



MASOUMEH RAHIMI

AUTONOMOUS LOCALIZATION AND NAVIGATION FOR A
RAILWAY INSPECTION AND REPAIR SYSTEM

SCHOOL OF AEROSPACE, TRANSPORT AND
MANUFACTURING
Phd In Manufacturing

PhD
Academic Year: 2020 - 2023

Supervisor: Dr. Isidro Durazo Cardenas
Associate Supervisor: Prof. Andrew Starr
April 2023



SCHOOL OF AEROSPACE, TRANSPORT AND
MANUFACTURING
Phd In Manufacturing

PhD

Academic Year 2020 - 2023

MASOUMEH RAHIMI

AUTONOMOUS LOCALIZATION AND NAVIGATION FOR A
RAILWAY INSPECTION AND REPAIR SYSTEM

Supervisor: Dr. Isidro Durazo Cardenas
Associate Supervisor: Prof. Andrew Starr
April 2023

© Cranfield University 2023. All rights reserved. No part of this
publication may be reproduced without the written permission of the
copyright owner.

ABSTRACT

Robotic and autonomous systems have brought numerous benefits to various industries, such as increased accuracy, safety, efficiency, cost-effectiveness, and reduced time. The railway industry has also leveraged robotic technologies for track maintenance jobs, albeit their application is often restricted to specific use cases. This research introduces the Robotic Inspection and Repair System (RIRS), which operates both on and around the railway track. A key consideration for autonomous systems is the need for absolute localization, which is essential for maintenance systems on the railway track. Therefore, this thesis focuses on implementing and developing an autonomous localization and navigation system for the RIRS, using the Global Positioning System (GPS). However, due to the railway environment complexity which includes electromagnetic interferences, tunnels, and dense vegetation, GPS inevitably degrades, making vehicle localization extremely challenging. The RIRS localization system is investigated in two separate modes: off-track and on-track. For the off-track phase, to achieve a higher frequency rate and accurate robot pose estimation even in GPS-denied environments, the Extended Kalman Filter (EKF) filter is applied to fuse continuous data with global pose estimates. This approach's effectiveness is also compared with the Real-Time Appearance-based Mapping (RTAB-Map) approach's odometry based on absolute and relative pose error. For the on-track phase, the RIRS aimed to identify track defects at the absolute level initially, but this is infeasible due to GPS unavailability in almost 20% of the railway network. Therefore, first, the RIRS starts navigating autonomously using GPS odometry. Then track-side object detection and 3D pose estimation is implemented to compensate for the error caused by GPS. The average error of 0.07 m in the vehicle's location demonstrates the reliability of this strategy for a maintenance vehicle.

Keywords: ROS, Sensor Fusion, Extended Kalman Filter, Object detection, 3D Pose Estimation

ACKNOWLEDGEMENTS

First and foremost, I would like to express my sincere gratitude to my supervisors, Dr. Isidro Durazo Cardenas and Prof. Andrew Starr, for their support, mentorship, and encouragement along the entire path toward my Ph.D. degree. This endeavour would not have been possible without their valuable comments and constructive suggestions. They helped me to have a better understanding of the research problem. They gave me the opportunity to gain valuable knowledge and experience of being and working at a professional team and a prosperous research group.

I also take this opportunity to thank Amanda Hall and Robert Anderson from the Network Rail company. They guided me with their valuable comments, and suggestions for improving the work.

I would like to thank all the academics who helped me get to this stage with their assistance, and their contributions to my research, especially those who were involved in this research project.

Finally, I express my special thanks to my parents, sisters and brothers for always supporting and encouraging me at every stage.

TABLE OF CONTENTS

ABSTRACT	i
ACKNOWLEDGEMENTS.....	iii
LIST OF FIGURES.....	vii
LIST OF TABLES	xi
LIST OF EQUATIONS.....	xii
LIST OF ABBREVIATIONS	xiii
1 INTRODUCTION.....	17
1.1 Scope of Research	17
1.2 Aims and Objectives	18
1.3 Thesis Overview	19
2 LITERATURE REVIEW	20
2.1 Importance of Railway Maintenance	20
2.2 Impacts of Railway Assets in Railway System Operation	23
2.3 Localization.....	25
2.3.1 Absolute Localization	26
2.3.2 Relative Localization	28
2.4 Rail Vehicle Localization Strategies	30
2.4.1 Elements in the Railway Environment.....	30
2.4.2 On-board Sensors.....	32
2.5 Localization Sensors.....	35
2.5.1 Wheel Odometry	36
2.5.2 IMU.....	36
2.5.3 Global Positioning System (GPS)	37
2.5.4 Light Detection and Ranging (LiDAR)	38
2.5.5 Vision sensor.....	38
2.6 Sensor Fusion.....	39
2.6.1 Sensor Fusion with the aim of Localization	40
2.7 Localization and Mapping Methods	42
2.7.1 GPS-IMU Fusion	42
2.7.2 A Priori Map-Based Localization	42
2.7.3 Simultaneous Localization and Mapping (SLAM).....	43
2.8 Uncertainty in railway localization performance	48
2.8.1 Method Performance Uncertainty.....	49
2.8.2 Sensor Hardware Uncertainty	49
2.8.3 Pre-process Uncertainty.....	50
2.8.4 Environment Uncertainty	50
2.9 Robot Operating System (ROS)	51
2.9.1 ROS Enhancement Proposal	52
2.10 Research Gap.....	53
2.11 Summary	56

3	METHODOLOGY	57
3.1	Robotic Inspection and Repair System (RIRS)	57
3.2	Accuracy and Precision	60
3.3	Autonomous Localization and Navigation	63
3.3.1	Navigation Module	64
3.3.2	Off-track Localization and Navigation.....	67
3.3.3	On-track Localization and Navigation.....	79
3.4	Summary	89
4	RESULTS.....	90
4.1	Evaluation of the Off-track Localization and Navigation.....	90
4.1.1	RTAB-Map	90
4.1.2	Off-track Path Planning and Collision Avoidance	100
4.1.3	Merging GPS with RTAB-Map Result.....	102
4.2	Evaluation of On-track Localization and Navigation.....	113
4.2.1	Test Area.....	114
4.2.2	Ground truth and Datasets	116
4.2.3	Navigation based on the fusion of GPS and SLAM	116
4.2.4	Object Detection and Distance Estimation	123
4.2.5	Performance of the RIRS at Different Speeds.....	130
4.3	Summary	131
5	DISCUSSION	133
5.1	General Discussion.....	133
5.2	Synchronization	134
5.3	RTAB-MAP	134
5.4	Off-track Localization and Navigation	135
5.5	On-track Localization and Navigation	139
5.5.1	On-track Navigation Using GPS and SLAM	139
5.5.2	On-track Object Detection	140
5.5.3	On-track Distance Estimation.....	141
5.5.4	Robot Location Evaluation	143
6	CONCLUSION	145
6.1	Off-track Localization and Navigation	146
6.2	On-track Localization and Navigation	147
6.3	Evaluation of The On-track Results	148
6.4	Contributions and the Potential Impact of this Research	149
7	FUTURE RECOMMENDATIONS.....	151
7.1	Applying Machine Learning Methods for Localization	151
7.2	Applying Map Matching method for Localization.....	151
7.3	Improving RTAB-MAP.....	152
7.4	Applying Machine Learning for Image processing	152
	REFERENCES.....	154
	APPENDICES	171

Appendix A Important Python Scripts	171
---	-----

LIST OF FIGURES

Figure 2-1 Total maintenance expenditure, 2011-12 to 2020-21 (2020-21 prices)	21
Figure 2-2 Track-side infrastructure	32
Figure 2-3 Components of a railway vehicle positioning	34
Figure 2-4 Graphical model of the SLAM problem [116]	45
Figure 2-5 Graphical model of the full SLAM problem [116]	46
Figure 2-6 Graphical model of the Online SLAM problem [115]	46
Figure 2-7 Fishbone diagram showing factors contributing to Location errors.	51
Figure 3-1 Warthog vehicle and its on-board sensors	58
Figure 3-2 (a) On-track and (b) Off-track navigation	60
Figure 3-3 Base station with the on-board RTK-GPS sensor	62
Figure 3-4 RTK-GNSS signaling accuracy estimation (a) STD of GPS data; (b) PDOP value; (c) Number of Satellites	63
Figure 3-5 subsystems of an autonomous navigation system	64
Figure 3-6 Subsystem of a ROS navigation stack	65
Figure 3-7 Global and local cost-maps; Global and local planners	66
Figure 3-8 Path planning	68
Figure 3-9 Dijkstra algorithm a) without obstacle b) with obstacle	69
Figure 3-10 A* algorithm a) without obstacle b) with obstacle	69
Figure 3-11 Front-end and Back-end of Graph-based SLAM architecture	73
Figure 3-12 Loop Closure detection	75
Figure 3-13 Nodes in robot-localization [143]	78
Figure 3-14 Merging SLAM odometry data with the other sensor data in robot-localization	78
Figure 3-15 The illustration of RIRS job phases	80
Figure 3-16 On-track localization and navigation framework	81
Figure 3-17 Architecture for on-track navigation to goal location	83
Figure 3-18 Interface between OpenCV and ROS	85
Figure 3-19 Procedure of feature matching in ROS	86

Figure 3-20 Track-side objects considered as references.....	86
Figure 3-21 Object detection procedure	87
Figure 3-22 Strategy of the 3D pose estimation based on camera.....	88
Figure 3-23 (a) Single object detection and (b) Multi object detection with their relative positions to the camera	88
Figure 3-24 Object detection and 3D Pose Estimation	89
Figure 4-1 Input sensors into RTAB-Map module	91
Figure 4-2 Hardware setup.....	92
Figure 4-3 The transform tree of the Warthog in ROS after launching RTAB-Map representing the frames and the hardware position relationships.....	93
Figure 4-4 The ROS graph of RTAB-Map representing the software functionalities.....	94
Figure 4-5 Mapping output of RTAB-Map (a) Indoor 2D map (b) Outdoor 2D map.	95
Figure 4-6 Visualization of RTAB-Map output in Rviz.....	96
Figure 4-7 representation of octomap_grid in Rviz	96
Figure 4-8 Loop closure and Proximity detection	97
Figure 4-9 Pose-Graph output of RTAB-Map	98
Figure 4-10 State of loop closure during the movement of the vehicle; (a) Green: Loop closures detection accepted; (b) Yellow: Proximity detection in space; (c) Red: Loop closure detection rejected.	99
Figure 4-11 Global and Local cost-maps; Global and Local path planners	100
Figure 4-12 Indoor autonomous navigation of the vehicle from its initial location to the specified goal.....	101
Figure 4-13 Outdoor autonomous navigation of the vehicle from its initial location to the specified goal.....	102
Figure 4-14 GPS output in the test area.....	104
Figure 4-15 Evaluation the trajectory output of RTAB-Map and EKF result ...	105
Figure 4-16 Translational and Rotational parts of pose of the vehicle over time	105
Figure 4-17 The Absolute Pose Error related to translation, mapped onto (a) global-odometry and (b) RTAB-Map-odometry trajectories.	106
Figure 4-18 Absolute positioning error results. (a) APE-Statistical result (b) APE-Box plot (c) APE-Histogram (d) APE-Violin histogram.....	108

Figure 4-19 The APE results with respect to the translation part of global-odometry.....	109
Figure 4-20 The APE results with respect to the translation part of RTAB-Map-odometry.....	109
Figure 4-21 The comparison curve of APE results with respect to the translation part (m). The curves of blue and green represent global-odometry and RTAB-Map-odometry respectively.....	110
Figure 4-22 Relative positioning error results. (a) RPE-statistical result (b) RPE-box plot (c) RPE-histogram (d) RPE-violin histogram.....	111
Figure 4-23 The RPE results with respect to the translation part of Global-odometry over time.....	112
Figure 4-24 The RPE results with respect to the translation part of RTAB-Map-odometry over time.....	112
Figure 4-25 The comparison curve of RPE results with respect to the translation part (m) over time. The curves of blue and green represent global-odometry and RTAB-Map-odometry respectively.....	113
Figure 4-26 The robotic inspection and repair system (RIRS) on the Northampton & Lampton Railway	114
Figure 4-27 Experiment environment for executing various designed tests ...	115
Figure 4-28 Views of the test site from Google Map.....	115
Figure 4-29 Initial and goal GPS locations specified for the vehicle.	117
Figure 4-30 Representation of on-track test in Rviz and Rtmapviz.....	117
Figure 4-31 Initiating the GPS 2D navigation goal node.....	118
Figure 4-32 The completion of navigation based on GPS waypoint.	119
Figure 4-33 Scattering of the GPS data of the goal location.....	120
Figure 4-34 Distance between GPS considered as ground truth and the GPS goal location of each test (m)	121
Figure 4-35 Status of GPS data	122
Figure 4-36 Latitude vs longitude of the captured GPS data.....	122
Figure 4-37 Track-side objects from the point of view of RGB-D camera.....	124
Figure 4-38 Track-side object detection and the related point cloud in Rviz ..	124
Figure 4-39 Point cloud filtering and estimating the distance to the object.	125
Figure 4-40 Distance from detected object to the LiDAR and the corresponding error	126

Figure 4-41 Scattering of the GPS data of the defect location.....	127
Figure 4-42 Distance between GPS considered as ground truth and the GPS of the defect location for each test (m)	128
Figure 4-43 Distance from defect location to the front wheel of trolley	129
Figure 4-44 Distance from defect location to the front wheel of trolley and the corresponding error	129
Figure 4-45 Impact of various speed on the vehicle localization	131
Figure 4-46 Impact of various speeds at the distance from detected object to the LiDAR	131
Figure 5-1 RGB and depth images of the RGB-D camera	142
Figure 5-2 Distance from the detected signal-post to the camera-link when the RIRS moving on the railway track.....	143

LIST OF TABLES

Table 2-1 Comparison between relative and absolute localization [21].....	26
Table 2-2 Train Positioning Sensor Characteristics [adapted from [52]].....	35
Table 2-3 positive and negative points of the SLAM Methods [117].....	48
Table 3-1 Warthog instrumentation hardware specifications.....	58
Table 3-2 RTAB-Map with the supported sensors, front-end and back-end algorithms.....	71
Table 4-1 APE statistical metrics (in meters) for global-odometry and RTAB-Map-odometry.....	108
Table 4-2 RPE statistical metrics (in meters) for global-odometry and RTAB-Map-odometry.....	111
Table 4-3 Mean, STD, and Variance of the GPS data of the goal location.....	120
Table 4-4 Mean, STD, and Variance of the GPS data of the defect location..	127

LIST OF EQUATIONS

(2-1)..... 44

(2-2)..... 44

(2-3)..... 45

(2-4)..... 45

(2-5)..... 46

(3-1)..... 82

(3-2)..... 82

(4-1)..... 103

(4-2)..... 103

(4-3)..... 103

(4-4)..... 103

(4-5)..... 103

(4-6)..... 104

(4-7)..... 104

LIST OF ABBREVIATIONS

APE	Absolute Pose Error
AV	Autonomous vehicles
BRIEF	Binary Robust Independent Elementary Features
C&C	Command and Control
DOF	degrees of Freedom
DWA	Dynamic Window Approach
EBAND	Elastic Band local planner
EKF	Extended Kalman Filter
EKF	Extended Kalman Filter
EMI	Electromagnetic Interference
FLANN	Fast Library for Approximate Nearest Neighbours
FOV	Field Of View
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
IAMS	Intelligent Asset Management Strategies
IMU	Inertial Measurement Unit
KF	Kalman Filter
LIDAR	Light Detection and Ranging
ORB	Oriented FAST and Rotated BRIEF
RAS	Robotics and Autonomous Systems
RGB-D	Red, Green Blue-Depth
RIRS	Robotic Inspection Repair System
RITA	Rail Innovation Test Area
ROS	Robot Operating System
RPE	Relative Pose Error
RTK	Real Time Kinematic
SIFT	Scale Invariant Feature Transform
SLAM	Simultaneous localization and mapping
STD	Standard Deviation
SURF	Speeded Up Robust Feature
TEB	Timed Elastic Band
UGV	Unmanned Ground Vehicle

UKF	Unscented Kalman Filter
WP	Work Package

LIST OF PUBLICATION

- Liu H, Rahman M, **Rahimi M**, Starr A, Durazo Cardenas I, Ruiz Carcel C, Hall A & Anderson R (2023) “An autonomous rail-road amphibious robotic system for railway maintenance using sensor fusion and mobile manipulator”. Computers & Electrical Engineering, Volume 110, September 2023, ELSEVIER.
- **Rahimi M**, Liu H, Durazo Cardenas I, Starr A, Hall A & Anderson R (2022) “A review on technologies for localization and navigation in autonomous railway maintenance systems”, Sensors, 22 (11) Article No. 4185.
- **Rahimi M**, Liu H, Rahman M, Ruiz Carcel C, Durazo Cardenas I, Starr A, Hall A & Anderson R (2021) “Localization and navigation framework for autonomous railway robotic inspection and repair system”. In: TESConf 2021: 10th International Conference on Through-life Engineering Services, Virtual Event: Host: University of Twente, 16-17 November 2021.
- Rahman M, Liu H, **Rahimi M**, Ruiz Carcel C, Kirkwood L, Durazo Cardenas I & Starr A (2021) “Investigating precision and accuracy of a robotic inspection and repair system”. In: TESConf 2021: 10th International Conference on Through-life Engineering Services, Virtual Event: Host: University of Twente, 16-17 November 2021.
- **Rahimi M**, Rahman M, Durazo Cardenas I, Starr A, Hall A & Anderson R, (2022) On-track Localization based on Object Detection. Poster presentation in UK-RAS 2022, Aberystwyth, UK.
- Rahman M, **Rahimi M**, Durazo Cardenas I, Starr A, Hall A & Anderson R (2022) An overview of the challenges for railway maintenance robots. Poster presentation in UK-RAS 2022, Aberystwyth, UK.
- Rahman M, **Rahimi M**, Durazo Cardenas I, Starr A, Hall A & Anderson R (2022) Challenges for a Railway Inspection and Repair System from Railway Infrastructure. In ICCMA 2022, Luxembourg, 9-12 November 2022.
- He F, Durazo Cardenas I, Liu H, Rahman M, **Rahimi M**, Starr A, Polter M (2022) A practical demonstration of autonomous ultrasonic testing for rail flaws inspection. In: TESConf 2022, 8-9 November 2022.

1 INTRODUCTION

1.1 Scope of Research

The global demand for rail transportation is steadily rising, especially in metropolitan areas with fast-growing populations [1]. So regular inspection and maintenance of railway assets such as tracks is crucial, as these assets experience continuous degradation and aging. However, railway maintenance tasks can be expensive, and failure to conduct preventive maintenance or inadequate maintenance can have serious consequences. The upkeep of railway tracks is a crucial financial consideration that must be factored into technical, administrative, and managerial decisions regarding railway systems [2]. In this regard, autonomous inspecting and maintaining of railway assets in railway tracks would improve the safety and efficiency and increase network availability. To undertake autonomous maintenance tasks on the railway track with higher accuracy and reliability, a precise and reliable estimation of the railway vehicle's location is highly important. The precise, and real-time rail vehicle localization is essential to the robot command & control, task execution, safety, and efficiency. Another key point of these systems is about finding the fault point repeatably which is a prerequisite for condition-based maintenance on the railway track. In other words, the measurement system of a fault symptom has to come back to the same point with confidence. Without that, the predictive capability would lose.

The requirements for localizing a robot are a combination of various factors, encompassing hardware, software, and environmental considerations. In this research, environmental challenges, localization measurements, rail vehicle localization strategies, sensors, data fusion and localization algorithms were investigated and researched in detail in the following sections.

Furthermore, the accuracy of localization of the robot was undefined, and Network Rail, the project sponsor, has not specified an accuracy target for the development of the inspection and repair concept. This is because the vehicle can carry multiple inspection and repair payloads such as applying Ultrasonic Testing Unit (UTU) inspection, or for the purpose of QR code detection.

Therefore, the requirement will vary in accordance with the inspection or repair target. For instance, the accuracy of localization based on RTK GPS resolution was assumed to be 2-4 cm, but in practice, however, the requirements were a function of the conducted industrial demonstrations. It means that for localization of the autonomous maintenance robot on the railway track, with the aim of finding the defect location, an accuracy of a few cm was expected. In contrast, for off-track navigation, the accuracy of a few meters was expected.

This research investigated the importance of maintenance systems in the railway industry and the introduction of artificial intelligence (AI) and cognitive analytics technologies into this industry. First, the most recent approaches for railway vehicle positioning, challenges, and the uncertainty sources which impact localization were reviewed. Then sensor fusion, and mapping approaches were provided. Finally, the strategies applied for implementing and developing the localization and navigation system for a Robotic Inspection and Repair System (RIRS) were cited, tested, and evaluated. The experiments were tested on a real railway track to demonstrate the capability of autonomous maintenance systems for assisting humans in track repair tasks.

1.2 Aims and Objectives

Enhancing the vehicle's location especially the location of the maintenance vehicle on the railway track is so critical. The autonomous maintenance vehicles on the railway tracks are designed to find the defect and stop by the defect location, to do some pre-designed repair tasks. In this regard, the main aim of this research is implementing autonomous localization and navigation for the RIRS on and around the railway track for the purpose of precisely enabling inspection and repair. For reaching this aim, there are a number of key challenges that need to be addressed. These challenges are including the unavailability of GPS in some parts of the railway environment, perception or understanding of the surrounding environment of the vehicle, simultaneous localization and mapping of the environment, autonomous navigation, path planning, etc. Considering all these challenges and the aim of the work, the following objectives

are considered for fulfilling the overall goal of achieving the absolute level of positioning for railway vehicles.

- Implementing, testing, and validating the off-track autonomous localization and navigation in a GPS and GPS denied environment.
- Enabling localization on the railway-track in absolute accuracy
- Assessment of the on-track localization and navigation system

1.3 Thesis Overview

This thesis presents autonomous localization and navigation of the robotic inspection and repair system on and around the railway track. In this regard, the following chapters are considered to cover this topic. In chapter 2, a comprehensive review of the importance of railway maintenance systems and the impacts of automation in maintenance jobs is stated. Then the impacts of railway assets on the railway system's operations are reviewed. After that, an investigation of different localization methods, strategies, the most used sensors for localization, sensor fusion approaches, and localization and mapping methods are demonstrated. Following that the uncertainty sources which impact the railway localization performance are discussed. Then an introduction to the robot operating system applied in this research is provided. In the last section of this chapter, research gaps have been discussed based on the literature review. Chapter 3 discusses the methodological approaches adopted to achieve the stated objectives. This chapter first starts by introducing the hardware, sensors, and their specification applied for the robotic inspection and repair system. Subsequently, the applied methods for implementing the autonomous localization and navigation of the robot on and around the railway tracks are developed and explained. The results, the performance of the implemented approaches, and the assessment of the results are provided in chapter 4. The discussion for both the off-track and on-track navigation and localization phases is stated. The conclusion of this research and related to each objective is mentioned in chapter 5 and finally, an outlook on potential future research directions is presented in chapter 6.

2 LITERATURE REVIEW

Localization has high importance, for the railway vehicles such as trains, and maintenance vehicles. Therefore, precise, and real-time rail vehicle localization is essential to the command & control section, task execution, safety, and efficiency. Among rail vehicles, those with the purpose of maintenance which applies modern robotics technology, have been taken into consideration in recent years.

In this chapter, first the importance of railway maintenance and applying autonomous maintenance methodologies in railway networks is discussed to find the shortcomings of the current maintenance systems. Then the impacts of railway assets on railway systems' operation is investigated in detail. Following that railway vehicle positioning approaches, with their principles, advantages, and disadvantages are highlighted. Then different types of position acquisition measurements, sensor technologies, sensor fusion methods for localization purposes, and various conventional localization, mapping, and navigation techniques for autonomous vehicles are stated. Furthermore, the uncertainty sources in railway vehicle positioning are discussed to address the challenge features from different sources, which impact the railway vehicle's localization accuracy and reliability. Finally, a gap analysis has been carried out to find the research gaps and probable solutions.

2.1 Importance of Railway Maintenance

The demand for rail transportation is increasing globally, especially in metropolitan areas with growing populations. Even in Europe where population growth is slower, there is a rise in the number of individuals using trains. This has led to the cost of maintaining and renewing national railways infrastructure, which includes over 300,000 km of combined track, to exceed €25 billion per year in Europe [1], [3]. By utilizing data and analytics, European rail-infrastructure operators could allocate their maintenance spending more efficiently. In the UK's rail network, which includes 40,000 bridges and tunnels, 9,000 level crossings, and 9,941 miles (16,000 km) of railway tracks, there are 4.7 million train journeys

per day [4], highlighting the importance of railway infrastructure. Regular maintenance is necessary to ensure safe operation of railways, which costs over £1 billion per year in the UK and accounts for 18% of Network Rail's overall expenses [5]. Figure 2-1 shows the trends in total maintenance expenditure in 2020-21 prices, with maintenance costs steadily increasing since 2013-14 and expected to reach over £1,830 million in 2020-21.

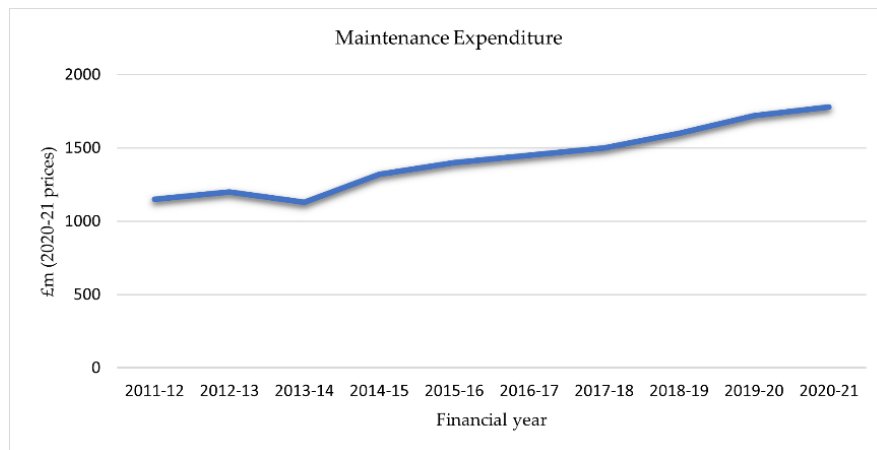


Figure 2-1 Total maintenance expenditure, 2011-12 to 2020-21 (2020-21 prices)

Besides the high budget, which is devoted to railway maintenance, some of the tasks are done by humans, which causes errors during maintenance tasks. Common types of human-related errors in railway maintenance are disassembly errors, inspection errors, decision-making errors, installation errors, lack of skill in using maintenance tools, improper communication, etc. Maintaining or enhancing the safety, reliability, and quality of the entire railway system is a key challenge and paramount for passengers, staff, and the entire rail network. If rail track maintenance is not reliable, the safety of the rail network would be jeopardized, and delays would become routine. Therefore, innovative maintenance methods for railway systems and incorporating maintenance into operation are continually researched and developed to ensure better management of the rail network.

Indeed, the use of robotics and automation in maintenance and repair applications in the industry has become increasingly common, and it has proven to be highly beneficial. These technologies can improve efficiency and

effectiveness, minimize inspection time, and allow for more frequent maintenance of railway networks, among other benefits [6].

The development of miniature robot models has led to the creation of specialized robots that can perform maintenance tasks in various scenarios, including highways, aircraft servicing, underwater facilities, power line maintenance [7], fault detection [8], and track cleaning or performing repair job such as 3D printing [9]. Furthermore, there are a significant number of robots available for track inspection, but few are available for repair purposes such as rolling stock maintenance [10], [11]. The condition of the railway track is also monitored as part of the maintenance strategy [12].

Currently, industries use separate equipment for inspection and maintenance activities; these two have not yet been merged. As high-speed railway traffic increases, railway infrastructure managers worldwide are seeking to develop automatic inspection systems that can identify rail defects, sleeper irregularities, and missing fastening elements. By integrating artificial intelligence (AI) and cognitive analytics technologies into these procedures, the entire process can become more dynamic and autonomous [10]. The European Shift2Rail program's IN2SMART2 [13] projects prioritize intelligent digitalization in railway maintenance, which aims to bring an AI-based robotic strategy to improve task decision support and a scalable asset information management framework. Inspection, repair, and related tasks for rail tracks are critical to the life-cycle engineering and management of track assets, from concept design to disposal.

In conclusion, robotic maintenance systems that combine autonomous sensing, control, decision-making, remote communication, and work execution for inspection and repair are highly demanded. As the industry continues to develop, it is expected that robotics and automation will play an increasingly significant role in maintenance and repair applications, leading to greater efficiency, effectiveness, and safety in various scenarios.

2.2 Impacts of Railway Assets in Railway System Operation

Network rail manages, maintains, and develops the assets which make up the railway network. Railway assets consist of signalling, electrification, bridges, tunnels, level crossings, vegetation, tracks, stations, trackside infrastructures, overhead lines, signal posts, light posts, and many other assets. Certain elements of the railway infrastructure, such as sleepers, clips, fishplates, and point motors, are relatively small and repetitive, while others, such as tunnels and stations, are quite large. These items of equipment, also known as trackside furniture, are positioned along the sides of the railway track and play a vital role in conveying information between the railway vehicles and the command-and-control and safety center [14], [15]. They also provide an advantage for the autonomous navigation system. These objects can be applied as a reference for improving the localization and navigation of the robot on the railway track.

These unique infrastructures pose some distinctive challenges to the rail vehicle's localization and navigation. It is necessary to find these challenges of the railway assets before implementing a global location and navigation system for the vehicle. This prior information would provide a safe and reliable localization and navigation system for the autonomous robot and help in adopting a proper strategy. In the following a number of these challenges from rail asset which impact the operation of the railway vehicles are cited [16]:

- **Baseline between the base station and receiver:** The rail vehicle may do some specified tasks in long range or short range of the railway track. As far as, Real-time kinematic positioning system (RTK-GPS) [17], is one of the main sensors for specifying the location of the vehicle in global scale, its precision would decrease since the baseline length increases. It works best (cm-level) when short baselines (less than 10 km) are considered [18]. In RTK GNSS measurements, the distance between the base station and the rover might vary from several kilometres to hundreds of kilometres. Along with the greater baseline, greater errors also occur on the observation data, such as ionospheric bias, tropospheric bias, and orbital error bias, which lead to unresolved ambiguity integer.

- **Electromagnetic Interference:** One of the main problems when using GPS receivers in the railway environment is the multiple Electromagnetic Interference (EMI) sources, due to numerous factors such as on-board power-supply installations, catenaries, electrical devices, Overhead Line Equipment (OLE), Signal and Telecommunication (S&T) devices [16]. Overhead catenary electric cables transfer energy to run the train through a pantograph system while the S&T are essential for the safe and speedy movement of the trains.

The interference effects generated by railway equipment on the performance of GPS receivers located on the railway vehicles have been barely analysed. Manufacturer's measurements are usually focused on particular railway vehicles, but there is no general norm about the maximum EMI radiation levels permitted for rail equipment in order to assure the correct operation of GPS receivers. Likewise, there is no uniform set of specifications for railway operations. Requirements addressing accuracy, integrity, continuity, or availability are significantly different according to the kind of railway industry or even if the specification is done by these industries or by component's manufacturers [16]. For instance, accuracy in rail vehicle positioning, based on the type of GPS sensor, may vary from 100 to 0.5 m, ranging from 10 to 50 m is the most common.

- **Multipath:** multipath effects pose a serious problem to railway applications and have a major influence on the positioning results of GPS sensor. As far as railway vehicles such as trains and robots have extensive metallic surfaces, they reflect satellite signals extremely. This results in heavy multipath effects especially in locations of metal constructions such as bridges and cranes or where a large number of containers are stored, and cars are parked [16].

- **Shadowing:** The shadowing problem was expected as a consequence of the complex environment such as tunnels, bridges crossing the tracks, buildings and trees being located quite near to the tracks - all of which served to reduce the number of available satellite signals. High speed railway vehicles often pass-through tunnels up to a length of ten kilometres which results in longer periods of shadowing. The length of a tunnel can be very long or short. During the time that the robot is inside the tunnel, the localization and navigation system of the robot would not work properly, and the robot cannot communicate with the control centre, because of the unavailability of GPS [16]. The effect is like faraday cage signal blockage. Lack of communication with the control centre can create a failure to localize the mobile manipulator in the main control centre. The system-inherent reacquisition time is a reason which affects GPS-receiver performance especially when the railway vehicles are traveling at high speed and where tunnels follow frequently. This time is required by the receiver to produce the first positioning result following a period of shadowing. The length of the reacquisition time depends on the layout of the receiver. During these periods the receiver cannot calculate a positioning result when passing the short open section of the track between two tunnels. The effect of a sequence of shorter tunnels is much the same as that when passing through a long tunnel.

2.3 Localization

Accurate localization in known, unknown, or uncertain environments is the main challenge for the development of fully autonomous vehicles [19]. Autonomous navigation requires a robot to continuously keep track of its location and surroundings. For an autonomous vehicle to navigate without human support, the vehicle needs to locate itself with respect to a map. The robot cannot reach its specified location and find its safe path, without knowing its location within an area and the location of other objects at any instant. The level of accuracy necessary for autonomous navigation varies depending on the objective of the vehicle. For instance, an autonomous vehicle that aims to travel from one place

to another place needs to be aware of its location globally. The autonomous vehicle must be able to sense its immediate surroundings and localise itself in relation to both static and moving objects, such as buildings, trees, pedestrians, other vehicles, or cyclists. Furthermore, the vehicle should also be aware of its location in relation to other adjacent objects or signs if it is built to move in a specific path. This is a requirement for the vehicle to know when to move, stop, change path, or make turns.

In terms of position acquisition measurement, there are two sorts of methodologies including absolute and relative measurement that are introduced and contrasted in [20]. Both absolute and relative localization have their own advantages and disadvantages, as summarized in Table 2-1. Combining them can obtain more accurate and robust results, as the two are mutually complementary. In the following table, these two methods are compared with each other in terms of reference, requirements, error, and application [21].

Table 2-1 Comparison between relative and absolute localization [21]

	Relative Localization	Absolute Localization
Reference	Relative to its initial position	Global reference frame
Requirements	No particular	Conditions must satisfy
Error	Error accumulated. Drift over time	No drift Independent of time
Application	Widely used	Possibility of outage

For this purpose, various sensors, techniques, and systems for mobile robot positioning have been developed and proposed by researchers and engineers, including wheel odometry, Global Positioning System (GPS), Inertial Navigation System (INS), and Visual Odometry (VO).

2.3.1 Absolute Localization

Absolute localization refers to a global solution that utilizes various methods such as beacons, active or passive landmarks [22], map matching, or satellite-based signals, like the Global Navigation Satellite System (GNSS) constellations [23]. It is about determining an object's position and orientation with respect to a global

reference frame. Unlike other localization techniques, absolute localization approaches are not affected by the passage of time or distance, and hence they are considered reliable and robust.

2.3.1.1 Global Positioning Systems

GPS offers precise positioning, navigation, and timing data to individuals with a GPS receiver at no cost. To determine its location and velocity, a GPS receiver uses trilateration, a sophisticated form of mathematical calculation. The GPS is classified as a time-of-arrival (TOA) mechanism, which implies that it computes the distance between a receiver and an authorized satellite by multiplying the transit time of a signal by the speed of electromagnetic waves in free space [24]. The timing is crucial since even a three-nanosecond delay can cause a range inaccuracy of up to one meter.

There are limits to the global positioning system (GPS). Its primary disadvantage is that it requires an unobstructed view of the sky, which is impossible to achieve in metropolitan areas where tall buildings or trees are present, as well as inside buildings, tunnels, or other enclosed spaces. The accuracy, reliability, and robustness of the localization reduces in such environments because of signal attenuation or scattering. In this situation, the localization accuracy is not very high, so Assisted GPS (A-GPS), Differential GPS (DGPS) and Real-time kinematic positioning (RTK-GPS) were developed in order to improve performance [25], [26]. Therefore, GPS by alone is unable to provide a safe autonomous navigation because even mere centimetres of error can result in crashes.

2.3.1.2 Landmark-Based Localizations

A method for absolute localization involves identifying landmarks, which are objects that can be detected by sensors. Landmarks can be natural objects [22] like trees or artificial ones [27] like reflectors, radio beacons, and streetlights. They are classified as either active or passive depending on whether they emit energy or not, which affects how sensors perceive them. If landmarks are stationary and their locations are known, absolute location can be specified by determining the relative position of the object with respect to the set of landmarks.

Landmark-based localization is an early and commonly used technique in the field of robotics. In areas with signal interference, this approach outperforms satellite navigation-based localization. The requirement to frequently and precisely detect the landmarks represents the most important challenge to the adoption of landmark-based localization for autonomous vehicles. The two most popular techniques for recognising the landmark features are LiDAR and vision sensors. LiDAR provides accurate information which only offers distance to an object. LiDAR can offer the landmark's geometric attributes, but it is difficult to use it for distinguishing landmarks with the same geometric shape. A potential solution to LiDAR's shortcomings is to use vision sensors to detect the landmark. Regardless of the implementation approach, landmark-based localization necessitates prior knowledge of the locations of specifically identifiable landmarks in the navigation environment. Therefore, for some small-scale well-known environments, landmark-based localization can be a useful approach.

2.3.2 Relative Localization

On the other side, the relative localization [28] is a local localization technique, which estimate the robot's pose relative to its initial pose based on sensors and kinematic models. In contrast to absolute position, relative location does not have a fixed reference. As a result, relative location will alter depending on the secondary location. In the following some of the relative localization techniques are highlighted.

2.3.2.1 Dead Reckoning

One of the most common and traditional techniques for estimating relative position is dead reckoning. The onboard sensor data about the current direction of motion, such as the first and second derivatives of attitude and heading, is combined with the previous position data to perform localization. In comparison to other localization sensors, these types of sensors are relatively cheap and have no signal interference. When there is signal intervention, dead reckoning is a suitable remedy for tunnel, urban, and interior environments. This, however, requires precise initial location data and is unable to comprehend the dynamic environment. Dead reckoning cannot be accurately accomplished without the

initial position. The main disadvantages of using dead reckoning for navigation include inaccuracies in the kinematic model and the accumulation of error over time. One of the reasons for these inaccuracies is the difficulty in accurately predicting the kinematic model, which makes it challenging to account for external factors such as wheel slippage or driving on unpredictable road surfaces [29], [30]. Additionally, the accumulation of error over time is due to the integration of measurements, and there are no alternative mechanisms to reduce the accumulated error. As a result, the error grows continuously without limit over time scales.

2.3.2.2 Inertial Navigation systems (INS)

Inertial navigation is similar to dead reckoning, but it relies only on inertial sensors such as accelerometers and gyroscopes [28]. An inertial navigation system (INS) typically includes at least three gyros and three accelerometers, which are used to determine the vehicle's acceleration, velocity, and position. The accelerometers integrate acceleration signals to calculate speed and position, while the gyroscopes measure the vehicle's angular rotation [31]. Different sensor configurations are combined to estimate the vehicle's pose and maintain the accuracy of inertial navigation. Inertial sensors are often combined with global satellite navigation systems to improve accuracy [32]. Inertial navigation is suitable for use in any environment or weather condition, including above and below ground.

2.3.2.3 Visual Odometry (VO)

Visual Odometry (VO) has gained attention in both the research and industrial fields due to the development of computational power and computer vision. It is considered a more precise method than dead reckoning [33], [34], and is frequently performed in real-time using an organized image sequence. VO aims to determine the camera's entire trajectory by computing the relative transformation between two consecutive images taken at each point in time. Unlike dead reckoning, VO is not affected by errors caused by terrain or vehicle parameter-dependent factors, such as vehicle wheel variation. Compared to LiDAR-based localization [35], VO is a more cost-effective solution that can

perform similar tasks with higher computational power. However, it relies on sufficient illumination in the environment surrounding the cameras to observe enough features and texture, unlike other sensing approaches.

2.4 Rail Vehicle Localization Strategies

The precise and dependable positioning of rail vehicles is an essential requirement for future railway systems, as it would be used for control and signalling applications. Since rail vehicles can only travel along the railway, even a small error in their location estimation may result in a hazardous overestimation of braking distance. Therefore, it is critical to have an accurate and reliable location estimation for rail vehicles such as trains, trams, subways, and rail-robots, for the administration of the rail system [36], [37]. To achieve this, a wide variety of sensors and infrastructures have been proposed and implemented for railway vehicle positioning, which requires their categorization. These sensors are mostly classified into two categories:

1. Elements in the railway environment
2. On-board sensors

2.4.1 Elements in the Railway Environment

Railway infrastructure plays a crucial role in determining the position of trains on the track, using various types of track-side sensors such as magnetic coils, cable loops, contacts, track-circuits, axle counters, transponders, and radio Balises [38], [39]. For instance, a track circuit, a simple electrical device, can detect the presence of a train on the rail tracks and indirectly locate it on a track segment. Other detectors along the railway track can also be used to determine the train's location. When the train passes over these sensors, transponders (Balises) interact with the train's onboard electronics [40]. While these localization solutions can provide train position information with reasonable accuracy, the installation and maintenance costs of these elements can be high. As a result, these infrastructures are typically distributed along the railway track, with distances ranging from tens of meters for current RFID-based systems to tens of kilometers for traditional commercial railway systems [36]. In subway applications, ground

sensor equipment such as Balises/tags and beacons [41], [42] are commonly used for localization systems. Various track-based objects are examined further in the following section.

2.4.1.1 Transponders

Passive and active transponders are track-based equipment that, when combined with on-board odometers, provide a reliable technique of measuring a train's position [43]. Numerous railway companies have employed them. The transponders send a signal to the train-based receiver that includes information about their position as well as signalling information in some cases. Raising the number of transponders along the track to reduce positioning error results in greater system expenses and lower system reliability [43].

2.4.1.2 Balise

Balise is a transponder or electronic beacon deployed between railway tracks as part of an automatic train protection (ATP) system. It gives the train a reference point as well as heading information. It is an example of a track-mounted transponder used to correct position ambiguity inside the train positioning subsystem over time. Balises, being magnetically coupled transponders, do not require a constant energy source [44]. The Balises are spaced at somewhat regular intervals, and the distance between them is controlled by two factors: train speed and railway complexity [45]. Because accurate stops at stations, point zones, and buffer stops are critical, up to 8 Balises are required at a minimum interval established by the design [45]. Using more than one Balise in crucial stopping areas is regarded to boost the chance of Balise detection by train and lower the possibility of train neglecting to read a Balise.

2.4.1.3 Doppler Radar

Doppler radar can calculate the train's immediate speed by analyzing the frequency difference between the radar transmitted and reflected wave [46], which is based on the theory of the Doppler frequency shift effect. It's a non-contact sensor with two microwave antennas that produces accurate and consistent results regardless of the reflecting surface or vibration [47]. Doppler

radar is proven to provide more accurate data than a tachometer [48]. Following a series of trials, it was proved in a study [49], that the use of 24 GHz coded Doppler radar can offer high accuracy data (within 20 db) for vehicle position and speed monitoring. On the other hand [50] claims that Doppler radar output is frequently influenced by noise and systematic. This issue is also cited in [43], and the sources of errors related to a very smooth reflective surface and change in radiation angle due to accelerating and braking action, vibration, and bias error have been discovered. A number of these sensors are shown in Figure 2-2.

