Navigation and Control for an Autonomous Robotic Fruit Harvesting System

Leo Geer

A thesis submitted for the degree of

Doctor of Philosophy

School of Computer Science and Electronic Engineering

University of Essex

April 2023

Abstract

The use of robotics in industrial environments has increased dramatically over the past decade. Advances in automated systems have given rise to a multitude of applications, from automated robotic security to drone delivery services. With research and development motivated by both economic incentives and accentuated by national and global issues, such as labour shortage, climate change and continuously increasing demand, the agricultural industry has recently found interest in automated robotic technologies. Robotics are an ideal solution to crop harvesting. This often repetitive, time consuming, labour intensive and costly task is perfectly suited to automation with a capable system. Automated robotics would enable 24 hour operation, increasing efficiency and improving resource management, whilst significantly reducing many associated costs. This task from a human's perspective is straightforward, yet when thought of in a robotic system's context requires many interlinked components including navigation and control methods and computer vision systems. The research of this thesis concentrates on the practical implementation of a navigation and control software system for a strawberry picking task. It includes a novel software architecture, a navigation and control method and initial work into a low cost implementation of an automated robotic fruit harvesting system. The software architecture is a configurable, built on top of the Robot Operating System (ROS). It contains perception and action systems to coordinate dual arms with a mobile base platform to pick strawberries. This architecture is able to host laser SLAM for building farm maps for autonomous navigation. The design and implementation of the software architecture, and the evaluation of the laser SLAM mapping building are included. The autonomous navigation and control software for the system incorporates the provision for both robotic arms and mobile base. The motion strategy of the mobile base and dual arms for picking strawberries is proposed, and the analysis of its performance in both a laboratory setting and an agricultural environment is included. Finally the thesis contributes the implementation of a single board computer for the software archi-

tecture. The focus of this portable implementation is the performance analysis of instance segmentation and SLAM systems based on an RGBD camera. The experimentation to determine the viability of implementing the system overall on to a single board computer is also introduced.

Acknowledgements

First and foremost i would like to express my gratitude and sincerest thanks to my supervisor Professor Dongbing Gu for his invaluable encouragement, guidance and support throughout my academic studies.

I'd like to thank Dr Vishwanathan Mohan and Robin Dowling for their encouragement, assistance and support throughout my PhD studies and for which I am extremely grateful for. I would also like to thank the University of Essex, Innovate UK, Tiptree Farm, Chris Newenham and Andrey Ivanov for the opportunity to have worked on the strawberry harvesting project.

Finally i would like to thank my family and friends for their continued support throughout my academic studies, with a special thanks to Nicola Wood, Hannah Hargreaves, Anca Ristin, Natasha Elliott, Martina Di Michele, Ben Allen, Jordan Hayward, Tom Hobbs, Sean Salter, Luke Eccles and Charley Wynn for their encouragement, support, the many discussions and generally keeping me sane throughout the last 4 years.

Contents

List of Figures

Notations

Yr	Y Robot
Yc	Y Camera
Xr	X Robot
Xc	X Camera
Zr	Z Robot
Zc	Z Camera
$Yreq$	Y Required
$Ycur$	Y Current
$Ypos$	Y Position

Abbreviations

SLAM	Simultaneous Localisation and Mapping
VSLAM	Visual Simultaneous Localisation and Mapping
VO	Visual Odometry
UGV	Unmanned Ground Vehicle
LiDAR	Light Detection and Ranging
SBC	Single Board Computer
CNN	Convolutional Neural Network
RGB	Image Colour Channels
RGB-D	RGB-Depth
IMU	Inertial Measurement Unit
FPS	Frames Per Second
mAP	Mean Average Precision
GPU	Graphics Processing Unit
GPS	Global Positioning System
RTABMAP	Real-Time Appearance Based Mapping
ROS	Robot Operating System
YOLACT	You Only Look At the Coefficients

ROI	Region Of Interest
RPN	Region Proposal Network
IoU	Intersection Over Union
MSG	Multi Scale Grouping
MRG	Multi Region Grouping
EKF	Extended Kalman Filter
RPE	Relative Pose Error
RMSE	Root Mean Squared Error
MAE	Mean Absolute Error
MSE	Mean Squared Error

Chapter 1

Introduction

Autonomous robotic systems, first invented over 70 years ago, have improved drastically since their first introduction. With continuously improving technology, affordability and decades of research and development, these systems are becoming increasingly capable, applicable and perhaps most importantly, useful, across many areas of society and in virtually all industries. Robotic systems with autonomous capabilities are being employed to assist or indeed entirely replace humans with both repetitive and more specialised tasks, providing an increase in productivity, efficiency whilst operating safety in almost any environment. Autonomous robots can be found in a wide range of applications, factory assembly lines and warehouse floor automation, exploration terrestrially in difficult to reach environments such as remote locations or underwater and extra-terrestrially and hazardous and dangerous environments with disaster recovery and security.

An interesting, difficult application of these systems, yet increasingly commonplace is in commercial settings with drone delivery systems and in industrial environments, these not necessarily more difficult for the robotic system, but requiring extremely strict safety con-

straints. One such environment, requiring complex navigation and control, operating in difficult, dynamic and unpredictable terrain and conditions is the agricultural sector, the focus of this research.

Agricultural robotic technology has seen a dramatic rise in interest over the last few decades with increasing capabilities through innovative research and development, a reduction in cost of component and sensor technology and robotic system resilience, enabling these systems to withstand outdoor, rigorous environments and varied climates. These technological and methodological improvements all contribute to agri-techs continued improvement.

