information is exchanged between the Intel NUC and its connected components.

7.2 ROS/Packages

7.2.1 Build Dependencies:

- nav2_costmap_2d: This package is crucial for building 2D costmap functionality necessary for path planning and obstacle avoidance in robot navigation systems.

- nav2_msgs: Provides message types for communication between different components of the Navigation2 stack, aiding in the development of navigation-related functionalities.
- nav_msgs: Similar to nav2_msgs, this package provides message types for navigation tasks, facilitating communication between different nodes involved in navigation.
- rclcpp: The ROS Client Library for C++, essential for building ROS-based applications in C++. It provides tools and libraries for creating ROS nodes and handling communication within a ROS-based system.
- sensor_msgs: Contains message types for common sensor data, vital for processing sensory information in robotics applications.
- std_msgs: Provides standard message types for communication in ROS, serving as a fundamental dependency for ROS-based systems.
- tf2: Used for transforming between coordinate frames in ROS, managing spatial relationships between different components in a robot system.
- tf2_geometry_msgs: Extends tf2 to support geometric operations with ROS message types, facilitating the transformation of geometric data between different coordinate frames.
- tf2_ros: Provides ROS-specific functionality for working with tf2, including utilities for publishing and subscribing to transform data, essential for integrating tf2 functionality into ROS nodes.
- visualization_msgs: Contains message types for visualization purposes in ROS, allowing nodes to publish visual representations of data for debugging, monitoring, and visualization.
- realsense2_camera_msgs: Provides ROS message types specific to Intel RealSense cameras, enabling integration with RealSense cameras for applications such as depth sensing and visual odometry.
- rclpy: Similar to rclcpp, but for Python. It's essential for building ROS2-based applications using Python.
- cv_bridge: Connects OpenCV and ROS by converting image data between ROS topics and OpenCV formats, necessary for integrating ROS image data with OpenCV-based image processing algorithms.
- numpy: Essential for handling multidimensional arrays and matrix calculations, used for converting image and depth frame data from ROS topics into numpy arrays, facilitating subsequent image processing operations and 3D coordinate calculations.
- sys: Python's standard library module used for modifying module search paths, employed in the code for certain configuration tasks.
- yolov5: Utilized for training and testing object detection models. YOLOv5 offers fast inference speed and high detection accuracy, providing various model sizes to balance performance and accuracy.

- imu_filter_madgwick: This package provides an IMU (Inertial Measurement Unit) filter implementation based on Madgwick's algorithm. It's used for sensor fusion, specifically for fusing data from inertial sensors like accelerometers and gyroscopes to estimate orientation and position.
- imu_tools: This package provides tools for IMU-related tasks such as calibration and data visualization. It's useful for calibrating IMU sensors and visualizing their data for analysis.
- robot_localization: This package provides tools for sensor fusion and state estimation for robotic systems. It integrates data from multiple sensors (such as GPS, IMU, and wheel odometry) to estimate the state of a robot (position, orientation, velocity, etc.) more accurately than individual sensors alone.
- interbotix_xs_modules: This is a custom ROS package to control the manipulator. The ROS package builds upon the ROS driver nodes found in the interbotix_ros_core repository. This package enables the manipulator to execute commands that describe a predetermined trajectory.

7.2.2 Execution Dependencies:

- joint_state_publisher: Publishes joint state information, crucial for visualizing and controlling joint states of a robot in simulation or visualization tools like RViz.
- joint_state_publisher_gui: Provides a graphical user interface for interactively controlling joint states, useful during robot development and testing phases.
- robot_state_publisher: Publishes the transform tree for a robot model, allowing other nodes to determine spatial relationships between different parts of the robot, crucial for visualization and navigation tasks.
- rviz: A 3D visualization tool for ROS, used for visualizing sensor data, robot models, and other information in a 3D environment, commonly used for debugging, monitoring, and verifying the behavior of ROS-based systems.
- xacro: An XML macro language used in ROS for parametrized robot descriptions, enabling creation of modular and reusable robot descriptions by defining macros for commonly used components and parameters.

7.2.3 Other Dependencies:

- Image Dataset: The dataset was created by team members photographing target objects. The dataset includes images and classes. The images include training images (1473 images) and validation images (264 images). Classes include training classes (1473 training classes) and validation classes (264 validation classes). Classes and images are one-to-one, which ensures the accuracy of YOLOv5 training model.

7.3 Software Mission Planning

As seen in Fig.14 and Fig.15 frontier-based exploration is a strategy used by robots to navigate unknown environments. Here's how it has been repurposed onto the robot:

1. The robot identifies unexplored areas, called frontiers.
2. The robot prioritizes exploring the closest frontier.
3. As the robot explores, it updates its map of the environment.
4. This process repeats until there are no more frontiers to explore.

Frontier-based exploration is a common technique used in robotics because it is relatively simple and efficient. It is useful for tasks such as mapping an unknown environment and it also proves to be extremely helpful to search for a specific object in an unknown environment. We utilize these frontiers to explore the environment until we come across our target object that can be identified using our trained models. The RQT graph for the simulation can be observed in Fig.19

