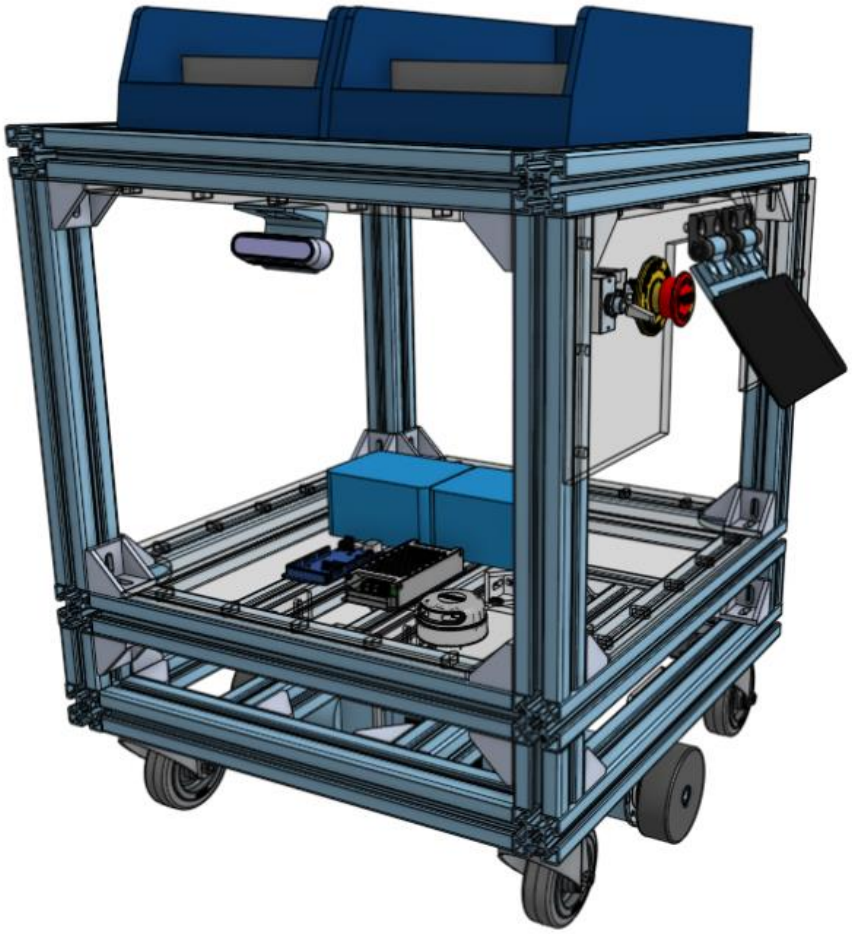
Autonomous Mobile Robot
Responsible Design

Final Report



Joe Bailey, Oliver Brunnock, Tom Coleman,
Jenny Öborn Sandström, Zal Motafram

Summary

This report presents the research and development of an Autonomous Mobile Robot. The objective is to develop an open-source, modular AMR capable of transporting loads in a dynamic environment with precision and safety.

The project is divided into five sub-systems: Electronic Control Unit & Navigation, Powertrain Management Systems, Electrical Power Management Systems, Frame and Carrying System, and User Interface. A thorough, clear, and concise project plan was adhered to, incorporating an agile Gantt chart that allocated roles and responsibilities and outlined the project stages. The project commenced with in-depth state-of-the-art research across all five sub-systems. A detailed design was formulated, including the selection of final components based on the initial research. Both CAD models and electrical circuit diagrams were created. Comprehensive testing of acquired components was conducted, including evaluations of sensors and motors for function and compatibility. Once the components were tested and evaluated, the subsystems were integrated into a full-scale system.

The result was a robot that was powered using a wired connection but could be controlled wirelessly through the control unit. Additionally, handover materials to continue the project and further development guidelines were prepared to facilitate future improvements.

Acknowledgements

We want to thank Roger Ngwompo for this incredible opportunity, our supervisors Christelle Grandvallet and Marie-Laure Perenon, our client Pierre David, workshop technicians Camille Devos and Quoc Bao Duong, and researcher Quentin Levant for all their help and support throughout this semester.

Keywords

Autonomous Mobile Robots (AMRs), Electronic Control Units, Navigation Systems, Simultaneous Localisation and Mapping, Path Planning, Electrical Power Management, Open-Source, User-Interface, Robot Operating System, Brushless DC Motors, Modular Design, Workshop Transport, V-model, Systems Design, Integrated Systems.

A Note on State-of-the-Art Reports

This project was started in February, which meant that there was a significant amount of initial research that was required to gain an understanding of the AMR field, and each specific subsystem. As such, a substantial portion of the project's effort was dedicated to extensive research and generating the state-of-the-art reports by each subsystem, due to the broadness of the topic.

The state-of-the-art reports are essential and paramount for understanding the remainder of this report, to recognise the reasoning behind each subsystem's design choices.

How to Navigate this Report

The report structure follows the framework of the V-model which is explained in *2.3 Design Process* and is exploited throughout the report, in reference to its methodology.

Where not specified, in the same blue italic text, all work was equally shared within that section.

Chapter 1, Introduction: Provides the background, scope and objectives for the AMR project, with an overview of the five subsystems that the system was split into. This section aids in understanding the context and the aim of this project.

Chapter 2, Methodology: Outlines the project management strategies and design processes used to ensure the success of the project. Including specific methods such as GANTT charts and the V-model; it provides insights into the project's execution.

Chapter 3, Research & Development: Refers to the state-of-the-art technologies and methodologies utilised in the development of the AMR. A significant amount of this project was focussed on this research of the five subsystems for the state-of-the-art report to gain insight and understanding of the areas. This chapter is important for understanding the technical aspects, challenges, and approaches used in the project.

Chapter 4, Discussion: This chapter compares the resulting prototype to existing technologies, details theoretical applications of the AMR, and includes discussions around the final system architecture. It is essential for grasping the final impact of the project, limitations, and potential improvements.

Chapter 5, Reflection: Offers a reflection on the methodological approach, challenges faced, solutions developed, and the group dynamics within the team. It also covers personal and professional development experienced by the team throughout the project.

Chapter 6, Conclusion: This serves as the concluding chapter and summarises key findings, discusses further development opportunities, and outlines handover processes for the next project steps.

Appendices & References: Contains supplementary material such as technical specifications, weekly report, technical data, and references that support the design decisions and research.

Contents

Table of Acronyms

Acronym	Description
AMR	Autonomous Mobile Robot
CAD	Computer Aided Design
DHCP4	Dynamic Host Communication Protocol 4
EPMS	Electrical Power Management System
ECU	Electronic Control Unit
ECUN	Electronic Control Unit and Navigation
FCS	Frame and Carrying System
GUI	Graphical User Interface
IMU	Inertial Measurement Unit
IDE	Integrated Development Environment
ISO	International Standards Organisation
IOS	iPhone Operating System
LiDAR	Light Detection and Ranging
Li-LFP	Lithium Ferro Phosphate
Li-Ion	Lithium Ion
Li-NMC	Lithium Nickel Manganese Cobalt Oxide
LAN	Local Area Network
Op-Amp	Operational Amplifier
PWM	Pulse Width Modulation
SLAM	Simultaneous Localisation and Mapping
SBC	Single-Board Computer
RoHS	Restriction of Hazardous Substances
ROS	The Robot Operating System
UML	Unified Modelling Language
UCD	User Centred Design
UI	User Interface
UIE	User Interface & Ergonomics
VM	Virtual Machine
WEEE	Waste from Electrical and Electronic Equipment

Table of Figures

List of Tables

[BLANK PAGE]

1 Introduction

This chapter discusses the project's background, its purpose, problem definition, and provides an overview of the various subsystems employed. The background outlines the context, and the purpose explains the goals. The problem definition describes the specific challenges that the project aims to address. Finally, the description of the subsystems provides insight into their roles and contributions to the project.

1.1 Background

The project is part of a course called Responsible Design, which is aimed at international exchange students Grenoble INP - Génie Industriel. The main objective of the course is to provide students with the ability to carry out an effective design process in groups using various tools and methods in design and project management to manufacture a product that meets a specific customer’s needs.

The report focuses on the development of an Autonomous Mobile Robot (AMR) for the university's operations management platform S.mart (Systems Manufacturing Academics Resources Technologies). This S.mart platform is used by the university as a means to develop digital models, multi-physics simulations, prototypes, and to set up the industrialisation and control process. An image of the platform is shown in *Figure 1*.

Figure 1: S.mart Platform

AMRs exemplify cutting-edge technology within the field of robotics, aimed at enhancing automation and efficiency across various industries. Unlike traditional Automated Guided Vehicles (AGVs), which operate along fixed paths, AMRs are equipped with advanced sensors, software, and onboard navigation systems [1]. These capabilities enable AMRs to dynamically perceive and interact with their environment, facilitating efficient navigation while avoiding obstacles and responding to real-time changes in surroundings.

The S.mart platform owns two Sherpa B robots *Figure 2* which are AMRs that can navigate to specific points within the workshop. However, there is a need for a new AMR solution which is open-source and tailored towards the school environment. The robot must effectively deliver parts in the workshop and perform tests on navigation algorithms, as existing solutions in the workshop are too large and lack open-source functionality.

Figure 2: Sherpa B Robot used in Smart Platform [2]

The project was conducted by five students, four of which are from the University of Bath, UK [3] and one from KTH Royal Institute of Technology, Sweden [4]. There are also three key individuals who play different roles in ensuring the success of the project: a customer, an advisor, and the teaching team at Grenoble-INP - Génie Industriel [5]. Pierre David is the customer contact and contributes to the project's needs. Together with him, product specifications are formulated to ensure that the final product meets his expectations. Christelle Grandvallet and Marie-Laure Perenon are the supervisors and support the project group in collaboration with the customer. They also assist in implementing and structuring the design to ensure that it meets the customer's requirements. The teaching team at Génie Industriel also supports the project group by providing technical expertise and guidance during the design process. The roles are summarised in *Table 1*.

The users of the AMR will consist of students or individuals without specialised training who will engage with the robot in educational settings, workshop technicians responsible for its maintenance and operation, and specialised programmers involved in its software development and customisation.

In a dynamic workshop environment, the efficient and safe transportation of materials is crucial for maintaining productivity. The aim of this project is to develop an open-source AMR capable of delivering two standard boxes within the S.mart workshop [6], with the capability to be modified or upgraded to maintain and improve performance. The robot must be able to carry these boxes and securely transport these to a location without human intervention.

Table 1: Project Roles

Role	Name	University
Project Team	Oliver Brunnock	Bath, UK
	Tom Coleman	Bath, UK
	Joe Bailey	Bath, UK
	Jenny Öborn Sandström	KTH, Sweden
	Zal Motafram	Bath, UK
Client	Pierre David	Grenoble INP – Génie Industriel
Supervisors	Christelle Grandvallet	Grenoble INP – Génie Industriel
	Marie-Laure Perenon	Grenoble INP – Génie Industriel
Users	Quentin Levent	Grenoble INP – Génie Industriel
	Camille Devos	Grenoble INP – Génie Industriel

1.2 Requirements & Architecture

In this section, the functional decomposition, the system architecture, and the technical requirements are presented, providing a comprehensive overview of the project's structure and specifications.

1.2.1 Functional Decomposition

The functional decomposition was created using Stuart Pugh's method of System Level Design [7], by breaking down primary functions into principal subfunctions and using these subfunctions to abstract a desired/technical function or further abstract an elementary function. At this abstracted level, technical solutions to the functions are found by using mechanisms, systems, or nature. Hence, as you go down the functional levels, it explains how the function is carried out and in reverse, it explains why the solution is required. This can be demonstrated by Stuart Pugh's functional decomposition tree of a car door, *Figure 3*. Decomposing the system-level design of an AMR enhanced the functional understanding of each subsystem, by creating a hierarchy of essential functions*, Figure 4, Figure 5.*

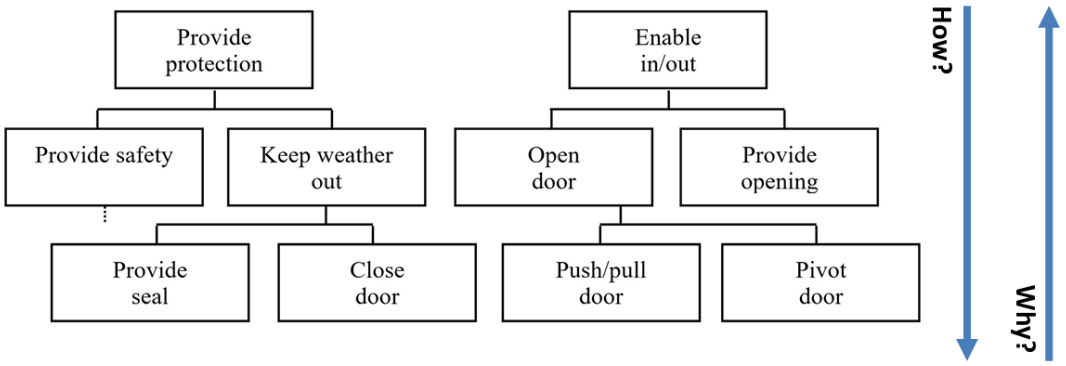


Figure 3: Stuart Pugh's Functional Decomposition tree of a car door [7].

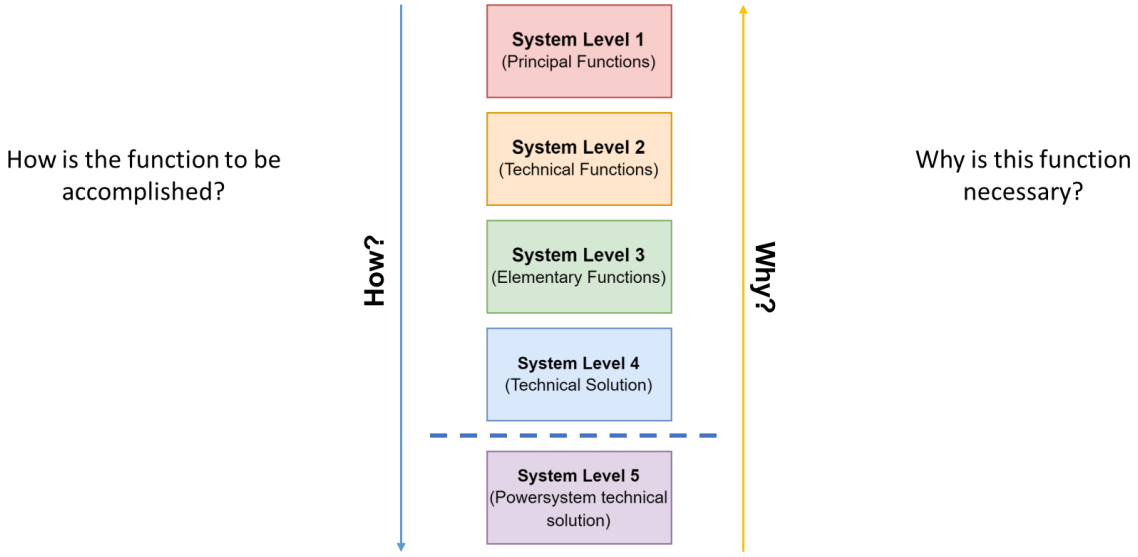


Figure 4: System Level Design key to Functional Decomposition in Figure 5.

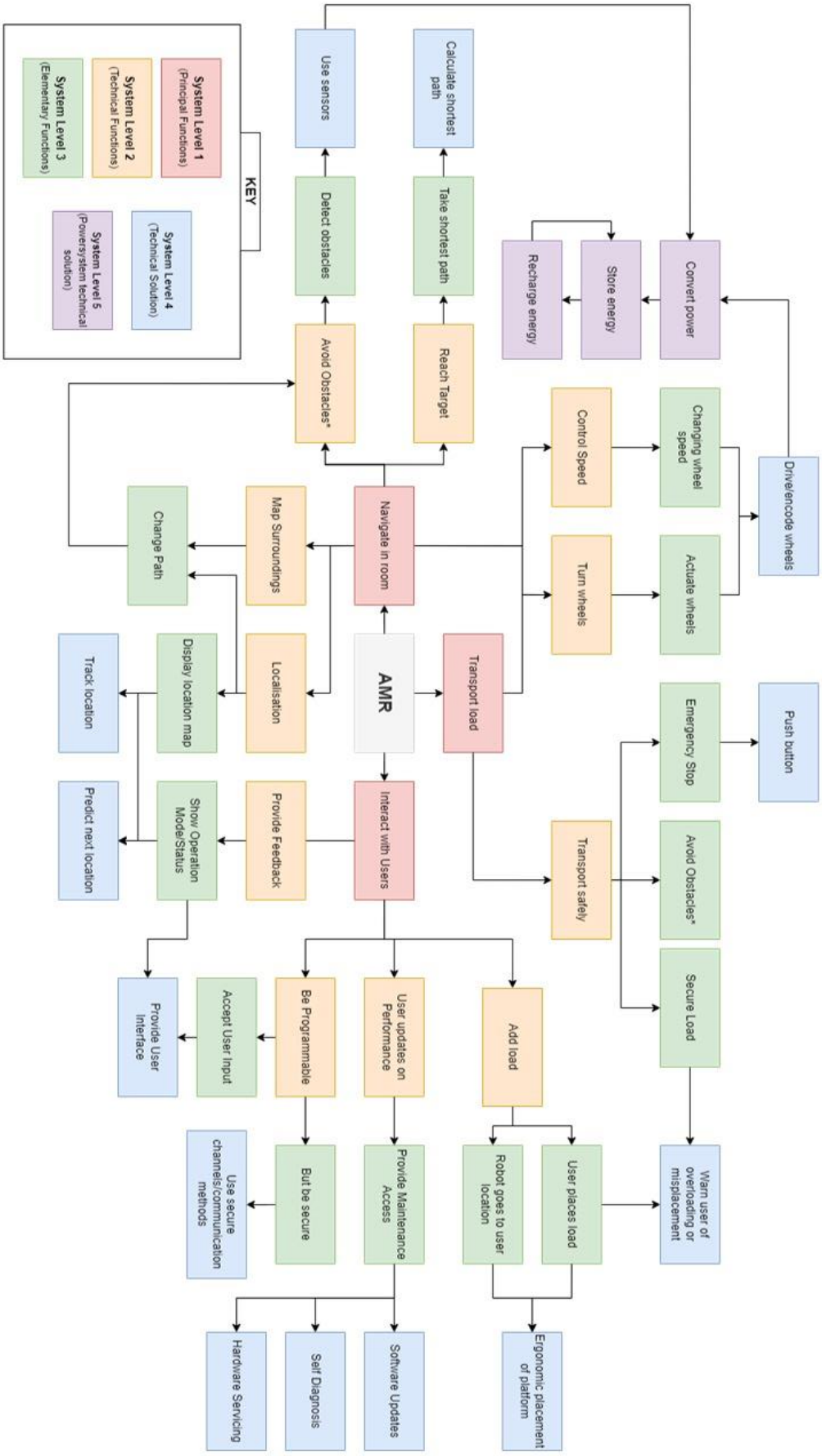


Figure 5: Functional Decomposition Hierarchy of AMR.

The functional decomposition for the AMR is shown in *Figure 5*. It was created using the 3 principal functions of “Navigate in room”, “Transport load”, and “Interact with Users” which were identified as such during the interviews and discussions with our client, and from the project brief. From these, Technical Functions were branched out, acting as the “how” these functions will be implemented. Similarly, the Elementary Functions and the Technical Solution were formulated in the same manner. The Power system technical solution was created afterwards to ensure that the appropriate considerations were made in terms of supplying power to facilitate other functions.

1.2.2 System Architecture

A system architecture *Figure 6*, based on the principal and elementary functions from the functional decomposition, formulated an approach for designing an AMR based on individual subsystems. As shown, the system revolves around the ECU which acts to control the other subsystems such as the movement, user interface, power systems, etc.

This diagram aided in splitting the design into the multiple subsystems shown inside the dashed lines, namely: ECU & Navigation, Powertrain Management System, Electrical Power Management System, UI & Ergonomics. It also helped to identify some key components necessary for operation, such as the battery charger, battery, and power converters in the Electrical Power Management System for example.

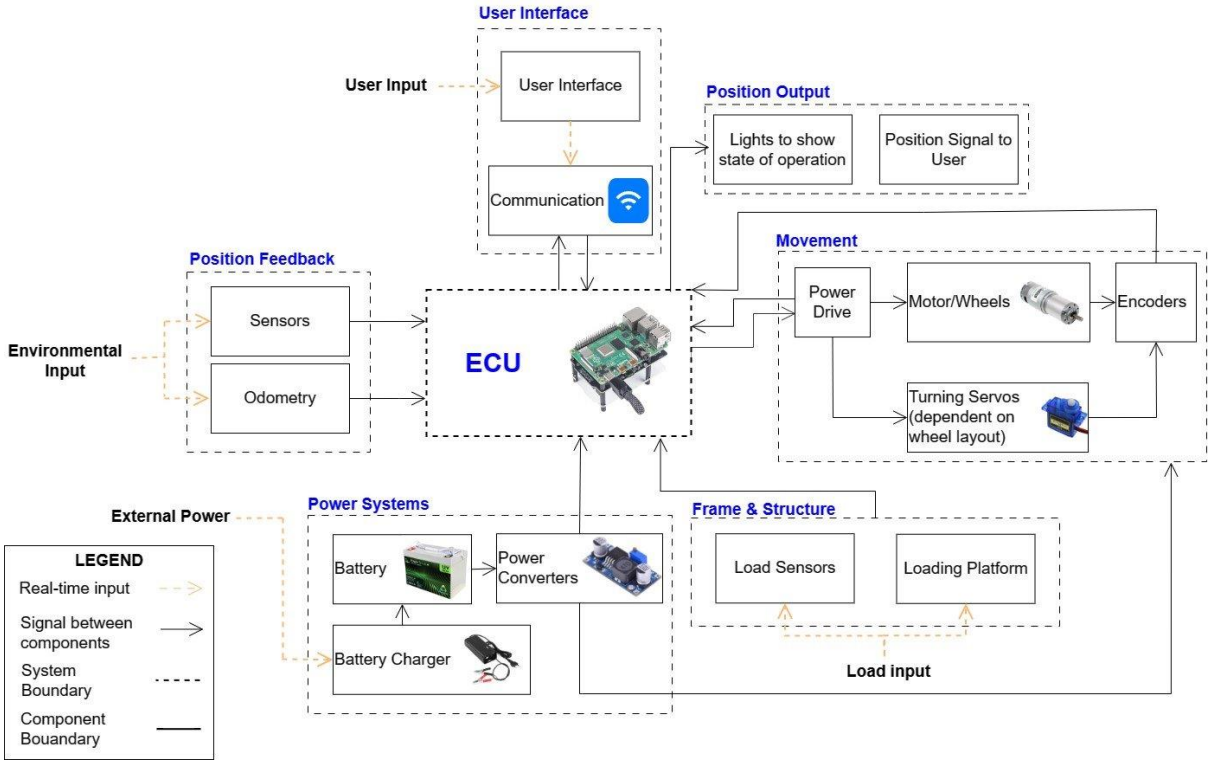


Figure 6: System Architecture of AMR.

1.2.3 Technical Requirements

The technical requirements are a clear and concise list of measurable objectives that the AMR must be able to achieve for it to be successful. This list was created by first gaining a detailed understanding of the project brief, and then using this to develop some performance metrics for the robot, such as acceleration targets, top speeds, operating time, weight, etc. Discussions were then had with our client Pierre David to refine these requirements further and to add others where necessary. After multiple iterations, the final technical requirements list was then completed. The entire list of requirements are in *Appendix B: Technical* Requirements, where the requirements are organised into categories and

Table 2 highlights the functional category of requirements, which have the highest priority of fulfilment in the requirements list.

For example, the functional technical requirement, 'Must be capable of accelerating at a given rate under full load,' was formulated during discussions with the client. The client specified that the robot must move as fast as the existing Sherpa B robots owned by the university. After reviewing the manufacturer's claimed accelerations, an acceleration rate of 0.5 m/s² was agreed upon with the client.

Table 2: Technical Requirements Example

Requirement	Performance Criteria	Value
Must be capable of carrying a maximum payload	Maximum Load	40kg
Must have a specified height	Height	70cm
Must be capable of accelerating at a given rate under full load	Acceleration	$0.5ms^{-2}$
Must operate under full load for a minimum time	Battery life	2 hours
Must be able to be loaded by hand in a minimum time	Time to load	15 seconds
Must be able to detect objects and avoid obstacles	Degree of visibility	180°

1.3 Subsystems

The work of developing the AMR has been structured around five different subsystems, each overseeing various aspects of functions and performances. Furthermore, each subsystem has been assigned to a member of the group which is shown in *Table 3.*

Table 3: Responsible student for each subsystem

Subsystem	**Name**
Electronic Control Unit & Navigation System	Oliver Brunnock
Powertrain Management	Tom Coleman
Electrical Power Management System	Joe Bailey
Frame & Carrying System	Jenny Öborn Sandström
User Interference & Ergonomics	Zal Motafram

1.3.1 Electronic Control Units & Navigation System (OB)

Completed by Oliver Brunnock

At the heart of an AMR is an Electronic Control Unit (ECU) which functions as the brain of an AMR and is responsible for processing data, decision making and controlling the robot's movements. ECUs process inputs from sensors which allow the robot to perceive its surroundings accurately. The ECU analyses this sensor data in real time and determines the most efficient route for the AMR to take. This path-planning involves avoiding obstacles and dynamically adjusting to changes within the environment. Therefore, the ECU is involved in ensuring the safety and efficiency of the robot's operations whilst allowing the AMR to interact intelligently within its environment.

The navigation system of an AMR enables the robot to move within a complex environment. This integrates sensors such as Light Detection and Ranging (LiDAR), 3D cameras, infrared, and Inertial Measurement Units (IMUs) to continuously gather information from surroundings. The AMR creates a real-time map of the environment which enables the identification of obstacles and effective route planning.

1.3.2 Powertrain Management System (TC)

Completed by Tom Coleman

The powertrain subsystem is concerned with the movement of the AMR. All actuator functions, including stopping and starting, turning, accelerating, and decelerating are covered by this subsystem.

The core of this system is the motors. These are the main actuators that allow the robot to move around and navigate rooms. Additional mechanical components are selected to allow the motors to carry the AMR's full frame at the desired speed and torque, as well as provide safety features for an emergency stop. These include gearboxes, wheels, and brakes. Electronic components are included to link to other systems and improve the AMR's navigation ability. These include motor drivers, encoders, and amplifier circuits to connect the motors to the ECU. These will allow their speed to be controlled by the system's microcontroller and provide feedback on speed to help the AMR determine its position and control its velocity.

All of these elements come together to produce a system that works closely with the ECU to allow the AMR to move anywhere it needs to, while avoiding obstacles.

1.3.3 Electrical Power Management System (JB)

Completed by Joe Bailey

This subsystem is responsible for ensuring that all the components onboard the AMR, such as the motors, sensors, and computational equipment, are provided with sufficient voltage and power. A compact and lightweight system is required to maintain the agility and operational range of the AMR.

The system is based around a battery, which stores the energy to power the AMR whilst untethered. Additional components, such as power converters, switches, relays, and fuses are used to ensure this energy can be delivered to each component safely and reliably.

1.3.4 Frame & Carrying System (JOS)

Completed by Jenny Öborn Sandström

The frame and carrying system form the foundation of the AMR unit, enabling its ability to move, handle loads, and perform tasks.

The frame forms the structural base providing stability and physical structure to the unit. Its design is crucial to securely hold all components in place. The frame is also equipped with mounts and installation sites facilitating easy integration of vital components such as sensors and navigation units, thereby enabling autonomous navigation and task execution.

The carrying system is responsible for efficiently managing and transporting the load that the AMR unit is intended to carry. It is designed to be versatile and adaptable to accommodate various types of loads and tasks.

1.3.5 User Interference & Ergonomics (ZM)

Completed by Zal Motafram

Despite AMRs being “Autonomous”, they still require a means of command input and programming, to ensure they complete the desired mission correctly. Therefore, the user interface is a critical component of the AMR, facilitating a seamless pathway of communication between the hardware, software, and the user.

This subsystem will identify and implement the most efficient strategies for outputting information to the user while ensuring a straightforward and effective means for user-generated input back into the system. It will differentiate between two distinct types of user inputs: those for commands and those for programming; providing each with a specialised interface.

Ergonomics serve a vital purpose when ensuring user comfort and safety when interacting with the AMR. Hence, ergonomics of safety features and user interfaces are also a focus of this subsystem.

Hence, using these key aspects of a user interface, the interactions between the user and the AMR can be separated into these three separate categories: **Command User Interface**; **Ergonomics and Safety**; and **Programming User Interface**.

2 Methodology

In this chapter, a comprehensive overview of the methodology employed in this design project is provided. The methodology encompasses both project planning and the utilisation of the V-model framework [8] for the development process.

2.1 Project Management

In this section, an overview of the project management is provided. Project management constitutes a critical phase in the process, ensuring a structured and efficient pathway toward achieving the project's goals.

2.1.1 GANTT Chart

Gantt charts are a method of graphical organisation used to clearly outline the schedule of project activities over time [9]. Structured as a timeline, each activity is represented as a bar, with the horizontal axis denoting time and the vertical axis indicating different activities or tasks. By displaying all project activities and their planned timeframes on a timeline with specified start and end dates for each activity, creating a realistic project plan and schedule is made easier.

To create the Gantt chart, all necessary activities for project completion are identified, ranging from planning and research to implementation and evaluation. Each activity receives start and end dates based on its estimated duration and relationship to other activities. Once all activities are identified and the timeline established, the Gantt chart is constructed, offering a visual depiction of the project's timeline, and revealing overlaps and interactions among activities.

The Gantt chart for this project is presented in *Figure 7.* The tasks and deliverables are listed on the left and consist of seven main phases: research (SoTA), needs analysis, ideation and conceptualisation, development and realisation, prototyping, testing, and documentation. The chart displays the percentage of completeness and the corresponding dates for each task.

The initial phase involves state-of-the-art research and needs analysis, conducted in parallel to outline the project's starting point. The design phase overlapped with this research, allowing for the development of initial ideas. Following the research phase, an intermediate defence presentation took place. Subsequently, the development, implementation, and integration phase began, marking the start of the build phase. During this period, the verification and validation phase commenced concurrently with testing.

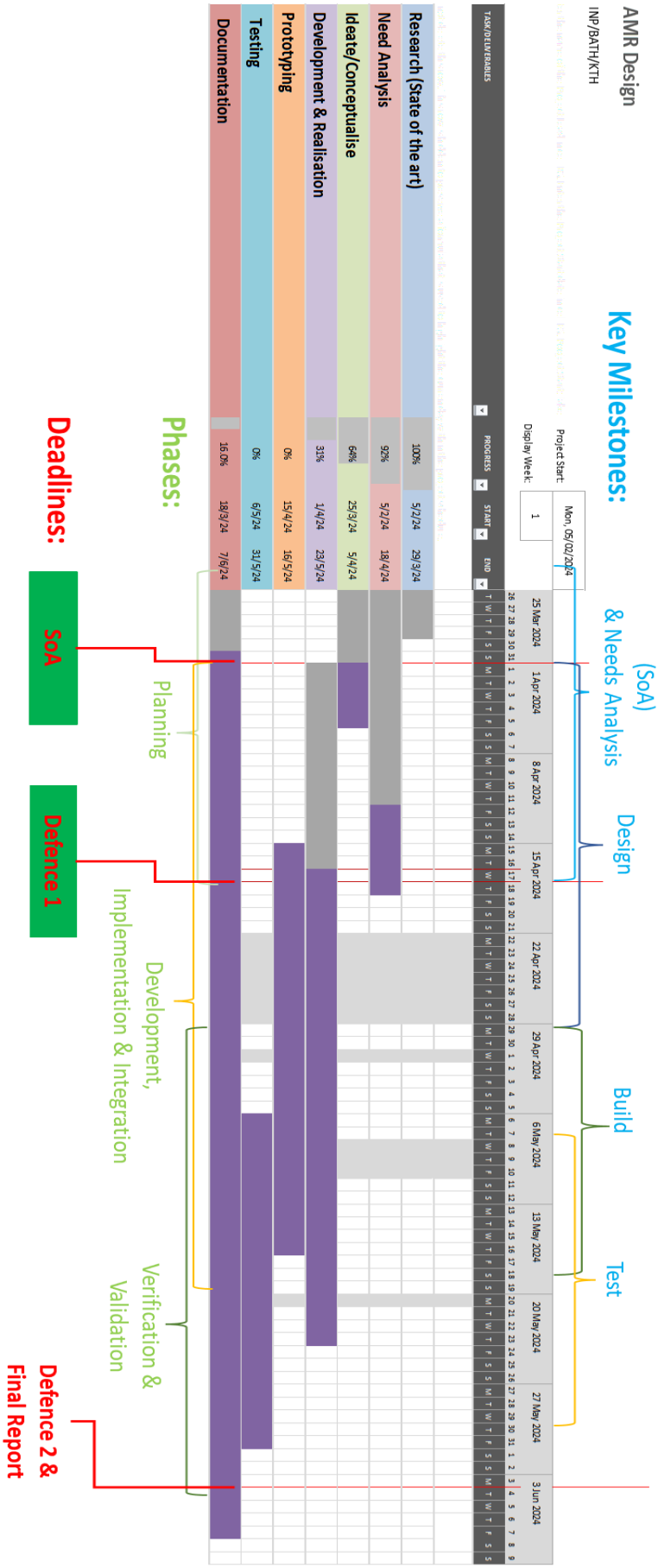


Figure 7: Gantt Chart used for project management.

2.1.2 Weekly Reports

During the project, weekly status reports were regularly generated and shared to provide a summary of progress and activities. These reports were vital for keeping the project team and advisor informed about the project's status. The reports typically included the following aspects:

- **Tasks completed this week:** A detailed summary of the activities and project milestones completed achieved since the previous report which included a description of specific tasks and project steps that have been finished. This included a specific date of completion and a section for who is assigned to each task.

- **Tasks in progress:** This provides an overview of what is currently underway with a percentage of completion and stated which tasks still need to be finished.

- **Tasks for next week:** This gave the team and advisor clear objectives for the following week with the priorities for each task rated from low to high.

- **Upcoming project deliverables:** An update on the project's progress toward the overall goals and milestones. This provides a clear indication of whether the project is on track with expectations or if adjustments are needed.

An example of a weekly report is enclosed within *Appendix A: Weekly Report Example*.

2.1.3 Risk Management

Project risk management analysis, *Table 4,* was important for the successful implementation of the AMR, especially given the size of the project. The GANTT Charts and weekly reports mentioned were key to tracking the progress of the project, flagging issues to our supervisors, and ensuring they were up to date on the project progress. Regular meetings with our client and supervisor also confirmed that the design was meeting their expectations, so that any adjustments could be made before too much time was wasted. This was done to spot mistakes earlier on and mitigate the risk of delaying the development of the AMR [10].

Table 4: Project Risk Analysis

Risk	Consequences	Probabiliity (1-5)	Impact (1-5)	Score (P x I)	Action
Communication issues	Misunderstandings, Project delays	4	4	16	Weekly reports, Clear communication channels (Teams and WhatsApp)
Team conflicts	Reduced efficiency, Poor work environment	2	5	10	Team building
Dependence on key personnel	Project delays if key personnel are unavailable	2	4	8	Regular communication with the client, the users, and supervisor
Delays in component deliveries	Project delays	4	3	12	Time buffers in schedule
Incorrect component orders	Additional costs, Project delays	1	5	5	Thorough Bill of materials (BOM) and order list
Technical issues with prototype	Project delays, Need for Redesign	3	4	12	Iterative design process
Insufficient testing	Defective final product	2	5	10	Comprehensive testing
Budget Overruns	Resource reallocation	2	4	8	Regular budget reviews, Thorough Bill of materials (BOM) and order list

2.2 Design Process (ZM)

Completed by Zal Motaframan

The V-model is an approach used to design complex systems and address multidisciplinary needs, ensuring a clear distinction between system and component design [8]. This approach emphasises the importance of verification and validation in each development phase and relies on iterative testing of smaller components and subsystems before, and during, their integration. The "V" model's shape visually represents the steps involved in designing and building a system as shown in *Figure 8*.

On the left side of the V, the process begins with understanding the project's concept and defining system requirements, progressing through stages of system and subsystem design, where more specific requirements are detailed at each level. This phase includes discussions with the client to understand the robot's requirements, such as acceleration, size, and system weight constraints. Additionally, this phase involves initial research and design, as documented in the subsystem's state-of-the-art reports.

The lowest point of the V represents the implementation of the system, where the different subsystems are tested. During the project, this involved individual testing of the motors and developing amplifier circuits to address compatibility issues.

As the model progresses to the right side of the V, it focuses on the integration of subsystems and corresponding testing phases at each functional level, ensuring that each element meets the specified requirements. For example, this stage involved connecting the motors to the frame and to the portable electronics onboard the robot, rather than directly to the lab power supply.

This methodical approach helps identify and resolve issues early in the design process, leading to a more reliable and robust system. By incorporating verification and validation at each step, the risk of errors is reduced, increasing the chances of successful project implementation.

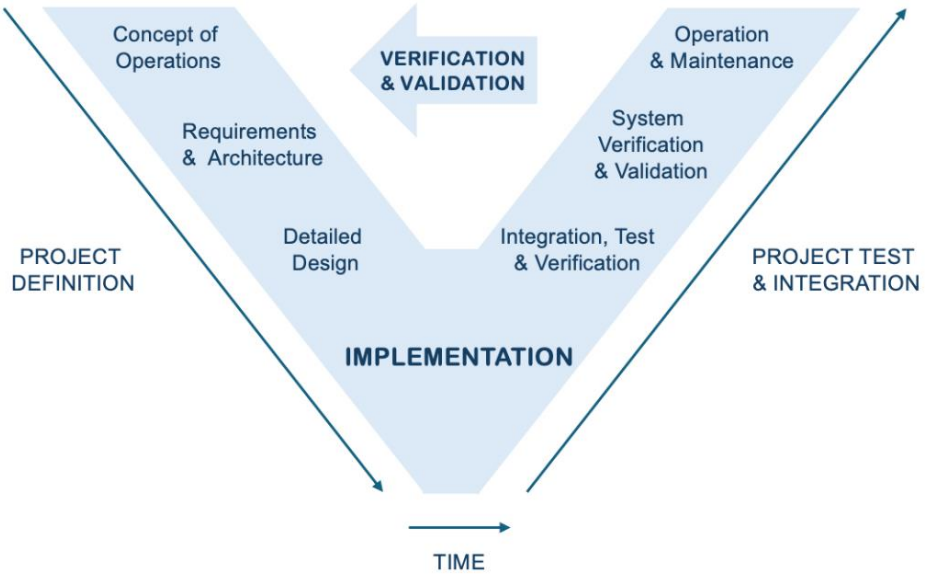


Figure 8: Illustration of the V-model, inspired by [8].