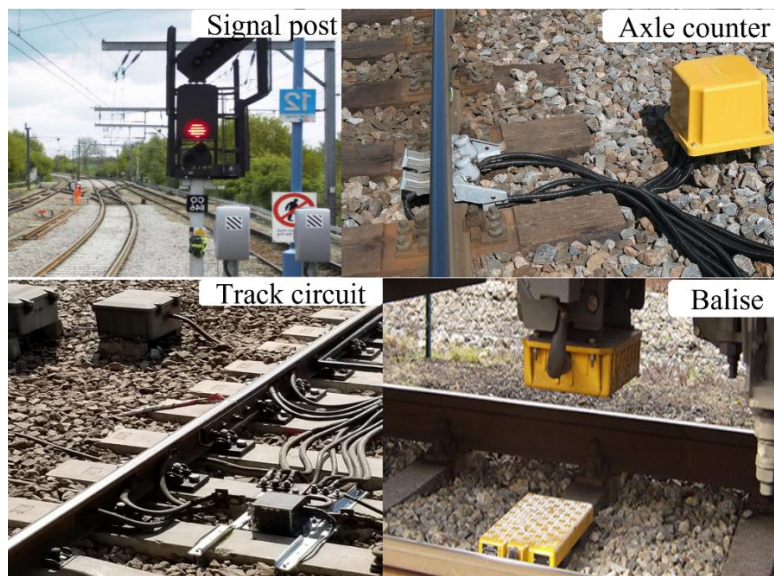


Figure 2-2 Track-side infrastructure

2.4.2 On-board Sensors

On-board sensors such as tachometers, inertial sensors, and satellite-based positioning systems can be used for rail vehicle positioning [51]. Unlike infrastructure-based localization methods, the necessary components and computations for on-board sensor-based localization are located on the vehicle itself, and a map of the railway tracks is also required [52], [53]. This map contains information about the track, its connections, and measurable track features that allow for accurate position estimation [52]. There are various techniques for rail vehicle localization using on-board sensors and a map, each with its own set of sensor types, processing methods, and evaluation criteria. Many studies have focused on using Global Positioning System (GPS) for vehicle positioning on the

railway track, particularly for train integrity validation in European Train Control Systems (ETCS) level-3 [54].

Due to the railway's environment which is complex (section 2.2), employing only GPS cannot provide trustworthy position data in certain scenarios such as tunnels, mountainous terrain, and urban canyons. As a result, approaches for multi-sensor fusion localization have been presented. For a localization system, the fusion of GPS and Inertial Navigation System (INS) is frequently utilised [55]. When there is continuous GPS correction, this system functions properly; otherwise, there is cumulative position error produced by INS. In other words, if no GPS correction is available, utilising the Inertial Measurement Unit (IMU) alone for an extended period of time will result in a significant offset. As a result, additional sensors and data sources, such as odometers, eddy-current sensors, accelerometers, digital maps, Balises, or other on-board sensors, are required to precisely and consistently identify the vehicles location [56].

Another research [57] discusses the need to augment GPS with additional metrics for operational and safety concerns [57]. A multi-sensor system that gathers information from a variety of sensors installed on the vehicle (specifically, an IMU and a GPS) and applies a Kalman-based filtering recursion is investigated in [58]. Regardless of the ground equipment on the particular lines, they concentrated on finding solutions that could be used to any rail vehicle. A particle filter-based localization strategy for a rail-guided robot is provided in another study [59]. The integration of odometry with inertial measurements, laser scans, and image data was contemplated using a particle filter. According to a study [59], an adaptive multi-sensor data fusion technique based on three on-board sensors, including a longitudinal accelerometer, an odometer, and a GPS receiver unit, is recommended for the precise determination of a railway vehicle's position and velocity.

Devices such as eddy current sensors [60], [61], Doppler radar [62], Inertial Measurement Unit (IMU), and optical imaging [63] can be used to determine a vehicle's position through dead reckoning by calculating its distance and direction of movement from a predetermined fixed point. However, the accuracy of these

systems can be compromised due to cumulative errors caused by wheel slippage, bias, wear of wheels and mechanical parts, and hysteresis [64]. As a result, they cannot be relied upon solely for extended periods in critical applications such as collision avoidance and vehicle automation [65]. To enhance their accuracy, they must be periodically reset and combined with GPS [43] or track-side markings like RFID-type devices or Balises [62]. Figure 2-3 shows the components of an onboard localization system that integrates information from onboard sensors to determine the position of a railway vehicle without relying on any trackside aids [66].

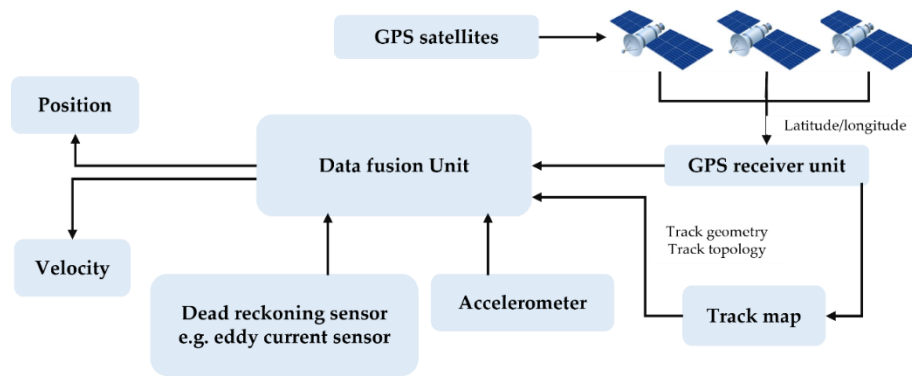


Figure 2-3 Components of a railway vehicle positioning

In Table 2-2, a variety of on-board sensors and track-side infrastructure are compared with each other in terms of absolute and relative localization, rate of frequency, long-term and short-term baseline, outage issues, and environmental impact. Absolute localization, as used in this table, refers to a global localization that depends on GNSS constellations or landmarks to recover the position and orientation data with respect to a global reference frame [28]. Relative localization is a local localization technique that estimates the robot's pose in relation to its starting pose using onboard sensors and kinematic models [28]. Baseline refers to the distance between the vehicle and the base station (reference point) while applying GPS [67]. The length of the baseline varies between, short, medium and large [67]. The outage issue, which relates to a sensor's temporary unavailability for a variety of causes, is another parameter in this table. For instance, GPS can have outages in tunnels, near the buildings, or in urban canyons [68]. The last

parameter is environmental impact which includes any environmental changes such as precipitation, snowfall, fog, or other natural environmental changes.

Table 2-2 Train Positioning Sensor Characteristics [adapted from [52]]

Type	Positioning sensors	Usual Rate of freq.	Absolute positioning	Relative positioning	Long-term solution (large baseline)	Short-term solution (short baseline)	Outage issue	Environment impact
On-board sensors	IMU	100 Hz	No	Yes	No	Yes	No	No
	Wheel sensor (tachometer or odometer)	10 Hz	No	Yes	No	No	No	Yes
	GNSS	20 Hz	Yes	No	Yes	Yes	Yes	Yes
	Eddy current	N/A	No	Yes	No	No	No	No
Track-side equipment	RFID	N/A	No	Yes	No	Yes	No	No
	Balise	N/A	No	Yes	No	Yes	No	Yes

2.5 Localization Sensors

Localization of a vehicle using real-time data collection is one of the most crucial and difficult tasks within any autonomous vehicle. Ensuring the precise and trustworthy measurement of on-board sensors is essential for vehicles, particularly those operating on railway tracks. Several sensors are available to measure the position and speed of railway vehicles, including tachometers, transponders, Balises, INS, Doppler radar, and GPS, each with its own benefits and drawbacks [43]. A comparative survey has identified these differences [69], [70], and assessed and classified the sensors based on multiple parameters, such as cost, accuracy, reliability, sensitivity, coverage, speed of response, and availability [43]. By intelligently combining the redundant or complementary data from multiple sensors, accurate and dependable odometry information can be achieved.

The localization sensors can generally be divided into two categories: proprioceptive and exteroceptive [71]. Proprioceptive sensors operate by measuring the internal state of the system, such as the vehicle's velocity and steering angle. Inertial navigation, encoders, gyroscopes, gyro-meters, and accelerometers are examples of onboard motion sensors used for localization and navigation purposes.

Exteroceptive sensors [72], on the other hand, sense external variables in the surrounding environment, with the aim of measuring information from the vehicle's surroundings. GPS, cameras, radio detection and ranging (radar), and light detection and ranging (LiDAR) are examples of exteroceptive sensors, which are mostly used for relative localization, but some can also be used for absolute localization via feature extraction.

In addition to this categorization, sensors can be classified as passive or active [31]. Passive sensors sense the surrounding energy to produce outputs, such as cameras and GPS, while active sensors transmit energy and measure the reflected energy, such as LiDAR and Radar. Autonomous vehicles use various combinations of active and passive sensors to perform two main tasks (1) environmental perception and (2) localization. In the following a number of these sensors which are applied in this research are introduced:

2.5.1 Wheel Odometry

wheel odometry is the simplest and most widely utilized method to estimate the position of mobile robots [35]. It involves using the number of wheel rotations in contact with the ground to approximate the position of a wheeled vehicle. The revolutions of the wheels can be precisely converted into linear distance with respect to the ground. Wheel odometry is a relative positioning technique. It suffers from position drift and inaccuracy because of wheel slippage, which leads to error accumulation over time. Translation and orientation errors in wheel odometry increase proportionally with the total travelled distance. Wheel odometry allows for high sampling rates and exhibits good short-term accuracy.

2.5.2 IMU

An inertial measurement unit (IMU) is a device that is designed to measure the six degrees of freedom (6-DOF) of a vehicle [73], including three linear acceleration components and three rotational rate components. The IMU consists of different tools that capture various types of data. The accelerometer, for example, measures the vehicle's acceleration by integrating acceleration signals and provides acceleration values across the x, y, and z axes in its local frame.

Similarly, the gyroscope measures angular velocity around the x, y, and z axes and provides accurate information about the vehicle's trajectory in both horizontal and vertical directions. Finally, the magnetometer measures the Earth's magnetic field and provides the vehicle's orientation measurements or heading.

2.5.3 Global Positioning System (GPS)

GPS, an essential element of any navigation system, utilizes satellite-based radio navigation technology to provide precise positional data along with a certain degree of error [28]. It is particularly beneficial due to its long-term stability and capability to prevent the accumulation of inaccuracies. While GPS is commonly utilized in outdoor navigation tasks, it proves ineffective in regions like underground, underwater, forests, and tunnels. The GPS sensors may encounter blockages, signal interference, low bandwidth, multipath effects, high noise content, or jamming, leading to outages. Despite being accurate to around 10m, which is acceptable for most outdoor navigation activities, this level of accuracy is inadequate for railway vehicle localization [74]. To enhance GPS accuracy and facilitate sub-meter or centimetre-level outdoor localization, Differential GPS (DGPS) and real-time kinematic GPS (RTK-GPS) were developed [26], [25]. DGPS and RTK rely on a base GPS station with a known high-accuracy position. In DGPS, the base station compares its position to that calculated by GPS and transmits the difference to the receivers for corrections. RTK, on the other hand, employs the carrier wave to determine the number of cycles between the receiver and the satellite and makes corrections accordingly [75].

Relative positioning techniques involve using multiple receivers simultaneously to track the same satellites. Differential GPS (DGPS) corrects bias errors of the user receiver by utilizing measured bias errors at the base station, which consists of a GPS receiver located at a known location, a user receiver (GPS receiver) called a rover, and a radio communication medium between the two receivers. Real-time kinematic GPS (RTK-GPS) offers centimetre-level accuracy in real-time measurements and provides two solutions: float and fix [76]. The float solution requires at least four common satellites between the base station and the receiver and yields an accuracy range of approximately 20 cm to 1 m. The fix

RTK-GPS solution necessitates at least five common satellites and achieves an accuracy level within 2 cm [77].

2.5.4 Light Detection and Ranging (LiDAR)

The LiDAR device is an optical instrument that uses laser light pulses to collect information from surfaces by creating "points" (3D coordinates) [78]. Its ability to detect obstacles up to 200 meters away and accurately estimate their positions makes it the most precise sensor for position estimation in self-driving cars. The distance between laser pulses emitted and the scattering reflected by targets is measured by LiDAR to calculate the interval and obtain distance [79].

Compared to cameras, the LiDAR sensor functions more dependably in varying weather conditions and is less affected by lighting or weather conditions because of the infrared laser that provides sufficient illumination. Thus, even in tunnels or beneath bridges, accurate measurements can be obtained [80]. The LiDAR sensor is also extensively used in railway applications for various purposes, such as identifying object presence and collision avoidance, detecting cars, trucks, or people on level crossings [81]. Train-borne localization is also made possible with LiDAR [82].

2.5.5 Vision sensor

Cameras are a highly accurate means of creating visual representations of the world, whether through photos or videos. Their advantages over other active sensors include a high data density and the ability to detect object boundaries and classify them accurately. However, despite their accuracy in providing visual information, cameras have certain limitations. For instance, while they can distinguish details in the environment [83], they require distance calculation to determine object location. Additionally, camera-based sensors struggle to detect objects in low-visibility conditions, such as fog, rain, or night-time. Despite their high resolution [35] and ability to detect RGB information, cameras have drawbacks such as vulnerability to sunlight, contrast issues, and lack of depth information.

One popular type of vision sensor is the stereo vision camera, which has two lenses [84]. Unlike a single-camera setup, stereo cameras can capture depth information by observing scenes from different perspectives [85]. However, monocular vision systems have a higher degree of uncertainty in position estimation compared to stereo vision. Stereo vision offers efficient and accurate triangulation but is more expensive and requires more intensive calibration. Nonetheless, both monocular and stereo vision sensors are more affordable and provide smaller-scale solutions than range sensors. They also offer valuable information such as the ability to perceive colors and textures of surrounding objects and more accurate feature recognition than other sensors.

In the field of railway inspection and maintenance, various publications describe the use of vision sensors to detect missing bolts, rail-head wear, and other surface geometry inconsistencies [86]. One example is a monocular thermal camera system mounted on a train that can detect rails in imagery and identify anomalies on the railway [87]. Another research describes a prototype railway object detection system installed in the cab of a train, which uses a single camera to acquire images of the railway scene from the front of the train and an infrared laser to provide illumination when lighting is insufficient [88].

2.6 Sensor Fusion

New approaches in instrumentation, technology, engineering and, in particular sensor fusion, have opened up new paths for achieving better reliability and accuracy in measurement. Sensor fusion is an essential aspect of most autonomous systems. It is the process of merging data from multiple sensors to overcome the uncertainty of individual sensors, and hence improve the efficiency of the overall autonomous vehicles (AV) [89]. Several advantages related to the sensor fusion methods over single sensor systems are presented as: reduction in uncertainty, increase in accuracy and reliability, extended spatial and temporal coverage, improved resolution and reduce system complexity [90]. On the other side, there are some challenges related to the sensor fusion that need to be addressed before it becomes mainstream. One challenge is sensor calibration, ensuring all sensors are aligned correctly and producing accurate data. Another

issue is sensor noise which can cause inaccurate readings. Finally, managing the large amount of data produced by multiple sensors can be difficult, that thanks to advances in big data management and analytics, this issue is solved [31].

Obtaining the most advanced fusion techniques and algorithms in sensor fusion is challenging due to the diverse range of proposed fusion methods in literature [91]. In a review study, fusion techniques and algorithms were classified into two categories: classical sensor fusion algorithms and deep learning sensor fusion algorithms [89]. Classical sensor fusion algorithms such as knowledge-based approaches, statistical methods, and probabilistic methods, employ theories of uncertainty [92] to combine sensor data, while deep learning sensor fusion algorithms use multi-layer networks to process raw data and perform complex tasks, such as object detection. There has been a noticeable increase in research on deep learning sensor fusion algorithms for autonomous vehicles (AV), with Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) being among the most commonly used perception systems. To enhance the real-time performance of object detection, an advanced weighted-mean You Only Look Once (YOLO) CNN algorithm was proposed in reference [93] to merge RGB camera and LiDAR point cloud data. Other examples of deep learning-based sensor fusion algorithms include ResNet or Residual Networks, which facilitate the training of deep networks [94], SSD or Single-Shot Multi-box Detector, which uses boxes with different sizes and aspect ratios per feature map location to detect objects of varying sizes [95], and CenterNet [96], which is the most advanced monocular camera 3D object detection algorithm.

2.6.1 Sensor Fusion with the aim of Localization

In [97], a multi-sensor fusion approach was developed for localization using filtering methods like Extended Kalman Filter (EKF) and Unscented Kalman Filter (UKF), which can integrate numerous diverse sensor measurements. An information fusion method that uses GPS, INS, and odometry sensors to improve the accuracy of autonomous vehicle localization was studied in [98]. The authors also introduced an artificial neural network as a substitute for GPS measurement prediction when the GPS signal is lost, along with a fuzzy logic system to minimize

wheel slippage errors that can lead to an increased error in position estimation from the odometer.

An EKF fusion approach that integrates GPS and camera measurements with odometers, using the odometers for the prediction step of the filter and GPS and cameras for the measurement step was implemented in [99]. Another research [100] discussed UKF-based localization and compared it with EKF-based localization, demonstrating that the UKF provides better average localization accuracy and consistency of estimates, thanks to its unscented transform approximation of the prior and measurement distributions. A navigation system based on UKF for vehicle pose estimation, fusing GPS, IMU, and wheel encoder measurements while utilizing a Stanley controller for navigation control was proposed in [101]. A multi-innovation UKF to integrate sensor measurements from different sources to estimate the pose and velocity of skid-steer mobile robots, using a dual-antenna GNSS, two encoders, and an IMU for robot localization and slip error component estimation was employed in [102].

Intelligent transportation systems have increasingly utilized multi-sensor data fusion techniques, as they offer numerous advantages [103], [104]. By integrating different types of position and speed sensors, a more comprehensive understanding of system conditions can be obtained, which is beneficial for monitoring and control tasks [43]. The optimal selection of sensor combinations should take into account various factors such as availability, reliability, speed of response, cost, and system accuracy to provide an improved on-board measurement system. For instance, the integration of tachometer and transponder sensors (a type of basic sensor fusion) can enhance accuracy levels by resetting position at fixed points. However, it cannot address errors that occur due to slip and slide between two transponders [105]. In [106], a localization algorithm is suggested to improve odometric estimation accuracy, especially in critical adhesion conditions, by fusing data from tachometers and an inertial measurement unit. Train localization systems proposed in [59] include three on-board sensors: longitudinal accelerometer, odometer, and GPS receiver unit, and a data fusion technique is proposed to achieve optimal results.

A widely adopted fusion approach involves the integration of GPS and INSs, which may result in errors in the estimation of vehicle position and speed due to drift [60]. To overcome this issue, a filter is utilized to reduce the discrepancy between data output signals from the INS and an external source, such as GPS [74]. In scenarios where GPS signal degradation occurs, such as indoor environments, dense forests, and areas near tall buildings, alternative solutions must be considered to compensate for signal loss. One such solution is the fusion of tachometer and IMU.

2.7 Localization and Mapping Methods

As it is noted earlier, localization is the task of finding geo-position relative to a reference frame in an environment [42], and it is fundamental to any mobile robot, especially crucial for autonomous vehicles. In this part, three of the most famous localization and mapping techniques are cited including fusion of GPS-IMU, a priori map-based localization, and Simultaneous Localization and Mapping (SLAM) approach.

2.7.1 GPS-IMU Fusion

The main principle of GPS-IMU fusion is addressing the low accuracy and interference issues in GPS as well as correcting the accumulated errors related to IMU sensor in intervals with absolute position readings [107]. In a GPS-IMU system, changes in position and orientation are measured by IMU, and this information is processed for localizing the vehicle with dead reckoning. The significant drawback of IMU, and in general dead reckoning sensors is that the error accumulates over time which this would often lead to failure in long-term operations [108]. For the aim of accurate vehicle localization, the combination of GPS-IMU systems is not enough as they do not meet the performance requirements [109]. GPS-IMU fusion are mostly used for initial pose estimation in tandem with LiDAR and other sensors.

2.7.2 A Priori Map-Based Localization

Priori map-based localization provides many advantages to achieving an accurate position of a mobile robot. Without a map as a reference, odometry

methods like dead reckoning would quickly begin to drift over time. Applying a map of the environment would enable the localization system to simply restrict or decrease the error in estimating the state of a mobile robot [110]. Matching is the core concept of a priori map-based localization systems. In this strategy, localization is accomplished by comparing information from online reads to that on a carefully constructed map and identifying the place of the best match [109]. The matching method frequently starts with an initial pose estimation, for instance using a GPS. There are several methods for creating maps and preferred modalities. Environment changes have a negative impact on the effectiveness of map-based techniques. This effect is particularly noticeable in rural areas where changes in roadside plants and buildings can lead the past information of the map to differ from the real environment [111]. Prior map-based localization has the main benefit of reducing uncertain outliers while simultaneously improving accuracy and reliability.

2.7.3 Simultaneous Localization and Mapping (SLAM)

Simultaneous Localization and Mapping, or SLAM, is an acronym for two key processes: localization and mapping. The SLAM technique investigates whether it is possible for a mobile robot to be deployed in an unknown environment and to incrementally create a reliable map of that environment while also establishing its position within that map. For the community of mobile robots, finding a solution to this issue has been crucial because it would provide them the tools to create genuinely autonomous robots [112]. By applying SLAM approach, robots can estimate their positions at various times and use this information to determine their trajectory and create maps of their surroundings. When combined with motion, range, or vision sensors, global navigation sensors can provide accurate global and local localization. However, these sensors are prone to error drift over time. To overcome this, SLAM compensates for the drifts and errors by constructing a global map and identifying previously visited locations.

2.7.3.1 SLAM Problem Definition

As referenced in [113], the SLAM problem refers to a scenario where a mobile robot navigates through an unfamiliar environment, beginning at a known location

with coordinates x_0 . Due to the robot's uncertain movement, it becomes increasingly challenging to determine its current position in global coordinates. The robot is equipped with sensors that enable it to perceive the surroundings as it moves. The difficulty of SLAM is about two main points. The first thing is about estimating the map and the path which are both unknown but correlated. The second difficulty is about finding the data association between what the robot has observed now, and what it has observed before. It might become sometimes difficult when there are a number of landmarks situated next to each other and the robot cannot find out any relationship between them.

2.7.3.2 Formulation of SLAM

Formally, SLAM is best described in probabilistic terminology. Here time is denoted by t , and the robot location by x_t . The sequence of locations, or path, is then given as [114], [115]:

$$X_T = \{x_0, x_1, x_2, \dots, x_T\} \quad (2-1)$$

The estimation algorithm frequently uses the starting location x_0 as a point of reference because other positions cannot be sensed. Odometry offers relative information between two adjacent locations. Here u_t stands for the odometry that defined the motion between time $t - 1$ and time t ; this information might be gleaned from the wheel encoders on the robot or from the control commands sent to those motors. Here T , is some terminal time (T might be ∞). The following sequence characterizes the relative motion of the robot:

$$U_T = \{u_1, u_2, u_3, \dots, u_T\} \quad (2-2)$$

For having a motion that is noise-free, U_T would be sufficient to retrieve the poses from the initial location x_0 . However, odometry measurements are noisy, and path integration methods inevitably deviate from the truth. Since the environment is composed of landmarks and objects, m would denote their locations. The robot measurements establish information between features in m and the robot location x_t . If it is assumed that the robot takes exactly one measurement at each point in time, the sequence of measurements would be as

$$Z_T = \{z_1, z_2, z_3, \dots, z_T\} \quad (2-3)$$

The variables involved in the SLAM problem are illustrated in Figure 2-4. It presents the order of locations and sensor measurements, as well as the causal relationships between these variables. This illustration serves as a graphic model. It aids in comprehending the SLAM problem's dependencies. In this figure, causal relationships are represented by arcs, and shaded nodes are visible to the robot. The objective of SLAM is to estimate the unobservable variables.

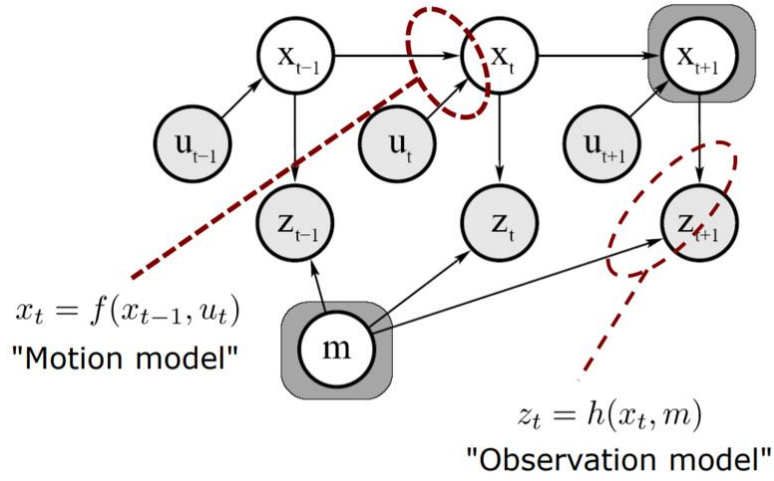


Figure 2-4 Graphical model of the SLAM problem [116]

Now the SLAM problem is about reconstructing a model of the world m and the sequence of robot locations X_T from the odometry and measurement data. The literature separates the SLAM problem into two primary categories, both of which have equal practical significance. One requires estimating the posterior over the complete robot path together with the map and is referred to as the full SLAM problem.

$$p(x_{0:T}, m \mid z_{1:T}, u_{1:T}) \quad (2-4)$$

The full SLAM problem is defined as the problem of estimating the joint posterior probability over X_T and m from the whole trajectory.

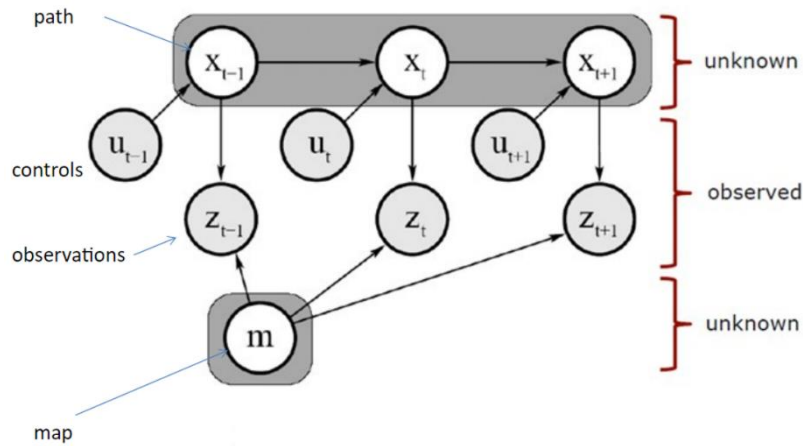


Figure 2-5 Graphical model of the full SLAM problem [116]

The second SLAM problem is the online SLAM problem which is defined via the following probability:

$$p(x_{t+1}, m \mid z_{1:t+1}, u_{1:t+1}) \quad (2-5)$$

Online SLAM seeks to recover the current pose of the robot, instead of the entire path, usually based on the last sensor information. Online problem-solving algorithms are typically incremental and can handle one piece of data at a time. Such algorithms are often referred to as filters in the literature. Graphical model of the online SLAM is shown in Figure 2-6.

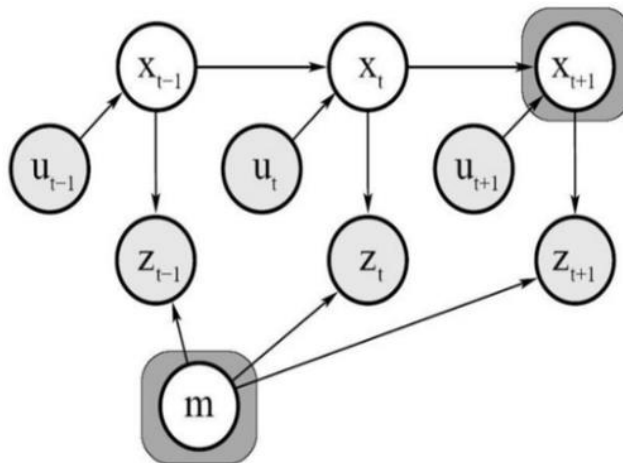


Figure 2-6 Graphical model of the Online SLAM problem [115]

2.7.3.3 SLAM types

Under the SLAM area there are two categories for estimating the pose of the vehicle. One of them is filter-based approaches which is formed based on Bayesian filter, and the other category is optimization-based approaches. The former corresponds to iterative processes that are thus suited to online SLAM and the latter methods are usually applied to solve the full SLAM problem. The Bayes theorem serves as the foundational idea in the filter-based category. It operates iteratively in two steps and makes use of the Bayesian filter [117]. KF (Kalman Filter), EKF (Extended Kalman Filter), PF (Particle Filter), and Fast-SLAM are included in the filter-based category [117]. One of the most popular filter-based category SLAM methods is the EKF. It can linearize the non-linear models that autonomous vehicles typically encounter in the actual world, and this is the main reason why EKF is the most widely used method among all the SLAM techniques. However, because to its limited computing capabilities, it has recently lost some of its popularity [118]. Additionally, it doesn't appear to be appropriate for self-driving cars, particularly in a more dynamic and larger environment. The second most popular filter-based category is Fast-SLAM, which makes use of particle filters, a type of non-parametric statistical filtering [119]. Fast-SLAM is a popular method for online SLAM and provides a pioneer solution to the data association problem in SLAM.

The most popular method in optimisation-based category is called Graph-SLAM. This SLAM approach is regarded as the graphical version of the Bayes' theorem, and it uses the matrix form to link the relationship between the landmarks in the map and the vehicle's pose [120]. It is now the primary paradigm for resolving the entire SLAM challenge since it successfully uses sparse nonlinear optimization techniques to the SLAM problem.

A comparison between filter-based and graph-based SLAM approaches have been carried out in [121]. It is noted that the advantage of particle filters is their ability to represent multimodal distributions and they are generally easier to implement. On the other hand, non-stochastic graph-based techniques can more easily satisfy functional safety requirements. It is also cited that the graph-based

localization achieves a higher level of accuracy than the particle filter. So, in areas which higher level of localization is intended, graph-based approaches are mostly used. In this research, a graph-based SLAM approach named Real-Time Appearance-based Mapping (RTAB-Map) is applied. In the next chapter, it will be defined more. In the following table, a summary of the positive and negative points of these methods are provided:

Table 2-3 positive and negative points of the SLAM Methods [117]

Method	Pros	Cons
EKF	- Able to deal with non-linear model. - Can provide good loop closure reference. - Large body of work.	- Problematic when dealing with large maps that lead to numerous issues.
Fast-SLAM	- Can accommodate any distribution. - Improve computational speed. - Require less storage memory. - Better data association accuracy	- Sampling data degeneracy issues. - Particle depletion during sampling process.
Graph SLAM	- Can provide better accuracy and consistency in its estimating technique. - Able to consume and process large area of mapping.	- high computational cost

2.8 Uncertainty in railway localization performance

Once the methods, sensors, and fusion approaches for localizing a vehicle have been identified, it is crucial to recognize the obstacles, potential factors, and uncertainties that affect location accuracy. The primary concerns are accuracy and the level of assurance provided by the position. The following four key elements, which contribute to a vehicle's position error, are outlined: sensor hardware, environment, information sources, and the positioning estimators [66].

2.8.1 Method Performance Uncertainty

Position estimation methods can introduce errors in several ways. To ensure accurate and reliable results, it is crucial to validate the accuracy and integrity of the received information before using it for positioning algorithms. Data validation should be done as early as possible, preferably as soon as the data is obtained from sensors, to avoid any inaccuracies or outliers that may affect the estimated positions and reduce the system's accuracy [2]. Additionally, the timeliness and consistency of processing data in the time domain are essential for stable localization. Careful handling of data fusion from multiple sources is another important factor to consider [43]. Proper selection of initial conditions and performance parameters for positioning algorithms is also critical, particularly for data fusion methods. Accurate settings ensure optimal performance of the positioning algorithms, while incorrect settings can result in increased positioning errors. For example, in the extended Kalman filter algorithm, correctly defining the process-noise-covariance and initial-estimate-covariance can lead to an optimal gain, which effectively weights the current state measurement and forecast state. An improperly specified setting, on the other hand, can lead to positioning errors.

2.8.2 Sensor Hardware Uncertainty

Sensors have limitations in terms of what they can detect, which are primarily influenced by two factors. The first factor is the physical laws that dictate the range and resolution of a sensor. The second factor is the unpredictable noise that interferes with sensor measurements, limiting the amount of information that can be extracted from them. It is important to physically test the hardware systems before use to ensure they are reliable, as any faults or failures can cause inaccuracies and serious problems in a real-world application [122]. Proper sensor selection is also critical, taking into account factors such as the sensor's characteristics and the application's requirements. For example, vision sensors like ASUS stereo cameras may not be suitable for outdoor use due to their inability to function correctly in direct sunlight. Other types of noise, such as image jitter, signal gain, and blurring, can also reduce the usable content of a colour

image. In railway applications, sensor specifications such as resolution, frequency, and sensing range should also be considered. For instance, if a stereo camera's maximum sensing range is only 10m, it may not be able to detect objects moving at the high speeds of a vehicle, causing blur motion in the camera imaging.

2.8.3 Pre-process Uncertainty

The accuracy of position estimation depends on the quality and quantity of information sources used. For safe rail vehicle positioning, it is important to have real-time, reliable, and accurate information. The rate of transmission of each technology is an important factor, as some received signal ranges may become outdated by the time they are used, making sensor fusion difficult. A new fusion strategy based on Kalman filter is proposed in [123] to overcome these challenges. In addition, the inherent accuracy of each technology can cause position error to range from a few meters for GPS to several hundred meters for GSM-R. Depending on the sensor characteristics and functionalities, stand-alone or fusion sensors may be used.

2.8.4 Environment Uncertainty

Another factor related to the environment that affects positioning estimation is the presence of obstacles or reflections. Obstacles, such as buildings or mountains, can obstruct the line-of-sight between the receiver and the signal source, leading to a reduction in signal strength or complete loss of signal. Reflections, on the other hand, occur when signals bounce off surfaces and create multiple paths, resulting in signal interference and degradation. In railway environments, metal objects, such as tracks, tunnels, and bridges, can also reflect and distort signals [124]. Therefore, it is essential to consider the environmental factors that may influence the accuracy and reliability of the sensor measurements when selecting and configuring the positioning system. In some cases, it may be necessary to use additional techniques, such as signal processing or signal strength modelling, to mitigate the effects of environmental factors and improve the accuracy of position estimation.

The main reasons of inaccuracy and errors that effect position estimation is depicted in the following figure:

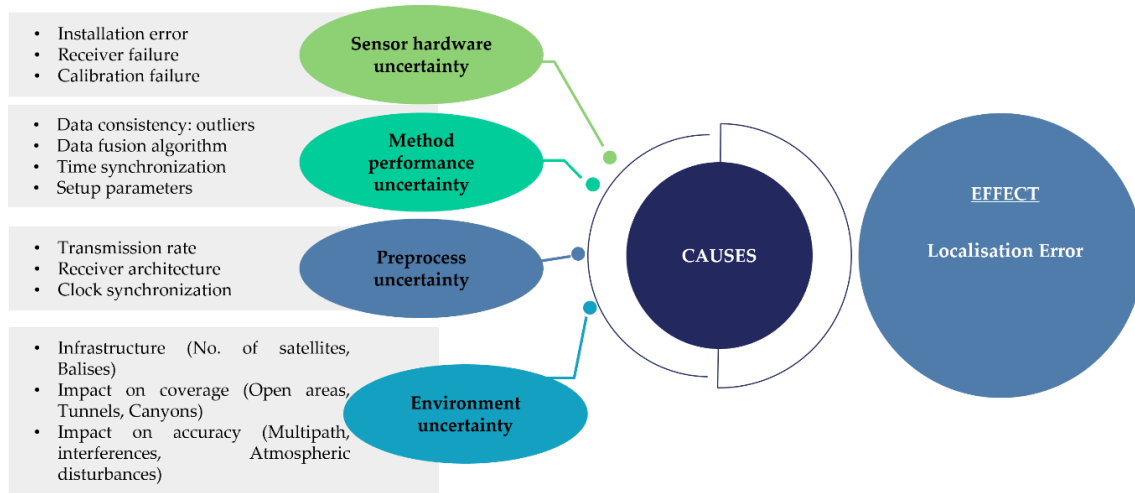


Figure 2-7 Fishbone diagram showing factors contributing to Location errors.

All the stated factors give rise to uncertainty. As robots are moving away from factory floors into increasingly unstructured environments, the ability to cope with uncertainty is critical for building successful robots. Because of this knowing the uncertainty sources and decreasing the impact of them as much as possible would lead to a result with more accuracy.

2.9 Robot Operating System (ROS)

All the experiments of this research are based on the ROS (Robot Operation System). It is a framework and set of tools that provides the functionality for programs to communicate with each other either on the same computer or over a network. Numerous features, including 2D planning and navigation, robot manipulator planning and navigation, mapping, visualisation, and more, are included into ROS. In this research, the version of “ROS-Kinetic” is utilized which is compatible with Ubuntu 16.04. The PC in the mobile robot is also integrated with ROS-Kinetic.

Some of the important concepts of ROS are including nodes, messages, topics, and services. Nodes can be thought as “software module” that contain particular functions. They are executable within ROS package. The term “node” is from the visualization of ROS system at runtime. Nodes are communicating with each

other by passing messages. A message is a simple data structure. The name of a message is formed by the name of the package plus the name of the *.msg* file. For example, RTAB-Map has a node which can produce visual odometry information, the message it published is called *rtabmap/odom*. ROS introduced a concept called topic. A node sends out a message by publishing it to a given topic. A node can specify what topics to subscribe, and all the messages belong to that topic, will be subscribed. Each module in ROS exists in the form of package, and the biggest advantage of ROS is that all function packages can be used universally, and the communication of all nodes can be transmitted in the form of topic messages. There are functions and commands in ROS that they are utilized frequently during the experiments. *roslaunch* is the command that can open the launch file which enables the executable of multiple nodes. Another command is called *rosbag* which can record ROS topics during a certain period of time. It can also republish the messages from a bag file. *rosbag* enables us to collect raw sensor data into a bag file and play it back through algorithm with different settings and configurations to debug.

Two important and most used tools in ROS are Rviz and Gazebo. Rviz is a 3D visualization tool for ROS using plugins for many kinds of available topics. With a topic as input, it visualizes that topic based on the message type being published, which allows us to see the environment from the perspective of the robot. Gazebo is the most popular simulator to work with ROS with good community support. The ability of a robot could be evaluated and tested in a customized 3D scenario without any harm to the real robot.

2.9.1 ROS Enhancement Proposal

When executing ROS in mobile platforms, two important ROS Enhancement Proposals (REP) should take into consideration: REP 103 (Standard Units of Measure and Coordinate Conventions) and REP 105 (Coordinate Frames for Mobile Platforms). Standards about preferred conventions of units and coordinate applied within ROS are provided in REP 103. The ROS convention for axis orientation of a body adheres to the right-hand rule and uses SI (international system of Units) by default. In the tests and experiments related to this research,

all topics from different packages and nodes, follow REP 103 standards. The naming standards for mobile robot frames and the connections between them are laid out in REP 105. The frames' names are:

base-link, odom, map, earth

The coordinate frame that attached to the mobile robot base named *base-link*. This frame is often located at the gravity centre of the robot. A world-fixed coordinate frame which usually stays at the start position of robot is called *odom*. Based on odometry sources, the pose of the robot in the *odom* frame is calculated. The pose of a robot in the *odom* frame is assured to be continuous, without discrete jumps. The short-term local reference provided by *odom* frame is precise, but it drifts over time. *Map* frame is also a world-fixed frame that stays at the start position of robot and is defined as long-term global reference. Based on the sensor observations, the pose in this frame is calculated, because of this it is not continuous and can change in discrete jumps at any time.

The coordinate frame at the origin of Earth Centre is named *earth*. It is designed to implement the communication between several robots. In this research, only one robot is utilized. Thus, the *earth* frame is not defined. The attached frames are shown in a tree representation, which means each frame can only have one parent frame but multiple child frames. Except for the earth frame, the rest three frames are attached by transforms as follows:

map → *odom* → *base link*

Except these four conventional frames, the other frames can be defined such as the frames related to the onboard sensors. The position of an onboard sensor can be defined by creating the static transform between its frame to the base link frame.

2.10 Research Gap

According to the highlighted information, the use of automation and computational intelligence techniques can significantly enhance the efficiency and effectiveness of maintenance. A significant proportion of the advancements

in robotic and autonomous systems for railway inspection and maintenance have been focused on rolling-stock and rail-track, accounting for 56% and 28% respectively [125]. For example, miniature robot models have been utilized in various maintenance tasks, such as detecting faults [8], cleaning tracks, and even performing repair jobs using 3D printing technology [9]. Wheeled robots with manipulators have also been deployed for remote and hazardous tasks, including tunnel inspections [126] and cleaning nuclear reactors [127]. These robots can also be useful for inspecting and maintaining railway tracks [11].

Some of these mobile manipulators can only navigate on the railway track or around the railway, or some of them lack a manipulator for repair or close inspection. They are only designed for inspection or maintenance tasks, which a significant number of them are for track inspection rather than maintenance. Considering the railway environment, there are much attention to apply mobile manipulators (a combination of a wheeled robot, and a manipulator) for the aim of facilitating the process of inspection and repair on and around the railway track. They can operate at long distances and perform remote operational tasks.

In this case the novelty of this research is in enabling the capability of navigation and positioning of a new autonomous inspection and repair vehicle via the application of effective localization knowledge. This research is new in terms of application in railway and transport systems, by deploying the off-the-shelf techniques and existing knowledge within the new robotic inspection and repair system in railway network environment. Applying the RIRS and implementing localization, mapping, and navigation modules within the system, makes it novel and capable of performing predefined maintenance tasks on the railway track with higher accuracy.

The scale problem is a characteristic of the location problem and refers to resolution scales in the transition from vehicle navigation to refined defect positioning. In this regard, the bigger scale is considered for navigation from the initial location to the specified location (road-rail conversion), and the smaller scale is considered for payload positioning.

The experiments conducted intended to replicate realistic railway conditions but were restricted in time and length due to the limited availability of operational railway tracks. However, up to 20 km navigation is expected for future vehicles.

The key attribute of the maintenance systems is localization and navigation with high accuracy if they are aimed to conduct the inspection and repair tasks. Despite the vast introduction of autonomous solutions to maintenance and repair systems in the railway industry, the accuracy and repeatability of positioning of these mobile maintenance systems have not received the same level of attention or maturity. In section 2.4, research and works related to train and rail vehicle localization approaches were cited. The accuracy of maintenance systems on railway tracks can vary widely based on the specific technology, implementations, and purpose of the system. Most of the off-the-shelf maintenance systems, work to several meters accuracy; track geometry refinement works at about meter-level in some cases. Localization of the rail maintenance robots at the absolute level (cm-level) is the gap of knowledge that this research addressed. The output of this research determines the location of RIRS on the railway track with absolute accuracy and ensures that the RIRS is capable of performing maintenance tasks in a real railway environment. This research represents the usability of robotic systems for railway track inspection and repair tasks.