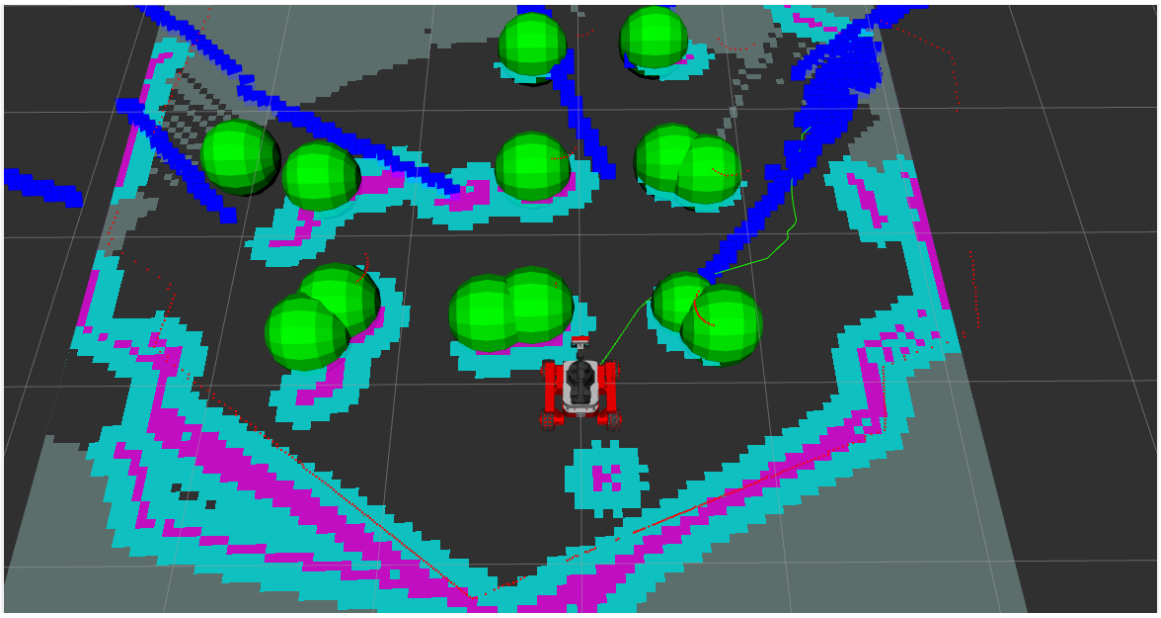


Fig. 14. The Simulated Robot using Frontier Based Exploration

We have also employed a decision tree and from Fig.16, we can observe that the decision tree is broken down into the following components:

7.3.1 Navigation Subtree:

The Navigation subtree is primarily responsible for managing the robot's movement through the environment. It encompasses the following key functionalities:

- Path Calculation: The ComputePathToPose action calculates a path from the robot's current position to a specified goal pose. This is achieved using the GridBased planner algorithm.

Fig. 15. Frontier Based Exploration implementation on the robot

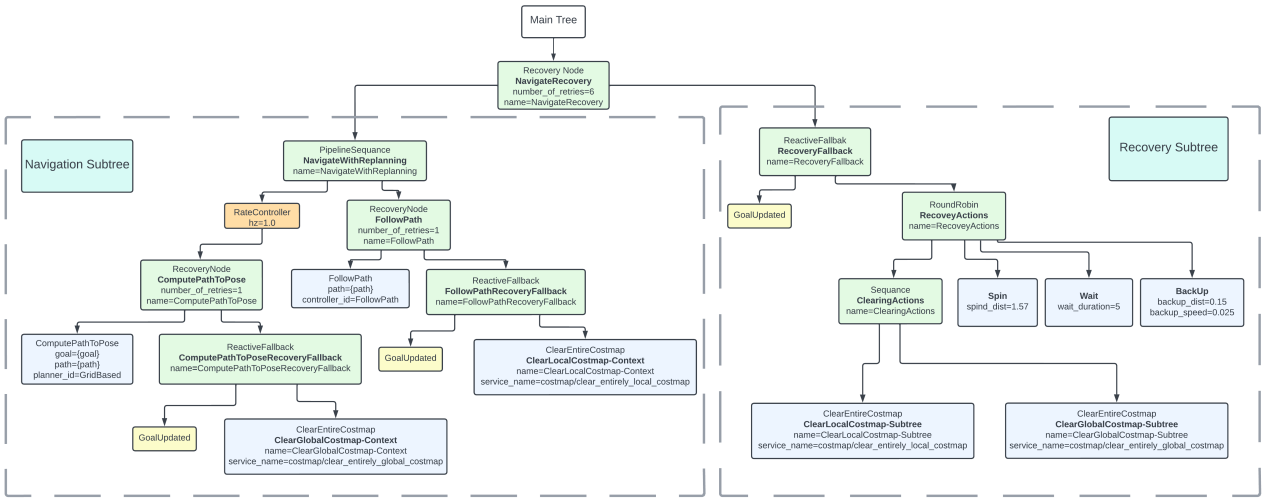


Fig. 16. The Robot using Frontier Based Exploration

- Path Following: The FollowPath action guides the robot along the computed path using a specified controller (FollowPath controller).
- Contextual Recoveries: If either the path calculation or path following fails, the subtree attempts contextual recoveries. These recoveries include checking if the goal has been updated and clearing the relevant costmap (global or local) to mitigate the failure contextually.

7.3.2 Recovery Subtree:

The Recovery subtree is activated when the Navigation subtree encounters failures beyond contextual recoveries. It addresses system-level failures and facilitates recovery actions to resume navigation. Key aspects include:

- Recovery Strategies: The Recovery subtree employs various recovery strategies to address system-level failures. These strategies include clearing both local and global costmaps, performing a spin action, waiting for a specified duration, and executing a backup action.
- ReactiveFallback Control: The top-level ReactiveFallback node orchestrates the execution of recovery strategies while asynchronously monitoring for goal updates. Upon receiving a new goal, all ongoing recovery actions are halted to prioritize navigation towards the updated goal.
- System-wide Recovery Attempts: The Recovery subtree continuously cycles through a set of recovery actions until successful navigation is achieved or the maximum number of retries is exceeded. If a recovery action succeeds, the robot transitions back to the Navigation subtree for path planning and execution.

7.4 Git Repository

The team's weekly progress is recorded in our GitHub repository, which can be accessed through the following link: `https://github.com/Team-7-UOM/DocumentationForLeoRover`.