2.3 User Centred Design (ZM, JOS)

Completed Zal Motafram, Jenny Öborn Sandström.

When considering the use cases of the AMR, the users are at the centre of the design of the AMR, particularly for the FCS and UIE subsystem. Therefore, this will be a User Centred Design (UCD) process [11] and for this, the needs of the user must be explored, by creating a profile of each expected user and will require consideration when designing a new AMR.

2.3.1 Primary Needs of the User

The main needs shared between all users are to have an open-source AMR to carry components, that they can modify and program to suit specific missions.

There are three types of expected users for the design of this AMR: Students or Unspecialised (Standard) users; Workshop technicians; and Specialised Programmers. This involves a thorough evaluation of existing interfaces to identify and incorporate the most effective elements into the new AMR design, ensuring it is accessible and user-friendly for all intended users.

This was demonstrated by using the Functional Decomposition in *Figure 5,* to create a Use Case Diagram. A Use Case Diagram is a graphical representation used to describe the functional requirements of a system, a technique used in software development in Unified Modelling Language (UML) [12], showcasing the interactions between users and the system itself through actors, use cases, and their associations. It serves as a tool for capturing the system's intended functionalities from the user's perspective to ensure the system meets their needs and to define the user profiles. The three users have direct interactions with the system, which trigger a use case and thus extended interactions can be performed by the user to trigger further use cases, as demonstrated in *Figure 9*, describing the expected user profile interactions with the AMR.

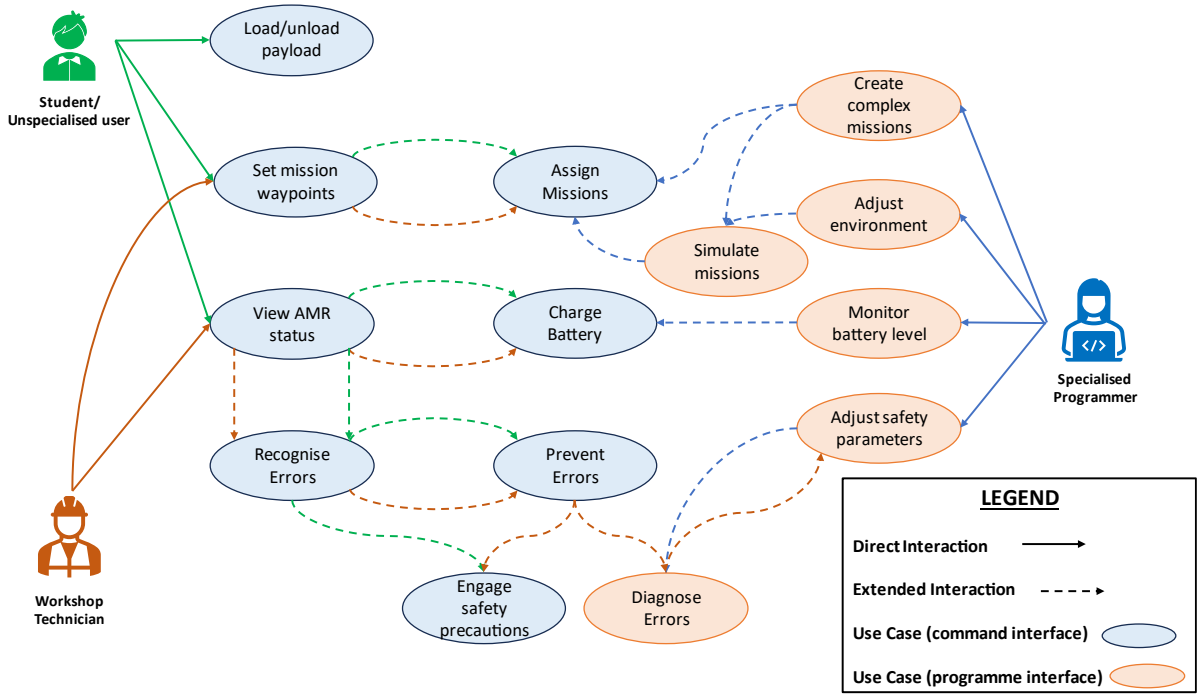


Figure 9: Use Case Diagram, describing the expected user profile interactions with the AMR.

2.3.2 Gestalt Rules of Visual Perception

First introduced by the psychologist Christien von Ehrenfels, the Gestalt rules of Visual Perception are fundamental principles which explain how visual impact is determined by the way humans process images [13].

Among the most prominent of these rules are proximity, similarity, closure, symmetry, and figure-ground. The principle of proximity suggests that objects close to each other are perceived as a group, while similarity has elements that are similar in shape, colour, or size are seen as belonging together. Closure involves the mind's tendency to see complete figures even when part of the information is missing, leading to the perception of a whole object. Good continuation implies that elements arranged in a line or curve are seen as more related than elements not on the line or curve, and figure-ground distinguishes objects from their surrounding background. Symmetry suggests that elements perceived as symmetrical to each other tend to be seen as part of the same group, facilitating a more organised and harmonious visual experience [14]. These rules are effectively demonstrated in an everyday UI, like the Apple IOS, as displayed in *Figure 10.*

The Gestalt principles have practical applications across various fields, including design, art, architecture, and user interface development. They help designers create harmonious compositions that are visually appealing and easy for the human eye to understand. Understanding and applying the Gestalt principles can significantly enhance the effectiveness of visual communication by aligning designs with the natural tendencies of human visual perception, making information more accessible and the visual experience more satisfying. The utility of these rules within design is to positively manipulate the user’s perception, ensuring ease of recognisability of functions within the interface [15].

These principles will be an invaluable tool when comparing different approaches designers have taken to creating the user interface for different AMR designs. It can be assessed whether the AMRs have followed these rules and how effectively they have done so to distinguish between different functions.

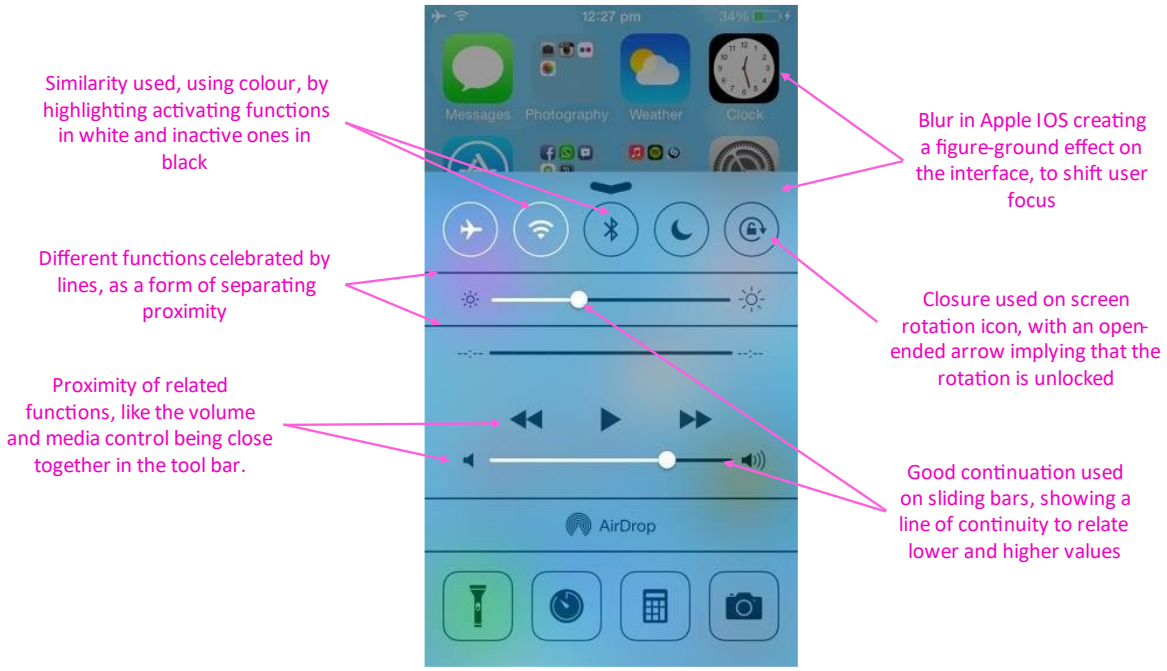


Figure 10: Apple IOS Quick Access toolbar [16]

2.3.3 Ergonomics & Safety Standards

For Safety, standards must be observed by most industrial machinery from the International Standards Organisation (ISO), including AMRs, in accordance with ISO 13849-1(:2023)/ 13850(:2015) [17] [18].

Additionally, the ergonomics of reaching the stop button must be a quick process for most users, to engage safety precautions as stated in the Use Case Diagram *Figure* 9 therefore, the smaller the movement for the user, the faster the action. This can be judged using the DINED database for anthropometrics, when looking at the average fist height distance of mixed adults from ages 31-60 (767mm), *Figure 11,* and safe lifting/bending distance, without bending for a load of 30kg, is 5” (127mm) from the body’s core [19] [20]. Any load above 30kg will require extra support to load/unload.

For example, the Sherpa B in *Figure 12,* has a platform height of 593.9mm and thus the reach from the fist height is 173mm. Hence, as this is greater than the safe lifting distance of 127mm, the user must bend their knees to move the load on the platform. However, for activating the stop button at a height of 943.9mm, the reach is 123.2mm. Hence, this is less than the safe bending/lifting distance of 127mm and permits activation without bending.

Furthermore, safety/error warnings and error correction/fault messages are essential for user communication when informing the user of its presence or faults, respectively, in accordance with ISO 10218-1 [21]. Hence, it is essential that most AMRs should adhere to these standards and issue the user with these basic levels of safety.

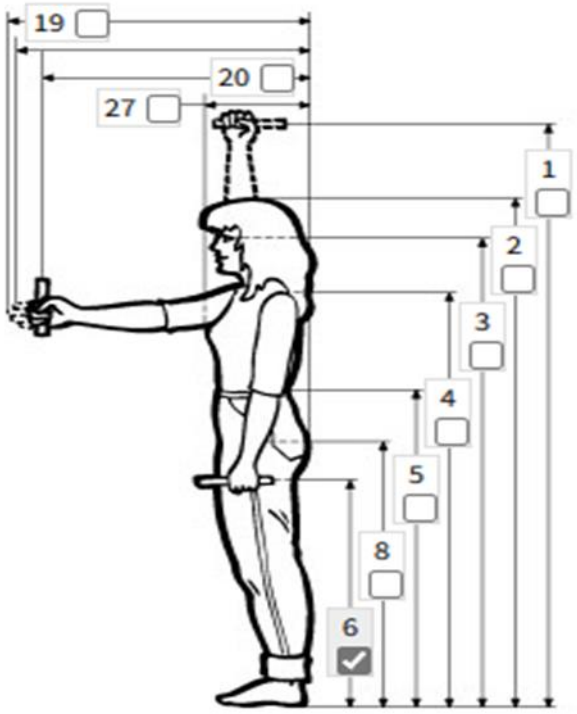


Figure 11: DINED database for Anthropometrics.

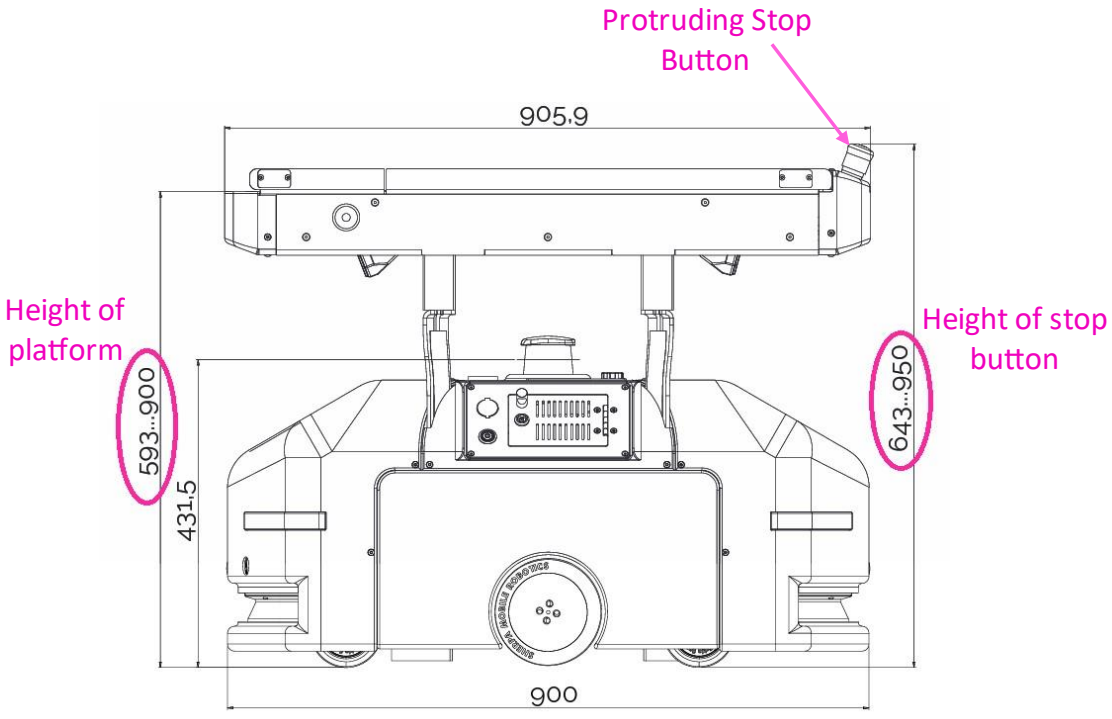


Figure 12: Dimensional Diagram of Sherpa B [22]

2.3.4 Nielsen's Heuristics of Usability

As stated in *2.3.2 Gestalt Rules of Visual Perception*, recognisability is of paramount importance as it simplifies the usability of a user interface. Thus, Nielsen's Heuristics are a set of ten usability principles designed by Jakob Nielsen to guide the design or evaluation of user

interfaces. Referring to *Figure 13*: Diagram of Nielsen's Heuristics of Usability, these principles include [23]:

1) **System visibility**, ensuring users are informed about what is happening.
2) **System match with the real world**, using familiar concepts and language to appear in a natural order.
3) **User control and freedom**, allowing easy reversal of actions.
4) **Consistency and standards**, adhering to platform conventions.
5) **Error prevention**, reducing the risk of errors.
6) **Recognition rather than recall**, minimising user memory load.
7) **Flexibility and efficiency of use**, accommodating both novice and expert users.
8) **Aesthetic and minimalist design**, focusing on relevant information.
9) **Help users recognise**, **diagnose**, **and recover** from errors, providing clear error messages.
10) **Help and documentation**, offering easily accessible and helpful guidance.

These heuristics form an essential framework that significantly enhances user interface design and elevates the overall user experience. Consequently, they provide a robust set of criteria for evaluating how AMRs have incorporated these principles into their design and the extent to which they facilitate communication with users. This evaluation will identify the effectiveness of AMRs in creating intuitive and user-friendly interactions, as required from the Use Case Diagram; *Figure 9*, underscoring the importance of these principles in the development of user-centred robotic systems design.

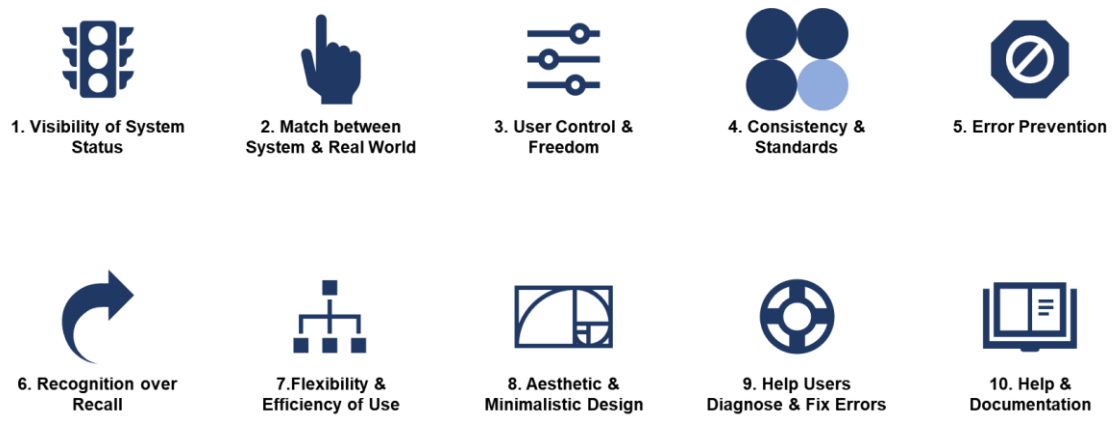


Figure 13: Diagram of Nielsen's Heuristics of Usability

2.3.5 Safety

Effective risk management was crucial for the successful implementation of the AMR. Given the need for an open-source AMR optimised for workshop settings, a risk management plan was employed to mitigate risks throughout the design process. Key risks involved ensuring the robot could be safely stopped in an emergency and maintaining the safety of both robot users and workshop personnel. Equipment design ensured there were no sharp edges or exposed electrical ports. Additionally, feedback from workshop users and technicians was incorporated to ensure suitability for the university environment. By managing these risks effectively, we were able to deliver a reliable and safe AMR that enhances the operational efficiency of the workshop.

3 Research & Development

In this chapter, the research and development of the AMR is explored in depth. It begins with a comprehensive state-of-the-art review of all aspects of current AMR technology. This review provides the essential background knowledge and expertise needed to proceed to the design and implementation phase with a solid foundation. The design phase involves detailed component selection, including sketches, block diagrams, and circuit diagrams. Each component is carefully chosen to ensure optimal performance and compatibility, considering availability from the approved suppliers. The chapter also completes testing of various components, such as sensors and motors, to ensure they meet the required specifications and function correctly within the system. Additionally, the chapter covers the integration of these components into the overall system, addressing any challenges and solutions encountered during this process. The testing phase includes evaluations to validate the functionality and reliability of the AMR, ensuring that it performs as expected in the dynamic workshop.

3.1 Sherpa B Analysis

One of the first steps in the research process was studying the internal features of the Sherpa B AMR as shown in *Figure 14*. The outer covers were removed and the hardware was studied. This was to familiarise the team with the typical functions to expect from an AMR beyond the datasheet and analyse what the engineers at Sherpa did to solve various problems in their design.

Notable features included the removable battery which could be charged independently of the Sherpa B. This made the charging of the AMR much easier as the battery could be carried to any convenient plug. It also enabled, with the use of multiple batteries, the charging of a battery while the AMR is running, which could then be swapped out when the inserted battery ran out of charge.

Another significant feature, which would be incorporated into this project's AMR design, was the LiDAR for mapping, positioned underneath the platform with thin struts supporting the load that would be placed on top of the AMR. This gave the LiDAR a near-360-degree range of vision while still allowing the top of the platform to be used for transporting components.

Figure 14: Sherpa B Internals

3.2 State of the Art

The State-of-The-Art (SoTA) analysis delved into the five key subsystems: ECU & Navigation System, Powertrain Management, Electrical Power Management System, Frame & Carrying System, and UI & Ergonomics. The purpose of the analysis was to provide a current overview of the latest developments in AMR technology. By compiling and analysing the best available information, the SoTA analysis served as a foundation for collecting knowledge, identifying challenges, and opportunities for the product development.

The state-of-the-art documents in full can be found with the other files submitted for the External Integrated Project assignment. They explain in detail the research done on existing and experimental technologies for each of the subsystems. They also include justification for selection criteria for the components that would go into each of the subsystems.

This project was started in February, which meant that there was a significant amount of initial research that was required to gain an understanding of the AMR field, and each specific subsystem. As such, a **substantial portion of the project's effort** was dedicated to extensive research and generating the SoTA reports by each subsystem, due to the broadness of the topic.

The state-of-the-art reports are <u>essential</u> and <u>paramount</u> for understanding the remainder of this report, to recognise the reasoning behind each subsystem's design choices.

Shown in the following subsections are summaries of the state-of-the-art reports for every subsystem.

3.2.1 ECU & Navigation (OB)

Completed by Oliver Brunnock

The report presents a state-of-the-art analysis of Automated Mobile Robots (AMRs) by examining the design and functionality of Electronic Control Units (ECUs) and Navigation Systems. The objective was to determine an optimal solution to develop an open-source, modular AMR capable of transporting loads with precision and safety.

Through a comparative analysis, various ECUs and sensor technologies are considered based on their performance, reliability, and integration capabilities. The report details the architecture and communication protocols as well as various sensors such as Light Detection and Ranging (LiDAR), infrared and 3D cameras. Additionally, navigation algorithms such as Simultaneous Localisation and Mapping (SLAM) were discussed as well as the differences between visual SLAM (vSLAM) and LiDAR based SLAM.

Optimal configurations that balance cost, complexity and functionality were identified. The research showed that SLAM is the superior navigation method due to its capabilities in mapping and localising, as well as its ability to support a wide variety of sensors for detecting obstacles, humans and path planning. The Raspberry Pi is highlighted as the most suitable physical ECU due to its affordability, compactness, and compatibility with the Robot Operating System (ROS) and Ubuntu.

For the sensors, LiDAR was shown to be the best for navigation due to its precision, while 3D cameras were identified as being best suited for obstacle detection because of their ability to interact with complex environments.

The results serve as a guide in selecting the most suitable technologies for the 'brain' of an AMR. The report also touches on the future of AMR technology in addition to the latest trends and innovations.

3.2.2 Powertrain Management (TC)

Completed by Tom Coleman

The aim of the state-of-the-art report is to identify, describe and evaluate components and design methods related to the powered movement of an Autonomous Mobile Robot (AMR). The system must allow the AMR to carry a load efficiently from one location to another while avoiding obstacles. The main section details the current commercial applications of common powertrain components: Wheels, motors, motor drivers, brakes, and gearboxes. The section also reviews some less-conventional components used for motion and transport in robotics and other scientific applications.

The report identifies and describes three case studies of existing AMR designs: The Sherpa B, the ROMR open-source project, and the Taobotics Gemini-O. The case studies are evaluated based on their use of powertrain components. This is to determine which cases are the most effective in key performance areas. Areas of comparison include motor power, load capacity, maximum speed, and stopping distance. The most effective components from each case are extracted for use in future designs. These elements have an emphasis on a powertrain that produces high power and torque, to carry a heavy load. A low speed of up to 2 m/s was selected to provide a short stopping distance for the purposes of safety.

Based on the analysis, it was concluded that the system would use a pair of brushless DC motors, swivel casters for stability, disk brakes for emergency stops, and compatible motor drivers for this system and the ECU. Omni-wheels would be tested in a prototype to determine their effectiveness compared to conventional wheels.

3.2.3 Electrical Power Management System (JB)

Completed by Joe Bailey

This state-of-the-art report's investigation into the electrical power management systems of AMRs highlights the complexities and interconnections between the four main areas of energy storage, power converter, battery charging, and electrical safety systems in AMRs, utilising real-world examples for comparison. It also identifies suitable technologies for the AMR, utilising the technical requirement list for performance benchmarks.

The exploration and numerical evaluation of energy storage technologies has established Lithium-Ion batteries as the current standard for AMR applications due to their optimal balance of energy density, efficiency, and cost. However, the need for continuous advancement like for solid-state batteries, is identified.

Power converters are introduced along with the importance of efficiency in these systems which is only attainable with switching power converters, or multiple supplies. Additionally, it was found that many considerations need to be made regarding battery voltages, which are heavily reliant on other subsystems such as powertrains. Multiple robots are analysed, particularly their additional onboard power supplies and their capabilities.

A discussion on battery charging emphasises the significance of selecting chargers that complement the specific chemistry used, and an appropriate power capability which allows for faster recharge rates.

The relevant electrical safety systems were analysed such as electrical protection, and design considerations were investigated focussing on labelling, and emergency stops. The relevant standards were explored to ensure safe operation of the AMR.

In summary, the interdependence of all these systems requires an integrated approach to the design, with a great emphasis on understanding the requirements of each subsystem that requires power, to fully realise an efficient and reliable solution.

3.2.4 Frame & Carrying System (JOS)

Completed by Jenny Öborn Sandström

A detailed overview of the subsystem is provided, outlining its specific technical requirements. From this, four main areas were identified: shape & design, dimensioning, materials, and communication. Furthermore, a thorough literature review and comparative analysis of existing market products have been conducted.

The findings of this study emphasise the importance of incorporating modularity in the design process and aligning dimensions and material choices to meet customer requirements and expectations. Additionally, the AMRs analysed in this study exhibit various designs and features that could be beneficial for this project.

This report serves as a foundational document for future work, providing guidance for subsequent design decisions and product development.

3.2.5 UI & Ergonomics (ZM)

Throughout the State-of-the-Art report, the distinctive features of three Autonomous Mobile Robots (AMRs) were meticulously examined, encompassing their command user interface, ergonomics, safety measures, and programming interfaces. Among the AMRs assessed, Sherpa B and Omron emerged with the most effective command user interfaces, particularly for unspecialised users, owing to their high recognisability and usability, with Omron showcasing exceptional visual perception utilisation. In contrast, Gemini's interface, though straightforward, proved overly complex for unspecialised users. However, comprehensive help and documentation were available for all AMRs, aiding user understanding.

The significance of Gestalt Rules of Visual Perception and Nielsen’s Heuristics in designing recognisable and user-friendly interfaces was evident. In terms of safety and ergonomics, Sherpa B and Omron surpassed Gemini, demonstrating superior emergency stop placement and adherence to industrial safety standards. Utilising anthropometric data and following industry standards proved crucial for ergonomic considerations. Despite Gemini's lesser recognisability in its programming interface, it excelled for specialised programmers due to its use of Python, offering greater flexibility compared to the graphical-based languages of Sherpa B and Omron. This underscores the importance of providing separate programming options to cater to diverse user needs.

Overall, the comparison underscores the efficacy of evaluation criteria, tools, and principles for designing AMR interfaces and ergonomics, emphasising the positive impact of visual perception principles and adherence to safety standards on user interaction and experience.

3.3 Design Integration & Implementation

In this section, we delve into the detailed design phase, where we explore the component selection, user cases, sketches, CAD (Computer-Aided Design), and bill of materials, providing an in-depth examination of the design from conceptualisation to further implementation.

3.3.1 Component Selection

Following the completion of the State-of-the-Art analysis, the in-depth research provided guidelines for making the final choices for components. It was important that each subsystem integrated with each other, so some initial ideas were adapted to meet other subsystem requirements. There was also some influence from what components were already available for use in the workshop and the budget of €4000.

3.3.1.1 ECU and Navigation (OB)

Completed by Oliver Brunnock

SBC

Due to its powerful processor, compactness, and wide range of available resources, a Raspberry Pi was chosen as the Single-Board Computer (SBC). A Raspberry Pi 4B was readily available in the workshop for initial prototyping, which is compatible with Ubuntu Server 20.04 LTS. However, the new Ubuntu 24.04 will be used in the future, allowing it to run on the more powerful Raspberry Pi 5. The Raspberry Pi 4B was used for testing, as the Raspberry Pi 5 had not yet been delivered. The Raspberry Pi 4Bs USB ports will be connected to the two vision sensors, LiDAR and to the Arduino. This is shown in further detail in *Figure 36.*

Microcontroller

While a microcontroller alone lacks the processing power and connectivity of a Raspberry Pi, the Arduino Mega was selected for its ability to complement the Raspberry Pi when connected via UART (Universal Asynchronous Receiver Transmitter). The Arduino Mega, with its 54 digital input/output pins, 256KB of flash memory, and 4 hardware serial ports, can support multiple sensors effectively. By connecting it to the Raspberry Pi using UART, efficient serial communication is achieved, ensuring that commands and sensor data are transmitted. This setup provides the necessary motor control and sensor feedback capabilities. The ECU architecture is shown in *Figure 15* [24] [25].

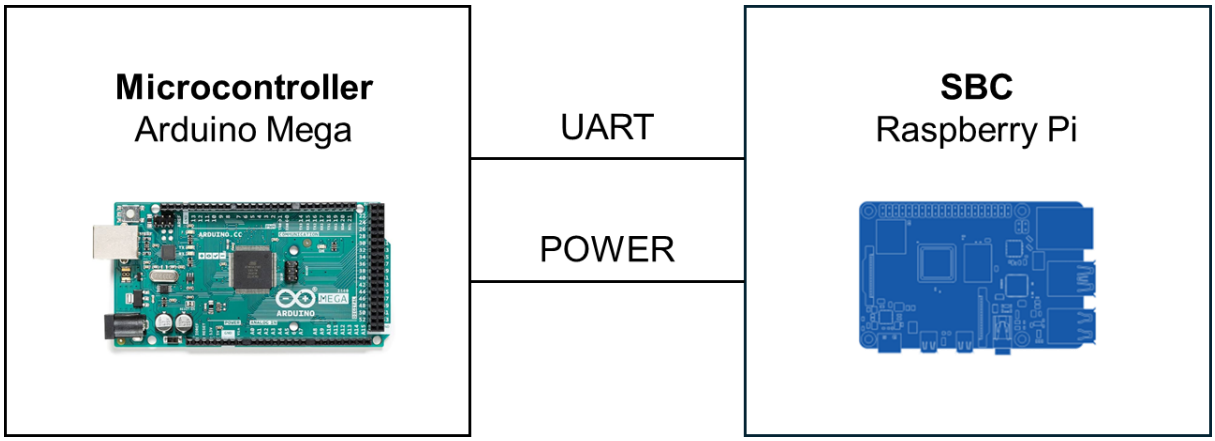


Figure 15: ECU Architecture [26] [27] [24]

LiDAR

The LiDAR selected is the RPLIDAR A3 was chosen as it provides a balance between cost and performance. Its 25m range can covers the entire workshop, and the scan rate of 5-20Hz meets the SLAM requirements. The 360-degree coverage means that only one LiDAR unit is needed. Its robust performance in mapping and navigation, reliability in various environmental conditions, and ease of integration with other system components means it is a suitable choice. The LiDAR is mounted as low as possible to ensure maximum obstacles such as chair and table legs are detected [28]. The LiDAR is shown in *Figure 16* is connected to the Raspberry Pi via a USB adapter.

Figure 16: RPLiDAR A3 [29]

Vision Sensor

The vision sensor selected for this project is the Intel RealSense D435i camera. One camera was already available in the lab, so another was ordered to ensure both vision sensors are of the same model. This camera features an integrated IMU with 6 degrees of freedom, a wide field of view, and a global shutter sensor, making it ideal for object detection. Its effective range of 0.3-3 meters is suitable for avoiding obstacles, including humans. This camera incorporates all the features of the Intel RealSense D435 with the addition of an IMU, where the performance metrics are analysed in the ECU and Navigation SoTA. This makes the vision sensor a suitable choice with its high resolution and depth capabilities. The two cameras will be positioned at the front and back of the robot to focus on detecting humans in its path. The Intel RealSense D435i is shown in *Figure 17* and both sensors are connected the Raspberry Pi via USB [30].

Figure 17: Intel RealSense D435i [31]

Infrared Sensor

As a fail-safe for the 3D cameras, infrared sensors are used to detect obstacles in close proximity or those that have been moved in front of the robot while it is turned off. These are modular, time-of-flight sensors that use the I2C protocol, enabling fast communication. They perform best in a range of 4 to 400 cm, effectively covering the close range that the vision sensors do not. The I2C protocol allows multiple sensors to operate in parallel. These are waiting on being delivered. The sensor is shown in *Figure 18* and will be mounted on the lowest rail in order to detect nearby objects [32].

Figure 18: Infrared Sensor [32]

3.3.1.2 Powertrain Management (TC)

Completed by Tom Coleman.

Motors

According to the technical requirements, the AMR would have a mass of up to 50 kg and carry a load of up to 40 kg. This gives an overall mass of 90 kg. The maximum acceleration was set at 0.5 m/s^2 and the nominal speed was set at 2 m/s.

The required motor torque to achieve an acceleration of 0.5 m/s^2 with a wheel diameter of 200 mm can be found using the following formula, derived from the equations for torque and Newton's Second Law of Motion [33] [34]:

Equation 1

$$T = mar$$

Where T is torque, m is the mass of the AMR, a is the acceleration, and r is the wheel radius.

The minimum torque required is 4.5 Nm. Accounting for a coefficient of friction of 0.7 between the wheels and the ground [35], the minimum torque is 6.42 Nm. This meant that each motor would need to produce 3.21 Nm of torque to achieve the nominal acceleration.

To achieve a speed of 2 m/s, the motors would need a rotational speed determined by this equation [36]:

Equation 2

$$\omega = \frac{v}{r} \times \frac{60}{2\pi}$$

This gives the rotational speed ω in revolutions per minute. V refers to the intended velocity of the AMR, and r refers to the wheel radius. This equation gives a rotational speed for a 200 mm diameter wheel of 191 RPM.

To save time when constructing the powertrain subsystem, an integrated off-the-shelf solution was chosen for the motor, the Crouzet 80189705 Brushless Geared DC Motor [37]. This was more expensive, but was operational out of the box, with minimal assembly. The motor had a maximum rated torque of 5.2 Nm, which was more than the 3.21 Nm required to achieve an acceleration of 0.5 m/s^2. The motor's maximum rotational speed was 120 RPM, which was lower than the ideal speed, but was the highest RPM available from an integrated geared motor which had sufficient torque to achieve the needed acceleration.

These geared motors formed the core of the subsystem. Each one incorporates a brushless DC motor, a planetary gearbox, a tachometer for measuring speed, and a driver to control speed, direction, and torque. They run off 24 V for the motor power. Speed and torque are controlled with a 0-10 V DC signal. The direction is controlled by a Boolean pin with the default direction at 0. The motor outputs 0-24V signals for the tachometer, torque limit, and direction feedback outputs.

Mounting Brackets

The brackets were used to fasten the motor to the frame. They would be manufactured using a laser cutter to get a component with specific dimensions for the project's needs. The

components were designed digitally. They were made with adjustable height to allow the drive wheels to be level with the swivel casters.

Voltage Conversion Circuits

A non-inverting amplifier circuit was needed to increase the voltage from the Arduino to control the speed of the motors. Both motors can be run from a single amplifier, as a high current is not needed for the speed control input, only a voltage of up to 10 V. As the direction of the motors is controlled independently of the speed, the motors can be run at the same speed, and still allow the AMR to turn effectively [38].

Potential divider circuits are needed to reduce the high-voltage feedback outputs from the motors, such as speed and direction. This allows the Arduino to process the signals [39].

The design of the voltage conversion circuits is covered in more detail in *3.3.9 Powertrain Circuit Testing*.

Wheels

The drive wheels connected to the motors are used to transfer power into movement for the AMR. A diameter of 200 mm was used to determine the motor's required torque and speed. 200 mm drive wheels are available from companies such as Blickle [40], but would require a spacer in the centre due to the motor having an axel that would not fit the wheel. The Blickle GTHN 100/20H7 [41] is a 100 mm diameter wheel that is compatible with the motor and was used in CAD models instead of the 200 mm wheel. However, due to challenges encountered when ordering the wheels, it was decided to 3D print a set of 100 mm diameter wheels out of ABS, that would fit the motor shafts.

Swivel Casters

The swivel casters are not powered but are used to stabilise the AMR and ensure that the load is distributed in a way that prevents the motor's mounting brackets from being damaged. An example of a swivel caster is shown in *Figure 19*. The AMR CAD was designed to incorporate both 75 mm and 100 mm diameter casters.

Figure 19: A swivel caster with a built-in lock. Image credit: Industrial Shelving Systems

Electrical Circuit Diagram

This circuit, shown in *Figure 20* incorporates elements to increase the output voltage or decrease the input voltage for the Arduino to use them effectively.

A non-inverting amplifier, powered by the same battery that powered the motors, increased the PWM voltage from the Arduino to a maximum of 10 V. Two 5.1 kΩ resistors were used in this diagram, but 4.7 kΩ resistors could also be used here.

For each motor control port, a potential divider was used to reduce the 24 V signal to a level usable by the Arduino, under 5 V. This required a ratio of R1/R2 as close to 19/5 as possible, but slightly higher if needed. 20 kΩ and 5.1 kΩ resistors were used, but similar resistors such as 18 kΩ and 4.7 kΩ are also usable.

The on/off and direction pins were controlled by digital pins on the Arduino.

All the elements share a common ground: The negative terminal of the battery. The Arduino's power supply is not shown in the diagram as it is connected to the Raspberry Pi via USB cable.

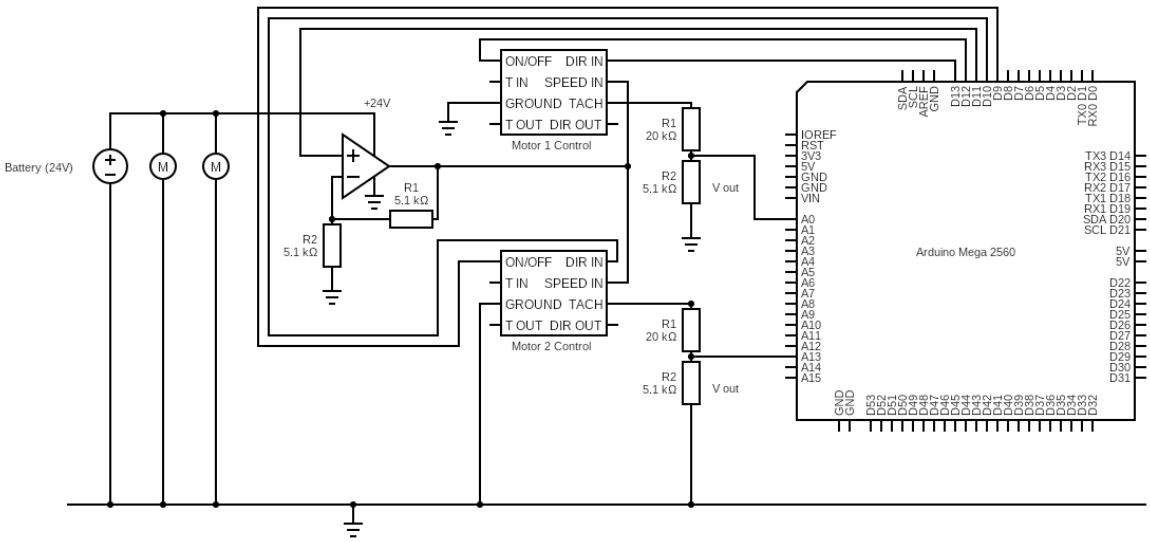


Figure 20: Circuit diagram of the final version of the motor control circuit.