The imperative to empower new autonomous inspection and repair vehicles with seamless navigation and precise positioning capabilities necessitates integrating state-of-the-art and efficient relevant navigation and positioning methods. By applying efficient techniques in navigation and positioning, these vehicles can transcend traditional limitations and operate with unprecedented accuracy, efficiency, and adaptability. This research ensures that the autonomous inspection and repair system not only navigates in complex environments but also positions itself with an unparalleled level of precision. This research answers the question of how the capability to navigate and position could be enabled for new autonomous inspection and repair vehicles using the latest, efficient, and relevant navigation and positioning methods.

Therefore, detailed research is required to investigate the localization and navigation of these systems as well as the uncertainty sources that impact the railway vehicle positioning performance. In this way, it would be possible to provide effective inspection and repair tasks.

2.11 Summary

This chapter started by describing the importance of railway maintenance to show how important it is in the railway industry. It was also noted why an autonomous maintenance system is required, and why its localization at the absolute level matters. Then the impacts of railway assets on the railway system's operations were investigated. A comprehensive review of mobile robot localization approaches in general and localization methods in railway environments in specific were provided. Subsequently, localization sensors, fusion approaches, localization, and mapping methods, especially SLAM approaches, their functions, and formulation were reviewed broadly. Finally, a brief narrative of ROS, and ROS Enhancement Proposals was indicated.

3 METHODOLOGY

For achieving the stated research aim and objectives a number of scientific and technical steps are considered. First, a description of the hardware of the state-of-the-art robotic inspection and repair system used in this research is illustrated, alongside the on-board sensors and their specifications. Then the concepts of accuracy and precision of the GPS data are examined. Afterward, the navigation module and its sub-components for having an autonomous system are reviewed. Afterward, different, and separate methods and tests are implemented based on the system trajectories. This includes the implementation of EKF, RTAB-Map, and local and global path planning for the off-track phase. Here, static object detection and collision avoidance are also implemented for safety purposes.

For an on-track scenario, autonomous navigation based on GPS odometry, alongside using RTAB-Map as the provider of the local vehicle's location and the environment map is implemented. Finally, object detection based on feature matching, and distance estimation based on fusion of data from 3D-LiDAR and RGB-D camera is executed.

3.1 Robotic Inspection and Repair System (RIRS)

The RIRS applied in this research shown in Figure 3-1. The RIRS main component is a Warthog Unmanned Ground Vehicle (UGV) from the company of Clearpath Robotic Inc [128] . It is a four-wheel drive unmanned ground electrical vehicle with a number of on-board sensors presented in Figure 3-1. It can run in most terrains at a maximum speed of 18 km/hr. The Warthog fully supports ROS and can be equipped with a variety of payloads, including on-board sensors such as 3D LiDAR, RGB-D camera, IMU, wheel odometer, RTK-GPS, and robot manipulator. It is equipped with on-board & remote emergency stops button to ensure safety and is mounted with a light indicator to present basic movability status. There is a main on-board PC for driving the vehicle as well [129].



Figure 3-1 Warthog vehicle and its on-board sensors

To enable RIRS autonomous functionality, multiple sensors are used to provide real-time perception including localization, motion pose, and surrounding detection. The following table presents the specifications of the sensors. For the localization, the RTK-GPS module provides a geo-location of the vehicle globally. The 3D LiDAR provides 3D perception of the surrounding environment for a safe navigation, as well as distance to the nearby objects. RGB-D camera gives RGB image of the track-side objects and a depth perception of the track and obstacles. Moreover, the PTZ (Pan-Tilt-Zoom) camera, which can rotate 360°, renders the view of the surroundings to the human operator through the HMI (human interface) or web services. For the motion pose, the Inertial Measurement Unit (IMU) detect the inertia, acceleration in 6 axes. Wheel odometer also determines the vehicle's change in position relative to some known position. Table 3-1 summarises the available sensors in RIRS.

Table 3-1 Warthog instrumentation hardware specifications

Devices	Brand & models	Key specification	Signal type

3D LiDAR	Velodyne Puck LiDAR (VLP-16)	Range: 100 m Number of laser/receivers: 16 pairs Field of View (vertical): +15° to -15° Field of View (Horizontal): 360° Rotation rate: 5Hz to 20Hz	Point cloud
RTK-GNSS	NovAtel SMART6-L [130]	Maximum data rate: 20Hz Horizontal positional accuracy: 2-3 cm	Decimal format GNSS signal
IMU	Lord 3DM- GX5-25	Pitch-roll static/dynamic accuracy $\pm 0.25^\circ/0.4^\circ$ Sampling rate: 100Hz Pitch-roll static/dynamic	Acceleration, angular velocity, and roll, pitch, yaw
RGB-D camera	Intel Real- sense D435i	Range: 0.3 m to 3 m Depth frame rate: Up to 90 fps Depth Field of View (FOV): $87^\circ \times 58^\circ$ RGB frame rate: 30 fps RGB sensor FOV (H $\times$ V): $69^\circ \times 42^\circ$	RGB & Depth image & point cloud
PTZ environmental camera	AXIS M5525- E PTZ Network Camera	Resolution: 1920 x 1080 Maximum frame rate: 30 fps Field of view (Horizontal): 61.8° to 6.7° Pan capability: 360°	RGB video & Images

		Tilt capability: 90° Zoom capability: 10x	
--	--	--	--

The RIRS can operate in two modes of navigation: on-track and off-track, shown in Figure 3-2. For the on-track navigation mode, after a road-rail conversion, RIRS is placed on the trolley and power transmits through friction, and belt and pulley mechanism. While for off-track navigation, RIRS runs along the railway track without the track trolley. Though in on-track navigation, the motion of the Warthog is only limited to forward and backward, in off-track navigation RIRS can both move in forward or backward direction and rotate in left or right. The on-track and off-track operations result in complexities. For example, the vehicle dynamic behaviour will be different in these two modes. In off-track operation, the pneumatic tires will absorb most of the vibration, whereas, in on-track navigation, the RIRS will run smoothly because of the rail wheel (but in high speed, it would lead to vibration).



(a)



(b)

Figure 3-2 (a) On-track and (b) Off-track navigation

3.2 Accuracy and Precision

Accuracy and precision are often used to describe how good is the position acquired by GPS receiver. However, a distinction should be made between accuracy and precision. Accuracy is the degree of closeness of an estimate to its true but unknown value and the precision is the degree of closeness of observations to their means. In terms of checking the accuracy of the available

Warthog GPS system, first data were sampled and then analysed through NovAtel Application Suite [131] and Inertial Explorer [132] software respectively. For working with NovAtel Application Suite and capturing the logs, it was needed to connect to the NovAtel SMART6-L over a serial cable, while it was situated in a stationary state. Therefore, the GPS sensor, situated on the top of the base station, shown in Figure 3-3, was considered for this test. The base station was placed in one location far from any objects and any GPS obstructions. The cable related to the SMART6-L has a COM1 serial end which was used for communication. A USB-Serial adapter was needed to do the communication with the PC. After connecting the PC to the base station port, data was captured, and then through the Inertial Explorer software, the following analyses were extracted.

Figure 3-4 (a) shows the estimated position accuracy plotted at one standard deviation related to the three GNSS values (latitude, longitude, altitude). The estimated standard deviation refers to a confidence range where the receiver estimates its antenna is situated in real-time, relative to the truth position (WGS84 global datum). According to this plot, the accuracy of latitude and longitude were about 1.5 cm (less than 2 cm) and altitude 5 cm.

The next plot (b) indicates the criteria of PDOP which illustrate the position dilution of precision. A value that quantifies the satellite geometry. The smaller value of this parameter indicates better satellite geometry. It was identified that poor accuracy, high PDOP, and float solutions can often be explained by the lack of available satellites. The sample rate of GPS technology is in Hz and indicates the number of times per second that the chipset and satellites communicate to establish the system location.

Figure 3-4 (c) shows that during the whole sampling period, the number of satellites is 7, exceeding the minimum number of RTK level which is 5 [77]. The precision of GPS data is also examined in the result section.

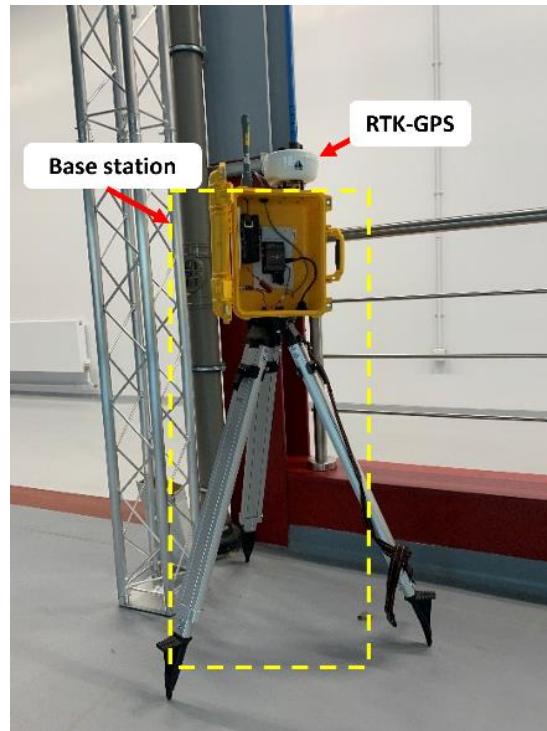
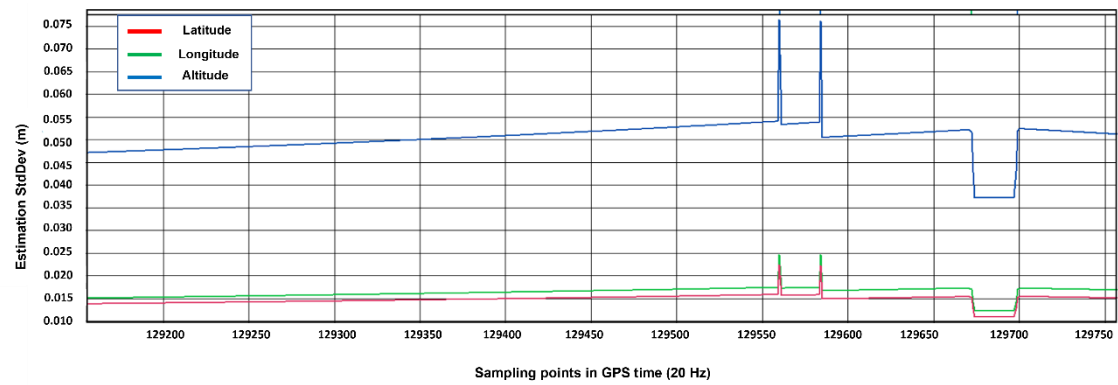
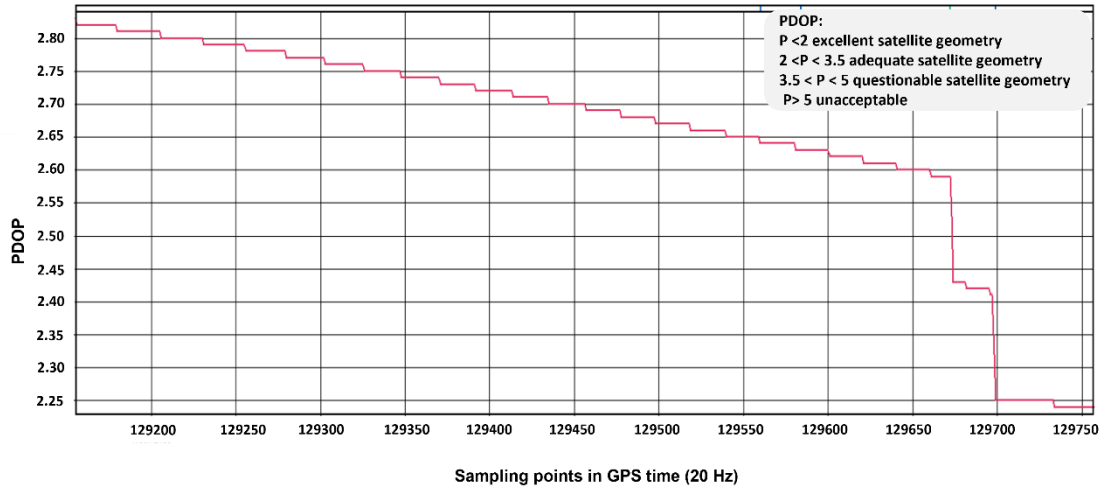


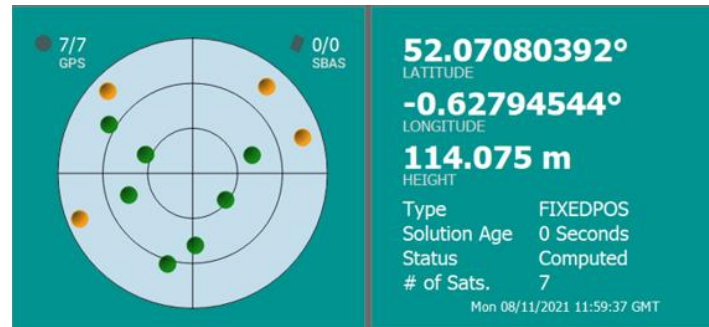
Figure 3-3 Base station with the on-board RTK-GPS sensor



(a)



(b)



(c)

Figure 3-4 RTK-GNSS signaling accuracy estimation (a) STD of GPS data; (b) PDOP value; (c) Number of Satellites

3.3 Autonomous Localization and Navigation

Considering the aim of this thesis, the autonomous movement of the RIRS to job location divides into two phases: first, the transit of the RIRS from its initial location to the goal location; and second the navigation of the vehicle on the railway track from its initial location to defect location [133]. These phases were implemented by combining the on-board sensing data, and prior information about the vehicle and infrastructure assets and objects in the environment. In this section, each of these phases are investigated separately in more detail.

3.3.1 Navigation Module

Localization and navigation are one of the most challenging competencies required for an autonomous mobile robot. For the successful operation of the autonomous vehicle, four main sections should work properly including perception which interprets the information which are sent by sensors to extract meaningful data and recognize objects, places, and so on in the environment; map building and localization modules create a numerical model of the environment around the robot, and estimate the robot's position relative to the map respectively; path planning specifies the movement of the robot for reaching to its goal, without colliding with the objects; and motion control provides the execution of the planned movements to achieve the desired trajectory . Figure 3-5 represents the main subsystems.

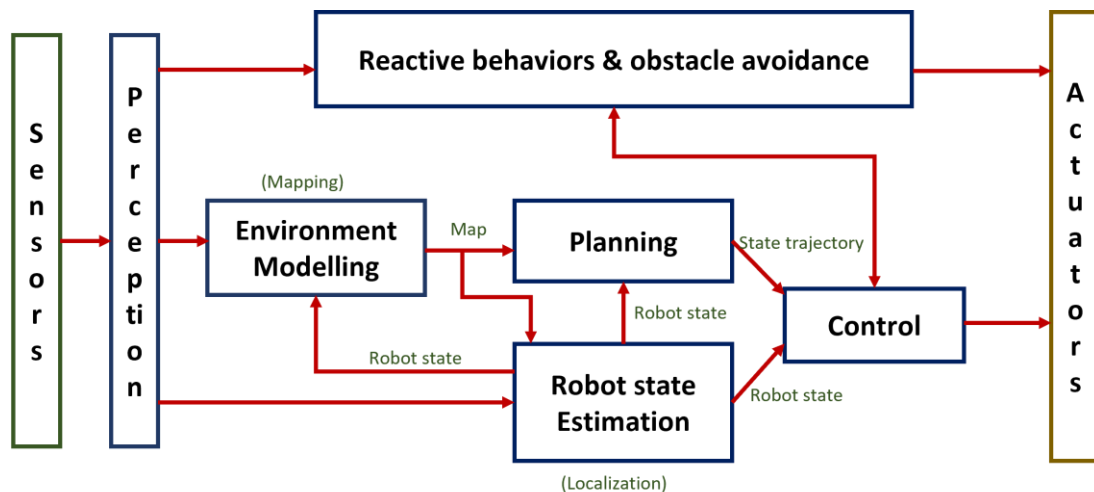


Figure 3-5 subsystems of an autonomous navigation system

The navigation module encompasses all the navigation-related software, which comprises deliberative driving, path planning, and motion planning. The module's primary goal is to use localization relative to the starting point to drive the robot autonomously throughout the track. With localization, the module can define a starting and an endpoint on the map, compute a plan between them, and execute the plan using the control system.

The navigation support for ROS is used as the foundation for this objective. ROS navigation support is integrated into the "*move_base*" package, which provides

complete autonomous navigation capabilities with internal navigation control software and behaviour definition. It has several inputs, including the robot's state, localization, and perception of the world. Figure 3-6 shows the architecture of the navigation in ROS.

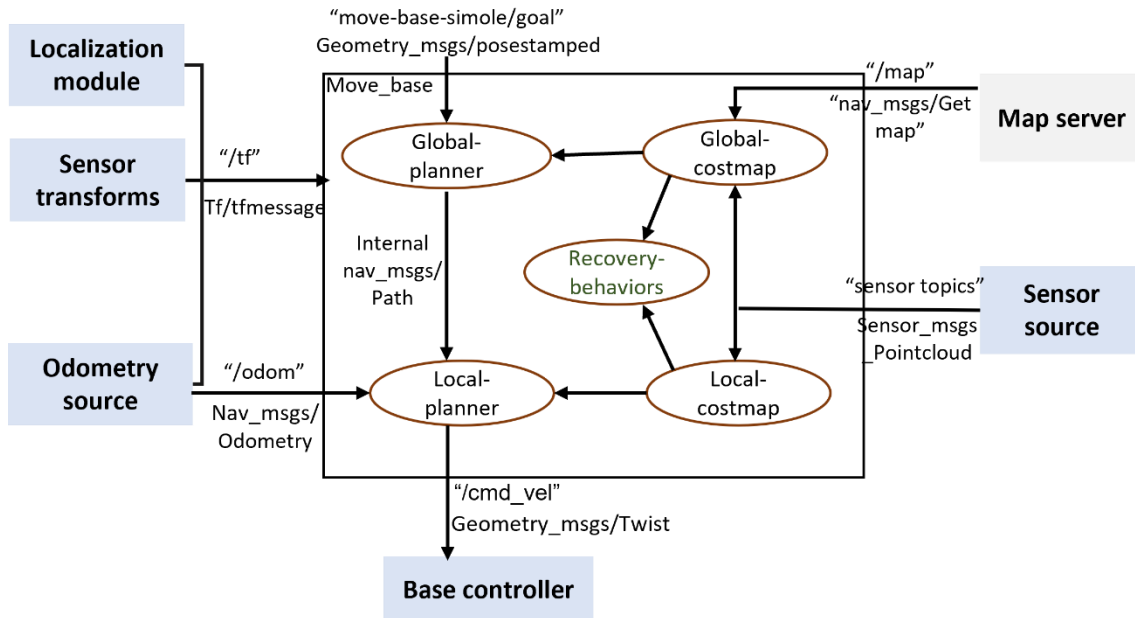


Figure 3-6 Subsystem of a ROS navigation stack

By examining the architecture, it becomes clear that once a goal pose is defined for the system and then localization, robot state, and world perception information are appropriately shared, the navigation module will publish action command (*/cmd_vel*) to guide the robot along a predetermined path. This navigation module allows for the definition of a global planner and a local planner for path planning and even allows for recovery behaviours to address any issues that may arise. The stack is primarily used because it publishes both local and global cost maps, which provide a comprehensive view of the world within the ROS environment. Each of the subsection of this architecture are defined in summary as follows:

Global cost-map: regions around obstacles which are used by the global planner to compute paths to navigate from one point of the map to another without getting too close to the static obstacles registered during mapping.

Local cost-map: similar to the global cost-map, but it is smaller and moves with the robot, it is used to take into account new features that were not present in the

original map. It is used by the local planner to avoid obstacles, both static and dynamic, while trying to follow the global path computed by the global planner.

Global Planner: this planner is in charge of calculating a safe path in order to arrive at that goal pose. This path is calculated before the robot starts moving, so it will not consider the readings that the robot sensors are doing while moving. It provides a fast interpolated navigation function that creates plans for a mobile base. The planner operates on a cost-map to find a minimum cost plan from a start point to an end point in a grid, such as using Dijkstra's algorithm. The global planner generates a series of waypoints for the local planner to follow.

Local Planner: this planner chooses appropriate velocity commands for the robot to traverse the current segment of the global path. It combines sensory and odometry data with both global and local cost maps. It can also recompute the robot's path to keep the robot from striking objects, yet still allowing it to reach its destination. Global and Local cost-maps as well as global and local planners are shown in Figure 3-7.

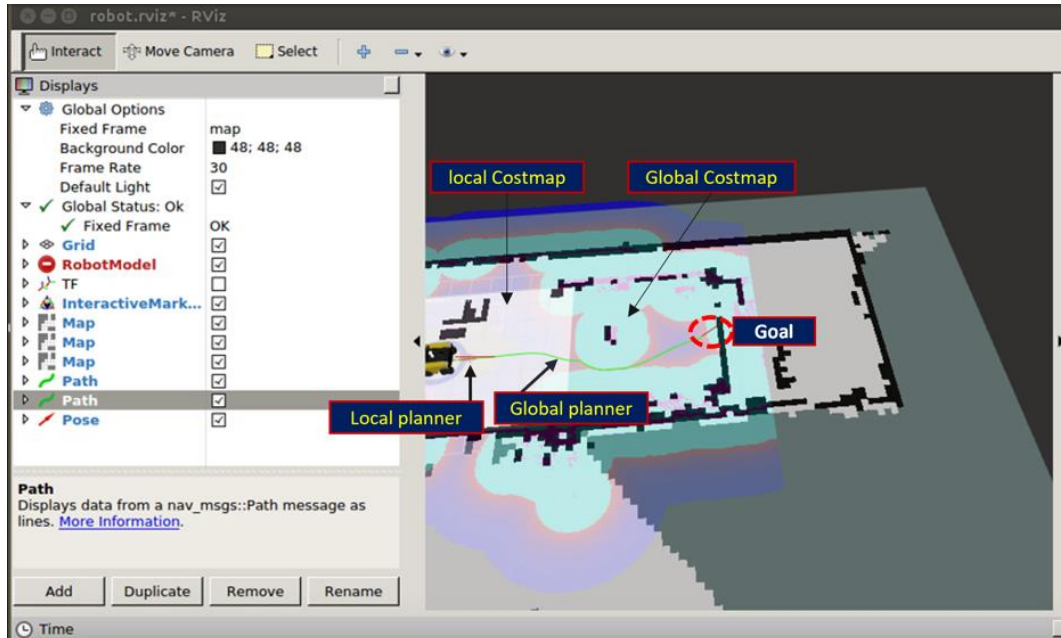


Figure 3-7 Global and local cost-maps; Global and local planners

3.3.2 Off-track Localization and Navigation

In the off-track navigation, the Warthog UGV transits from the initial location to the goal location autonomously with the help of its sensors and actuators (devices that convert electrical signals into physical actions or movements). This navigation is using GPS to localize the vehicle globally. For this purpose, fusion of GPS and inertial navigation systems on their own are not enough, because GPS accuracy is affected by a number of factors such as GPS satellite signal (number of visible satellites, satellite geometry, satellite position, signal delay, etc), GPS receiver, and environmental factors (tunnels, forests, near the buildings, etc). To solve this issue and provide a promising estimation of the pose of the vehicle, the idea of merging the RTAB-Map odometry information with the other sensory data was implemented. In this way, the vehicle was able to localize itself, even when GPS was not working properly. In the following sections, first off-track path planning, and collision avoidance procedure are illustrated. Then the function of RTAB-Map approach and its main features are cited, and finally, navigation based on the fusion of GPS and SLAM is discussed.

3.3.2.1 Off-track Path planning

Path planning is a component of autonomous navigation, and it involves searching for a path for an autonomous system to follow within a specified environment [134]. The goal of path planning is to guide the vehicle towards the destination, taking the closest possible route, and assuming that the map of the area is already available. It is important to know where the robot itself is and have a map of the given environment. It is also important to find the optimized route among the various routing options, avoiding obstacles and navigating safely. An illustration of the path planning is shown in Figure 3-8.

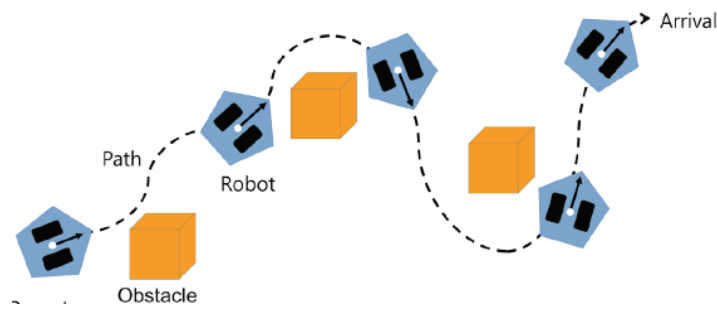
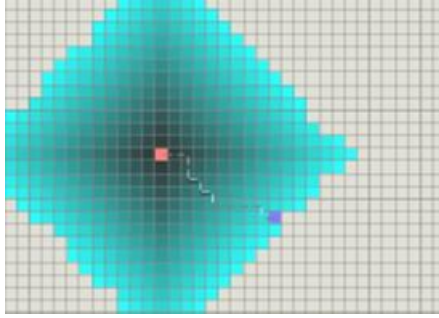


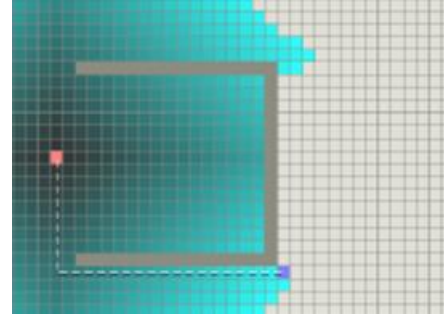
Figure 3-8 Path planning

Selecting the appropriate navigation technique is crucial for a robot's path planning in both simple and complex environments. Mobile robot navigation is divided into two categories: global and local navigation [134]. Global navigation refers to the ability to determine the position of objects in the environment in relation to a reference axis and navigate towards a predetermined destination. On the other hand, local navigation involves identifying the dynamic conditions of the environment and establishing positional relationships among different elements [134]. For global path planning, all the environmental information context is known to the robot before starting. In contrast, for the local path planning, almost all the information about the environment is unknown.

The Dijkstra algorithm, depicted in Figure 3-9, is a crucial global path planning algorithm [135]. Its purpose is to determine the shortest route between two nodes, namely the initial and goal location, by visiting every node in the graph and discovering the shortest distance to each one. If a grid map is considered a graph, where cells are nodes and each cell transition has a cost, the Dijkstra algorithm can be used for path planning.



a)



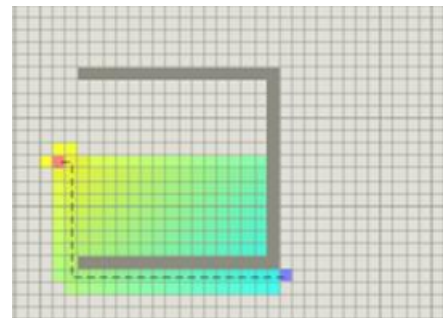
b)

Figure 3-9 Dijkstra algorithm a) without obstacle b) with obstacle

Figure 3-10 illustrates another path planning algorithm called the A* algorithm [136]. Similar to Dijkstra, the A* algorithm traverses the graph to locate the shortest path to the goal. However, the A* algorithm's graph exploration is guided by a heuristic function. This approach makes searching through the graph more efficient, although a heuristic function for the goal is required.



a)



b)

Figure 3-10 A* algorithm a) without obstacle b) with obstacle

There exist different global planners. Their selection typically depends on the setup (the type of the used robot, the environment it navigates in, and so on). The most important ones include *Navfn*, *carrot planner*, and *global planner* [137]. In this research *Navfn*, and *global planner* are applied. The *Navfn* planner is the most commonly used global planner for ROS Navigation. It uses Dijkstra's algorithm to calculate the shortest path between the initial pose and the goal pose. On the other side, *global planner* is a more flexible replacement for the *Navfn* planner. It allows us to change the algorithm used by *Navfn* (Dijkstra's algorithm) to calculate paths for other algorithms.

Once the global planner has calculated the path to follow, this path is sent to the local planner [138]. The local planner, then, will execute each segment of the global plan (the local plan can be imagined as a smaller part of the global plan). Therefore, given a plan to follow (provided by the global planner) and a map, the local planner will provide velocity commands in order to move the robot. Unlike the global planner, the local planner monitors the odometry and the laser data and chooses a collision-free local plan for the robot. Therefore, the local planner can recompute the robot's path in order to keep the robot from striking objects, yet still allowing it to reach its destination. The most important local planners are including *base_local_planner*, and *dwa_local_planner* [139], [140]. In this research *dwa_local_planner* is applied. It provides an implementation of the Dynamic Window Approach (DWA) algorithm. It is basically a rewrite of the base local planner's DWA option, but it is a lot cleaner and easier to understand. This is the most commonly used option for local planners.

3.3.2.2 Off-track Collision avoidance

Integrating obstacle avoidance techniques with path planning results in a more reliable path planner module. Real-time obstacle avoidance involves selecting the optimal direction among several non-obstructed options, while avoiding obstacles, no-go zones, and negative objects like potholes. Safe navigation of an autonomous system necessitates the ability to avoid obstacles while in motion. Vehicles must be able to navigate their surroundings securely. Obstacle avoidance can be more difficult to accomplish than path planning.

There are two kinds of obstacles that must be avoided in the environment: (i) static obstacles and (ii) mobile obstacles. In outdoor RIRS, static obstacles such as trees, buildings, and barriers must be avoided, as well as mobile obstacles such as pedestrians and vehicles that may unpredictably appear in the path. The state of each object must be updated at every instance of time, and this task becomes more challenging when the environment is cluttered with obstacles. It should be noted that, for the off-track navigation phase in this research, only static object avoidance is considered, and dynamic objects are outside the scope of this study.

The perception module in a navigation system is primarily responsible for two tasks: obstacle detection and scene understanding. To accomplish these tasks, point cloud data acquired from LiDAR sensors is utilized to interact with the environment, estimate distances, and avoid obstacles. Additionally, the camera captures images that are analyzed for feature extraction and object and landmark recognition in the environment, aiding in location mapping.

3.3.2.3 RTAB-Map SLAM

For localizing and mapping of the environment, a pose graph-based algorithm named RTAB-Map was applied. RTAB-Map, short for Real-Time Appearance-Based Mapping. Appearance-based means that the algorithm uses data collected from vision sensors to localize the robot and map the environment. It is a graph-based solution for full SLAM problem which is an intuitive way to address the SLAM problem. A graph-based SLAM problem builds the graph of the environment using the raw sensor measurements. The structure of the map is a graph with nodes and links. Nodes contain information of corresponding robot poses estimated by odometry and observations and the edges or links contain the measured transitions between them. Links represents the relationship between two nodes in terms of spatial and temporal connectivity. It defines how the robot has moved or transitioned from one node to another. Measurements are either acquired from odometry or loop closure detection. In the following table, sensors supported, front-end, and the back-end approaches used in RTAB-Map are provided.

Table 3-2 RTAB-Map with the supported sensors, front-end and back-end algorithms

System	Sensors Support	Front-end	Back-end
RTAB-Map	- RGB-D or stereo camera - 2D or 3D LiDAR	- Visual odom: GFTT/ BRIEF (for feature detection) - NNDR (for feature matching) - PnP RANSAC (for motion estimation).	- GTSAM (default) - G2O, TORO

		- LiDAR odom: ICP	
--	--	-------------------	--

RTAB-Map provides robust and low drift odometry information. It means when a loop closure happens, it can correct most of the odometry drift in the map. Because it might happen that the robot could not properly localize itself in the map, maybe because there aren't enough discriminative features in the environment. So, it is expecting that during that time also odometry drift be minimized as possible. Another positive feature of RTAB-Map is that it can provide robust localization. It might happen that the robot fails to localize itself or provide a wrong localization because of the dynamic environment which is often due to the presence of moving objects or obstacles, illumination change or even repetitive environment. In this case, the robot cannot identify whether it is a pre-visited location or not.

In terms of the structure of graph-based SLAM approaches, they are composed of two parts including front-end and back-end, shown in Figure 3-11. In RTAB-Map, the front-end deals with constructing the graphs utilized for feature optimization techniques, leveraging the robot's odometry and sensory measurements. This involves interpreting sensory data, generating the graph, and continuously augmenting it with new nodes and edges as the robot moves through the environment. This stage considers both odometry and loop closure constraints. The former may arise from wheel encoders, LiDAR, or visual odometry.

The back end of RTAB-Map comprises two main components: graph optimization and the creation of an occupancy grid based on graph data. The completed graph, containing all the constraints, serves as input to the back end, while the output is the most likely configuration of the robot's poses and map features. The back end operates as an optimization process, using all the constraints to identify the system configuration that produces the lowest error.

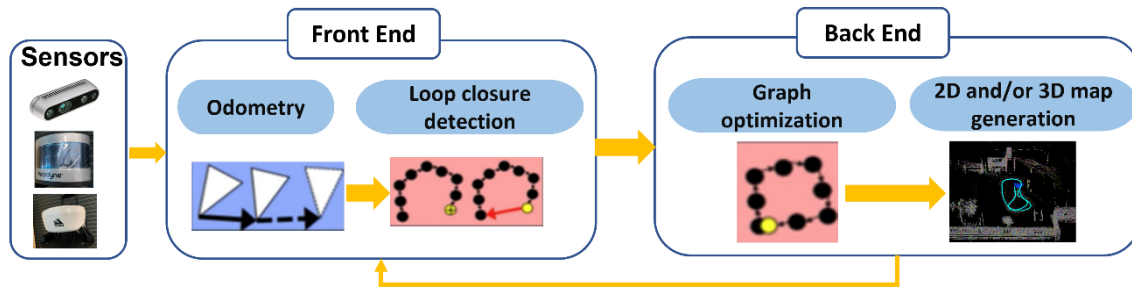


Figure 3-11 Front-end and Back-end of Graph-based SLAM architecture

To compute the geometric constraint between loop closure images, RTAB-Map necessitates the use of an RGB-D camera or a stereo camera. As the robot continues to explore the surroundings, the map dimensions expand, resulting in an increase in memory requirements and processing time for state estimation. The high dimensionality of the SLAM methods poses a challenge, but RTAB-Map addresses this issue through a memory management technique that restricts the number of locations deemed potential candidates during loop closure detection. This approach is a critical aspect of RTAB-Map and enables real-time loop closure.

3.3.2.3.1 Odometry

The use of motion sensors to determine the robot's change in position relative to some known position is called odometry. This can be done by applying a simple sensor, such as wheel odometer, or a complicated one such as camera or a LiDAR. RTAB-Map implements an odometry node that is able of computing visual Odometry (VO) or LiDAR Odometry (LO) in case that there is no external source of odometry i.e., wheels odometry.

- **Visual Odometry**

The process of determining the position and orientation of a robot by analysing the associated camera images. For this kind of odometry information, the transform between the camera coordinate frame and robot frame are needed.

- **LiDAR Odometry**

The LiDAR odometry is very similar to the one presented in the visual odometry. In this case laser sensor data with the transform between the LiDAR coordinate frame and the robot's base frame, should be given as input together. In addition, external odometry like wheels odometry could be added to improve motion prediction as LiDAR odometry can lose its track if not enough features are detected. LiDAR is also less affected by lighting conditions compared to visual sensors. In low-light environments or situations with inconsistent lighting, LiDAR can provide more reliable odometry.

Estimating odometry with exteroceptive sensors such as camera and LiDAR can be very accurate when there are enough features in the environment. However, using only one sensing modality can be problematic and prone to localization errors if their tracked features in the environment are no longer visible.

3.3.2.3.2 Loop Closure and Proximity Detection

Loop closure is used to determine whether the robot has seen a location before or not. As the robot travels to new areas in the environment, the map is expanded, and the number of images that must be compared with the new images increase. This causes the loop closures to take longer but with complexity increasing linearly. RTAB-Map is optimized for large-scale and long-term SLAM by using multiple strategies to allow for loop closure to be done in real-time. The loop closure is happening fast enough that the result can be obtained before the next camera images are acquired. In Figure 3-12 loop closure is specified during the movement of the vehicle in the area. In the result section, more detail is investigated about this feature. RTAB-Map can also be used without a camera, but loop closure detection wouldn't work. In that case, there is a feature called proximity detection which is still working with scans. Therefore, some loop closures can be detected as long as odometry doesn't drift too much. Proximity detection does localization when the robot traverses back to the same location in a different direction. The yellow link in Figure 3-12 are indicative of this feature.

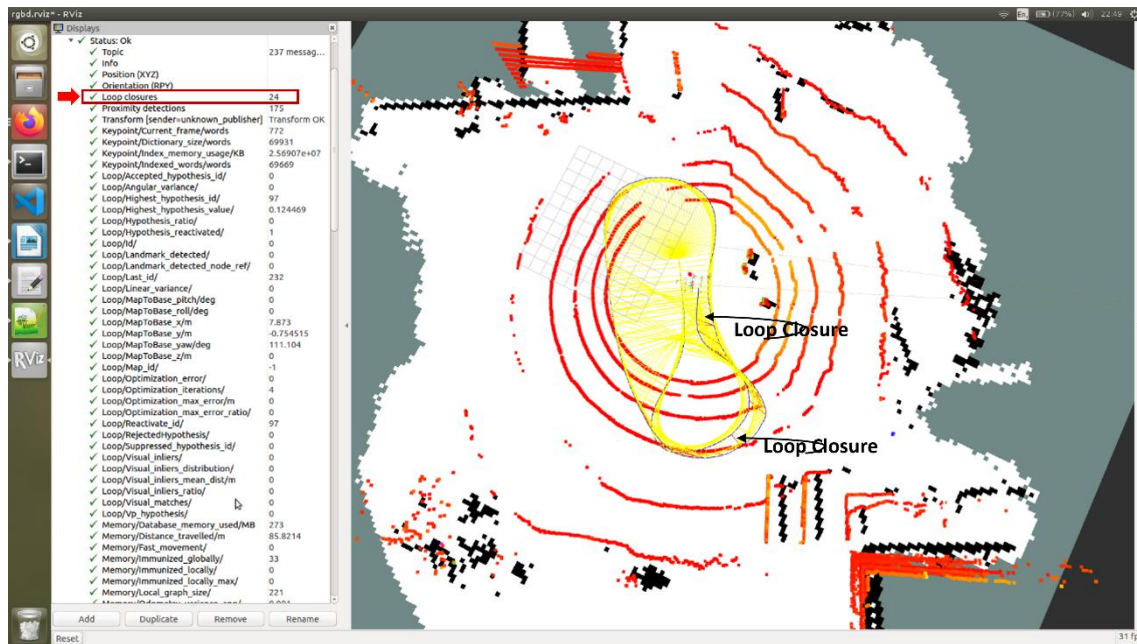


Figure 3-12 Loop Closure detection

3.3.2.4 Merging GPS with SLAM

Combining SLAM and GPS navigation has significant potential for practical applications [141]. By applying only GPS, high-accuracy large-scale outdoor mapping cannot be accomplished. Though there are high-precision GPS receivers like RTK-GPS, there must be a constant, unobstructed line of sight to several satellites for them to function correctly.

The accuracy of GPS-based position data may be compromised or non-existent in areas with limited visibility of the sky. The use of GPS is especially challenging in environments that include electrical devices, Overhead Line Equipment (OLE), tunnels, and varying times of the day (section 2.2). These situations are important when precise localization would be most valuable. While GPS data can be combined with other motion estimation systems, such as odometry or inertial sensing, this solution is not ideal. During a period of GPS outage, the growth of pose uncertainty is unbounded, making it impossible to achieve high-precision navigation. It is generally preferred to limit localization error, regardless of whether GPS is available or not.

Considering the field of SLAM, there are various ideas on how to bound localization without the aid of a pose sensor, and how to propagate high-precision

information through a map to enhance the estimates of both the robot's location and pose, when the vehicle traverses the area of high uncertainty. Having a Kalman filter, a high-quality IMU, and a suitable GPS are the first basic methods for having a navigation module. However, the fusion of GPS-IMU is not sufficiently robust and accurate enough to deal with complex and unstructured environments. The two main issues of GPS-based navigation are outages and accuracy restrictions. SLAM offers techniques for coping with protracted periods of error integration and continuously improving positioning estimates. By fusion of GPS and SLAM, there are several possible advantages, such as enhancing robustness in dealing with intricate and disorganized environments, augmenting scalability, and refining localization accuracy [90], [142].

3.3.2.5 Navigation Based on the fusion of GPS and SLAM

Besides the positive points examined about the fusion of GPS odometry information with SLAM, it leads to obtaining accurate and high-rate global pose data as well. As it is noted before, RTAB-Map is considered as the SLAM approach in this research. RTAB-Map solution, because of its complexity makes the rate of pose data slow (around 10 Hz) but fusing the output of RTAB-Map with GPS data, and inertial sensors of the robot leads to obtaining higher frequency and more accurate robot pose data. In this research, to overcome the disadvantage of the low data frequency from RTAP-Map and obtain a higher rate of frequency and accurate robot pose estimates, an EKF filter was utilized. In the real world, especially in navigation, the system models (observation and transition) are usually nonlinear. EKF is one of the early extensions of KF which copes with nonlinear models by re-linearizing them about the latest state estimate. In this regard, the *robot-localization* ROS package applies EKF to fuse sensory data at local and global levels [143]. It is a collection of non-linear state estimators for the robot's moving. Each of the state estimators can fuse an arbitrary number of sensors (IMUs, odometers, GPS receiver) to track the 15-dimensional state of the robot. This 15-dimensional matrix includes positions (x , y , z), orientation (roll, pitch, yaw), velocity ($\dot{x}$, $\dot{y}$, $\dot{z}$), angular velocity (roll', pitch', yaw'), and linear acceleration ($\ddot{x}$, $\ddot{y}$, $\ddot{z}$). The *robot-localization* state estimator

nodes accept measurements from an arbitrary number of pose-related sensors. It merges all the nodes with the following types of messages:

- *nav_msgs/Odometry* (position, orientation, and linear and angular velocity),
- *geometry_msgs/PoseWithCovarianceStamped* (position and orientation)
- *geometry_msgs/TwistWithCovarianceStamped* (linear and angular velocity)
- *sensor_msgs/Imu* (orientation, angular velocity, and linear acceleration) messages.