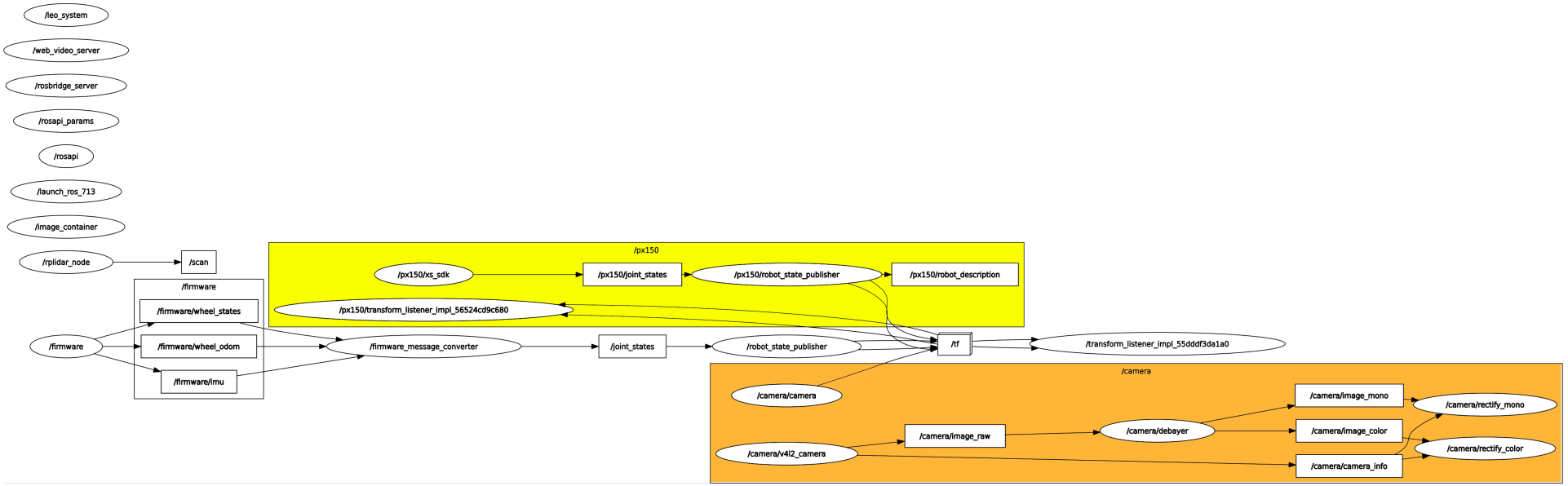


Fig. 17. RQT Graph

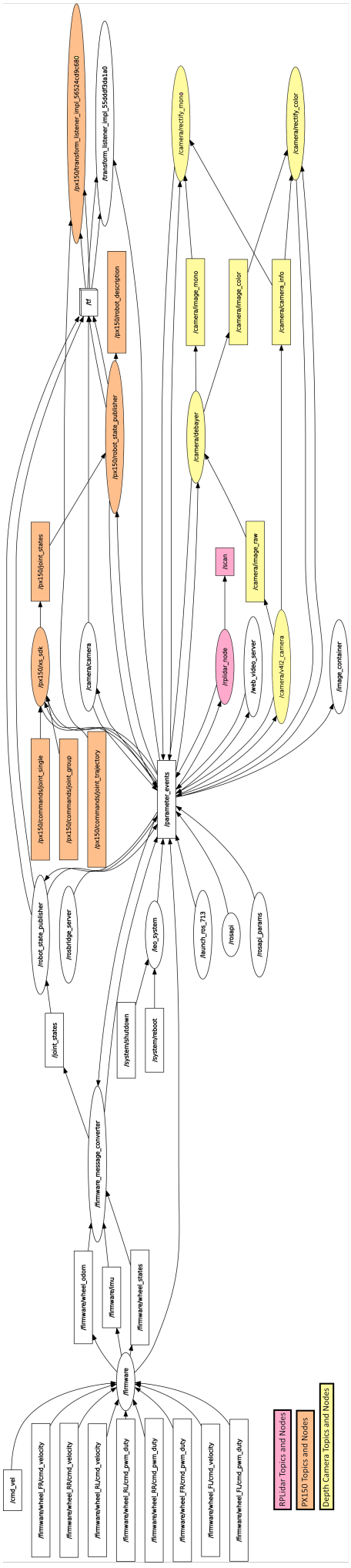


Fig. 18. RQT Graph with parameter events

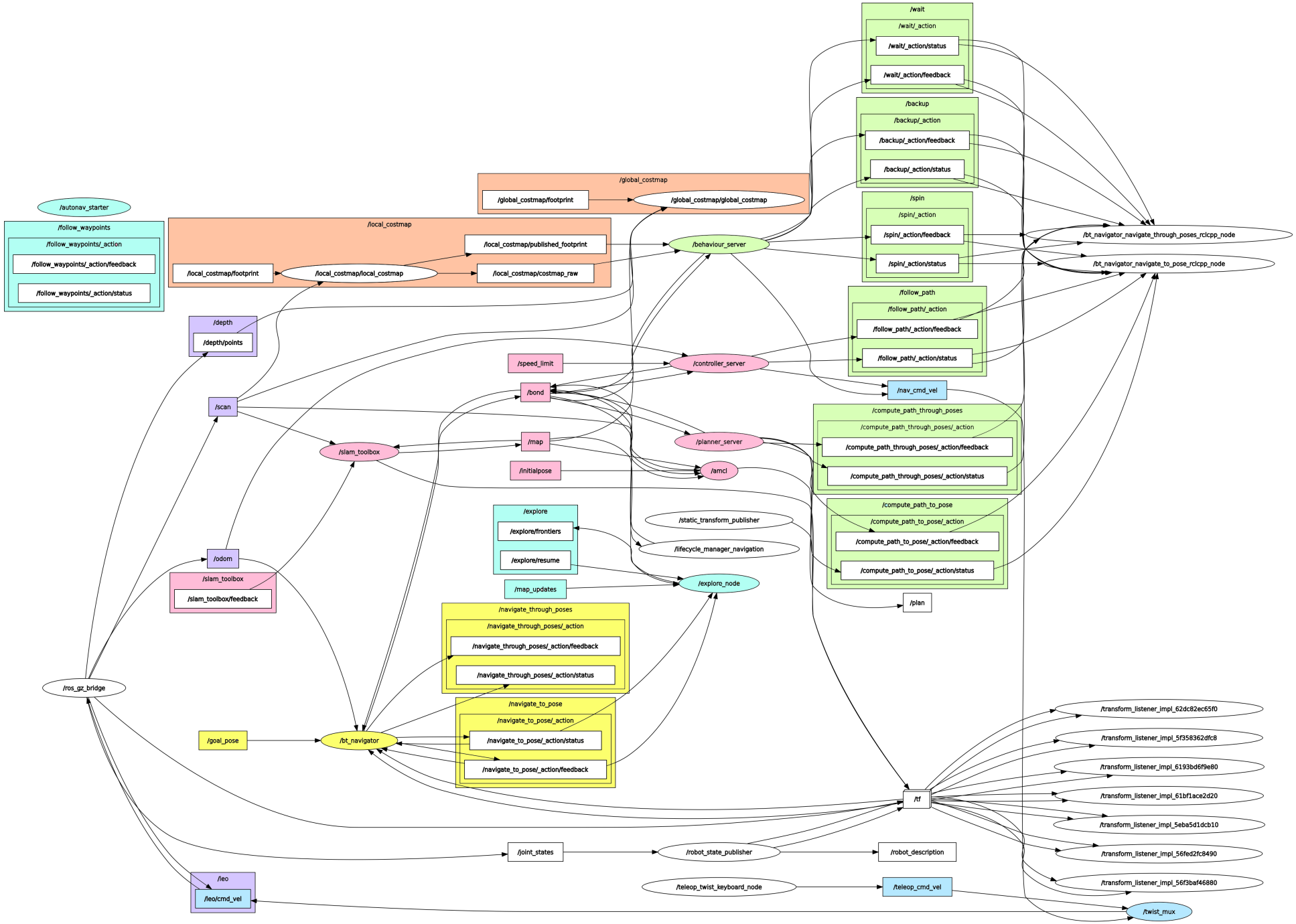


Fig. 19. RQT Graph for the simulated Robot

8 Analysis

8.1 Requirements Verification Matrix

Requirements Verification Matrix as observed in [8] and Table 1 is a tool used to ensure that project requirements are met. This matrix helps the team track requirements, ensuring that each requirement has gone through the proper verification process and corresponds to the corresponding verification methods.

