Klison Gashi, Rejin Jacob Vadukkoot, Muhammed Daniyal Syed, Mohan Shanmugavel

Smart Hangar Robotics Mobility – The Localisation Challenge

School Of Aerospace, Transport and Manufacturing
Robotics

MSc
Academic Year: 2023 - 2024

Supervisor: Dr Angelos Plastropoulos
Associate Supervisor: Dr Amaka Adiuku
April 2024

This thesis is submitted in partial fulfilment of the requirements for the degree of MSc Robotics

Academic integrity declaration

I declare that:

- the thesis submitted has been written by us alone.
- the thesis submitted has not been previously submitted to this university or any other.
- that all content, including primary and/or secondary data, is true to the best of my knowledge.
- that all quotations and references have been duly acknowledged according to the requirements of academic research.

I understand that to knowingly submit work in violation of the above statement will be considered by examiners as academic misconduct.

Abstract

A pre-flight check is a visual inspection which is completed prior to every departure of an aeroplane, to ensure it is fit for travel. This is a time-consuming procedure; therefore, automation could significantly reduce both time and costs associated with pre-flight checks. The main theme of the project was to contribute to the development and use of smart hangars, which are hangars outfitted with sensors and systems to allow autonomous systems to be implemented in various aircraft operations. In this study, our main challenge was localising in a featureless environment. To address this challenge, we successfully programmed a mobile robot to traverse the path of a pre-flight check in simulation by employing various ROS packages for navigation, localisation and map generation. A dummy camera was implemented to simulate the inspection process being completed to demonstrate the feasibility of the localisation and navigation. Potential areas of further work were also discussed.

Keywords:

Navigation

SLAM

ROS

Gazebo

Acknowledgements

We would like to thank our supervisors, Dr. Plastropoulos, Dr. Adiuku and Dr Fan, for their guidance and support throughout this project. We would also like to thank some of the students of the Digital MRO course, Tyron Sequeira, Nimisha Nagarkar and Dineshkumar Prabhakaran, for providing their assistance and literature on MRO and pre-flight checks. Finally, we would like to thank Dr Gilbert Tang for his assistance and guidance both during this project and throughout the year.

Table of Contents

Academic integrity declaration i
Abstract ii
Acknowledgements iii
List of Figures vi
List of Tables vii
List of Abbreviations viii
1 Introduction and Literature Review 9
1.1 Project Overview 9
1.2 Aircraft Inspection 10
1.3 Sensors 12
1.4 Simultaneous Localisation and Mapping 14
1.5 Localisation in a Featureless Environment 15
1.6 ROS Packages 16
1.6.1 SLAM Toolbox 17
1.6.2 Cartographer 17
1.6.3 LeGO-LOAM 17
1.6.4 RTAB-MAP 17
1.6.5 HDL Graph SLAM 18
1.6.6 Loop Closure Detection 18
1.6.7 AMCL 19
1.7 Navigation 20
2 Objectives 21
3 Methodology 22
3.1 Robot Platforms 22
3.2 DARTeC Virtual Environment Creation 25
3.3 Pre-Flight Check 28
3.4 Contingency Plan 28
3.5 Experimental Procedure and Setup 29
3.5.1 Assumptions 29
3.5.2 Inspection Modelling 30
3.5.3 2D Mapping 31
3.5.4 3D SLAM Algorithm Selection 33
3.5.5 Localisation and Navigation Algorithm Selection 34
3.5.6 3D Mapping 34
3.5.7 2D Navigation 36
3.5.8 3D Navigation 37
4 Experimental Results and Analysis 38
4.1 2D Mapping and Navigation 38
4.2 3D Mapping and Navigation 40
4.3 Feasibility Analysis 40

4.4 Challenges 41
5 Conclusion 43
5.1 Overview 43
5.2 Further Work 43
References 46
Appendices 50

List of Figures

Figure 1 Flow chart of conducting an automated inspection process 10

Figure 2 Overview of the Nav2 design and structure.. 21

Figure 3 Husarion Panther [28] 22

Figure 4 Image of TurtleBot stuck because of DARTeC's uneven terrain [29] . 24

Figure 5 Image of DARTeC hangar [30]... 25

Figure 6 CAD Model of the DARTeC Hangar and surrounding environment ... 25

Figure 7. DARTeC World with the fences removed for efficiency..................... 26

Figure 8. Virtual Waypoints added to the world file 27

Figure 9. DARTeC world with the static obstacles.. 27

Figure 10 Typical pre-flight check path around an aeroplane........................... 28

Figure 11 Image of the camera FOV for modelling purposes as seen during the simulation 30

Figure 12 Flowchart for 2D mapping 31

Figure 13 2D Map Generated using 30cm Height Setting 32

Figure 14 2D Map generated using the 4m height setting................................ 33

Figure 15. Flowchart for 3D mapping 35

Figure 16 Generated 3D Map of DARTeC environment................................... 35

Figure 17 Flow chart for 2D Navigation. 36

Figure 18 Flow chart for 3D mapping procedure 37

Figure 19 Shift in map representation – 30cm.. 38

Figure 20 Before shift in map representation – 4 m... 39

Figure 21 Shift in map representation – 4 m... 40

List of Tables

Table 1 Comparison table of the TurtleBot and Panther to determine suitability for use in aircraft inspection 23

Table 2 Sensors included in the Panther used for this project 24

Table 3 Comparison table for different ROS SLAM Algorithms 34

List of Abbreviations

MRO	Maintenance, Repair and Overhaul
GPS	Global Positioning System
DARTeC	Digital Aviation Research Technology Centre
UAV	Unmanned Aerial Vehicle
LiDAR	Light Detection and Ranging
IMU	Inertial Measurement Unit
iGPS	Indoor Global Positioning System
UGV	Unmanned Ground Vehicle
EKF	Extended Kalman Filter
UKF	Unscented Kalman filter
UWB	Ultra-Wide Band
SLAM	Simultaneous Localisation and Mapping
vSLAM	Visual SLAM
PF SLAM	Particle Filter SLAM
GBF SLAM	Grid-Based Fast SLAM
ROS	Robot Operating System
AMCL	Adaptive Monte Carlo Localisation
LeGO-LOAM	Lightweight and Ground Optimised Linear Odometry and Mapping
LOAM	Linear Odometry and Mapping
RTAB-MAP	Real-Time Appearance-Based Mapping
RGB-D	Red-Green-Blue-Depth
DoF	Degrees of Freedom
HDL	High-Definition LiDAR
MCL	Monte Carlo Localisation

1 Introduction and Literature Review

1.1 Project Overview

Robots are playing an ever-increasing role in the aerospace industry, with various operations nowadays being completed with some degree of autonomy. One unconquered region of research is the development of a robot capable of carrying out some form of maintenance autonomously. The term maintenance covers a variety of procedures, one of which being the pre-flight check. This is a visual inspection done before every flight to check for damage in key areas of the aircraft, which is a slow and tedious task. There is the potential of automating this task in numerous ways.

One of the ways this can be completed is by employing a mobile robot platform with onboard sensors to be traverse the path of the pre-flight check and inspect the plane using a camera or other device. This overall task can be broken down into various steps: mapping the environment; traversing and navigating the path through localisation; inspecting the aircraft; and understanding and processing signs of damage. In this project, we aim to solve a few of these challenges, namely the mapping and traversing. This process can be seen in Figure 1 below.

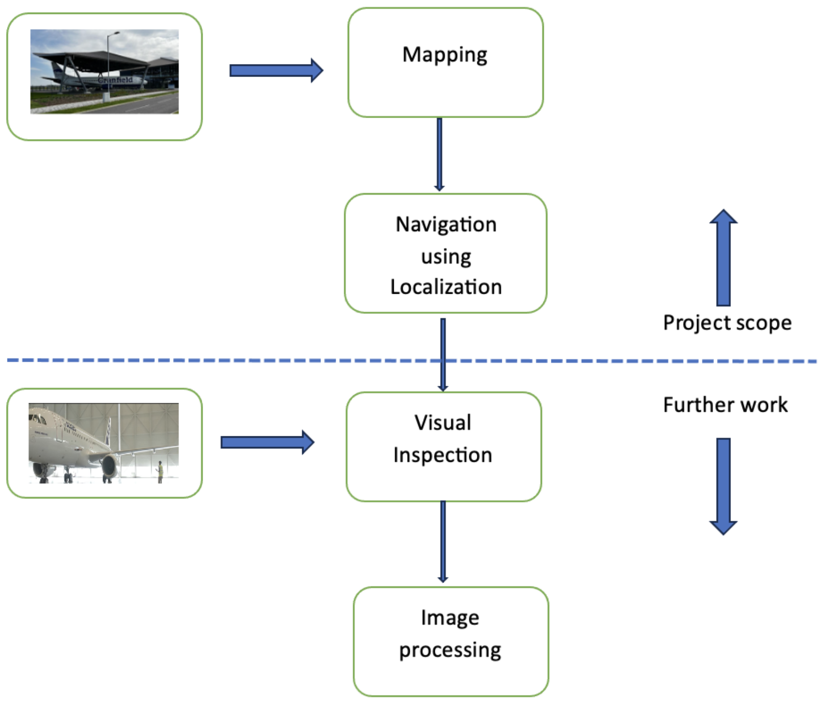


Figure 1 Flow chart of conducting an automated inspection process

Localising the robot can prove difficult in an environment with sparse features, such as a hangar. Therefore, the main challenge, as indicated by the title, is to solve the localisation challenge associated with autonomous robots operating within a hangar. We aim to provide experts in the maintenance field with a robotic solution to solve the localisation and locomotion aspect of conducting an autonomous pre-flight check, with the potential for the solution to be incorporated into other autonomous procedures within the aerospace industry.

1.2 Aircraft Inspection

The aviation industry is governed by strict rules of compliance, with technical standards shaping the production and maintenance processes. Airline operators are required to carry out pre-flight checks before every air departure to ensure that the aircraft is suitable for travel [1]. Maintenance, Repair and Overhaul (MRO) operations contribute significantly to aircraft operating costs, delays and cancellations. This in turn means that airlines spend more on maintenance than fuel or crew, despite improvements in aircraft lifespan and engine technology [2]. Therefore, automation of the MRO process is a key factor in reducing costs and increasing the efficiency of the entire process. One MRO operation is a pre-flight check, which is conducted before and after every flight to ensure the aircraft is in optimal condition for flying. The focus of this paper is the pre-flight inspection check, which is an exterior visual inspection of the aircraft, to ascertain whether there are any visible signs of damage, such as dents, punctures and scratches. A specific path is navigated by a member of maintenance personnel, which the key areas inspected being the wings, control surfaces, the engine and propellors, the landing gear, and the lights and antennas [3].

Automation of this process has seen in increase in recent years, following the development and advancement of autonomous navigation, localisation and machine vision algorithms, as well as from the rise in computational power and efficiency. To effectively automate the process, the hangar requires an upgrade, typically done by adding a multitude of different sensors. Some examples of sensors which can be added are cameras, depth sensors and global positioning systems (GPS), which can be fused with the robot's onboard systems to effectively localise the robot. These upgrades often constitute what is known as a "smart hangar", as discussed in [4]. In future, these hangars will have the capability to guide the coordination of a fleet of automated vehicles. The smart hangar in question utilised this project is Cranfield University's Digital Aviation Research and Technology Centre, or DARTeC, which hosts various sensor technologies. The environment within the DARTeC hangar is notably featureless, lacking discernible walls or prominent features which typically aid navigation and localisation, the most prominent structure being the aircraft. This is a non-permanent feature given that different aircraft utilise the hangar, which proposes challenges with defining a fixed reference point.

There are existing solutions to the issue of autonomous hangar inspection. One such solution is the use of Unmanned Aerial Vehicles (UAVs), as discussed in [5]. The two onboard sensors, LiDAR and Inertial Measurement Unit (IMU), were fused together along with an iGPS system via the Kalman Filter, allowing for the conglomeration of local and global systems for accurate localisation. The results indicated a high level of accuracy, but suffered from blind spots in the iGPS which caused a deviation in the results, sometimes up to 2m. UAVs have issues associated with the, such as typically having a poor battery life. Another issue stems from the discomfort of human operators in the area having drones flying around near head height, posing safety concerns. An Unmanned Ground Vehicle (UGV) would be able to overcome these issues, by holding a larger battery or using mains supplied power and by remaining on the ground. However, there are potential concerns that a ground-based vehicle will not be able to effectively inspect the engine given its height off the ground and concave nature.