3.3.1.3 Electrical Power Management System (JB)

Completed by Joe Bailey.

Battery

The power requirements for the battery were calculated by summating the instantaneous power draw of all the components installed as a worst-case scenario. The motor startup can draw 144W each [42] and the raspberry pi can draw up to 27W [43], resulting in a 315W maximum power draw, which is 13.1A at 24V.

The required battery size was estimated by calculating a value for the Wh/kg for multiple existing robots and multiplying this value by the maximum total loaded weight of our robot to give a value in Wh, which was then converted to Ah at 24V as shown in *Table 5*. Due to the unpredictable nature of the navigation algorithms and missions that the AMR will carry out, it is difficult to predict the exact energy required to satisfy the technical requirement of 2 hours of run time. However, basing the size from existing AMR solutions with an additional margin should give a sufficient solution.

Table 5: Battery Capacity Estimate [44], [45], [46], [47]

AMR	Battery Capacity [Wh]	Loaded Weight [kg]	Energy Density [Wh/kg]
Sherpa B	1392	333	4.180
MiR250	1631.34	333	4.899
PAL Aran	1440	197	7.310
HD-1500	3360	2085	1.612
		Average:	**4.500**

AMR	Average Energy Density [Wh/kg]	Loaded Weight [kg]	Estimated Battery Capacity [Wh]
Our Robot	4.500	90	405.01

AMR	Battery Capacity [Wh]	Voltage [V]	Estimated Battery Capacity [Ah]
Our Robot	405.01	24	16.9

The LiPO4 batteries selected in the state-of-the-art report were not available from any of the approved suppliers for the university. After considering the various alternative options, Li-NMC batteries were selected from INNPO due to their lightweight nature, high energy density and relatively low cost. Two 24V 12Ah batteries (*Figure 21*) were specified to be run in parallel increasing the total capacity to 24Ah to meet the estimated capacity of 16.9Ah (*Table 5*).

The two batteries in parallel were needed also to meet the calculated instantaneous power requirement of 13.1A, which assuming a 1C discharge rate for the batteries, means that the batteries can provide 24A when run in parallel which is sufficient. 24V batteries were selected to reduce the complexities involved in charging and balancing two batteries in series.

Figure 21: 24V INNPO Battery Selected [48]

DCDC Converter

The original 15W converter selected in the SoTA report was changed to a 60W 24-5V converter shown in *Figure 22* to fulfil the greater 27W power supply requirement of the Raspberry Pi 5 [43] which includes its LiDAR, depth cameras, Arduino and touchscreen power requirements. The 33W margin was added on top of this to allow for additional components to utilise the 5V supply in the future.

Figure 22: RSD-60G-5 Meanwell DCDC Converter Selected [49]

Battery Charger

The accompanying INNPO 24V battery charger shown in *Figure 23* was selected for compatibility with the selected battery, and for its low cost and high power of 72W.

Figure 23: 24V Li-Ion Battery Charger Selected [50]

3.3.1.4 Frame and Carrying System (JOS)

Completed by Jenny Öborn Sandström.

Materials

The choice of material for building the AMR is aluminium, and more specific aluminium profile type B, *Figure 24.* These profiles are known for their strength and durability. They provide a robust frame structure that can withstand the stresses and vibrations encountered during the operation of an AMR.

B-Type aluminium profiles are designed to be modular, meaning they can be easily adapted and reconfigured as needed. This is a significant advantage in the project, as it allows for quick adjustments, the addition of new components, or design changes without having to start from scratch. The profile system simplifies the assembly and disassembly of different parts of the robot, which also facilitates maintenance and upgrades. The cost-effectiveness of aluminium profiles compared to other materials with similar strength and durability is another advantage. The modular nature can further reduce costs for prototypes and redesigns, as the same components can be used in different ways in various iterations of the design.

These profiles are compatible with a wide range of accessories such as corners, brackets, and connectors, *Figure 24*. Such accessories make it possible to build desired structures and ensure stable connections between different parts of the robot.

A professional appearance of the finished robot is important according to client. Aluminium profiles contribute to an industrial aesthetic, making the product more appealing to users. These profiles are frequently used in the S.mart workshop, and the material was already available at the university.

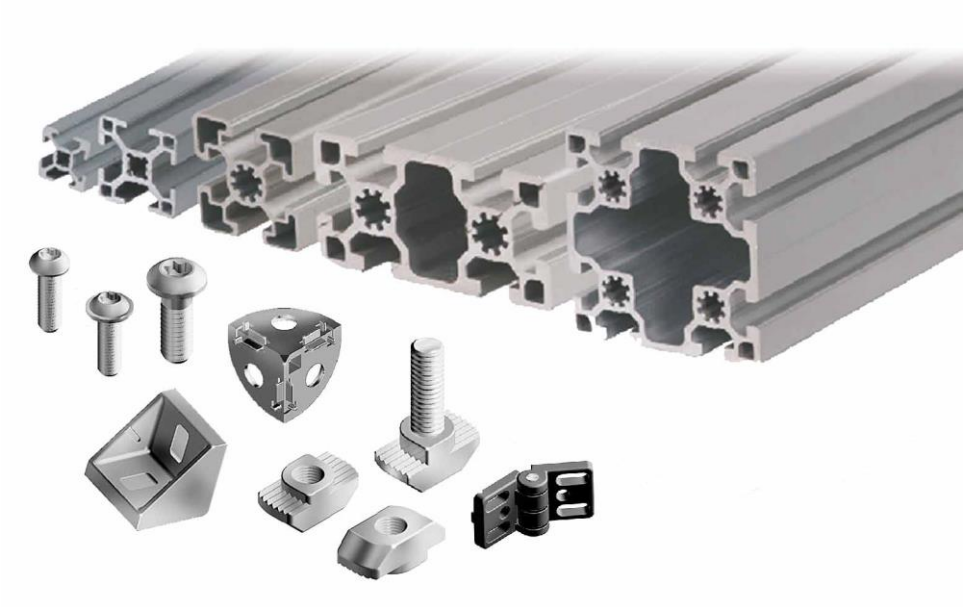


Figure 24: Aluminium profiles and accessories [51].

For the details on the AMR, 3D printing and laser cutting of plastic parts were chosen. 3D printing offers flexibility and precision, allowing the creation of complex and customised parts quickly and efficiently. This facilitates rapid design adjustments and iterations. Laser cutting also provides the ability to produce flat components with high precision and clean edges, which is important for structural parts and mounting plates.

The ABS and acrylic used are lightweight materials, which reduces the overall weight of the robot. Additionally, plastic is cost-effective and durable, making it an ideal material for both prototypes and finished components.

Load Sensors

It was decided not to include load sensors in the first prototype of the AMR. Although the use of load sensors would have been beneficial for monitoring and managing the robot's load capacity, this was a wish and not a requirement in the technical specification.

Project priorities focused on ensuring that required functions were achieved. Load sensors, while valuable, would have required additional time for integration and testing.

3.3.1.5 UI and ergonomics (ZM)

Completed by Zal Motaframm.

User Interface Design Layout

Utilising the best features of the SoTA AMRs and their user interface, a preliminary layout was created that highlights the flow of what the user interface should look like, to cater to the needs of the client and the technical requirements. As seen in *Figure 25*, the flow demonstrates three distinct options for the user, based on the user type.

Thus, the programmer and technician users will have access to higher levels of functionality within the Programme Interface, as desired initially by the Use Case Diagram *Figure 25*, with a layer of security to defend against unspecialised error. The standard user, who is expected to be unspecialised, will have access to lower-level functions within the Command Interface for assigning and programming simple missions. Therefore, with the selected components and ROS, the design of the user interface flow can be implemented and integrated into the AMR system.

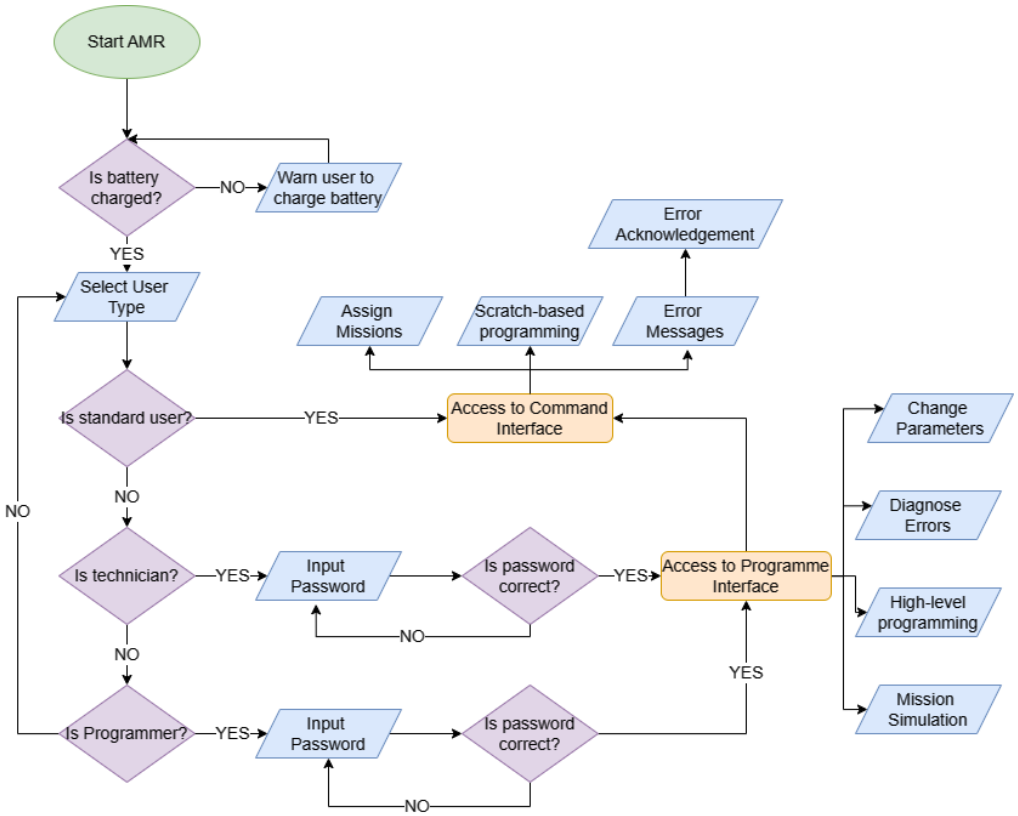


Figure 25: User Interface Flowchart diagram.

To display the output of the user interface, which will be coming from the Raspberry Pi 5 , the selected ECU for the AMR, a compatible Liquid Crystal Display (LCD) screen is required. Furthermore, to adhere to the usability heuristics mentioned in the SoTA, it would be desirable that the LCD is touchscreen and large enough for the user to assist with recognisability. Therefore, a 7” Raspberry Pi official display screen was selected, which included an extra display driver [52] as seen in *Figure 26*. However, as this display screen was made before the release of the Pi 5, an extra display adapter cable is required to assist with compatibility [53]. Furthermore, this display was chosen as it has mounting holes available in positions that are compatible with panel mounting options and can be mounted so that the screen is flush with

the borders of the AMR. By reviewing three separate supplier quotations, the best suppliers were found for both the LCD screen [54] and display adapter cable [55].

Figure 26: Raspberry Pi Touch Display Screen, with display driver and panel mounting accessories [52].

Emergency Stop Design Flow

As discussed in the SoTA, the emergency stop is a vital interaction between the user and the system, when concerned with user safety and regulation. Therefore, to make sure this feature is implemented, the flow of procedures was created to make sure it adheres to the Stop 1 Category in ISO-13850:2015 [18]. This category of stop is used mostly in small industrial control systems, where the power is removed, by use of electromechanical switching devices, once the machine actuators have ceased movement using the machine's power.

Thus, this procedure must be integrated with the EPMS, ECUN and the PTM subsystems, by using electromagnetic relays to switch the flow of power, to actuate the breaks, after which power is removed from the motors. To make sure the user is safe, there must be a level of user acknowledgment and verification of safety, to then continue the normal flow of operations. To demonstrate the flow of procedures, a flow diagram was created in *Figure 27*, to assist with the design of the architecture.

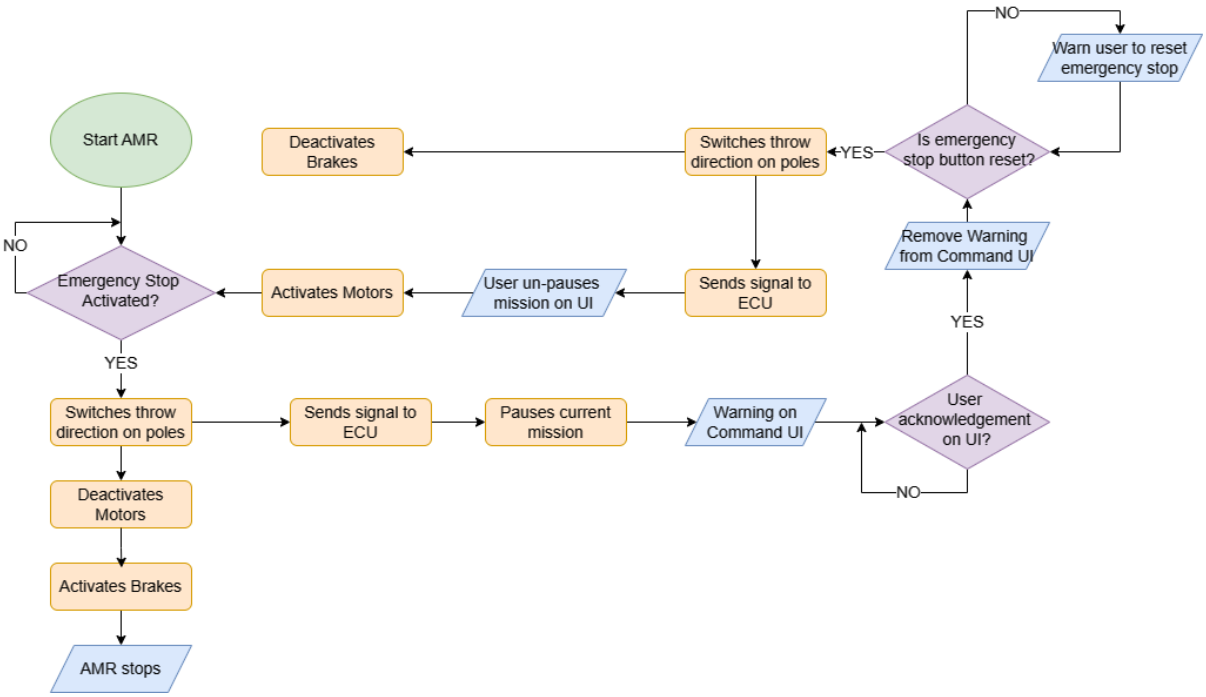


Figure 27: Emergency Stop flow of procedures chart diagram.

Therefore, to make sure this flow is maintained, the appropriate type of emergency stop must be selected. For this, there are two modes to switch between when the emergency stop is pressed and there are two flows of signals when the emergency stop is activated. The two flow branches are demonstrated in *Figure 27*, when the stop is activated, there are two signals sent: one to the ECU and the other to deactivate the motors. Hence, this set-up would require a specific type of switch known as a Double Pole, Double Throw (DPDT). This would have to be integrated with an electromagnetic relay, as required for a Category 1 stop and would be a monostable switch that has current flowing from two pole to two throws at each given state – switching between 2 Normally Open and 2 Normally Closed (2NO/2NC) states, *Figure 28* [56].

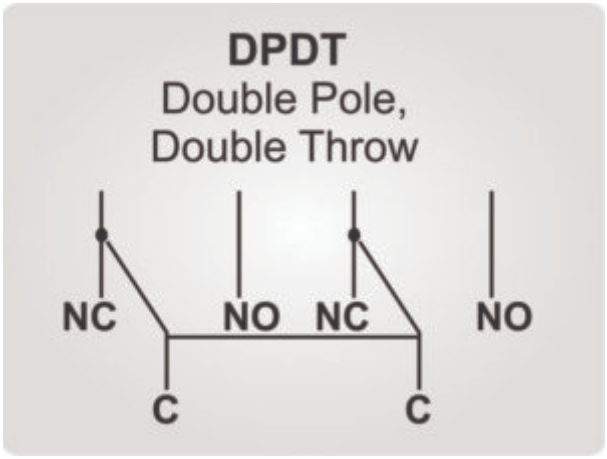


Figure 28: DPDT 2NO/2NC switch diagram [56].

Therefore, to select the correct emergency stop, the design of the operator command panel must be considered and thus, the button must be able to be panel mounted and be available from an approved supplier. The best switch available was a RS OEM (France) switch, the *"RS PRO Illuminated Emergency Stop Push Button, Panel Mount, DPDT, IP65", Figure 29* [57].

Figure 29: RS PRO Illuminated Emergency Stop Push Button, Panel Mount, DPDT, IP65, No. 745-2442 [57]

3.3.2 Sketches (JOS)

Completed by Jenny Öborn Sandström.

Early sketches were created to explore and visualise various design ideas for the AMR, *Figure 30.* The sketches allowed for iteration of concepts, providing the team with a better understanding of the aesthetic aspects of each design. By starting with sketches, potential problems could be identified, and solutions developed before investing more resources in detailed design.

Once the basic shape was determined, more detailed sketches were created, *Figure 31*. Subsequently, CAD models were developed, laying the foundation for the prototype construction.

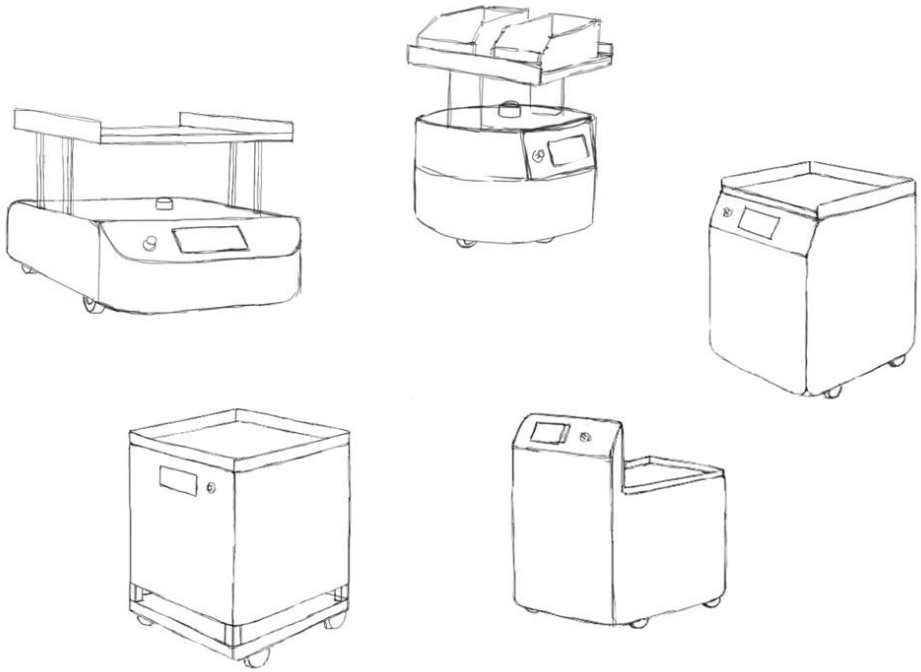


Figure 30: Early sketches of ideas for the AMR.

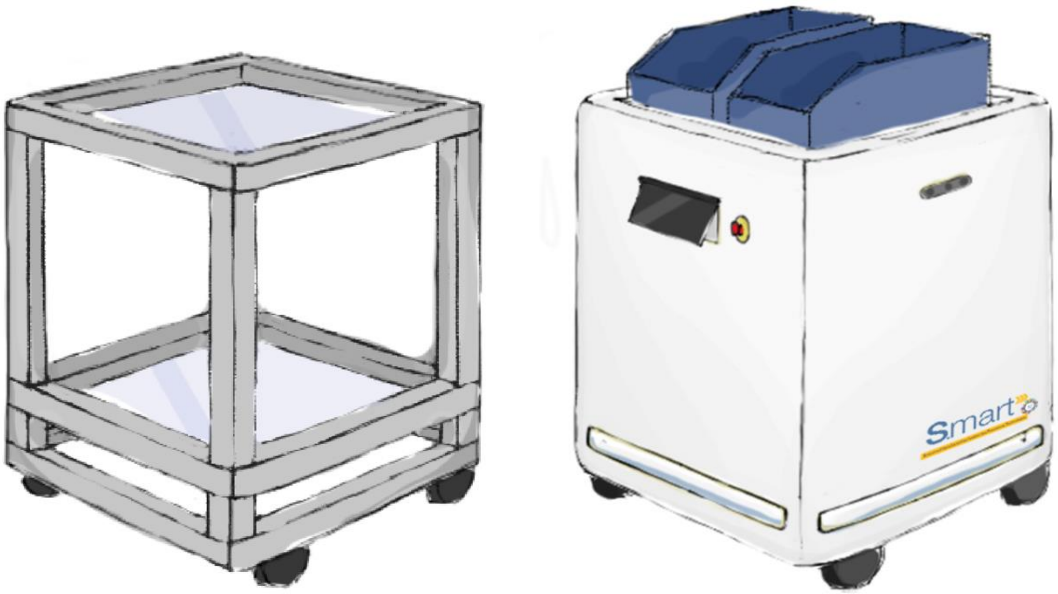


Figure 31: Sketch of AMR

3.3.3 Bill of Material (JB, OB, TC, ZM, JOS)

Completed by Joe Bailey, Oliver Brunnock, Tom Coleman, Zal Motafram, Jenny Öborn Sandström.

The Bill of Materials (BOM) was created in order of Noun Name (NNBOM) and part-level, which detailed all the components and parts required for the design of the AMR and is shown in *Appendix C: Noun Name* Bill of Materials Supplier Quotes. This allowed us to calculate the estimated cost of all the parts required to make the robot and to adjust our design accordingly. It also ensured that the required parts were able to be ordered by the University considering delivery and lead times. It also allowed the tracking of the weight of each component. From a list of approved suppliers, three quotes were acquired for each component, and so three columns were added in the BOM to facilitate this.

The Bill of Materials was used to dictate the ordering of components. The intention of the implementation process was to adapt to the lead time of the components and incorporate each one into the design as it arrived. The BOM structure for the different subsystems is shown below. This provided a clear breakdown of roles and responsibilities for ordering different components.

BOM Structure: (System Level Based)

- High-Level Assembly Number (Level 0): A00000
  - ECUN (level 1): A01000
    - Level 2+: A01XXX
  - PTM (level 1): A02000
    - Level 2+: A02XXX
  - EPMS (level 1): A03000
    - Level 2+: A03XXX
  - FCS (level 1): A04000
    - Level 2+: A04XXX
  - UIE (level 1): A05000
    - Level 2+: A05XXX
  - Miscellaneous (level 1+): A0000X

Part Labels

Based on the BOM naming structure, each part was labelled in a way that made its designation clear to anyone examining the AMR. For duplicate components like the motors and wheels, an additional number was appended to the designation to differentiate between components of the same type. For example, the motors were labelled "A02001_1" and "A02001_2", as shown in *Figure 32*.

Figure 32: Photos showing the labels of Motor 1 and Motor 2

RoHS Compliance

RoHS is a directive enforced by the European Union, intended to reduce the use of hazardous substances such as lead in electrical components [58].

As *Figure 33* shows, 19 of the 21 electrical components are RoHS compliant. Two of the components are not applicable. One of them, the batteries, instead follow a different protocol:

the Waste Electrical and Electronic Equipment (WEEE) protocol [59]. The remaining component is the Raspberry Pi 5 display cable, which does not contain any potentially hazardous substances. Any other components in the BOM are not electrical and are thus not applicable.

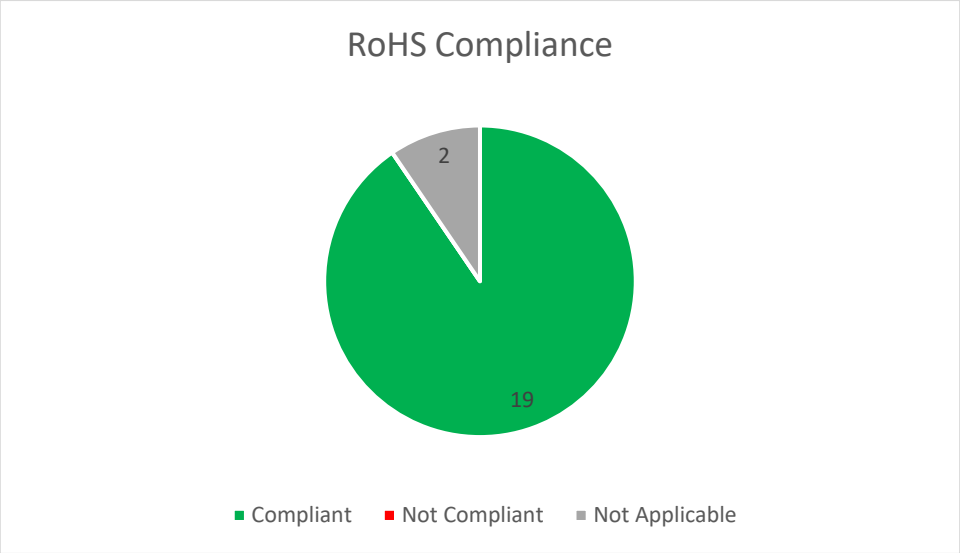


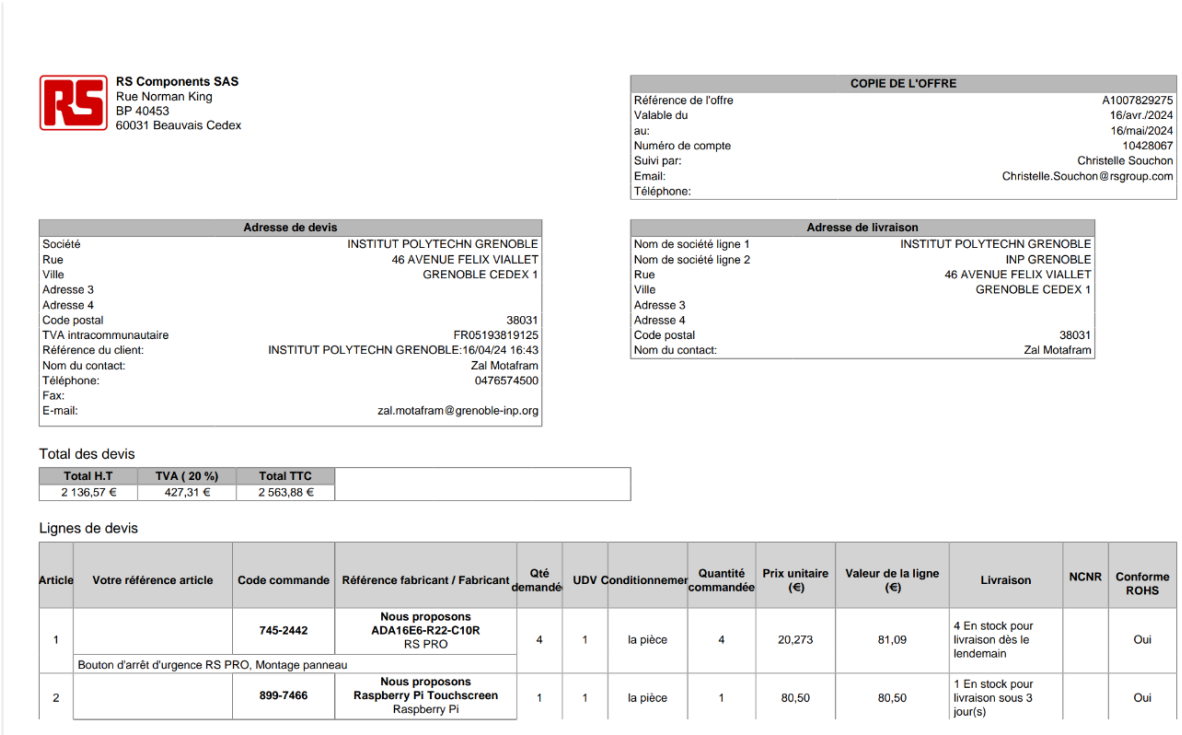
Figure 33: Graph of RoHS compliance of electrical components.

Supplier Quotes

To ensure that the components came from reputable suppliers, it was required to provide three quotes for each component. These quotes would be sent to the university's internal ordering team to be purchased using the project's budget.

The university had a list of trusted suppliers, which made up the majority of the sources for components. This was done to reduce lead time, as their purchases were approved more quickly than for non-trusted suppliers.

Components such as the motors and depth cameras were purchased from trusted suppliers, so they arrived more quickly. Components like the LiDAR and the Raspberry Pi 5 were ordered from non-trusted suppliers, so took longer to arrive. Components such as the motors and depth cameras were purchased from trusted suppliers, so they arrived more quickly. Components like the LiDAR and the Raspberry Pi 5 were ordered from non-trusted suppliers, so took longer to arrive. This limited the scope of the assembly until all of the components arrived. *Figure 34* shows an example of a quote received from RS Components, for the Raspberry Pi 5 and emergency stop button. It shows the price given for each unit, and the RoHS compliance of the devices.



RS Components SAS
Rue Norman King
BP 40453
60031 Beauvais Cedex

COPIE DE L'OFFRE	
Référence de l'offre	A1007829275
Valable du	16/avr./2024
au:	16/mai/2024
Numéro de compte	10428067
Suivi par:	Christelle Souchon
Email:	Christelle.Souchon@rsgroup.com
Téléphone:	

Adresse de devis	
Société	INSTITUT POLYTECHN GRENOBLE
Rue	46 AVENUE FELIX VIALLET
Ville	GRENOBLE CEDEX 1
Adresse 3	
Adresse 4	
Code postal	38031
TVA intracommunautaire	FR05193819125
Référence du client:	INSTITUT POLYTECHN GRENOBLE:16/04/24 16:43
Nom du contact:	Zal Motafram
Téléphone:	0476574500
Fax:	
E-mail:	zal.motafram@grenoble-inp.org

Adresse de livraison	
Nom de société ligne 1	INSTITUT POLYTECHN GRENOBLE
Nom de société ligne 2	INP GRENOBLE
Rue	46 AVENUE FELIX VIALLET
Ville	GRENOBLE CEDEX 1
Adresse 3	
Adresse 4	
Code postal	38031
Nom du contact:	Zal Motafram

Total des devis

Total H.T	TVA (20 %)	Total TTC
2 136,57 €	427,31 €	2 563,88 €

Lignes de devis

Article	Votre référence article	Code commande	Référence fabricant / Fabricant	Qté demandé	UDV	Conditionnemer	Quantité commandée	Prix unitaire (€)	Valeur de la ligne (€)	Livraison	NCNR	Conforme ROHS
1		745-2442	Nous proposons ADA16E6-R22-C10R RS PRO	4	1	la pièce	4	20,273	81,09	4 En stock pour livraison dès le lendemain		Oui
	Bouton d'arrêt d'urgence RS PRO, Montage panneau											
2		899-7466	Nous proposons Raspberry Pi Touchscreen Raspberry Pi	1	1	la pièce	1	80,50	80,50	1 En stock pour livraison sous 3 jour(s)		Oui

Figure 34: Quote from RS Components for the emergency stop button and Raspberry Pi

The BOM also aided in budgeting the project to ensure it was under the €4000 maximum cost, with the total sum of bought components totalling €2,987.23 with a breakdown shown in *Figure 35*. To clarify, as some items were already stocked in the workshop inventory, the order quantity does not match the quantity number mentioned in the BOM.

Item	Quantity	Total Cost (Including TVA)
Raspberry Pi 5	1	€ 110.55
RPLiDAR A3	1	€ 541.97
Intel Realsense	1	€ 343.36
Arduino Mega	1	€ 36.00
Crouzet Motors	2	€ 1,470.12
24V Batteries	2	€ 328.92
24-5V Converter	1	€ 42.13
Battery Charger	1	€ 29.00
Touchscreen	1	€ 55.80
Display Cable	1	€ 0.91
Mouse & Keyboard	1	€ 21.25
USB A - USB B Cable	1	€ 2.62
USB C - USB A Cable	1	€ 4.60
Total		**€ 2,987.23**

Figure 35: Project Purchases Breakdown

3.3.4 Block Diagram (OB)

Completed by Oliver Brunnock.

The detailed Electrical block diagram of the AMR is shown in *Figure 36*. The ECU comprises a Raspberry Pi connected to an Arduino Mega via UART. Two Intel RealSense 435i vision sensors and the LiDAR are connected directly to the Raspberry Pi via USB. The Raspberry Pi features a touch display screen through HDMI and connects to the host computer via Wi-Fi. A battery supplies 5V to the Raspberry Pi through a step-down converter. The Arduino Mega, serving as the microcontroller, is connected to four time-of-flight infrared sensors. The Arduino sends odometry, digital signals, and PWM signals via an amplifier circuit. The motors and this amplifier circuit are powered by a 24V battery.

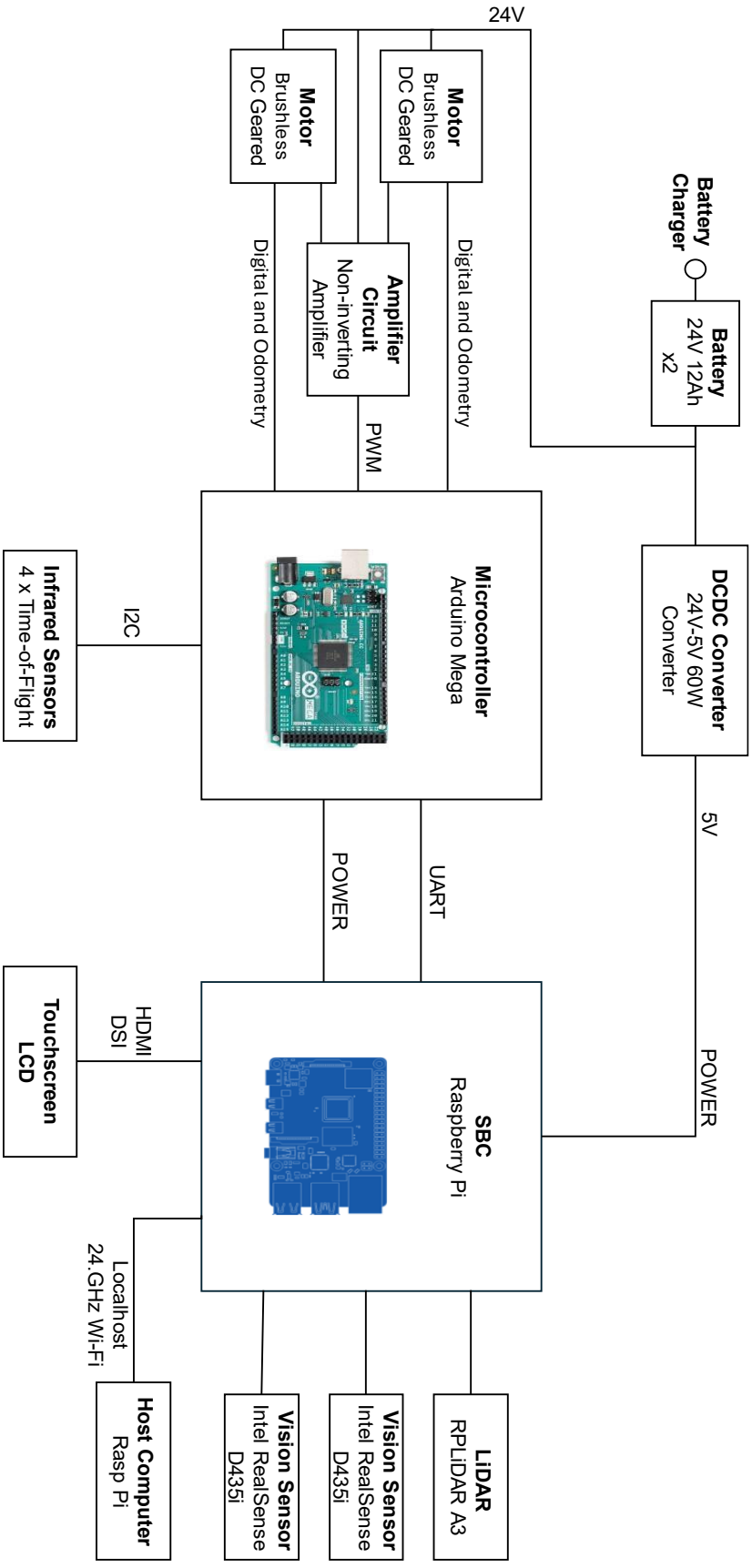


Figure 36: Detailed System Block Diagram

3.3.5 Circuit Diagram (JB, TC)

Completed by Joe Bailey, Tom Coleman.

Overall Electrical System

Figure 37 shows the complete circuit diagram, covering all the electronics used in the AMR's subsystems. For the ECU and Navigation, this includes the Raspberry Pi that acts as the central processor for the AMR, the Arduino used by the Pi to connect to the actuators, and the sensors such as LiDAR used for object detection and navigation. For the Powertrain, it contains the motors, and the voltage conversion circuits to allow them to be controlled by the Arduino. For the User Interface, it includes the touchscreen intended for user programming and control. For the Power Management, it contains the batteries and the DC/DC converters intended to supply power to the components, as well as the safety features like fuses, switches, and emergency stops to prevent damage to the AMR and its surroundings.