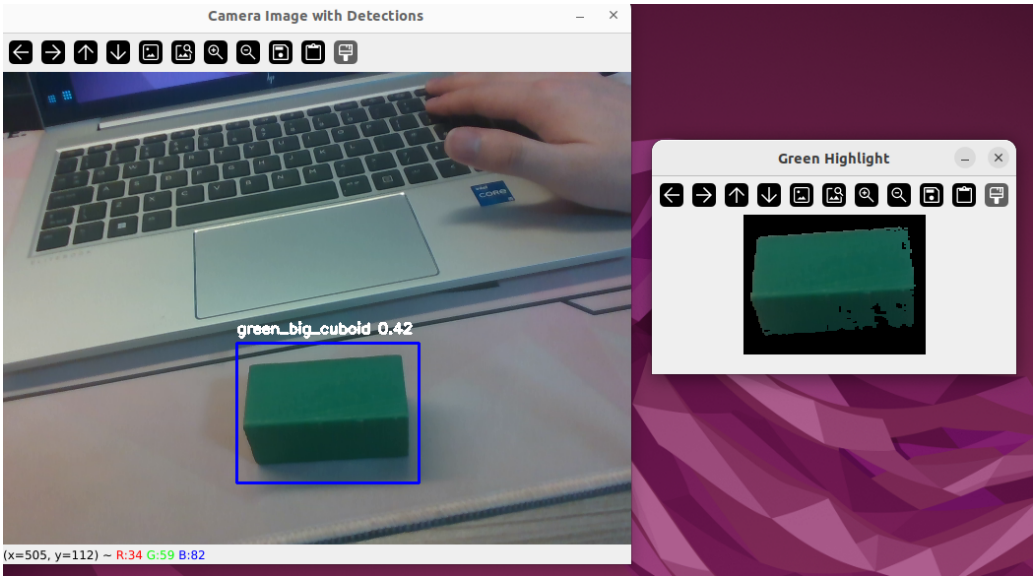
Table 1. Robot Requirements and Verification

Req. No.	Requirements Statement	Verification Success Criteria	Verification Method	Verification Status
1	The robot detects and distinguishes a target object.	The robot has a depth camera that can detect objects at a minimum distance of 28cm.	Inspection, Testing	Verified -The robot can distinguish different objects using a depth camera and contours on the object
2	The robot maps out a static unknown environment.	The robot has a LiDAR that can detect obstacles at a distance of 0.2m to 15m away from the LiDAR.	Testing, Simulation	Verified - The LiDAR's range was verified with objects at different distances. It is also capable of effectively mapping out an environment in a simulation and in a test-area.
3	The robot navigates the unknown environment autonomously.	The robot can autonomously navigate with a degree of autonomy between 1 and 6 to reach the target object.	Testing, Simulation	Verified - The robot's feed was used to move around in an unknown environment using teleoperation.
4	The robot reaches, grasps, and securely holds the target object.	The manipulator can pick up the specified target object, has 5 degrees of freedom, a reach of 450mm, a working payload of 50g, and a transparent acrylic shield.	Inspection, Testing	Verified - The manipulator was tested with a maximum weight of 50g which it was successfully able to pick up and move around with no vibration. The Manipulator reach (full range) was tested using a measuring tape.
5	The robot traverses around obstacles.	The robot has 4 in-hub DC motors and wheels with a diameter of 130mm made of rubber with foam inserts.	Testing, Inspection	Not Verified - can only be tested once all the mounts and peripherals are set up on the robot.

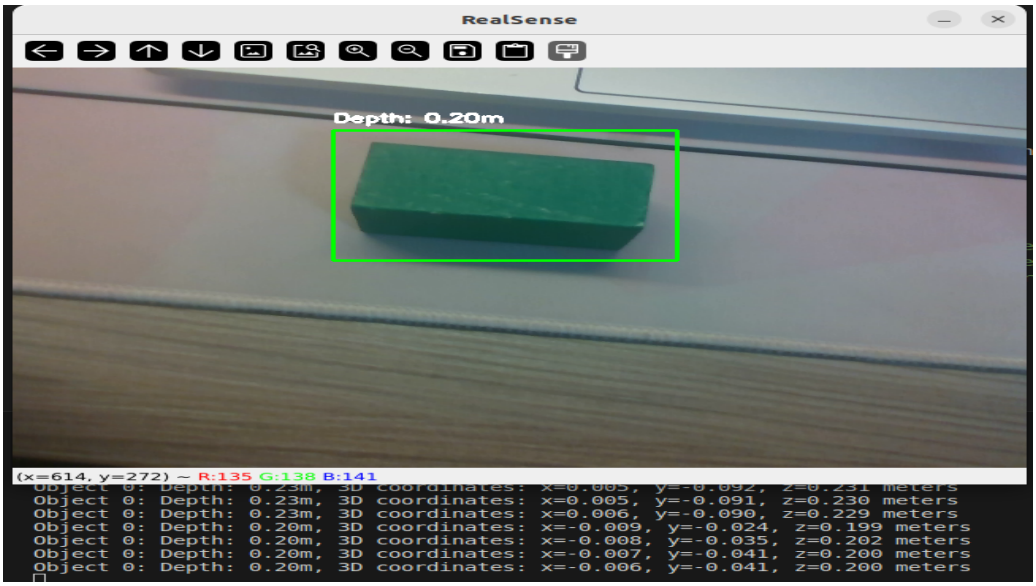
Req. No.	Requirements Statement	Verification Success Criteria	Verification Method	Verification Status
6	The robot is Wi-Fi enabled with a user interface.	The robot has Wi-Fi connectivity with a range of up to 100m for teleoperation and a user interface linked to the camera feed.	Testing, Inspection	Verified - The robot was teleoperated and the user kept moving away to test the strength of the connectivity, however, it was noted the presence of walls affected the distance of connectivity.
7	The robot has it's mass distributed equally.	The base robot weighs less than 7 kilograms, has dimensions of 447x433x249 mm, can carry a maximum payload of 5 kilograms, and has a mounting platform of 299x183 mm.	Inspection, Testing	Partially verified - The robot weighs less than 7kg on a weighing scale, however, the maximum payload is yet to be tested.
8	The robot is safe to use for the customer.	The robot does not have sharp corners.	Inspection, Manufacturing	Verified - The designs were made to match the requirement, avoiding any sharp corners.
9	The robot is dust-protected and withstands vibrations.	The robot's components, structure, and joints can withstand vibrations caused by the robot during traversal.	Testing, Simulation	Undergoing Verification - The individual components are being evaluated for vibrations, however, testing is still pending for the entire robot
10	The robot can return to the starting position.	The robot can accurately return to the starting position after collecting the target object.	Testing, Inspection	Not Verified - Autonomous traversal is still left to be implemented
11	The robot logs data and generates reports.	The robot can log data and generate reports on its activities.	Inspection, Testing	Verified - The robot's log data can be viewed by subscribing to the topics on the robot as observed in Fig.18
12	The robot completes the mission without running out of battery.	The robot has a battery capacity of 5000 mAh.	Testing, Inspection	Not Verified - The robot is still not mission-ready.
13	Emergency-stopping procedures are implemented on the robot.	The robot has implemented emergency-stopping procedures.	Inspection, Testing	Not Verified - This feature has not been implemented yet for testing.