1.3 Sensors

Sensors are of paramount importance for localisation and navigation, both within robotics and other fields. However, they often provide noisy measurements, which can cause deviations from the true state of environment or object in question and introduce unwanted effects. The Kalman filter is an algorithm, proposed by R.E. Kalman in [6], that is used as an optimal observer for filtering sensor data. A mathematical model is employed to create a prediction for the subsequent timestep based on a state space representation of the system. This is used in conjunction with the sensor data to provide a more accurate representation of the system's state than just the sensor alone. The underlying assumptions are that the error in the sensor data follows a Gaussian distribution, and that the system is linear. However, in many cases, such as in this paper, there are non-linearities in the dynamics of the system and the measurements. This requires an alternative form of the Kalman filter known as the Extended Kalman Filter (EKF). In this algorithm, the first-order nonlinear Taylor expansion of the estimation is used and thus transforms the nonlinear system into a linear equation. This decreases the error seen from the linearities of the Kalman Filter. If the initial state estimate is incorrect, however, the EKF may quickly diverge due to linearisation [7].

The Unscented Kalman Filter (UKF) is a variant of the Kalman Filter which also addresses the challenges associated with non-linear state estimations. However, unlike the EKF, the UKF employs a deterministic sampling technique to approximate both the mean and covariance of the state distribution, where a set of representative sigma points is selected. This technique accounts for the non-linearities more accurately than the EKF, which makes it especially suitable for systems suffering from non-Gaussian noise [7].

Kalman Filters are often used in conjunction with sensor fusion architectures. Sensor fusion is the process of combining sensor data to gain a more accurate understanding of the state of a system by providing redundancy [8]. The most common approach to sensor fusion is centralised fusion, where data is processed in a “fusion centre”. The group sensor model is a centralised fusion model where sensors which are observing the same variable are grouped together. A composite observation is taken, where the measurements and noise are each combined into a vector. There are three types of sensor fusion architectures: complementary, competitive and cooperative. Complementary sensors provide non-overlapping information to provide a “bigger picture” to be formed. Competitive sensors provide information on the same subject, which has the potential for conflicts to occur. Cooperative sensors are similar to competitive sensors, except that they typically use differing sensor types. Using sensor fusion allows for a decrease in computation and storage requirements, given that the data is processed, filtered and fused, rather than processing various individual sensor observations [9]. Given that this paper is focusing on localisation, cooperative sensing is the most appropriate given the multitude of differing sensor types available on both the Panther robot and in the smart hanger.

Many different types of sensors can be utilised for autonomous navigation, both internal and external to the robot. Ultra-Wide Band (UWB) is a wireless communication technology for short-range communication which can be used as an external, global positioning system for localisation and is available in the smart hangar. The transmission is completed over a wide range of frequencies, which allows for precise positioning and resistance to interference [10]. Regarding positioning, the working principle is that a tag is attached to the robot which serves as a type of beacon. The beacon sends a signal into the environment, which is detected by the anchors. The anchors are stationary receivers that serve as a positional reference point. Once a signal is received by the anchors, a time of flight is taken to find possible locations of the robot. The time of flight provides a circle of possible locations, from each anchor. The overlap of these circles is then found to identify the location. Due to noise and inaccuracies, a small region is identified as the location of the robot rather than a single specific point. This can then be processed using a Kalman filter to provide a more accurate positional estimate. [11]

Light Detection and Ranging, or LiDAR, is a type of sensor which generates 3D information using the reflection of a laser off an object and finding the time of flight on the return. The information is in the form of a range, i.e., distance, and bearing angle, thus constituting a polar representation of the location [12]. The information collected by the LiDAR can be used to generate a point cloud, which is a series of points containing angle and distance information.

1.4 Simultaneous Localisation and Mapping

Simultaneous Localisation and Mapping, or SLAM, is an algorithm which solves the dilemma of requiring localisation to generate a map, while conversely requiring a map to localise. Landmarks are defined and the distance from the landmarks is extracted at each timestep. LiDAR SLAM, as the name suggests, uses a LiDAR sensor to accomplish localisation and mapping. Features can be extracted by using a point cloud, from which consecutive point clouds can be compared to find the relative motion distance, thus achieving localisation. LiDAR SLAM is preferred over visual SLAM (vSLAM) since it is not easily affected by environmental conditions, such as light [13]. This poses significant challenges in the case of the smart hangar, which is an outdoor environment and thus would experience adverse environmental conditions which could diminish the accuracy.

Another SLAM algorithm which is often used is EKF SLAM utilises the aforementioned EKF algorithm in collaboration with SLAM, thus incorporating sensor fusion. Information on the robot pose and the landmark locations are stored within a high dimension state vector. Initially, the state vector is predicted using control vectors in combination with a motion model. Then, the state vector is updated using the landmark observations via the observation model [14].

An alternative SLAM algorithm is Particle Filter (PF) SLAM, which is a probabilistic technique. The robot's state is represented using a set of particles, where each particle signifies a potential hypothesis of the robot's pose and the features of the map. While the robot moves, the particles are updated using a prediction which is based on a motion model – which accounts for uncertainties in the robot's motion. Subsequently, sensor measurements are integrated to adjust the particles' weights, with higher weightings applied to the particles which appear to align with the observed data. The particles are redistributed via resampling and the lower weighted particles are discarded. The robot's trajectory is refined by repeating this process at every time step. A technique which can improve the accuracy is loop closure detection, which recognises locations which were already visited. PF SLAM is typically robust, particularly regarding non-linearities and uncertainties, but has a high computational demand [15].

EKF and PF SLAM can be combined to create FastSLAM, where the particle filter is used for the vehicle states and the EKF is used to track the landmark positions. However, unlike PF SLAM, FastSLAM factorises the posterior distribution i.e., representing the robot's pose and the map as separate objects. This allows the algorithm to update the robot's pose using a low-dimensional state vector, thus allowing for a lower computational power consumption. Furthermore, FastSLAM employs a technique called Rao-Blackwellisation, where the map estimate is marginalised out. This reduces the map estimate to a lower dimensional vector also, thus enhancing computational efficiency but maintaining high accuracy [16]. An extension of FastSLAM is Grid-Based FastSLAM (GBFSLAM), which utilises a grid-based representation of the environment for increased efficiency. The environment is divided into a grid of cells, where each cell is associated with a probability distribution that represents the likelihood that the grid is occupied, which simplifies the map representation and therefore further reduces the computational power required.

1.5 Localisation in a Featureless Environment

One of the challenges of localisation is how to localise when presented with a featureless or limited feature environment. Given that SLAM typically works using landmark identification, it can be difficult to successfully implement it. One method, as discussed in [17], is the use of EKF with LiDAR, combining all the onboard sensors into one system. The LiDAR scan is used to identify any features which are present. Having less landmarks leads to a lower accuracy, and thus other sensors are employed to better localise the robot. In [17], the researchers used an IMU, a magnetometer with an odometer, and a LiDAR. This algorithm allowed for a positioning error of 0.43m, compared to 5.7m and 1.67m for 2 alternative algorithms respectively.

An alternative novel method is discussed in [18]. The approach used is LiDAR-assisted UWB positioning, where a combination of local and global sensors is employed. An EKF algorithm is applied to both the UWB measurements, giving the position, and the LiDAR measurements, which supply a change in position or delta position. The fusion is applied to the sensors before combining and therefore constitutes a group sensor model. This combination, as opposed to UWB alone, allowed for an approximately 42% decrease in positional error, therefore demonstrating the superiority of combining differing sensor types and using an amalgamation of global and local sensor systems [18]. LiDAR assisted UWB, with incorporates EKF, is the most suitable for the proposed research problem given that the measurements taken from LiDAR, i.e., range and bearing, are nonlinear in nature, thus requiring EKF. Furthermore, LiDAR is the principal sensor on the robot, and DARTeC is equipped with UWB sensors. The odometry systems of the Panther can also be utilised given that the EKF is being employed.

1.6 ROS Packages

ROS, or Robot Operating System, is a framework for developing robot software. It contains a comprehensive set of software libraries and tools for various robot applications, such as device drivers and communication. It is beneficial for this project given that it offers a variety of different packages for localisation and mapping algorithms which can be used for both simulation and live use. For the case outlined in this project, both 2D and 3D mapping will be used. Some algorithms, which were outlined in [19], were considered and described below. Once the map has been created, the robot must use localisation and navigation algorithms to successfully and accurately locomote to the specified waypoints. This is the primary focus of this project since the issue we are attempting to conquer is localisation in a featureless environment.

1.6.1 SLAM Toolbox

One package available is The SLAM Toolbox [20] is a mapping algorithm which can be used with a technique called Adaptive Monty Carlo Localisation (AMCL) – described later – to accurately map and localise. The SLAM Toolbox was created in response to the accuracy deficiencies present in other mapping and localisation packages in large, dynamic indoor environments. However, it is only available for 2D mapping.

1.6.2 Cartographer

Cartographer [21] is an alternative package that can create both 2D and 3D maps. It fuses various sensor inputs, some of which being LIDAR and the IMU, to perform SLAM. Cartographer is particularly known for its accurate mapping capabilities.

1.6.3 LeGO-LOAM

A different method available is Lightweight and Ground Optimised Linear Odometry and Mapping, commonly abbreviated as LeGO-LOAM [22], is an efficient and accurate method developed as an extension of LOAM. The difference between the algorithms is the optimisation of the process to increase efficiency and robustness. Noticeably, it is a lightweight design, but maintains high accuracy and reliability by using loop closure detection.

1.6.4 RTAB-MAP

RTAB-MAP [23], or Real Time Appearance-Based Mapping, is primarily a Red-Green-Blue-Depth (RGB-D) based approached, which utilises a global loop detector. When a loop closure is detected, the map graph has a new constraint added, so that the graph optimiser can then minimise the errors seen in the map. In order to limit the number of locations used for loop closure – for optimisation and computational efficiency – a memory management approach is used, so that real-time constraints in large-scale environments are addressed. This approach means that it is suitable for a hangar environment. Furthermore, RTAB-MAP allows for 6 DoF mapping when used with a 3D LiDAR.

1.6.5 HDL Graph SLAM

The final algorithm considered for use is High-Definition Lidar Graph SLAM, or HDL Graph SLAM [24] as it is commonly abbreviated. It works by creating a detailed 3D point cloud and constructing a graphical representation of the environment. Within this graph exist nodes that represent the robot poses and edges. By optimising the graph, the algorithm both refines the estimate of the location and updates the map, simultaneously. It is known to be particularly effective in situations requiring high precision and reliability, because, like LeGO-LOAM, it uses loop closure detection.

1.6.6 Loop Closure Detection

Loop closure detection, discussed in [25], refers to the detection of instances when the robot visits a previously visited location. By comparing the current sensor data with the data collected when the location was previously visited, the accumulated error in the robot’s position estimate can be corrected. This process is crucial for improving the accuracy of the map and pose estimate. Various techniques are employed to complete loop closure detection, including feature-, appearance- and graph-based methods. Feature-based methods extract distinctive features from the sensor data, such as laser scan matching. Appearance-based methods use visual information to detect loop closures, often using image retrieval or place recognition. Graph-based methods represent the robot’s trajectory as a graph, with nodes corresponding to robot poses and edges which represent constraints between poses. Once a loop closure it detected, the SLAM system can perform correction, which adjusts the estimated trajectory and map to reduce any accumulated errors. This correction typically involves optimising the entire trajectory to ensure consistency is achieved.

This technique is beneficial for long-term mapping and localisation when a robot may revisit certain locations over a long period of time, such as in the case of a pre-flight check, given its robustness and accuracy, as well as its ability to handle environments with significant changes.

1.6.7 AMCL

One of the main available methods for pure localisation when a map has already been generated is the Adaptive Monte Carlo Localisation, (AMCL) technique, as discussed in [26]. AMCL is an extension of the Monte Carlo Localization (MCL) algorithm, which uses a particle filter to represent the robot's pose probability distribution. Unlike traditional localization methods that rely on a static map, AMCL can adapt to changes in the environment and handle uncertainties more effectively. AMCL utilises a set of particles, each representing a hypothesis of the robot's pose. These particles are randomly distributed throughout the map and are assigned weights based on how well they match sensor measurements. During the prediction step, the particles are moved according to the robot's motion model. Then, in the update step, the weights of particles are adjusted based on sensor measurements, such as from the laser scans or camera data.