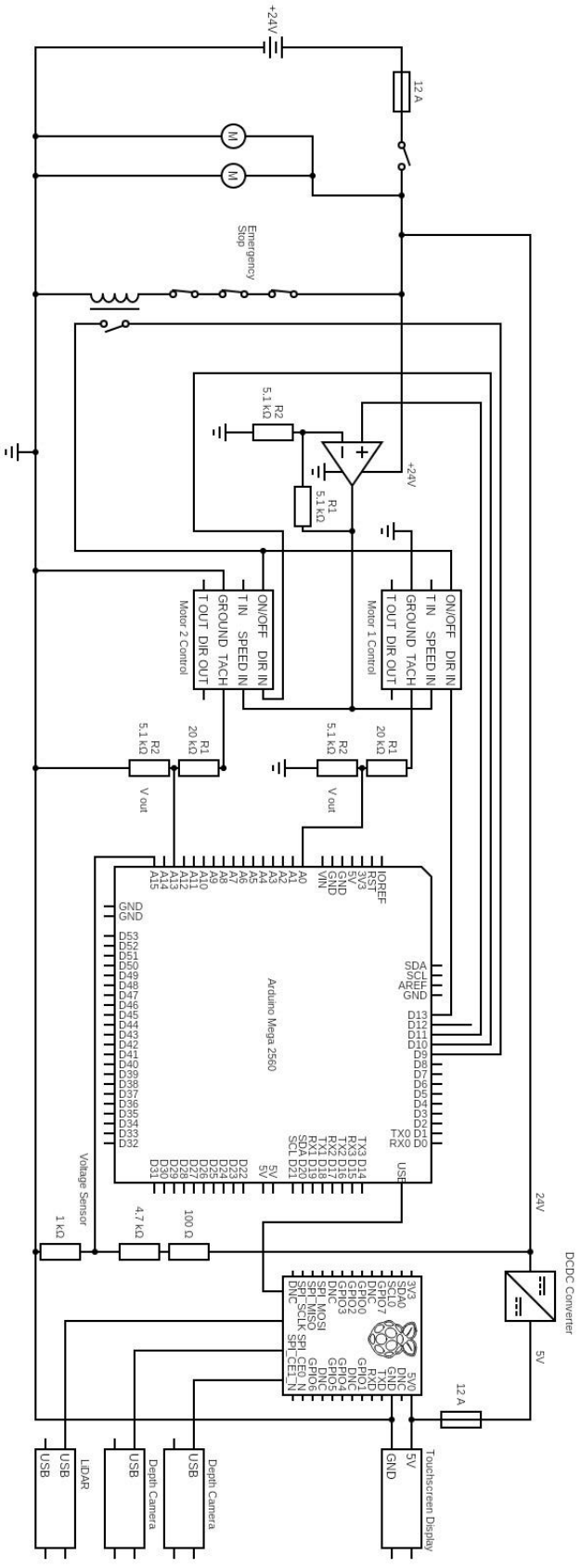


Figure 37: Circuit diagram for the overall electronic systems.

3.3.5.1 Powertrain (TC)

Completed by Tom Coleman.

The following circuit diagram in *Figure 38* shows all the electronic components needed for speed control and feedback between the Arduino and the motors.

In this diagram, the motors were controlled by a single 5 V PWM pin from the Arduino. A non-inverting amplifier boosted that voltage to up to 10 V. This allowed the Arduino to control the full range of speed. However, due to using a single output, the motors would be run at the same speed.

The on/off and direction signals were controlled by digital pins that output 0 or 1. The direction pins would enable the AMR to move in straight lines and turn about its central axis.

The tachometer pins on the motors outputted 24V PWM signals, which would be reduced by a pair of potential dividers and input into two analogue pins on the Arduino. The exact pins were not important, the pins selected for the circuit diagram were done for the sake of visual clarity.

Note that this circuit does not allow for torque control or direction feedback, as time constraints prevented those functions from being properly researched. In order for the torque to be fully controlled, an additional non-inverting amplifier and two additional potential dividers would need to be added to the circuit, to cover the input and output signals. The direction outputs also would need two potential dividers to reduce the 24 V signal to below 5 V.

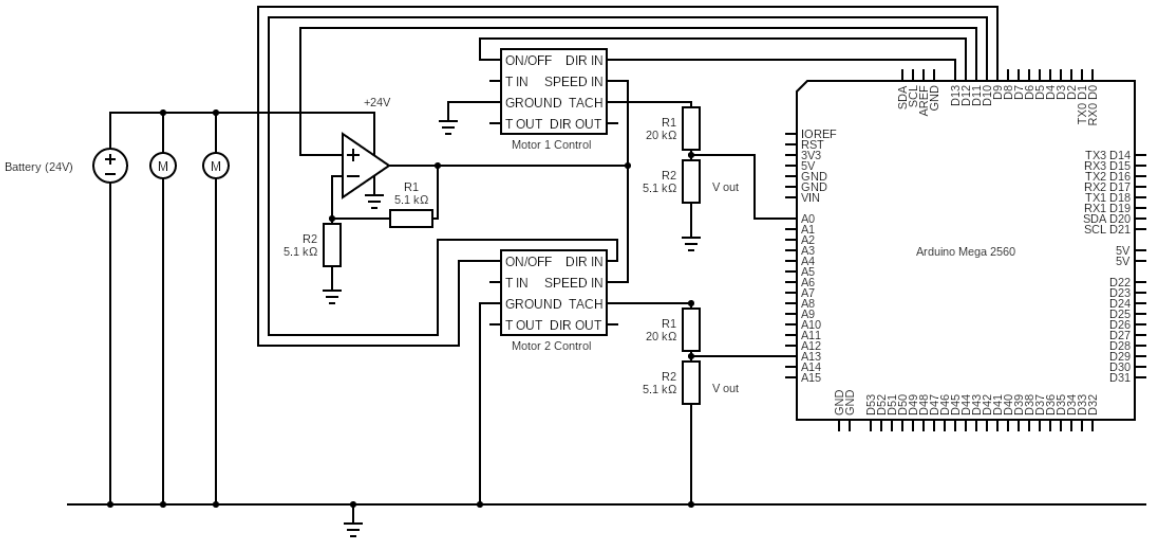


Figure 38: Circuit diagram of the Powertrain Subsystem Electronics

3.3.6 Vision Sensor Testing (OB)

Completed by Oliver Brunnock.

The selected vision sensor is the Intel RealSense D435i vision sensor. For initial testing of the camera, the Intel RealSense Viewer Software was used to capture high resolution images and depth accuracy under different object geometries [30]. This software was also launched on ROS 2 to make sure all the correct libraries were downloaded.

To measure the distance of a desired object, a Python script was developed which utilised the depth information provided by the camera. [60] By moving the cursor over the objects, the

script calculated the distance to the object based on the corresponding depth value. The camera was placed in a fixed position and connected to a laptop as shown in *Figure 39.*

Figure 39: 3D Camera Test Setup

The RealSense viewer enabled toggling between the different camera functionalities:

Stereo Module

Two Infrared Cameras capture the same object from slightly different angles. By comparing these two images, the camera calculates the depth of the objects. *Figure 40* shows Stereo mode being tested.

RGB Camera

This captures colour images which are used to provide colour data that can be overlaid on the depth images.

Motion Module

This includes the motion module which providing tracking data which is necessary for controlling the AMR. *Figure 41* shows the full functionality of the camera with stereo, RGB and the motion module overlayed.

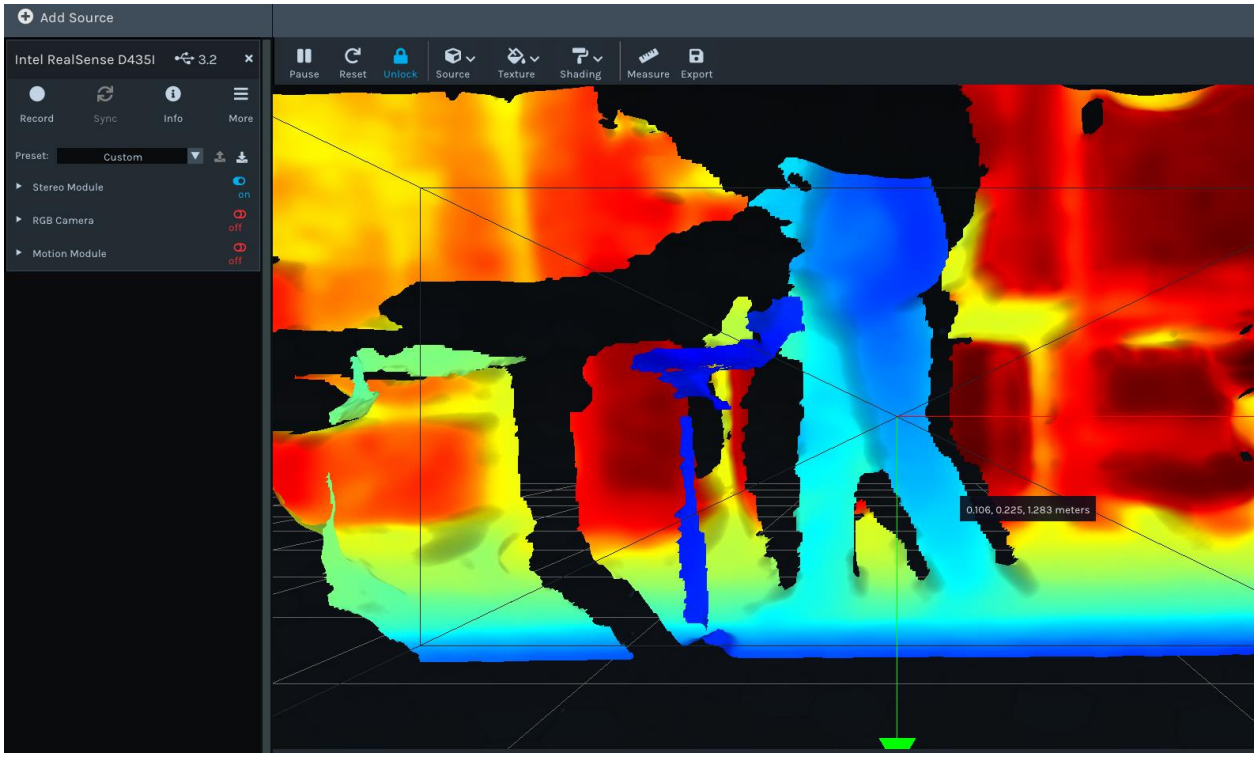


Figure 40: Intel RealSense Viewer: Stereo Mode

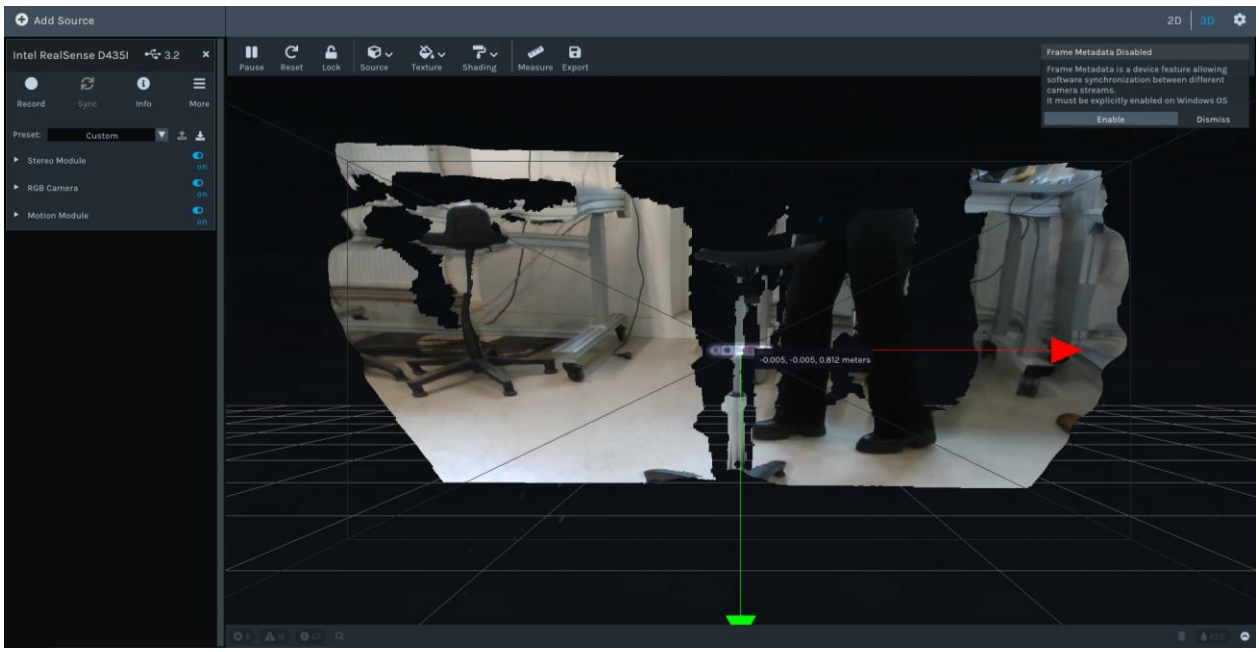


Figure 41: Intel RealSense Stereo, RGB Camera and Motion

With the Python code utilised, the RealSense D435i demonstrated to be able to detect humans from a range of heights and distances one of which is shown in *Figure 42* which is within the 'ideal range' of 0.3 – 3m. [61] The depth frame is shown on the right hand side. A minimum object detection was measured to be approximately 152mm as shown in *Figure 43*. The maximum range that was able to be detected is 9.8m as shown in *Figure 44.*

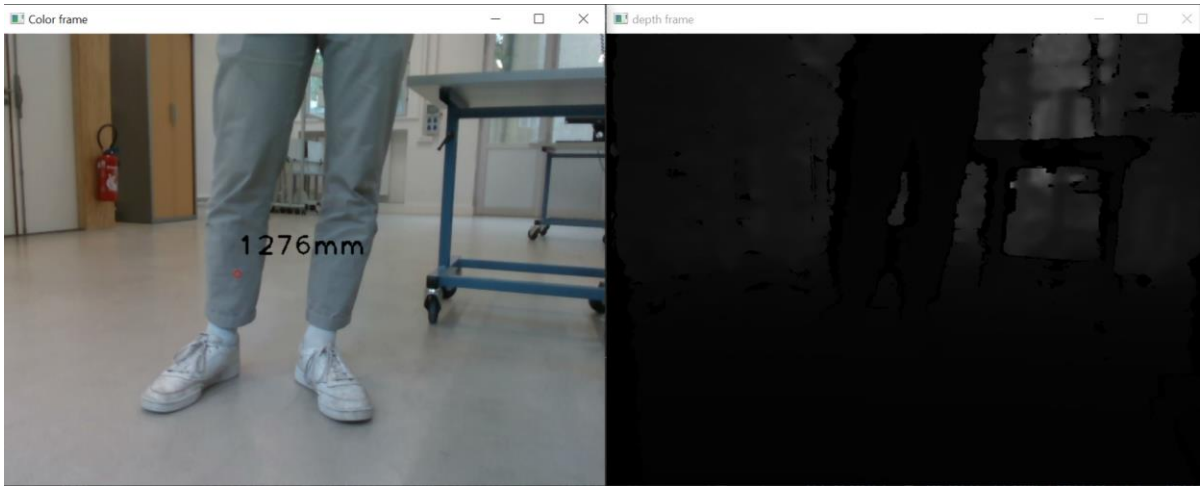


Figure 42: Vision Sensor - Human Detection

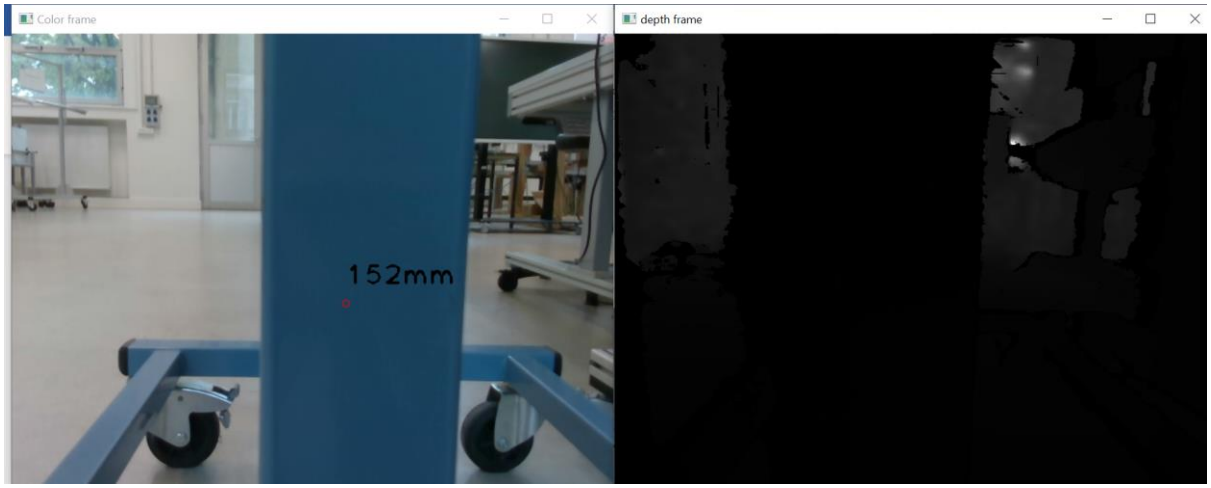


Figure 43: Vision Sensor: Minimum range.

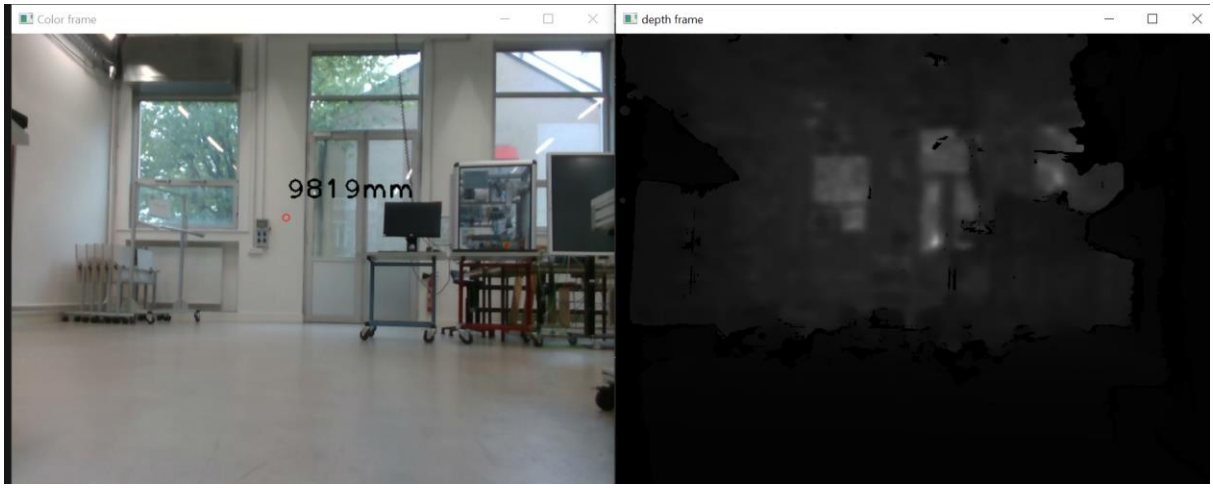


Figure 44: Vision Sensor: Maximum range.

After further testing and a discussion with the programmer, Quentin Levent, it was determined that placing the cameras higher on the robot allows for better identification of humans by their heads and shoulders, rather than their legs or torso. Consequently, the cameras have been adjusted to sit below the level at which the containers are carried, optimising the field of view. Also, the 3D printed mount which they sit on is angled upwards to further enhance the human

detection. This is shown in *Figure 45.* Overall, with these slight adjustments, the cameras now meet the requirements for human and obstacle detection.

Once the vision sensor was mounted appropriately, the camera packages were installed onto ROS2 using the librealsense [62] on GIT [63], and the vision sensor was able to be displayed on the on-board screen as shown in *Figure 46.*

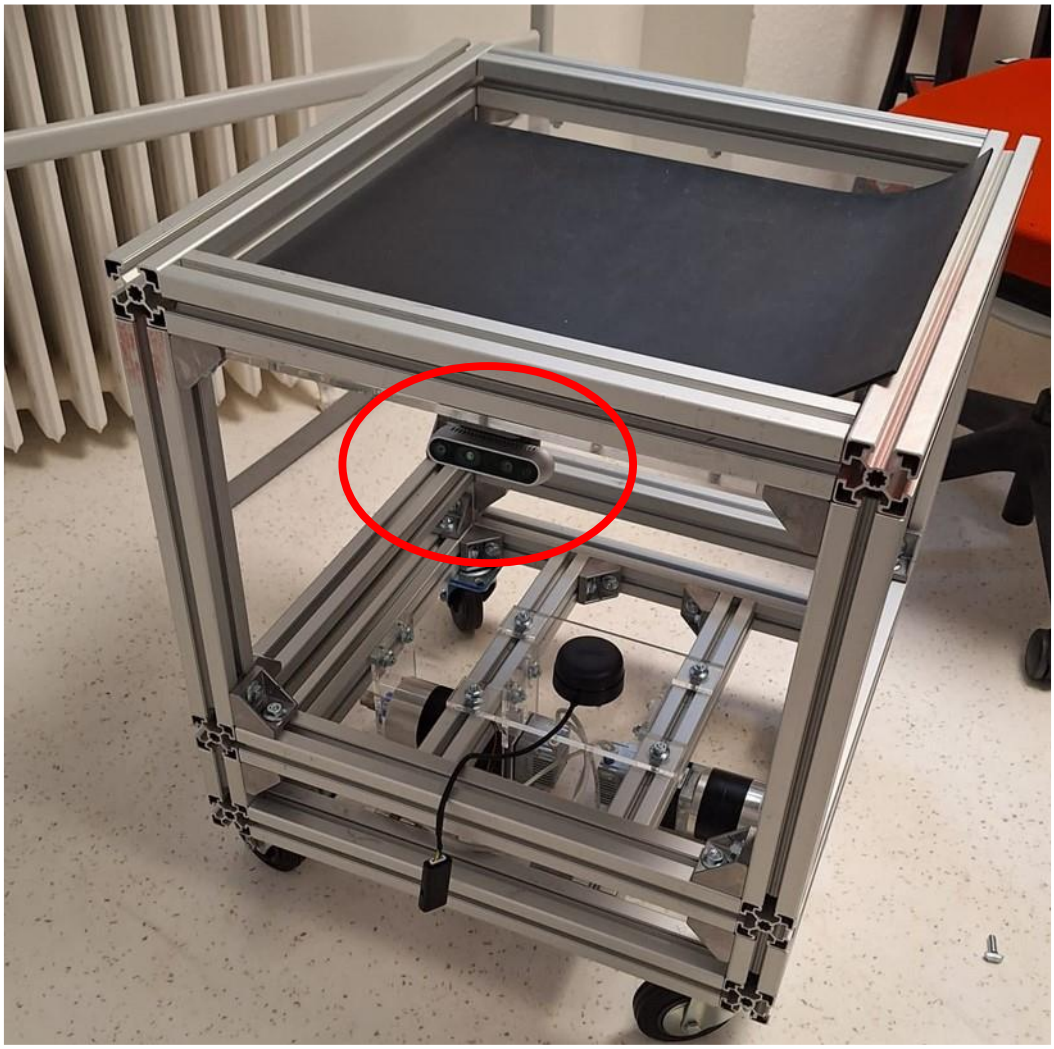


Figure 45: Intel RealSense D435i Positioning

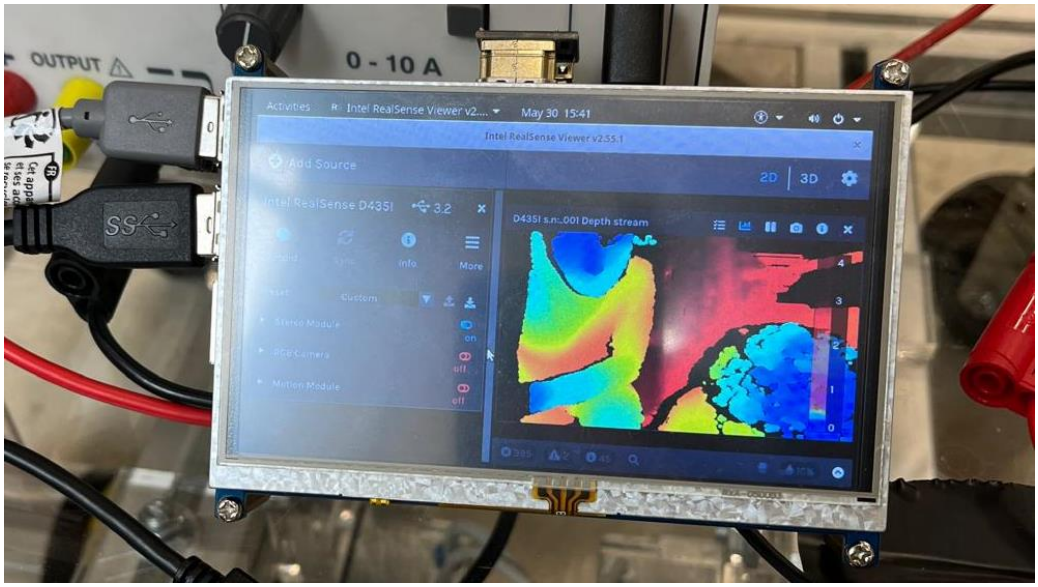


Figure 46: Vision Sensor on Display Screen with ROS2

3.3.7 LiDAR Testing (OB)

Completed by Oliver Brunnock.

The RPLiDAR A3 is mounted centrally on the robot, positioned as low as possible to optimise obstacle detection, including low-profile objects such as table and chair legs. The LiDAR is strategically placed between two 45x45mm beams, which creates a gap allowing the laser to perform a comprehensive 360-degree scan with minimal obstruction. The four core pillars of the robot's structure can be filtered out using the SLAM algorithm. This setup ensures maximum coverage and accuracy in mapping the surroundings. The exact mounting position and configuration are illustrated in *Figure 47.*

To validate the setup, the LiDAR was tested by downloading and compiling the RPLiDAR ROS package. It was connected to a host Raspberry Pi 4B via USB. Running the package enabled the point cloud to be displayed in Rviz, as shown in *Figure 48.* Various obstacles, including tables, chairs, walls, and humans, were accurately recorded within a range of 25 meters around the AMR. This is shown with the different colour points on the grid which enable live feedback of the obstacles in the workshop. Rviz also offers the option to measure these distances on the screen and rotate the plane into a 3D perspective, as shown in *Figure 49* [64].

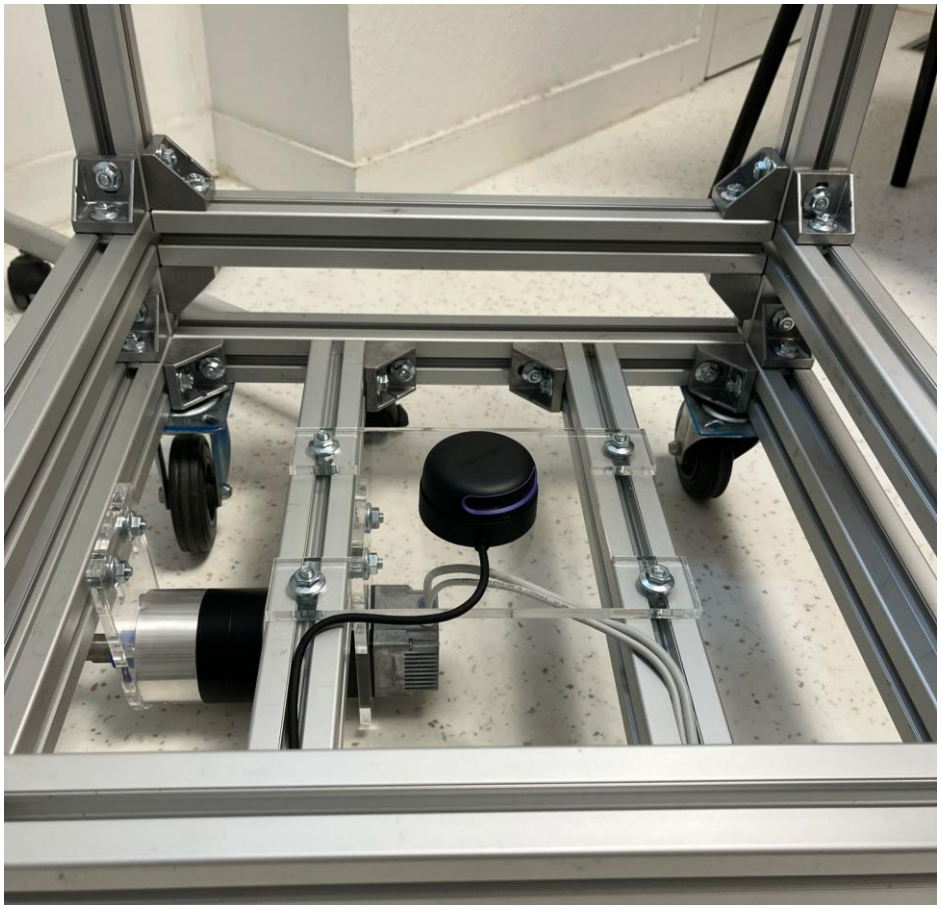


Figure 47: LiDAR Mounting

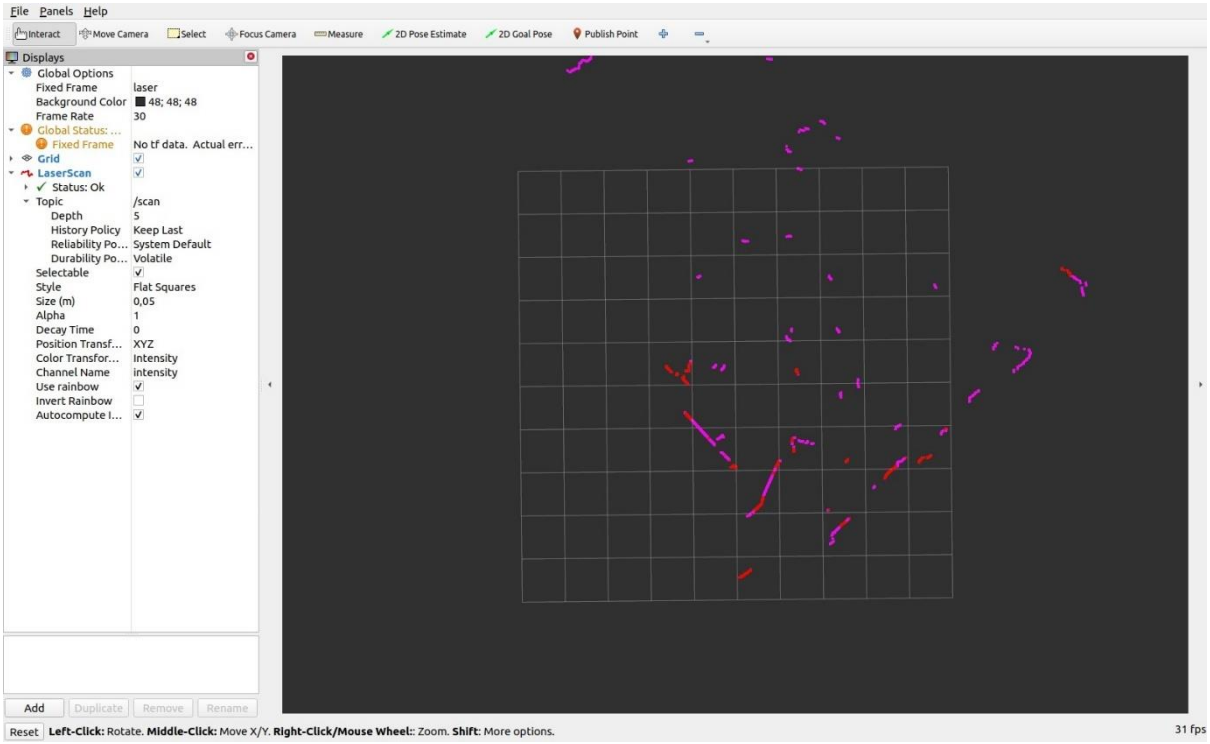


Figure 48: LiDAR Point Cloud in RViz

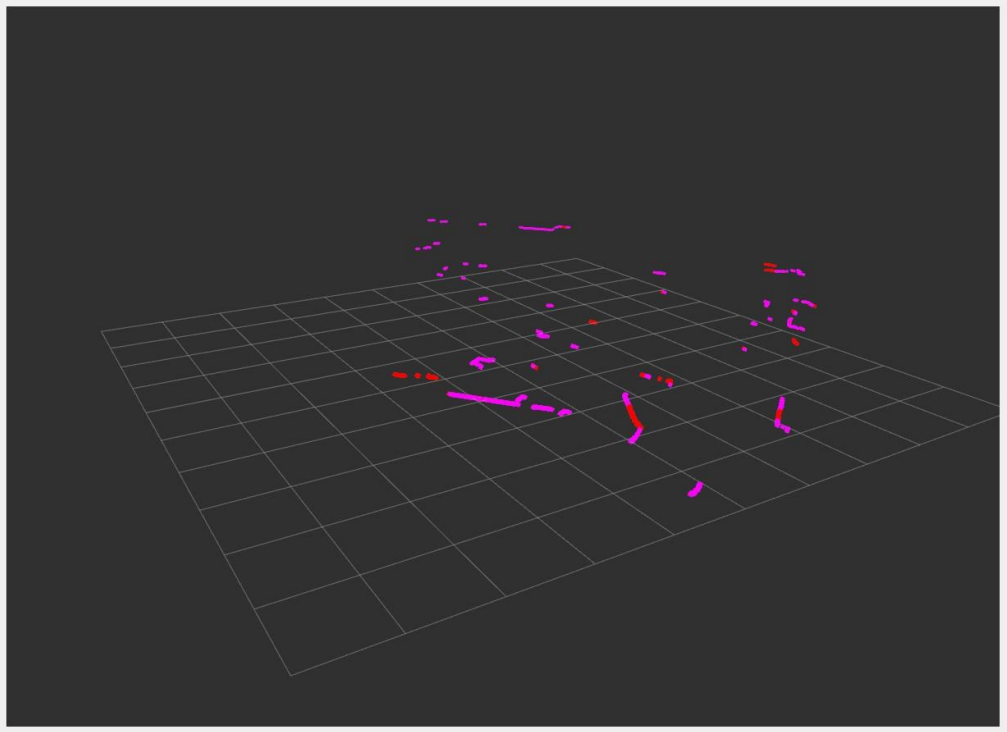


Figure 49: LiDAR Point Cloud in RViz 3D perspective.

3.3.8 Brushless DC Motor Testing (TC)

Completed by Tom Coleman

Once the DC motors were ordered, they were tested to determine their function and how they would be connected to the larger system. Initial testing was done with the motor, a 24 V DC power supply, and an Arduino Mega.

The motor has a 24 V power and ground cable. This provides the power for the motion of the motor.

The motor also has a control system for parameters like speed and direction. The control system has eight pins that perform the following functions [37]:

1. Green – On/Off
2. Yellow – Direction Input
3. Blue – Torque Limit Input
4. Orange – Speed Input
5. Black – Logic Ground
6. Brown – Tachometer Output
7. Purple – Torque Limit Output
8. Red – Real Direction Output

Pins 1 and 2 accept Boolean inputs from the Arduino. Pins 3 and 4 accept 0-10 V and are compatible with PWM.

Pins 6-8 output up to 24 V, which are not safe for an Arduino. Voltage conversion circuits were used to reduce the output voltage to safe levels for later tests.

Basic Movement

The highest voltage that the Arduino can output on its own is 5 V [24]. Because of this, initial tests used 0-5 V inputs for the speed control pin. This allowed the Arduino to control the motor up to half of its maximum speed without additional circuits.

The motor was connected to the 24 V power supply. The On/Off pin was connected to the Arduino and given an “On” input. The speed control pin was also connected, with the Arduino running 5V into it. The motor turned successfully. The speed, measured using a stopwatch, was determined to be roughly 80 RPM, over half of the motor’s maximum rated speed of 121 RPM [37].

The Direction control pin was connected and given an “On” output. The motor turned at the same speed as the previous test, in the opposite direction.

The setup of the first test is shown in the circuit diagram of *Figure 50*. With the first version of the circuit diagram, the speed of the motors was limited to half of the maximum. This was because the Arduino could output a maximum of 5 V into the 0-10 V speed control port.

Shown in the circuit diagram in *Figure 50* are the minimum components required to run the motors from the Arduino, with signals in on/off, direction, and speed control.

Figure 51 shows the first test of the brushless DC motor. The Arduino was controlled via USB cable from a laptop, and the control pins of the motor were connected to the PWM output pins of the microcontroller.

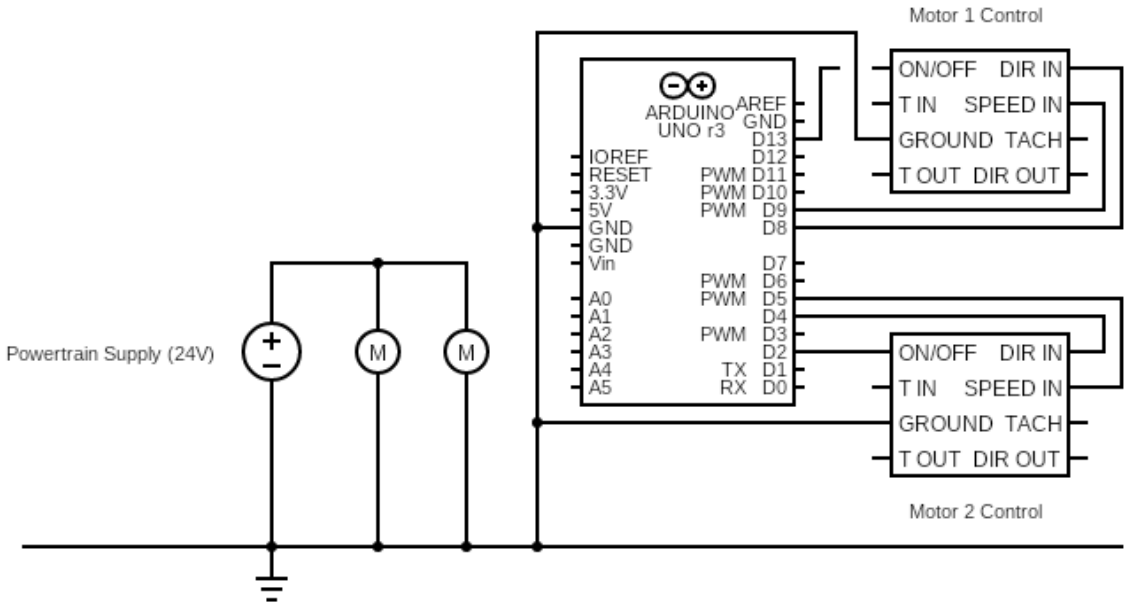


Figure 50: Circuit diagram of the first version of the motor control circuit.