Any sensor that produces messages in any of these formats can be fed to the EKF to estimate the robot's position. Based on these measurements, the state estimators publish the filtered position, orientation, and linear and angular velocity (*nav_msgs/Odometry*) on the */odometry/filtered* topic. This package provides two state-estimation nodes which are based on EKF and another node for integrating GPS data into the estimator. One of the instances of the *robot-localization* package gets all the continuous data (e.g., wheel odometry and IMU) to produce locally accurate state estimate (*odom* $\rightarrow$ *base_link*), and on the other instances, fuse continuous data with non-continuous data (e.g., GPS) to provide an accurate and complete global state estimate (*map* $\rightarrow$ *odom*). Together, they will estimate the full *map* $\rightarrow$ *odom* $\rightarrow$ *base_link* transform chain. The frame called "*base_link*" is firmly attached to the robot. Although the robot's location in the "*odom*" frame will gradually change over time, it is reliable in the short run and should remain uninterrupted. The "*map*" frame, similar to the "*odom*" frame, is a fixed world frame of reference and provides the most precise global estimate of the robot's position. However, it is susceptible to sudden, distinct jumps, such as those resulting from the integration of GPS data. The received GPS data are in latitude and longitude format, so the *navsat-transform-node* transforms the GPS data into a frame that is consistent with the robot's world frame starting pose (position and orientation). In the following figure the setup related to the integration of sensors and the related messages and topics are depicted:

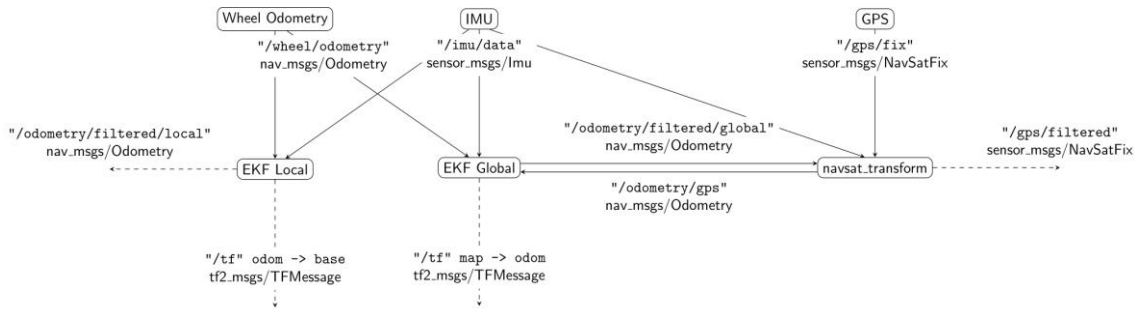


Figure 3-13 Nodes in robot-localization [143]

As it is reviewed, the aim is to merge RTAB-Map odometry information with the other global odometry information (GPS) to exploit the benefits and advantages of both systems and enable the system to estimate the pose of the robot even in GPS-denied environment. In the following figure, RTAB-Map odometry data is integrated with the other odometry data from wheel odometer, GPS, and IMU.

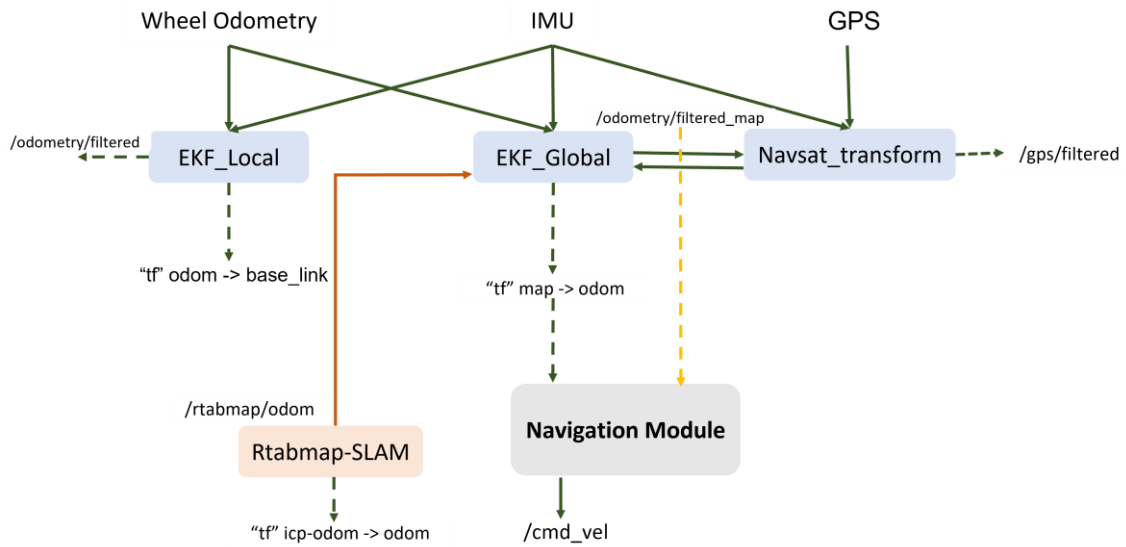


Figure 3-14 Merging SLAM odometry data with the other sensor data in robot-localization

In this scenario, if GPS information is available, navigation will be based on the fusion of GPS and RTAB-Map odometry information. In this case, the RTK-GPS could act like an odometry input to *robot_localization* along *icp_odometry* (output of RTAB-Map). If GPS data has an outage in some parts of the environment, the pose will be computed from visual odometry velocities which provide information about the motion of a camera or sensor, and by extension, the robot to which it

is attached. Once the camera motion between frames is estimated, the velocities can be computed based on the estimated motion. In other words, by integrating the velocities over time, the robot's pose can be estimated.

In short, navigation will be based on RTAB-Map odometry information and the other fused data. EKF will still continue to estimate the pose of the robot with odometry information generated by LiDAR or a camera.

3.3.3 On-track Localization and Navigation

Rail vehicle localization is a task of fundamental importance to provide reliable and accurate vehicle positioning information for railway maintenance, safety, and control. The purpose of movement of the RIRS on the railway track is for detecting the defect and performing some designed repair tasks. For achieving to this aim, an accurate on-track localization is needed, that this would be possible when there are sufficient reference inputs for position optimisation and error reduction. Rail vehicle positioning, as it is cited in section 2.4, generally classified into two main categories including track-side infrastructure and onboard sensors. It is also explained that rail environment highly impacts the GPS data, as well as localisation of the system moving on the rail. Considering these points, a test designed for the RIRS to show how it can localise itself and find the defect location on the railway track. First an autonomous navigation based on GPS odometry is designed and then finding the defect location based on track-side objects detection and distance estimation. In the following section these methods are explained in more detail.

3.3.3.1 On-track Localization Procedure

In this section, the procedure which is designed for on-track navigation and localization of the RIRS is provided. Figure 3-15 illustrates the on-track phase which is composed of two main sections. First, implementation of autonomous navigation of the RIRS to the GPS waypoint which was specified as a goal, and also simultaneous localization and mapping of the environment by RTAB-Map. Second, execution of autonomous navigation of the RIRS on the rail to find the defect location using the track-side objects as references and finally stop at the

defect location [144]. The term GPS waypoint is about providing a set of GPS points (i.e., a set of latitude/ longitude pairs) for the robot. Then the robot is commanded to autonomously navigate from its current location to the defined waypoint. Waypoint is a point of reference that can be the specific latitude and longitude of a location, a well-known building, or natural features. Here a location on the railway track with a specific latitude and longitude is defined as waypoint.

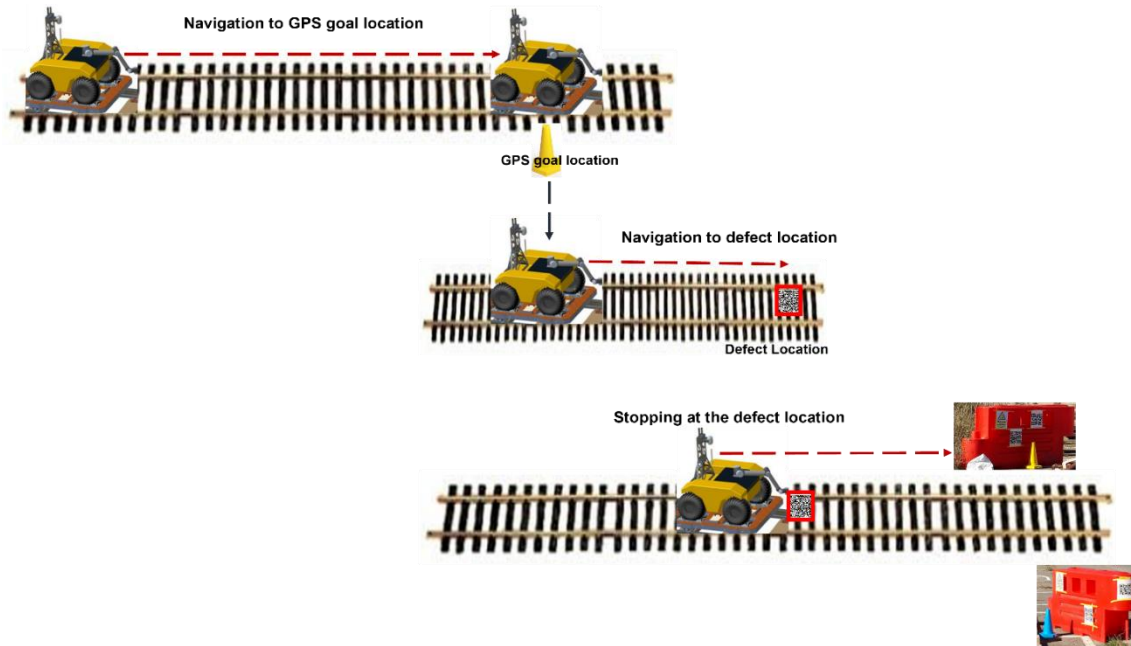


Figure 3-15 The illustration of RIRS job phases

For reaching the defect location, several key information from the environment and sensors especially GPS, RGB-D camera, and LiDAR, were saved as references and threshold points in the memory of the system. This information would amend vehicle localization and result in finding the defect location with higher accuracy. Figure 3-16 demonstrates the on-track framework for localization and navigation of the RIRS. In the inspection phase, the latitude and longitude of the goal location are saved for navigation purposes and the latitude and longitude of the defect location are saved for later analysis and further processing. The images of the track-side objects captured through the RGB-D camera are also saved in the memory of the system to be used for the feature-matching process. LiDAR provides us with the distance to the detected object. This information will be applied during the localization phase.

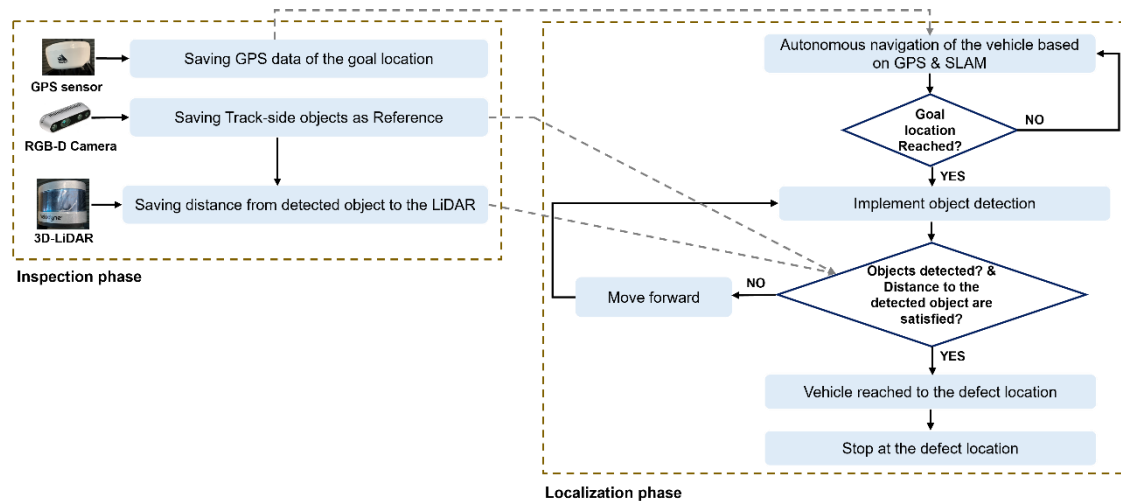


Figure 3-16 On-track localization and navigation framework

3.3.3.2 Localization and Navigation using GPS odometry and RTAB-Map

As it is stated, the first phase of on-track operation represents autonomous localization and navigation of the RIRS on the railway track using GPS waypoint and RTAB-Map for creating local odometry and a detailed map of the environment. In this step, the gps waypoint (latitude and longitude) of the specified location were saved as a goal location. In this regard a ROS node was defined which converts decimal GPS data, which is published on the topic "*waypoint*", into a 2D Navigation Goal for the navigation module. It would convert decimal GPS coordinates of waypoint to ROS position vector relative to the current gps position of the robot. This would be updated by the movement of the vehicle on the railway track.

To return the distance from the current vehicle's pose to GPS waypoint in metres, the haversine formula using the curvature of the earth, is considered in the calculation.

- **Haversine Formula:**

The haversine formula is a very accurate way of computing distances between two points on the surface of a sphere using the latitude and longitude of the two points. The haversine formula determines the great-circle distance between two points on a sphere given their longitudes and latitudes. One of the primary applications of trigonometry is in navigation.

It is a special case of a more general formula in spherical trigonometry, the law of haversines, that relates the sides and angles of spherical triangles. The word "Haversine" comes from the function:

$$\text{haversine}(\theta) = \sin^2\left(\frac{\theta}{2}\right) \quad (3-1)$$

The haversine of the central angle (which is $\frac{d}{r}$) is calculated by the following formula:

$$\begin{aligned} \left(\frac{d}{r}\right) = & \text{haversine}(\phi_2 - \phi_1) \\ & + \cos(\phi_1) \cos(\phi_2) \text{haversine}(\lambda_2 - \lambda_1) \end{aligned} \quad (3-2)$$

Where ϕ_2 and ϕ_1 is the latitude of two points, λ_2 and λ_1 is the longitude of two points, r is earth's radius (mean radius = 6,371 km) and d is the distance between two points.

The algorithm for navigation based on GPS on the railway track is provided in Figure 3-17. In this procedure, as it is noted first the validity of the GNSS signal is checked to see if there has been a GPS fix within the last GPS-validity-timeout seconds. If it is approved the distance between the GPS goal location and the vehicle is calculated. With the movement of the vehicle on the railway track, this distance starts decreasing. By reaching the vehicle to the goal location, the RIRS would stop.

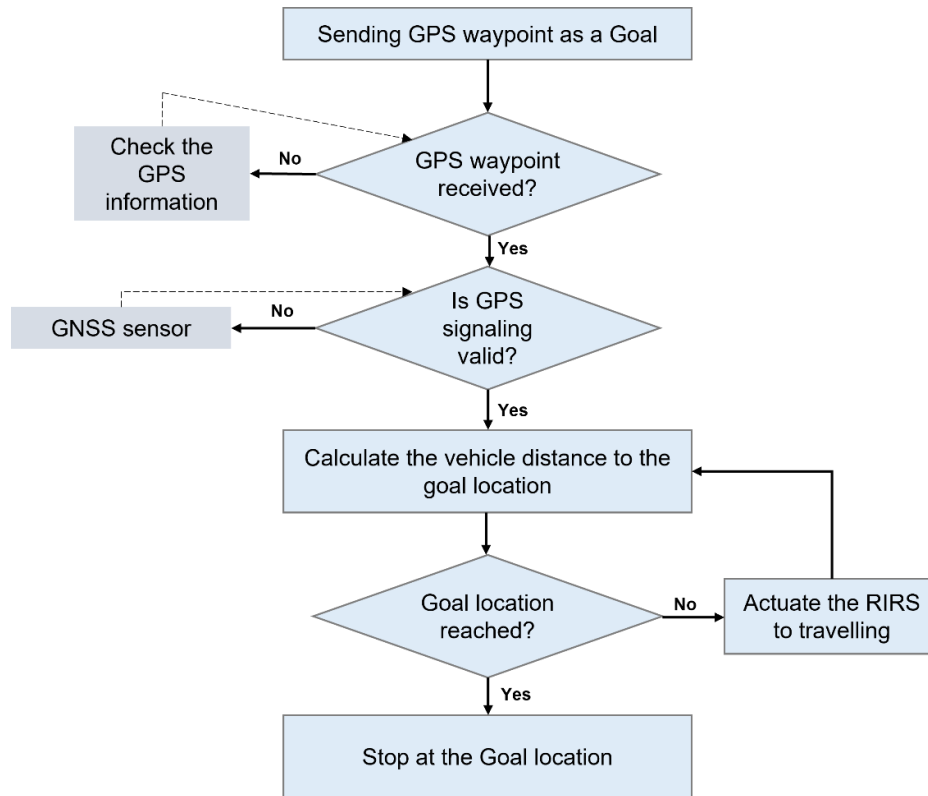


Figure 3-17 Architecture for on-track navigation to goal location

3.3.3.3 Object detection

Object detection in autonomous vehicles applied for various reasons such as detection of vehicles, objects, barriers, etc, to ensure a safe and robust movement of the robot. For the on-track phase, the algorithm of object detection based on feature feature-matching was utilized to provide a promising reference for localizing the vehicle and finding the defect location at the absolute level. Considering the railway environment, the maintenance vehicle might not accurately find the defect position due to the real-time GNSS error. In this case track-side objects would operate as datum points for finding the defect location. For this reason, object detection process should be fast and accurate enough to enable the control and braking system to react on time and stop the vehicle at the defect location. In the following, more information about object detection method is provided.

Object detection is a computer vision technique that works to determine if a predetermined object is contained within a given image and its location within the

overall image (2D Space). Often, but not always, object detection can also determine the position, orientation, and scale of the object within the 3D space that is represented by the image. This is possible if the camera provides both RGB and the depth image at the same time, such as RGB-D camera, or Zed camera. In this research, RGB-D camera, is utilized. But in the following it will be examined that the depth image provided from RGB-D camera is not reliable enough for 3D-pose estimation.

The technique employed in object detection is feature matching, which involves detecting and matching features across images to identify regions of interest. Features refer to distinctive structures in the image, such as points, lines, corners, edges, or objects. These features can be divided into two primary categories:

- Key-point features that are present in specific image locations, such as peaks or corners, and are typically characterized by the appearance of surrounding pixel patches at the point location.
- Features, which can be matched based on their orientation and local appearance (edge profiles) and can serve as strong indicators of object boundaries and occlusion events in the image sequence.

Generally, feature-based matching approaches typically involve the following steps:

- **Feature Detection**

In feature detection, the procedure is to identify the feature points or interest points, in which the direction of the boundary of the segment/object changes suddenly.

- **Feature Description**

A feature descriptor is an algorithm which takes an image and outputs feature descriptors/feature vectors. Feature descriptors encode interesting information into a series of numbers and act as a sort of numerical “fingerprint” that can be used to differentiate one feature from another. ORB, FREAK, BRISK, and BRIEF are some of the examples of feature descriptors. Ideally, this information would be invariant under image

transformation, so the features can be found again even if the image is transformed in some way.

- **Feature Matching**

Feature matching or generally image matching, which is an integral part of various computer vision applications such as camera calibration, image registration, and object recognition, involves establishing correspondences between two images of the same scene or object, with one acting as a template or sample. The process typically involves detecting a set of interest points in each image and associating them with image descriptors. Once features and descriptors have been extracted from two or more images, the next step is to establish initial feature matches between them. The performance of matching technique based on interest points depends on the properties of the underlying interest points and the choice of associated image descriptors.

3.3.3.3.1 Object detection in ROS

For solving the real-time computer vision problems, OpenCV (Open-Source Computer Vision library), was applied, and for importing the functionality of the OpenCV into ROS, an interface called “*Vision-opencv*” was utilized. This interface consists of two sub packages named: *cv_bridge* and *image_geometry*. In “*Vision-opencv*”, feature descriptors and feature matching are being used.

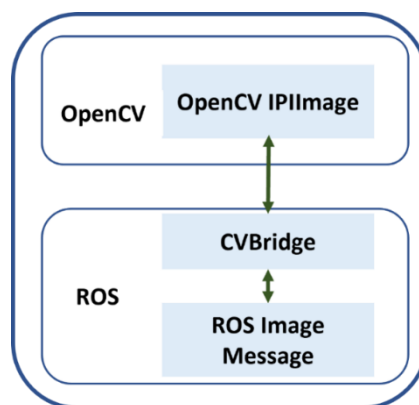


Figure 3-18 Interface between OpenCV and ROS

The following figure illustrates the procedure of feature matching in ROS. In this procedure the first captured images will be saved in ROS through *cv-bridge*. Then

after reading the reference and the template images, common features in both images will be extracted. Finally, a matcher function will find the highest match between them.

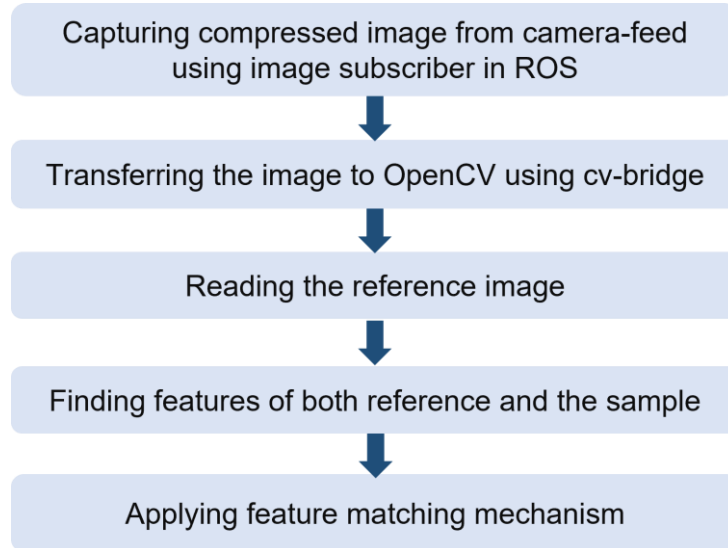


Figure 3-19 Procedure of feature matching in ROS

In this research, track-side infrastructures such as signal-post, light-post, and mast considered as references to test the object detection method. The point about the considered objects is that they should have a size/dimension which is not too small or too big. These objects should be in the point of view of the camera and LiDAR sensors. Colour, shape, and discriminative objects in the environment help the object detection process to be done with more robustness and accuracy. A number of these objects are shown in Figure 3-20.



Figure 3-20 Track-side objects considered as references.

In this research, the process of object recognition was done by applying “*find_object_2d*” node from “*find_object_2d*” package from ROS. This package implements this process in two main steps including teaching and recognition parts. The teaching process was executed before the main robot operation.

During this step, objects were presented to the vision system, and then the image and extracted set of features were saved as a pattern. Then the actual recognition was executed during the movement of the robot on the railway track. Every frame of the camera was processed, and image features were extracted and compared to the data set in the memory. If enough features match the pattern, then the object is recognized. The detected objects were specified with colourful bounding boxes. For finding the position of any pixel in the image target and the corresponding pixel in the search image, a Homography matrix was applied. This matrix is a transformation matrix that relates the transformation between two different perspectives of the same scene, even when there are changes in perspective, rotation, and scale. With this information, the 2D or 3D pose of the planar objects within the search image was determined. In the following figure, the procedure of feature matching is represented.

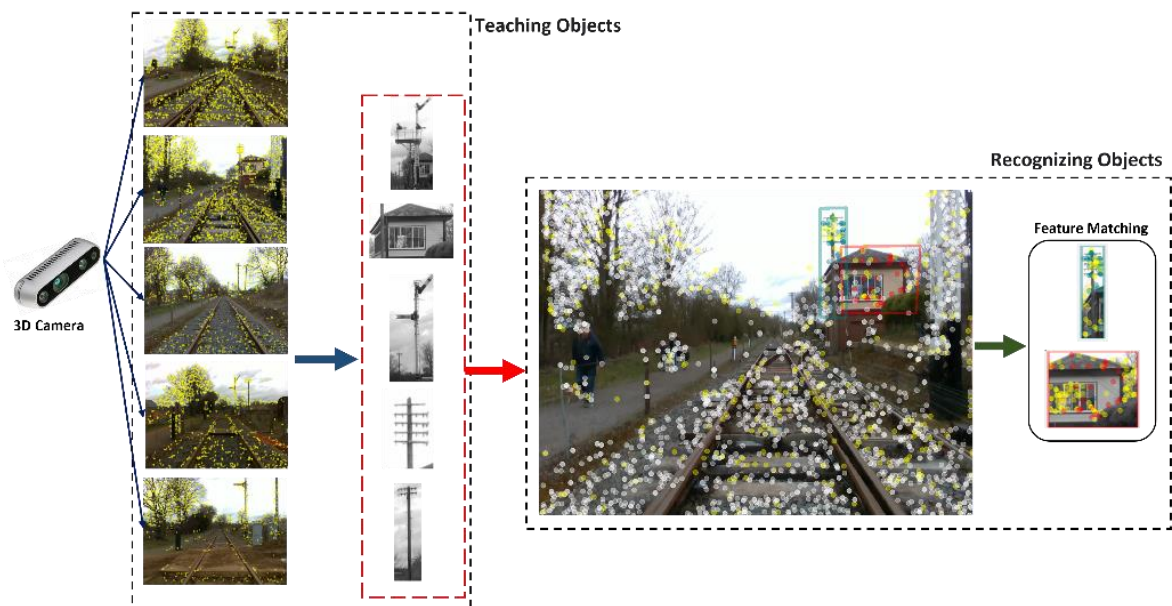


Figure 3-21 Object detection procedure

An RGB-D camera, besides the RGB images, provides in-depth images too. As the depth image contains the point cloud, it can provide the 3D pose of the detected object as well. But in the next section, it would be shown that the acquired pose from this type of camera is not reliable enough. The following figure shows how the process of pose estimation is performed using an RGB-D camera.

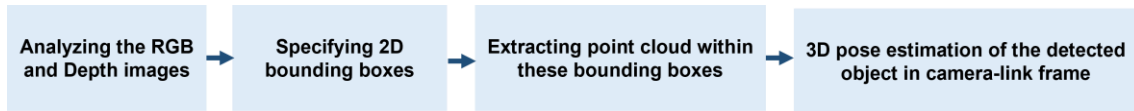


Figure 3-22 Strategy of the 3D pose estimation based on camera.

In Figure 3-23 multi-object detection is represented. In Figure 3-23 (a) one object, and in (b) two objects are identified with the colorful bounded boxes. In the corresponding terminal, the 3D pose of these objects to the *camera-link* is published. The recognized objects are identified with a unique ID, which can help with later processing. In section 4, more results related to track-side object detection are illustrated.

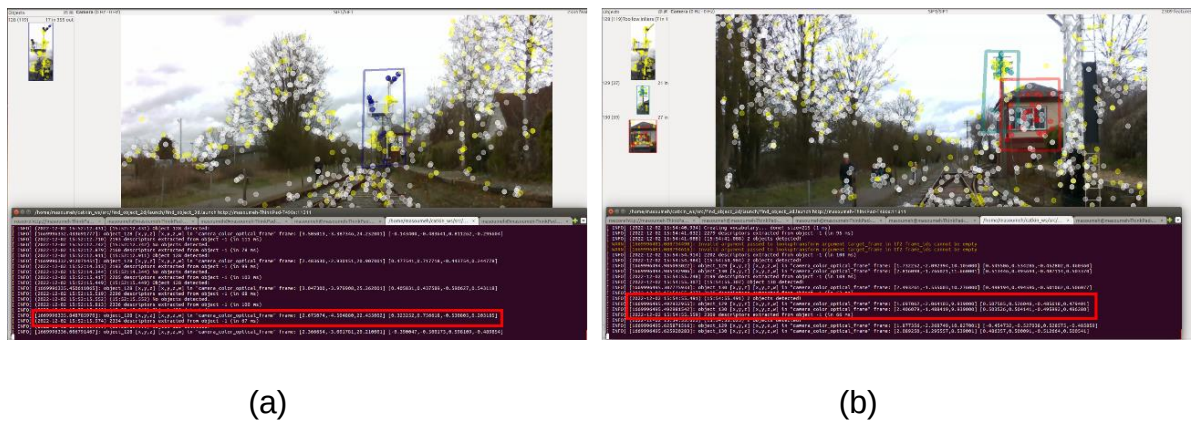


Figure 3-23 (a) Single object detection and (b) Multi object detection with their relative positions to the camera

3.3.3.4 3D-pose Estimation of the Detected Object

By combining data from camera and LiDAR sensors, railway object detection can be improved since each sensor has different strengths and weaknesses. Cameras can provide high-resolution images with context information that is useful for robust and accurate object detection, but they are not as effective in estimating the distance to the object. On the other hand, LiDAR sensors can provide precise depth information, making them very accurate in measuring the distance to the object, but less robust in object detection. With a 3D LiDAR sensor, it is possible to obtain 3D position and depth information of objects in the environment. By considering the strengths and limitations of both sensors,

combining their data can enhance object detection and distance estimation on the railway track.

In this research, it was decided to apply LiDAR for the aim of distance estimation. For this aim point clouds were downsampled in X, Y, and Z directions, and just kept the points related to the object. Figure 3-24 represents, object detection and 3D pose estimation of the track-side object. The results based on this configuration are analysed in section 4.2.4.

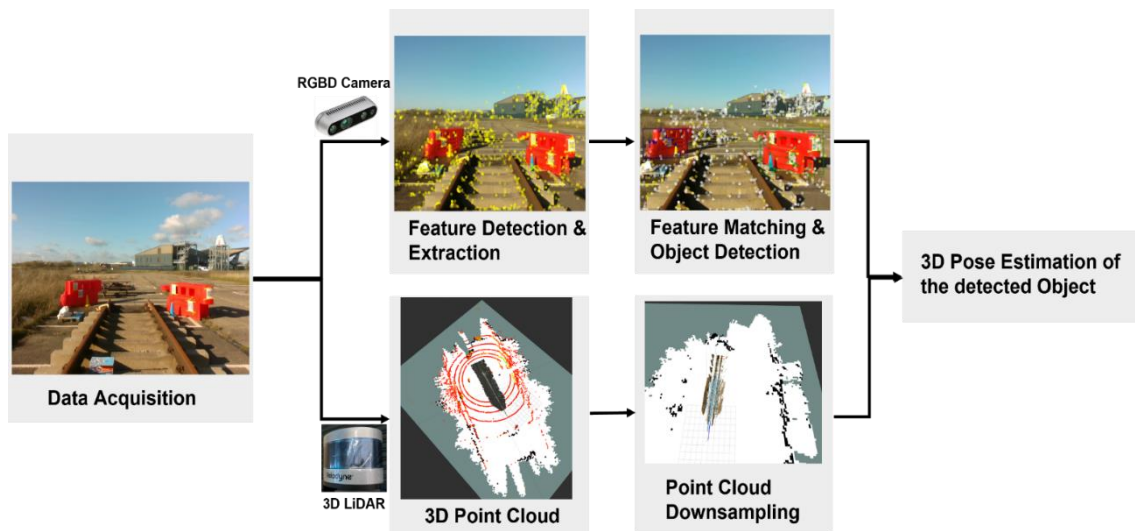


Figure 3-24 Object detection and 3D Pose Estimation

3.4 Summary

This chapter provides a full overview of the applied methods and techniques for both off-track and on-track navigation phases. First, an introduction to RIRS was given, with its on-board sensors, instrumentation, and their specifications. Then the accuracy and precision terminologies related to the GPS data were examined. In terms of off-track navigation procedure, path planning, static object detection, RTAB-Map, and EKF were reviewed, and explained. Regarding the on-track navigation phase, navigation based on GPS waypoint, track-side object detection, and distance estimation was presented. The results related to all these methods are provided in the next chapter.

4 RESULTS

This chapter presents the results obtained from the different experiments conducted to reach the Ph.D. objectives. Initially, the off-track localization and navigation including the results related to the RTAB-Map algorithm, path planning, and implementation of the EKF method were evaluated. Then, an assessment of the gained results was performed by applying two evaluation metrics, named Absolute Pose Error (APE) and Relative Pose Error (RPE).

These were then followed by the results related to the on-track phase including the experimental setup, autonomous navigation based on GPS and RTAB-Map, and then localization of the vehicle in GPS-denied environments. Finally, an evaluation of the results was conducted.

4.1 Evaluation of the Off-track Localization and Navigation

In this section, results related to the off-track localization and navigation are presented. These results include autonomous localization and navigation, path planning, static object detection, RTAB-Map approach, and the fusion of GPS with other odometry information through EKF estimator. An evaluation of the pose estimation of the vehicle based on RTAB-Map, and global pose estimation are also provided. A discussion of the provided results has been provided as well.

4.1.1 RTAB-Map

As was reviewed in section 3, the simultaneous localization and mapping approach applied in this research was RTAB-Map, an RGB-D SLAM module. This method was distributed as an open-source library since 2013. In ROS, the related package is named *RTABMap_ros*. Figure 4-1 shows the robot configuration including sensors and the related messages which were fed into the RTAB-Map approach. These sensors include 3D-LiDAR, RGB-D camera, RTK-GPS, IMU, and a wheel-odometer. The related message of each sensor are as follows:

- 3D LiDAR: *sensor_msgs/Pointcloud2*
- Odometry: *nav_msgs/Odometry*
- IMU: *sensor_msgs/Imu*

- GPS sensor: *sensor_msgs/NavSatFix*
- A vision sensor (Kinect-like sensor): *sensor_msgs/Image (rgb)*, *sensor_msgs/CameraInfo*, *sensor_msgs/Image (depth_registered)*

In this structure, *rtabmap* node reads the data from the sensors and then turns it into a map data and occupancy grid map. These outputs are visible from the client/PC side through Rtabmapviz and Rviz, visualization tools. Map data, containing the latest added node with compressed sensor data and the graph, can be seen through both tools.

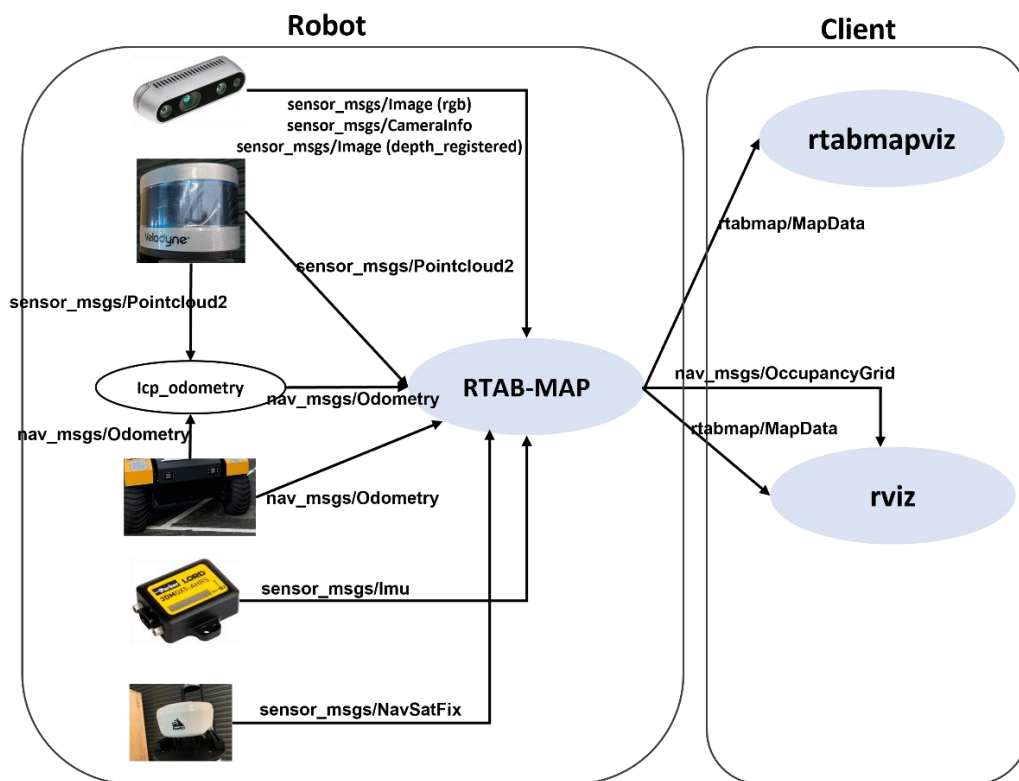


Figure 4-1 Input sensors into RTAB-Map module

The hardware setup of the applied sensors is shown in Figure 4-2. 3D-LiDAR and RTK-GPS are situated in the top tower of the Warthog, RGB-D camera is attached to the body of the vehicle. IMU and wheel odometer were embedded in the body of the system. In addition, a trolley was also designed for the navigation of the vehicle on the railway track. Warthog UGV comes with an onboard computer with Ubuntu 16.04 and Robot Operating System (ROS) as middleware.



Figure 4-2 Hardware setup

After defining the sensors with their specific topics and messages in the RTAB-Map configuration file, and then launching the *rtabmap* node, the connection between frames is generated. As LiDAR was considered for estimating the movement of the robot over time, the relative odometry was computed by Iterative Closest Point (ICP) registration, and the related topic was */rtabmap/odom*, with the frame called *icp-odom*. The tf system in ROS keeps track of multiple coordinate frames and maintains the relationship between them in a tree structure. Tf is distributed, so that all coordinate frame information is available to every node in the ROS network. Part of the tf-tree related to using RTAB-Map as a map provider is represented in Figure 4-3.

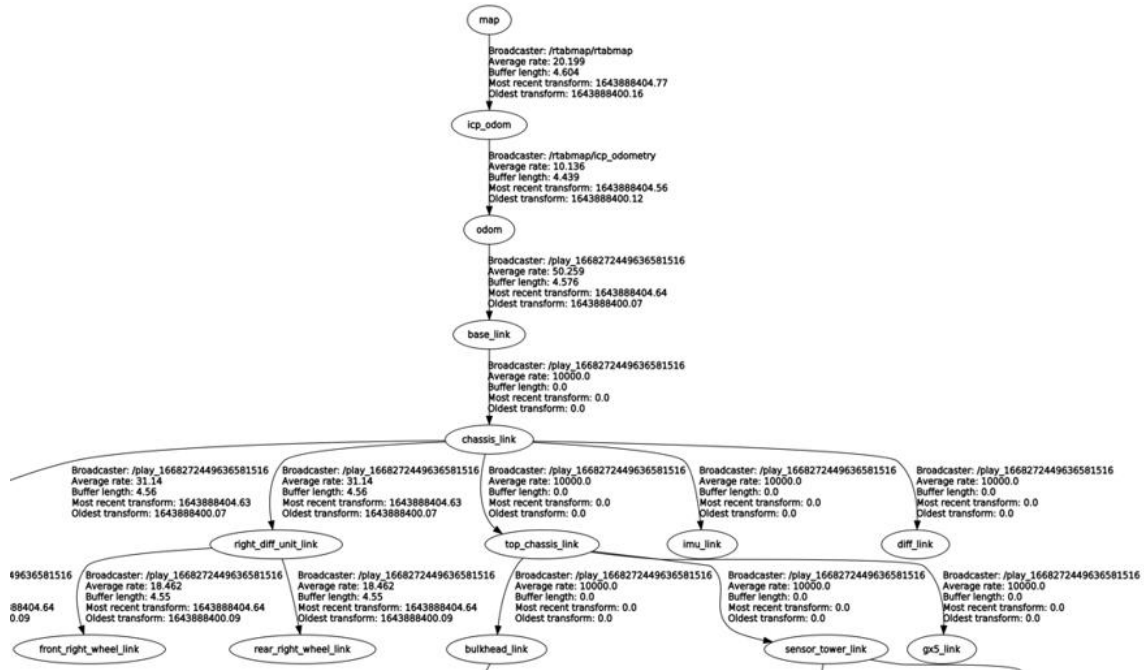


Figure 4-3 The transform tree of the Warthog in ROS after launching RTAB-Map representing the frames and the hardware position relationships.

For achieving a proper result, first the algorithm was run in mapping mode, and during the mapping of the environment, the mapping data was saved in the *rtabmap.db* database. The mapping part was controlled with more than 100 parameters which were so complicated to tune it. Here the approach for tuning the parameters was in this way that started with loading the default config file and running the application, and then changing a parameter, and checking the application again. This procedure was repeated several times and tested the results each time until it yielded a good performance. Then RTAB-Map was run in localization mode to estimate the poses of the robot under icp odometry while employing the corresponding map generated.

The ROS graph of the input and output of the RTAB-Map is presented in Figure 4-4. In this graph, the rectangular shapes are the topics, and the ellipses shapes are the nodes. It also shows sensors including IMU, RTK-GPS, RGB-D camera, and LiDAR as the input of RTAB-Map and the created *map* as the output. Several key topics and nodes are marked in the following figure.

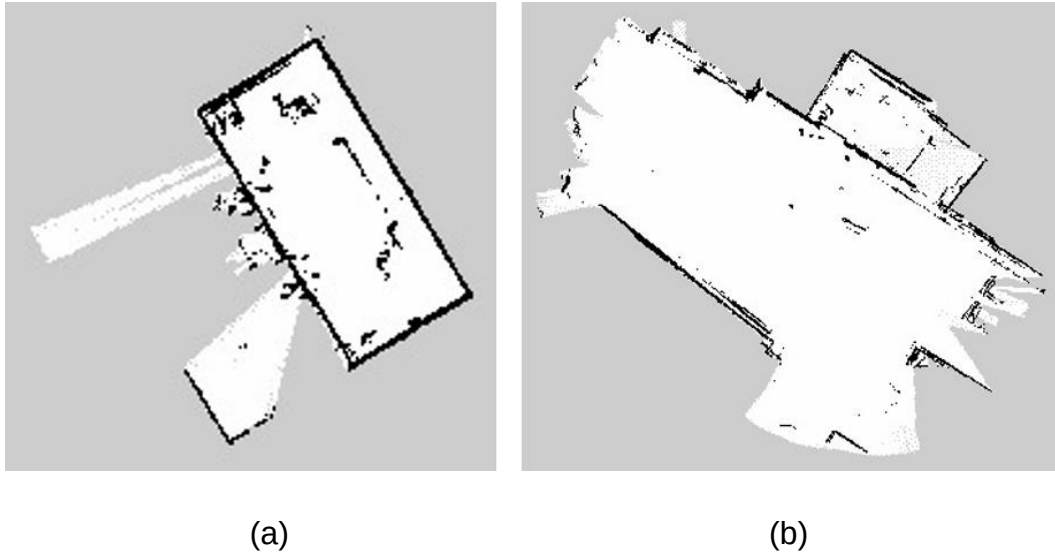


Figure 4-5 Mapping output of RTAB-Map (a) Indoor 2D map (b) Outdoor 2D map.

Steps 1 to 6 of Figure 4-6, show the generated map from RTAB-Map over time. The colored points represent the point cloud data, and the gray cloud shows the 3D map cloud. The cloud is created incrementally, by the movement of the robot, in Rviz. When the graph is changed, all point clouds added in Rviz will be transformed into new poses. It has the same properties as a point cloud display but with a different configuration.

In graph-based SLAM approach, as the map expands, more time is necessary for loop closure detection, graph optimization, and map assembly. Moreover, the integration of other processing modules like control, navigation, obstacle avoidance, and object recognition could also curtail the CPU time allocated for SLAM. Here, RTAB-Map has the ability to restrict the computational load to prevent latency issues with other modules [145].