What makes AMCL adaptive in comparison with the MCL is its ability to adjust the number of particles and their distribution as the robot moves through the environment. This adaptation ensures that computational resources are allocated efficiently, focusing more particles in areas of higher probability and reducing their number in less likely regions. By doing so, AMCL can handle changes in the environment effectively, such as dynamic obstacles or changes in landmark positions.

One of the key challenges in AMCL is the data association problem, where the algorithm must determine which sensor measurements correspond to which landmarks in the map. To address this, AMCL employs techniques such as the nearest neighbour approach or the use of clustering algorithms to associate measurements with map features accurately.

1.7 Navigation

Building a map is only half the challenge in autonomous locomotion; traversing the path has many challenges. Once the map has been created, a path must be both generated and followed. The latter uses the abovementioned localisation techniques. However, path generation, commonly known as path planning, requires a different approach and system. Path planning can be separated into two categories: local and global. Global path planning is simply planning a path from the robot's current location to a defined waypoint. Local path planning is often used for situations where the global path cannot be followed, such as avoidance of unexpected obstacles. This scope of this project only includes static obstacle avoidance.

By far the most popular navigation package used in ROS2 is the Nav2 package [27]. It uses a plugin-based architecture to do path planning and following, utilising both global and local planning. For global planning, Nav2 supports plugins such as the A* algorithm, Dijkstra and potential fields, which can be used to generate the path from the robot's current pose to the goal pose. For local path planning, Nav2 offers plugins such as the Dynamic Window Approach, Timed Elastic Band, and Model Predictive Control. The package is designed used a behaviour tree structure, Figure 2, where different internal severs are used for nodal communication between the various aspects of the navigation stack, such as the controller, planner and cost map.

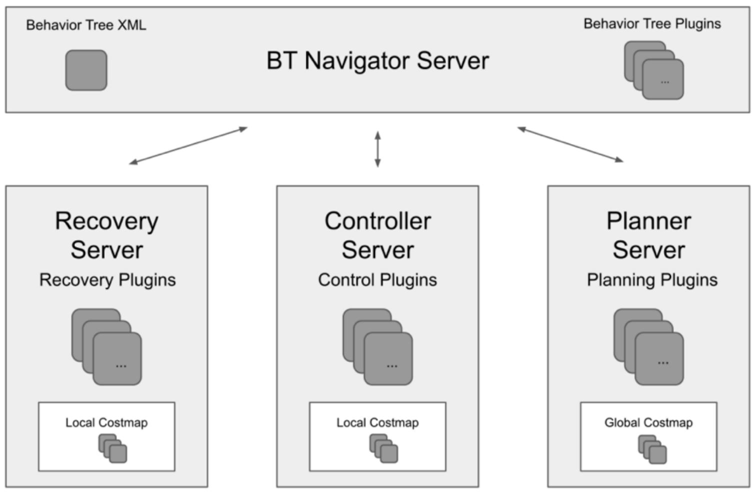


Figure 2 Overview of the Nav2 design and structure

2 Objectives

The success of this project will be measured against the following objectives:

1. Address the challenge of localisation in an open location with little to no landmarks by reviewing state of the art literature. Existing solutions will be analysed and critiqued, with the best-fit solutions incorporated within the project.

2. Create a simulation environment for validation, tuning and proof of concept purposes; initially with the Turtlebot3, and then with the Husarion Panther robot. A simple layout was initially used to better understand the Gazebo environment, after which the 3D environment generated by the previous group was used to test and validate.

3. Utilise sensor fusion and navigation algorithms found in literature and ROS to successfully navigate the virtual environment.

4. Successfully navigate the proposed path with a high level of accuracy, preferably remaining within 0.5m of the path at all times. This should also serve as a platform for future research, specifically for inspection capabilities to be integrated into the real-world system to successfully automate the pre-flight inspection process.

3 Methodology

3.1 Robot Platforms

For initial testing, the TurtleBot3 Waffle Pi was employed. This was done to gain valuable, hands-on experience with ROS and SLAM. However, the primary robot platform used for this project is the Husarion Panther, Figure 3, which can come with various options and features depending on use.

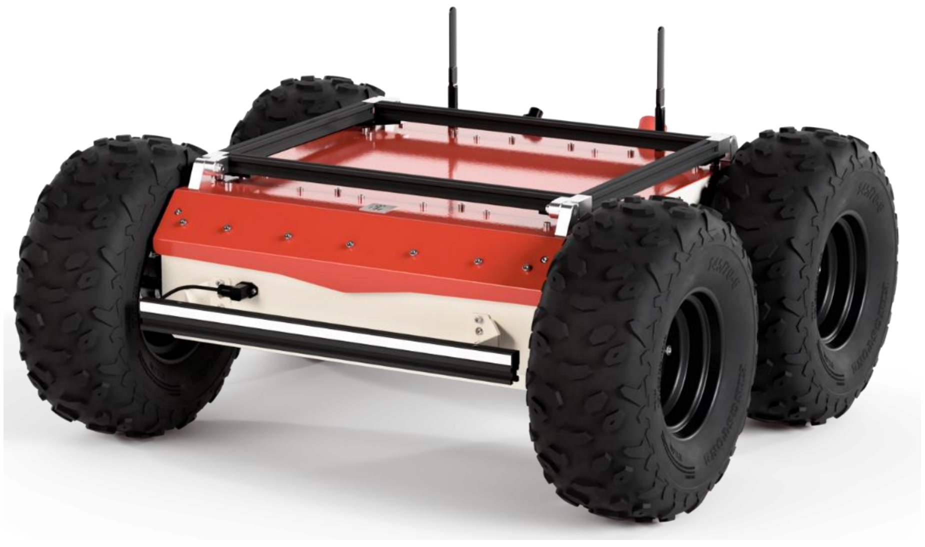


Figure 3 Husarion Panther [28]

The specific Panther being used for this project has a multitude of sensors, including a 3D LiDAR sensor with a 50m range and a 90-degree vertical field of view; an inertial navigation system consisting of a 3-axis gyroscope, compass, and accelerometer; and a ROS API. This allows for simple programming and interfacing in multiple programming languages, with the most support available being for Python and C++. Table 1 below contains a comparison of the TurtleBot and Panther.

Table 1 Comparison table of the TurtleBot and Panther to determine suitability for use in aircraft inspection

Feature	TurtleBot	Panther
LiDAR Range	8m	50m
LiDAR Accuracy	Up to ± 40cm	± 1.5 - 5cm
LiDAR Type	2D	3D
Scan Frequency	5 Hz	10 or 20 Hz
Processing	Raspberry Pi	Inter Core i7
Connection	Single Antenna	Dual Antenna
Travel Speed	0.26 m/s	1.94 m/s
Battery Life	Up to 2 hours	Up to 8 hours
Wheel Diameter	66mm	370mm

From Table 1, it is evident that the Panther has a vastly superior LiDAR, with a higher accuracy, range and dimensional capability. The use of the 3D LiDAR is essential in localisation of the DARTeC environment, given that the ground level features are sparse and spaced out. The increased travel speed and battery life allow for a quicker execution of the task and is thus a more feasible use case. A notable issue faced by the previous research group, as discussed in [29], was locomotion on the uneven terrain, as seen in Figure 4 below. The vastly increased wheel diameter allowed the Panther to navigate the terrain more easily. Additionally, the previous group faced the issue of having to follow the TurtleBot around with the computer due to the poor communication range. The Panther does not have this issue because of the dual antenna and extended range and power supply. Table 2 shows the list of available sensors onboard on panther.

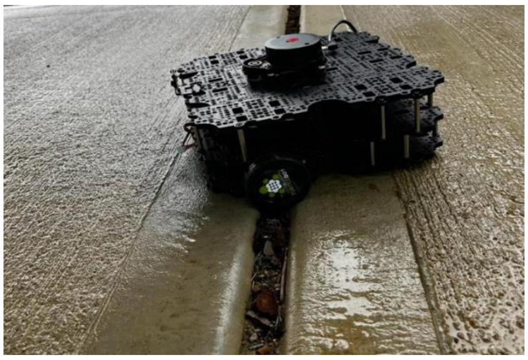


Figure 4 Image of TurtleBot stuck because of DARTeC's uneven terrain [29]

Table 2 Sensors included in the Panther used for this project

Code	Description	Details
CAM04	ZED 2i Stereo Camera (2.1mm, with polarizer)	RGBD camera
LDR10	Ouster OS0-32	UltraWide View LIDAR
ANT02	Antenna 003R-00253 (WiFi + LTE + GPS)	
IMU	PhidgetSpatial 3/3/3 Basic (3-axis compass, a 3-axis gyroscope, and a 3-axis accelerometer	Inertial Measurement Unit

3.2 DARTeC Virtual Environment Creation

Cranfield's DARTeC, Figure 5, is a state-of-the-art smart hangar which is used for research purposes to address challenges being faced by the aerospace industry. One key area is increasing the efficiency of airports through technological advances [30], which includes MRO. A noticeable characteristic of DARTeC is the lack of walls and small number of features, making it an outdoor hangar. This presents some challenges with using the Panther's LiDAR for SLAM. Although, it is equipped with UWB sensors, allowing for precise localisation to assist the robot with this task. A 3D CAD model was provided of DARTeC was provided, as seen in Figure 6, to be used for simulation and testing.

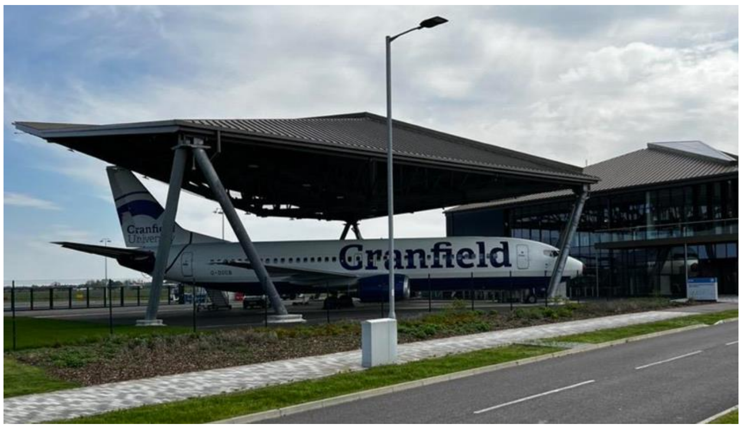


Figure 5 Image of DARTeC hangar [30]

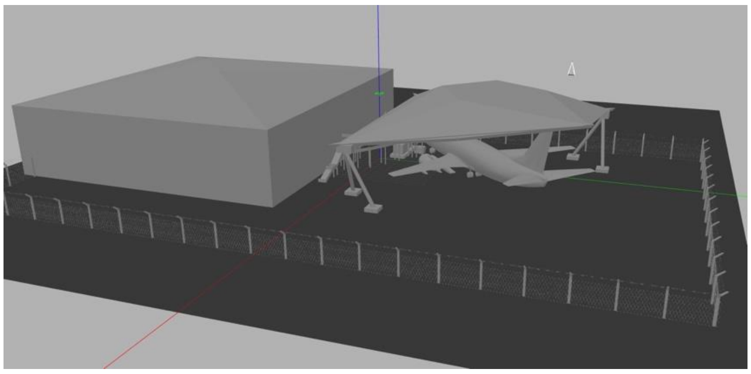


Figure 6 CAD Model of the DARTeC Hangar and surrounding environment

It was noticed that the provide 3D model has a lot of fine features which caused the simulation to load slowly. The majority of the issues were caused by the fences which has fine meshes adding to the latency in terms of loading times. This also caused the simulation clock to be relatively slow when running any simulation. These fences were removed in order to improve simulation performance and cycle times. Figure 7 shows the world file of the DARTeC environment with the fences removed.