8.2 Design Analysis

1. The robot detects and distinguishes a target object: The depth camera uses yolov5 to successfully distinguish objects and identify the color of the object, as shown in Fig.20. The 3D position of the object relative to the camera is obtained through the depth information inside the camera. The combination of depth information and learning models enhances robot capabilities.



(a) Identify wood cube



(b) Measure 3D coordinates

Fig. 20. Trained Model visualizations for the depth camera

2. The robot maps out a static unknown environment: The LiDAR's verification indicates that the obstacle detection and mapping functionality are promising. The robot can map out an unknown environment and can effectively detect obstacles.

3. The robot navigates the unknown environment autonomously: Teleoperation testing pro-

vides insight into manual control, and there is a successful implementation and testing of autonomous navigation. However this is a critical aspect for real-world applications, and thorough testing is needed for reliable performance.

4. The robot reaches, grasps and securely holds the target object: Successful testing of the manipulator with a specified target object and compliance with design parameters (degrees of freedom, reach, payload capacity) demonstrate that the robot can effectively perform grasping tasks.

5. The robot traverses around obstacles: The potential obstacle traversal capability is acknowledged, but practical verification awaits the setup of additional components. Full testing is required to ensure obstacle avoidance and navigation in complex environments.

6. The robot is Wi-Fi enabled with a user interface: Successful teleoperation and Wi-Fi connectivity up to 100m highlight positive aspects. However, the impact of obstacles on connectivity suggests that environmental factors need consideration for robust performance.

7. The robot can safely store the collected target object: A storage section was designed as a part of the upper platform shown in Fig. 8 & 12. Weight compliance is confirmed, but pending testing of the maximum payload raises questions about the robot's ability to safely handle heavier objects. Further testing is crucial to ensure safety and functionality.

8. The robot is safe to use for the customer: Design considerations, such as avoiding sharp corners and controlling motor rotation speed, confirm the robot's safety features. This proactive approach in design is crucial for user safety.

9. The robot is dust-protected and withstands vibrations: The ongoing verification process indicates a proactive evaluation of components for vibrations. Full-system testing is necessary to ensure that the robot can withstand operational vibrations and environmental conditions.

10. The robot can return to the starting position: The pending implementation and testing of autonomous traversal, including the ability to return to the starting position, is a critical feature for mission success. This should be prioritized for comprehensive validation.

11. The robot logs data and generates reports: Successful verification of data logging and reporting is a positive aspect. The accessibility of log data through subscription demonstrates transparency and accountability in the robot's operation.

12. The robot completes the mission without running out of battery: The current status of not being mission-ready emphasizes the importance of completing battery testing. This is a fundamental requirement for the robot to perform its intended tasks reliably.

13. Emergency-stopping procedures are implemented on the robot: The pending implementation and testing of emergency-stopping procedures are critical for user safety and system integrity. This feature should be prioritized to ensure swift response in case of unexpected events following the guidelines provided in the risk assessment.

8.3 Application software

1. Rviz: Rviz is an integral visualization platform that displays data from multiple robot sensors, such as LIDAR and depth cameras, and planning information. This facilitates a clear understanding of the robot's environmental perception and navigational planning, essential for team collaboration and decision-making.
2. Gazebo: Gazebo provides a robust simulation environment with accurate physics, enabling the creation of custom scenarios for robotic testing. This tool is crucial for developing and refining robotic systems in a controlled, virtual setting before real-world deployment.
3. VScode: Visual Studio Code (VScode) is a code editor that enhances productivity through features like IntelliSense, which suggests code completion, and code refactoring tools that help maintain clarity in complex codebases. These features are vital for efficient software development in collaborative projects.
4. Fusion 360: Fusion 360 allows for the intricate design and fabrication of physical models, such as the Leo Rover's base plate, depth camera attachments and the case for the intel NUC. It also provides comprehensive analysis tools, including static, modal, thermal, and nonlinear analyses, to optimize designs post-creation.

Appendices

A Updated Design Requirements

A.1 Terminology

Utilising the guidelines provided in [8] and [9], we can define the terminologies as:
“shall” – Normative or mandatory requirement
“should” – preferable to have (goal)
”will” – facts or declaration of purpose

A.2 Specifications

This section highlights the main functional and product design requirements to follow while designing the robot.