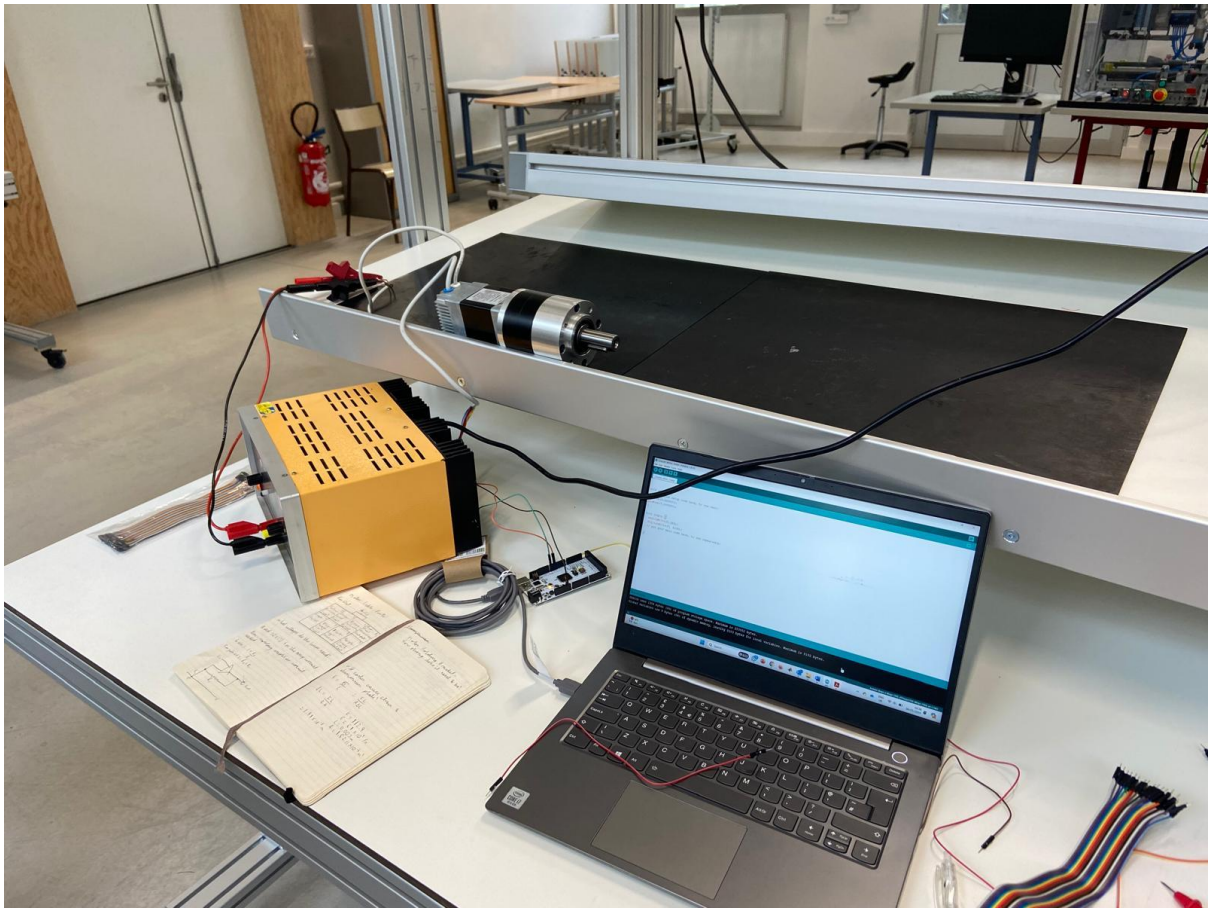


Figure 51: The first test of the brushless DC motor.

3.3.9 Powertrain Circuit Testing (TC)

Completed by Tom Coleman.

Amplified Voltage

Based on tests with just the Arduino, the motors could be successfully controlled, but only up to half speed with a 5 V signal. A non-inverting amplifier circuit like the one seen in *Figure 52* was used to boost the PWM voltage from the Arduino up to a maximum of 10 V. The circuit used a LM324 quad-op-amp and two 5.1 kΩ resistors. Note that only two op-amps were used in this case, but the LM324 would allow the system to be expanded if more voltage conversion circuits were needed.

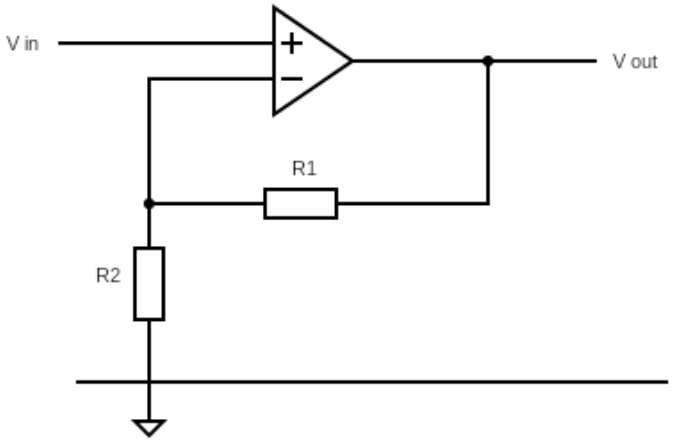


Figure 52: Circuit diagram of a generic non-inverting amplifier circuit.

For the non-inverting amplifier, a gain of 2 was required to boost the PWM voltage from 0-5 V to 0-10 V. Gain is represented as the ratio between the voltage output and the voltage input as shown in *Equation 3.* The gain of the circuit is determined by the size of the resistors R_1 and R_2 [38].

Equation 3

$$\frac{V_{OUT}}{V_{IN}} = 1 + \frac{R_2}{R_1}$$

For a gain of 2, resistors of equal value are needed.

Initial testing of the amplifier circuit was done to verify the voltage output before it was used with the motor. The expected gain of the circuit was equal to 2.

The power supply of the op-amp was firstly varied to determine its effect on the output. The LM324 has a rated power supply of 3-32 V, so it was determined what the ideal limit was, and if the device could be run from the same 24 V supply used for the motors.

Table 6 shows the output voltage based on different voltages used to power the op-amp. All results shown are for an input voltage of 5 V. Below 12-volt supply, the output voltage was lower than it should be. The output voltage was capped at 10 V when above a 12-volt supply was used. This showed that the amplifier could be driven directly by the 24 V supply used to power the motors, without an additional circuit to reduce it.

Table 6: Op-amp output voltage compared to supply voltage

Supply Voltage	Output Voltage
10	9.0
12	10.0
24	10.0

Once the amplifier circuit was tested on its own, it was connected to the Arduino. A PWM signal was output from the Arduino into the amplifier. A multimeter was used to measure the output

voltage. *Table 7* shows the results. Output voltage was slightly lower than expected for all PWM values measured, but within ±1% of the expected voltage.

Table 7: Multimeter readings of the amplified PWM signal from the Arduino.

Pin Value	Effective Arduino Voltage	Expected Output V	Actual Output V
0	0	0	0.003
51	1	2	1.970
102	2	4	3.960
153	3	6	5.960
204	4	8	7.960
255	5	10	9.970

Full Motor Test

The motor and LM324 were powered in parallel by the 24 V supply. The motor ran successfully up to its maximum speed, with a maximum current draw of 1.6 A.

Following the success of this test, both motors were powered by the supply as well as the LM324. The chip incorporated four separate op-amps which ran from the same power supply, so the Arduino PWM voltage could be amplified for both motors.

Both motors ran successfully, with a maximum total current draw of 3.3 A.

Single Amplifier System

Following the success of the test where each motor was run from a different PWM pin from the Arduino, each connected to a different op-amp on the LM324, the circuit was rearranged to test if both motors could be run from a single PWM pin and op-amp.

The second op-amp was removed from the circuit. The speed control pins of the motors were connected in parallel to the same output of the first op-amp. This would reduce the current that each would draw, but as the voltage was what dictated the speed, the loss of current had a negligible effect on the motors' performance.

The benefits of this simplified circuit were that it required fewer components and required only one PWM output from the Arduino.

The main drawback of this circuit compared to using two op-amps is that the motors' speeds cannot be controlled individually. They must be run at the same speed. Due to the directions being controlled by separate pins, the motors would still spin in opposite directions and would enable the AMR to turn. However, it would only be able to turn about its central axis. It would be unable to make turns with a larger radius like a road vehicle.

Tachometer Readings

For the motor to control its position, it must get feedback on its speed. The best source of feedback is the motor's built-in tachometer. However, initial testing showed that the tachometer outputted 24 V at its highest signal, making it unsafe for the Arduino's 5 V input pins. The signal from the tachometer needed to be fed back into the Arduino with reduced voltage. This was done with a potential divider circuit, as shown in *Figure 53*.

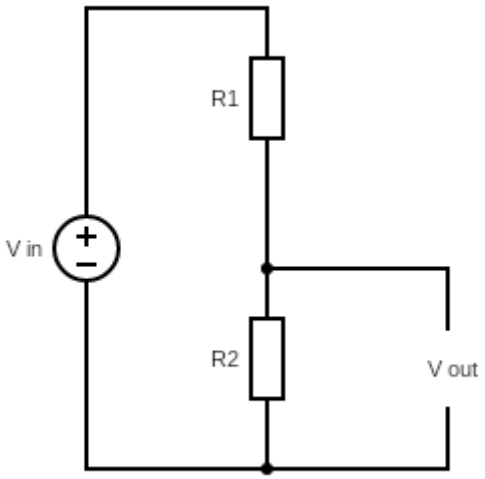


Figure 53: Circuit diagram of a potential divider.

For the potential divider, the input voltage is split proportionally based on the respective sizes of the resistors as shown in *Equation 4*. Output voltage is taken across the second resistor R_2.

Equation 4

$$V_{OUT} = \frac{R_2}{R_1 R_2} V_{IN}$$

Rearranging in terms of the ratio of resistances gives *Equation 5* [39]:

Equation 5

$$\frac{R_1}{R_2} = \frac{V_{IN}}{V_{OUT}} - 1 = \frac{19}{5}$$

The closest approximation to this ratio with the resistors available were two 10 kΩ resistors in series for R_1 and a single 5.1 kΩ resistor for R_2. This gives a maximum output voltage of 4.87V. This is within the acceptable limit for the Arduino's input ports.

The Arduino PWM output is represented by an integer from 0 to 255, with 255 representing a duty cycle of 1, or constant DC voltage.

An oscilloscope was used to measure the output waveform for different PWM duty cycles.

The output took the form of a square wave resembling a PWM cycle. As the duty cycle of the speed input increased, the duty cycle of the tachometer output reduced, with more time spent per period with a low signal. Unlike a PWM cycle, however, the frequency of the square wave increased with an increase in input duty cycle. The time within each period with a low signal stayed the same, but the high signal time reduced.

Note that a square wave was output by the tachometer regardless of if a true DC or PWM signal was input for speed control.

The waveform was inconsistent in the time domain, with the low signal sections experiencing significant disturbance and changing the lengths of the periods. The frequency of the wave could not be measured accurately, and often gave different results depending on when the oscilloscope's screen was paused.

Figure 54 shows how the tachometer output varies with the Arduino's PWM duty cycle.

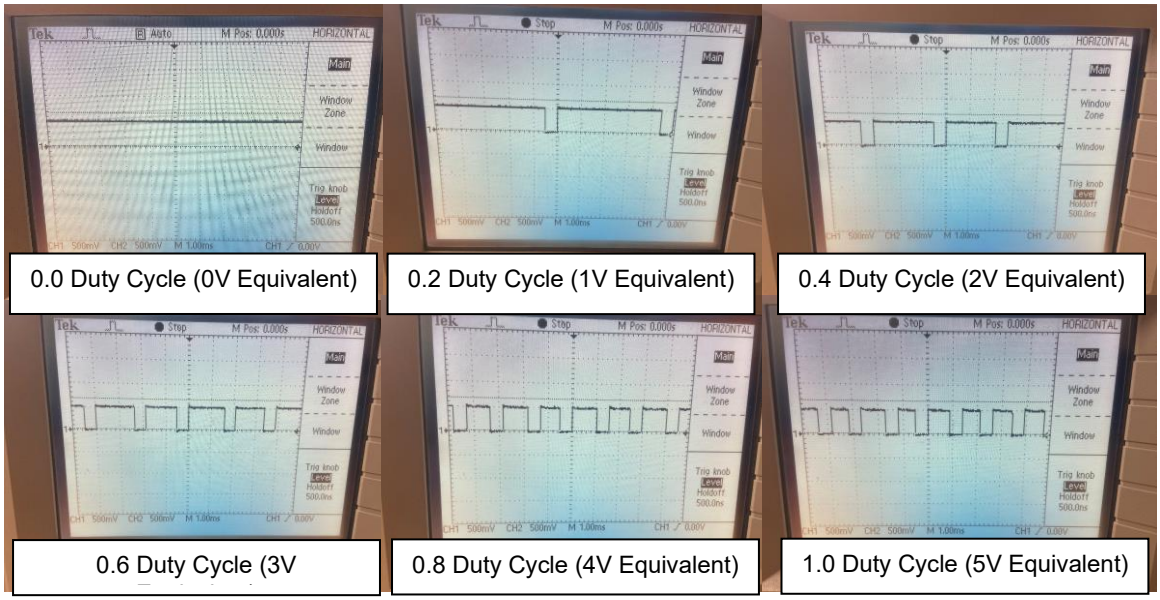


Figure 54: Results for how the tachometer output varies with the Arduino's PWM duty cycle.

Final Powertrain Circuit

Following the success of the previous tests, a circuit was produced according to the circuit diagram shown in *Figure 20*. A photo of the circuit can be seen in *Figure 55*. The circuits were designed to be modular, so they could be switched out if needed. The amplifier circuit incorporated an LM324 multi-op-amp (of which one amplifier was used) and two 4.7 kΩ resistors. Each potential divider incorporated an 18 kΩ and a 4.7 kΩ resistor. The resistors used were different from the original circuit diagram due to availability. However, the ratios of resistance, the most important factors for both circuits, were kept the same.

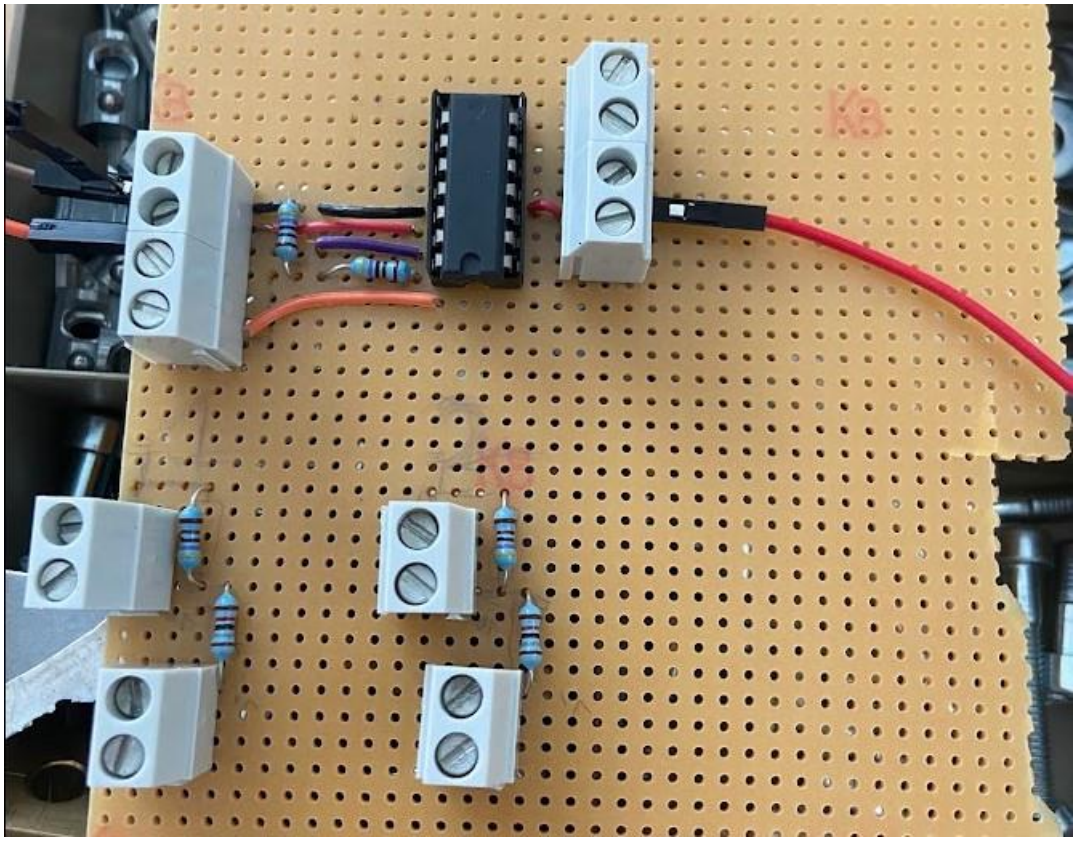


Figure 55: Photo of the non-inverting amplifier (top) and potential divider circuits (bottom).

3.3.10 ROS Development (OB, ZM)

Completed by Oliver Brunnock, Zal Motafram.

Running ROS 2 with a Virtual Machine

A virtual machine (VM) is a digital replica of a physical computer. Multiple virtual machines with their own operating systems can run on the same host [65]. Originally, the Raspberry Pi 5 was the intended ECU; however, this meant that ROS, specifically ROS2, had to be built onto the operating system, from source, due to the Pi 5 supporting Ubuntu 24+ and ROS being compatible with anything before Ubuntu 22. The VM was a solution to test ROS on the Ubuntu 24.04 operating system, to test if it was forward compatible by building it from source on the OS. Initially, a virtual machine was used to run ROS Humble and launch Rviz, as shown in *Figure 56*. This approach provided a way to create a computer environment that could be uploaded directly onto the Raspberry Pi. However, after further testing, this method was found to be prone to errors as a result of differing forward and backward compatibility. Consequently, a more suitable approach using Docker was adopted.

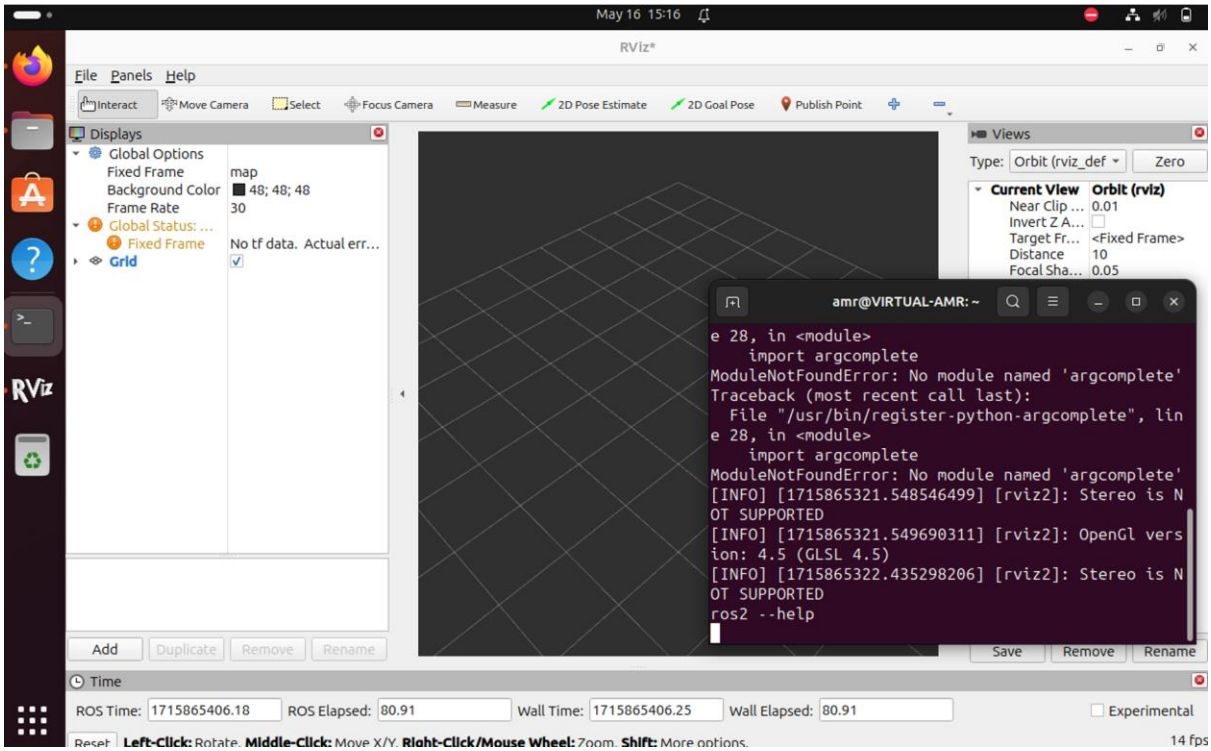


Figure 56: Virtual Machine – Rviz.

Running ROS 2 Nodes in Docker

With further development, it was clear that using a Docker would be more appropriate for running ROS2 Foxy on the robot than a virtual machine. Docker is an open-source platform utilised by developers to encapsulate software into standardised units known as containers. Each container includes the application code along with its environment, encompassing libraries, system tools and runtime [65].

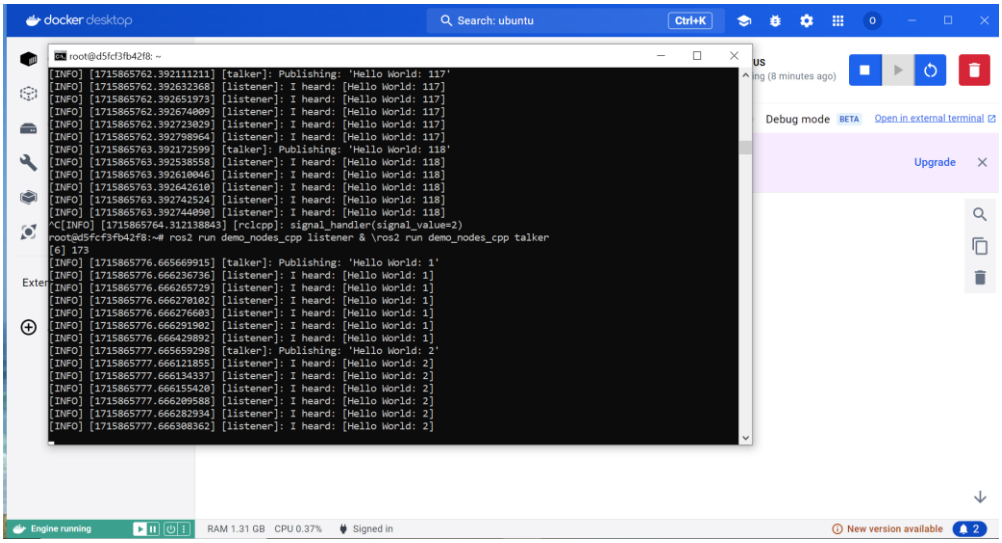


Figure 57: Docker running a node, with a publisher and subscriber

While the docker was successful in creating a node, as shown with the publisher and subscriber in *Figure* 57, it lacked the output of a graphical interface, which is essential for client needs and the UIE subsystem. However, the use of a docker will still prove useful for purely running commands and creating nodes on external computers that are connected to the same

network. Therefore, while still utilising the docker for potential future programming needs, a graphical solution would be more beneficial to the user [66] [67].

Building ROS on Raspberry Pi 4B on Ubuntu 22.04 desktop

By using an older Raspberry Pi, specifically the 4B, compatibility issues with building ROS were resolved. This was done by flashing an SD card with the Ubuntu 22.04 desktop and loading it on the Pi 4B. Following the process of building it from source, as laid out on the ROS development website, this allowed the Pi to successfully run ROS while simultaneously providing a graphical interface for the system. However, there was limited success in building the necessary tools required for the Navigation2 system, particularly Rviz 2 and Gazebo. These tools are essential for the programming interface as their absence in ROS would lead to programming difficulty and a great reduction in the simplicity of the interface [68].

Building Ubuntu 20.04 LTS Server and ROS on Raspberry Pi 4B

As using the Ubuntu desktop proved a limited success, the SD card was flashed again with an older version of Ubuntu; however, this was an Ubuntu LTS Server, which loads a system that is purely a terminal window to process commands. While formatting the SD card with Ubuntu 20.04, certain parameters had to be pre-set to make sure the server would have a name, a username, a password and be able to successfully establish an internet connection upon booting. This formatting proved successful [69].

Once the Pi 4B was booted up, the process followed on the ROS development website was followed and successfully built ROS from the Debian package. The server was able to run a node, with the publisher and subscriber test script running. Despite all the required packages being downloaded and ROS working, this version of Ubuntu still ran as a server terminal window and lacked a graphical output. Thus, a light Ubuntu desktop, “xubuntu” with essential applications such as the Firefox web browser, was built onto the server, using the steps from the Ubuntu website. Once built, the Pi 4B was rebooted and the server automatically launched the desktop with ROS pre-built into the system. A graphical login screen and desktop was now provided for the user [70] [71].

From the desktop, the Terminal window application could be used to access the server and run ROS. As the graphical output was now available, the essential tools for Navigation2 were able to be downloaded and launched. Therefore, this Pi 4B system was now ready to be programmed using ROS and provided the user a graphical interface when required [72].

3.3.11 User Interface & ROS Network Development (ZM)

Completed by Zal Motafram.

As explored in section *3.3.10 ROS Development*, different operating systems were tested, which included testing their interfaces. An Ubuntu OS was essential for operating ROS and as discussed in the SoTA for the user interface, Ubuntu is the best ROS programming platform for the programmer user. Therefore, the exploration of the different operating systems provided a useful insight into the methods of programming the AMR.

LCD Touch Display

The LCD Touch display is an essential extension that adds onto the Raspberry Pi and is mounted directly to it, as shown in *Figure 58*. Originally, a 7” touch display was ordered and designed to work in fusion with the Raspberry Pi 5; however, due to order lead time delays, a 5” touch display was used in its stead for testing purposes.

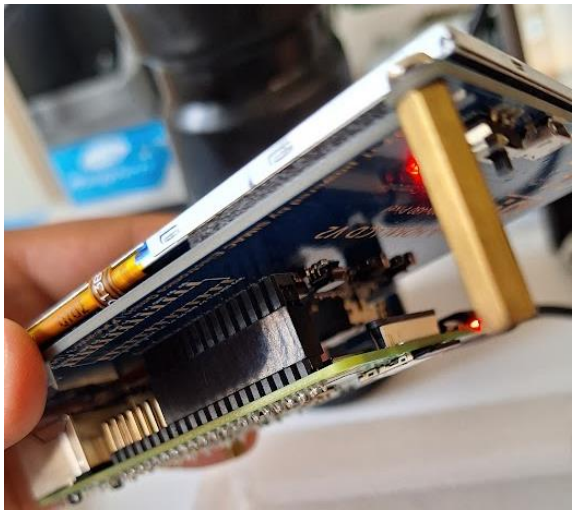


Figure 58: 5" LCD touch display mounted directly to Raspberry Pi 4B, for testing.

The touch display allows the user to have a visual interface on the AMR and see what tasks the Raspberry Pi is processing. Along with a suitable operating system, in this case Ubuntu 20.04, it allows all users to interact with it. As Ubuntu 20.04 is an old operating system and was released before the era of touchscreen display, the *"config.txt"* booting file on the Raspberry Pi SD card had to be modified remotely on a personal computer, by appending the lines highlighted in *Figure 59.*

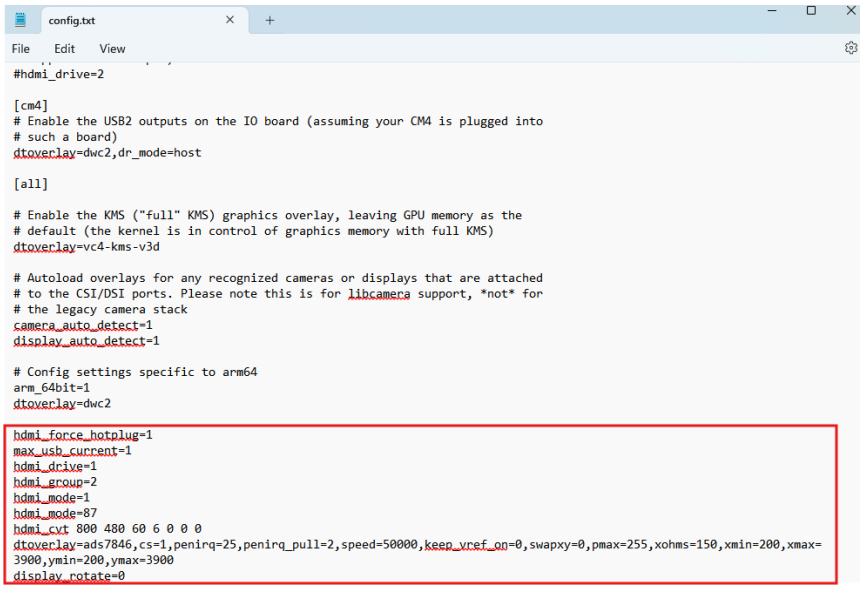


Figure 59: Lines appended to config.txt boot file, to enable touchscreen, highlighted in red.

Furthermore, the touchscreen feature of the display allows the user to access the interface of the Raspberry Pi without the need of a keyboard and mouse. This is important, as it gives the user more freedom and flexibility for usability and simplifies the approach when creating a command interface. However, a keyboard and mouse are still essential for use by the programmer and workshop technician, to modify the Raspberry Pi and ROS scripts faster than is they were to use a touchscreen. Hence, a wireless keyboard and mouse, connected via Bluetooth on a single USB tethering device was used.

Ubuntu 20.04 Interface

Ubuntu 20.04 LTS Server is used, which is a light-weight operating system that focuses on command line processes. As was previously demonstrated, a light-weight graphical desktop environment was built onto the server and functions normally as a Linux operating system. As discussed in the SoTA, this OS utilises the gestalt rules effectively through similarity and proximity but also provides the user with plenty of freedom and flexibility for its usability. Hence, this interface is ideal for both programming user and technician, and the light-weight environment gives the user creative control in setting it up on the Raspberry Pi. However, an effective command interface for standard unspecialised users is yet to be built and must create a programming environment that is simpler than the command line interface.

The simultaneous operation of the command line terminal and graphical interface is demonstrated through the use of applications such as Rviz and Gazebo. Rviz is used to create and show the output of node, such as the Lidar map in *Figure 48*. Gazebo is used to run simulations of the navigation algorithms, as demonstrated using the test commands *"ros2 topic pub /demo/cmd_demo geometry_msgs/Twist '{linear: {x: 1.0}}' -1", Figure 60.* Both applications run via the command line and can run simultaneously, to process information at the same time.

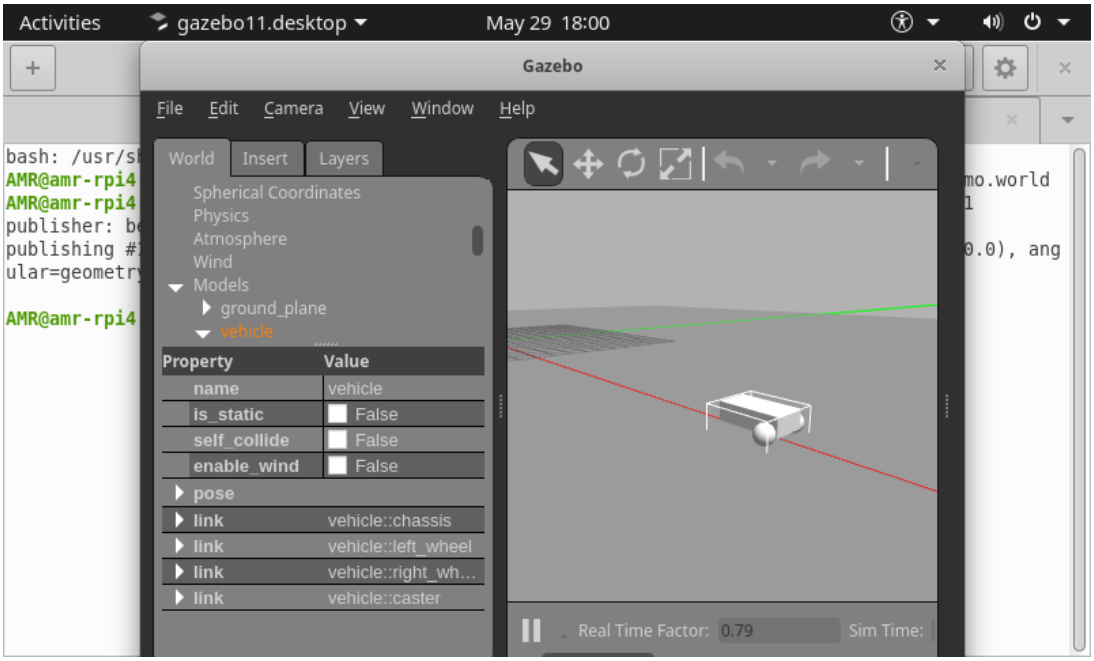


Figure 60: Gazebo running test script of vehicle moving forward.

Local Area Network (LAN) setup on host computer

The Raspberry Pi 4B and Pi 5 have a processor speed of 1.8GHz [73] and 2.4GHz [74], respectively, they have a limited processing speed and will struggle to process both navigation algorithms and sensory input at the same time. Therefore, so that the Raspberry Pi does not have to process tasks, sensor inputs and navigation algorithms simultaneously and "bottleneck" the processor, an external host computer can do these tasks for the Raspberry Pi by processing sensory information and sending back commands [75].

During testing, the Pi 4B was used and this was achieved by making sure both the Pi and the host computer were both connected to the same LAN Wi-Fi connection. Conveniently, the S.mart workshop already had such a computer available, with ROS Foxy distribution and Navigation2 packages preloaded and a secure Wi-Fi network.

As the Raspberry Pi was running on the LTS Server image of Ubuntu, network manager was unavailable and will have to connect using the command line interface. To make sure the Pi was able to connect to a different Wi-Fi, the NetPlan file *"50-cloud-init.yaml"* was edited with the Wi-Fi networks credentials, using the steps highlighted on this Linux community article and made sure the Dynamic Host Configuration Protocol 4 (DHCP4) was enabled on both the Pi and the host computer [76]. The host computer was running a desktop image of Ubuntu, and the Wi-Fi protocol was changed easily by using the network manager. This is demonstrated in the following architecture diagram *Figure 61*, highlighting the communication between the Raspberry Pi and the host computer.

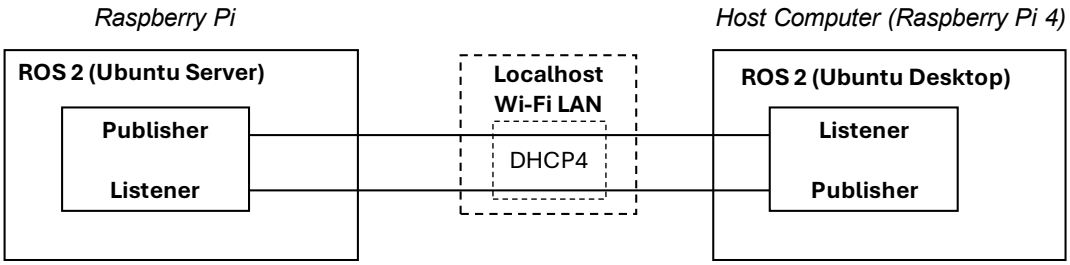


Figure 61: Communication between Host and Raspberry Pi.

Once this was achieved, the DCHP4 being enabled ipso facto allows both devices to communicate with each other over the LAN. Therefore, a publisher node was successfully opened on the Raspberry Pi, by running the C++ test script in the command line and by running the Python subscriber test script, to make sure the ROS framework functions across the network and through the two high-level programming languages. The test was successful and the vice-versa of switching the computer of the publisher node was also successful *Figure 62*.

Figure 62: ROS test publisher node and subscriber command scripts functioning successfully over LAN.

Utilising Nodes and LAN DHCP

As the LAN connection was established and the DHCP successfully tested, this proves that multiple nodes can be opened to process sensory information and send commands between the two computers, via the nodes. Therefore, to test this using the LiDAR and 3D Camera Vision sensor, nodes were opened on the Raspberry Pi to publish the sensory information and Rviz2 was opened on the host computer to subscribe to these opened nodes. To verify the host computer has these nodes available to subscribe to, the command *"ros2 node list"* can

be used to list the opened nodes on the terminal window. Therefore, if the nodes are listed, these can be added as topics into Rviz2 as demonstrated successfully in *Figure 63.*

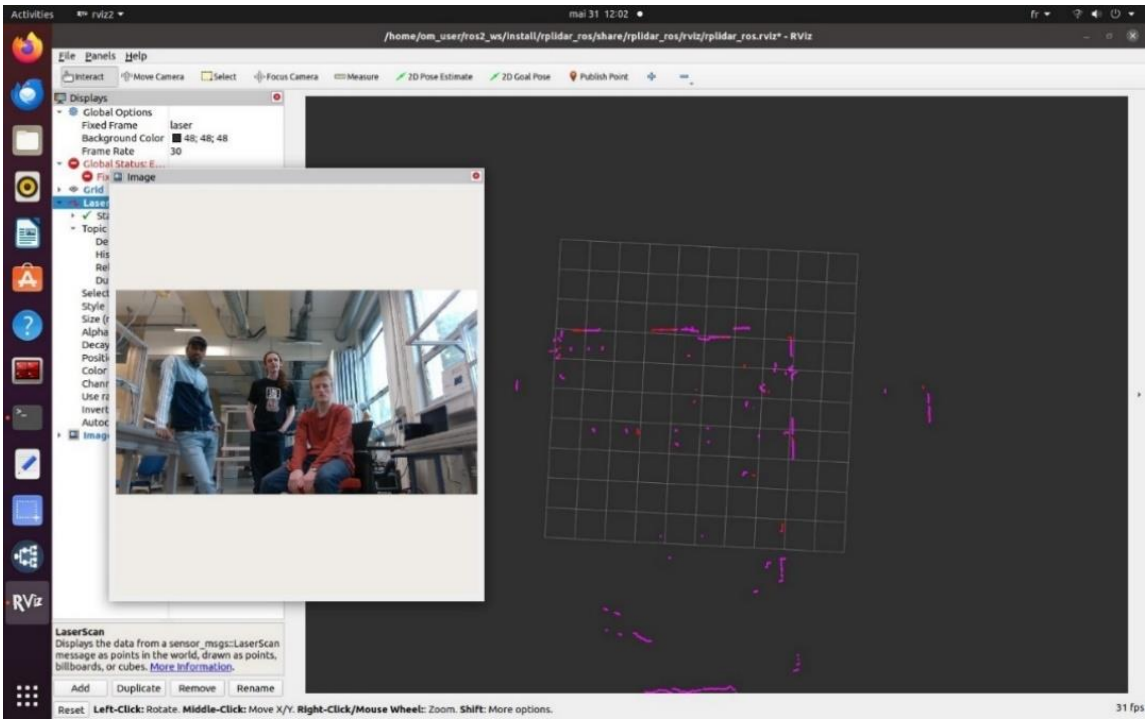


Figure 63: Rviz2 subscribing to both /rp-lidar/ and /color/image_raw nodes.