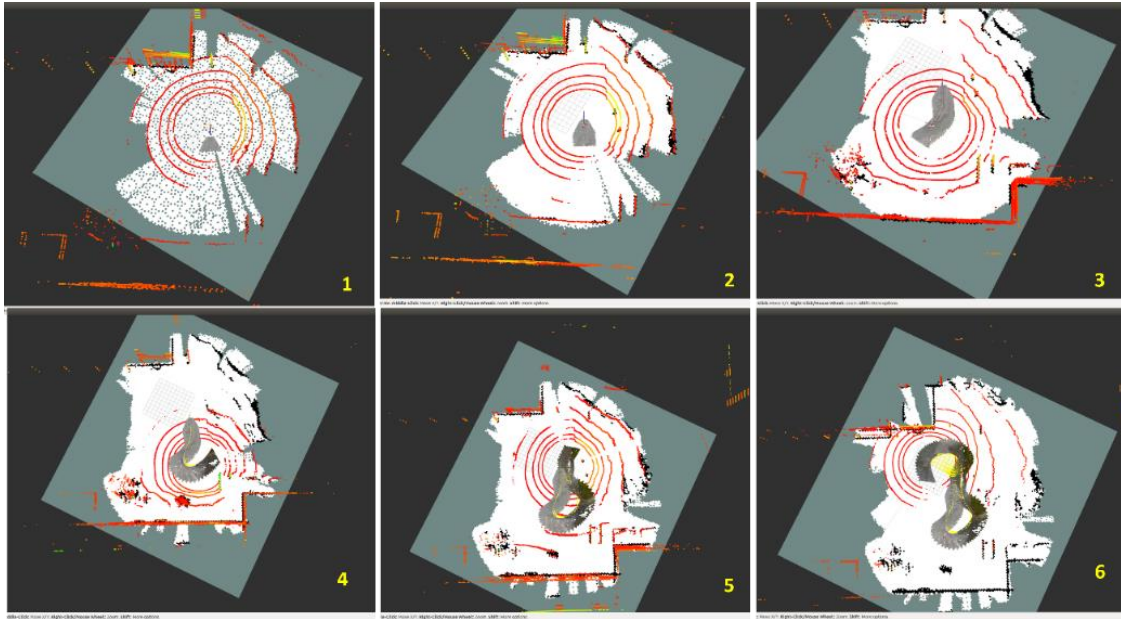


Figure 4-6 Visualization of RTAB-Map output in Rviz

Other outputs of RTAB-Map include `octomap_grid` (`/rtabmap/octomap_grid`). Octomap is the representation of 3D environment models from sensor data. In another words, it is the point cloud of the obstacles, occupied space, ground, and empty spaces. Now RTAB-Map provides the projection of the OctoMap into a 2D occupancy grid map. The occupancy grid map is the famous and widely used representation of a location-based map. It has been frequently utilized in mobile robotics for environment modelling, due to its ease of use and implementation. Figure 4-7 shows the `octomap_grid` in Rviz.

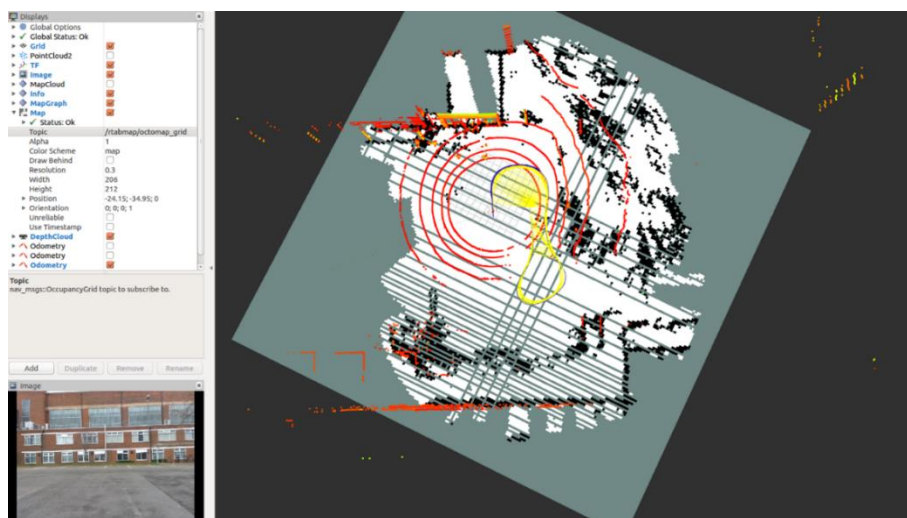


Figure 4-7 representation of octomap_grid in Rviz

As mentioned before, two main features of RTAB-Map are loop closure and proximity detection which are specified in Figure 4-8. Proximity detections are visible as yellow links, and loop closures are specified as red links. In Rviz from the topic display section, under `/info` topic, the numbers of detected loop closure and proximity detection are counted during the movement of the robot in the environment. There is a localization correction after each loop closure that corrects most of the odometry drift that happens during the run. If the environment lacks visual features, a camera may just not be effective for loop closure detection. Without a camera, the loop closure detection doesn't work. However, proximity detection can still work with scans, so some loop closures can be detected if odometry doesn't drift too much. The obtained localization would also be a combination of odometry pose (`/odom -> /base_link`) and odometry correction (`/map -> /odom`). The map would be updated, and drift would be diminished when a loop closure happens.

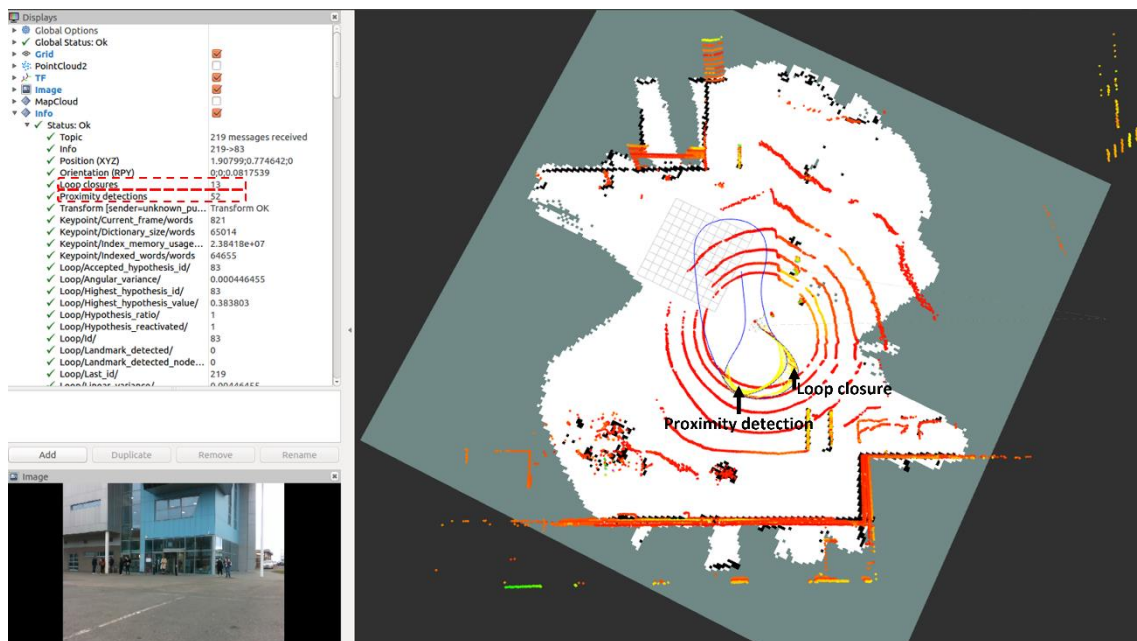


Figure 4-8 Loop closure and Proximity detection

3D map of the environment and the loop closure detection can also be visualized through the visualization interface of RTAB-Map named Rtabviz. It is a wrapper of the RTAB-Map GUI library. It has the same purpose as Rviz but with specific options for RTAB-Map. The following figure shows the pose graph

composed of nodes and the links between them, created during the movement of the robot. Every node in the graph corresponds to spatial information, like the poses of the robot in different time instants, and every edge between two nodes represents spatial constraints between two nodes or between a node and a landmark.

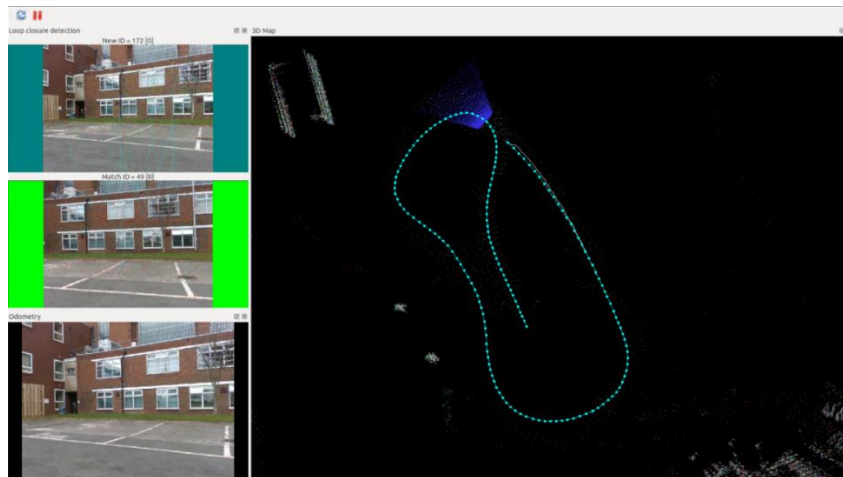
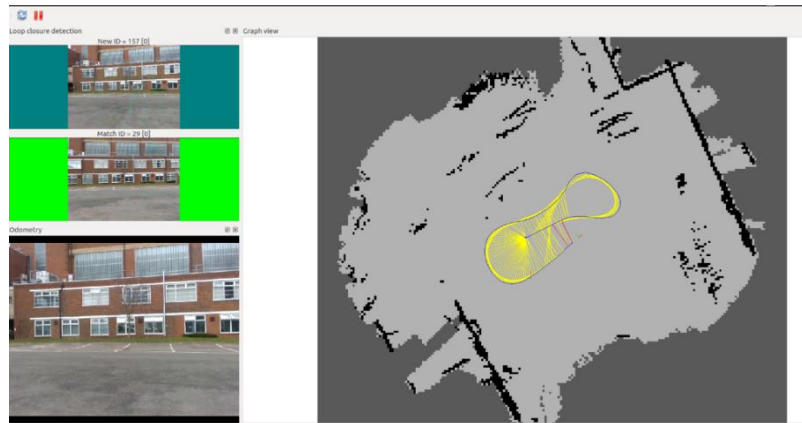


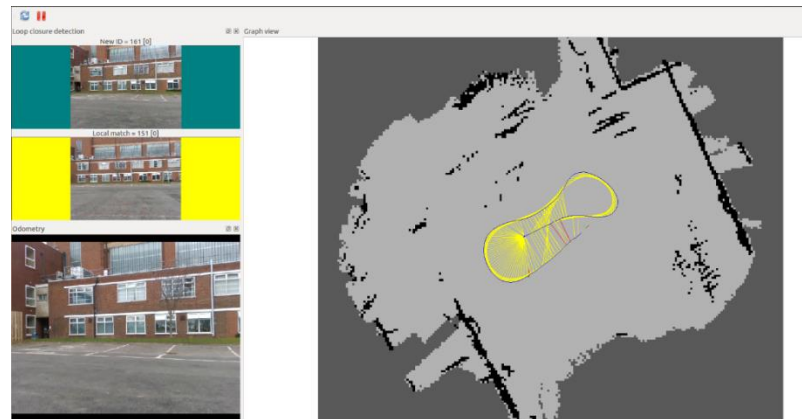
Figure 4-9 Pose-Graph output of RTAB-Map

Graph optimization spreads the calculated error throughout the entire graph to reduce odometry drift whenever a new loop closure or proximity link is introduced. Odometry correction to derive the robot localization in the map frame is also available through *tf* ("*/map -> /odom*").

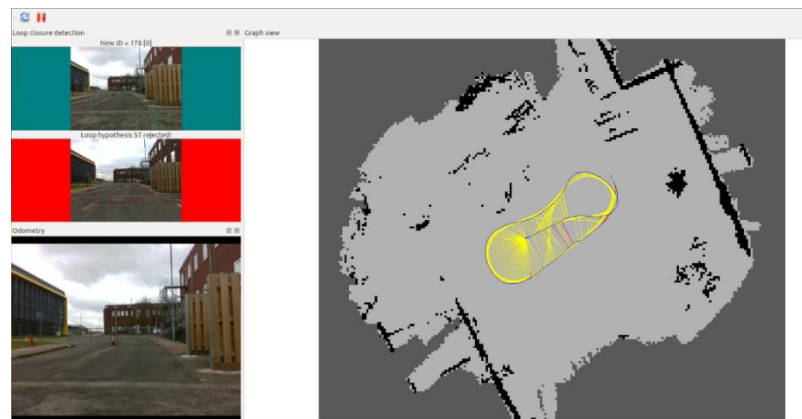
In Rtabmapviz, it is possible to visually check where and when loop closure has happened. For instance, Figure 4-10 shows that, if the background of middle window turns to green, it means loop closure is detected, if it is yellow, it means proximity link is detected in space, and when it gets red it means loop closure is rejected. The following figure shows a summary of loop closure in 3 different states.



(a)



(b)



(c)

Figure 4-10 State of loop closure during the movement of the vehicle; (a) Green: Loop closures detection accepted; (b) Yellow: Proximity detection in space; (c) Red: Loop closure detection rejected.

4.1.2 Off-track Path Planning and Collision Avoidance

Path planning is composed of global planner and the local planner. A global path planner is paired with a local path planner, and both work in a complementary manner. As it is reviewed, the planners are the one that compute the trajectory from the current location of the robot to the goal destination. Global planner uses the global cost-map to compute the long-term path to follow to the final destination. In this research *navfn/NavfnROS* is considered as base global planner. On the other side, local planner uses the local cost-map to generate short-term paths in the vicinity of the robot, to avoid unexpected new obstacles that appear in the robot space. *Dwa_local_planner/DWAPlannerROS* is considered as base local planner. In this procedure, cost-maps represent places that are safe for the robot to be in a grid of cells. In the following figure local and global cost-maps as well as global and local path planners are shown, using Rviz for visualisation. This test aims to integrate the obstacle-avoiding algorithm with path-planning techniques to create a system that can navigate in an environment avoiding static objects.

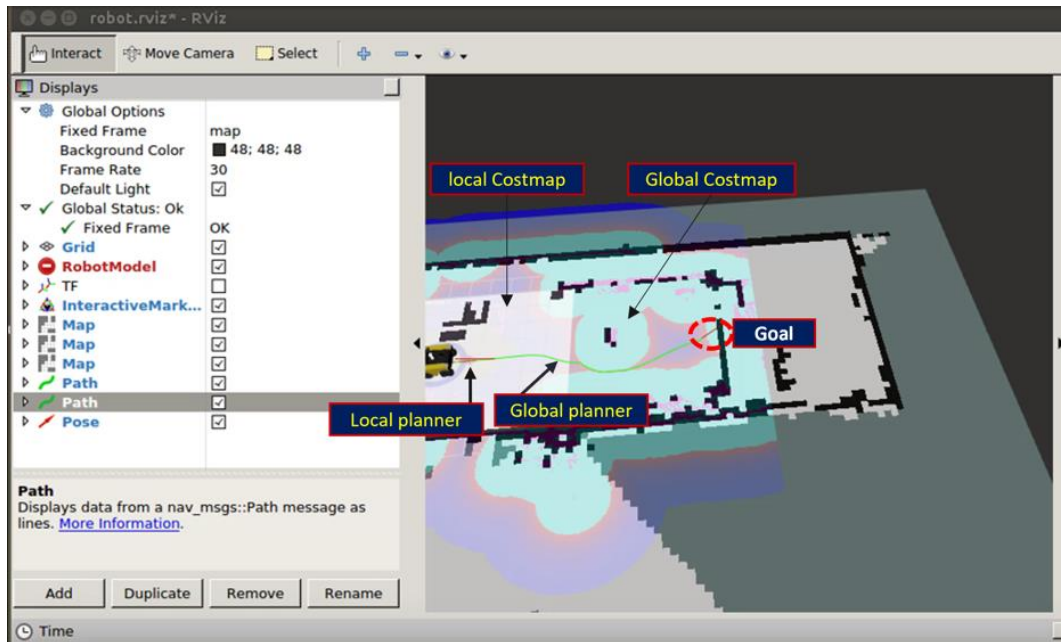


Figure 4-11 Global and Local cost-maps; Global and Local path planners

The autonomous movement of the vehicle from its initial location after receiving the odometry information, up to the time that reach the specified goal are shown

in Figure 4-12. For this test, first RTAB-Map was in navigation mode to save the map of the environment in the memory of the system, and then change the mode of RTAB-Map into localization to start localizing itself relative to the map while navigating. In the corresponding terminal the status of the movement of the vehicle including receiving the odom information, and then getting a new plan, and finally the time that goal is reached, were published. As can be seen, local and global cost-maps are also created around the vehicle and the static objects to provide a safe zone for the movement of the vehicle. This test was done inside the lab to check the functionality of RTAB-Map in indoor environments.

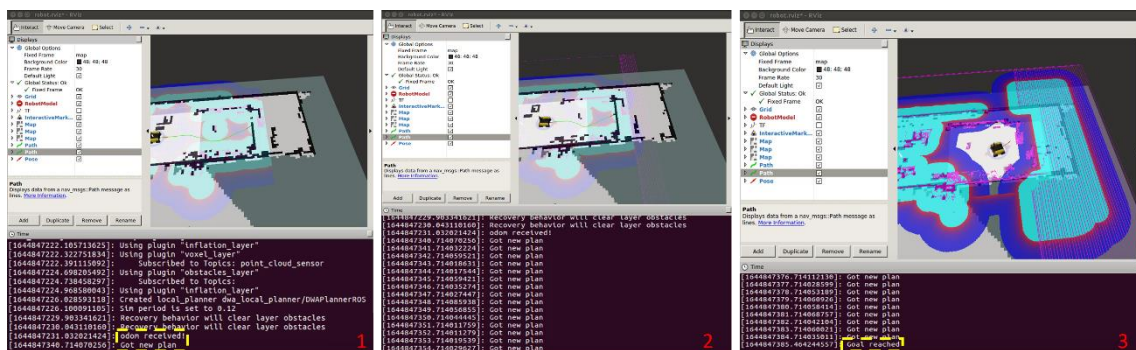


Figure 4-12 Indoor autonomous navigation of the vehicle from its initial location to the specified goal

The autonomous navigation of the vehicle was also tested in an outdoor environment. Again, a location was specified as a goal for the vehicle. Then the navigation module, by receiving the coordinates of this point relative to the map frame, started publishing the global and local planner. By receiving the information from camera and LiDAR, the obstacles around the robot were identified and local planner found a safe path for the vehicle to detour the near objects. The global planner also generated a long safe path to the goal location for the vehicle. Here by reading the coming information from sensors, the objects in the environment and the path of the vehicle were identified. As it can be seen in Figure 4-13, in step 1 it received the odometry information and the pose of the goal, in step 2 the vehicle detoured the detected object, and finally in step 3, it reached the specified goal location.

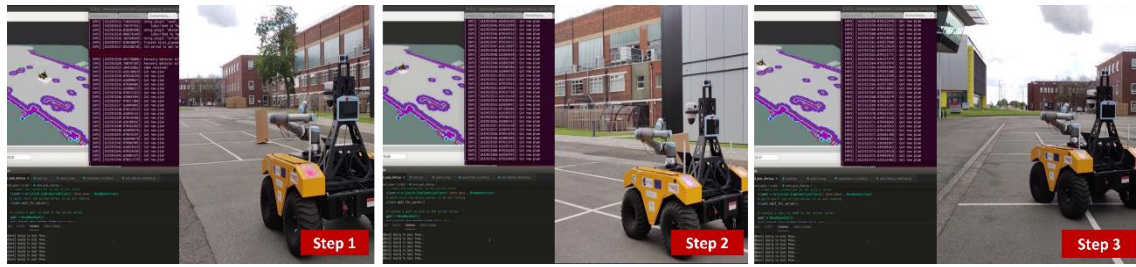


Figure 4-13 Outdoor autonomous navigation of the vehicle from its initial location to the specified goal

4.1.3 Merging GPS with RTAB-Map Result

In the methodology section, the fusion of GPS and SLAM and their applications are illustrated. The main aim of this fusion in this research is to enable navigation of the RIRS in both GPS and GPS-denied environment. This sensor fusion is implemented by fusing global pose estimates from RTAB-Map with onboard sensor data through EKF estimator. Fusion of sensors with different rates of frequency through EKF is done in ROS. In this way that ROS does a *spinOnce* which is responsible for handling communication events, e.g., arriving messages. Then all the sensor measurements are waiting in the ROS call-back queue. They have been put in a C++ *priority_queue*, which is ordered by their time stamps. So, after all the call-backs fire, there would be an ordered measurement queue that is waiting to be processed. Then considering the time stamp of the most recent state/EKF output (t_{state}), and the time stamp of the first sensor measurement in the queue (t_{m0}), the time delta would be calculated, and then an EKF prediction step would be carried out over that. Then by using the measurement obtained at t_{m0} , the prediction would be corrected, and the EKF state would be updated. Now it would be $t_{state} = t_{m0}$. Again, the next measurement from the priority queue would pop up, and this EKF cycle would repeat until the priority queue gets empty. Finally, the EKF state would be published in a ROS message and via *tf*.

Then for evaluating and comparing the trajectory output of RTAB-Map algorithm and global odometry, a Python package named EVO, which is developed by M. Grupp, is applied. This open-source ROS package provides executables and a small library for handling and evaluating the pose results of various SLAM

systems. It conducts a quantitative analysis of the localization accuracy. In this section, the estimated trajectory from RTAB-Map, and the global pose estimated through EKF, are compared relative to the ground truth. In this research ground truth is the GPS data received from the RTK-GPS sensor.

In this regard, two prominent error metrics are applied for evaluating the localization performance including Absolute Pose Error (APE) and the Relative Pose Error (RPE). The APE is well-suited for measuring the performance of visual SLAM systems such as RTAB-Map. In contrast, the RPE is well-suited for measuring the drift (the local accuracy) of a visual odometry system, for example the drift per second. The APE directly measures the difference between the estimated pose and the reference pose obtained from the ground truth. For the evaluation, it is assumed that a sequence of poses from the estimated trajectory are given:

$$P_1, \dots, P_i \in SE(3) \quad (4-1)$$

$$E_i = P_{est,i} \theta P_{ref,i} = P_{ref,i}^{-1} P_{est,i} \in SE(3) \quad (4-2)$$

θ is the inverse compositional operator, which calculates the relative pose. APE in EVO takes into account rotation error as well as translational error.

$$(\text{Translation error}) APE_i = ||trans(E_i)|| \quad (4-3)$$

$$(\text{Rotation error}) APE_i = | (angle(log_{SO(3)}(rot(E_i))))| \quad (4-4)$$

$log_{SO(3)}(.)$ is the inverse of $exp_{SO(3)}(.)$ (Rodrigues' formula)

Thereafter, median, mean, root mean square error, maximum and minimum error values are calculated. For translation, we use the well-known mean, standard deviation and root mean square error (RMSE). They are calculated with the following formula:

$$trans_{mean} = \frac{1}{N} \sum_{i=1}^N ||trans(E_i)|| \quad (4-5)$$

$$trans_{std} = \sqrt{\frac{1}{N} \sum_{i=1}^N (trans(E_i) - trans_{mean})^2} \quad (4-6)$$

$$trans_{rmse} = \sqrt{\frac{1}{N} \sum_{i=1}^N ||trans(E_i)||^2} \quad (4-7)$$

On the other side, the relative pose error measures the local accuracy of the trajectory over a fixed time interval. Therefore, the relative pose error corresponds to the drift of the trajectory which is in particular useful for the evaluation of visual odometry systems. In this regard, in order to evaluate the estimated pose of the vehicle in both GPS and GPS-denied environments, a test was run in an environment where the GPS was available in some parts, but near the buildings, GPS data fluctuated and there were outages at some points. The GPS map of the environment is shown in Figure 4-14.



Figure 4-14 GPS output in the test area

In the following results related to both APE and RPE are presented. The first result shows the trajectory output of EKF named *global_odometry* which is specified with the blue line. EKF fuses the continuous sensor data with the global pose estimates from RTAB-Map and GPS to provide an accurate and high-rate global pose data of the vehicle. Here the rate of global-odometry data is about 20Hz. The green line also shows the output of RTAB-Map named *rtabmap_odom* illustrates the changes of the vehicle position based on visual odometry

information. The dashed line is related to the GPS data which is considered as ground truth. As it can be seen, in some parts (near the buildings) GPS has outages and fluctuates a lot, but in the middle of the path, far from buildings and objects, GPS has better output.

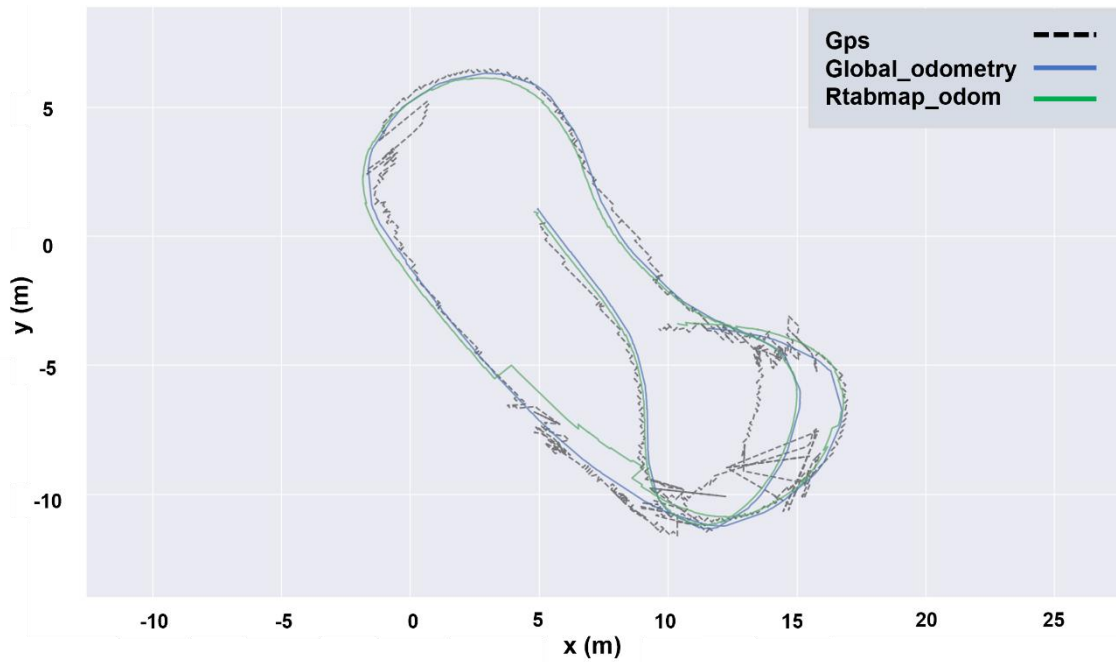


Figure 4-15 Evaluation the trajectory output of RTAB-Map and EKF result

In the following two plots, translation and rotational parts of the trajectories are presented. In the translational graph related to the x and y sections, it is visible that when the vehicle is near the buildings and wants to turn, GPS values have a jerky behavior.

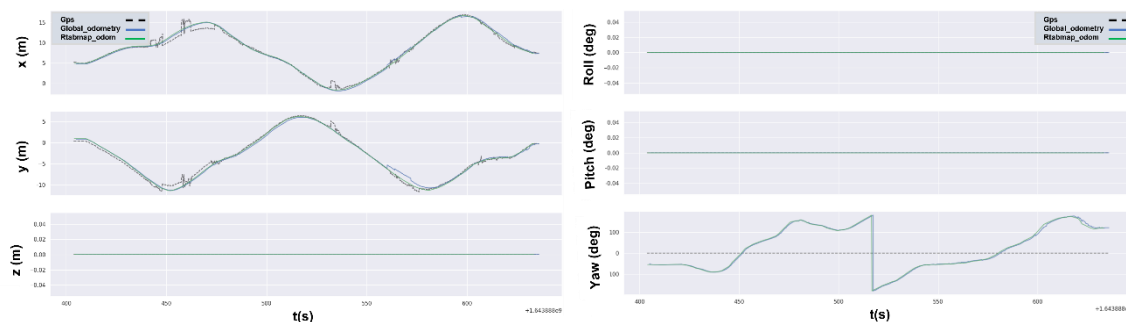


Figure 4-16 Translational and Rotational parts of pose of the vehicle over time

To further allow the evaluation of different localization configurations (RTAB-Map and EKF), the estimated poses are visualized as a trajectory separately for each configuration. In this regard to have a better understanding of the severity of the error related to each trajectory, the absolute pose error is mapped onto each predicted route. The reference line (dashed line) indicates the ground truth of the position, this is generated from the RTK-GPS sensor. The colored line in Figure 4-17 (a) shows the trajectory estimated by EKF and in Figure 4-17 (b) is the pose estimated by RTAB-Map, with the color describing the translation aspect of the APE relative to ground truth. The dark blue color and the dark red color are indicative of the least and the highest error respectively. Based on this plot the error mostly happens near the U-turns of the vehicle, while in the middle of the path which is far from any GPS disturbances, has the least error.

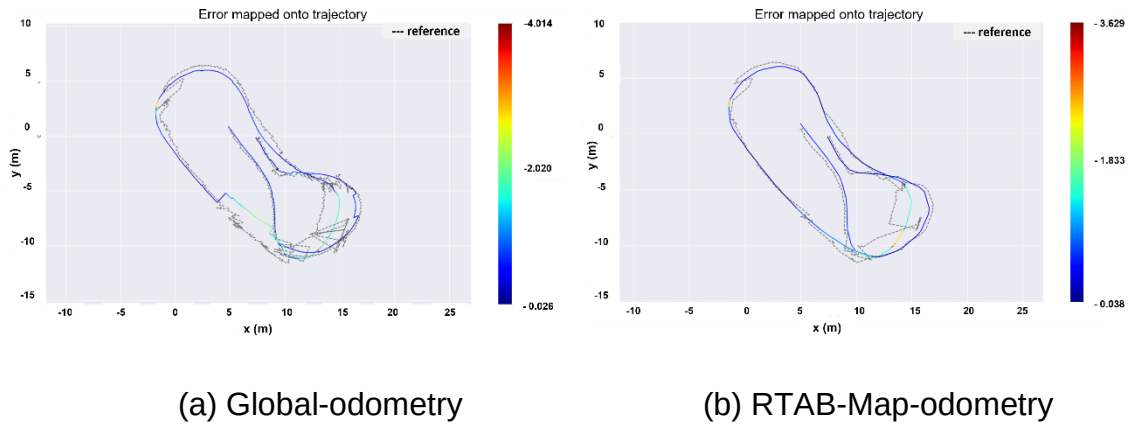
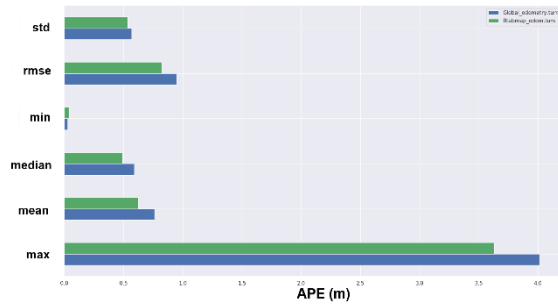


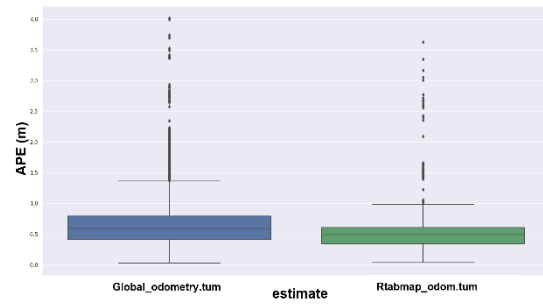
Figure 4-17 The Absolute Pose Error related to translation, mapped onto (a) global-odometry and (b) RTAB-Map-odometry trajectories.

Figure 4-18 depicts the translational part of absolute pose error related to global-odometry and RTAB-Map-odometry. Figure 4-18 (a) shows the statistics for the absolute error of the trajectory in a bar plot. The statistics are, from top to bottom, standard deviation (STD), RMSE, minimum error value, median, average error, and the maximum error value. The horizontal axis is the magnitude of the errors in meters. In this plot the minimum APE related to global-odometry is around 0.026 m and for the RTAB-Map-odometry is around 0.038 m. Where GPS is available, fusion of GPS and SLAM performs better than only applying SLAM approach, and the minimum error happens in those parts. Another measurement

of variability in the dataset is STD which controls the spread of the distribution from the mean value in a set of values. A smaller value of STD indicates that the data points are tightly clustered around the mean, whereas a high STD indicates that the data are spread out over a large range of values. Based on the acquired results, the STD for global-odometry is around 0.566731 m, and 0.536695 m for RTAB-Map-odometry. For better understanding of the distribution of error, the histogram of pose error is considered. Figure 4-18 (c) depicts the histogram of odometry results. According to these two histogram plots, most of the error results are situated around their mean values that are 0.760602 m and 0.624923 m for global-odometry and RTAB-Map-odometry respectively. Figure 4-18 (b) shows the statistics in a box plot format. In this plot the lower line is minimum, the upper dots represent outliers including the maximum one, and the line in the middle is median. The last plot is violin histogram that is a combination of box plot and kernel density plot. Unlike box plot which can only shows a summary of statistics, violin plot depicts a summary of the statistics and the density of the variables as well. For instance, in the first violin plot related to the global-odometry, most of the distribution of pose error is around 0.7 m.



(a)



(b)

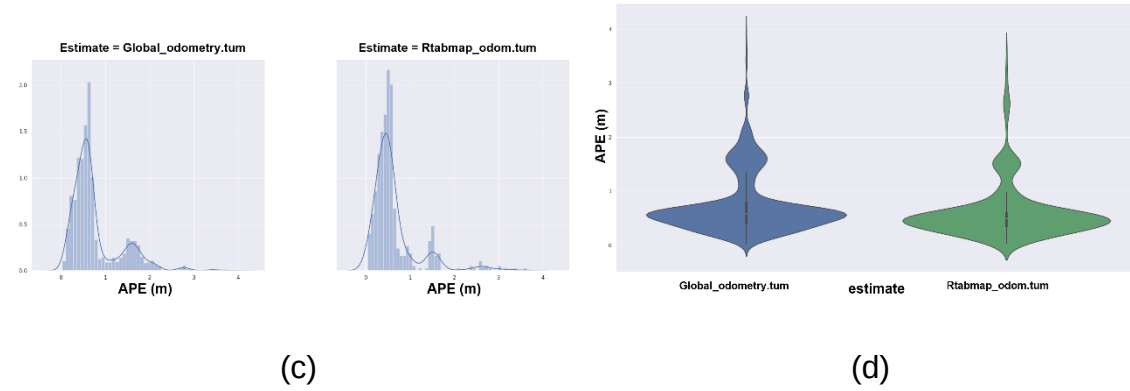


Figure 4-18 Absolute positioning error results. (a) APE-Statistical result (b) APE-Box plot (c) APE-Histogram (d) APE-Violin histogram.

In the following section the APE statistics, seen in Figure 4-18, are presented in tabular form in Table 4-1. The translation error is based on the pose estimate versus the ground truth. This error is in meters while the rotation error is in degrees.

Table 4-1 APE statistical metrics (in meters) for global-odometry and RTAB-Map-odometry.

	MAX	MEAN	MEDIAN	MIN	RMSE	STD
Global-odometry	4.01381	0.760602	0.590148	0.0263803	0.948525	0.566731
Rtabmap-odometry	3.62909	0.624923	0.491668	0.0377201	0.823754	0.536695

Then, a further error analysis is conducted, and the APE is calculated over time, to evaluate the overall consistency of the trajectory relative to the ground truth. Figure 4-19 and Figure 4-20 show the variation of APE over time with respect to the translation component of global-odometry and RTAB-Map-odometry during the exploration of the vehicle in the environment. For instance, Figure 4-19 shows the global localization with a pose estimate error of about 4 m and an average error of 0.7 m, indicating that a low drift of global localization was obtained. For off-track navigation, the error of a few meters around 3 to 4 meters is acceptable in GPS-denied environments.



Figure 4-19 The APE results with respect to the translation part of global-odometry

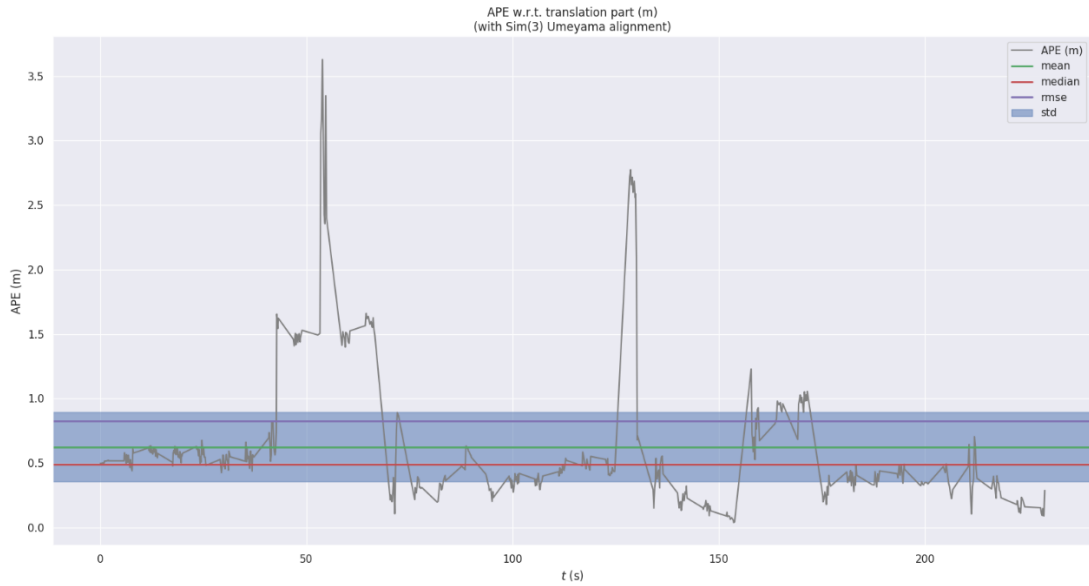


Figure 4-20 The APE results with respect to the translation part of RTAB-Map-odometry

Figure 4-21 presents the results from different localization configuration including EKF and RTAB-Map providing global-odometry and RTAB-Map-odometry respectively. In this figure, the absolute pose error related to the translation part is shown in relation to the time, where a small value represents a small error

between the ground truth and the current pose estimations. As it is clear at three different times, the APE has a number of jumps in its values. This is related to those times that the vehicle has U-turns, so the wheels begin slipping causing drift in the pose of the vehicle. In this plot, the trajectory has absolute timestamps (Absolute means using the original value of the timestamp in the plot) that are large numbers, because they are UNIX epoch timestamps. The long value is just displayed more compact in the plot by subtracting an offset, as can be seen from the offset $+1.643888e9$ in the lower right corner.

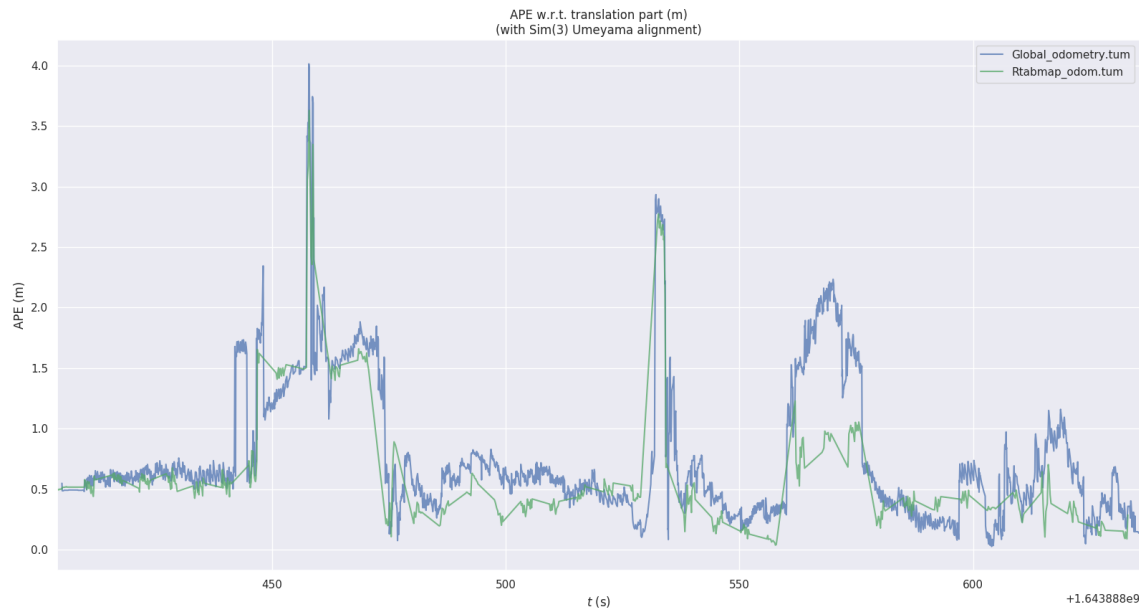


Figure 4-21 The comparison curve of APE results with respect to the translation part (m). The curves of blue and green represent global-odometry and RTAB-Map-odometry respectively.

As defined, another metric for estimating the error of trajectory is RPE. RPE is used to calculate the difference between ground truth and estimated pose changes at the same time interval, which is suitable for estimating the drift of the system. In Figure 4-22 the statistics of the translation component of the relative pose error is presented in a bar plot. The unit for the translation error is in meters. As it is stated this type of metric is applied to measure the local accuracy of the trajectory over time. Figure 4-22 (a) is statistical results including standard deviation, RMSE, minimum error value, median, average error, and the maximum error value. Figure 4-22 (b), (c), and (d) show the RPE-box plot, histogram, and

violin histogram results respectively. The RPE metric of the experiments are summarized in Table 4-2.

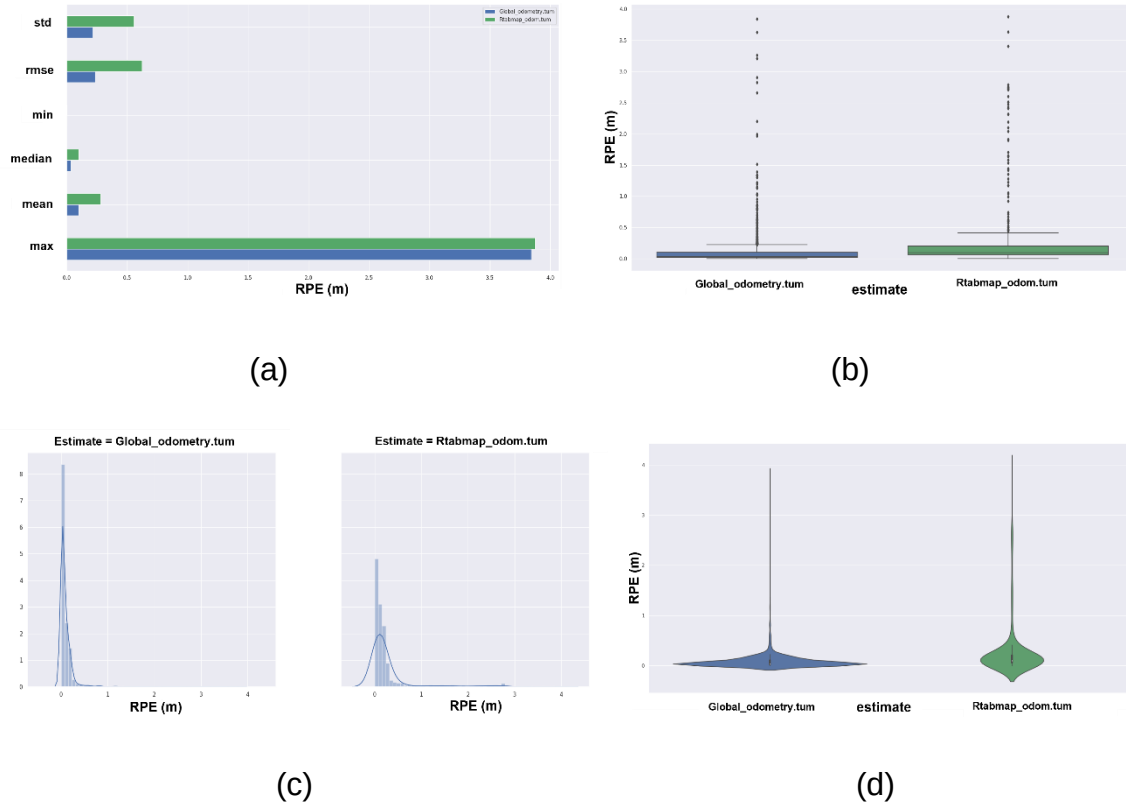


Figure 4-22 Relative positioning error results. (a) RPE-statistical result (b) RPE-box plot (c) RPE-histogram (d) RPE-violin histogram.