1. The robot shall detect and distinguish a target object.
    - The robot should have a depth camera that can detect objects at a minimum distance of 28cm.
2. The robot shall be able to map out a static unknown environment.
    - The robot shall have a LiDAR that can detect obstacles at a distance of 0.2m to 15m away from the LiDAR.
3. The robot shall navigate the unknown environment with a degree of autonomy between 1 and 6 to reach the target object.
4. The robot shall reach, grasp, and securely hold the target object using a manipulator mounted on top of the robot.
    - The manipulator should be able to pick up the specified target object on its own. In the case of it not being able to distinguish the objects, it should request human intervention.
    - The manipulator will have 5 degrees of freedom.
    - The manipulator shall have a reach of 450mm.
    - The manipulator shall have a working payload of 50g.
    - The manipulator will weigh within 2.5 kilograms.
    - The manipulator will be constructed from extremely rigid 20mm x 20mm extruded aluminum and all aluminum brackets.
    - The manipulator will have a transparent acrylic shield to keep the robotic arm’s electronics free from debris as well as impact from the arm itself.
5. The robot will be able to traverse around obstacles and not over them.

- The robot will have 4 in-hub DC motors
- The robot's wheels will have a diameter equal to 130mm.
- The robot's wheels will be made up of rubber with foam inserts.

6. The robot will be Wi-Fi enabled and have a user interface linked to the camera feed, to monitor and track the mission's progress.
    - The robot will have a connection range of up to 100m for tele-operation from a human user.
7. The robot should be able to safely store the collected target object when needed without disturbing the dynamics of the robot.
    - The base robot in itself will not weigh more than 7 kilograms
    - The robot without any fittings will be of the dimensions 447x433x249 mm.
    - The robot shall be able to carry a maximum payload of 5 kilograms.
    - The robot's mounting platform for peripherals and the robotic arm will have a dimension of 299 x 183 mm.
8. The robot should be safe to use for the customer.
    - The robot shall not have any sharp corners.
9. The robot should be dust-protected and withstand vibrations, and splashing of water to a certain degree.
    - The robot's components, structure, and joints shall be able to withstand vibrations caused by the robot during traversal.
10. The robot shall be able to return to the starting position after collecting the target object.
11. The robot should be able to log data and generate reports on its activities.
12. The robot shall complete the mission without running out of battery.
    - The robot will have a battery capacity of 5000 mAh.
13. Emergency-stopping procedures shall be implemented on the robot in case anything goes wrong.

B Updated EDIA Workplace Charter

B.1 Our Comments

1. Equality

    We will ensure that each group member has an equal opportunity to participate in the various stages of the projects. Whether in the decision-making process or task allocation, we will treat each member with fairness and equality and avoid discrimination.

2. Diversity

   We will respect and appreciate diversity within the group, including the cultural, academic background, and viewpoint of each team member. When faced with problems or conflicts, we encourage each member to share their unique insights to foster open communication.

3. Inclusion

   No matter how big or small the ideology is, members are always encouraged to share their input or even share their grievances. Our team members strive to ensure that every member can feel included, and the contribution of every team member is vital to achieve our common goals.

The above commitments are to ensure that each member of the team will not be subject to discrimination, bullying, harassment or victimization.

B.2 Team Rules

1. All members should be respectful and polite to each other, irrespective of gender, age, or ethnicity.
2. When performing a certain task, team members should share their task progress to ensure that each member can understand the progress of the entire task.
3. All members should participate in team activities, including meetings, project discussions, etc.
4. Members of the team should ensure the authenticity and validity of the results of their completed tasks so that academic misconduct is prohibited.
5. Encourage members to share opinions and suggestions to ensure diversity and inclusion within the team.
6. Each team member must ensure that the laboratory is tidy.
7. Encourage members to resolve conflicts positively and cooperatively. If necessary, some measures can be taken.
8. Every member of the team should have a collaborative and team spirit, and encourage members to work hard to achieve goals.
9. Members must respect each other's contribution to the team, and must not discriminate against one another based on qualification or experience.
10. In the event of a disagreement or conflict of opinion regarding a specific task, a vote between the team members will be conducted to collectively determine the resultant course of action. In the event of ties, professors can be invited to provide their perspectives, thereby influencing the outcome of the vote.

11. Team members are expected to adhere to prescribed working hours. If a member is unable to work at the scheduled time, the member must explain the reason to other team members in a timely manner and adjust work arrangements according to the situation. Team members are not expected to do any work outside of the agreed working hours already established by the team

B.3 Purpose of our policy

The purpose of the equality, diversity, and inclusion policy is to ensure that all team members are treated equally, to ensure that each member is treated fairly and there is a mutual respect for diversity within the team. In addition, this policy can also promote inclusion within the team. Another purpose of this policy is to eliminate stereotypes based on age, disability, gender reassignment, marriage or civil partnership, pregnancy and maternity, race, religion or belief, sex and sexual orientation discrimination, inequality and prejudice. The establishment of this policy can ensure that every member of the team has equal opportunity and can fully develop their potential, thereby ensuring that the assigned tasks and course objectives can be successfully completed.

The primary goal of the Equity, Diversity, and Inclusion (EDIA) policy is not to punish; Rather, it is designed to make it easier for individuals to understand their mistakes and create a harmonious and inclusive work environment.

B.4 Our disciplinary and grievance procedures

When a member of the team feels discriminated against, bullied, harassed, victimized, or encounters technical disagreements, the team's own internal grievance procedure is set to help resolve the conflict.

When team members feel discriminated against, bullied, harassed or victimized. The team should identify and acknowledge the existence of conflict and encourage open communication so that all members involved in the conflict can safely express their views and feelings. Group members not involved in the conflict can act as mediators thereby forming a panel, and after hearing both the parties in question, can propose a solution that de-escalates the situation.

If the disagreement is about technology, the team leader can hold a meeting and allow team members to discuss it freely. Team members should provide data and research to support their opinions, reviewing relevant data and evidence together to ensure that all decisions are based on evidence and objective information as well.

If both parties are not cooperative during the mediation, the panel will temporarily suspend the mediation and will take a short break. In this temporary suspension, the panel will discuss possible solutions to avoid further conflict during the mediation. When a new round of discussion starts, the panel will set a time period for the discussion to avoid it becoming endless and potentially creating more problems.

It is imperative that when discussing among members, one should avoid using an accusatory tone. Additionally, when each member of the team reaches a consensus, they are required to reflect on the conflicts that occurred within the team so that they can better handle similar situations in the future.

The designated team leader will make sure that each member of the team feels included in any discussion and during regular meetings, it is their duty to check in with the team members to address any issues they are facing.