The test was successful and as seen, multiple nodes can be opened to send across sensory information. The LiDAR has almost instantaneous updates across the LAN connection and the information can be processed to map out the environment. However, the 3D camera has a lot of information due to the resolution and framerate; hence, when the node is subscribed, the resolution of the image is high, but this creates a bottleneck and the framerate that is received is severely slowed down to roughly 1 frame every 30-60 seconds. This bottleneck also causes the LiDAR node's performance to slow down significantly [75].

This is an issue experienced on the LAN as there is no lag experienced when opening the node via Rviz2 on the Raspberry Pi, but only on the host computer. A solution was to turn off the GLSL rendering on the RealSense application accessed on the Raspberry Pi; however, there is still a significant lag [77]. Therefore, the better solution to this problem would be to compress this information when opening the node into the LAN and uncompressing it in post processing, after the node is subscribed to by the host. This solution was not explored due to the project's time constraint; however, it can be pursued by the programming user to resolve it.

Overall, by achieving this, a ROS network has been created and can be utilised by all three intended users to send commands, process information, map the environment and begin to programme missions. However, as discussed in the SoTA for the user interface, a simpler command interface and a scratch-based mission editor is yet to be created. Regardless, this is the ideal environment for a programming user interface and can be fully exploited by the programmer and workshop technician users.

Docker Interface on Personal Computer

The docker method used was effective and as stated previously, it could be used by the programmer to open nodes and send commands. This will allow the programmer to run processes either on their personal computer or on a host computer, without the requirement of a graphical interface. A graphical interface requires a lot more processing power than a terminal interface; therefore, any processes or nodes running on the docker will be faster and will allow the AMR to simply send sensor feedback and run commands send by the host.

Ubuntu OS on Virtual Machine

The Virtual Machine would serve a good testing platform for the programmer. Ubuntu 24.04 was used on the VM test, but ROS did not build properly; however, if the virtual machine had the 20.04 server that was used on the Raspberry Pi 4B, it could be used to build and simulate ROS successfully in the same environment as the Raspberry Pi. The VM could also be used to run ROS on a Personal Computer and with a network connection on the same LAN, it could be used to open nodes and send commands to the AMR.

3.3.12 Controlling Motors with Raspberry Pi (JB, TC, ZM)

Completed by Joe Bailey, Tom Coleman, Zal Motaframm.

After the powertrain circuits were tested with the Arduino and a laptop, a script was generated with a preprogrammed movement routine to test the AMRs ability to move around. After this, we needed a way to take live instructions and transmit them to the Arduino without reuploading the code. To do this, the serial monitor was utilised in the Arduino IDE, which allowed the input of characters. The forward, backward, left, right movements were assigned to the keyboard arrow keys (*Figure 64*) to allow for intuitive control of the AMR, and Arduino code was written to facilitate this as shown in *Appendix D: Arduino Code for Controlling Motors from Raspberry Pi*.

Figure 64: AZERTY Keyboard Command Layout. Adapted from [78]

The Arduino IDE was subsequently installed onto the Raspberry Pi, and when used with a wireless keyboard, removed the need for the trailing ethernet cable connected to a laptop. This solution proved that the motors could be controlled simply from the onboard Raspberry Pi which is required for autonomous navigation.

3.3.13 CAD Models & Production (JB, OB, TC, ZM, JOS)

Completed by Joe Bailey, Oliver Brunnock, Tom Coleman, Zal Motafram, Jenny Öborn Sandström.

OnShape was used to model the CAD for this project as it is open source, allows collaborative working, and can be accessed via the cloud. It is also the University's CAD software of choice.

It can be accessed via this link.

The CAD design as shown in *Figure 65* was originally created based on the dimensions of the two storage containers and the height requirement of 700mm. This height makes sure the user does not need to reach, using their back or knees, to load/unload the AMR and ensures user comfort in this situation.

Using the frame features for the 2020 aluminium extrusion and standard brackets, a frame was created around the boxes. The components were imported into the model to assist with the layout and placement with the motors constraining the minimum width of the robot due to their size.

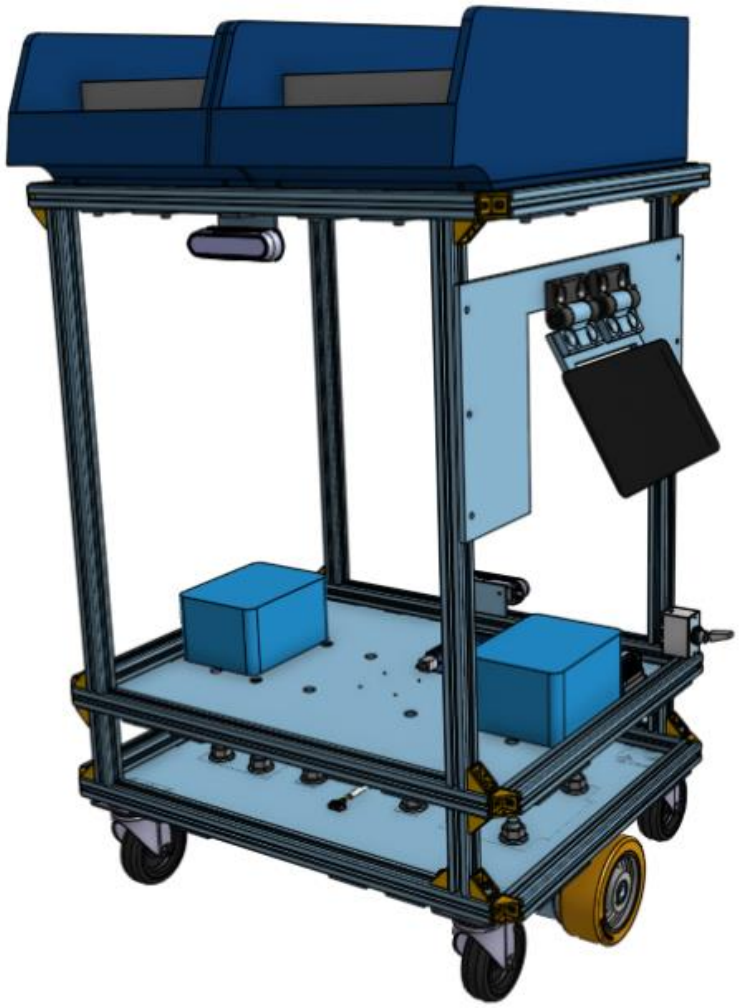


Figure 65: Initial 20x20mm Profile CAD Design

Following this initial design, discussions were had with lab technicians to look at the feasibility of this design. Due to low stock of the 20x20mm extrusion, the possibility of ordering more extrusion was considered, but due to long delivery times for the short-time scale project, the

design was adapted to suit the heavier 45x45mm aluminium extrusion, which was already in stock, eliminating the wait times.

Additionally, the component masses were added to the CAD model including the two 20kg storage boxes, to assess the centre of mass, which was calculated to be around 550mm above the ground. Because of this, the frame was made bigger from 400x490mm to 560x560mm. The final CAD model is shown in *Figure 66* with the actual frame built shown in *Figure* 67. The similarity between the two images highlights how critical the CAD was in building the AMR, allowing virtual fit checks and revisions in design to be carried out before material was cut, reducing wasted material and time spent.

It should be noted that the CAD model includes supplementary components such as the prototype 3D printed drive wheels which will be swapped out for ordered parts once they have been ordered.

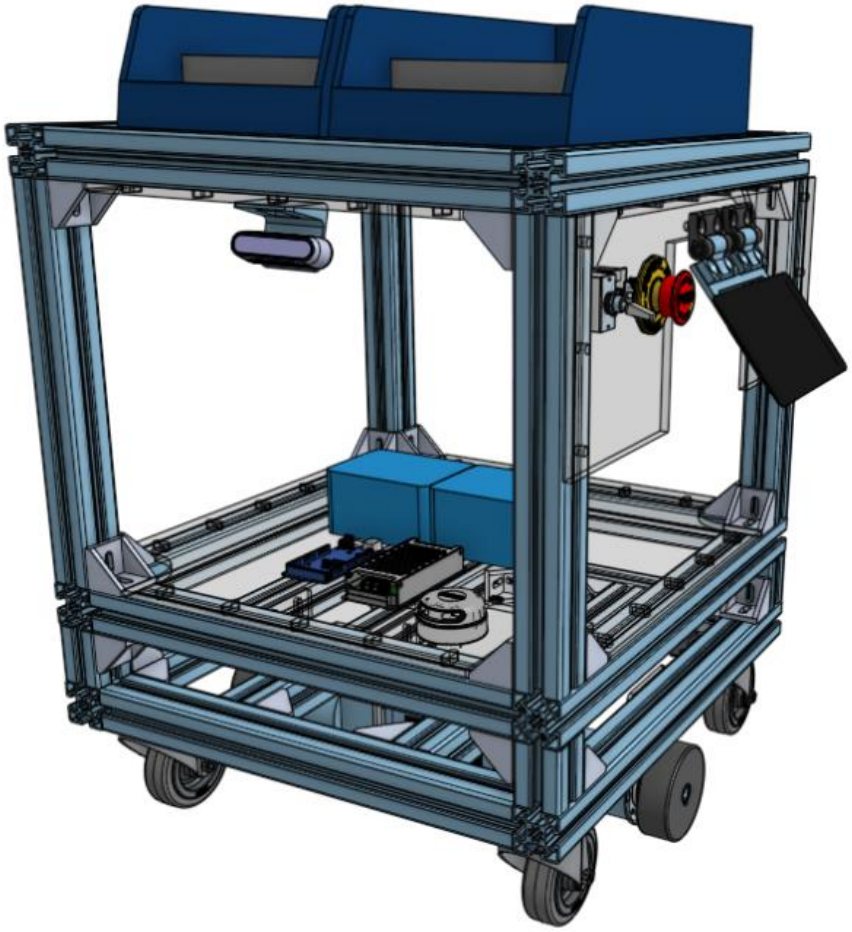


Figure 66: Final CAD Design

Figure 67: AMR Prototype V0

3.3.13.1 Drive Wheel Development (JB)

Completed by Joe Bailey.

The original drive wheel selected for this project was an off-the-shelf solution shown in *Figure 68*.

Figure 68: 100mm Drive Wheel [41]

Due to issues with delivery times, 3D printed solutions were developed shown in *Figure 69*, iterating from left to right. Using OnShape to design and generate STL files, initial keyway fit checks were printed to reduce wasted material and time. Once the keyway fit appropriately, the full wheel was printed in grey ABS using the Zortrax M200 Pro Printers. These designs were sliced using the Z-Suite software, *Figure 70*.

Due to the smooth surface of the ABS part, a “tyre” was required to give the robot enough traction on the smooth floor of the workshop. Firstly, a TPU tyre was designed for the Ender 5 S1 3D printer to produce a seamless and custom fit tyre. After issues with the extruder, a bike inner tube was glued to the outside of the wheel to provide traction. As shown, there were issues getting to remove the seam between the wheel and the tyre. After this, the final solution was created by gluing wide rubber bands to around the wheel to create a seamless surface shown in *Figure 71*.

Figure 69: 3D Printed Wheel Prototypes

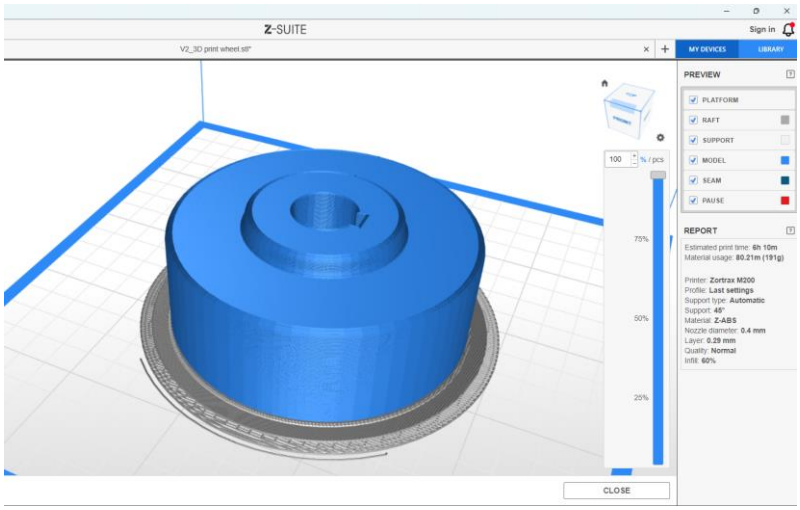


Figure 70: Z-Suite Slicer Software Screenshot

Figure 71: Final Wheel Design

3.3.13.2 Plate Design (JB, OB, TC, ZM, JOS)

Completed by Joe Bailey, Oliver Brunnock, Tom Coleman, Zal Motafram, Jenny Öborn Sandström.

The plates were originally designed to be made from 5mm sheet aluminium to give the robot added strength, and to ensure that the motors could be mounted directly to the plate as shown in *Figure 72*.

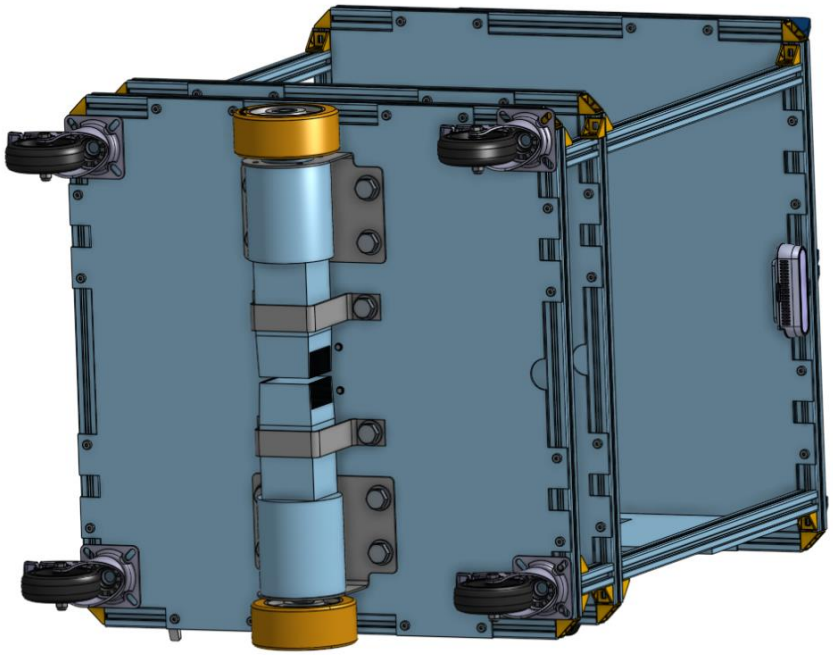


Figure 72: Initial Motor Mounting Metal Plate Design

After discussions with workshop supervisors, it was determined that the facilities were not available to produce aluminium plates on site, and to reduce time and cost the design was reconsidered. Since the laser cutter on site seemed the most appropriate machine for cutting sheets of material, the plate was instead redesigned and laser-cut from an 8 mm thick acrylic sheet, as shown highlighted in yellow in *Figure 73*.

The acrylic plates did not have the strength to mount the motors to securely, and so the bottom frame was redesigned to incorporate horizontal bars that the motors could be attached to as shown in *Figure 73*.

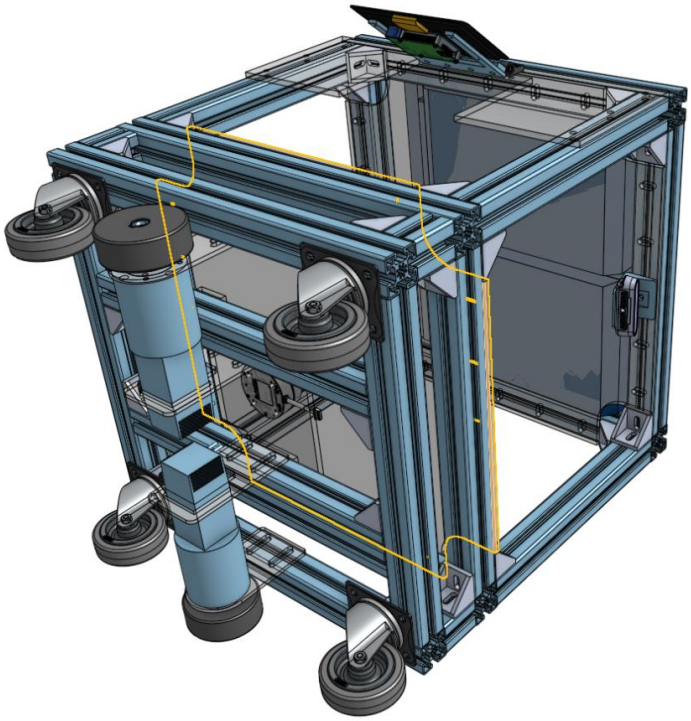


Figure 73: Final Motor Mounting and Plate Design

3.3.13.3 Motor Mounting Brackets (JB, ZM, TC)

Completed by Joe Bailey, Zal Motafram, Tom Coleman.

The components intended to fix the motors to the frame went through significant changes over the course of the AMR's design and implementation.

The first version of the bracket was designed as a box that would fully enclose the motor. The face of the motor could be mounted to the bracket via the screw holes, and the body of the bracket would stabilise the body of the motor. Four bolts would secure the bracket to the frame. The bracket was planned to be constructed out of 5mm thick aluminium sheets. *Figure 74* shows the CAD model for this bracket.

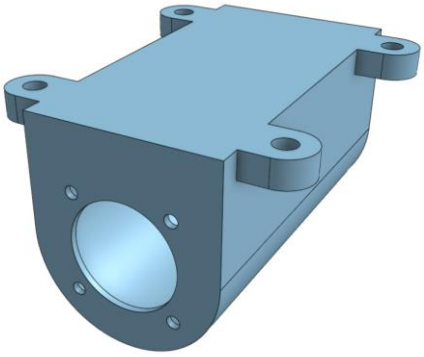


Figure 74: Initial Motor Mounting Bracket.

It was determined that the box bracket would be too challenging to manufacture with the facilities available and would be too heavy. This was fixed by using two brackets: one for mounting the face of the motor to the frame, and the other for holding the weight of the back side of the motor. These were planned to also be cut from 5 mm thick aluminium sheets, and

then bent at right angles to form the full shape. A total of 6 bolts would mount the brackets to the frame. *Figure 75* shows the CAD model for the brackets in use with the motor. *Figure 76* shows the technical drawing for the mounting bracket shown to workshop technicians to discuss the design.

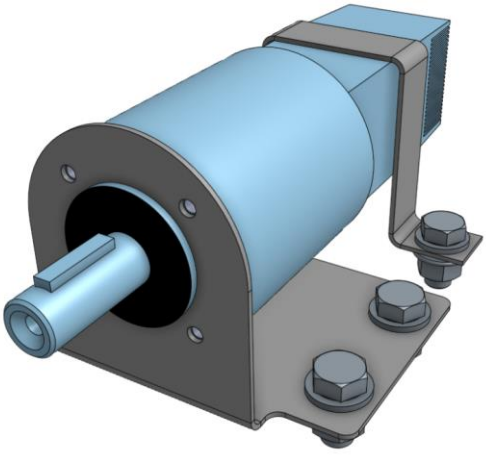


Figure 75: Second Iteration Motor Mounting Bracket.

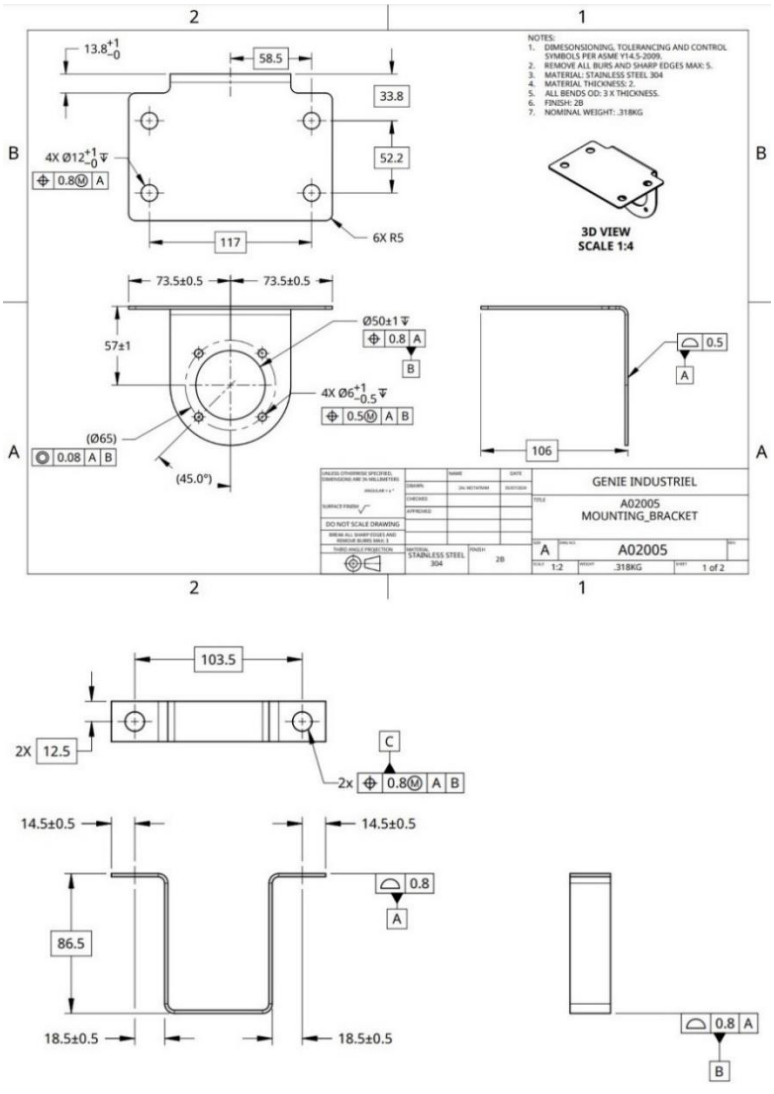


Figure 76: Second Iteration Motor Mounting Bracket Technical Drawing

The previous aluminium or stainless teel brackets were the preferred design for securing the motor, but time constraints prevented them from being manufactured before the end of the project. An alternative set of brackets was designed, with much simpler profiles that could be laser cut and with the ability to adjust the height of the motors relative to the frame. *Figure 77* shows the CAD model of the brackets in use with the motor. Note the slots for mounting to the frame, allowing the height of the motors to be changed to distribute the AMR's weight between the wheels equally. *Figure 78* shows the technical drawing for the main mounting bracket.

These brackets were laser-cut from 8 mm thick acrylic sheets, with the additional thickness intended to compensate for the lower fracture toughness of the material compared to aluminium [79] [80].

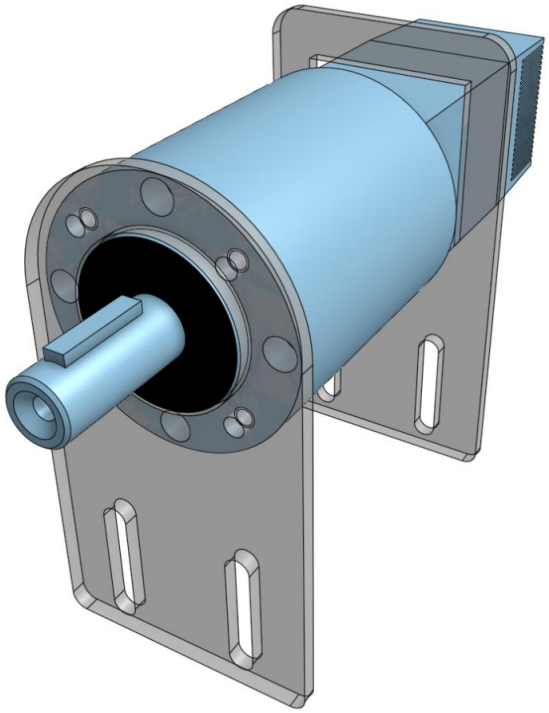


Figure 77: Third Iteration Motor Mounting Bracket.

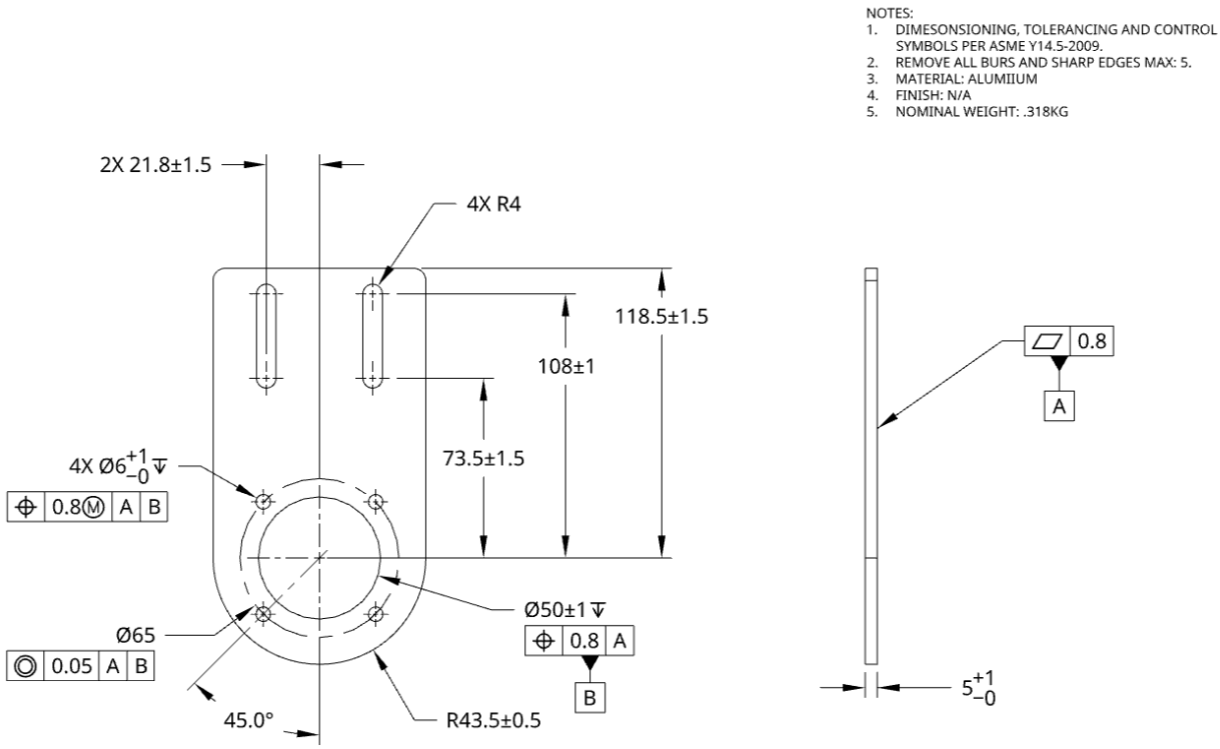


Figure 78: Third Iteration Motor Mounting Bracket Technical Drawing

3.3.13.4 3D Camera Mount (ZM)

Completed by Zal Motaframe.

The 3D camera mounting bracket is designed to mount the Intel RealSense D435i camera using its back mounting holes, according to the camera's technical drawing dimensions [61]. Initially, the camera was going to be mounted on the 20x20mm strut profile frame, with the 5mm grove size and was tested as such in CAD for a virtual fit check and physically after 3D printing in ABS *Figure 79*. Furthermore, the initial mounting location was meant to be on near the bottom of the AMR on the central plate; however, after consulting the client and showing the CAD, the mounting location was moved up higher *Figure 80*.

Figure 79: Physical fit check of camera mount (REV_A) using 20mm strut profile.

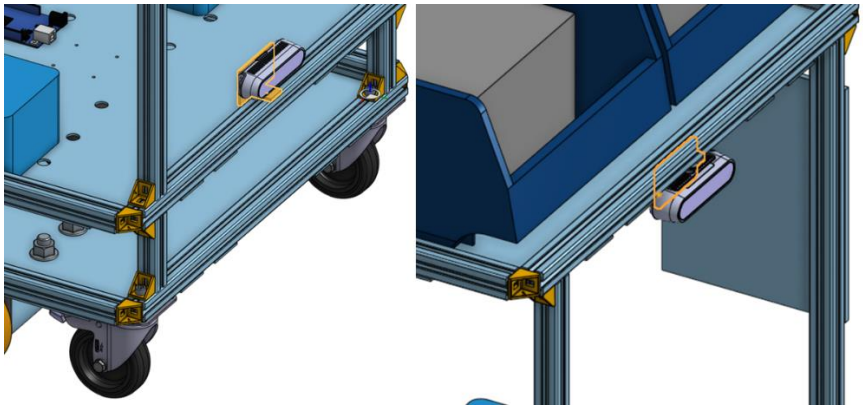


Figure 80: Initial mounting location of camera (left); updated mounting location of camera (right).

Once the strut profile size was changed to 45mm with 8mm grove size, a new revision (Rev B) of the camera mounting bracket was made. However, after both virtual and physical mounting tests, the client mentioned that is would be desirable to see at least the shoulders of any possible human hazards, as a part of its object detection algorithm. Therefore, a further revision (Rev C) was designed to angle the camera up, while still being stable enough to mount onto the struts by creating an angled grove to accommodate the flange nut*, Figure 81, Figure 82*. As this design was stable, it is recommended that the material for the mount should remain ABS and the best way to print this complex shape was by 3D printing shown in *Figure 83*.

Figure 81: REV_B on top; angled iteration, with grove for flange nut REV_C (bottom).

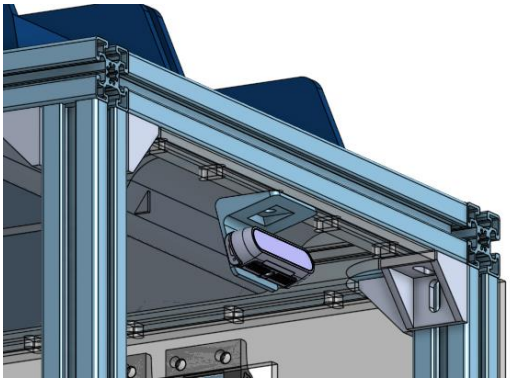


Figure 82: Assembled CAD of REV_C.

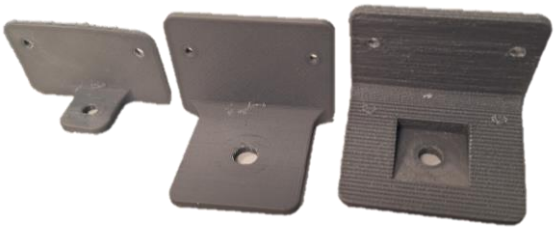


Figure 83: Printed evolution of camera mount.

3.3.13.5 LiDAR Mounting (ZM)

Completed by Zal Motafram.

The LiDAR has mounting holes specified on its underside and the dimensions are specified in its datasheet [28]. Initially, as there was going to be bottom plate to help with motor mounting, the LiDAR mounting holes were included in it, as seen in *Figure 84*.

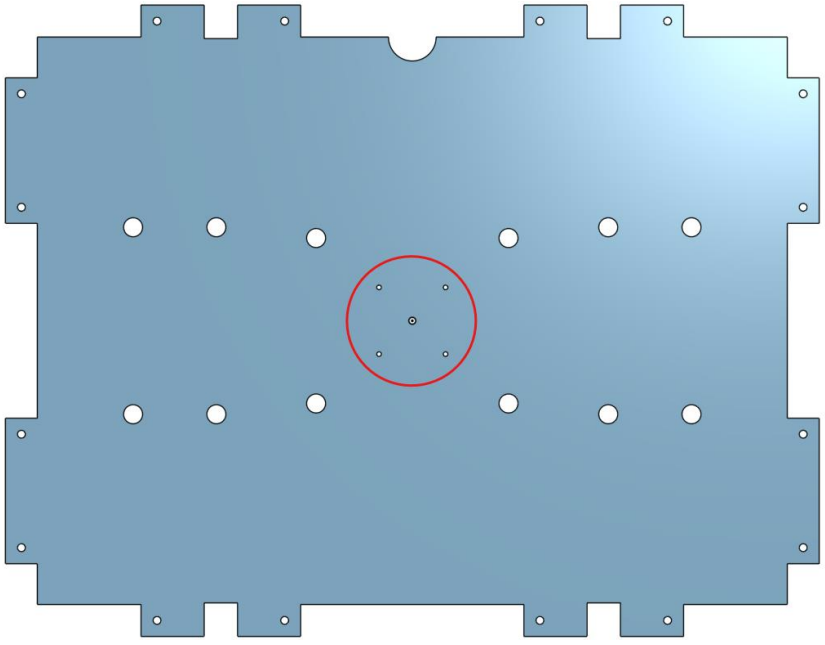


Figure 84: LiDAR mounting holes on bottom plate, highlighted in red circle.

However, as the motor mounting method changes, a separate mounting panel for the LiDAR was designed. This also accommodates the bracket for the motor mounting, to make sure they sit flush on the surface of the profile struts as show in *Figure 85* and *Figure 47*.

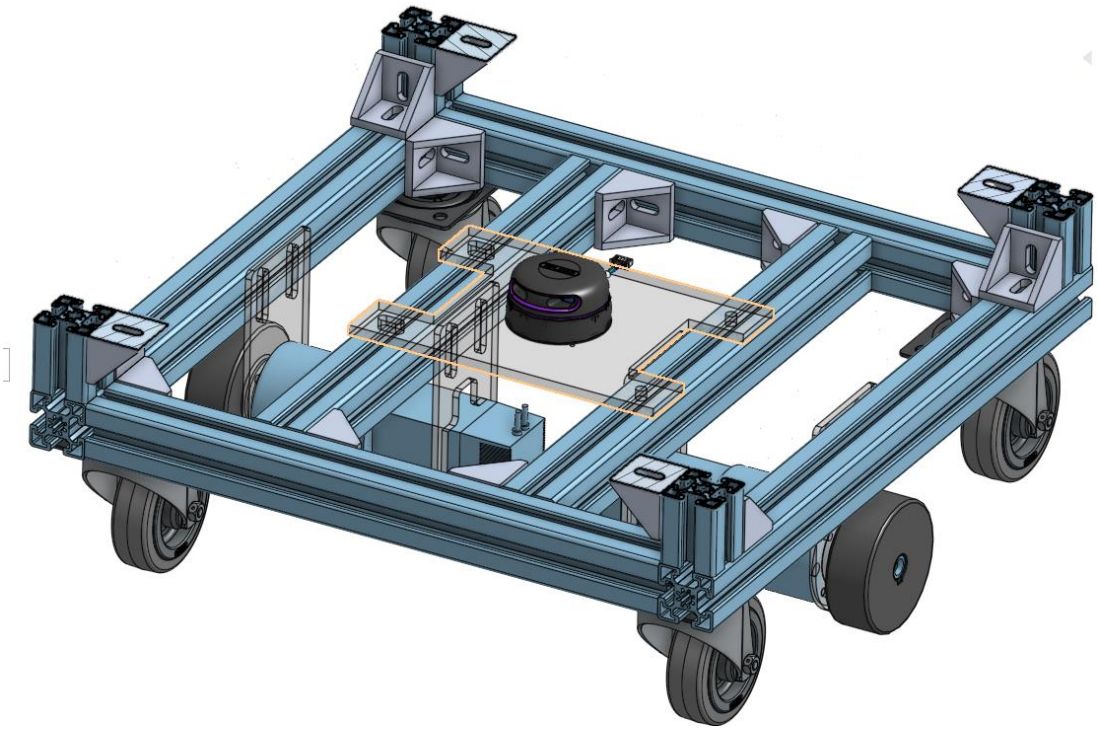


Figure 85: Sectioned assemble view, showing LiDAR mounting panel highlighted in an orange outline.

3.4 System Verification

After each system is integrated and tested, the next step is to verify and validate the system. This is the point where the project is handed over. This guideline outlines the next steps, ensuring that the Automated Mobile Robot (AMR) system requirements are clear, complete, and testable.

The initial requirements need to be verified, for instance, we need to ensure that the AMR can effectively detect and avoid obstacles and navigate to set waypoints. Additionally, the robot should be capable of operating with a full load of two 20kg boxes for a minimum of two hours, and its maximum speed and acceleration should be 2 m/s and 0.5 m/s², respectively.

Next, we validate that the software meets the specified requirements. This includes ensuring that the SLAM algorithm is correctly tuned for the workshop environment and the robot's size. Integration testing is then conducted to verify the interaction between integrated components, ensuring that all parts of the system work together seamlessly and identifying any issues that arise when components are combined.

Various safety and security tests are also conducted to ensure the system's integrity. For example, we need to ensure that images taken by the vision sensors are not accessible to unauthorised users, thus protecting the system's security. This phase allows for the early detection of defects, enabling the team to refine the design and avoid further costs. Ultimately, it ensures that the final product meets all specified requirements and performs as expected, so that the AMR system is reliable, efficient, and ready for deployment.

3.5 Operation & Maintenance

Once the AMR has passed the verification steps, it will be deployed in the workshop, the environment where it will be operational. Users will receive training to ensure they are familiar with how to operate the robot. The training will be tailored to different user groups, including general lab users, technicians, and students.

Monitoring the robot's performance is necessary to ensure it operates correctly. This involves tracking data from the SLAM algorithms and gathering user feedback. Any defects that arise will be addressed through maintenance before the robot is allowed back in use, which includes implementing necessary software fixes. Given that the workshop is a dynamic environment, the AMR may need to adjust to new changes such as additional chairs and tables. This adaptability ensures that the AMR remains functional and relevant over time, reducing the risk of unexpected failures. Continuous improvements will enhance user satisfaction and help avoid major costly repairs.