Here the RPE statistics which are shown in Figure 4-22, are summarized in Table 4-2. From this table, it can be seen that the relative pose mean error for Global-odometry and RTAB-Map-odometry are 0.0966 m and 0.2783 m respectively. It shows the relative pose error of fusion of GPS and RTAB-Map odometry is much less than the only applying RTAB-Map.

Table 4-2 RPE statistical metrics (in meters) for global-odometry and RTAB-Map-odometry

	MAX	MEAN	MEDIAN	MIN	RMSE	STD
Global-odometry	3.84377	0.0966925	0.0332677	2.74367e-06	0.235396	0.21462
Rtabmap-odometry	3.87756	0.278381	0.0973642	0.000719767	0.621471	0.555635

Figure 4-23 and Figure 4-24 represent the translation part of RPE over time, related to the Global-odometry and RTAB-Map-odometry respectively. It can be seen that global-odometry has a lower mean, median, RMSE, and STD compared to RTAB-Map-odometry.

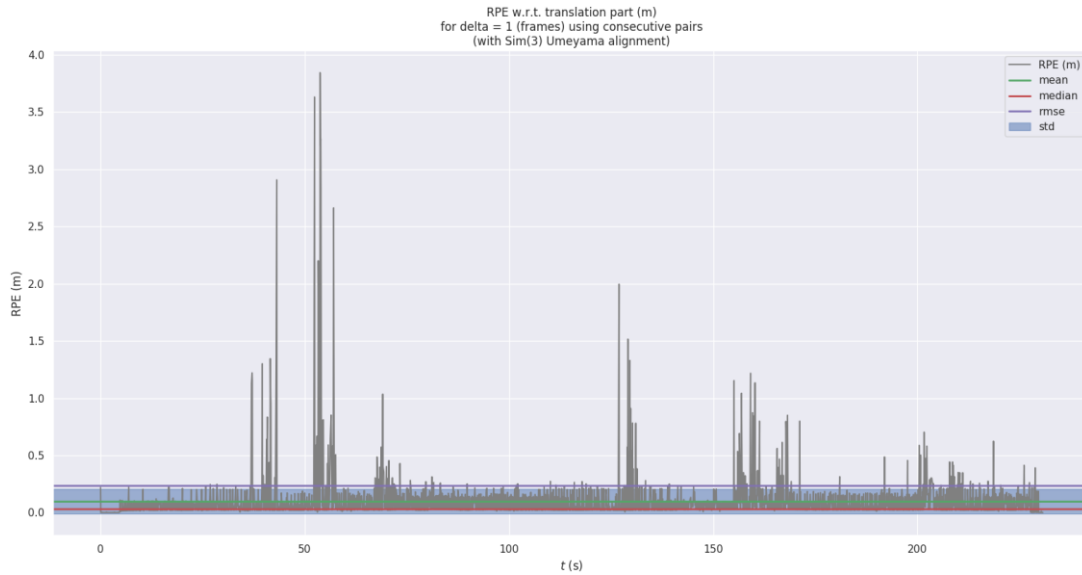


Figure 4-23 The RPE results with respect to the translation part of Global-odometry over time.

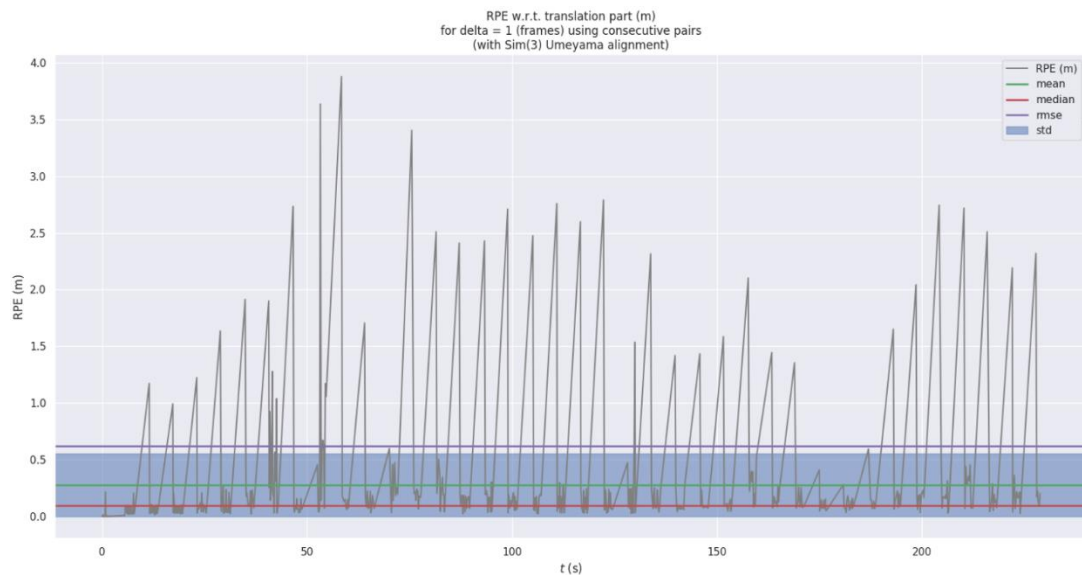


Figure 4-24 The RPE results with respect to the translation part of RTAB-Map-odometry over time.

Figure 4-25 represents RPE of both implemented methods over time in the same plot. This figure gives a better view of the differences between the two methods over time. The relative pose error of Global odometry is more consistent throughout the trajectory compared to the RTABMAP odometry.

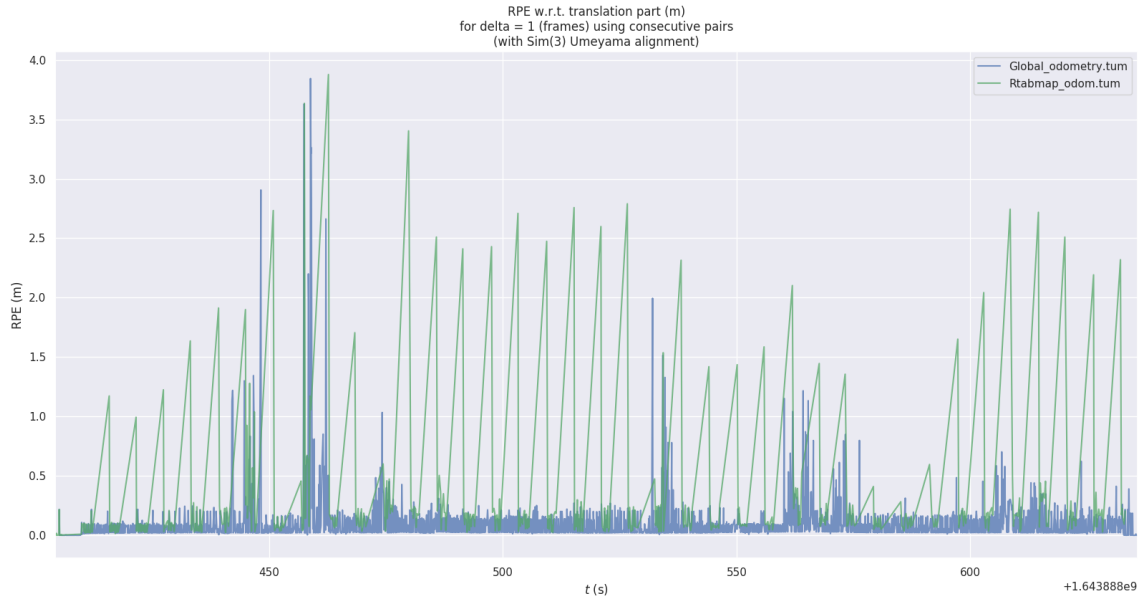


Figure 4-25 The comparison curve of RPE results with respect to the translation part (m) over time. The curves of blue and green represent global-odometry and RTAB-Map-odometry respectively.

Based on the provided results, it can be concluded that when GPS is available, the fusion of RTAB-Map odometry and GPS data through the EKF estimator provides less pose error at the absolute and relative levels, compared to only applying RTAB-Map.

4.2 Evaluation of On-track Localization and Navigation

This section presents the performance and the results of the applied localization and navigation methods on the railway track. In this regard a number of methods and techniques alongside a combination of python libraries, and ROS packages were utilized, implemented and tested on the real railway track. In this regard, the procedure is in this way that, first, the vehicle starts moving autonomously based on GPS odometry to reach the GPS waypoint defined for the system, while mapping of the environment is continuously running through RTAB-Map. After

that the next step will be executed, the vehicle is commanded to start moving while searching for the references which are already saved in the memory of the system. After detecting the preferred object, it starts estimating the distance to the detected object. This phase is fulfilled by merging information from an on-board RGB-D camera and 3D LiDAR. In the following first the test area considered for capturing the data is described and thereafter the acquired datasets, and the results of the applied approaches are provided and analyzed.

4.2.1 Test Area

The first test site considered for the demonstration of the performance of the RIRS was the Northampton & Lamport Railway (N&LR), a heritage railway in Northamptonshire, England, illustrated in Figure 4-26. It was about 400 m.



Figure 4-26 The robotic inspection and repair system (RIRS) on the Northampton & Lamport Railway

In this test area different designed experiments were executed for different purposes that one of them was the autonomous navigation of the RIRS from base to goal location based on GPS waypoint. For this aim, first, a place on the rail was designated as the base position for the RIRS, and another place as the goal location. The latitude and longitude of the goal position were recorded as the desired point that the vehicle should reach. But as the GPS does not work properly in some parts of the railway environment, the RIRS did not reach the goal location precisely. So, for this reason, the idea of track-side object detection

and considering them as reference points were proposed to compensate for the error caused by GPS. In order to implement this idea, and also have precise measurement, reliability, and repeatability, it was decided to do the tests under control mode and on the railway track situated in the Cranfield test area. Figure 4-27 shows part of a real railway track, considered for the demonstration of the RIRS.



Figure 4-27 Experiment environment for executing various designed tests

Therefore, the Cranfield's Rail and Innovation Test Area (RITA) was considered for this aim. Network Rail through the In2Track2 program, procured and provided the 20-yard-long decommissioned track panel that was used for this research. Figure 4-28 shows RIRS at the test area and the Google map view of the area. The first 5.5 m, from the initial location to the goal location (specified with the yellow cone), was considered for the first phase, and the next 5.7 m was considered for the implementation of the second phase. In this research, a dummy defect, a QR code has been attached on the top of the left rail. So, the vehicle was considered to stop 1.5 m before the QR-code to be able to do some designed repair tasks with the universal arm, attached to the body of the vehicle. In this test, the final location where the vehicle should stop is called the defect location.



Figure 4-28 Views of the test site from Google Map

4.2.2 Ground truth and Datasets

To evaluate the performance of the implemented localization approach, a ground truth was chosen at the beginning as a reference. In this research GPS data gained from the NovAtel SMART6-L [130] was considered as ground truth. Before starting the test, a set of GPS data (latitude and longitude) was captured and stored as ground truth. The frequency rate of GPS data was 20 Hz.

To verify the accuracy of the RIRS location on the railway track and evaluate the performance of the defined techniques for the on-track part, the tests were carried out 16 times. Considering the same condition for the tests, and also the low speed of the vehicle (0.3 m/s), and the time for running each test, just these number of tests were conducted. The repeatability of the vehicle location is fundamental to the efficient detection and monitoring of rail track faults, and subsequent corrective actions. Finally, the captured results were compared with the reference values. Sensors considered for these tests were RTK-GPS, IMU, wheel odometer, RGB-D camera, and 3D LiDAR. The speed of the vehicle during all these tests was at a constant value of 0.3 m/s. It should be noted that the tests were conducted at this speed because the RIRS performs inspection at a slow speed (maximum 1 m/s), and also this was an appropriate speed for forcing the trolley to move on the railway track. Needless to mention that there will be a speed loss during on-track navigation because of the frictional contact between the pneumatic wheel of Warthog and the contact roller of the trolley.

4.2.3 Navigation based on the fusion of GPS and SLAM

In this phase, the RIRS was at the initial location, and a GPS waypoint was specified for the vehicle, shown in Figure 4-29. The GPS waypoint was intentionally considered a few meters (5.7 m) away from the defect location, to show that if the maintenance vehicle, on a real-world track, stops far from the defect location, how it would navigate and find the defect location precisely?



Figure 4-29 Initial and goal GPS locations specified for the vehicle.

The visualization of the on-track test in Rviz and Rtabmapviz is displayed in the following figure. In this plot, the camera provides images of the on-track and track-side objects, and LiDAR represents the 3D point cloud of the environment. Rtabmapviz shows the 3D map of the environment, and the blue line represents the camera path, attached to the body of the vehicle. This graph-based path is generated by RTAB-Map SLAM.

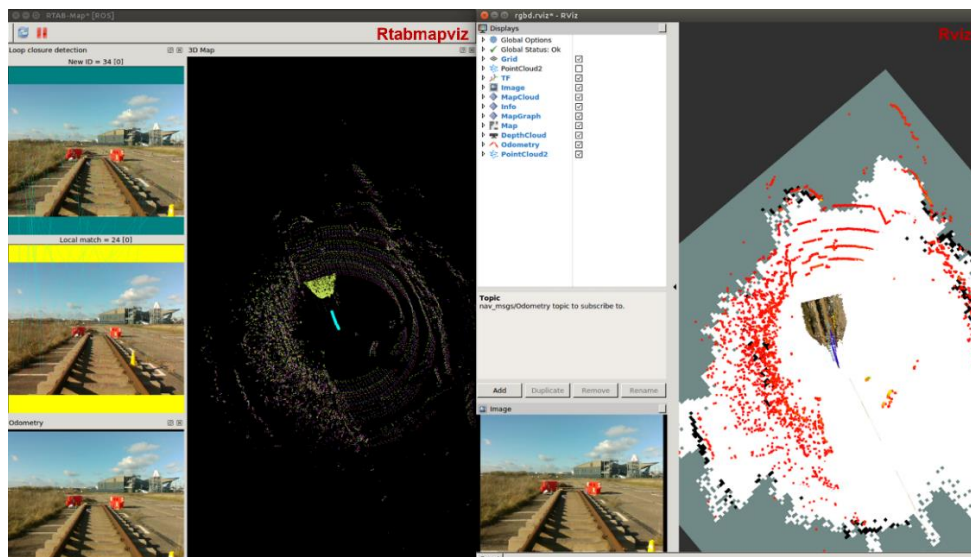


Figure 4-30 Representation of on-track test in Rviz and Rtabmapviz

The autonomous navigation process would start by running a python code with a predefined GPS waypoint as a goal for the vehicle (appendix 7.4A.1 and 7.4A.2).

Then the code related to the navigation of the vehicle for reaching the GPS waypoint was run (appendix 7.4A.3). In the corresponding terminal the initiation of the GPS 2D navigation goal node, and the GPS status was published. The navigation of the robot was based on the integration of current position of the robot odometry (*rtabmap/odom*) and the GPS odometry information. Figure 4-31 shows the first state of the vehicle after receiving the navigation command. By the movement of the vehicle on the railway track the amount of distance that the vehicle is heading toward the goal is published in meters.

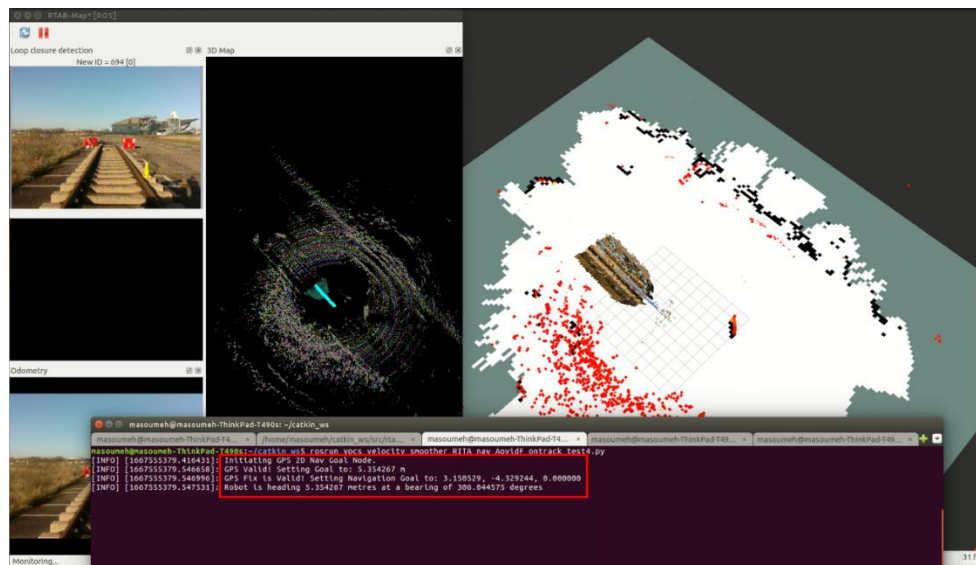


Figure 4-31 Initiating the GPS 2D navigation goal node.

By the movement of the vehicle, mapping of the environment is performed by RTAB-Map, represented in Figure 4-32. The local location of the vehicle, relative to the map reference and global location based on GPS, is published continuously in the related terminal. By reaching the defined goal location, the status of the completion of the navigation is shown in the terminal.

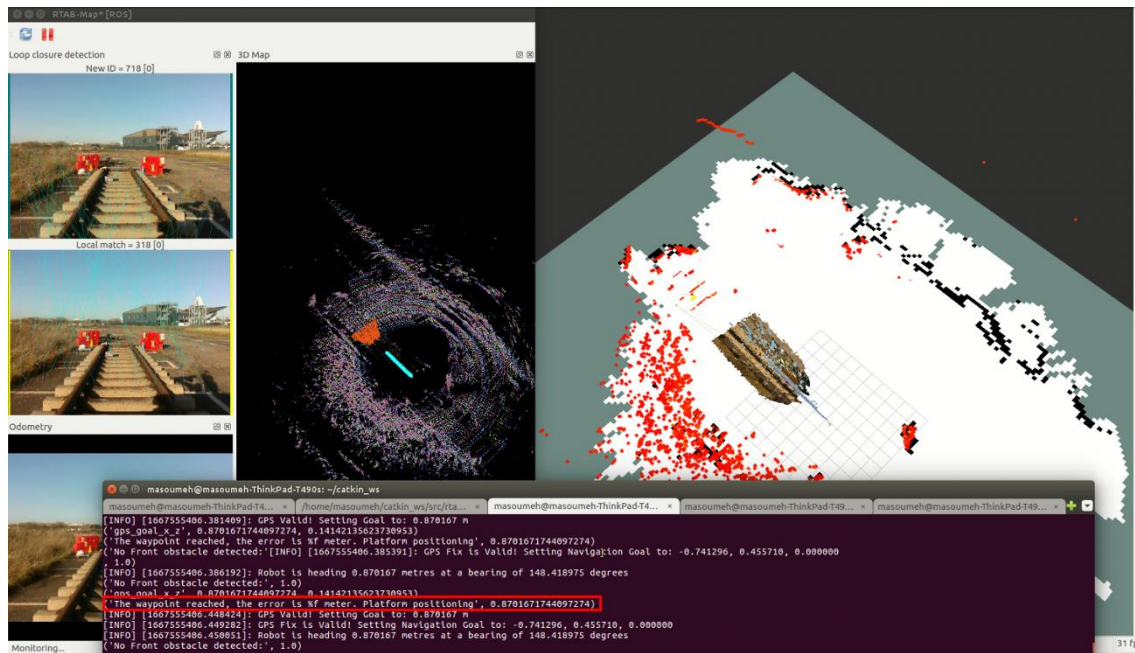


Figure 4-32 The completion of navigation based on GPS waypoint.

The latitude and longitude of the vehicle's location gained from each run of the test were saved and then plotted in Figure 4-33 to see how they were scattered around the mean value. In this way, it was possible to specify whether they are scattered a lot from each other or not. In other words, the precision of the acquired GPS data can be specified.

The precision is the degree of closeness of observations to their mean value. The acquired GPS data is specified with its latitude and longitude coordinates which are in decimal degrees. Decimal degrees are the notation for expressing the latitude and longitude of geographic coordinates as decimal fractions of a whole degree. The space between each line of latitude or longitude representing 1° is divided and expressed as decimals. Data were obtained from the RTK-GPS receiver attached to the tower of the Warthog vehicle.

In the following figure, it is shown that 87% of latitude and 69% of longitude of the whole dataset were situated within one standard deviation (STD) of their mean value. The STD regulates the dispersion of the distribution. A diminished STD suggests that the data is closely grouped near the mean (resulting in a taller normal distribution), while an increased STD implies that the data is more dispersed around the mean (yielding a flatter and broader normal distribution).

So according to the definition of precision and the scattering of data in Figure 4-33, it can be said that three-quarters of the latitude data and more than 50% of the longitude data are placed within one STD of the mean, and this indicates a relatively level of high precision of the whole dataset. It's not extremely tight, but it's reasonably consistent. Having a majority of the data clustered closely around the mean suggests that the measurements are consistent and have a small spread or variability. This indicates a higher level of precision compared to situations where the data is more widely dispersed.

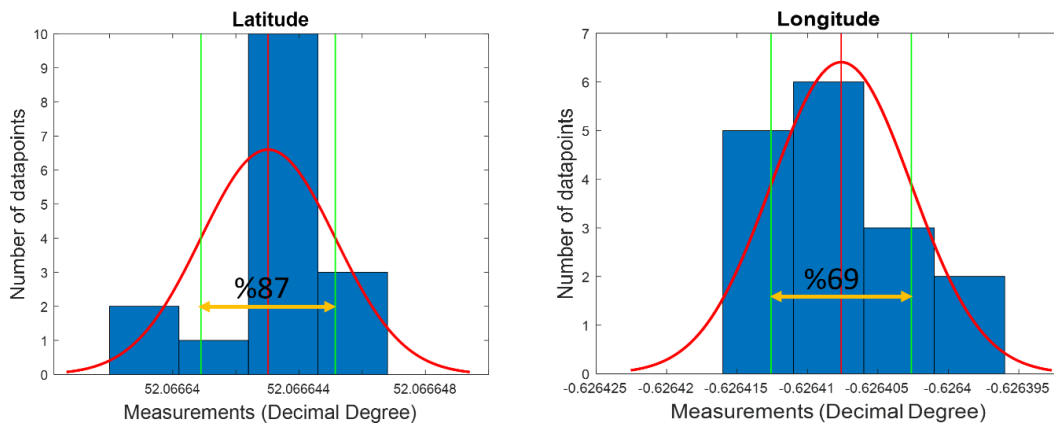


Figure 4-33 Scattering of the GPS data of the goal location.

In the following table the values related to the mean, STD, and variance of the acquired GPS data of the goal location are provided. The measured values are in decimal degrees.

Table 4-3 Mean, STD, and Variance of the GPS data of the goal location

	Mean	STD	Variance
Latitude	52.066643	2.127322e-06	4.525501e-12
Longitude	-0.626408	4.980285e-06	2.480324e-11

Figure 4-34 figures out how much the GPS location that the vehicle has stopped at, is far from the ground truth. In this regard, the difference between GPS considered as ground truth and the GPS location received from each test were computed in meter scale. According to this plot, the least difference was about

30 cm, and the highest one was about 75 cm. This error means that the vehicle has stopped around the goal location. Needless to mention that the error related to the GPS data, on the day that the test was running, was also calculated. In such a way that while the vehicle was stopped in one location, the distance between two gained GPS points was calculated based on the haversine formula, and it was varying between 10 cm to 20 cm.

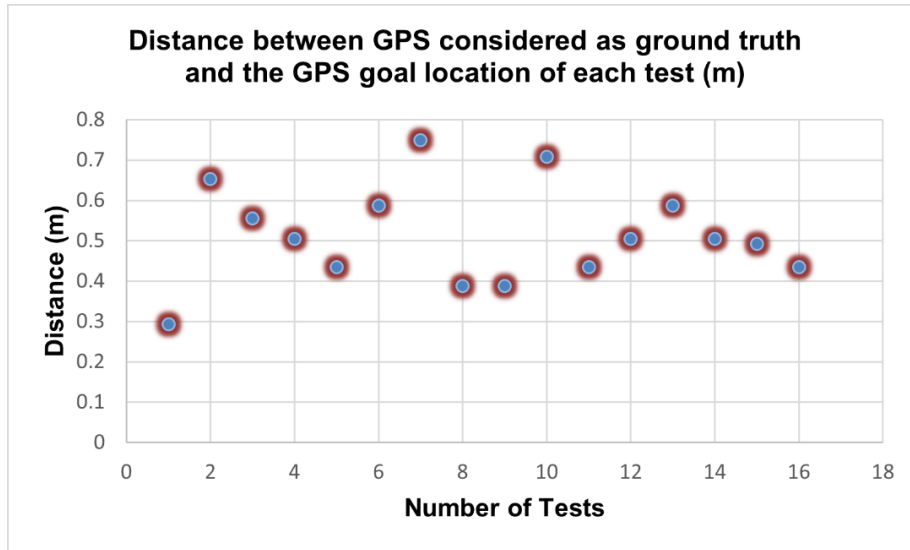


Figure 4-34 Distance between GPS considered as ground truth and the GPS goal location of each test (m)

Based on the provided analysis of the RTK-GPS data, it can be said that despite RTK-GPS provides a much higher rate of accuracy compared to the consumer-grade GPS, it can also suffer from a number of uncertainty sources such as the number of available satellites, satellite constellation, weather conditions, atmospheric Interference, or large obstacles close to the receiver, which can lead to multi-path errors, satellite occlusion, and specifically for railway environments signal reflectance and noise due to overhead electrical power lines. As far as RITA (test area) is far from any obstacles and GPS obstruction, the number of satellite and their positioning as well as the weather condition can be considered as the main reasons for these results. In the following, it is shown that during the test the robot has achieved a DGPS fix level. As it is reviewed before, the GPS fix data is available via the standard `/fix` topic. To determine whether the robot has achieved the RTK fix level or not, the easiest way is to check the “Status” of

the */fix* topic. An RTK fix will display “2” in the status field, if it is “1”, it is indicative of DGPS fix level. After extracting the status of the captured GPS data of the test day, showing the robot has achieved the DGPS fix level throughout the test. Therefore, this means that the number of satellites is less than 5 one in common between the base station and the receiver attached to the tower of the Warthog. The repositioning of the base station was also done, but the results did not change.

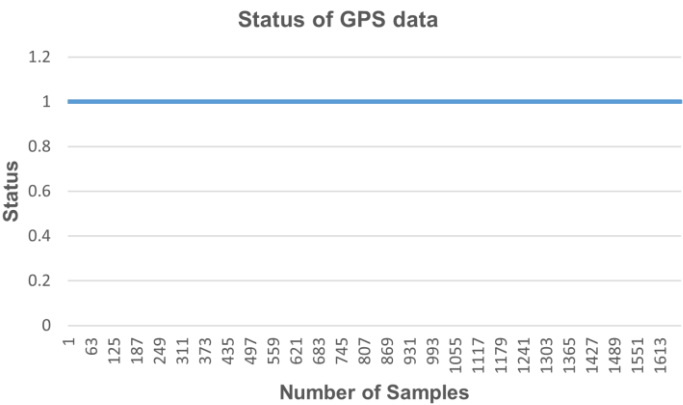


Figure 4-35 Status of GPS data

The latitude versus longitude of the GPS data, captured during the movement of the robot on the railway track, is also plotted in the following figure. The values are in decimal degrees.

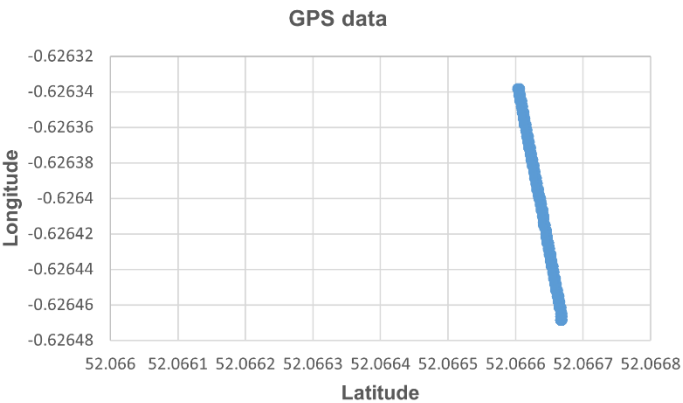


Figure 4-36 Latitude vs longitude of the captured GPS data

4.2.4 Object Detection and Distance Estimation

This phase was designed to eliminate the error that might happen by the GPS outage in some parts of the railway network. As it is reviewed before, railway assets and the rail environment influence the positioning system and might lead to GPS error. This error might cause the autonomous vehicle to stop a few meters away from the determined location (defect location). In this phase, track-side objects were employed as references to improve the vehicle positioning and achieve a higher level of accuracy as well as find the defect location more precisely. This phase has high importance for an autonomous maintenance vehicle, as errors in vehicle location cause errors in maintenance jobs, and higher errors, might lead to the failure of the job.

The strategy in this phase is in this way that a Python code related to object detection and distance estimation was run. Another code was also run to move the RIRS forward with a constant speed of 0.3 m/s. Therefore, the RIRS started moving autonomously on the railway track and upon detecting the track-side objects, the distance to the identified objects were estimated. In this test, two objects (barriers) were considered as track-side references, shown in Figure 4-37. More objects provide more validation to the location of the vehicle and increase the precision of defect location detection. In the object detection process, environment illumination, colour, shape, and size of the objects are important and impact the result. As the colour and shape of these objects are the same, and there are no more discriminative features in the environment, for avoiding the failure of object detection and also adding more discriminative features to them, a number of images were attached to the body of these objects, and a number of cones were put next to them. In this way, objects were detected with a higher probability.



Figure 4-37 Track-side objects from the point of view of RGB-D camera

The following figure represents track-side objects which were detected based on the feature matching approach and specified with the colourful bounding boxes. The RTAB-Map approach was also running continuously and provided a detailed map of the environment, and the location of the vehicle locally. The point cloud was represented in Rviz (right-side window).

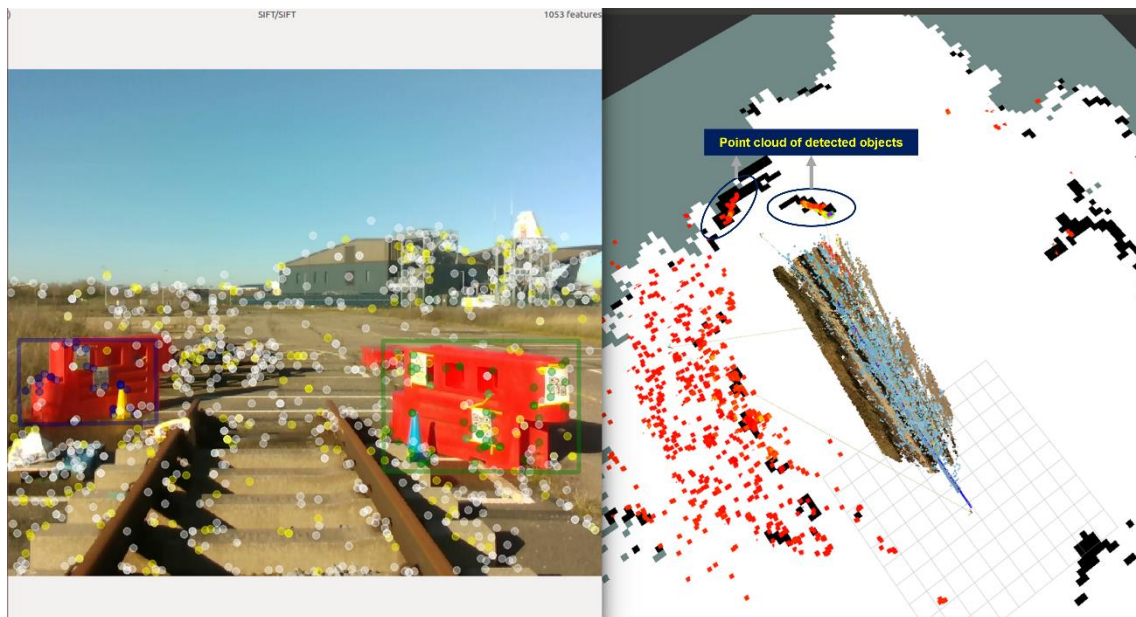


Figure 4-38 Track-side object detection and the related point cloud in Rviz

Now for accessing the point cloud of the detected object and then estimating the corresponding distance, first the points which are representative of noise and other objects in the environment need to be filtered out. So, for this reason, a

voxel grid filtering was applied. The voxel grid filter is a form of point cloud down-sampling that maintains the majority of the information stored in the point cloud. The parameters in this filter can be defined in a way that down-sample the points in the x, y, and z axis, and only keep the points related to the determined object.

Figure 4-39 shows the detected object in the left window and the related filtered point cloud in the right window. Point cloud provides a message of *sensor_msgs/PointCloud2* which holds a collection of N-dimensional points, which may contain additional information such as normals, intensity, etc. The point data is stored as a binary blob, its layout described by the contents of the "fields" array. So, first, this type of message should be converted to an array with points in the directions of x, y, and z. This array indicates the distance from the LiDAR sensor to each point in the x, y, and z directions. Among these points, the nearest one was extracted and considered the point for specifying the distance. The related terminal in Figure 4-39 shows the LiDAR start estimating the distance to the detected object, and by reaching the specified threshold, it would stop. The threshold was already determined and recorded in the inspection phase for the first time.

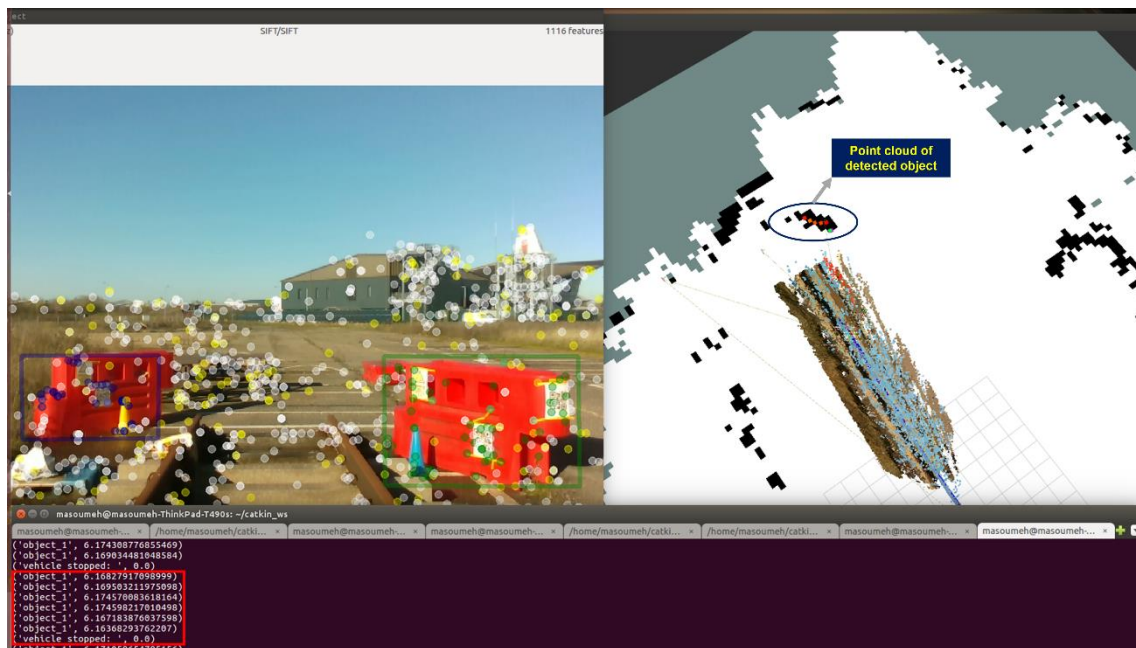


Figure 4-39 Point cloud filtering and estimating the distance to the object.

As it is noted before, the on-track test was carried out 16 times to check the repeatability and reliability of the vehicle positioning. Then each time, after stopping the vehicle at the defect location, the distance from object to the LiDAR was recorded and then plotted to see how much the error is. Figure 4-40 represents the results. The reference distance from object to the LiDAR was 6.5 m, but after each test, this distance was less than the reference one. These results were captured from the LiDAR output and read from the related terminal, indicating the vehicle has stopped after the determined location. Based on the error plot, most of the error results are between 0.3 m and 0.4 m. But in two of the tests, number 9 and 10, the error reached its highest values, 0.7 m, and 0.8 m respectively.

As object detection and distance estimation modules are correlated, and it has been defined that after object detection, and validation of the environment, it should start estimating the distance. So, one reason which led to this error was the object detection procedure, which was not fast, and robust enough. As discussed in section 3.3.3.3, some reasons impact the object detection results such as the type of the applied feature descriptor, weather conditions, and illumination of the environment. Another reason is the frictional power loss between the Warthog's pneumatic wheel and the trolley's contact roller. This caused the vehicle to stop with more delay, compared to the situation that the vehicle moving on the ground.

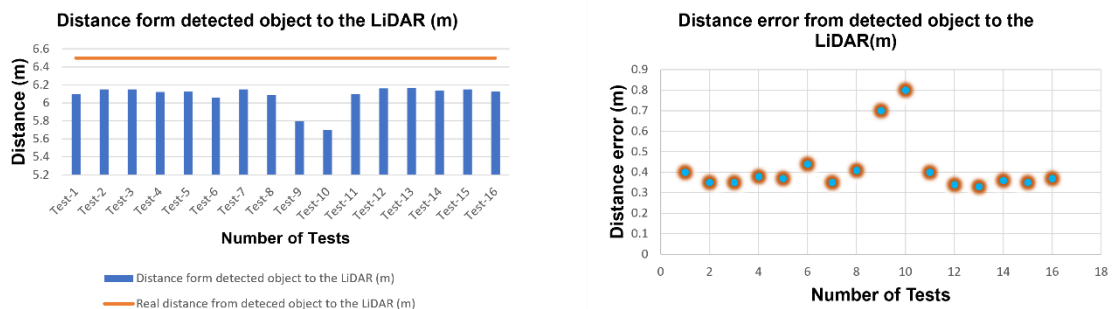


Figure 4-40 Distance from detected object to the LiDAR and the corresponding error

The other measurement, which was conducted to figure out the accuracy of the vehicle location at the stop location was GPS data. The acquired GPS of the vehicle's location after each stop was compared with the GPS data considered as ground truth. Figure 4-41 demonstrates the latitude and longitude of the vehicle's location after each stop, which was saved and then plotted. This shows how GPS data were situated relative to their mean value. 88% of the latitude data and 75 % of the longitude were within one STD of the mean, which shows the high precision of the acquired data. The latitude and longitude coordinates are in decimal degrees.

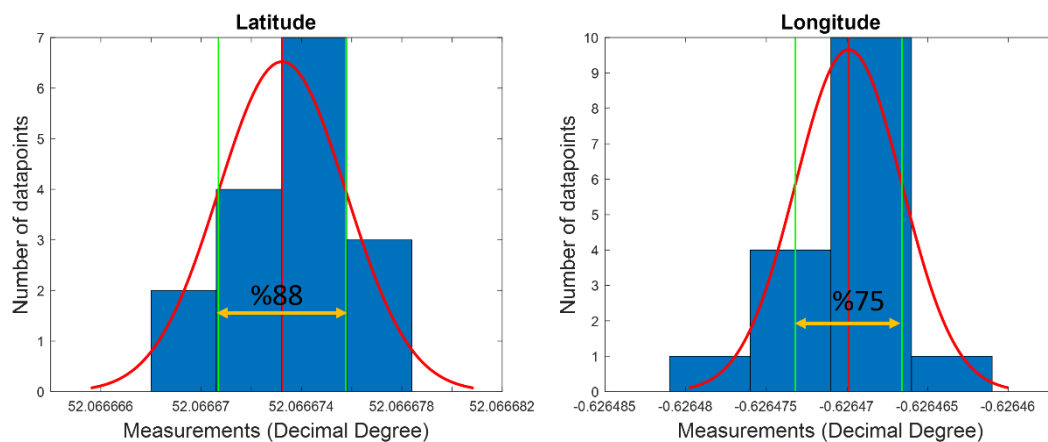


Figure 4-41 Scattering of the GPS data of the defect location.

In the following table mean, STD, and variance related to the measured latitude and longitude of the GPS of the defect location are provided. The measured values are in decimal degrees.

Table 4-4 Mean, STD, and Variance of the GPS data of the defect location

	Mean	STD	Variance
Latitude	52.066673	2.543601e-06	6.469906e-12
Longitude	-0.626470	3.303687e-06	1.091435e-11

In this part of the test, GPS was not applied as an input of the applied methods, just utilized to check how much the GPS of the location that the vehicle has stopped at, is far from the ground truth. The distance between the GPS

considered as ground truth and the GPS location received from each test were computed in meter scale. According to the Figure 4-42, the least distance error is about 16 cm, and the highest error is about 57 cm.

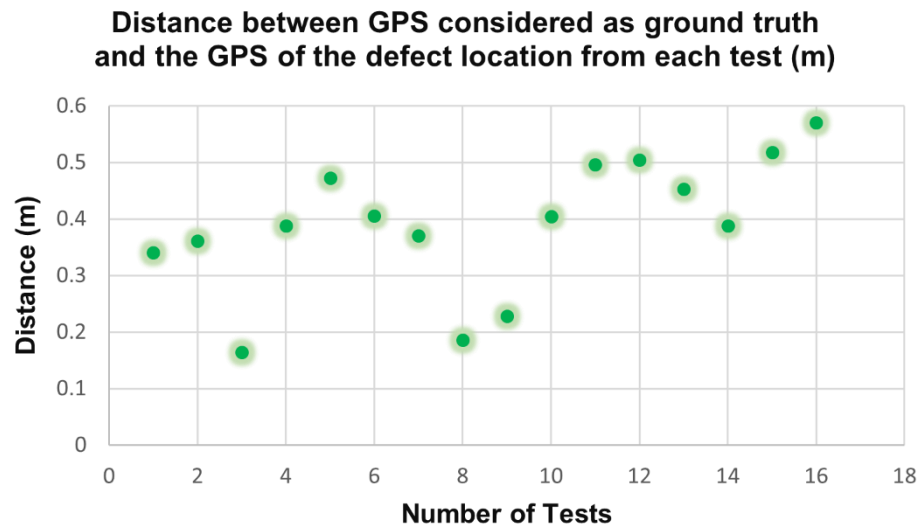


Figure 4-42 Distance between GPS considered as ground truth and the GPS of the defect location for each test (m)

Another measurement that was done for validating the vehicle location was the distance between the front wheel of the trolley and the defect location, specified in Figure 4-43. As it is noted before a 1.5 m distance was considered between the dummy defect and the front wheel of the trolley, for the arm to perform a number of designed maintenance jobs at the defect location. It was expected that after detecting the track-side object and estimating the distance from the identified object to the LiDAR sensor, the vehicle be stopped at a distance of 1.5 m from the defect location (dummy defect). But the vehicle stopped with a delay. As the localization of the vehicle matters, finding the error rate in this part is also critical in better performing the maintenance job.