On this basis, if team members encounter a conflict that cannot be resolved within the team, team members should be encouraged to escalate the issue or conflict outside the team, such as reporting it to senior academic staff or management, to ensure that the issue is handled fairly and professionally. This step is a last resort to ensure the well-being of team members, especially when conflict is severe. The specific flow can be referred to Fig.21.

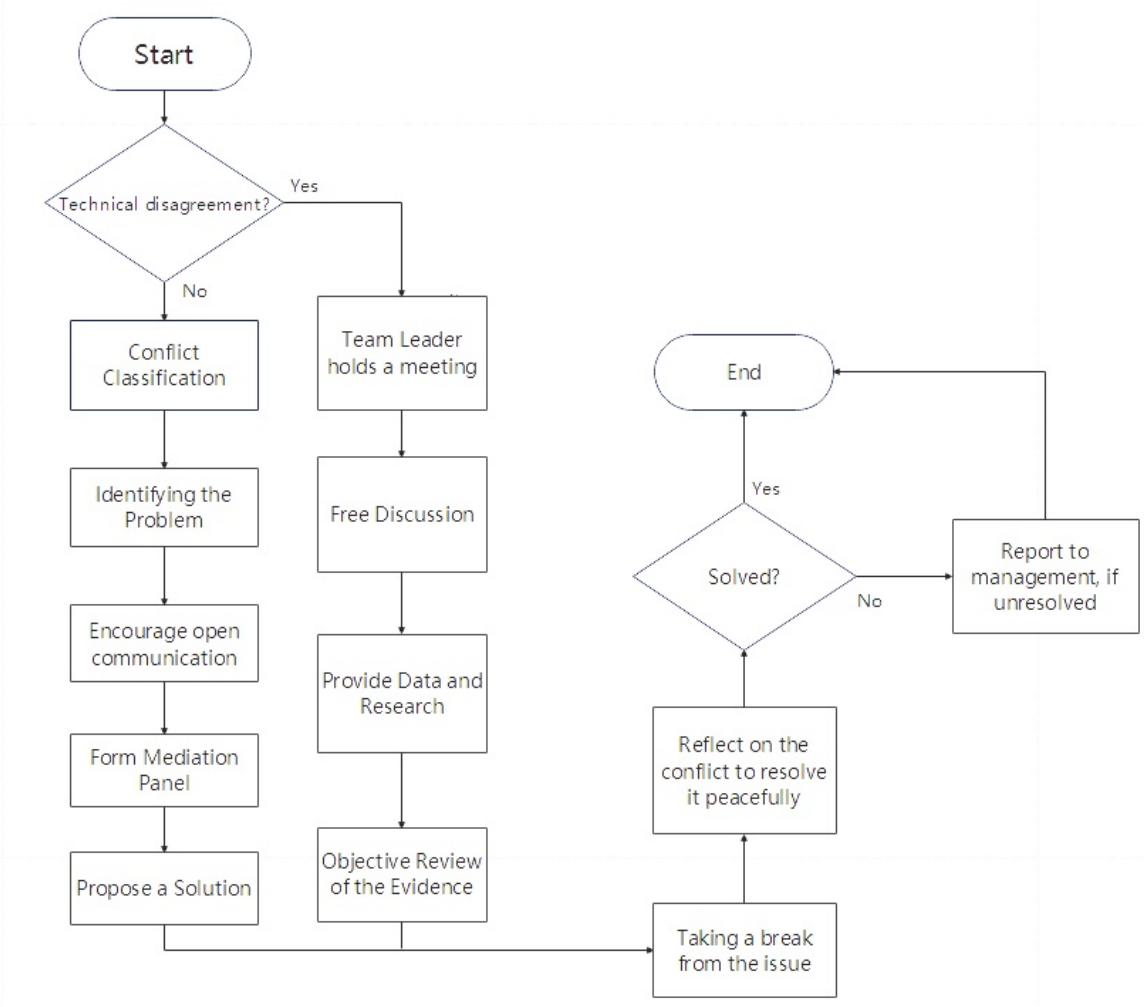


Fig. 21. Flow chart of disciplinary and grievance procedures

C Additional Mechanical Drawings

This section highlights the mechanical drawings for the death camera mount attachment for the Manipulator as well as the mechanical drawings for the NUC case.