The addition of Failure Modes and Effects Analysis (FMEA) into the project can significantly enhance the reliability and safety of the AMR. FMEA could enable the team to identify potential failure modes within the AMR's subsystems and their resulting effects on the system's performance. This approach helps in prioritising which issues to address based on their severity and likelihood, and also facilitates the development of strategies to mitigate risks. Ultimately, FMEA can lead to a more robust and dependable AMR design, ensuring that it performs safely and effectively in dynamic environments [81].

4 Discussion

4.1 Comparison to Existing Technologies

The university currently owns two Sherpa B robots, capable of autonomously navigating the workshop on the S.mart platform. However, the development of this new AMR was necessary because the Sherpa B and its competitors are not open source. This limitation prevents the installation of new navigation and obstacle avoidance systems for research purposes. The prototype created in this project is completely open source, utilising ROS2 installed on the Raspberry Pi's Ubuntu operating system. This setup provides PhD researcher Quentin Levant with full access to the system, a feature not available with the Sherpa B robots; hence, it gives the programming user more freedom and flexibility. By using common components like the Raspberry Pi and Arduino, the project ensures extensive support from the coding community through videos, literature, and software. This makes the system ideal for researchers like Quentin, who are already familiar with these ecosystems.

The robot has been designed to be as cost-effective as possible, balancing affordability with functionality. Key components, such as the LiDAR, were carefully selected in consultation with the user, Quentin Levant, during the ordering process. The final cost to the university was €2,987, significantly less than any commercial solution. In contrast, the Sherpa B robot uses high-end components, including three LiDARs costing a total of €11,100, which is more than 3.5 times the cost of the AMR developed in this project. The RPLiDAR A3, chosen for this prototype, costs €542, highlighting the cost efficiency achieved without compromising essential functionality [82] [83] [84].

The robot is designed with a highly modular structure based on 45x45mm aluminium extrusions. Due to the use of brackets to fasten these together, the dimensions of the robot can be easily changed by unbolting the existing lengths, and sliding in different sections, giving the AMR a great deal of flexibility for different purposes and payloads. Additionally, with the large headroom in the 24V and 5V supplies onboard and additional space within the frame, additional equipment can easily be added onto the AMR for research. Commercial robots such as the Sherpa B cannot easily accommodate new equipment with a lack of mounting points and space inside to facilitate the installation of additional equipment within, making them less useful for these research purposes. However, the commercial solutions are much more visually aesthetic compared to the blockier and simpler geometry of the robot developed here.

4.2 Theoretical Application

SLAM was chosen as the navigation system for our autonomous robot. The type of SLAM selected is LiDAR based due to its superior precision and suitability for high-speed applications such as a dynamic workshop. In this case, a 2D LiDAR was selected for its effectiveness with mapping and navigating a workshop.

The robot was designed to handle high-energy SLAM algorithms which consistently update the map environment and track its surroundings. To manage the required processing power, a Raspberry Pi was selected to be connected via UART to an Arduino. The Arduino handles multiple inputs and outputs from the motors and infrared sensors. The RPLiDAR A3 which was chosen as the main sensor for navigation, offers a 25-metre range and a high scan rate of 5-20Hz covering the whole workshop. This setup should provide accurate and real-time mapping capabilities. The combination of the Raspberry Pi and Arduino should handle the processing demands and sensor integrations and the LiDAR should ensure quick updates and precise object detection. Once the robot is fully integrated and ready to run the SLAM algorithm, performance data will be collected, and these tests will provide information into how the algorithm can be tuned if needed [28] [85].

During testing, there may be some discrepancies between simulating the robot and the actual robot. For example, there might be blind spots not covered by some of the sensors. This would require the addition of more sensors or the repositioning of existing ones. Additionally, if the processing components such as the Raspberry Pi overheat during prolonged use, adding a heatsink or other cooling solutions may be necessary. Sensor limitations and software errors are other potential sources of discrepancies. For example, the LiDAR may be affected by environmental factors such as dust or reflective surfaces. It is important to identify these errors early on in initial testing.

Overall, while the theoretical application of SLAM in our AMRs navigation system promises high precision and efficiency, practical testing is important to address any limitations. This process ensures that the final implementation meets the high standards required for the dynamic workshop environment.

4.3 System Architecture & Integration

After the project requirements were realised, the system architecture diagram was developed to understand what would be involved in meeting the key requirements in *Figure 6*. From this the project was split into 5 subsystems for the 5 people involved, namely: ECU & Navigation System, Powertrain Management, Electrical Power Management System, Frame & Carrying System, and UI & Ergonomics.

The separation of subsystems was also based on the functional decomposition *Figure 5*. The diagram showed distinct sets of functions that could be grouped together. For example, supplying power, emergency stop, and being rechargeable could be grouped into the Electrical Power Management System. Each function was assigned the group that it was most closely connected to.

The splitting of the work into subsystems was done to aid in separating the initial research tasks and work during the project, which was very successful. After the research had been completed and the components had been selected and ordered, a more thorough system block diagram was created *Figure 86* to represent the system. This shows that while the five subsystems remained, the ECU & Navigation and the UI & Ergonomics subsystems became much more linked.

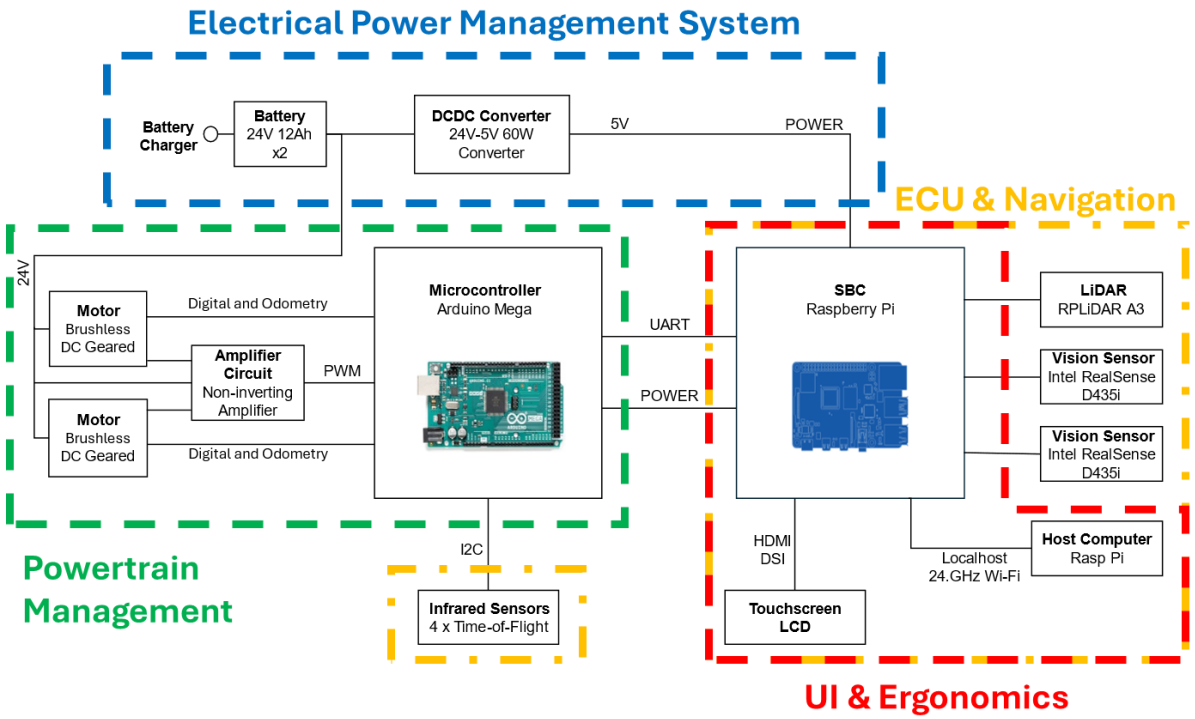


Figure 86: System Block Diagram with Labelled Subsystems

Integrating this architecture around the Raspberry Pi was very successful in terms of modularity and scalability. The connections using the standardised USB made physically connecting the sensors and the Arduino very straightforward, and the AMR benefitted from having only two voltages present, requiring few parts to the Electrical System. The use of a low power single board computer greatly reduces the complexities of the AMR, and the capability of running the computationally intensive tasks remotely and using the Pi as a transmitter of data, is very powerful. For very high-speed navigation, it may be beneficial to have a more powerful computer onboard to run the SLAM algorithms to reduce latency issues, but this was not a requirement for this project.

The distinction between the subsystems allowed the team members to work in parallel, only putting the systems together during the final weeks, which greatly increased the amount of work that could be done. In addition, due to the clear requirements of each subsystem from the block diagram shown in *Figure 86*, the integration of these subsystems was made simpler and took little time to connect them together for a functioning system.

4.4 Achievement of Objectives

Whilst the project was generally a success given the short timeline, creating a functioning robot with all of the necessary hardware operational for autonomous navigation, the success can also be quantified by addressing the completion of the set of 54 rigorous technical requirements formulated early in the project. A pie chart is included in *Figure 87* to provide an overview of the requirements which are either Tested & Completed, Requires Test, Partial Completion, or Incomplete.

The project met definitively met 19 of the 54 objectives set out in the technical requirements, including the utilisation of ROS2, the ability to carry the standardised boxes, and the unloaded system being under 50kg which are all measured successes.

Due to the time constraints, there were 7 objectives that are expected to be fulfilled, but have not been tested yet, such as the $0.5ms^{-2}$ acceleration and the 7km/h top speed.

12 requirements were left partially complete, such as that the system must be able to receive destination commands as so far commands are able to be sent from a host computer to the AMR via the ROS2 framework over a LAN connection, but destination commands are yet to be implemented. This is due to the ROS2 framework, on the Raspberry Pi, not yet being integrated with the Arduino.

Additionally, there were 16 objectives that were left incomplete. Some of these were because of unforeseen issues such as the battery life of 2 hours, due to the battery not being delivered in time. Others were because of the time constraints such as having no burrs or sharp edges. Some requirements would be relevant to later versions of the AMR, when it would be used outside of a controlled testing environment. This includes the addition of hardware and software to produce an emergency stop. If this project is to be picked up again by another group of people, this list of incomplete requirements should form the basis of improvement to the AMR.

Overall, we were able to address 70% of the requirements, 35% of which we were able to test and complete, whereas the remaining 30% still requires fulfilment and implementation into the architecture as shown in *Figure 87*.

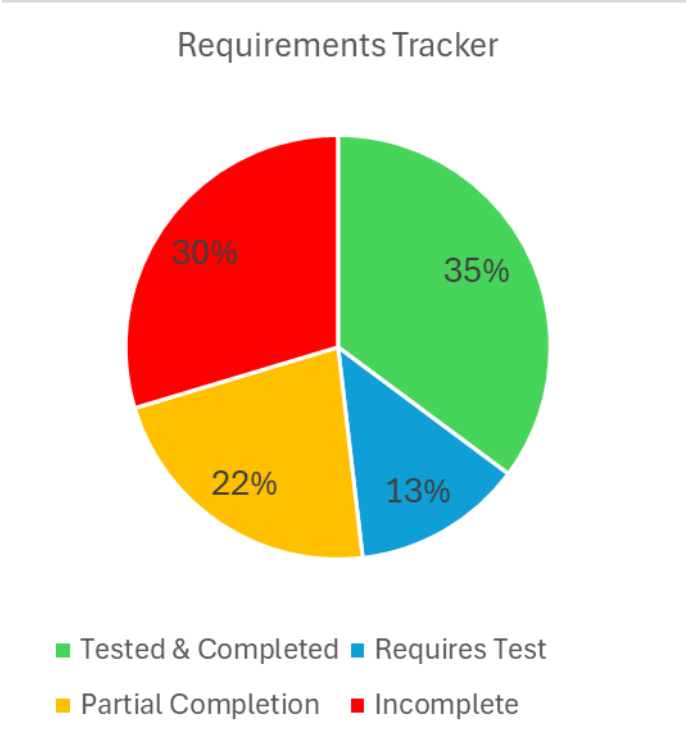


Figure 87: Pie Chart Detailing Technical Requirement Completion

5 Reflection

5.1 Methodological Insights

The V-model for system design shown in *Figure 8* was used during the entirety of this project. Following the left arm of the V (system definition), at the beginning of the project a great deal of time was spent finalising the requirements with our client in a concise and clear list with numerical performance metrics to ensure we could deliver a robot that would perform as expected. This clarity from the start ensured we knew exactly what the robot had to be capable of, which was essential during both the research, where we compared technologies based on how well they met these requirements, and in the design stage, where calculations were carried out to size appropriate components to satisfy these criteria.

An example of this was defining the acceleration as $0.5ms^{-2}$ early during the requirements refining process. Using this, and other requirements such as system weight not exceeding 50kg and the payload of 40kg, the motor sizing calculations were carried out to ensure the robot could perform as expected. This reduced time spent deducing the requirements during the design process, reducing our time and our clients time wasted.

The implementation and the project test & implementation sections were spread over only a few weeks, but due to the solid base of requirements and the extensive design work carried out on the left side of the V, specifically on the CAD, the time taken to assemble and test the systems was greatly reduced.

As mentioned in the discussion, there were still a number of incomplete and untested requirements for the project, which highlights the need to have spent more time on the Verification & Validation section of the V-model. If we had progressed through the other stages faster, we would have had more time to systematically complete all of the requirements.

The methods used (primarily the V-model) were successful in this project and assisted greatly in structuring the project properly and logically, allowing us to focus our attention in the correct areas, and to ensure that we did not make mistakes like ordering unsuitable components. It was more due to the time pressure that there were still unfinished requirements and not the methodology and given more time all of the requirements could be met.

5.2 General Challenges & Successes

Challenges

The biggest challenge faced was the longer than anticipated delivery time of a few critical components such as the battery and Raspberry Pi 5, which did not arrive in time for the end of the project. However, to complete the prototype we overcame these problems by utilising a lab bench power supply in place of the battery, and a Raspberry Pi 4B and an accompanying touch screen as a temporary solution for example. We also adapted our design such as changing the aluminium profiles from 20x20mm to 45x45mm to use what was already in stock to eliminate wait time, and this along with our use of rapid manufacturing techniques such as 3D printing and laser cutting greatly increased the amount we could do in the short amount of time. In hindsight, if we had completed our research a week or two earlier, it may have meant that these parts could have been delivered in time, but perhaps at the expense of more detailed research.

Successes

The CAD model of the design was incredibly useful in reducing the time taken to finalise the prototype, with the virtual fit checks ensuring that we had considered many issues before we cut the material, making the design much more thorough. It also reduced wasted material, with

the only wasted parts being the 3D printed iterations carried out for the wheels and camera mounts, reducing the environmental impact of the project.

The incorporation of the University's own manufacturing facilities into the production of components reduced lead time significantly. It also gave greater control over the dimensions of supporting components like mounting brackets. The low cost, low waste, and low lead time of additive manufacturing methods like 3D printing supported rapid prototyping, which worked to the strengths of the agile V-model which encouraged frequent verification and iteration of designs.

5.3 Team Collaboration & Dynamics

During the project, our team worked 09:00-17:00 in person. These regular hours together were constructive in allowing us to easily talk through any questions we had, and have discussions about next steps in the project. This, along with the separation of the project into subsystems, meant that the work was fairly distributed between the group, and the large amount of teamwork required on this complex project could be carried out through a simpler approach.

In-person collaboration led to great cross-functional communication between the team which resulted in the group not appointing a project manager, since everyone was adept at understanding where the project was, and working together to complete as much as possible, even if it meant moving between subsystems. An example of this was Oliver & Zal working together on the ROS2 integration; Tom, Zal & Joe working in tandem on the motor mounting brackets; or Jenny & Oliver spending time on the report whilst Joe, Tom, & Zal were finalising the last parts of the build.

The speed at which the project developed, especially during the final weeks after critical components such as the motors had been delivered, was very impressive and a result of thoroughly planning the build of the AMR. After completing the research on our subsystems separately for the SoTA report, we worked incredibly well together during the time pressured and fast-paced final weeks to build and test our prototype. Although challenging, especially during the final weeks, we balanced our time well between the report and presentation writing and the project itself.

There were moments throughout the project of disagreement, for example during the component selection stages of the project where making a battery from individual cells was being discussed, which perhaps was not sensible with the short project timeline.

However, by taking an educated approach through sophisticated and academic debate, we constructively criticised each other's decisions to avoid bad choices that would be challenging to fix further down the line. This approach was taken, as it is more challenging, time consuming and expensive to correct design mistakes later on during Product Development than earlier [10]. Once again, the in-person working meant we were always discussing what we were doing throughout the project, so any disagreement was had straight away and dealt with which was positive.

5.4 Personal & Professional Growth

This project allowed the team to learn a variety of new skills. It was incredibly useful having a mixture of backgrounds such as Integrated Mechanical & Electrical Engineering and Design Engineering in the project as this allowed us to learn a great deal from each other.

At the beginning, some of us had very little CAD experience, but due to the nature of the project this was greatly developed to make the complex final model shown in *Figure 66.* This was a critical engineering skill to develop and is a skill that is hard to get better at by watching lectures in isolation.

The introduction of ROS2 for the first time to the group was challenging, but Oliver and Zal spent a great deal of time learning how to incorporate ROS2 into the project from scratch which proved beneficial for the project allowing us to operate the LiDAR, depth cameras for testing purposes natively, and for their own skill development.

We all learned lots of information regarding design processes such as the V-model and the refinement of requirements during this process which is useful for any design project taken in the future. Research skills were also developed greatly during the technical SoTA reports, especially when trying to objectively compare appropriate technologies in an unfamiliar area and carry our research in an area that needs to be learned by yourself.

The various meetings and defences throughout the project have allowed us to improve our presenting skills too, especially with the large amount of feedback we have received from our supervisor Christelle Grandvallet regarding these.

The team has improved greatly in communication with the client, users, supervisors, and each other during the project. Through scheduling weekly meetings and being realistic with timelines we have avoided disappointments, and through discussing ideas with each other throughout we have become much more open to new ideas.

5.5 Subsystem-Specific Reflections:

5.5.1 Electronic Control Unit & Navigation (OB)

Completed by Oliver Brunnock

Challenges

The ECU and Navigation aspects of the project presented several challenges. Firstly, the initial research topic was extensive and contained a significant amount of content. Much time was spent digesting the initial technical requirements from Quentin and Camille and dividing the research into manageable sections: sensors, navigation systems, software architecture, ECUs, and exemplar AMRs. Even within these topics, there was a considerable amount of content, requiring thorough review of scientific papers and robot websites.

The topic was new, and it was learnt from scratch. Understanding navigation methods such as Monte-Carlo Localisation and SLAM was particularly challenging due to their technical complexity. Selecting a suitable LiDAR sensor also proved difficult. The broad range of available sensors meant finding one that met our requirements while remaining cost-effective was a challenge. LiDAR sensors are typically expensive, so it was crucial to balance cost with performance for major components.

We also faced challenges with component deliveries. The Raspberry Pi 5 did not arrive in time, so initial prototyping began on a Raspberry Pi 4B obtained on campus, using a different version of Ubuntu. This meant that ROS 2 would need to be reinstalled once the Raspberry Pi 5 arrived. Additionally, we were only able to test one vision sensor, as the other one did not arrive. It would have been beneficial to have both sensors working together for comprehensive testing. Unfortunately, the infrared sensors also did not arrive, preventing us from conducting tests with them.

Visualising the sensors on the Raspberry Pi 4 was challenging due to the small 5-inch screen available in the workshop. Programming the command line interface for ROS 2 on such a small screen was difficult. The 7-inch screen compatible with the Raspberry Pi 5 will make interfacing easier, especially once mounted onto the AMR.

Successes

Despite the delayed arrival of some components, thorough testing was conducted on the LiDAR and vision sensor. Observing the point cloud from the LiDAR displaying a real-time map of the workshop was particularly rewarding. Additionally, testing the 3D camera and using Python to develop a program that measures distance in real time provided valuable practical experience. Launching this in ROS offered an excellent learning opportunity. The process of running ROS on a virtual machine, then on Docker, and eventually building it from source, provided a comprehensive understanding of setting up ROS. This experience will streamline future setups, making them significantly faster. The distinctions between virtual machines and Docker, along with their respective advantages, were also clarified through this process.

The initial research phase contributed significantly to understanding the science behind LiDARs and 3D cameras. It was insightful to compare different navigation strategies, such as behaviour-based robotics, Monte Carlo Localisation, and SLAM. After careful evaluation, SLAM was selected as the optimal navigation strategy due to its superior precision and real-time mapping capabilities, which are crucial for dynamic workshop environments. The connection between the Raspberry Pi and Arduino via UART worked exceptionally well, providing a strong foundation for implementing SLAM. This successful integration ensured reliable communication and control, which is essential for the robot's overall functionality.

Involvement in the CAD process and contributing to the optimal positioning of sensors was also a notable success. For instance, positioning the LiDAR as low as possible enabled the detection of table and chair legs, while angling the 3D camera upwards facilitated the detection of humans by their heads and shoulders.

5.5.2 Powertrain Management (TC)

Completed by Tom Coleman

Challenges

By far the biggest challenge for the powertrain management subsystem was in researching components and deciding which ones were the most appropriate for their respective applications. There are hundreds of options available for components like motors, gearboxes, and wheels, as well as solutions that combine several elements together. Components needed to not only fulfil the needs of the AMR, but also be compatible with each other, and link to other subsystems seamlessly.

Another significant challenge was in adapting the methods and solutions used based on changing factors that the plans did not account for, such as component availability and compatibility. For example, the project originally intended for 200 mm drive wheels to be used, but a virtual fit check showed that they would not fit with the motors chosen, and 100 mm wheels were opted for instead. 100 mm swivel casters from storage were used in the final design rather than the planned 75 mm, because the lead time for ordering new casters was too long.

Successes

A significant success of the powertrain subsystem was the research process focused on case studies. This focus on well-established industry products narrowed the focus of the research, and enabled justifications for component selection based on existing success rather than experimental speculation.

The testing process for the motors was also successful. The approach, focused on progressively enabling direction control, speed control, and then speed feedback, allowed the

powertrain to be integrated with the rest of the system quickly and produce a functional AMR that fulfilled as many technical requirements as possible.

5.5.3 Electrical Power Management (JB)

Completed by Joe Bailey

Challenges

The component selection part of this subsystem had its difficulties, mainly due to the dependence on other subsystems such as the motor selection which directly affected the voltages and capacities required. However, once these components and so the maximum power requirements had been finalised, the selection was more straightforward. Additionally, due to the intermittent movement of AMRs, it is difficult to fully understand the energy requirements which influences the Ah capacity of the battery.

One of the largest challenges was component availability from the list of approved suppliers for the project, with the main vendors such as RS, Digikey, Farnell, and Conrad not stocking any appropriate lithium 24V batteries. This resulted in a design change from the originally selected LiPO4 batteries to a Li-NMC 18650 system in the final design. It would have been beneficial to carry out product research solely on components that were available to be ordered through the list of approved suppliers to ensure that the products selected in the SoTA could be sourced.

The layout of this power management on the initial prototype was not ideal, mainly due to the high time pressure during the final weeks, incorporating breadboards and perf board. Ideally, a DIN rail (*Figure 88*) would be utilised onboard to allow clear access to the 24V and 5V voltage levels on the AMR and the addition of components like fuses and switches, making it easier for those less familiar with the system to make adjustments and modifications.

Figure 88: DIN Rail Example [86]

Successes

Aside from the inconvenience of the trailing lead of the lab bench power supply, this subsystem was successful in powering all of the demands of the components onboard the AMR, supplying the 24V for the powerful motors, and also utilising the 24-5V step-down converter to supply the computing systems and sensors without issue. Additionally, no components were damaged or broken by short-circuits or incorrect installation during the testing phase of the project.

5.5.4 Frame & Carrying System (JOS)

Completed by Jenny Öborn Sandström

Challenges

One of the initial challenges was pinpointing and defining which aspects were relevant to research and include in the state-of-the-art review. This process required careful consideration to ensure the research focused on the most critical elements while avoiding extraneous information.

Another challenge was adjusting decisions about the subsystem during the project. As the design needed to adapt to the materials available at the school, we often had to make quick iterations. This dynamic approach meant constantly revising our plans to align with available resources, such as specific types of aluminium profiles.

Time constraints also posed a challenge, particularly during the creative phase. The limited time available for exploring different ideas meant that only a few concepts were thoroughly investigated. The transition from the research phase to the prototyping phase occurred rapidly, which restricted the depth of exploration in the ideation phase.

Successes

A significant positive outcome was the availability of 45x45mm aluminium beams at the university, instead of the originally intended 20x20mm beams. This unexpected resource resulted in a more stable prototype, enhancing the overall structural integrity of the AMR.

Collaboration during the implementation phase was another success. Integrating all subsystems with the frame and carrying system required cohesive teamwork. The group demonstrated excellent cooperation, ensuring that the subsystems worked together seamlessly.

Additionally, the collaboration during the prototyping of the frame and carrying system was exemplary. The entire group provided support, contributing to the efficient construction and assembly of the subsystem. This collective effort not only expedited the prototyping process but also fostered a sense of teamwork and shared responsibility.

5.5.5 User Interface & Ergonomics (ZM)

Completed by Zal Motafram

Challenges

The main challenges faced by the UIE subsystem was creating a command user interface and integrating the emergency stop. These challenges were mainly caused by the unexpected lead time on component arrival and the impending time constraint of the project.

As the Raspberry Pi 5 did not arrive on time, alternatives were sort out to first build ROS onto the system and have a functioning programming interface. This was achieved on the Raspberry Pi 4B; however, due to the time constraint, the command interface was not created. Basic commands and tasks can be sent or programmed into the AMR, but this is only achieved though the programming interface. To develop the command interface, via the programming interface, is not a simple task; however, there are plenty of tutorials and information available on the ROS development website, which highlights the benefit of ROS being an open-source framework for developers to exploit [87] [88].

Additionally, this led to the challenge of the ROS2 framework being unable to integrate with the Arduino within the project's timeframe. This impacted the PTM Subsystem, who had

created achieved motor movement with the Arduino, but the Raspberry Pi was only able to send command to the Arduino via the IDE and not through ROS. Therefore, the AMR's movements were limited and had to be run directly through the Raspberry Pi and not the network.

Furthermore, as the 7" LCD touchscreen only arrived at the latter-end of the project's timeframe, an alternative 5" LCD touchscreen was used in conjunction with the Raspberry Pi 4B alternative. The testing of the alternative architecture was successful, as both components worked seamlessly together. Therefore, the operator panel and mounting architecture, as shown in the CAD development, was never carried out as the intended components were not available and the time constraint restricted the time to implement the 7" touchscreen after its arrival.

Due to the use of the back dated Ubuntu 20.04 LTS Server being used, the touchscreen feature had to be implemented manually in the boot files. This initially cause calibration issues and the touchscreen currently does not work on the extreme axis of the display. Therefore, it is still necessary to navigate the programming interface using the mouse and keyboard, affecting the usability of the interface. The touchscreen is mainly intended for the command interface and if the same occurs on the 7" touchscreen, the interface can be designed within the extremes of the display axis and function properly.

The required DPDT emergency stop, nor the required battery architecture arrived within the project's timeframe and there were no viable alternatives available within the workshop's inventory. Thus, the implementation and testing of the architecture was never carried out, despite being thoroughly though-out and designed. Therefore, this impacted the testing of the user safety architecture considered by both the UIE and EPMS subsystems.

Regardless, there is enough design material from the architecture and SoTA that these tasks can be carried out in the future. The implementation and integration of the 5" LCD with the Raspberry Pi 4B will be a parallel process to the implementation and integration of the 7" with the Pi 5, which is well-documented within this report. By using the Gestalt Principles and Neilsen's Heuristics, an effective command interface and operator panel can be designed while still following the intended architecture. The emergency stop architecture and flow is simple, which can be tested once all the necessary components are made available.

Successes

The successes for the UIE subsystem are significant and these successes allowed for the entire AMR system to progress. It was paramount that a GUI was provided for the Raspberry Pi and that ROS was built onto the Pi, to carry out necessary testing.

Despite the intended architecture not being available, the Raspberry Pi 4B and the 5" touchscreen still worked together seamlessly. The collaboration with the ECUN subsystem during the build and testing of the AMR, paved the success in implementing ROS onto the Pi 4B and getting the touchscreen working, and thus, proved the architecture of the design, allowing for the development and function of an effective programming user interface.

The implementation of a light-weight Ubuntu Server image, rather than a desktop image, allowed for more processing space on the Raspberry Pi and increased the interface's efficiency for processing. Furthermore, the Ubuntu image is commonly used among ROS developers, proving the consistency and standards heuristics considered in the design of the AMR.

Furthermore, by iteratively testing different methods, with the ECUN subsystem, of building and using ROS allowed for the exploration of different techniques to programme the AMR. This meant that there was not only a programming interface available on the Raspberry Pi, but also on a host computer, a docker and a virtual machine. This greatly enhanced the freedom and flexibility required for the usability of a programming interface.

Additionally, the successful creation of a ROS network via a LAN connection, gave the user an external computer with the ability to send and receive information for conducting processes. This further increased the user control, freedom, flexibility, and efficiency heuristics demonstrated by this AMR. This also provides the user the ability to integrate their own PC to the LAN and use a docker or VM to communicate with AMR via the ROS framework.

Design choice made in parallel by both the FCS and UIE subsystems, allowed for the implementation and integration of a robust, as well as an ergonomically considerate, frame. The frame is 700mm tall and the user does not require any extra reach to load/unload the AMR. Furthermore, the designed placement of the operator panel in the CAD and the hinged display screen, also ensures the user is still upright when reaching the emergency stop or the command/programming user interface, by adjusting the display screen to a comfortable position.

Close collaboration with the EPMS, and consideration of International Safety Standards, allowed for the successful creation of the emergency stop architecture design. This made sure that it was well though-out, followed the consistency and standards of industry, and can now be tested to prove the design. By integrating the Arduino with the Raspberry Pi, allowed the PTM subsystem to test their motors on the AMR and made sure their programme functioned as intended.

Due to the extensive documentation, organised file, and CAD structure, as well as the weekly reports and PowerPoint presentations, there is plenty of help and documentation available. Despite not being in the traditional sense as a user manual, this has highlighted that the usability of the AMR is provided for suture developers and users to utilise.

Overall, the successes of the UIE subsystem are attributed to wide-ranging research, technical design choices and extensive cross-functional collaboration between subsystems. The iterative design and prototyping process, encouraged by the V-model, allowed this subsystem, together with the others, to make significant progress in proving the designed architecture of the AMR.

6 Conclusion

During this project there has been significant success as well as challenges. Together, all the subsystems progressed in designing, building and testing their architecture and successfully proved the system design of the AMR. Due to the time constraint and delivery lead time, there were challenges faced in the build of the architecture; however, alternative means of testing were sought out and conducted.

Therefore, this provides a proof of concept for the AMR and is evidence that it is feasible to build a functional AMR in the S.mart workshop, giving full freedom to the user. Overall, this project has achieved significant milestones throughout and has progressed very well within the given time frame.

6.1 Further Development

As mentioned previously in the discussion, there were several objectives not met due to time constraints even if the system is usable for autonomous navigation algorithms. We have included all of the information necessary for completion of all of the objectives such as circuit diagrams, Arduino code, and block diagrams.

The calculations for the energy requirements for the AMR were limited to an estimate based off battery information from commercial robots. This is because the power draw of the robot at any given moment is highly dependent upon the computations running, and whether the robot is accelerating or decelerating, which is hard to predict unless the exact mission is known. A future research direction could be to further investigate these power requirements, possibly even installing power monitors onboard the AMR to measure how closely the real data follows a simulated robot power simulation. It would be useful to optimise the size of the battery for specific mission requirements, reducing the mass of the robot, cost, and improving agility.

The initial prototype of the robot is also open to several improvements. Once the batteries are delivered, a clear and concise cable management plan can be implemented. This plan would include clearly labelled terminal blocks, as well as industry-standard connectors and headers for all connections. Additionally, all wires could be neatly concealed by the side panels. A future idea could look like the sketch in *Figure 31* which is an aesthetic design which would seamlessly fit in the S.mart workshop.

The State of the Art for the Powertrain subsystem discussed experimental methods of robotics movement such as omni-wheels. While the final design incorporated more conventional wheels to ensure a working product, there is room for future research into the benefits of using new locomotion technologies. For example, omni-wheels would not produce the unpredictable variations in position caused by swivel casters turning about an axis that is not in line with the centre of their wheels.

As a ROS Network was established, this utility can further be expanded to create a wider network over the LAN connection. As shown in *Figure 89*, a potential framework for a ROS Network is presented. It utilises all the methods of ROS that have been explored thus far in the project. It demonstrates how multiple nodes could be running and how multiple computers, potentially with different programming developers, are able to simultaneously communicate with each other, via the DHCP4 protocol.

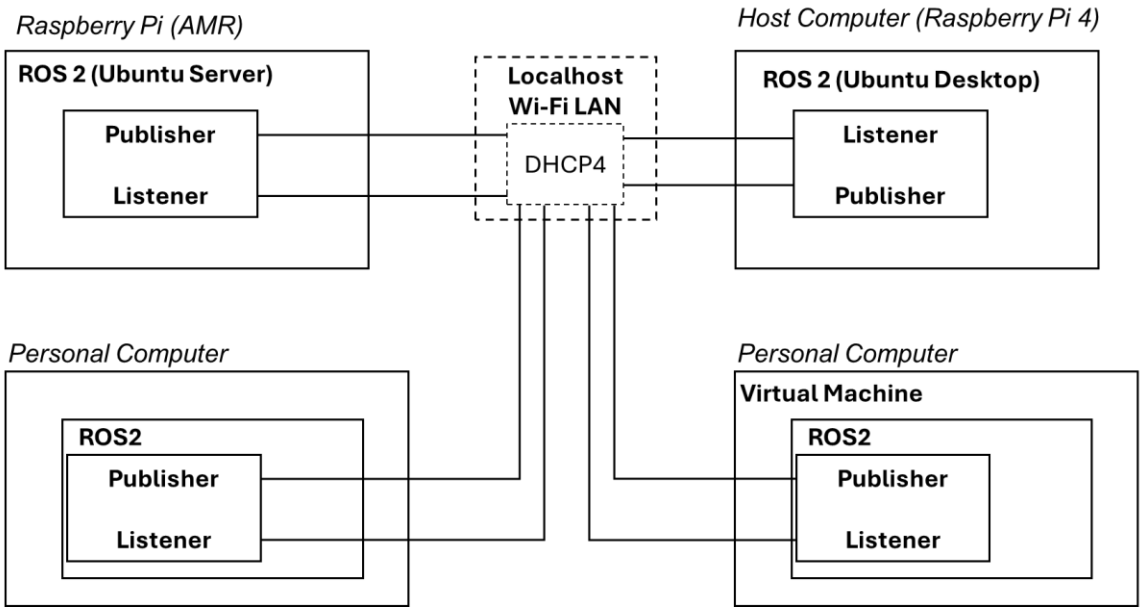


Figure 89: Potential framework for a ROS network, utilising different methods of ROS.

6.2 Handover

The handover process will be important for any students or staff working with this project in the future. A collection of documents and a list of goals aimed at fulfilling the remaining technical requirements have been prepared. Included in this collection is a timeline describing the tasks needed to achieve a complete AMR, when they should be done, and how long they are expected to take.

This documentation is included within the Final Defence PowerPoint and a separate Word Document, which includes useful tutorials and suggestions for developing the AMR. Furthermore, throughout this document, there are multiple references to useful open-source documentation and methods used by each subsystem that has assisted in the development of the AMR and could help in further developing its function.

6.3 Personal Feedback

6.3.1 Electronic Control Unit & Navigation (Oliver)

Studying abroad at Grenoble INP – Génie Industriel has been a fantastic experience. I have enjoyed every day working with the team and staff, who have been incredibly supportive. I have learned a lot about AMRs as well as gaining valuable general project experience. Leading the ECU and Navigation part of this project, the 'brain' of the robot, has allowed me to understand the integration and functionality of all system components. This has enhanced both my technical and leadership skills.

In addition to the academic and professional growth, I am grateful to have had the opportunity to swim, run, and ski in my free time here in Grenoble. It has also been wonderful to bring the team together in a weekly run club. I will definitely miss it here and will fondly remember my few months studying in the Alps.

6.3.2 Powertrain Management (Tom)

My stay in Grenoble has felt like the culmination of all my previous years of studying engineering. I was able to work with fellow students in a project that would have a real impact on Génie Industriel and allow its own students to gain more experience in robotics themselves. I was able to apply skills that I had learned in classes, but never put into practice outside of coursework. These included the use of CAD, solving mechanics problems related to motors, and designing and producing circuits, to name a few. This project was well-suited for my specific skillset as a student of Integrated Mechanical and Electrical Engineering.

My favourite experience was being able to collaborate with other students and professionals in the university. Being able to bounce ideas off them and work together to solve problems improved my ability to communicate and work with others whose expertise and opinions differed from my own.

6.3.3 Electrical Power Management (Joe)

Studying here has been the highlight of my university experience. I've enjoyed getting involved in such as complex and interesting project and getting freedom to figure things out for ourselves as a team, while still having a great level of support from our supervisors. As well as the research, I was fortunate to experience the hands-on aspects, especially the additive manufacturing and rapid prototyping part of the project which was a great learning experience. I was incredibly lucky to be part of such an exceptional team who worked consistently throughout the semester and were great to be around.

I've not only loved the university aspect, but the city itself and being an Erasmus student. I've got to do things I never thought I would, like skiing for the first time. I've also had the opportunity to get involved in music here and had the opportunity to play at the Génie Industriel Gala alongside researchers at the university which was brilliant. I became a member of Oliver's weekly run club, regularly played squash with friends, and went on weekly hikes with the great group we have here too. I'm genuinely going to miss Grenoble and this extraordinary lifestyle.

6.3.4 Frame & Carrying System (Jenny)

Studying at Grenoble INP in the French Alps has been an amazing experience. When I started my studies, I joined a project group with four people from England. We quickly got to know each other, and everyone was very helpful and friendly. Working closely with them has also significantly improved my English.

One of the things I have truly appreciated here is that the people in the city are very sporty and open to adventure. The active lifestyle in Grenoble has inspired me to participate in various outdoor activities.

I will miss the nature around Grenoble, with its endless opportunities for outdoor pursuits. Skiing, cycling, and hiking are just a few of the activities I have enjoyed during my time here. Having access to such fantastic nature has truly enriched my time studying in a unique way.

6.3.5 User Interface & Ergonomics (Zal)

I have really enjoyed my time here, at the Génie Industriel, while conducting the Responsible Design project. The AMR project has been one of my most enjoyable works, especially in the dynamic and collaborative environment alongside my peers and the professionals at the INP.