Figure 4-43 Distance from defect location to the front wheel of trolley

Here, after each test, this distance was measured and compared with the reference value (1.5 m). According to the acquired results, all the vehicle's stop locations were after the specified location. Higher error in the vehicle's location increases the probability of failure of the repair tasks. The measured error for two of the tests (number 9 and number 10) was not good, but for the rest the results were satisfactory. The total errors were between 0.01 m and 0.22 m. The highest error rate was related to test numbers 9 and 10 with 0.22 m and 0.21 m respectively. On average the error was around 0.07 m.

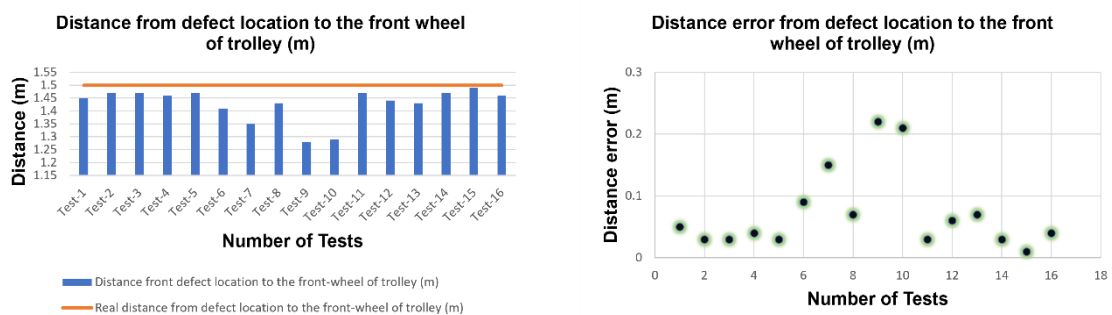


Figure 4-44 Distance from defect location to the front wheel of trolley and the corresponding error

One of the main reasons which caused this error was the slippage between the Warthog wheel and the contact rollers of the trolley. In other words, the slippage

of the Warthog wheels on the trolley made the vehicle stop after a delay. Another reason was the high inertia and low friction of the wheels of the trolley on the rail which resulted in the robot taking longer to come to a stop and increased the braking distance. The high inertia means that the vehicle has a greater resistance to changes in its velocity, while the low friction reduces the opposing force that would slow it down. As a result, the vehicle would continue moving for a longer period of time before eventually coming to rest. This added a few cm errors to the results. The other reason was the delay that happened during the object detection and distance estimation processes. The object detection process was not trained for different weather conditions, so it was so sensitive to illumination and weather conditions.

4.2.5 Performance of the RIRS at Different Speeds

The previous results were gained while the speed of the vehicle was at a constant value of 0.3 m/s. It is obvious that the speed of the vehicle impacts the results, and it is one of the criteria which should be taken into consideration. In this regard, in this section object detection and distance estimation procedures were executed at different speeds to see the impact on the results. In this test, it became clear the threshold related to the highest speed at which the RGB-D camera and LiDAR would be able to detect the object and estimate the 3D pose of the object.

Figure 4-45 represents the distance from the defect location to the front wheel of the trolley at the speeds of 0.4 m/s (blue line), 0.5 m/s (orange line), and 0.6 m/s (grey line). Based on these results, with the increment of the speed, this distance starts decreasing, which means that the object detection module and distance estimation part are impacted by the higher speed, causing the vehicle stops with a more delay. This test was also done with a speed of 0.7 m/s, in which the vehicle failed to stop. It passed through the defect location. Finally, it stopped with the remote emergency stop. Based on this result, a speed of 0.6 m/s can be considered as the threshold that the vehicle would stop around the defect location but with a higher error compared to the lower speeds.

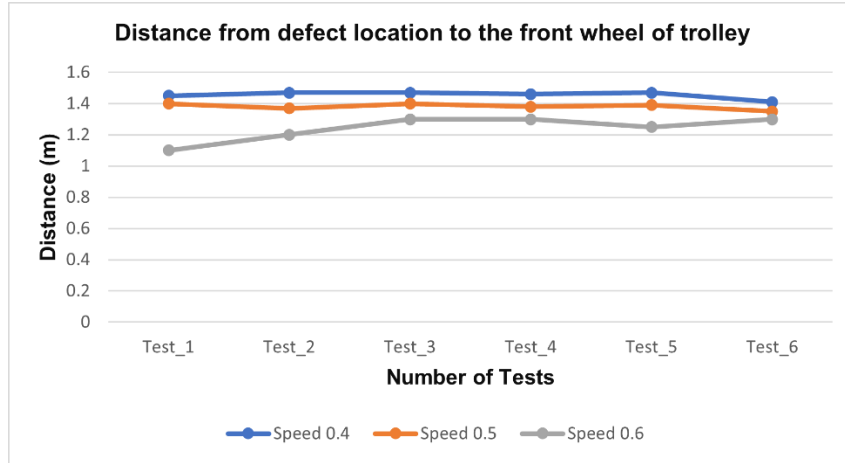


Figure 4-45 Impact of various speed on the vehicle localization

In the following figure, the impact of various speeds on the other results can also be seen. For instance, the distance from the detected object to the LiDAR starts decreasing with the increment of the speed. As it can be seen, at the speed of 0.6 m/s, the vehicle stops after traveling 0.8 m from the determined location.

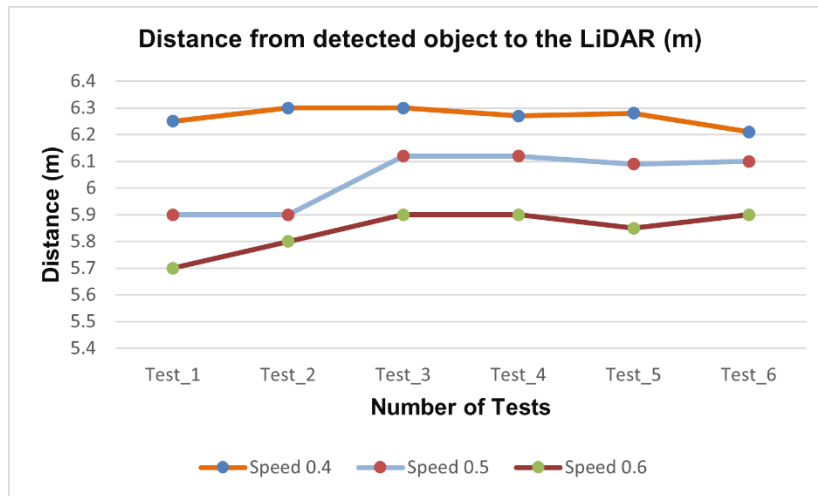


Figure 4-46 Impact of various speeds at the distance from detected object to the LiDAR

4.3 Summary

In this chapter results and analysis of off-track and on-track localization and navigation were illustrated. To estimate the pose of the vehicle more accurate and have a higher rate of global pose, EKF estimator was implemented for the off-track navigation. At this stage odometry information generated from RTAB-

Map approach also merged with GPS odometry and other sensor odometry information, to estimate the pose of the vehicle even during GPS outages. For the off-track phase, static object detection and collision avoidance also implemented for the RIRS with the aim of increasing the safety of the system. For the on-track phase, localization and navigation based on GPS was conducted. In this phase, first Navigation based on GPS odometry was implemented, and then for reaching to the determined location (defect location) track-side objects are considered as a reference for finding the location. For this reason, object detection based on feature matching and distance estimation to the detected object were executed.

5 DISCUSSION

5.1 General Discussion

Autonomous maintenance robots, deployed on railway tracks, play a crucial role in augmenting the overall operational efficiency and safety of railway infrastructure. These advanced robotic systems contribute significantly to the early detection and rectification of defects, thereby mitigating potential hazards and minimizing service disruptions. In this research, a maintenance robot named RIRS was considered for operating on the railway track to do inspection and repair tasks.

Precise localization and mapping are indispensable components of autonomous maintenance systems for railway tracks. Accurate localization ensures that the robot can determine its precise position along the track, thereby enabling comprehensive coverage of the railway network. Simultaneously, precise mapping allows the system to create a high-fidelity representation of the railway infrastructure. These functionalities were achieved through a fusion of advanced sensor technologies, such as RTK-GPS, IMU, and wheel odometer, coupled with a graph-based algorithm named RTABMAP for simultaneous localization and mapping (SLAM). Such a robust localization and mapping system empowers the robot to navigate along the tracks with precision, ensuring that it can access specific locations efficiently.

Moreover, one of the main roles of this autonomous maintenance system is in finding the fault point repeatably which is a prerequisite for condition-based maintenance on the railway track. This robot has four pneumatic wheels which makes it capable of navigating on the side of the railway track and performing some tracksides maintenance tasks such as removing vegetation, removing debris, inspecting, and repairing trackside furniture, inspecting point machines, etc. So, to enable the vehicle to be able to transit from its base location to the specified location under world mode autonomous navigation, several approaches and algorithms were developed and implemented. In this regard, for implementing navigation using GPS, some collection of ROS packages as well

as several python libraries to enable the robot to move from one target location to another, were utilized. Although these packages are available open-source for GPS navigation, none of them work out of the box and require extensive tuning of parameters. The tuning of parameters is not limited to the parameters related to the EKF but also involves tuning the parameters in the navigation module. These parameters provide the setting for the collision avoidance and pre-configured path planners as well.

5.2 Synchronization

Before running any Python program and the RTAB-Map SLAM approach, the synchronization between the Warthog and the PC was established. First, the *ntpdate* function was installed. It sets the local date and time by polling the NTP servers specified to determine the correct time. Then the *chrony* function was applied to adjust the system clock that runs in the kernel to synchronize with the NTP clock server on both the PC and Warthog sides. Finally, the IP addresses in the *chrony.conf* was defined.

Here ROS can maintain synchronized communication between multiple computational devices using *roscnode* and *rostopic*. In RIRS, the PC inside the Warthog vehicle acts as a master computer while the connected PC acts as a slave. The master computer runs all the drivers, emergency stops, and some sensors (LiDAR, IMU, wheel odometer, GPS, RGBD camera). On the other hand, the slave PC runs, for instance, the RGB-D camera, processes the data generated from the camera, and transfers the output with the master via *roscnode* and *rostopic*. Moreover, all the executable scripts are running on the PC.

5.3 RTAB-MAP

For simultaneous localization and mapping of the environment for both around and on the railway track, RTABMAP algorithm was deployed. It employs odometry information to estimate the robot's pose (position and orientation) as it moves along the railway track, to build an effective map.

In section 4.1.1, RTABMAP was run to generate the 3D map of the environment, localization information, and loop closure detection during the movement of the

vehicle. Another key feature of this method is localization correction after each loop closure to correct most of the odometry drift happens over run (section 4.1.1). For achieving better results in terms of localization and mapping, RGB-D camera, 3D LiDAR, IMU, Wheel odometer and the RTK-GPS were deployed as the input sensors. Localization module uses the stored map, and by accessing the current sensor data, tries to define where the robot is located on the map.

By the movement of the vehicle in the outdoor environment, a high amount of data would be received that needs to be managed, so it necessitates RTABMAP being run on a high-resource system to operate effectively without any performance issues or even crashes. It also requires careful parameter tuning to achieve the optimal performance. Finding the right balance between accuracy and real-time performance requires careful parameter tuning (section 4.1.1). During the tests, it was also identified that RTAB-Map not working that much properly on a cloudy and rainy day. Since cloudy days can result in diffused lighting, it can reduce the contrast of features in the environment, which potentially affects the ability of the system to accurately track and map the surroundings.

The presence and quality of features in the environment play a significant role in the performance of RTABMAP as well. It was also identified that If the environment lacks distinctive features or has low-texture regions, it impacts the results in several ways such as difficulty in localization, failure of Loop Closure detection, reducing the mapping accuracy, and increased uncertainty. It was tried to do the tests in an environment with a sufficient number of features.

5.4 Off-track Localization and Navigation

After applying RTAB-Map alongside running the navigation module, the robot starts moving toward the defined destination (global plan), while trying to avoid obstacles as well. In this step, global and local planners were defined for the vehicle to have a safe navigation along its movement to the goal. In this phase, tuning of the parameters such as speed, acceleration, goal tolerance, path and goal biases are highly important as it impacts the smooth movement of the vehicle [146]. The friction between the wheels and the ground for having a smoother

movement of the vehicle, should be taken into consideration as well (section 4.1.2).

As far as the environment alongside the railway track is not fully covered by GPS signal and many places have a very weak GPS signal, the navigation based on GPS would not be completed successfully. For achieving an accurate and high-rate global pose data of the vehicle, EKF estimator was applied. It fuses the continuous sensor data with the global pose estimates from RTAB-Map and GPS.

Section 4.1.3 represents the results related to the fusion of GPS odometry information with the RTABMAP odometry information through the EKF estimator. The APE and RPE were two metrics that were applied to measure the estimated pose of the robot relative to the reference pose from ground truth and measure the drift of the trajectory, respectively. In Figure 4-15 the trajectory output of RTABMAP and EKF were compared with the reference values. Based on this result, where GPS is available, the localization output based on the integration of GPS and RTAB-Map odometry is close to the ground truth, while where RTK-GPS is not received EKF will still continue to estimate the pose of the vehicle with the visual odometry information, but with a more drift relative to the ground truth. To represent the severity of the error during the path, the error is mapped on each estimated route. Figure 4-17 represents that the error mostly happens near the U-turns of the vehicle, where it is near the buildings, while the middle of the path which is far from any GPS disturbances, has the least error.

The ground truth in these two plots in Figure 4-17 is not different, just different poses were used for the comparison because of matching. Poses in ground truth were matched to the poses in the trajectory by matching the timestamps. If fewer timestamps can be matched or there were fewer poses in general, the number of poses would vary. One of the main reasons that EKF also fails to provide high accuracy during the U-turns of the robot is that the angular acceleration is not modelled in the EKF estimator.

Then the absolute pose error of the trajectory was examined in terms of several statistics including Standard deviation (STD), RMSE, minimum error value, median, average, and the maximum error value in Figure 4-18. The magnitude of

the errors is in meters. Lower values of these parameters are preferable and indicate that the robot is performing well in terms of accuracy and consistency in the intended trajectory.

Lower standard deviation suggests that the errors in the robot's trajectory are more consistent and less variable. In other words, the robot is consistently following the intended path with minimal fluctuations. Based on the bar plot, the STD of the global odometry is about 0.03 m higher than the RTABMAP odometry.

In terms of RMSE, lower values means that the robot is closer to the intended trajectory, and the deviations from the intended path are relatively small.

Minimum error happens in some parts of the environment where GPS works properly, and in those parts, the fusion of GPS and SLAM through EKF outperforms applying the SLAM approach. The minimum error in this trajectory containing U-turns is critical, where precision in following the reference path matters. Based on Figure 4-18 the minimum error of global odometry is 0.0113 m less than the minimum error of RTABMAP odometry.

Based on this plot, in a GPS-denied environment, EKF estimates the pose of the robot with a higher error relative to the RTABMAP algorithm. The relative error is around 0.38472 m (Table 4-1). A lower maximum error value suggests that even in a challenging situation such as U-turns, the robot's worst-case deviations from the intended path are limited.

Regarding the median criterion, RTAB-MAP odometry output provides a lower median compared to the Global odometry, about 0.09848 m. Lower median indicates that a significant portion of the errors were smaller.

On average, RTAB-MAP odometry tends to stay closer to the intended trajectory, with 0.13567 m less, compared to the odometry from EKF. This can be justified by the presence of GPS as the input of EKF.

Here the aim was to show that the fusion of GPS and RTAB-MAP odometry can estimate the pose of the vehicle in both GPS and GPS-denied environments. When GPS is working properly, Global odometry provides better accuracy

compared to only applying the SLAM approach, and when GPS is not available, EKF will continue to estimate the location of the vehicle with the visual odometry information, but with a more drift relative to the ground truth. Based on the result acquired in Table 2-1, absolute positioning error where GPS is available is around 0.02 m, and where GPS is not working properly, is about 4 m. For navigation around the railway track, the error of a few meters around 3 to 4 meters is acceptable.

Figure 4-19 and Figure 4-20 show the absolute pose error related to the global localization and RTAB-MAP odometry respectively. The statistics that were compared with each other in a bar plot, are presented on each technique correspondingly. For instance, the average error for global odometry is around 0.7 m, indicating that a low drift of global localization was obtained.

The absolute pose error with respect to the translation part over time, for both global odometry and RTABMAP odometry, are represented in Figure 4-21. According to this plot, a smaller APE value denotes a small difference between the current pose estimations and the ground truth. The graph reveals instances of abrupt changes at specific time points, corresponding to moments when the vehicle executes U-turns. During these manoeuvres, wheel slippage occurs, causing drift in the pose of the vehicle.

RPE was another criterion that was calculated for measuring the local accuracy of the trajectory over time and checking the difference between the ground truth and the estimated pose changes at the same time interval. Both techniques have an increasing and decreasing trend in their estimated RPE (Figure 4-25).

The increasing trend in RPE indicates a growing error in the translation estimation over time. This could be due to factors like sensor drift, cumulative errors, or inaccuracies in the localization system during the movement of the robot. Conversely, the decreasing trend suggests that the system is improving its accuracy in estimating the translation over time. Based on Figure 4-25, global odometry has a more consistence trend over time compared to the RTAB-MAP odometry.

5.5 On-track Localization and Navigation

The on-track operation focuses on the localization of the RIRS while moving on the railway track to search for the defect location. The RIRS was programmed to autonomously move on the railway track while searching for the defect location. After finding the defect, the RIRS performs several programmed jobs in accordance with the defined task. Certain repairs might require multiple moves between the base and defect location, for example, to collect specific tools or parts. Therefore, it necessitates the capability of finding the defect location repeatedly and with high confidence.

For this phase, two separate procedures were designed to show first the navigation of the RIRS based on GPS odometry and then how it can find the defect location if the defect is situated in a GPS-denied environment.

5.5.1 On-track Navigation Using GPS and SLAM

For the first step, the GPS coordinate was deliberately set at a distance of 5.7 meters from the site of the defect (section 4.2.3), to demonstrate how a maintenance vehicle, that has stopped at a significant distance from the defect location because of the GPS outage, would accurately locate the defect. A Python script was created to receive the GPS coordinates of the base and the goal locations in latitude and longitude format and then converted them into the distance (sections 3.3.3.2 and 4.2.3). The RTAB-Map also provides a detailed map of the environment for the vehicle to localize itself related to the objects in the environment (section 4.2.3). The RIRS started navigating autonomously to reach the specified GPS waypoint location. This test was repeated 16 times to show the repeatability and reliability of the vehicle location, as it is important for an autonomous maintenance system to detect and monitor the rail track faults and do the subsequent corrective actions. According to Figure 4-33, the recorded GPS data points of the location of the vehicle after each stop, were around the ground truth. The lowest error was about 30 cm, and the highest one was about 75 cm. The error related to the GPS data, while the vehicle was motionless, was also measured which was fluctuating between 10 cm to 20 cm, as it was cited it could be because of satellite constellation changes, multipath interference,

and/or signal interference. By further investigation, it was also identified that the number of satellites working at the RTK level was not enough (at least 5 Satellites). Hence with this number of satellites, the GPS system was at the DGPS level, and the accuracy of this level is between a few cm to a few meters. As a result, the range of error for this step can be justified based on these reasons.

5.5.2 On-track Object Detection

For the next part of the railway track, it was assumed that GPS was not available. during the autonomous movement of the RIRS on the railway track, RTAB-Map provides the map of the environment as well as the location of the vehicle relative to the map frame (section 4.2.4). But for finding the defect's location, and identifying where and when the vehicle should stop, trackside furniture or trackside equipment which were positioned on the side of the railway track were considered as reference points. They helped the localization system of the vehicle to localize itself more precisely and provided an advantage for the robot navigation. For detecting the track-side reference points, object detection method based on feature matching and for 3D pose estimation of the detected object, LiDAR was deployed (section 4.2.4).

It was identified that the type of the camera impacts the object detection results. In this research D435i RGBD camera was deployed, while if the ZED 2i camera was used, the results were more reliable [147]. Because the ZED 2i generates a more reliable and accurate depth map of the real scene, if the objects were texture-less, and very dark, along with shadow and low illumination, while the depth image provided by D435i would not be so accurate for these conditions (section 3.3.3.4).

Totally, based on the tests done and the results obtained, the following points including illumination of the environment, weather conditions, and the size of the object should be considered for having a faster and more successful object detection process.

- Shape of the object; the properties of an object impact the detection. These properties correspond to characteristics of objects' position, size, and shape. Boundaries (connected edges) capture the characteristics of the shape object. Size or dimension of the object should not be too small or so big. Small objects contain low number of features, which makes it difficult to be detected. The size of the considered objects should be in the point of view of the vision and LiDAR sensors.
- Colour of the object; the 'colour' of an object is the wavelengths of light that it reflects, so if the colour of the object has a low wavelength, the object would not be detected, and the process of detection would lead to failure.
- Illumination of the environment; environmental factors such as illumination changes impact on visual perception. Object detection in low-illumination environments is very challenging. Due to insufficient reflected light, image captured contains a lot of dark areas and noise. Object detection under low illumination is often ignored.
- Discriminative objects provide discriminative features, and they would prevent any wrong detection due to repetition of the same objects around the railway track.

5.5.3 On-track Distance Estimation

In this research an RGB-D camera (D435i) was utilized for the aim of distance estimation as well, but it was subsequently found that the depth image of this type of camera does not provide a reliable pose estimation. To show this, first, the 3D position of the detected object was extracted and then the relative depth image point cloud was processed. The estimated 3D position was obtained through applying "*find_object_2d*" package from ROS. A *tf* frame was also created automatically for the detected object. The object position is with respect to the camera-link frame. Once the object was added to the ROS *tf* frame, the pose (position and orientation) of the object was extracted.

By further analysis and investigation of the relative depth image, it was found that the range of the provided depth image by the RGB-D camera varies between 3

to 4 meters, and it would not be a trustworthy source for medium and long-range distance estimation. Furthermore, the relative depth image would not define a clear point cloud of the large-size or tiny-size objects. Figure 5-1 demonstrates RGB and depth images of one frame of the camera. It is obvious that the detected signal-post in the RGB image does not have a clear point cloud in the related depth image. The point cloud of the signal post is merged with the point cloud of the sky resulting in a noisy image.

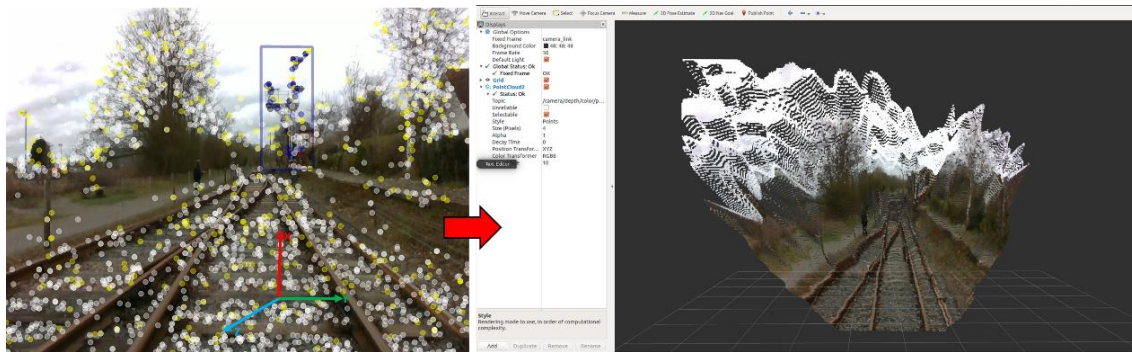


Figure 5-1 RGB and depth images of the RGB-D camera

In the next phase, the distance from the detected object to the camera, in the x-axis, which is along the railway track, was extracted. It was expected that by the movement of the vehicle on the railway track, the distance starts decreasing, but as Figure 5-2 shows, it had an erratic behaviour. Based on this output, it is not clear which of these values is reliable and indicative of the correct distance between the object and the camera-link frame. Based on this investigation, it was concluded to apply LiDAR for distance estimation.

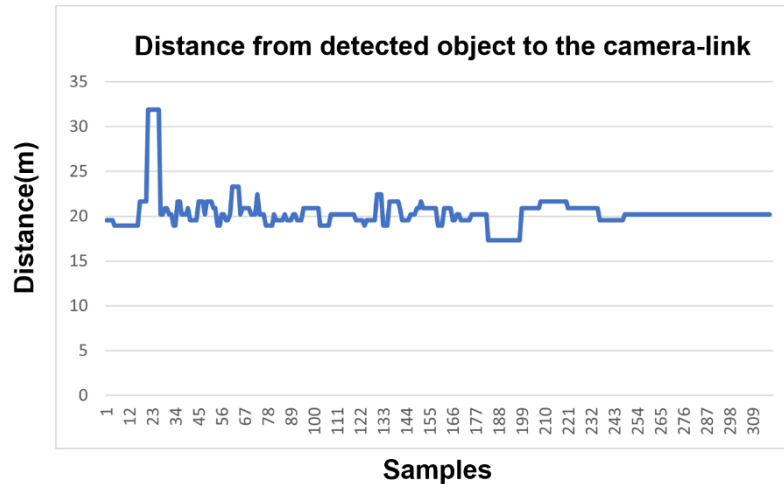


Figure 5-2 Distance from the detected signal-post to the camera-link when the RIRS moving on the railway track

Based on Figure 4-40, the distance error from the detected object to the LiDAR sensor was between 0.3 m to 0.8 m, indicating the vehicle was stopped after the determined location. Here the distance was defined from the LiDAR-link frame to the captured nearest point of the detected object. Because of the positioning of the object, it was difficult to extract distance from LiDAR-link frame to the centre of the object. Owing to the movement of the vehicle on the rail, which has a bit of slippage, it might cause the location of the captured nearest point might vary each time. Railway tracks and trackside areas are uneven as well due to factors such as stones, sleepers, and slopes, so the vehicle encounters vibration and/ or slippage while navigating. Then, this would cause the LiDAR not to be able to capture the same points every time. For this reason, this would cause an error in the estimated distance, and the distance might be different each time.

5.5.4 Robot Location Evaluation

After detecting the reference objects around the railway track and estimating the distance to one of the objects, the RIRS stops by the defect location, but with an error. Section 4.2.4 examined the main reasons that caused errors in the robot's location. Figure 4-41 shows how the GPS data were situated relative to their mean value. Based on Table 4-4, the low standard deviation and variance of the collected GPS data indicate that the GPS system is precise, and that the data is

reliable. Figure 4-42 Distance between GPS considered as ground truth and the GPS of the defect location for each test (m)Figure 4-42 shows the distance difference between GPS gained from the position of the vehicle from each test and the GPS considered as ground truth, varying between 16 cm and 57 cm. The unavailability of enough numbers of satellites and environmental conditions impact the GPS data accuracy.

Figure 4-44 depicts the distance error between the front wheel of the trolley and the defect location. It was between 0.01 m and 0.22 m. This error was triggered by the RIRS's motion on the railway track, where despite the smoothness facilitated by the rail wheel of the trolley, the issue stemmed from the low friction between the Warthog's pneumatic wheel and the contact roller of the trolley, coupled with high inertia. Another reason caused a positional inaccuracy, was the slippage of the Warthog on the trolley. This can be due to insufficient traction between the wheels of the warthog and the surface of the trolley. The trolley surface is smooth or slippery, so slippage is likely. Another factor contributing to the issue was the inadequate performance of the employed object detection algorithm. This inadequacy arose from the algorithm's lack of training for diverse conditions such as varying weather, illumination, and objects of different sizes, among other factors.

6 CONCLUSION

Aim statement: implementing autonomous localization and navigation for the RIRS on and around the railway track for the purpose of precisely enabling inspection and repair.

Robotic maintenance systems are highly demanded (section 2.1) owing to their flexibility and ability to combine the autonomous sensing, control, decision making, and remote communications for inspection and repair. They are proposed to conduct maintenance tasks with higher accuracy and reliability. In this thesis a robotic solution named RIRS was introduced for the inspection and repair tasks on the railway track. This vehicle runs with Robot Operating System (ROS) kinetic version, and is equipped with various sensors including Wheel encoder, IMU, RTK-GPS, RGB-D camera, and 3D LiDAR. Fusion of these sensors at local and global levels were utilized for the environmental perception, static object detection, pose estimation, mapping, and navigation of the vehicle.

The on-track and off-track operational capability of the RIRS, has made it unique and provided many advantages over other conventional robotic and autonomous systems. Localization and navigation are the two most important aspects of a robot operating in a railway environment, and the subject of this thesis. Higher location accuracy capability is an enabler of the RIRS concept and essential, for detecting and potentially monitoring of rail flaws along the railway track. As it conducts inspection and repair tasks, it might need to find the fault point repeatedly and with high confidence. However, vehicle localization with high precision, considering the railway environment, is extremely challenging. Section 2.1 discusses the uncertainty sources which impact the vehicle's operation.

Various localization strategies, positioning measurements, fusion techniques, and mapping approaches were investigated and tested to address the gap in the knowledge (section 2.10). In this thesis, an autonomous localization and navigation system for the RIRS was implemented, and enabled, and finally assessed. In this research, two separate phases of applying RIRS in railway network was examined: off-track and on-track localization and navigation.

6.1 Off-track Localization and Navigation

Objective statement: Implementing, testing, and validating the off-track autonomous navigation in a GPS and GPS denied environment.

For the off-track phase, a graph-based simultaneous localization and mapping approach named RTAB-Map was implemented. It provides robust and low-drift odometry information, robust localization, and supports input from RGB-D camera, and LiDAR. It also combines two main algorithms including loop closure detector and graph optimization. The architecture, and sensor fusion in this type of technique were covered in section 3.3.2.3. The achieved results related to localization, mapping and identifying loop closure were provided in section 4.1.1.

Then, an Extended Kalman Filter (EKF) estimator was applied for merging GPS data with the navigation module in ROS (section 3.3.2.5). EKF estimates vehicle's pose by merging sensor data locally and globally. Here, the Wheel odometer and IMU were fused to provide a locally accurate state estimate, and in another level GPS and SLAM odometry were fused with other continuous sensor data to provide an accurate and complete global state estimate (section 4.1.3). EKF also estimates the pose of the vehicle in locations where GPS not working properly, with a low drift. The uncertainty sources which contribute to vehicle position error and influence positioning accuracy and reliability were also investigated (section 2.2). These sources included objects around the railway track, tunnels, stations, overhead line equipment, and so on.

In GPS-denied environments EKF still continue to estimate the pose of the robot from visual odometry velocities (section 3.3.2.5). EKF has also provided the pose of the vehicle at a higher rate of frequency (20 Hz) and overcome the disadvantage of the low data frequency from RTAP-Map (10 Hz) (section 3.3.2.5).

In section 4.1.3, the performance of the localization and navigation of the RIRS in GPS and GPS-denied environments was analysed and verified. For the assessment of the gained results, two different evaluation metrics, named Absolute Pose Error (APE) and the Relative Pose Error (RPE) were utilized. Furthermore, path planning as well as detecting and detouring static objects were

implemented and tested. Off-track object detection was for the aim of increasing the safety of the RIRS.

6.2 On-track Localization and Navigation

Objective statement: Enabling localization on the railway-track in absolute accuracy.

For the on-track phase, the precise RIRS positioning had much more importance as it was essential to the robot command & control, task execution, safety, and efficiency. But in almost 20% of the railway network, GPS might not be available (section 2.2), and the RIRS which was commanded to find the defect location on the railway track, would fail to find it and stop a few metres away from it. Therefore, a test was designed to demonstrate how the RIRS would operate and reach the determined goal location when GPS is available, and how it would find the defect location situated in a GPS-denied environment (section 3.3.3.1).

First, autonomous navigation based on GPS odometry was implemented, alongside executing RTAB-Map for local localization and creating a map of the environment. After that an evaluation of the vehicle's location at the goal location was performed, to find out the rate of error. This error was gained by calculating the distance between GPS considered as ground truth and the GPS measured at the vehicle's stop location after each test (section 4.2.3).

Then for reaching the defect location, which was assumed that it was situated in a location without GPS, a feature matching approach and pose estimation were fulfilled. For feature matching the track-side objects were considered as the reference points for finding the defect location and specifying the vehicle location with more accuracy. For evaluating the error after reaching the vehicle to the defect location, three parameters were evaluated after each test including distance from the detected object to the LiDAR, distance from the defect location to the front wheel of the trolley, and distance error between GPS considered as ground truth and the GPS of the defect location.

Finally, this scenario was tested at different speeds (section 4.2.5) to find out how much the maximum speed is that the vehicle would be able to find the defect and finally stop at the defect location, and how much the error is at different speeds.

6.3 Evaluation of The On-track Results

Objective statement: Assessment of the on-track localization and navigation system.

A large number of measurements were conducted to assess the localization and navigation of the RIRS on the railway track. The test for this phase was conducted 16 times to check the repeatability of the system for reaching the same defect point. These measurements included the calculation of the distance between the GPS considered as ground truth and the GPS of the goal and defect locations, the distance from the detected object to the LiDAR, and the distance from the defect location to the front wheel of the trolley. In terms of evaluating the GPS of the goal location, the error was between 0.3 m to 0.75 m, and for the time that the vehicle reached the defect location the distance error was between 0.16 m and 0.57 m. Error related to the distance from LiDAR to the detected object was between 0.3 m to 0.8 m, considering that most of the error results were around 0.3 to 0.4 m. Finally, the error related to the measured distance from the front wheel of the trolley to the defect location was between 0.16 m to 0.57 m (section 4.2.4).

The performance of the robot at different speeds was tested in section 4.2.5. It was identified that with the increment of the speed, the accuracy of object detection and distance estimation techniques would decrease, and the RIRS would stop with more error or may fail to stop at higher speeds (0.7 m/s). The outcome of this research has proven the usability of robotic technology for railway track maintenance jobs and real industrial applications. Most of the off-the-shelf maintenance systems work to several meters accuracy; track geometry refinement work at about meter-level, while this research addressed vehicle localization in cm-level of accuracy. This ensured the RIRS was capable of performing maintenance tasks in a real railway.

6.4 Contributions and the Potential Impact of this Research

This research is part of a larger project (IN2SMART2), which supports the intelligent railway infrastructure asset management in the ERTMS (European Rail Traffic Management System). In this research, multiple approaches, and sensor fusion techniques were reviewed, analysed, and implemented to enable the autonomous localization and navigation for the RIRS on and around the railway track. The contributions of this research can be described as follows:

- Implanting Extended Kalman Filter (EKF) for incorporating the sensors locally and globally to make the RIRS operate in GPS and GPS-denied environments with a higher rate of frequency and an accurate pose.
- Analysing and Implementing object detection based on feature matching and 3D pose estimation for localizing the RIRS on the railway track in GPS-denied environments.
- Implementing on-track navigation based on GPS odometry, alongside implementing RTAB-MAP SLAM for providing local localization and a detailed map of the environment.

Enabling localization for the RIRS on and around the railway track in GPS and GPS-denied environments has made it capable of performing track inspection and repair jobs. In this regard, some of the potential impacts of this research can be summarized in the following points:

- Providing a context for future railway off-track maintenance jobs.
- Leading the maintenance tasks to be completed faster (save time) and improving the network availability.
- Enhancing the safety zone for the vehicle by implementing static object detection and collision avoidance (in the off-track scenario).
- Providing business benefits to the railway industry by reducing the need for human labour, which can potentially lower costs associated with track maintenance. Additionally, by catching and addressing issues early, the system can help prevent more serious and costly problems from developing.

- Having potential for new applications in other industries, such as manufacturing, agriculture, or mining, where autonomous vehicles could improve efficiency and safety.

Overall, this research has the potential to make a significant impact on the railway industry by improving safety, efficiency, and cost-effectiveness, while also paving the way for new autonomous systems in other industries.

7 FUTURE RECOMMENDATIONS

RIRS was designed for the purpose of inspection and repair jobs on the railway track. Therefore, specifying the localization of such a kind of system has high importance. RIRS demonstrated the high potential of autonomous systems deployment in maintenance tasks on the railway track. For this reason, the research opportunities in this field will continue in the future. By identifying and diminishing the shortcomings of current research, the performance of this system can be improved.

7.1 Applying Machine Learning Methods for Localization

One of the main points which can be considered for the future is the application of machine learning approaches for the localization and navigation field. Although these approaches have been applied already in the field of environment perception, deep learning methods in localization and navigation has not received the same level of attention and maturity. There is a lot of promise in using deep Learning algorithms, especially RNN to tackle sequential localization data and improve it further. Deep learning approaches can be used to replace certain modules of SLAM algorithms. Learning algorithms may in the future provide an end-to-end deep learning localization and mapping system that can avoid feature modelling and data association, reducing errors and uncertainties associated with unmodeled dynamics and imprecise modelling.

7.2 Applying Map Matching method for Localization

Another way for achieving high accuracy for vehicle localization in a railway environment is to use a technique called "map matching". In this approach, the vehicle's position is estimated by matching its sensor measurements to a known map of the railway environment. By using prior knowledge of the track layout, speed limits, and other relevant features, the vehicle's position can be estimated with high accuracy even in GPS-denied environments. Map matching can be performed using various techniques, such as probabilistic filtering or pattern recognition algorithms, and it can be used in combination with other positioning technologies such as INS or track circuits to improve accuracy.

One of the advantages of map matching is that it can be used even in GPS-denied environments, as it relies on prior knowledge of the environment rather than on a satellite signal. However, one limitation is that the accuracy of the map matching solution is highly dependent on the accuracy and completeness of the map data. Therefore, it is crucial to ensure that the map is accurate and up to date to achieve reliable localization results.

7.3 Improving RTAB-MAP

Improving the performance of the RTAB-Map approach which provides simultaneous localization and mapping of the environment, would lead to better accuracy, faster performance, and increased reliability. In this regard, there are a few approaches that can be considered such as

- Optimizing its performance by reducing the number of features used for visual feature extraction.
- Optimizing the algorithm uses CPU and GPU resources.
- Enhancing feature detection and matching by using different feature detectors and descriptors
- Combining multiple feature detectors to improve the robustness of the system.
- Improving the accuracy and reliability of loop closure detection which can lead to a more accurate SLAM system.

This could involve investigating ways to incorporate additional information such as odometry or GPS data to improve the detection of loop closures.

7.4 Applying Machine Learning for Image processing

The machine learning approaches can be applied to the object detection process. In this research, RGB-D camera was used for object detection, using open-source ROS packages named "*find_object_2d*". Though this ROS package was successful in detecting the track-side objects in good lighting condition, it might fail to detect in difficult lighting condition. So here machine learning algorithms and deep neural networks can be used to improve the object detection process and pose estimation. A dataset of images in different weather condition in

unstructured lighting condition can train an image processing algorithm for object detection to be applied in real world scenario.

REFERENCES

- [1] "Using analytics to get European rail maintenance on track | McKinsey."
<https://www.mckinsey.com/industries/public-and-social-sector/our-insights/using-analytics-to-get-european-rail-maintenance-on-track>
(accessed Mar. 17, 2022).
- [2] Y. K. Al-Douri, P. Tretten, and R. Karim, "Improvement of railway performance: a study of Swedish railway infrastructure," *J. Mod. Transp.*, vol. 24, no. 1, pp. 22–37, Mar. 2016, doi: 10.1007/S40534-015-0092-0/FIGURES/11.
- [3] "REPORT FROM THE EIM-EFRTC-CER WORKING GROUP ON MARKET STRATEGIES FOR Track Maintenance & Renewal FOLLOW UP TO THE CONCLUSIONS OF EC INNOTRACK PROJECT/WP5."
- [4] "Rail Track Maintenance: A Career Explained - Rail Futures."
<https://railfutures.co.uk/rail-track-maintenance-a-career-explained/>
(accessed Jan. 27, 2022).
- [5] "Cost benchmarking of Network Rail's maintenance and renewals expenditure Annual report: year 2 of control period 6," 2021.
- [6] C. D. Martland, "Analysis of the potential impacts of automation and robotics on locomotive rebuilding; Analysis of the potential impacts of automation and robotics on locomotive rebuilding," 1987. doi: 10.1109/TEM.1987.6499032.
- [7] L. E. Parker, "ROBOTICS APPLICATIONS IN MAINTENANCE AND REPAIR," 1998.
- [8] N. Mahfuz, O. A. Dhali, S. Ahmed, and M. Nigar, "Autonomous railway crack detector robot for Bangladesh: SCANOBOT," *5th IEEE Reg. 10 Humanit. Technol. Conf. 2017, R10-HTC 2017*, vol. 2018-January, pp. 524–527, Feb. 2018, doi: 10.1109/R10-HTC.2017.8289014.
- [9] J. Dhiyaneswaran, M. Vimal Raja, B. Ashok Kumar, and C. B. Muthu

- Kumaran, "Design and fabrication of railway track cleaning bot," *Mater. Today Proc.*, vol. 37, no. Part 2, pp. 2677–2680, Jan. 2021, doi: 10.1016/J.MATPR.2020.08.524.
- [10] "Robotics and industrial AI for track maintenance - Global Railway Review." <https://www.globalrailwayreview.com/article/114137/robotics-and-industrial-ai-track-maintenance/> (accessed Mar. 16, 2022).
- [11] S. D. Errico, "Simulation of a railway asset maintenance robot," vol. 44, no. 0, p. 750111, 2018.
- [12] C. Esveld, "Modern Railway Track Second Edition."
- [13] "Intelligent Innovative Smart Maintenance of Assets by integRated Technologies 2 | IN2SMART2 Project | Fact Sheet | H2020 | CORDIS | European Commission." <https://cordis.europa.eu/project/id/881574> (accessed Mar. 30, 2023).
- [14] M. Rahman, M. Rahimi, A. Starr, I. Durazo Cardenas, A. Hall, and R. Anderson, "An overview of the challenges for railway maintenance robots," *UKRAS22 Conf. "Robotics Unconstrained Environ. Proc.*, vol. 5, pp. 84–85, 2022, doi: 10.31256/bx2wd2o.
- [15] M. Rahman, M. Rahimi, A. Starr, I. D. Cardenas, A. Hall, and R. Anderson, "Challenges for a Railway Inspection and Repair System from Railway Infrastructure," *2022 10th Int. Conf. Control. Mechatronics Autom. ICCMA 2022*, pp. 210–215, 2022, doi: 10.1109/ICCMA56665.2022.10011603.
- [16] G. Schanzer, U. Schneider, and J. Troelsen, "The challenges of using satellite navigation systems for high precision railway positioning," 2000, Accessed: May 23, 2023. [Online]. Available: www.witpress.com.
- [17] "Real-Time Kinematic (RTK) | NovAtel." <https://novatel.com/an-introduction-to-gnss/resolving-errors/rtk> (accessed Apr. 26, 2023).
- [18] "Which correction method? | NovAtel." <https://novatel.com/an-introduction-to-gnss/resolving-errors/which-correction-method> (accessed May 23,

2023).