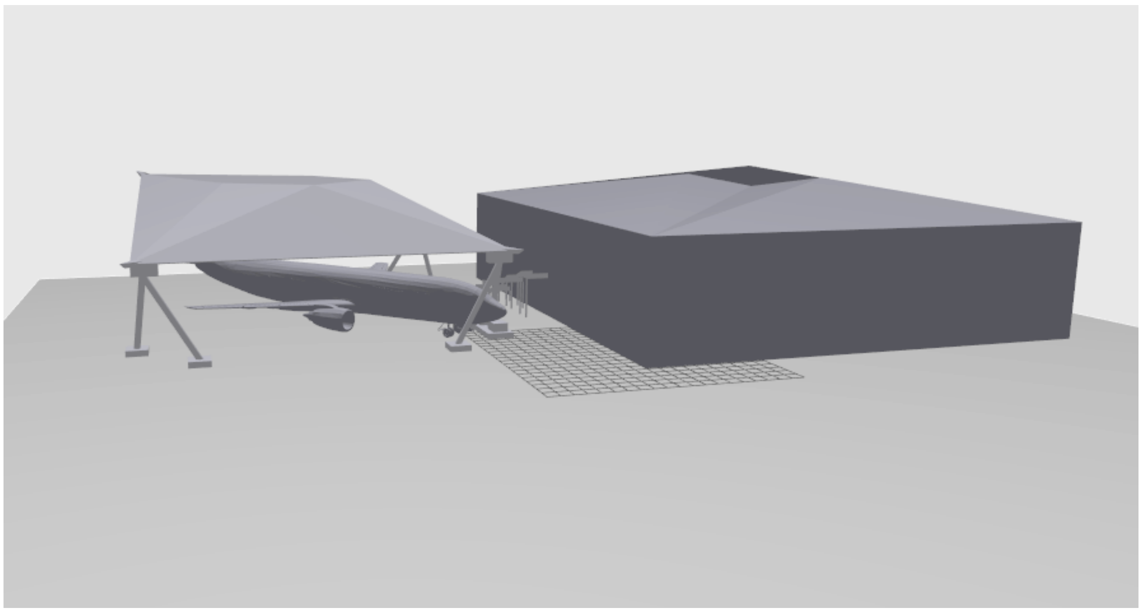


Figure 7. DARTeC World with the fences removed for efficiency

Since the repeatability of the robot during the pre-flight check path had to be accessed, virtual waypoint markers were added onto the world file for better visualization. Figure 8 shows the markers added on to the simulated world. The location of the markers varies from the actual locations for pre-flight check as discussed in the upcoming section. This is to make sure that the part being inspected is in the view of the fixed cameras FOV. This world file was primary used for mapping.

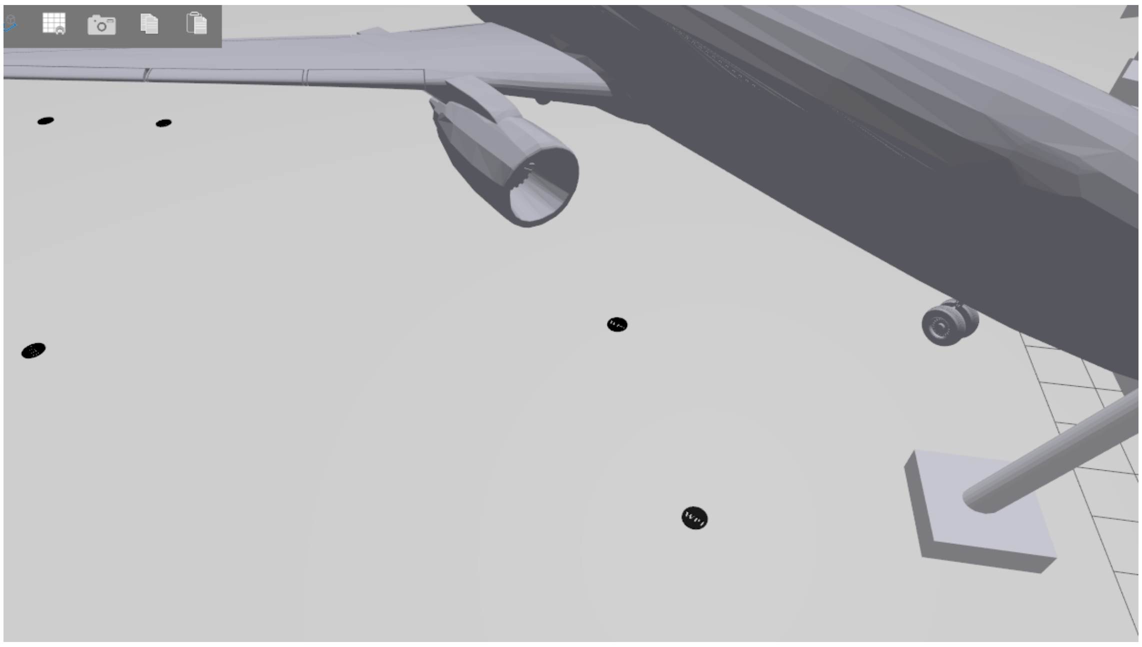


Figure 8. Virtual Waypoints added to the world file

In addition to this, an alternative world file was created with obstacles it. This is the world file which was used for testing various navigation approaches. The environment add static obstacles likes ladders and people near the airplane model. Figure 9 shows a snip of the DARTeC world file with the obstacles added to it.

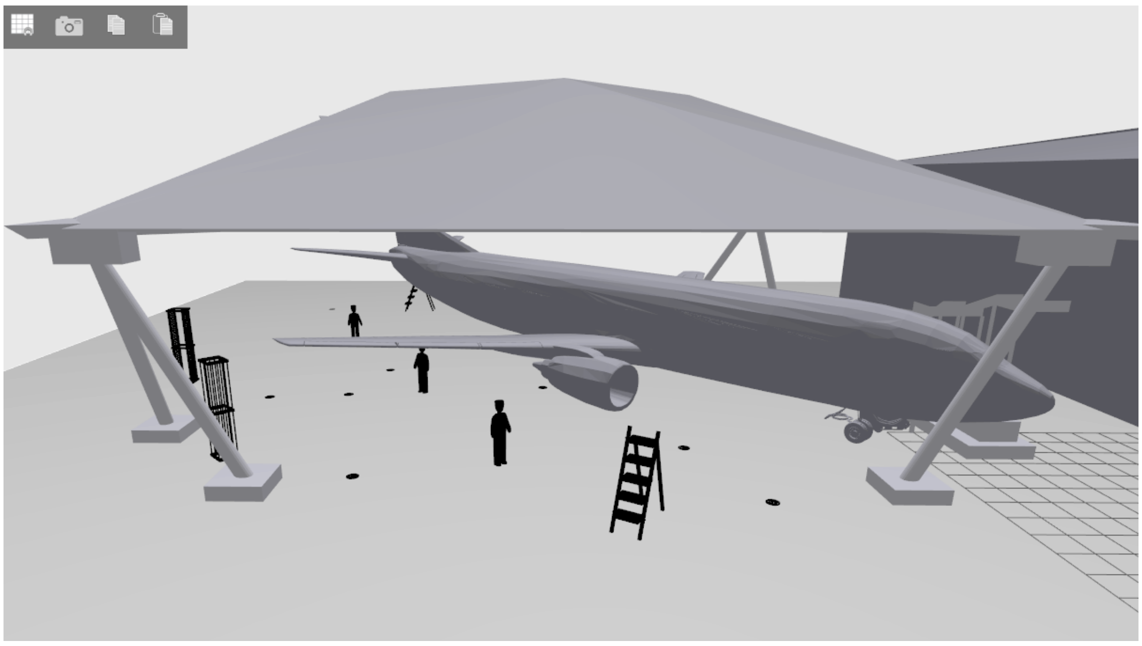


Figure 9. DARTeC world with the static obstacles

3.3 Pre-Flight Check

The pre-flight check, Figure 10, is typically carried out by a member of a maintenance crew. The path is intended to allow the person carrying out the inspection to be able to visually examine key areas of the aircraft for damage, such as the fuselage, engines, wings and control flaps such as the ailerons. Indications of damage could be chips, scratches, punctures and dents present at any location. The checkpoints seen in Figure 10 were implemented as virtual waypoints within the simulation, which the robot will aim to accurately reach.

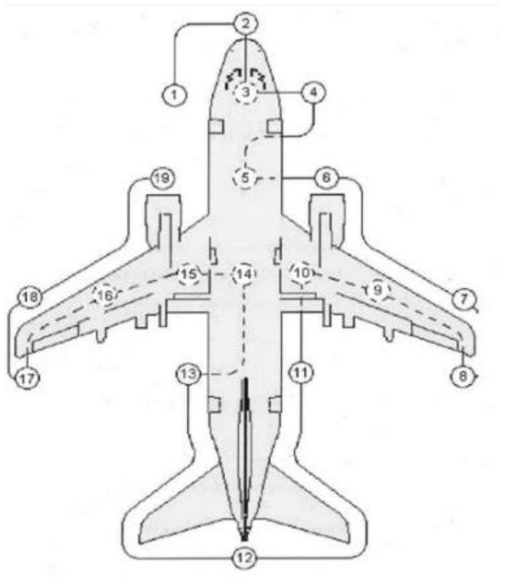


Figure 10 Typical pre-flight check path around an aeroplane

3.4 Contingency Plan

When the planning for project was initiated, it was decided that ROS 2 will be used as the primary platform for development of the project, with ROS 1 as a fall-back option in the event of failure. The reason ROS1 was selected as a backup option is because ROS 1 has been around for a long time and is widely adopted in the robotics community. Moreover, ROS1 has a large ecosystem of packages, libraries and tools, and a mature system with a stable codebase.

Development of the project was across ROS 2 and ROS 1 for initial testing done with the Turtlebot3. ROS2 was the primary focus of the developments as it offered various advantages in terms of the available navigation packages. Even though the support for ROS2 is minimal, porting libraries from ROS1 was comparatively easier, hence ROS2 was decided to be adopted as the primary development platform.

The manufacturers for Husarion Panther provide development workspaces and packages to be used with the physical robot as well as in simulation. At the time of project planning, it was noticed that a ROS2 developmental version was already in the pre alpha release and the support for ROS1 was ending. Since most of the working drivers were in ROS1 it was kept as a backup in case we faced issues with ROS2 during the project development.

3.5 Experimental Procedure and Setup

To access the localisation capability of the robot in a featureless environment two separate approaches were tested.

1. 2D Mapping and Navigation
2. 3D Mapping and Navigation

The Panther robot houses a 3D Lidar, although, a 2D mapping and navigation approach was developed and tested as it allows better efficiency in terms of computations resources being used. This allows for faster and better navigation in case a dynamic obstacle avoidance is implemented in future. In order to use all the available sensors onboard panther an appropriate 3D mapping algorithm was selected and tested. The selection criteria for the 3D mapping algorithm is discussed in the following section.

3.5.1 Assumptions

In order to conduct the experiments on the simulated environment a few assumptions were made:

1. The simulated environment is an accurate representation of the actual world scenario
2. The aeroplane is a fixed feature and will not change i.e., it is permanent
3. All sensors onboard Panther robot represents physical sensors with noise

4. All the obstacles are static
5. The ground surface is flat with no bumps

3.5.2 Inspection Modelling

For this project, the scope and focus were to achieve accurate localisation in a limited feature environment, whereas completing the visual inspection process is a different problem entirely. However, to demonstrate the feasibility of the localisation – by completing the pre-flight check path to a high level of accuracy – a hard-coded camera was implemented as a sensor on the robot. This camera is used for demonstration purposes and does not contain the capability to carry out the inspection and identify damage to the plane. However, adding this camera prompts further work to be done by using the camera with machine vision algorithms to identify problematic areas on the aircraft. The FOV of the camera was also shown during the simulation to imitate the inspection process, as seen in Figure 11 below.