The opportunity to apply my degree in design engineering to this project was very fulfilling. Introducing the V-model for project management and structure was a significant in the team's progression from concept to design and integration. For my subsystem, I applied design tools such as the Gestalt principles, Nielsen's heuristics, and ergonomics for user centred design – all of which I had learnt in my previous semester at Bath. This hands-on application of theoretical knowledge was incredibly rewarding, and I was very happy to see the practical application of my design. There is nothing more satisfying for an engineer, than seeing your design and ideas come to life.

I fall short when expressing my immense gratitude to both the Génie Industriel and the University of Bath for this opportunity. I thoroughly enjoyed all the amazing activities and landscapes around me, while studying in the middle of the Alpes.

I will miss the culture, the beauty of Grenoble and the surprisingly affordable skiing.

Above all, I will miss the people.

References

[1] WAREHOUSE AUTOMATION, “What are automated guided vehicles?,” RIVER SYSTEMS, 19 01 2023. [Online]. Available: https://6river.com/what-are-automated-guided-vehicles/. [Accessed 06 03 2024].

[2] F. France, “Mobile robot Sherpa B,” FSI France, 2024. [Online]. Available: https://www.fsi-france.fr/produit/robot-mobile-sherpa-b/. [Accessed 13 02 2024].

[3] University of Bath, Faculty of Engineering and Design, “Faculty of Engineering & Design,” University of Bath, 2024. [Online]. Available: https://www.bath.ac.uk/faculties/faculty-of-engineering-design/. [Accessed 2024].

[4] KTH Royal Institute of Technology, “KTH Sweden,” KT, 2024. [Online]. Available: https://www.kth.se/en. [Accessed 2024].

[5] Grenoble INP - Génie industriel, “Génie industriel,” Université Grenoble Alpes, 2024. [Online]. Available: https://genie-industriel.grenoble-inp.fr/. [Accessed 2024].

[6] Grenoble INP, “S.mart Grenoble-Alpes,” Grenoble INP, 2024. [Online]. Available: https://www.grenoble-inp.fr/fr/organisation/aip-primeca-dauphine-savoie#page-presentation. [Accessed 2024].

[7] S. Pugh, Total Design - Integrated Methods of successfull product engineering, Wokingham, England: Addison-Wesley Pub. Co., 1991.

[8] K. Forsberg, H. Mooz and H. Cotterman, Visualizing Project Management, New York: John Wiley and Sons, 2005.

[9] J. M. Wilson, “Gantt charts: A centenary appreciation,” *European Journal of Operational Research 149,* pp. 430-437, 2002.

[10] S. C. Wheelwright and K. B. Clark, Revolutionizing product development: quantum leaps in speed, efficiency, and quality, New York: Simon & Schuster UK, 1992.

[11] Interaction Design Foundation - IxDF, “What is User Centered Design (UCD)?,” Interaction Design Foundation - IxDF, 2024. [Online]. Available: https://www.interaction-design.org/literature/topics/user-centered-design. [Accessed 05 03 2024].

[12] IBM Documentation Help, “Use-case diagrams,” IBM Corporation, 21 09 2023. [Online]. Available: https://www.ibm.com/docs/en/rational-soft-arch/9.7.0?topic=diagrams-use-case. [Accessed 04 03 2024].

[13] C. v. Ehrenfels and B. Smith, “On 'Gestalt qualities' (trans. B. Smith). In Barry Smith (ed.), Foundations of Gestalt Theory,” *Philosophia,* pp. 82-117, 1988.

[14] J. Wagemans, J. Feldman, S. Gephstein, R. Kimchi, J. Pomerantz, v. d. Helm and Peter, “A Century of Gestalt Psychology in Visual Perception: II. Conceptual and Theoretical Foundations,” *Psychological Bulletin,* vol. 138, no. 6, pp. 1218-1252, 2012.

[15] M. Haverkamp, “Gestalt Principles,” in *Synesthetic Design: Handbook for a Multi-Sensory Approach*, Berlin, Boston: Birkhäuser, 2013, pp. 128-132.

[16] Apple Inc., “Use and customize Control Center on iPhone,” 2024. [Online]. Available: https://support.apple.com/en-gb/guide/iphone/iph59095ec58/12.0/ios/12.0. [Accessed 29 03 2024].

[17] The British Standards Institution, *Safety of machinery — Safety-related parts of control systems,* London: BSI Standards Limited 2023, 2023.

[18] The British Standards Institution, *Safety of machinery — Emergency stop function — Principles for design,* London: BSI Standards Limited 2019, 2015.

[19] J. Dirken, “Productergonomie - ontwerpen voor gebruikers,” DUP Blueprint, Delft, 2001.

[20] DINED, “DINED Anthropometric Database Tool,” TU Delft factulty of Industrial Design Engineering, 2004. [Online]. Available: https://dined.io.tudelft.nl/en/database/tool. [Accessed 25 02 2024].

[21] The British Standards Institution, *Robots and robotic devices — Safety requirements for industrial robots,* London: BSI Standards Limited 2014, 2014.

[22] SMR, “sherpa_datasheet_sherpab_gb.pdf,” 2022. [Online]. Available: https://files.mynorcan.com/norcan/sherpa_datasheet_sherpab_gb.pdf. [Accessed 19 02 2024].

[23] J. Nielsen, “10 Usability Heuristics for User Interface Design,” Nielsen-Norman Group, 24 04 1994. [Online]. Available: https://www.nngroup.com/articles/ten-usability-heuristics/. [Accessed 25 02 2024].

[24] Visiter La boutique Arduino, “Arduino Mega 2560 REV3 microprocesseur [A000067],” Amazon, 2024. [Online]. Available: https://www.amazon.fr/Arduino-Mega-2560-R3-Microcontr%C3%B4leur/dp/B0046AMGW0. [Accessed 23 05 2024].

[25] B. Diver, “PLCs vs. Microcontrollers: What Is the Difference?,” Industrial, 15 02 2022. [Online]. Available: https://industrialautomationco.com/blogs/news/plcs-vs-microcontrollers-what-is-the-difference. [Accessed 05 03 2024].

[26] Husarion Docs, “ROSbot (2R | 2 PRO | 2),” ROSbot, 2024. [Online]. Available: https://husarion.com/manuals/rosbot/. [Accessed 21 02 2024].

[27] Raspberry Pi, “Raspberry Pi 4 Tech Specs,” 2024. [Online]. Available: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/. [Accessed 04 03 2024].

[28] SLAMTEC, “RPLIDAR A3 Low Cost 360 Degree Laser Range Scanner Introduction and Datasheet,” 24 01 2018. [Online]. Available: https://www.generationrobots.com/media/LD310_SLAMTEC_rplidar_datasheet_A3M1_v1.0_en.pdf. [Accessed 23 02 2024].

[29] Amazon, “RPLIDAR A3M1 360 Degree 2D Laser Range Sensor Kit, 15Hz Scan Rate and 25 Meters Distance Ladar Scanner Module for Intelligent Obstacle/Robot/Maker

Education," SLAMTEC, 2024. [Online]. Available: https://www.amazon.com/RPLIDAR-A3M1-Distance-Intelligent-Education/dp/B08ZDKQM9K. [Accessed 01 06 2024].

[30] Intel RealSense, "Intel RealSense Depth Camera D435i," Intel, 2024. [Online]. Available: https://www.intelrealsense.com/depth-camera-d435i/. [Accessed 14 05 2024].

[31] Unmanned Tech, "Intel RealSense Depth Camera D435i," 2024. [Online]. Available: https://www.unmannedtechshop.co.uk/product/intel-realsense-depth-camera-d435i/. [Accessed 01 06 2024].

[32] M5Stack, "M5Stack Time-of-Flight Distance Unit (VL53L1X)," RobotShop, 2024. [Online]. Available: https://eu.robotshop.com/fr/products/m5stack-time-of-flight-distance-unit-vl53l1x?gad_source=1&gclid=CjwKCAjwrIixBhBbEiwACEqDJUmldJBO0xnACwdCdp0rJ030-24E4u4n_JDW7KgMHucafy9gFoJx_hoCnSUQAvD_BwE. [Accessed 23 05 2024].

[33] T. E. o. E. Britannica, "torque," Encyclopedia Britannica, 10 05 2024. [Online]. Available: https://www.britannica.com/science/torque. [Accessed 01 06 2024].

[34] Encyclopaedia Britannica, "Power | Energy, Force & Work," Encyclopaedia Britannica, 23 01 2024. [Online]. Available: https://www.britannica.com/science/power-physics. [Accessed 25 03 2024].

[35] E. R. (. Jones and R. L. Childers, Contemporary College Physics, Addison-Wesley, 1993.

[36] T. E. o. E. Britannica, "angular velocity," Encyclopedia Britannica, 15 05 2024. [Online]. Available: https://www.britannica.com/science/angular-velocity. [Accessed 01 06 2024].

[37] Crouzet Motors, "Industrial Brushless Geared Motor 801897 TNi20 80W," 11 02 2015. [Online]. Available: https://soda.crouzet.com/pn/?i=80189705. [Accessed 31 05 2024].

[38] Analog Devices, "Non-Inverting Op Amp," Analog Devices, [Online]. Available: https://www.analog.com/en/resources/glossary/non-inverting-op-amp.html. [Accessed 31 05 2024].

[39] Electronics Notes, "What is a Voltage Divider or Potential Divider," Electronics Notes, [Online]. Available: https://www.electronics-notes.com/articles/basic_concepts/voltage/voltage-potential-divider.php. [Accessed 31 05 2024].

[40] Blickle, "product GEVN 200/25H7," Blickle, [Online]. Available: https://www.blickle.com/product/GEVN-200-25H7-755847. [Accessed 01 06 2024].

[41] "Blickle GTHN 100/20H7 transport wheel," RUBIX, [Online]. Available: https://fr.rubix.com/fr/roulette-pivotante-en-tole-dacier-version-fortes-charges-avec-platine-a-visser-avec-blocage-ideal-stop-roue-fortes-charges-avec-bande-de-roulement-en-polyurethane-blickleextrathane-avec-corps-de-roue-en-aluminium/p-G1038003072. [Accessed 28 05 2024].

[42] Crouzet, "Moteurs a courant continu BRUSHLESS," [Online]. Available: https://docs.rs-online.com/b747/0900766b806501a0.pdf. [Accessed 01 06 2024].

[43] Raspberry Pi, "Raspberry Pi 27W USB-C Power Supply," [Online]. Available: https://www.raspberrypi.com/products/27w-power-supply/. [Accessed 01 06 2024].

[44] “Notice d'instructions Sherpa B,” Sherpa, 09 2022. [Online]. Available: https://files.mynorcan.com/norcan/notice_instructions_sherpab.pdf. [Accessed 21 03 2024].

[45] “MiR250,” MIR, 31 03 2021. [Online]. Available: https://www.scanditron.com/wp-content/uploads/2021/03/MiR250-_-Mobile-Industrial-Robots.pdf. [Accessed 20 03 2024].

[46] Robotics, PAL, “PAL Robotics Aran,” [Online]. Available: https://pal-robotics.com/robots/aran/. [Accessed 01 06 2024].

[47] “HD-1500 Platform User's Manual,” Omron, [Online]. Available: https://www.manualslib.com/manual/1895894/Omron-Hd-1500-Platform.html. [Accessed 18 03 2024].

[48] INNPO, “Lithium batteries 18650 battery 24V 12Ah,” [Online]. Available: https://innpo.fr/batteries-au-lithium-rechargeable/batteries-lithium-18650-batterie-24v-12ah-innpo-batteries-au-lithium-rechargeable.html. [Accessed 01 06 2024].

[49] Digikey, “RSD-60G-5,” MeanWell, [Online]. Available: https://www.digikey.fr/en/products/detail/mean-well-usa-inc/RSD-60G-5/7706261. [Accessed 31 05 2024].

[50] INNPO, “24V Lithium Battery Charger Pack,” INNPO, [Online]. Available: https://innpo.fr/piles-lithium-de-chargeurs/chargeur-de-batterie-au-lithium-de-24v-pack-innpo-piles-lithium-de-chargeurs.html. [Accessed 31 05 2024].

[51] Alstrut, “Aluminium Profile Systems,” 2024. [Online]. Available: https://www.alstrut.com/productdetail/aluminium-profile-systems. [Accessed 2024].

[52] Raspberry Pi, “Raspberry Pi Touch Display,” Raspberry Pi, 2024. [Online]. Available: 29.

[53] Raspberry Pi, “Raspberry Pi Display Cable,” Raspberry Pi, 2024. [Online]. Available: https://www.raspberrypi.com/products/display-cable/. [Accessed 29 03 2024].

[54] Mouser Electronics, Inc., “SC0025,” Mouser Electronics, Inc., 2024. [Online]. Available: https://www.mouser.fr/ProductDetail/Raspberry-Pi/SC0025?qs=T%252BzbugeAwjgCqls6B%252BePxw%3D%3D&src=raspberrypi. [Accessed 01 04 2024].

[55] Farnell, “RASPBERRY-PI SC1131,” Premier Farnell Limited, 2024. [Online]. Available: https://fr.farnell.com/raspberry-pi/sc1131/fpc-cablesdisplay-200mm/dp/4263053. [Accessed 01 04 2024].

[56] Moniteur Devices Inc., “What is the Difference Between, SPST, SPDT and DPDT?,” Moniteur Devices Inc., 2019. [Online]. Available: https://moniteurdevices.com/knowledgebase/knowledgebase/what-is-the-difference-between-spst-spdt-and-dpdt/. [Accessed 24 04 2024].

[57] RS PRO, “Bouton d'arrêt d'urgence RS PRO, Montage panneau,” RS Components, 2024. [Online]. Available: https://fr.rs-online.com/web/p/boutons-d-arret-d-urgence/7452442. [Accessed 05 04 2024].

[58] European Commission, “RoHS Directive - European Commission,” 03 06 2012. [Online]. Available: https://environment.ec.europa.eu/topics/waste-and-recycling/rohs-directive_en. [Accessed 03 06 2024].

[59] European Commission, “Waste from Electrical and Electronic Equipment (WEEE) - European Commission,” [Online]. Available: https://environment.ec.europa.eu/topics/waste-and-recycling/waste-electrical-and-electronic-equipment-weee_en. [Accessed 03 06 2024].

[60] Pysource, “Distance Detection With Depth Camera (Intel Realsense D435i),” Pysource, 11 03 2021. [Online]. Available: https://pysource.com/2021/03/11/distance-detection-with-depth-camera-intel-realsense-d435i/. [Accessed 14 05 2024].

[61] Intel® RealSenseTM, “Intel® RealSenseTM Product Family D400 Series Datasheet,” 03 2024. [Online]. Available: https://www.intelrealsense.com/download/21345/?tmstv=1697035582. [Accessed 14 05 2024].

[62] INTEL RealSense, “librealsense,” GIT, 2024. [Online]. Available: https://github.com/IntelRealSense/librealsense. [Accessed 02 06 2024].

[63] GIT, “Librealsense,” GIT, 2018. [Online]. Available: https://github.com/IntelRealSense/librealsense. [Accessed 31 05 2024].

[64] Slamtec, “rplidar_ros,” GIT, 2024. [Online]. Available: https://github.com/Slamtec/rplidar_ros. [Accessed 25 05 2024].

[65] AWS, “What’s the Difference Between Docker and a VM?,” Amazon, 2024. [Online]. Available: https://aws.amazon.com/compare/the-difference-between-docker-vm/#:~:text=A%20VM%20lets%20you%20run,%2C%20processes%2C%20and%20network%20capabilities.. [Accessed 20 05 2024].

[66] ROS, “Using GUIs with Docker,” ROS.org, 2024. [Online]. Available: https://wiki.ros.org/docker/Tutorials/GUI. [Accessed 31 05 2024].

[67] OpenRobotics, “Running ROS 2 nodes in Docker,” ROS 2, 2024. [Online]. Available: https://docs.ros.org/en/foxy/How-To-Guides/Run-2-nodes-in-single-or-separate-docker-containers.html. [Accessed 31 05 2024].

[68] Open Robotics, “Ubuntu (Debian packages),” ROS 2, 2024. [Online]. Available: https://docs.ros.org/en/humble/Installation/Ubuntu-Install-Debians.html. [Accessed 01 06 2024].

[69] Canonical Ltd. Ubuntu and Canonical, “Ubuntu 20.04.6 LTS (Focal Fossa),” Ubuntu, 2018 . [Online]. Available: https://releases.ubuntu.com/focal/. [Accessed 01 06 2024].

[70] Open Robotics, “Installing ROS 2 via Debian Packages,” ROS2, 2024. [Online]. Available: https://docs.ros.org/en/crystal/Installation/Linux-Install-Debians.html. [Accessed 01 06 2024].

[71] Xubuntu, “Xubuntu is a community developed operating system that combines elegance and ease of use.,” Canonical , 2012. [Online]. Available: https://xubuntu.org/. [Accessed 01 06 2024].

[72] Automatic Addision, “How to Install ROS 2 Navigation (Nav2),” ROS 2, 24 03 2021. [Online]. Available: https://automaticaddison.com/how-to-install-ros-2-navigation-nav2/. [Accessed 01 06 2024].

[73] C. Davies, “Free upgrade: Raspberry Pi 4 now with 1.8GHz instead of 1.5GHz,” PiCockpit, 19 11 2021. [Online]. Available: https://picockpit.com/raspberry-pi/free-upgrade-raspberry-pi-4-now-with-1-8ghz-instead-of-1-5ghz/. [Accessed 03 06 2024].

[74] Raspberry Pi, “Raspberry Pi 5,” [Online]. Available: https://www.raspberrypi.com/products/raspberry-pi-5/. [Accessed 03 06 2024].

[75] A. Froelich, “Bottleneck,” West Gate Networks, TechTarget, 09 2021. [Online]. Available: https://www.techtarget.com/searchnetworking/definition/bottleneck. [Accessed 29 05 2024].

[76] L. Rendek, “Ubuntu Server 20.04: Connect to WiFi from command line,” 13 08 2023. [Online]. Available: https://linuxconfig.org/ubuntu-20-04-connect-to-wifi-from-command-line. [Accessed 23 05 2024].

[77] GitHub Community Forum, “Dropped frames when using network camera,” GitHub, 2021. [Online]. Available: https://github.com/IntelRealSense/realsense-ros/issues/1808. [Accessed 30 05 2024].

[78] A. Maupate, “From AZERTY to QWERTY,” Medium, 11 08 2019. [Online]. Available: https://ambroisemaupate.medium.com/from-azerty-to-qwerty-1a9b294040f6. [Accessed 02 06 2024].

[79] S. &. K. K. M. Doddamani, “Review of Experimental Fracture Toughness (K IC) of Aluminium Alloy and Aluminium MMCs,” vol. 1, pp. 38-51, 2016.

[80] G. C.-E. P. H.-F. F. Carrillo-Sánchez, “A study of the fracture toughness of acrylic composites using the essential work of fracture method,” *Polymer Testing,* vol. 29, no. 5, pp. 565-571, 2010.

[81] Quality One, “Introduction to Failure Mode and Effects Analysis (FMEA),” [Online]. Available: https://quality-one.com/fmea/#:~:text=Failure%20Mode%20and%20Effects%20Analysis%20(FMEA)%20is%20a%20structured%20approach,harmful%20outcomes%20for%20the%20customer. . [Accessed 03 06 2024].

[82] S. M. ROBOTICS, “SHERPA®,” 09 2022. [Online]. Available: https://files.mynorcan.com/norcan/notice_instructions_sherpab.pdf. [Accessed 18 02 2024].

[83] eibabo, “Capteur LiDAR 2D - Capteur de distance optique 70112344,” eibabo technology store, 2024. [Online]. Available: https://www.eibabo.fr/pepperl-fuchs/capteur-lidar-2d-capteur-de-distance-optique-70112344-

eb16390725?utm_source=Portals&utm_medium=CPC&utm_campaign=eibabo-FR_GoogleShopping_FR. [Accessed 07 03 2024].

[84] SICK Sensor Intelligence, “S30B-3011DA,” SICK, 28 02 2024. [Online]. Available: https://cdn.sick.com/media/pdf/9/49/849/dataSheet_S30B-3011DA_1056429_en.pdf. [Accessed 11 03 2024].

[85] MathWorks, “SLAM (Simultaneous Localization and Mapping),” MathWorks, 2024. [Online]. Available: https://www.mathworks.com/discovery/slam.html. [Accessed 03 03 2024].

[86] L. Eitel, “DIN rail: Basic construction and versions,” Motion Control Tips, [Online]. Available: https://www.motioncontroltips.com/din-rail-basic-construction-and-versions/. [Accessed 01 06 2024].

[87] Open Robotics, “ROS Tutorials,” 2024. [Online]. Available: https://wiki.ros.org/ROS/Tutorials. [Accessed 20 05 2024].

[88] Open Robotics, “Implementing custom interfaces,” 2024. [Online]. Available: https://docs.ros.org/en/foxy/Tutorials/Beginner-Client-Libraries/Single-Package-Define-And-Use-Interface.html. [Accessed 30 05 2024].

Appendix

Appendix A: Weekly Report Example

Weekly Status Report

Project Name	Automated Mobile Robot Project (AMR)
Time Period	05.02.2025 to 05.06.2024
Project Members	Joe Bailey, Oliver Brunnock, Tom Coleman, Zal Motafram, Jenny Öborn Sandström

Week 2 - 12.02.2025 to 16.02.2024

Tasks Completed this Week:

Task	*Date Completed*	*Deadline*	*Responsible*
Made a start on SOTA documents	07/02/2024		All
Understood the user needs	09/02/2024		All
Created a list of user needs	09/02/2024		All
Drafted a list of system requirements	09/02/2024		All
Created a Problem statement	15/02/2024		All
Meeting with Client discussing needs and asking questions	15/02/2024		All

Tasks In Progress:

Task	*Progress Update*	*Deadline*	*Responsible*
State of the Art – First draft	40%	23/02/2024	All
Technical Requirements	70%	23/02/2024	All
Fully finalise Gantt Chart	66%	21/02/2024	All

Tasks for Next Week:

Task	*Priority level*	*Deadline*	*Responsible*
Refine the list of system requirements	M	20/02/2024	All
Complete first draft of SOTA	H	22/02/2024	All
Discuss System Requirements with PD and create short presentation for the meeting	M	23/02/2024	All
Finalise functional hierarchy	M	23/02/2024	All
Brainstorm technical solutions to the task	M	23/02/2024	All
Begin to form individual subsystem specification, based on system requirements	L	23/02/2024	All

Upcoming Project Deliverables:

Intermediate Deliverable	*Deadline*
Technical Requirements Overview	17/02/2024
State of the Art Draft – To ensure we are on the right track	22/02/2024
Main Deliverables	***Deadline***
State of the Art (report individual)	Mid-March
Needs Analysis (Intermediate report and defence – group)	Mid-April
Opening (Prototype V0)	Mid-April
Development/Realisation (Final Report & Defence – group)	Mid-June

Appendix B: Technical Requirements

No. #	Version	Demand/Wish	Priority	Responsibility	Subsystem	Description	Performance Criteria	Value	Tolerance	Test Method
Functional Requirements										
A.1.1	V3	Demand		Oliver	ECU/Navigation (ECUN)	The system must be able to recognise objects on the ground	Detection of objects of a certain size	>30mm		
A.1.2	V3	Demand		Oliver	ECU/Navigation (ECUN)	Must be able to detect and avoid obstacles	Degree of visibility/Obstacle Avoidance	180°/50mm		
A.1.3	V3	Demand		Oliver	ECU/Navigation (ECUN)	Must be able to map environment, to a certain degree of accuracy	Navigation in room using straighest path	(algorithm)		
A.1.4	V3	Demand		Joe	EPMS	System must operate under full load for a minimum time	Battery life	2 hours		
A.1.5	V3	Demand		All	AMR All SSTs	AMR must navigate autonomously to set way points	Autonomous navigation to waypoints from given target location			
A.1.6	V4	Demand		Oliver	ECU/Navigation (ECUN)	Must be able to identify different objects	Identifies objects using 3D camera			
Load/Unloading										
A.2.1	V3	Demand		Jenny	Frame & Carrying System (FCS)	The system must be able to be loaded by hand within a maximum time	Time to load AMR	15 seconds	+/-5s	
A.2.2	V3	Demand		Jenny	Frame & Carrying System (FCS)	The system must be able to be unloaded by hand within a maximum time	Time to unload AMR	15 seconds	+/-5s	
A.2.3	V3	Demand		Jenny	Frame & Carrying System (FCS)	The system must be capable of carrying a maximum payload	Maximum load	40kg	+/-10kg	
A.2.4	V3	Demand		Jenny	Frame & Carrying System (FCS)	System must have a specified height	Dimensions	70cm	+/-1cm	(Virtual) Fit Check
A.2.5	V1	Wish	LOW	Jenny & Zal	FCS & UIE	Should be able to adjust shelf height dynamically for different table heights	Range of shelf height	20cm (adjusability)	+/-1cm	
Storage										
A.3.1	V3	Demand		Jenny & Tom	FCS & PTM	Load must not fall off platform during transportation, accounting for all braking scenarios as stated in the Failure Modes and Effects Analysis (FMEA)	Maximum allowable slide at max acceleration	2cm	+/-0.5cm	Emergency stop from maximum speed
3.2	V3	Demand		Jenny	Frame & Carrying System (FCS)	Must be capable of transporting a minimun of two standardised boxes (400x230x150mm)	Shelf surface area	500x560mm	+100mm	
3.3	V3	Wish		Jenny & Zal	FCS & UIE	Should be able to accommodate different types of loads	Loading and stability test			
3.4	V3	Demand		Jenny	Frame & Carrying System (FCS)	Payload shape and size is not limited	Shelf surface area			
3.5	V3	Demand		All	AMR All SSTs	Unloaded system weight must be below a specified weight	Unloaded system weight with batteries	50kg	+-10kg	
3.6	V3	Demand		Jenny	Frame & Carrying System (FCS)	AMR must be stored safely in the working environment, when not in operation	Risk Assesment			
Safety										
4.1	V4	Demand		All	AMR All SSTs	Must have no burrs or sharp edges	Corner bevel minimum radius	1mm	+/-0.5	ISO 13715
4.2	V3	Demand		Joe, Tom, Zal	EPMS, PTM & UIE	Emergency stop must be visible	Must conform to relevant standards			ISO13850
4.3	V3	Demand		Joe, Tom, Zal	EPMS, PTM & UIE	Must be capable of stopping within a defined distance from full speed and full load	Stopping Distance	400mm		
4.4	V1	Demand		Oliver	ECU/Navigation (ECUN)	Videos taken by the robot for navigation must not be stored	GDPR			
4.5	V3	Wish	HIGH	Zal	UI & Ergonomics (UIE)	Safety distance parameter should be adjustable by the user	Ease of adjustability			
4.6	V3	Demand		Joe	EPMS	User must be protected from electric shocks	High voltage systems inaccessible while on			
4.7	V3	Demand		Zal	UI & Ergonomics (UIE)	System must be visible and audible to users of the lab	Detectable sounds and lights			
4.8	V2	Demand		All	AMR All SSTs	System must not move when powered off	Push test			
Maintenance										
5.1	V3	Demand		All	AMR All SSTs	Minimum 60% parts must be replacable/maximise OEM parts	Ease of replaceability/standard parts	60%		
5.2	V3	Demand		Joe	EPMS	Must be able to charge overnight	Battery charge time	10 hours		
5.3	V1	Demand		Tom & Zal	PTM & UIE	The user must be able to manually apply the brake	Accessible through interface			
5.4	V2	Demand		Joe, Zal	EPMS & UIE	Must be able to be plugged in to charge	Time to plug in	20 seconds		
5.5	V2	Wish	MEDIUM	All	AMR All SSTs	Should be able to reach every accessible components using workshop tools	No interference between tooling and components	30 minutes		(Virtual) Fit check
Movement										
6.1	V3	Demand		Oliver	ECU/Navigation (ECUN)	Must be within a certain radius of target destination	Radius measurement	50mm		
6.2	V4	Demand		Oliver	ECU/Navigation (ECUN)	Must be within a certain orientation of target destination	Orientation	5°		
6.3	V1	Demand		Tom	Powertrain Mamagement (PTM)	Maximum speed must be above a certain speed	Maximum Speed	1.94m/s (7km/h)	+0.57m/s (+2km/h)	
6.4	V2	Demand		Tom	Powertrain Mamagement (PTM)	Must accelerate at a given rate	Acceleration	0.5m/s^2		
Communication										
7.1	V3	Wish	HIGH	Jenny, Oliver, Zal	FCS, ECUN, UIE	Should warn the user if platform is over loaded	Load sensitivity			
7.2	V3	Wish	MEDIUM	Oliver	ECUN & UIE	Should provide real-time feedback on its location	Maximum feedback delay	3 seconds		
7.3	V1	Demand		Oliver	ECUN & UIE	Must be able to receive destination commands	Interpretation of destination signals			
7.4	V1	Demand		Zal	UI & Ergonomics (UIE)	Must have a programmable interface	Use of ROS libraries			
7.5	V2	Wish	HIGH	Joe, Zal	EPMS & UIE	Should have a battery charge level indicator	Low Battery Warning Displayed and Cutoff			
7.6	V2	Wish	LOW	Jenny, Oliver, Zal	FCS, ECUN, UIE	Should warn the user if load is misplaced	Load distribution sensitivity			
7.7	V3	Demand		Oliver, Zal	ECUN & UIE	Must use Robot Operating System	N/A			
Configuration										
8.1	V2	Demand		Joe, Tom, Oliver, Zal	EPMS, PTM, ECUN & UIE	All parts must use connectors	No Soldering between parts			
8.2	V3	Wish	HIGH	Joe	EPMS	Top platform should include additional power supplies for user modifications	Extra power supply for additional components	5V and 24V Supplies		
Documentation										
9.1	V3	Demand		All	AMR All SSTs	All parts must be detailed	Bill of materials must be prepapred			
9.2	V3	Demand		All	AMR All SSTs	All parts should be identifiable	Labels easy to read and cross-referenced to BOM/Schematic			
9.3	V4	Wish	MEDIUM	All	AMR All SSTs	Should predict potential failure modes	Design for Failure Mode and Effects Analysis (FMEA)			
9.4	V3	Demand		All	AMR All SSTs	Material, tooling and equipment cost must be accounted for	Equipment and Tooling list, to append onto BOM			
9.5	V3	Demand		All	AMR All SSTs	User must have access to system information	Service & Instructions Manual must be available for the user			
9.6	V3	Demand		All	AMR All SSTs	Risk of AMR in ideal working environment must be assessed	Risk assessment documentation carried out			
Sustainability										
10.1	V1	Wish	MEDIUM	All	AMR All SSTs	Should perform for a minimum lifetime	Machine lifetime	5 years		
10.2	V3	Wish	MEDIUM	All	AMR All SSTs	Electronic equipment should be easily reused or recycled at the end of life	Compliance with WEEE & RoHS directive			
10.3	V3	Wish	MEDIUM	All	AMR All SSTs	Should maximise the use of recyclable and biodegradable materials	Use of sustainable materials	60%		
10.4	V3	Wish	HIGH	All	AMR All SSTs	Should minimise the carbon footprint of the AMR	Perform a Life Cycle Analysis			
Aesthetics										
11.1	V3	Wish	MEDIUM	Jenny, Zal	FCS & UIE	Should have a corporate aesthetic, adhearing to Gestalt rule of good continuation	Complete survey	Over 70% of people like it		

Appendix C: Noun Name Bill of Materials Supplier Quotes

Part Number	NOUN NAME	Description	Quanity	Ordered	Supplier 1 Name & Link	Supplier 1 Stock #	Supplier 1 Unit Cost	Supplier 1 Emailed?	Supplier 1 Quote Received?	Supplier 2 Name & Link
A01000	01. ECUN	ECU and Navigation								
A01001	ELECTRONIC CONTROL UNIT	RASPBERRY PI 5B 8GB STARTER KIT	1	✓	FARNELL	4341789	€ 110.55	✓	✓	Pearl
A01002	LIDAR	Scanner Laser 360° RPLIDAR A3 Slamtec (25 m)	1	✓	Reichelt	RPLIDAR A3M1	€ 733.51	✓	✓	Robotshop
A01003	VISION SENSOR	3D CAMERA	1	✓	RS	202-6352	€ 558.13	✓	✓	Reichelt
A01004	MICROCONTROLLER	MEGA 2560 Rév. 3 Arduino	1	✓	Reichelt	ARDUINO MEGA	€ 36.00	✓	✓	CONRAD
A01005	INFRARED	Time-of-Flight Distance Unit	4	✗	Robotshop	SKU: RB-Mst-365	€ 40.88	✓	✗	M5STACK
A02000	02. PTM	Powertrain Management								
A02001	MOTOR ASSEMBLY	Motoréducteur CC Crouzet, 24 V c.c., 80 W, 120 tr/min, dia. de l'arbre 19mm	2	✓	RS Components	499-0272	€ 735.06	✓	✓	Mouser
A02002	BRAKE	Brake Pad 2 pc(s) Black	1	✗			€ -	✗	✗	
A02003	DRIVE WHEEL	Heavy duty drive wheel, with hub keyway	2	✗	RUBIX	0174-7028061	€ 191.28	✓	✓	Blickle via FIREM
A02004	SWIVEL CASTER	Swivel Caster - 75mm diameter	4	✗	RS COMPONENTS	458-9743	€ 17.86	✗	✗	Tent
A03000	03. EPMS	Electrical Power Management System								
A03001	BATTERY	24V 12Ah Li-Ion Battery	2	✓	INNPO	18650_PACK_7S5P	€ 164.46	✓	✓	
A03002	24V-5V CONVERTER	RSD-60G-5	1	✓	DIGIKEY	1866-5464-ND	€ 42.13	✓	✓	FARNELL
A03003	CHARGER	3A Charger	1	✓	INNPO	CAR-29.4V-3A	€ 29.00	✓	✓	
A03004	BATTERY DISCONNECT SWITCH	S822D	1	✓	DIGIKEY	360-3230-ND	€ 33.11	✓	✓	MOUSER
A03005	CONTACTOR	Contactor for Emergency Stop	3	✓	DIGIKEY	PB2137-ND	€ 6.54	✓	✓	FARNELL
A03006	VOLTAGE READOUT	Voltage Readout	1	✗	DIGIKEY	1528-1145-ND	€ 7.44			
A04000	04. FCS									
A05000	05. UIE	User Interface & Ergonomics								
A05001	EMERGENCY STOP BUTTON	RS PRO Illuminated Emergency Stop Push Button, Panel Mount, DPDT, IP65	2	✓	RS	745-2442	€ 20.27	✓	✓	
A05002	DISPLAY TOUCH SCREEN	7-inch Raspberry Pi Display, Capacitive Touchscreen	1	✓	RS	899-7466	€ 80.50	✓	✓	Mouser
A05003	ADAPTER CABLE	Raspberry Pi5 Display Cable	1	✓	Farnell	4263053	€ 0.91	✓	✓	Mouser
A00000	A. AMR	High-Level Assembly & Miscellaneous								
A00001	KEYBOARD AND MOUSE	Standard AZERTY wireless keyboard & mouse set (France)	1	✓	RS	238-0619	€ 21.25	✓	✓	FARNELL
A00002	USB A TO USB B	Câble USB RS PRO USB A vers USB B, 0.5m, Noir	2	✓	RS	182-8547	€ 4.59	✓	✓	Reichelt
A00003	USB A TO USB C	Câble USB RS PRO USB C vers USB A, 0.5m	1	✓	RS	236-9157	€ 4.60	✓	✓	DIGIKEY

Appendix D: Arduino Code for Controlling Motors from Raspberry Pi

```
//Declare names of pins
int RMotorOn = 2;
int RMotorDir = 3;
int RMotorTorque = 4;
int MotorSpeed = 5;
int LMotorOn = 8;
int LMotorDir = 9;
int LMotorTorque = 10;
float speedDutyCycle = 0.2; //slowed down for testing

void setup()
{
  Serial.begin(9600);
  pinMode(RMotorOn, OUTPUT); // Right Motor On/Off
  pinMode(RMotorDir, OUTPUT); // Right Motor 1 Direction
  pinMode(RMotorTorque, OUTPUT); // Right Motor 1 Torque
  pinMode(MotorSpeed, OUTPUT); // Motor Speed
  pinMode(LMotorOn, OUTPUT); // Left Motor 2 On/Off
  pinMode(LMotorDir, OUTPUT); // Left Motor 2 Direction
  pinMode(LMotorTorque, OUTPUT); // Left Motor 2 Torque
  //pinMode(11,OUTPUT); // Motor 2 Speed. Comment out if both motors run
off the same circuit for speed control

  //Set initial speed to zero
  analogWrite(MotorSpeed, 0);
  //Enable motors
  digitalWrite(RMotorOn, HIGH);
  digitalWrite(LMotorOn, HIGH);
  //Define initial direction
  digitalWrite(RMotorDir, HIGH);
  digitalWrite(LMotorDir, HIGH);
  //Torque
  analogWrite(RMotorTorque, 0);
  analogWrite(LMotorTorque, 0);
}

void loop()
{
  //Check if user has inputted a new command
  if (Serial.available())
  {
    switch (Serial.read())
    {
      //Logic based on different keystrokes - uses french keyboard layout
      case 'a': Serial.println(F("Left"));//Serial print for
troubleshooting
        //Direction
        digitalWrite(RMotorDir, HIGH); //Combination of directions found
through testing
        digitalWrite(LMotorDir, HIGH);
        //On
        analogWrite(MotorSpeed, (speedDutyCycle * 255));
        break;

      case 'd': Serial.println(F("Right"));
        //Direction
        digitalWrite(RMotorDir, LOW);
        digitalWrite(LMotorDir, LOW);
        //On
        analogWrite(MotorSpeed, (speedDutyCycle * 255));
```

```
        break;

      case 'w': Serial.println(F("Forward"));
        //Direction
        digitalWrite(RMotorDir, HIGH);
        digitalWrite(LMotorDir, LOW);
        //On
        analogWrite(MotorSpeed, (speedDutyCycle * 255));
        break;

      case 's': Serial.println(F("Backward"));
        //Direction
        digitalWrite(RMotorDir, LOW);
        digitalWrite(LMotorDir, HIGH);
        //On
        analogWrite(MotorSpeed, (speedDutyCycle * 255));
        break;
    }
    //Move for this period of time before stopping
    delay(1000);
    //Stop AMR
    analogWrite(MotorSpeed, 0);
  }
}
```