- [19] R. Siegwart and I. Nourbakhsh, "Prediction of Position," *Introd. to Auton. Mob. Robot. MIT Press.*, pp. 159–230, 2011.
- [20] D. Kim, "Commercializing Localization for Automated Driving: Absolute vs. Relative," pp. 1–7, 2019.
- [21] X. Guo, "Feature-Based Localization Methods for Autonomous Vehicles," p. 129, 2017.
- [22] U. Scheunert, H. Cramer, and G. Wanielik, "Precise vehicle localization using multiple sensors and natural landmarks," *Proc. Seventh Int. Conf. Inf. Fusion, FUSION 2004*, vol. 2, pp. 649–656, 2004.
- [23] B. A. Renfro, J. Rosenquest, A. Terry, and N. Boeker, "An Analysis of Global Positioning System (GPS) Standard Positioning System (SPS) Performance for 2015," *Tr-Sgl-17-04*, p. 105, 2017.
- [24] J. L. Wang, S. Y. Huang, and C. S. Liao, "Time and frequency transfer system using GNSS receiver," *Radio Sci.*, vol. 49, no. 12, pp. 1171–1182, 2014, doi: 10.1002/2014RS005460.
- [25] Y. Morales and T. Tsubouchi, "DGPS, RTK-GPS and StarFire DGPS performance under tree shading environments," *IEEE ICIT 2007 - 2007 IEEE Int. Conf. Integr. Technol.*, no. February, pp. 519–524, 2007, doi: 10.1109/ICITECHNOLOGY.2007.4290370.
- [26] K. Autonomi, "GPS Correction Comparisons: RTK versus DGPS," 2014, [Online]. Available: <http://www.kairosautonomi.com/uploads/files/129/Bulletin---RTK-vs-DGPS-010400.pdf>.
- [27] S. Hofmann, C. Kuntzsch, M. J. Schulze, D. Eggert, and M. Sester, "ACCURACY ANALYSIS of A LOW-COST PLATFORM for POSITIONING and NAVIGATION," *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.*, vol. 1, no. September, pp. 1–6, 2012, doi: 10.5194/isprsannals-I-2-1-

2012.

- [28] P. Goel, S. I. Roumeliotis, and G. S. Sukhatme, "Robust localization using relative and absolute position estimates," *IEEE Int. Conf. Intell. Robot. Syst.*, vol. 2, pp. 1134–1140, 1999, doi: 10.1109/iros.1999.812832.
- [29] C. R. Carlson, J. C. Gerdes, and J. D. Powell, "Error sources when land vehicle dead reckoning with differential wheelspeeds," *Navig. J. Inst. Navig.*, vol. 51, no. 1, pp. 13–27, 2004, doi: 10.1002/j.2161-4296.2004.tb00338.x.
- [30] J. Borenstein and L. Feng, "Measurement and Correction of Systematic Odometry Errors in Mobile Robots," *IEEE Trans. Robot. Autom.*, vol. 12, no. 6, pp. 845–857, 1996, doi: 10.1109/70.544768.
- [31] M. B. Alatis and G. P. Hancke, "A Review on Challenges of Autonomous Mobile Robot and Sensor Fusion Methods," *IEEE Access*, vol. 8, pp. 39830–39846, 2020, doi: 10.1109/ACCESS.2020.2975643.
- [32] N. El-Sheimy, E.-H. Shin, and X. Niu, "Kalman Filter Face-Off: Extended vs Unscented Kalman Filters for Integrated GPS and MEMS Inertial," *Insid. GNSS*, pp. 48–54, 2006.
- [33] D. Scaramuzza and F. Fraundorfer, "Visual Odometry [Tutorial]," no. December, 2011.
- [34] A. Howard, "Real-time stereo visual odometry for autonomous ground vehicles," *2008 IEEE/RSJ Int. Conf. Intell. Robot. Syst. IROS*, pp. 3946–3952, 2008, doi: 10.1109/IROS.2008.4651147.
- [35] M. O. A. Aqel, M. H. Marhaban, M. I. Saripan, and N. B. Ismail, "Review of visual odometry: types, approaches, challenges, and applications," *Springerplus*, vol. 5, no. 1, 2016, doi: 10.1186/s40064-016-3573-7.
- [36] "High Output machines - Network Rail." <https://www.networkrail.co.uk/running-the-railway/looking-after-the-railway/our-fleet-machines-and-vehicles/high-output/high-output->

machines/ (accessed Dec. 24, 2021).

- [37] O. Heirich, "Localization of Trains and Mapping of Railway Tracks," 2020.
- [38] T. Albrecht, K. Lüddecke, and J. Zimmermann, "A precise and reliable train positioning system and its use for automation of train operation," *IEEE ICIRT 2013 - Proc. IEEE Int. Conf. Intell. Rail Transp.*, pp. 134–139, 2013, doi: 10.1109/ICIRT.2013.6696282.
- [39] Frauscher, "Track Circuits vs . Axle Counters," 2020, [Online]. Available: https://www.frauscher.com/en/2020-02-Solution-Sheet-Track-Circuits-EN-US-SCREEN.pdf_dl_1901.
- [40] M. Lauer and D. Stein, "A Train Localization Algorithm for Train Protection Systems of the Future," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 970–979, Apr. 2015, doi: 10.1109/TITS.2014.2345498.
- [41] European Railway Agency, "Railway Safety Performance in the European Union 2008," *Office*, p. 44, 2008.
- [42] W. G. Temple, B. N. Anh Tran, B. Chen, Z. Kalbarczyk, and W. H. Sanders, "On Train Automatic Stop Control Using Balises: Attacks and a Software-Only Countermeasure," 2017, doi: 10.1109/PRDC.2017.52.
- [43] A. Mirabadi, N. Mort, and F. Schmid, "Application of sensor fusion to railway systems," *IEEE Int. Conf. Multisens. Fusion Integr. Intell. Syst.*, no. January, pp. 185–192, 1996, doi: 10.1109/mfi.1996.572176.
- [44] R. Sharma and R. M. Lourde, "Crosstalk reduction in balise and infill loops in automatic train control," *Proc. 2nd IEEE Int. Work. Adv. Sensors Interfaces, IWASI*, no. July 2007, 2007, doi: 10.1109/IWASI.2007.4420020.
- [45] J. Moreno García-Loygorri Antonio Pérez-Yuste César Briso, M. Berbineau Alain Pirovano, and J. Mendizábal, "Communication Technologies for Vehicles," 2018, Accessed: Dec. 23, 2021. [Online]. Available: <http://www.springer.com/series/7411>.
- [46] X. Zhan, Z. H. Mu, R. Kumar, and M. Shabaz, "Research on speed sensor

- fusion of urban rail transit train speed ranging based on deep learning,” *Nonlinear Eng.*, vol. 10, no. 1, pp. 363–373, Jan. 2021, doi: 10.1515/NLENG-2021-0028/MACHINEREADABLECITATION/RIS.
- [47] L. Bazant, A. Toma, and H. Mocek, “Description of methodology for data record sorting and saving Satellite Technology for Advanced Railway Signalling,” no. 687414, p. 114, 2017.
 - [48] A. Mirabadi, N. Mort, and F. Schmid, “Design of fault tolerant train navigation systems,” *Proc. Am. Control Conf.*, vol. 1, pp. 104–108, 1999, doi: 10.1109/ACC.1999.782749.
 - [49] P. Heide, V. Magori, and R. Schwarte, “Coded 24 GHz Doppler radar sensors: A new approach to high-precision vehicle position and ground-speed sensing in railway and automobile applications,” *IEEE NTC 1995 - Microw. Syst. Conf. Conf. Proc.*, pp. 101–104, 1995, doi: 10.1109/NTCMWS.1995.522870.
 - [50] M. Malvezzi, B. Allotta, and M. Rinchi, “Odometric estimation for automatic train protection and control systems,” *Veh. Syst. Dyn.*, vol. 49, no. 5, pp. 723–739, 2011, doi: 10.1080/00423111003721291.
 - [51] J. Zhou, H. Xiao, W. Jiang, W. Bai, and G. Liu, “Automatic subway tunnel displacement monitoring using robotic total station,” *Meas. J. Int. Meas. Confed.*, vol. 151, p. 107251, 2020, doi: 10.1016/j.measurement.2019.107251.
 - [52] J. Otegui, A. Bahillo, I. Lopetegi, and L. E. Diez, “A Survey of Train Positioning Solutions,” *IEEE Sens. J.*, vol. 17, no. 20, pp. 6788–6797, 2017, doi: 10.1109/JSEN.2017.2747137.
 - [53] O. Heirich, B. Siebler, S. Sand, A. Lehner, and O. G. Crespillo, “Measurement methods for train localization with onboard sensors,” *2020 Eur. Navig. Conf. ENC 2020*, 2020, doi: 10.23919/ENC48637.2020.9317435.
 - [54] J. Marais, J. Beugin, and M. Berbineau, “A Survey of GNSS-Based

- Research and Developments for the European Railway Signaling,” *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 10, pp. 2602–2618, Oct. 2017, doi: 10.1109/TITS.2017.2658179.
- [55] J. Wang, D. Liu, W. Jiang, and D. Lu, “Evaluation on loosely and tightly coupled GNSS/INS vehicle navigation system,” *Proc. 2017 19th Int. Conf. Electromagn. Adv. Appl. ICEAA 2017*, pp. 892–895, Oct. 2017, doi: 10.1109/ICEAA.2017.8065396.
- [56] K. Kim, S. Seol, and S.-H. Kong, “High-speed Train Navigation System based on Multi-sensor Data Fusion and Map Matching Algorithm,” *Int. J. Control. Autom. Syst.*, vol. 13, no. 3, pp. 503–512, 2015, doi: 10.1007/s12555-014-0251-9.
- [57] T. Zhang and X. Xu, “A new method of seamless land navigation for GPS/INS integrated system,” *Measurement*, vol. 45, no. 4, pp. 691–701, May 2012, doi: 10.1016/J.MEASUREMENT.2011.12.021.
- [58] F. G. Toro, U. Becker, D. E. D. Fuentes, D. Lu, and W. Tao, “Accuracy Analysis for GNSS-based Urban Land Vehicle Localisation System,” *IFAC-PapersOnLine*, vol. 49, no. 3, pp. 191–196, Jan. 2016, doi: 10.1016/J.IFACOL.2016.07.032.
- [59] B. Malakar and B. K. Roy, “Train localization using an adaptive multisensor data fusion technique,” *Transport*, vol. 34, no. 4, pp. 508–516, 2019, doi: 10.3846/transport.2019.11313.
- [60] B. Allotta, P. D’Adamio, M. Malvezzi, L. Pugi, A. Ridolfi, and G. Vettori, “A localization algorithm for railway vehicles,” *Conf. Rec. - IEEE Instrum. Meas. Technol. Conf.*, vol. 2015-July, pp. 681–686, 2015, doi: 10.1109/I2MTC.2015.7151350.
- [61] G. P. S. Carvalho and R. R. Costa, “Localization of an Autonomous Rail-Guided Robot Using Particle Filter,” *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 5642–5647, 2017, doi: 10.1016/j.ifacol.2017.08.1112.
- [62] S. Hensel, C. Hasberg, and C. Stiller, “Probabilistic rail vehicle localization

- with eddy current sensors in topological maps,” *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 4, pp. 1525–1536, 2011, doi: 10.1109/TITS.2011.2161291.
- [63] F. Boehringer, “Train location based on fusion satellite and train-borne sensor data,” *Locat. Serv. Navig. Technol.*, vol. 5084, no. August 2003, p. 76, 2003, doi: 10.1117/12.487062.
- [64] A. Acharya, S. Sadhu, and T. K. Ghoshal, “Train localization and parting detection using data fusion,” *Transp. Res. Part C Emerg. Technol.*, vol. 19, no. 1, pp. 75–84, 2011, doi: 10.1016/j.trc.2010.03.010.
- [65] R. Shenton, “Train positioning using video Odometry ESG100 Train positioning using video Odometry Problem presented by.”
- [66] M. Rahimi, H. Liu, I. D. Cardenas, A. Starr, A. Hall, and R. Anderson, “A Review on Technologies for Localisation and Navigation in Autonomous Railway Maintenance Systems,” *Sensors*, vol. 22, no. 11. p. 4185, 2022, doi: 10.3390/s22114185.
- [67] V. OKOROCHA and O. OLAJUGBA, “Comparative Analysis of Short, Medium and Long Baseline Processing in the Precision of GNSS Positioning.” Engaging the Challenges – Enhancing the Relevance, FIG Congress 2014. Kuala Lumpur, Malaysia. 16-21 June 2014.,” no. June, pp. 1–15, 2014.
- [68] N. Li *et al.*, “Indoor and outdoor low-cost seamless integrated navigation system based on the integration of INS/GNSS/LIDAR system,” *Remote Sens.*, vol. 12, no. 19, pp. 1–21, 2020, doi: 10.3390/rs12193271.
- [69] L. Chang, X. Niu, and T. Liu, “Gnss/imu/odo/lidar-slam integrated navigation system using imu/odo pre-integration,” *Sensors (Switzerland)*, vol. 20, no. 17, pp. 1–18, 2020, doi: 10.3390/s20174702.
- [70] S. Kuutti, S. Fallah, K. Katsaros, M. Dianati, F. McCullough, and A. Mouzakitis, “A Survey of the State-of-the-Art Localization Techniques and Their Potentials for Autonomous Vehicle Applications,” *IEEE Internet*

- Things J.*, vol. 5, no. 2, pp. 829–846, 2018, doi: 10.1109/JIOT.2018.2812300.
- [71] C. C. Semester, “COS 495 - Lecture 9 Autonomous Robot Navigation,” pp. 1–25, 2011.
- [72] A. Woo, B. Fidan, and W. W. Melek, “Localization for Autonomous Driving,” *Handb. Position Locat.*, pp. 1051–1087, 2019, doi: 10.1002/9781119434610.ch29.
- [73] M. E. Qazizada and E. Pivarčiová, “Mobile robot controlling possibilities of inertial navigation system,” *Procedia Eng.*, vol. 149, no. June, pp. 404–413, 2016, doi: 10.1016/j.proeng.2016.06.685.
- [74] E. Bertran and J. A. Delgado-Penín, “On the use of GPS receivers in railway environments,” *IEEE Trans. Veh. Technol.*, vol. 53, no. 5, pp. 1452–1460, Sep. 2004, doi: 10.1109/TVT.2004.832416.
- [75] “Difference Between RTK and DGPS - RTK Base | Inertial Sense [Video].” <https://inertialsense.com/difference-between-rtk-and-dgps/> (accessed Apr. 18, 2023).
- [76] K. W. Hall, P. Gagliardi, and D. C. Lawton, “GPS accuracy part 2: RTK float versus RTK fixed,” *CREWES Res. Rep.*, vol. 22, no. 2, p. 1, 2010.
- [77] F. van Diggelen, “GPS and GPS+GLONASS RTK,” *Proc. ION GPS*, vol. 1, no. September, pp. 139–144, 1997.
- [78] B. Schwarz, “INDUSTRY PERSPECTIVE TECHNOLOGY FOCUS LIDAR Mapping the world in 3D,” 2010. doi: 10.1038/nphoton.2010.148.
- [79] “How LiDAR Detection Works. Exploring how LiDAR technology is... | by Jeremy Cohen | Heartbeat.” <https://heartbeat.fritz.ai/how-lidar-detection-works-8db24556a8d9> (accessed Sep. 09, 2021).
- [80] D. Stein, M. Lauer, and M. Spindler, “An analysis of different sensors for turnout detection for train-borne localization systems,” 2014, doi: 10.2495/CR140691.

- [81] “Blog - LiDAR Sensing Applications in the Railroad Industry - LeddarTech.” <https://leddartech.com/blog-lidar-sensing-applications-in-the-railroad-industry/> (accessed Mar. 30, 2022).
- [82] D. Stein, M. Spindler, J. Kuper, and M. Lauer, “Rail detection using lidar sensors,” *Int. J. Sustain. Dev. Plan.*, vol. 11, no. 1, pp. 65–78, 2016, doi: 10.2495/SDP-V11-N1-65-78.
- [83] S. Gu, T. Lu, Y. Zhang, J. M. Alvarez, J. Yang, and H. Kong, “3-D LiDAR + Monocular Camera: An inverse-depth-induced fusion framework for urban road detection,” *IEEE Trans. Intell. Veh.*, vol. 3, no. 3, pp. 351–360, 2018, doi: 10.1109/TIV.2018.2843170.
- [84] N. Kemsaram, A. Das, and G. Dubbelman, “A Stereo Perception Framework for Autonomous Vehicles,” *IEEE Veh. Technol. Conf.*, vol. 2020-May, 2020, doi: 10.1109/VTC2020-Spring48590.2020.9128899.
- [85] C. Häne, T. Sattler, and M. Pollefeys, “Obstacle detection for self-driving cars using only monocular cameras and wheel odometry,” *IEEE Int. Conf. Intell. Robot. Syst.*, vol. 2015-Decem, pp. 5101–5108, 2015, doi: 10.1109/IROS.2015.7354095.
- [86] S. Liu, Q. Wang, and Y. Luo, “A review of applications of visual inspection technology based on image processing in the railway industry,” *Transp. Saf. Environ.*, vol. 1, no. 3, pp. 185–204, 2019, doi: 10.1093/tse/tdz007.
- [87] A. Berg, K. Öfjäll, J. Ahlberg, and M. Felsberg, “Detecting Rails and Obstacles Using a Train-Mounted Thermal Camera,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9127, pp. 492–503, 2015, doi: 10.1007/978-3-319-19665-7_42.
- [88] T. Ye, Z. Zhang, X. Zhang, and F. Zhou, “Autonomous Railway Traffic Object Detection Using Feature-Enhanced Single-Shot Detector,” *IEEE Access*, vol. 8, pp. 145182–145193, 2020, doi: 10.1109/ACCESS.2020.3015251.
- [89] J. Fayyad, M. A. Jaradat, D. Gruyer, and H. Najjaran, “Deep learning

- sensor fusion for autonomous vehicle perception and localization: A review,” *Sensors (Switzerland)*, vol. 20, no. 15, pp. 1–34, 2020, doi: 10.3390/s20154220.
- [90] J. Carlson and C. Thorpe, “Mapping Large, Urban Environments with GPS-Aided SLAM Thesis Committee,” 2010.
- [91] R. F. Brena, A. A. Aguilera, L. A. Trejo, E. Molino-Minero-Re, and O. Mayora, “Choosing the best sensor fusion method: A machine-learning approach,” *Sensors (Switzerland)*, vol. 20, no. 8, Apr. 2020, doi: 10.3390/S20082350.
- [92] D. J. Yeong, G. Velasco-hernandez, J. Barry, and J. Walsh, “Sensor and sensor fusion technology in autonomous vehicles: A review,” *Sensors*, vol. 21, no. 6, pp. 1–37, 2021, doi: 10.3390/s21062140.
- [93] J. Kim, J. Kim, and J. Cho, “An advanced object classification strategy using YOLO through camera and LiDAR sensor fusion; An advanced object classification strategy using YOLO through camera and LiDAR sensor fusion,” 2019.
- [94] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition.” [Online]. Available: <http://image-net.org/challenges/LSVRC/2015/>.
- [95] W. Liu *et al.*, “SSD: Single Shot MultiBox Detector,” doi: 10.1007/978-3-319-46448-0.
- [96] X. Zhou, D. Wang, and P. Krähenbühl, “Objects as Points.” [Online]. Available: <https://github>.
- [97] T. Moore and D. Stouch, “A generalized extended Kalman filter implementation for the robot operating system,” *Adv. Intell. Syst. Comput.*, vol. 302, pp. 335–348, 2016, doi: 10.1007/978-3-319-08338-4_25.
- [98] S. Yousuf and M. B. Kadri, “Robot localization in indoor and outdoor environments by multi-sensor fusion,” *2018 14th Int. Conf. Emerg. Technol.*

ICET 2018, Jan. 2019, doi: 10.1109/ICET.2018.8603597.

- [99] H. Kim and I. Lee, "Localization of a car based on multi-sensor fusion," *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci. - ISPRS Arch.*, vol. 42, no. 1, pp. 247–250, 2018, doi: 10.5194/isprs-archives-XLII-1-247-2018.
- [100] A. Giannitrapani, N. Ceccarelli, F. Scortecci, and A. Garulli, "Comparison of EKF and UKF for spacecraft localization via angle measurements," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 47, no. 1, pp. 75–84, Jan. 2011, doi: 10.1109/TAES.2011.5705660.
- [101] V. I. George and B. K. Roy, "Lecture Notes in Electrical Engineering 660 Advances in Control Instrumentation Systems," Accessed: Jan. 08, 2022. [Online]. Available: <http://www.springer.com/series/7818>.
- [102] F. Liu, X. Li, S. Yuan, and W. Lan, "Slip-Aware Motion Estimation for Off-Road Mobile Robots via Multi-Innovation Unscented Kalman Filter," *IEEE Access*, vol. 8, pp. 43482–43496, 2020, doi: 10.1109/ACCESS.2020.2977889.
- [103] B. Khaleghi, A. Khamis, F. O. Karray, and S. N. Razavi, "Multisensor data fusion: A review of the state-of-the-art," *Inf. Fusion*, vol. 14, no. 1, pp. 28–44, Jan. 2013, doi: 10.1016/J.INFFUS.2011.08.001.
- [104] L. Kong, X. Peng, Y. Chen, P. Wang, and M. Xu, "Multi-sensor measurement and data fusion technology for manufacturing process monitoring: A literature review," *Int. J. Extrem. Manuf.*, vol. 2, no. 2, 2020, doi: 10.1088/2631-7990/ab7ae6.
- [105] B. Allotta et al., "An innovative localisation algorithm for railway vehicles," *Vehicle System Dynamics*, vol. 52, no. 11, pp. 1443–1469, 2014, doi: 10.1080/00423114.2014.943928.
- [106] M. Malvezzi, G. Vettori, B. Allotta, L. Pugi, A. Ridolfi, and A. Rindi, "A localization algorithm for railway vehicles based on sensor fusion between tachometers and inertial measurement units," *Proc. Inst. Mech. Eng. Part F J. Rail Rapid Transit*, vol. 228, no. 4, pp. 431–448, 2014, doi:

10.1177/0954409713481769.

- [107] F. Zhang *et al.*, “A sensor fusion approach for localization with cumulative error elimination,” *IEEE Int. Conf. Multisens. Fusion Integr. Intell. Syst.*, pp. 1–6, 2012, doi: 10.1109/MFI.2012.6343009.
- [108] W. Ochieng *et al.*, “Integration of GPS and dead reckoning for real-time vehicle performance and emissions monitoring,” *GPS Solut.*, vol. 6, no. 4, pp. 229–241, 2003, doi: 10.1007/s10291-002-0035-1.
- [109] J. Levinson, M. Montemerlo, and S. Thrun, “Map-based precision vehicle localization in urban environments,” *Robot. Sci. Syst.*, vol. 3, pp. 121–128, 2008, doi: 10.15607/rss.2007.iii.016.
- [110] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, “A Survey of Autonomous Driving: Common Practices and Emerging Technologies,” *IEEE Access*, vol. 8, pp. 58443–58469, 2020, doi: 10.1109/ACCESS.2020.2983149.
- [111] N. Akai, L. Y. Morales, E. Takeuchi, Y. Yoshihara, and Y. Ninomiya, “Robust localization using 3D NDT scan matching with experimentally determined uncertainty and road marker matching,” *IEEE Intell. Veh. Symp. Proc.*, no. February 2018, pp. 1356–1363, 2017, doi: 10.1109/IVS.2017.7995900.
- [112] Z. Zhang, M. Liu, C. Diao, and S. Chen, “A Survey of the Simultaneous Localization and Mapping (Slam) Based on Rgb-D Camera,” no. Meet, pp. 48–58, 2019.
- [113] M. E. Jefferies and W. K. Yeap, *Robot and cognitive approaches to spatial mapping*, vol. 38. 2008.
- [114] H. Taheri and Z. C. Xia, “SLAM; definition and evolution,” *Eng. Appl. Artif. Intell.*, vol. 97, no. April 2020, p. 104032, 2021, doi: 10.1016/j.engappai.2020.104032.
- [115] C. Stachniss, J. J. Leonard, and S. Thrun, “Simultaneous Localization and

Mapping,” pp. 1153–1175.

- [116] L. Ferreira, “Localization and navigation in an autonomous vehicle,” 2013, [Online]. Available: <http://ria.ua.pt/handle/10773/11672>.
- [117] T. T. O. Takleh, N. A. Bakar, S. A. Rahman, R. Hamzah, and Z. A. Aziz, “A brief survey on SLAM methods in autonomous vehicle,” *Int. J. Eng. Technol.*, vol. 7, no. 4, pp. 38–43, 2018, doi: 10.14419/ijet.v7i4.27.22477.
- [118] C. Cadena *et al.*, “Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age,” *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, 2016, doi: 10.1109/TRO.2016.2624754.
- [119] A. Monjazez, J. Z. Sasiadek, and D. Neculescu, “Autonomous navigation among large number of nearby landmarks using FastSLAM and EKF-SLAM - A comparative study,” *2011 16th Int. Conf. Methods Model. Autom. Robot. MMAR 2011*, pp. 369–374, 2011, doi: 10.1109/MMAR.2011.6031375.
- [120] I. Deutsch, M. Liu, and R. Siegwart, “A framework for multi-robot pose graph SLAM,” *2016 IEEE Int. Conf. Real-Time Comput. Robot. RCAR 2016*, pp. 567–572, 2016, doi: 10.1109/RCAR.2016.7784092.
- [121] D. Wilbers, C. Merfels, and C. Stachniss, “A Comparison of Particle Filter and Graph-Based Optimization for Localization with Landmarks in Automated Vehicles,” *Proc. - 3rd IEEE Int. Conf. Robot. Comput. IRC 2019*, pp. 220–225, 2019, doi: 10.1109/IRC.2019.00040.
- [122] M. Hamadache, S. Dutta, O. Olaby, R. Ambur, E. Stewart, and R. Dixon, “On the fault detection and diagnosis of railway switch and crossing systems: An overview,” *Appl. Sci.*, vol. 9, no. 23, 2019, doi: 10.3390/app9235129.
- [123] G. Lederman, S. Chen, J. H. Garrett, J. Kovačević, H. Y. Noh, and J. Bielak, “A data fusion approach for track monitoring from multiple in-service trains,” *Mech. Syst. Signal Process.*, vol. 95, pp. 363–379, Oct. 2017, doi: 10.1016/J.YMSSP.2017.03.023.

- [124] M. Yonekawa and T. Tanaka, "Relations between positioning result and each error factor in GPS," *2006 SICE-ICASE Int. Jt. Conf.*, pp. 1361–1365, 2006, doi: 10.1109/SICE.2006.315574.
- [125] R. K. W. Vithanage, C. S. Harrison, and A. K. M. Desilva, "Importance and applications of robotic and autonomous systems (RAS) in railway maintenance sector: A review," *Computers*, vol. 8, no. 3, 2019, doi: 10.3390/computers8030056.
- [126] E. Menendez, J. G. Victores, R. Montero, S. Martínez, and C. Balaguer, "Tunnel structural inspection and assessment using an autonomous robotic system," *Autom. Constr.*, vol. 87, pp. 117–126, Mar. 2018, doi: 10.1016/J.AUTCON.2017.12.001.
- [127] A. Kitamura, T. Namekawa, K. Hiramatsu, and Y. Sankai, "Operating Manipulator Arm by Robot Suit HAL for Remote In-Cell Equipment Maintenance," <http://dx.doi.org/10.13182/NT13-A24988>, vol. 184, no. 3, pp. 310–319, 2017, doi: 10.13182/NT13-A24988.
- [128] "Warthog Unmanned Ground Vehicle Robot - Clearpath." <https://clearpathrobotics.com/warthog-unmanned-ground-vehicle-robot/> (accessed Sep. 14, 2022).
- [129] H. Liu *et al.*, "An autonomous rail-road amphibious robotic system for railway maintenance using sensor fusion and mobile manipulator," *Comput. Electr. Eng.*, vol. 110, no. August, p. 108874, 2023, doi: 10.1016/j.compeleceng.2023.108874.
- [130] S. Smart-I, "Dual-Frequency GNSS Smart Antenna Featuring NovAtel'S Powerful OEM6 Technology," pp. 6–7, 2016, [Online]. Available: <http://www.novatel.com>.
- [131] "NovAtel Application Suite | NovAtel." <https://novatel.com/products/firmware-options-pc-software/novatel-application-suite> (accessed Apr. 26, 2023).
- [132] "Inertial Explorer® | NovAtel." <https://novatel.com/products/waypoint-post->

processing-software/inertial-explorer (accessed Apr. 26, 2023).

- [133] M. Rahimi *et al.*, “Localisation and Navigation Framework for autonomous Railway Robotic Inspection and Repair System,” no. November, 2021.
- [134] B. K. Patle, G. Babu L, A. Pandey, D. R. K. Parhi, and A. Jagadeesh, “A review: On path planning strategies for navigation of mobile robot,” *Def. Technol.*, vol. 15, no. 4, pp. 582–606, 2019, doi: 10.1016/j.dt.2019.04.011.
- [135] “global_planner - ROS Wiki.” http://wiki.ros.org/global_planner (accessed Apr. 20, 2023).
- [136] “Introduction to A*.” <http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html> (accessed Apr. 20, 2023).
- [137] H. Y. Zhang, W. M. Lin, and A. X. Chen, “Path planning for the mobile robot: A review,” *Symmetry (Basel)*, vol. 10, no. 10, 2018, doi: 10.3390/sym10100450.
- [138] “base_local_planner - ROS Wiki.” http://wiki.ros.org/base_local_planner (accessed Apr. 20, 2023).
- [139] P. Marin-Plaza, A. Hussein, D. Martin, and A. De La Escalera, “Research Article Global and Local Path Planning Study in a ROS-Based Research,” 2018, doi: 10.1155/2018/6392697.
- [140] “dwa_local_planner - ROS Wiki.” http://wiki.ros.org/dwa_local_planner (accessed Apr. 20, 2023).
- [141] D. Kiss-Illés, C. Barrado, and E. Salamí, “Gps-slam: An augmentation of the orb-slam algorithm,” *Sensors (Switzerland)*, vol. 19, no. 22, pp. 1–22, 2019, doi: 10.3390/s19224973.
- [142] D. Schleicher, L. M. Bergasa, M. Ocaña, R. Barea, and E. López, “Real-time hierarchical GPS aided visual SLAM on urban environments,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5717 LNCS, pp. 326–333, 2009, doi: 10.1007/978-3-

642-04772-5_43.

- [143] “robot_localization wiki — robot_localization 2.6.11 documentation.” http://docs.ros.org/en/melodic/api/robot_localization/html/index.html (accessed Sep. 09, 2021).
- [144] M. Rahimi, M. Rahman, I. Durazo Cardenas, A. Starr, A. Hall, and R. Anderson, “On-track Localisation based on Object Detection,” *UKRAS22 Conf. "Robotics Unconstrained Environ. Proc.*, vol. 5, pp. 42–43, 2022, doi: 10.31256/br7bk8o.
- [145] M. Labbé and F. Michaud, “RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation,” *J. F. Robot.*, vol. 36, no. 2, pp. 416–446, 2019, doi: 10.1002/rob.21831.
- [146] K. Zheng September, “ROS Navigation Tuning Guide.”
- [147] V. Tadic *et al.*, “Perspectives of RealSense and ZED Depth Sensors for Robotic Vision Applications,” *Mach. 2022, Vol. 10, Page 183*, vol. 10, no. 3, p. 183, Mar. 2022, doi: 10.3390/MACHINES10030183.

APPENDICES

Appendix A Important Python Scripts

In the following some codes are provided regarding the on-track waypoint publisher, GPS goal publisher, and autonomous navigation based on GPS odometry.

A.1 Code for on-track waypoint publisher

```
#!/usr/bin/env python
import rospy
import roslib
from std_msgs.msg import Float64
from sensor_msgs.msg import NavSatFix
from geographic_msgs.msg import WayPoint
from geographic_msgs.msg import KeyValue

class GPS_ontrack_waypoint_generate():
    def __init__(self):
        self.fixwaypoint = WayPoint()
        self.gps_signal = NavSatFix()
        rospy.loginfo("Initiating GPS 2D Nav Goal Node.")

        self.lat_fencing = Float64()
        self.lon_fencing = Float64()
        rospy.set_param
        ('gnss_nav_error_params/lat_fencing',[50.0656033333, 55.0676666667])
        rospy.set_param ('gnss_nav_error_params/lon_fencing',[-
0.6053433333, -0.6574933333])
        self.lat_fencing.data =
rospy.get_param('gnss_nav_error_params/lat_fencing')
        self.lon_fencing.data =
rospy.get_param('gnss_nav_error_params/lon_fencing')

        rospy.Subscriber("/gps/fix", NavSatFix, self.gpslistening)
        self.waypoint_pub = rospy.Publisher('/waypoint', WayPoint,
queue_size = 20)

    def gpslistening(self, NavSatFix):
        rospy.set_param('/job_selected/latitude', 52.0666366667)
        rospy.set_param('/job_selected/longitude', -0.6264)
        a = rospy.get_param('/job_selected/latitude')
        b = rospy.get_param('/job_selected/longitude')
        lat_fencing = Float64()
        lon_fencing = Float64()
```

```

lat_fencing.data = self.lat_fencing.data
lon_fencing.data = self.lon_fencing.data
self.fixwaypoint.position.latitude = a
self.fixwaypoint.position.longitude = b
self.fixwaypoint.position.altitude = 0
self.waypoint_pub.publish(self.fixwaypoint)
def loop(self):
    rospy.spin()

```

A.2 Code for GPS goal publisher

```

#!/usr/bin/env python
import roslib
import rospy
from math import radians, cos, sin, asin, sqrt, pow, pi, atan2
import numpy as np
from std_msgs.msg import String
from std_msgs.msg import Float64
from std_msgs.msg import Int8
from sensor_msgs.msg import NavSatFix
from geometry_msgs.msg import PoseStamped
from geometry_msgs.msg import PoseWithCovarianceStamped
from nav_msgs.msg import Odometry
from geometry_msgs.msg import Vector3
from geographic_msgs.msg import WayPoint
from ontrack_waypoint_publisher import GPS_ontrack_waypoint_generate

class GPS_based_ontrack_SimpleGoal():
    def __init__(self):
        self.debug = False
        self.gpsstatus = Int8()
        self.latCur = 0.0
        self.lonCur = 0.0
        self.latWP = 0.0
        self.lonWP = 0.0
        self.altWP = 0.0
        self.error_level = 0.0

        self.earthRadius = 6371000.0 #Metres
        self.currPosX = 0.0
        self.currPosY = 0.0
        self.currPosZ = 0.0

        self.WPUpdateState = False
        self.lastValidFixTime = 0.0
        self.gpsValidityTimeout = 10.0 #Seconds
        GPS_ontrack_waypoint_generate()
        rospy.Subscriber("/gps/fix", NavSatFix, self.gpsSubscriber)

```

```

        rospy.Subscriber("/waypoint", WayPoint, self.waypointSubscriber)
#Subscribe to "pose", "fix" and "waypoint" ROS topics
        self.gpsnavgoal_ontrack = rospy.Publisher('ontrack_simple/goal',
PoseStamped, queue_size=20)
        self.onedimensional_distance = rospy.Publisher('on_track/goal',
Vector3, queue_size=20)

    def haversineDistance(self, latCur, lonCur, latWP, lonWP): #Returns
distance to waypoint in Metres
        latWP1, lonWP1, latCur1, lonCur1 = map (radians, [latWP, lonWP,
latCur, lonCur]) #Convert into Radians to perform math
        a = pow (sin ((latWP1 - latCur1)/2),2) + cos(latCur1) *
cos(latWP1) * pow (sin((lonWP1 - lonCur1)/2),2)
        return self.earthRadius * 2.0 * asin(sqrt(a)) #Return calculated
distance to waypoint in Metres

    def bearing (self, latCur, lonCur, latWP, lonWP): #Bearing to
waypoint (degrees)
        latWP1, lonWP1, latCur1, lonCur1 = map (radians, [latWP, lonWP,
latCur, lonCur]) #Convert into Radians to perform math
        dLon = lonWP1 - lonCur1
        return atan2(sin(dLon) * cos(latWP1), cos(latCur1) * sin(latWP1)
- (sin(latCur1) * cos(latWP1) * cos(dLon)))

    def gpsSubscriber(self, gpsMsg): #GPS Coordinate recieved from ROS
topic, run this function
        gps_covar = Float64()
        if gpsMsg.status.status > -1: #If there is a GPS fix (Either
Augmented or Unaugmented)
            self.lastValidFixTime = rospy.get_time()
            self.gpsstatus = gpsMsg.status.status
            self.latCur = gpsMsg.latitude
            self.lonCur = gpsMsg.longitude
            gps_covar = gpsMsg.position_covariance
            aa=sqrt(gps_covar[0])
            bb=sqrt(gps_covar[4])
            self.error_level= sqrt(aa*aa+bb*bb)
            if self.debug == True:
                rospy.loginfo("GPS Fix Available, Latitude: %f,
Longitude: %f", self.latCur, self.lonCur)

    def gpsFixIsValid(self): #Check to see if there has been a GPS fix
within the last <gpsValidityTimeout> seconds
        if (rospy.get_time()- self.lastValidFixTime) <
self.gpsValidityTimeout:
            return True
        else:

```

```

        rospy.loginfo("GPS Fix Invalid! Last valid update was: %f
seconds ago", rospy.get_time()- self.lastValidFixTime)
        return False

    def waypointSubscriber(self,WPMsg): #Waypoint Command recieved from
ROS topic, run this function
        data_waypoint = WPMsg
        self.latWP = data_waypoint.position.latitude
        self.lonWP = data_waypoint.position.longitude
        self.altWP = data_waypoint.position.altitude
        if self.gpsFixIsValid() == True: #If there is a valid GPS fix,
publish nav goal to ROS
            self.posePublisher()

    def posePublisher(self): #Convert absolute waypoint to vector
relative to robot, then publish navigation goal to ROS
        simpletrackdistance = Vector3()
        desiredPose = PoseStamped()
        desiredPose.header.frame_id = "/gps_link"
        desiredPose.header.stamp = rospy.Time.now()
        distToWP=0
        bearingToWP=0
        if self.debug:
            rospy.loginfo("LatWP: %f, LonWP: %f, LatCur: %f, LonCur: %f",
self.latWP, self.lonWP, self.latCur, self.lonCur)
            distToWP = self.haversineDistance(self.latCur, self.lonCur,
self.latWP, self.lonWP)
            bearingToWP = self.bearing(self.latCur, self.lonCur, self.latWP,
self.lonWP)
            rospy.loginfo ("GPS bearing: %f", bearingToWP)

        rospy.loginfo("distToWP:, %f, %f", distToWP, bearingToWP)
        desiredPose.pose.position.x = self.currPosX + (distToWP *
cos(bearingToWP)) #Convert distance and angle to waypoint from Polar to
Cartesian co-ordinates then add current position of robot odometry
        desiredPose.pose.position.y = self.currPosY + (distToWP *
sin(bearingToWP))
        desiredPose.pose.position.z = self.altWP - self.currPosZ
#Assuming CurrPosZ is absolute (eg barometer or GPS)
        desiredPose.pose.orientation.x = 0
        desiredPose.pose.orientation.y = 0
        desiredPose.pose.orientation.z = 0
        desiredPose.pose.orientation.w = 1

        simpletrackdistance.x = distToWP
        simpletrackdistance.y = (bearingToWP * 180/pi + 360) % 360
        simpletrackdistance.z = self.error_level

```

```

        self.gpsnavgoal_ontrack.publish(desiredPose)
        self.onedimensional_distance.publish(simpletrackdistance)
        rospy.loginfo("GPS Valid! Setting Goal to: %f m ",
simpletrackdistance.x)
        rospy.loginfo("GPS Fix is Valid! Setting Navigation Goal
to: %f, %f, %f", desiredPose.pose.position.x,
desiredPose.pose.position.y, desiredPose.pose.position.z)
        rospy.loginfo("Robot is heading %f metres at a bearing of %f
degrees", distToWP, (bearingToWP * 180/pi + 360) % 360)
    def loop(self):
        rospy.spin()

if __name__ == '__main__':
    rospy.init_node('gps_ontrack_goal', anonymous=False)
    ontrack_gps_goal = GPS_based_ontrack_SimpleGoal()

    ontrack_gps_goal.loop()

```

A.3 Code for navigation based on GPS

```

#!/usr/bin/env python
import pynput
import roslib
import rospy
import math
import os
import sys
import time
import numpy as np
from pynput import keyboard
from geometry_msgs.msg import Twist
from geometry_msgs.msg import Vector3
from std_msgs.msg import Float64
from std_msgs.msg import Int8
from RITA_GPS_ontrack_speed_Generate1 import GPS_nav_ontrack_speed

class GPS_based_ontrack_run():
    def __init__(self):
        self.target_distance = 20.0 #input ("Type objective distance:")
        self.target_x_speed = 0.6 #input ("Type speed:")
        self.x_speed_changerate = 1.0 #input ("Type Acceleration rate:")
        self.cmd_decision_status = 0
        self.aaaa = True
        if (self.aaaa == True):
            self.isForward = True
        else:
            self.isForward = False
        if (self.isForward ==True):

```

```

        self.inverse = -1
    else:
        self.inverse = 1
    GPS_nav_ontrack_speed(self.target_x_speed,
self.x_speed_changerate, self.cmd_decision_status)
    self.FB_obs_decision = Vector3()
    self.aux = Twist ()
    self.speed_pub = rospy.Publisher('/cmd_vel', Twist, queue_size =
1)
    rospy.Subscriber('/warthog_cmd_vel', Twist,
self.ontrack_inverse_speed, queue_size = 10)
    print ("RIRS running!!!! ")
    self.rate = rospy.Rate(30)
    self.FB_obs_decision.x = 1.0
    while not rospy.is_shutdown():
        if self.isForward == True:
            if self.FB_obs_decision.x == 1.0:
                self.speed_pub.publish(self.aux)
                print ("No Front obstacle detected:",
self.FB_obs_decision.x)
            else:
                self.aux.linear.x = 0
                self.aux.linear.y = 0
                self.aux.linear.z = 0
                self.aux.angular.x = 0
                self.aux.angular.y = 0
                self.aux.angular.z = 0
                self.speed_pub.publish(self.aux)
                print ('Front obstacle detected, please clean front
path!!!!')
            self.rate.sleep()
        else:
            if self.FB_obs_decision.y == 1.0:
                self.speed_pub.publish(self.aux)
            else:
                self.aux.linear.x = 0
                self.aux.linear.y = 0
                self.aux.linear.z = 0
                self.aux.angular.x = 0
                self.aux.angular.y = 0
                self.aux.angular.z = 0
                self.speed_pub.publish(self.aux)
                print ('Back obstacle detected, please clean rear
path!!!!')
            self.rate.sleep()

def ontrack_inverse_speed(self, msg):

```

```

        self.aux.linear.x = self.inverse * msg.linear.x
        self.aux.linear.y = msg.linear.y
        self.aux.linear.z = msg.linear.z
        self.aux.angular.x = msg.angular.x
        self.aux.angular.y = msg.angular.x
        self.aux.angular.z = msg.angular.x
    def loop(self):
        rospy.spin()

if __name__ == '__main__':
    rospy.init_node('nav_job_location', anonymous=False)
    ontrack_GPS_nav = GPS_based_ontrack_run()
    ontrack_GPS_nav.loop()

    print ("good?")

```