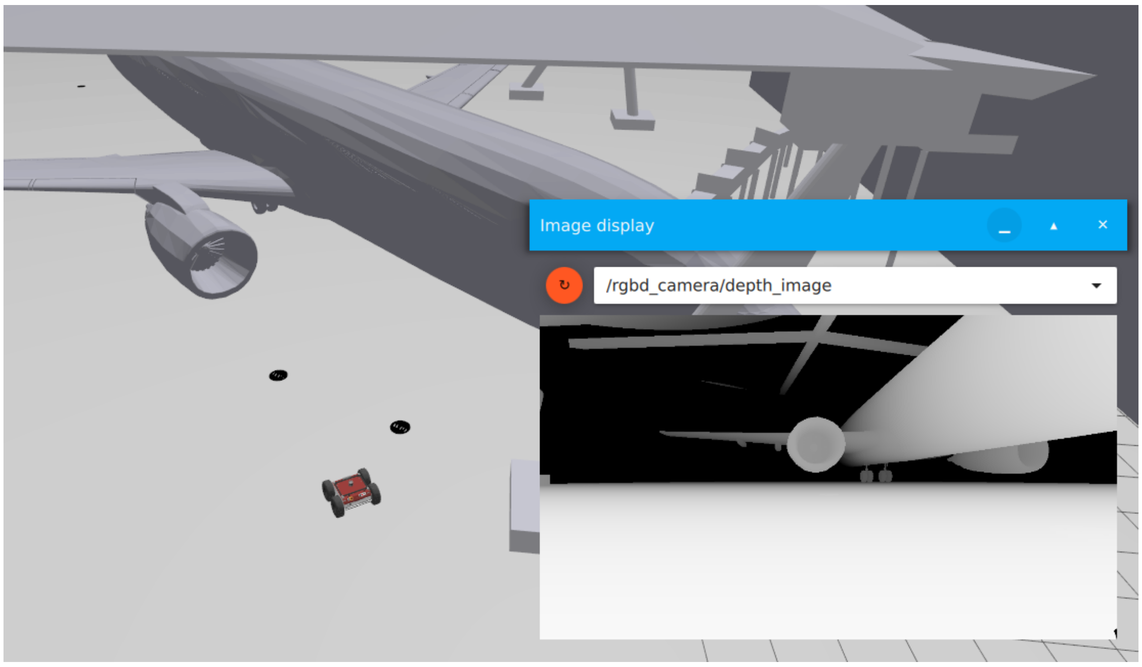


Figure 11 Image of the camera FOV for modelling purposes as seen during the simulation

3.5.3 2D Mapping

Initially, a 2D mapping and navigation approach was used to determine the feasibility and allow computational resource costs to be low. Since there are no 2D LIDAR available on the panther robot we used a ROS package ‘*pointcloud_to_laserscan’* to convert the point cloud data generated by the 3D lidar to ‘*laserscan’* data format. The package allows us to filter the point cloud data based on the height. The points which pass through the filter are then projected on the ground plane. Since the ‘*laserscan’* represented in a range and bearing angle format, there can only be a single point along any bearing angle. Therefore, a second filtering of the projected points is carried out to remove any additional points along the same bearing angle. (The package implements these features). The ‘*laserscan’* generated is then treated as a 2D ‘*laserscan’* data generated from a virtual 2D LIDAR attached to the Panther robot.

‘Slam_ToolBox’ package was used to generate the map of the DARTeC environment. Figure 12 shows the flow chart for the mapping setup used. *‘Slam_ToolBox’* uses input of data from odometer, IMU and virtual 2D LIDAR to map the environment. Once the Map is generated it was saved to be reused later for localization during the navigation phase.

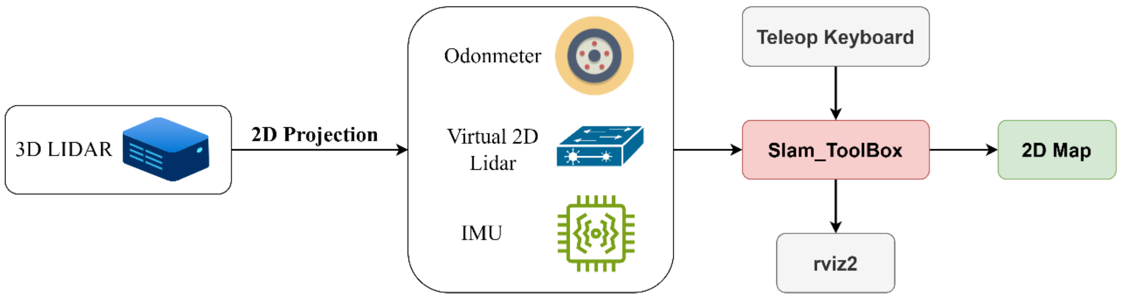


Figure 12 Flowchart for 2D mapping

Since a height-based filtering approach was used to generate the virtual 2D LIDAR data, it was possible to generate the projection for different heights allowing us to capture features which will not be visible from a 2D LIDAR which is placed at the robot's height. Two separate maps were generated for different heights. First map was generated at a height of 30cm. Which means that any point cloud data which were above this height were discarded. The generated map allowed us to capture features which are closer to the ground level. There features can be considered as obstacles for the robot. Figure 13 below shows the generated map.

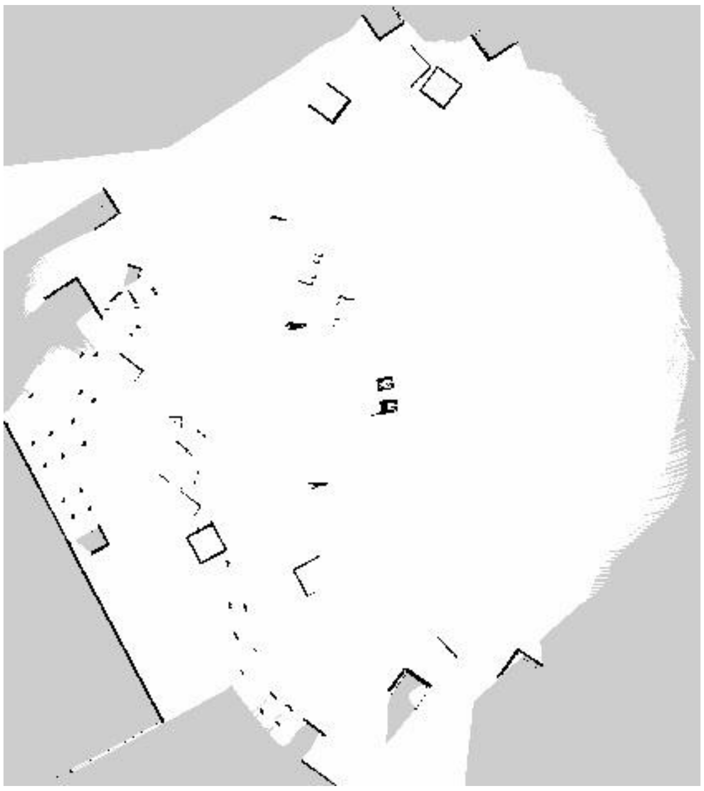


Figure 13 2D Map Generated using 30cm Height Setting

The second 2D map was generated using the height of 4 meters. The height was selected such that the entire plane could be projected down as a feature. Figure 14 shows the map generated using this method. It can be noticed that only the one edge is being projected down even if there are multiple point clouds which passes through the height filter. This is due to the '*laserscan'* data format limitation explained. Thus, we can get an outline of the aeroplane as a feature in the map generated.

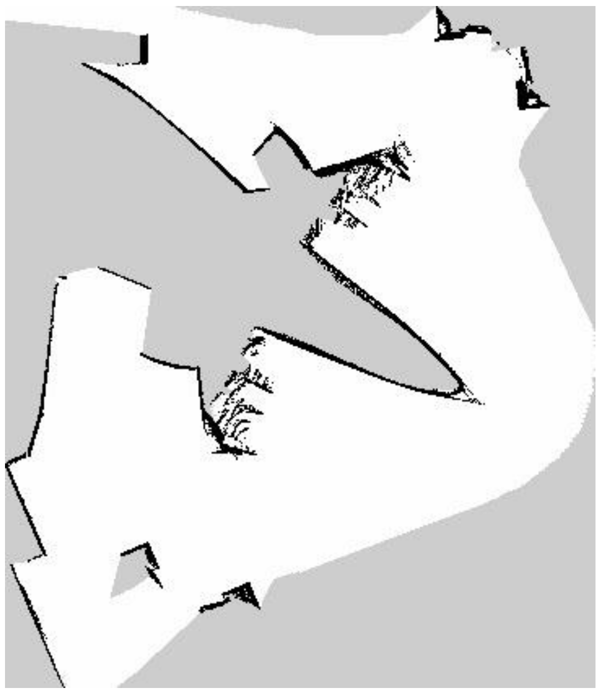
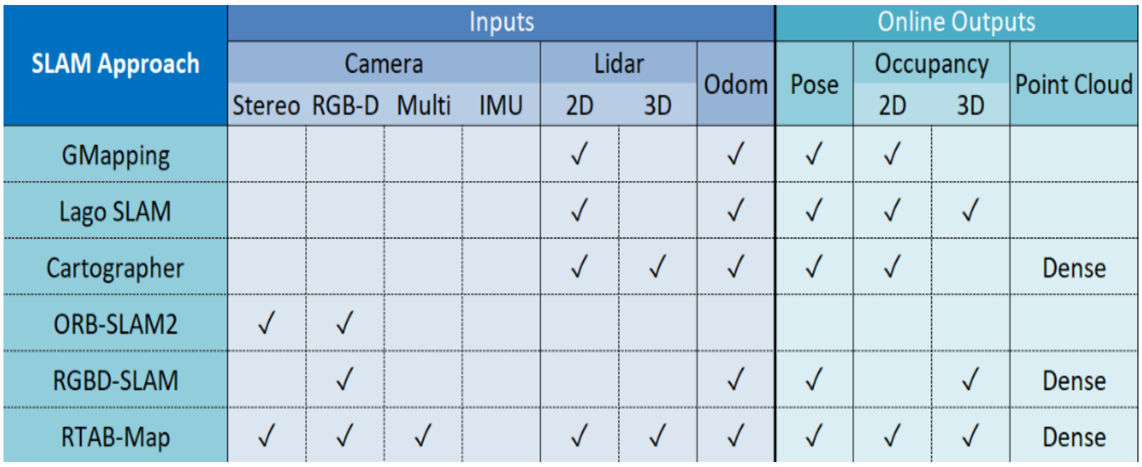


Figure 14 2D Map generated using the 4m height setting

3.5.4 3D SLAM Algorithm Selection

For this project, various packages and algorithms were utilised to achieve the desired simulation result. The mapping of the environment was done using RTAB-Map, because of its advantages over other algorithms, which can be seen in Table 3 below. In particular, the ability to use a 3D LiDAR in addition to a camera is highly beneficial, given that a camera was used to complete the inspection process. This means that the camera can be multipurposed in future work, allowing for a more accurate mapping since multiple sensors can be used and fused together to achieve the desired point cloud. Additionally, RTAB-Map uses loop closure, which as previously mentioned, is advantageous for long term use. Furthermore, RTAB-Map is reported in literature as having a high accuracy, further demonstrating its feasibility for use in this application. On a non-technical note, the RTAB-Map package has a lot of supported documentation, allowing for mitigation if any unexpected problems arise.

Table 3 Comparison table for different ROS SLAM Algorithms



SLAM Approach	Inputs							Online Outputs			
	Camera				Lidar		Odom	Pose	Occupancy		Point Cloud
	Stereo	RGB-D	Multi	IMU	2D	3D			2D	3D	
GMapping					✓		✓	✓	✓		
Lago SLAM					✓		✓	✓	✓	✓	
Cartographer					✓	✓	✓	✓	✓		Dense
ORB-SLAM2	✓	✓									
RGBD-SLAM		✓					✓	✓		✓	Dense
RTAB-Map	✓	✓	✓		✓	✓	✓	✓	✓	✓	Dense

3.5.5 Localisation and Navigation Algorithm Selection

Following the map being generated, the AMCL algorithm was utilised to achieving localisation when navigating between the waypoints. This was used in conjunction with the Nav2 stack, given its high accuracy and vastly available documentation. For the simulations, Gazebo and RViz were used. Gazebo is a 3D simulation software used to visualise the robot's movement and the surrounding map, whereas RViz is used to demonstrate what the robot's sensors are perceiving, i.e. what the robot "thinks" is happening. This is useful to determine any issues which may be present in the sensors and localisation of the robot, by comparing any discrepancies between the "real" world and the robot's internal representation of the current situation.

3.5.6 3D Mapping

Pertaining to the advantages of the RTAB-Map slam as mentioned in the previous section, it was used to generate the 3D map of the DARTeC environment. RATB-Map based map generation ensured full utilisation of the onboard sensors to capture more features of the environment. Figure 15 shows the flow chart of the RTAB-Map setup used to generate the map.