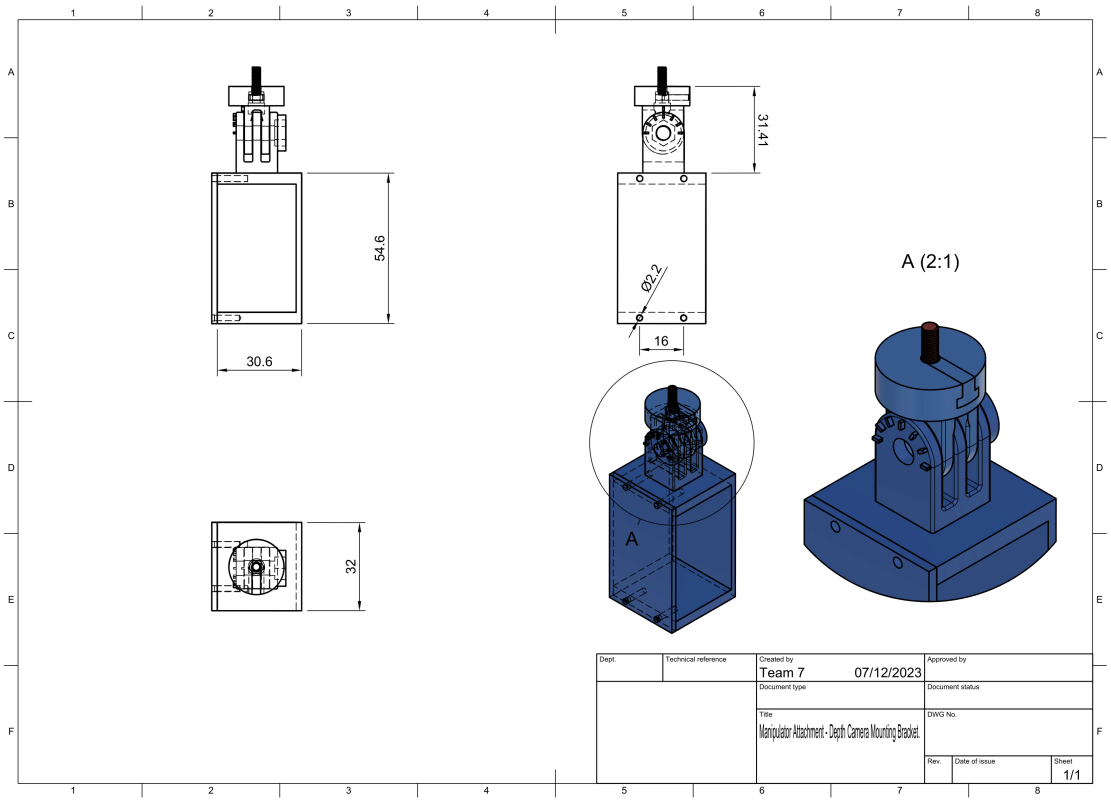


Fig. 22. Mechanical drawings of the depth camera mount for the Manipulator

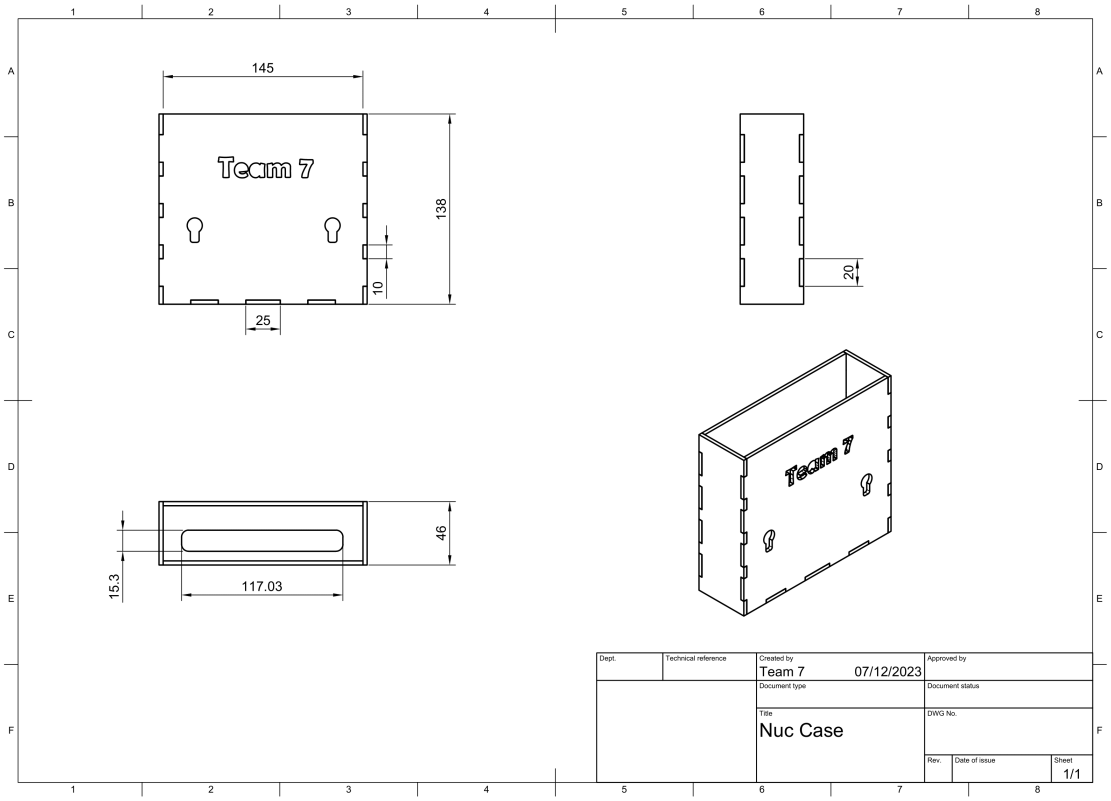


Fig. 23. Mechanical drawings NUC case

References

[1] R. Sánchez-Flores, S. Cruz-Sotelo, S. Ojeda-Benitez, and M. Ramírez-Barreto, “Sustainable supply chain management—a literature review on emerging economies,” *Sustainability*, vol. 12, no. 17, Aug. 2020. DOI: 10.3390/su12176972 (cited on p. 8).

[2] N. Vidakis, M. Petousis, A. Maniadi, E. Koudoumas, A. Vairis, and J. Kechagias, “Sustainable additive manufacturing: Mechanical response of acrylonitrile-butadiene-styrene over multiple recycling processes,” *Sustainability*, vol. 12, no. 9, Apr. 2020. DOI: 10.3390/su12093568 (cited on p. 8).

[3] “Characterization of emissions from carbon dioxide laser cutting acrylic plastics,” vol. 30, no. 4, pp. 182–192, DOI: 10.1021/acs.chas.3c00013. [Online]. Available: https://pubs.acs.org/doi/10.1021/acs.chas.3c00013 (cited on p. 9).

[4] M. Mahasin and I. A. Dewi, “Comparison of cspdarknet53, cspresnext-50, and efficientnet-b0 backbones on yolo v4 as object detector,” vol. 2, no. 3, Sep. 2022. DOI: 10.52088/ijesty.v2i3.291 (cited on p. 9).

[5] *Electromagnetic compatibility regulations 2016: Great britain*, Mar. 2024. [Online]. Available: https://www.gov.uk/government/publications/electromagnetic-compatibility-regulations-2016/electromagnetic-compatibility-regulations-2016-great-britain (cited on p. 9).

[6] J. Hildermeier, C. Kolokathis, J. Rosenow, M. Hogan, C. Wiese, and A. Jahn, “Smart ev charging: A global review of promising practices,” *World Electric Vehicle Journal*, vol. 10, no. 4, 2019, ISSN: 2032-6653. DOI: 10.3390/wevj10040080. [Online]. Available: https://www.mdpi.com/2032-6653/10/4/80 (cited on p. 9).

[7] F. 360, *07010 rplidar a2m8 adapter*, https://kellideas.autodesk360.com/g/shares/SH35dfcQT936092f0e43b1f69cae62dfdb9e (cited on p. 18).

[8] N/A. NASA, 2007, ISBN: 978-0-16-079747-7. [Online]. Available: https://app.knovel.com/hotlink/toc/id:kpNASASEH2/nasa-systems-engineering/nasa-systems-engineering (cited on pp. 33, 38).

[9] M. Pfeifer. Elsevier, 2009, ISBN: 978-0-7506-8287-9. [Online]. Available: https://app.knovel.com/hotlink/toc/id:kpMEDTMEP8/materials-enabled-designs/materials-enabled-designs (cited on p. 38).