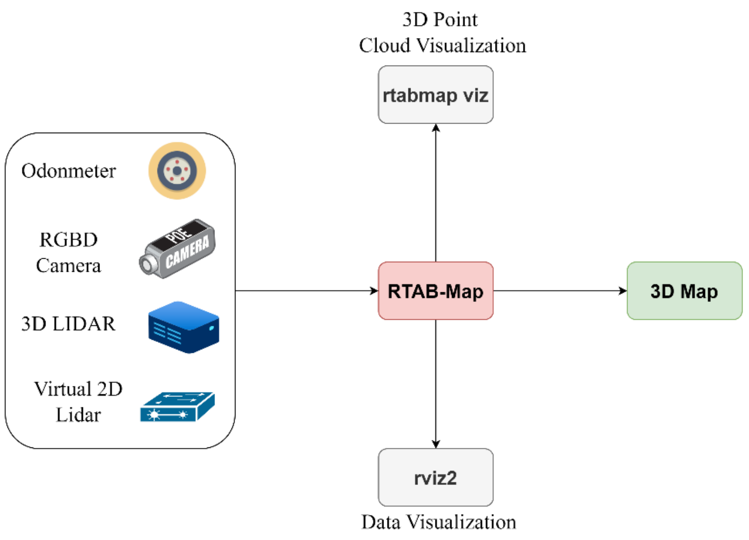


Figure 15. Flowchart for 3D mapping

One of the important features of the RTAB-Map is the ability to simultaneously produce a 2D map which can be used as an input to any navigation package to conduct 2D navigation. However, we were unsuccessful in getting the 2D map generation facility to work properly. Figure 16 shows the 3D map generated. The light blue trail shows the path followed by the robot during the mapping process. The output is a 3D point cloud data-based map which can be loaded and used later for localization.

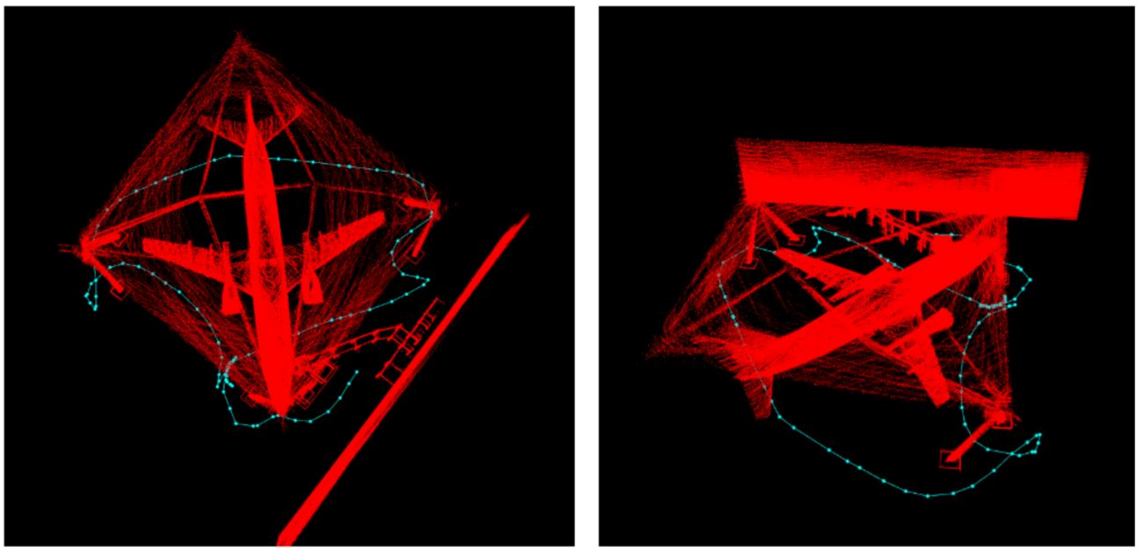


Figure 16 Generated 3D Map of DARTeC environment

3.5.7 2D Navigation

Once we have the map generated, we are no longer dealing with a SLAM problem, instead we are dealing with localization and navigation problem. The task of the SLAM packages in mapping and localization problem is to provide the ‘*map’* to ‘*odom’* transform.

The 2D map generated during the mapping process is used in the navigation phase to act as the static obstacle layer. The Nav2 ROS package was used for navigation along with AMCL for localization which is the part of the Nav2 package. It was possible to use *‘Slam_ToolBox’* for localization during the navigation phase but using AMCL which is the part of the navigation stack allows for a better process and data flow. Figure 17 shows the flowchart of the navigation task.

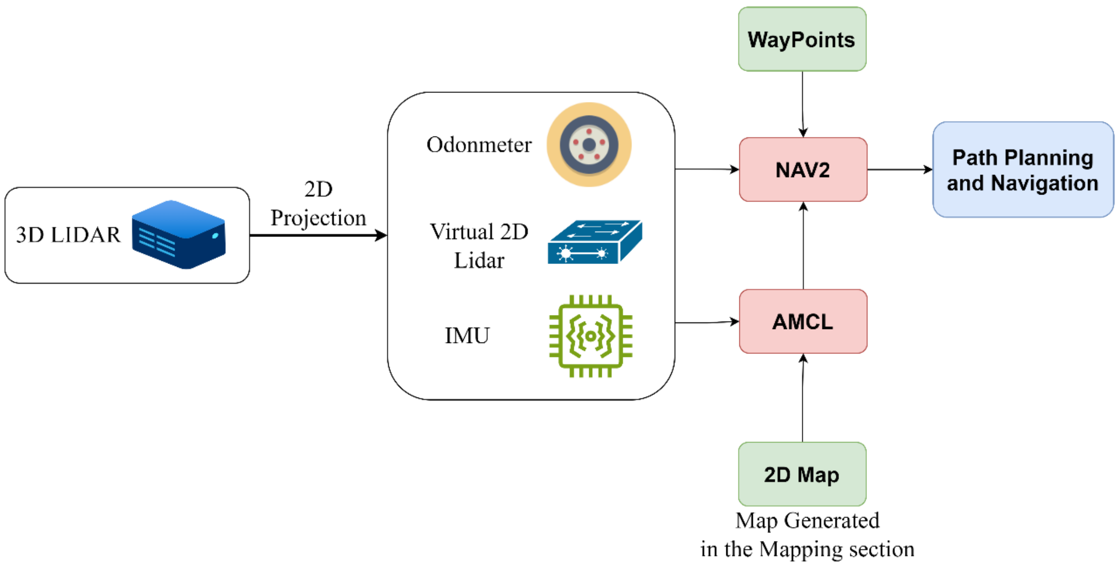


Figure 17 Flow chart for 2D Navigation.

3.5.8 3D Navigation

This section discusses the navigation task conducted using the 3D localization provided using the RTAB-Map. Navigation conducted using 3D localization is supposed to provide better accuracy as we have more features to localize against. In this case RTAB-Map provides the '*map'* to '*odom'* transform which is required by the Nav2 stack for conduction navigation tasks. Another important thing to notice is that even though we are localizing using a 3D map, the DARTeC environment is still represented as a 2D surface, thus allowing the use of Nav2 stack which is a 2D navigation package. Due to issues in the 2D map generation capabilities of the RTAB-Map package, we had to rely on the 2D map generated during the 2D mapping section. This does not affect the performance of the robot as the generated 2D map is used as an input to the static layer of the costmap in Nav2, which is purely used for path generation and static obstacle detection. Another important factor to note is that we had to reuse the '*pointcloud_to_laserscan'* as Nav2 stack's obstacle layer required live '*laserscan'* for static obstacle detection. Ideally, the local and global costmap data must be provided by the RTAB-Map package along with the map data. RTAB-Map also has the capability to provided projected data for the obstacle layer. However, due to the aforementioned issues, we were not able to generate the 2D data from RTAB-Map package. Figure 18 shows the flowchart for the 3D navigation approach used.

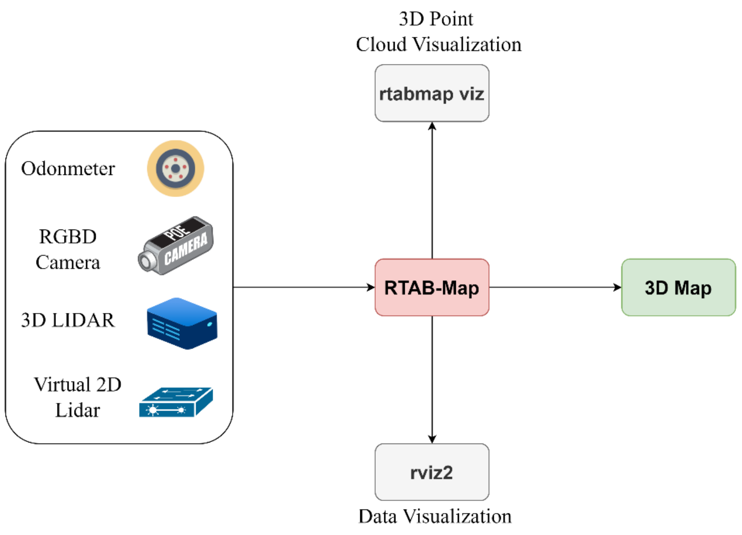


Figure 18 Flow chart for 3D mapping procedure

4 Experimental Results and Analysis

In order to generate the results required for calculating the performance of the developed localisation system, virtual navigation way points were defined on the map which corresponds to the locations of the preflight check. Only the first 4 waypoints were used for the analysis of the performance. The waypoints were fed to the robot using the Nav2 stack's *'simple_commander'* API.

4.1 2D Mapping and Navigation

The accuracy of 2D maps generated by autonomous robotic navigation and mapping is directly influenced by the scan height of laser range finders. This section of the report explores the findings from conducting scans at two different heights: 30 and 400cm.

At a scanning height of 30cm, the Panther robot's laser range is reduced to smaller operative distance, which limits the environmental contact radius. When navigating locations with limited close-range features, such as the aircraft's tail within the DARTeC environment, the limited detection capabilities resulted in the absence of necessary landmarks for accurate localisation. The lack of environmental data causes the robot's internal mapping system to lose reference points, which leads to a shift in map representation. This shift is shown by the misalignment of the environmental features seen in Figure 19.

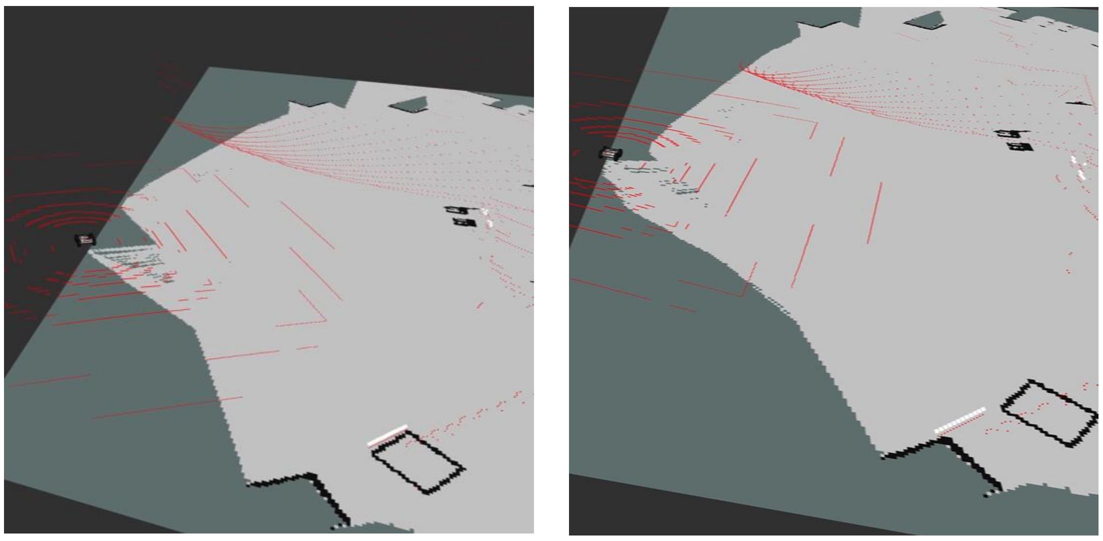


Figure 19 Shift in map representation – 30cm.

These inconsistencies pose a challenge which indicate that the ability to retain spatial orientation at lower scanning heights depends on a high-density feature environment.

On the other hand, at the height of 400cm, the robot benefits from an extended detection range, which allows it to map a greater number of environmental features into its mapping process. Theoretically, the denser feature set provided by the increased scope will help the robot localise better. However, this advantage is counterbalanced by the laser scanner's inability to detect closer, lower-lying features when the robot navigates close to large objects or terrain. This creates blind spots in the data collection, as demonstrated in Figure 20 and 21. where the scan data is shifted due to the laser's overpass of the plane's body. These distortions can lead to inaccuracies in the map as the robot struggles to match its current readings with the previous mapped data.

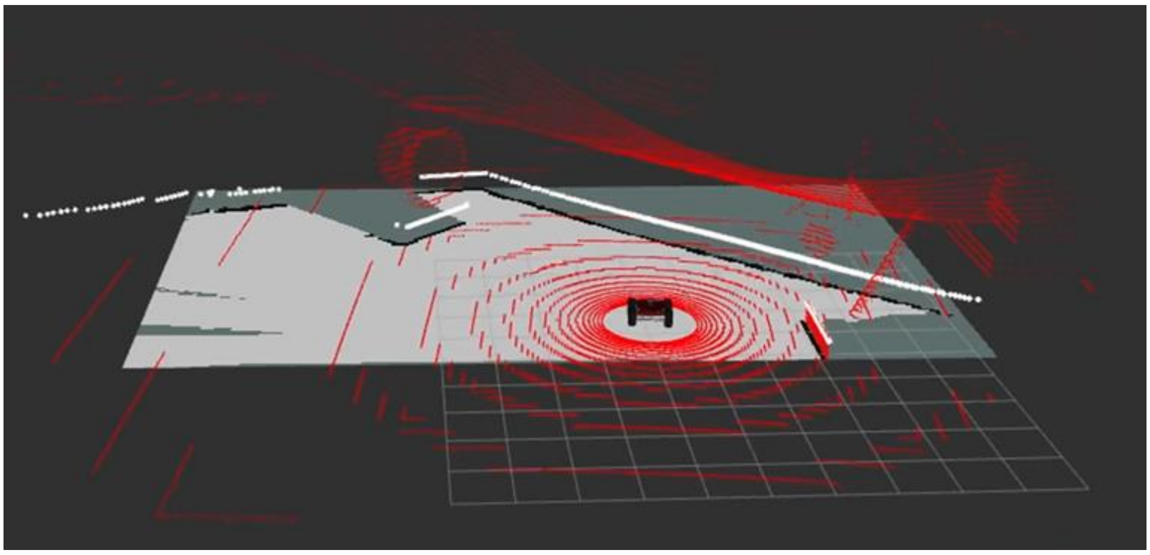


Figure 20 Before shift in map representation – 4 m.

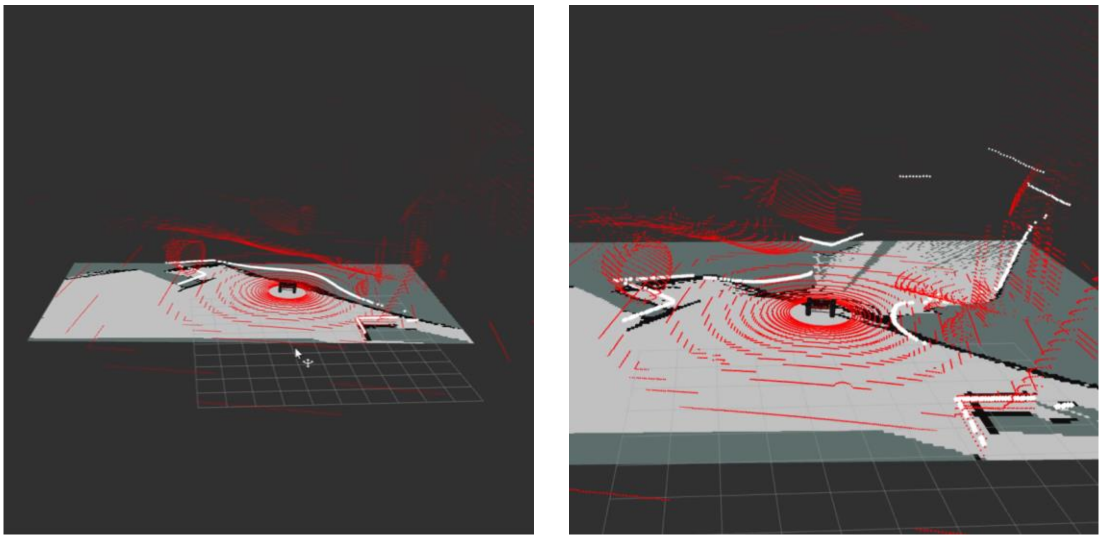


Figure 21 Shift in map representation – 4 m.

4.2 3D Mapping and Navigation

It was found that localization using 3D map gave us the best accuracy. It was observed that the robot accurately followed all the initial 4 waypoints as compared to the 2D localization approach in which the errors in the waypoint following accumulated. One of the major drawbacks of this approach was the computational cost. It was observed that the cycle time for the loop closure detection was about 0.2sec. One way to reduce the cycle time is to reduce the size of the depth camera image used for loop closure detection. Another approach could be to reduce the number of LiDAR point cloud data used for map generation.

4.3 Feasibility Analysis

The mapping solutions generated incorporate loop closure detection, which allows for long term usage of the map. This means that repeated runs will not cause any large-scale deviations due to accumulated error, thus allowing the solution to be robust when utilised multiple times. In the context of MRO, this is beneficial for using the map when multiple pre-flight checks are required over a long period of time. However, this uses the assumption that the same aircraft will be used and remain in the same position every time. Although, AMCL can deal with dynamic features, landmarks and obstacles, as discussed in the literature review. Thus, it maintains its feasibility as a solution for a solution to the pre-flight check inspection localisation challenge even with variable situations, however, there will be errors and issues caused because of the changes in the environment.

4.4 Challenges

Since the entire development work was carried out in the simulation, it was important to get the simulation engine to work as smoothly as possible. In order to achieve this less relevant features were removed from the map, this included an array of fences which were not contributing to the localization of the robot.

One of the major challenges in the analysis of the results were to be able to get the ground truth coordinates from the gazebo simulation environment. There are existing plugins which can help us achieve this through topics on gazebo which can be bridged to ROS. But the coordinates frames of the world in the gazebo and Map were different, thus not being able to compare the accuracy of the waypoints achieved.

When simulating sensors in Gazebo, it is important to get the '*frame_id*' correct in the data being send from the gazebo topic. But for some sensors it is impossible to set the '*frame_id*' in the gazebo. Part of this due to active development of the gazebo simulation software and not all features being implemented. As a workaround we had to create static transform publishing nodes to get the missing transform to be added to the transform tree. Another issue with the gazebo simulation environment is the way fixed joints in the URDF file is handled. Gazebo tends to collapse all the fixed joints in a single joint thus causing a change in the name of the links and joints used, whereas the transform tree created by ROS does not show these behaviours thus causing a mismatch in the frames.

Most of the default drivers and simulation resources provided for Panther were for ROS1 and gazebo classic simulation engine. Since we were doing the development on ROS2 Humble and Ignition Gazebo, which is the supported simulation engine, almost all the examples needed to be ported. Major issues were faced regarding the senor plugin disparities in Gazebo classic and Ignition Gazebo.

Although RTAB-Map is a popular ROS1 SLAM package, development work for porting the package to ROS2 is still ongoing. This has led to less documentation and most of the less relevant features not being implemented. We were also not able to get the 2D mapping functionality of the RTAB-Map to work properly in the setup thus needing to rely on the 2D map generated using the *'Slam_ToolBox'* package.

5 Conclusion

5.1 Overview

In summary, this project was focused on developing a solution to localisation and navigation in an environment with no- or limited-features. This challenge was within the context of an aircraft hangar, where a UGV was employed to traverse the path of a pre-flight check within simulation. During the research stage, we explored various ROS packages to determine the most suitable for our application, and subsequently setup and executed a 3D simulation of the robot. A dummy camera was implemented to model the inspection process. Our solution was to use the RTAB-Map package to complete mapping of the environment, together with the Nav2 navigation stack to complete path planning and following. The localisation was achieved using AMCL, which proved to be effective and allowed the robot to understand it’s position via the 3D point cloud generated from the mapping process. The UGV was able to successfully reach the waypoints of the path with a high level of accuracy, being able to maintain the appropriate trajectory while avoiding static obstacles. Loop closure detection would also allow for long term use by maintaining the accuracy of the map and localisation. These achievements demonstrate the feasibility of our solution, prompting further development to be completed.

An issue which would need addressing in future is the fact that the DARTeC Hangar wall is made of glass, not a solid material as indicated in the simulation. Glass can prove to be a problem when using LiDAR point-cloud based localisation, since the glass would refract the light and potentially cause inaccuracies to be present.

5.2 Further Work

One of the key tasks we were not able to complete in time was the carrying out of live testing at the DARTeC hangar. Simulation environments are useful for tuning and practicing, with continuous accessibility. On the other hand, live testing requires booking the hangar, ensuring health and safety procedures are met, and mitigation of any problems live on-site. Completing live testing, though, would further demonstrate the viability of the solution and would help to uncover any unbeknownst issues which were not present in the simulation.

Furthermore, there are various developments which can be made to further the usability and accuracy of our solution. This could be done by utilising the smart hangar sensors by fusing them with the onboard sensors to achieve localisation. In particular, the UWB sensor systems could be implemented, both in simulation and in live testing, to further assist the localisation process. This would make it more robust by introducing redundancy. However, something which would have to be considered and accounted for is the fact that the UWB system will face a blind spot issue when underneath the aircraft, something which was experienced by the previous research group. This can decrease the accuracy severely if not done carefully.

A different improvement which could be made is to add dynamic obstacle avoidance. In the simulation, only static obstacles were considered, which is appropriate for objects like hangar pillars and the plane itself. However, a hangar will often be populated by people walking around and, potentially, other autonomous systems carrying out various other tasks. Although the Nav2 navigation stack is capable of dealing with dynamic obstacles, this was not explored because of time constraints.

An alternative route is to move towards the implementation of inspection capabilities. The simulation we created uses a hard-coded dummy camera to model the process and demonstrate the viability of the navigation and localisation. This can be developed to incorporate machine vision algorithms in order to complete the visual inspection portion of the pre-flight check, thus providing a more complete solution to the overall problem of automating the inspection.

A different development which can be made is the use of the hangar smart cameras to identify the aeroplane which currently occupies the hangar. Our solution assumes the plane to be a fixed, permanent feature when completing the navigation following the generation of the map. In the real world, the hangar would be occupied by different planes, and the planes would be parked slightly differently every time. This would render the generated map incorrect and unusable. Therefore, adding the capability of identifying the plane’s location and type would allow for a more practical solution.

References

[1] M. Bugaj, A. Novák, A. Stelmach and T. Lusiak, "Unmanned Aerial Vehicles and Their Use for Aircraft Inspection," *2020 New Trends in Civil Aviation (NTCA)*, Prague, Czech Republic, 2020, pp. 45-50, doi: 10.23919/NTCA50409.2020.9290929.

[2] Uniting Aviation. *The future of MRO: emerging technologies in aircraft maintenance* [Online]. Available: https://unitingaviation.com/news/capacity-efficiency/the-future-of-mro-emerging-technologies-in-aircraft-maintenance/.

[3] Sherburn Aero Club (2023, June 26). *Pre-Flight Checklist: Everything You Need to Know Before Taking Off* [Online]. Available: https://www.sherburnaeroclub.com/blog/pre-flight-checklist#pre-flight-checklist.

[4] J. Holl (2016, Dec. 6). *Hangar of the future* [Online]. Available: https://www.airbus.com/en/newsroom/news/2016-12-hangar-of-the-future.

[5] R. Pugliese, T. Konrad and D. Abel, "LiDAR-Aided Relative and Absolute Localization for Automated UAV-based Inspection of Aircraft Fuselages," *2021 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, Karlsruhe, Germany, 2021, pp. 1-6, doi: 10.1109/MFI52462.2021.9591172.

[6] R. E. Kalman, “A new approach to linear filtering and prediction problems,” *Journal of Fluids Engineering*, vol. 82, no. 1, pp. 35 - 45, 1960.

[7] Q. Li, R. Li, K. Ji and W. Dai, "Kalman Filter and Its Application," *2015 8th International Conference on Intelligent Networks and Intelligent Systems (ICINIS)*, Tianjin, China, 2015, pp. 74-77, doi: 10.1109/ICINIS.2015.35.

[8] J. K. Hackett and M. Shah, "Multi-sensor fusion: a perspective," *Proceedings., IEEE International Conference on Robotics and Automation*, Cincinnati, OH, USA, 1990, pp. 1324-1330 vol.2, doi: 10.1109/ROBOT.1990.126184.

[9] W. Elmenreich, “An Introduction to Sensor Fusion,” Vienna University of Technology, Austria, Rep 502, 2002.

[10] K. Siwiak, "Ultra-wide band radio: introducing a new technology," *IEEE VTS 53rd Vehicular Technology Conference, Spring 2001. Proceedings (Cat. No.01CH37202)*, Rhodes, Greece, 2001, pp. 1088-1093 vol.2, doi: 10.1109/VETECS.2001.944546.

[11] D. J. Seo, T. G. Kim, S. W. Noh and H. H. Seo, "Object following method for a differential type mobile robot based on Ultra Wide Band distance sensor system," *2017 17th International Conference on Control, Automation and Systems (ICCAS)*, Jeju, Korea (South), 2017, pp. 736-738, doi: 10.23919/ICCAS.2017.8204325.

[12] R. T. H. Collis, "Lidar," *Applied Optics*, vol. 9, issue 8, pp. 1782-1788, 1970.

[13] L. Huang, "Review on LiDAR-based SLAM Techniques," *2021 International Conference on Signal Processing and Machine Learning (CONF-SPML)*, Stanford, CA, USA, 2021, pp. 163-168, doi: 10.1109/CONF-SPML54095.2021.00040.

[14] C. Chen and Y. Cheng, "Research of Mobile Robot SLAM Based on EKF and its Improved Algorithms," *2009 Third International Symposium on Intelligent Information Technology Application*, Nanchang, China, 2009, pp. 548-552, doi: 10.1109/IITA.2009.381.

[15] S. Cui, M. Chen, L. He, Y. Zhang, M. Sun and X. Duan, "Design of Mobile Robot Interaction System Based on Slam Particle Filter," *2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, Chongqing, China, 2021, pp. 93-97, doi: 10.1109/IAEAC50856.2021.9390996.

[16] H. Jo, S. Jo, E. Kim, C. Yoon and S. Jun, "3D FastSLAM algorithm with Kinect sensor," *2014 Joint 7th International Conference on Soft Computing and Intelligent Systems (SCIS) and 15th International Symposium on Advanced Intelligent Systems (ISIS)*, Kitakyushu, Japan, 2014, pp. 214-217, doi: 10.1109/SCIS-ISIS.2014.7044862.

[17] F. Zhang, B. Zhang and C. Sun, "A Robust Lidar SLAM System Based on Multi-Sensor Fusion," *2022 11th International Conference on Control, Automation and Information Sciences (ICCAIS)*, Hanoi, Vietnam, 2022, pp. 130-135, doi: 10.1109/ICCAIS56082.2022.9990085.

[18] J. Feng, L. Wang, J. Li, Y. Xu, S. Bi and T. Shen, "Novel LiDAR-assisted UWB positioning compensation for indoor robot localization," *2021 International Conference on Advanced Mechatronic Systems (ICAMechS)*, Tokyo, Japan, 2021, pp. 215-219, doi: 10.1109/ICAMechS54019.2021.9661496.

[19] Malam. *What is Mapping in robotics? How to create Map in Ros* [Online]. Available: https://medium.com/@mansooralam129047/what-is-mapping-in-robotics-how-to-create-map-in-ros-8c002d409c07.

[20] S. Macenski, I. Jambrecic, "SLAM Toolbox: SLAM for the dynamic world", *Journal of Open Source Software*, vol. 6, May 13, 2021.

[21] W. Hess, D. Kohler, H. Rapp, and D. Andor, " Real-Time Loop Closure in 2D LIDAR SLAM," *in 2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016. pp. 1271–1278

[22] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Madrid, Spain, 2018, pp. 4758-4765, doi: 10.1109/IROS.2018.8594299.

[23] M. Labbé and F. Michaud, "RTAB-Map as an Open-Source Lidar and Visual SLAM Library for Large-Scale and Long-Term Online Operation," in *Journal of Field Robotics*, vol. 36, no. 2, pp. 416–446, 2019.

[24] K. Koide, J. Miura, E. Menegatti, "A Portable 3D LIDAR-based System for Long-term and Wide-area People Behavior Measurement," *International Journal of Advanced Robotic Systems*, Apr. 2019.

[25] P. Yu, X. Ruan and X. Zhu, "The loop closure Detection Algorithm Based on Bag of Semantic Word For Robot Navigation," *2020 IEEE International Conference on Information Technology, Big Data and Artificial Intelligence*

(ICIBA), Chongqing, China, 2020, pp. 54-58, doi: 10.1109/ICIBA50161.2020.9277317.

[26] D. Fox, “Adapting the Sample Size in Particle Filters Through KLD-Sampling,” *The International Journal of Robotics Research*, vol. 22, issue 12, pp. 985 – 1003, Dec. 2003.

[27] S. Macenski, F. Martín, R. White and J. G. Clavero, "The Marathon 2: A Navigation System," *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Las Vegas, NV, USA, 2020, pp. 2718-2725, doi: 10.1109/IROS45743.2020.9341207.

[28] Husarian Docs. *Panther options* [Online]. Available: https://husarion.com/manuals/panther/panther-options/

[29] A.B. Nadir, B. E. Y. Diallo, M. Hafsi, S. S. Nama, A. E. Usah, “Smart Hangar Robotics Mobility,” Group Research Project, SATM, Cranfield Univ., Cranfield, UK, 2023.

[30] Cranfield University. *Digital Aviation Research and Technology Centre* [Online]. Available: https://www.cranfield.ac.uk/centres/digital-aviation-research-and-technology-centre

Appendices

Appendix A Ethical approval letter

Reference: CURES/21513/2024

Project ID: 24746

Title: Smart hangar robotics mobility – The localisation challenge

We are pleased to inform you that you have successfully declared that your research project is a **Literature Review – based solely on openly available literature which is in the public domain, and you are undertaking desk-based research not involving any other form of data or information.**

You have also confirmed that your project **does not meet** any of the literature review specified exceptions, listed both within the relevant section of the CURES form and below:

1) Your supervisor has requested that you apply for approval through CURES because:

The journals or data are in a sensitive area (please discuss this with your supervisor)
The project will be embargoed **i.e. will not be publicly available via the Cranfield library immediately or longer term**

2) Approval is/will be specifically required by another external body e.g. journal publishers

Therefore, **you do not require ethical approval** and your CURES application will be automatically closed.
Please keep a copy of this letter safe, if this exception is in relation to your thesis project, you will need to include a copy with your final thesis submission.

If you have any queries, please contact CURES Support. We wish you every success with your project.

Regards, CURES Team

Appendix B Workload Distribution

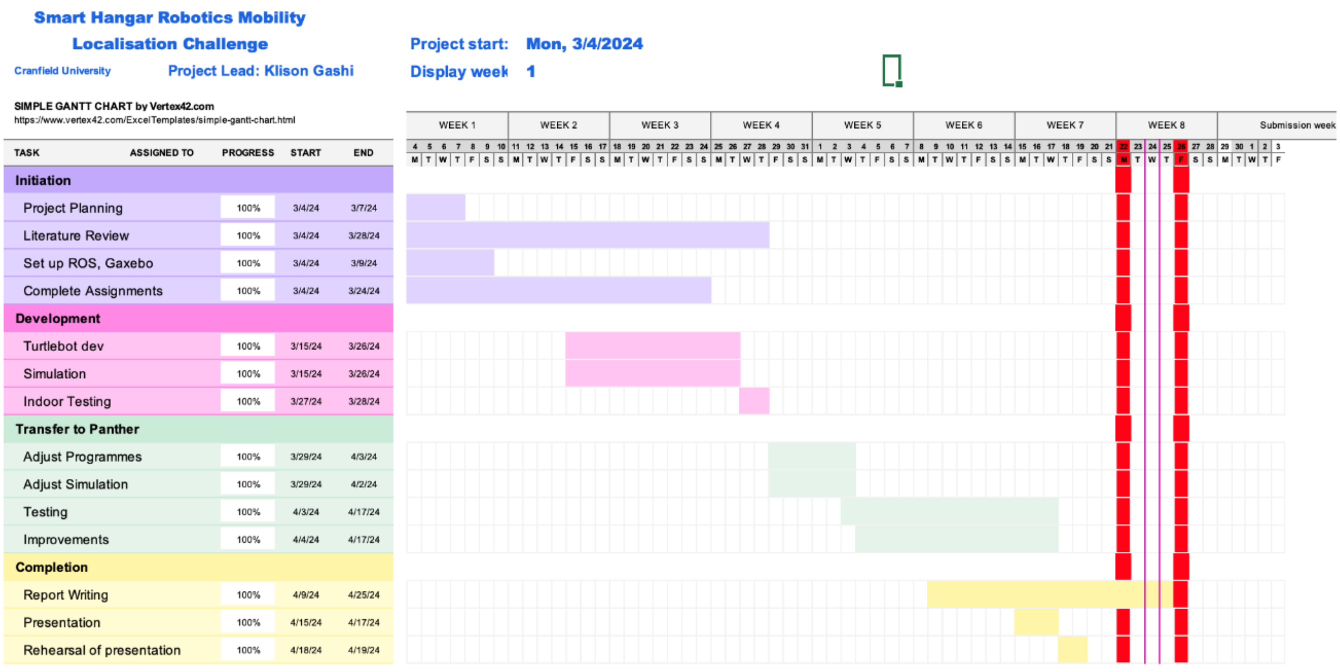
Table B-1 below contains the names and contribution percentage for everyone in the group.

Table B-1 Contribution outline for the group

Name	**Student ID**	**Roles**	**Contribution**
Klison Gashi	411789	Team Manager, Research, ROS 2 Software Development	25 %
Mohan Shanmugavel	431089	ROS 1 Software Development	25 %
Daniyal Syed	441031	ROS 2 Software and Simulation Development, Communications Manager	25 %
Rejin Vadukkoot	424794	ROS 2 Software and Simulation Development	25 %

Appendix C Project Management

Figure C-1 below contains the Gantt chart used to plan and track our progress throughout the project, with the large, vertical red bars indicating the presentation and report submission dates, respectively.



Smart Hangar Robotics Mobility Localisation Challenge

Cranfield University — Project Lead: Klison Gashi

Project start: Mon, 3/4/2024

Display week 1

SIMPLE GANTT CHART by Vertex42.com
https://www.vertex42.com/ExcelTemplates/simple-gantt-chart.html

TASK	ASSIGNED TO	PROGRESS	START	END
Initiation				
Project Planning		100%	3/4/24	3/7/24
Literature Review		100%	3/4/24	3/28/24
Set up ROS, Gaxebo		100%	3/4/24	3/9/24
Complete Assignments		100%	3/4/24	3/24/24
Development				
Turtlebot dev		100%	3/15/24	3/26/24
Simulation		100%	3/15/24	3/26/24
Indoor Testing		100%	3/27/24	3/28/24
Transfer to Panther				
Adjust Programmes		100%	3/29/24	4/3/24
Adjust Simulation		100%	3/29/24	4/2/24
Testing		100%	4/3/24	4/17/24
Improvements		100%	4/4/24	4/17/24
Completion				
Report Writing		100%	4/9/24	4/25/24
Presentation		100%	4/15/24	4/17/24
Rehearsal of presentation		100%	4/18/24	4/19/24

Figure C-1 Gantt chart used for this project

Appendix D Meeting Minutes

Table D-1 contains the meeting minutes completed weekly with the supervisors to discuss the project's progress and next steps. Every team member attended all the meetings.

Table D-1 Minutes from weekly meetings with supervisors

Week	Previous Period	Challenges	Plan for this period	Decisions	Potential Problems
1	Discussed last year's project, problem definition	None	Continue with assignments, begin working with turtlebot, begin literature review	None	Difficulties with assignments, ROS Compatibility issues
2	Setting up ROS and Ubuntu, completed much of literature review	Assignment slowing down progress, Docker not yet working	Fix issues, begin with indoor testing of turtlebot, broaden literature review	ROS 2 will be primary platform used; ROS 1 done in parallel for backup	Difficulties with assignments
3	Some simulation and programming completed, literature review improved	Ubuntu setup issues, unable to connect to turtlebot, progress slower than desired	Continue simulation and programming of turtlebot, conduct live demonstration, meet with MRO group, complete CURES form	Complete all work with turtlebot by 1st April	Programming difficulties, timing constraints

4	Loading Panther into DARTeC world, creating interim presentation	Slow progress due to easter break	Try some different SLAM algorithms, Nav2 and localisation	Cease use of Turtlebot, potentially no live testing with panther due to time	Timing constraints
5	Simulation progress with panther	Issues with using LiDAR, virtual environment crashing	Focus on panther development, begin report writing	Focus only on simulation because of time	Compatibility and programming issues
6	Completion of mapping in simulation	Issues with navigation package	Do navigation and localisation	Complete final presentation	Timing for presentation and report
7	Completion of technical work	None	Complete report writing	None	None

Appendix E GitHub

We created a GitHub repository to store the code and video files for this project. The link for the GitHub is available here:

https://github.com/DaniyalSyed24/DARTeC-TheLocalisationChallenge