These systems are required to operate in often complex, dynamic and technologically inhospitable environments, whilst ensuring the safety of any humans that may be in the area and whilst ensuring they are capable of matching or exceeding a human's performance in their intended application.

Agricultural robotics have been in use again for decades, with automated tractors being of particular note, using GPS to navigate around farm land. With improving technological capabilities, new and novel uses are being realised, monitoring the health of crops, pest control and crop planting. Requiring precise movement and functionality of a system are becoming increasingly viable alternatives, reducing human labour requirements whilst improving the efficiency of these tasks.

A considerably difficult agri-tech application is in the harvesting of plants such as fruits, whether at ground level or above. This task requires many systems to operate in parallel. For a ground based target, this could be to detect, move over and harvest the plant a difficult task in itself. This becomes further challenging with fruit above ground, such as strawberries, requiring a method of detecting a given fruit, moving to it, picking the fruit without damaging

it, often in occluded or articulate places.

This task presents an interesting problem, is an autonomous fruit harvesting system possible in this type of environment and would it be capable of harvesting efficiently whilst matching or exceeding the productivity of a human, being cost effective and in a safe manner. These systems, although reducing in cost are still initially relatively expensive in comparison with human labour, but are able to operate 24 hours a day when required. This potentially is able to reduce wasted fruit, and if able to match a human in regard to harvesting rate, may prove to be an effective and economically viable solution to this task.

This research aims to answer these questions, by proposing, implementing and evaluating a novel navigation and control system for an autonomous fruit harvesting system, in a challenging, dynamic vertical growing environment.

This chapter contains the motivation behind this research, the objectives and challenges it presents, the methodology and contributions made and an overview of the thesis.

1.1 Motivation

Robotics is an interesting, exciting and innovative field of research, often combining many disciplines, including navigation systems, control methodologies and computer vision solutions. Their applicability extends to almost all environments, and their use can be greatly beneficial to society. Autonomous robotic systems in place of humans are able to explore environments that are often remote, inhospitable or hard to reach. They can be used in place

of humans in hazardous, dangerous environments such as disaster recovery and explosive ordinance disposal or to entirely replace humans in many industries. With current global issues such as climate change leading to a reduction in viable agricultural land and labour shortages, they are able to replace humans for mundane, laborious and time consuming tasks, whilst potentially doing so in a more efficient, economical and productive way. An example of this can be found in strawberry harvesting. This task to a human is reasonably straight forward, with skilled labourers able to pick up to 50 strawberries a minute. To a robotic system however, this requires complex, adaptable and dynamic behaviour, such as first perceiving a target strawberry then determining how to reach and harvest the fruit. To develop a system capable of these applications, many complex problems require solving.

Autonomous navigation is in of itself a challenging problem. Different terrain and dynamic environments require various sensors to enable a system to perceive its surroundings. When these systems are to be used with humans, safety also becomes an important factor, a problem to which there is yet a perfect solution. Navigation is one aspect of an autonomous system, for it to be applicable to many tasks, complex control methods are also required dependent on the tasks the system is developed for and the components required for these. Many applications require these systems to contain additional components such as robotic arms, effectively making these systems of systems, these requiring novel solutions to incorporate these additional components and to effectively use them dependent on their application.

These systems present challenging, exciting and interesting problems. This research aims to contribute to the field of robotics by proposing, implementing and evaluating a solution to a multi-robot system for autonomous strawberry harvesting, although intended for this application, the solutions proposed may be applied to autonomous robotics in many other

applications.

1.2 Objectives and Challenges

The objective of this research is to propose and implement a novel solution for an automated strawberry harvesting robot in a vertical growing farm.

This problem presents interesting challenges, a method of integrating multiple components and systems, a novel solution for navigation and control in this type of environment and for this application and to do so in an efficient, economically viable way. The main objectives of this research are summarised as follows:

- The design and development of an adaptable, versatile software architecture to facilitate the multiple components and systems required.

- The development and implementation of a novel navigation and control method, capable of operating efficiently and safely in an agricultural environment.

- To determine the efficiency of the system, if it is possible for it to be applied to this task with the same if not improved performance of a human, whilst remaining economically viable, and if any improvements could be made.

These objectives contribute individual aspects of this research. A novel solution to the systems architecture must first be developed, before a navigation and control methodology can be implemented followed by an evaluation of the system and its performance, determining if any improvements can be made. The following subsections detail the challenges involved in solving these problems.

1.2.1 Fruit Harvesting Robotic System Architecture

The use of robotics in industrial or factory automation applications is often a single, repetitive task. A system can be developed to complete one specific objective efficiently, but does not often require much if any adaptation or reconfiguration except in replacing faulty or upgrading existing components. These systems can be tightly coupled, with no impact on their application as they are only intended to complete one task.

The system developed for this research requires multiple robotic components, sensors and software modules to operate in parallel in order to complete its intended application. This system of systems, particularly in its initial development, may require many adaptations with additional components and sensors being added or removed, these requiring accompanying software that must also operate in combination with the rest of the system. The environment the system is intended for is itself dynamic, a configuration that is applicable to a laboratory environment may require adaptation and reconfiguration for application in a vertical farm environment. The robot will also be working alongside humans in the vertical farm, this also requires safety standards to be met, for example the TS ISO 15066 international safety standards and ISO 10218-2 collaborative robot safety standards. These include ensuring the robot operates within its specification boundaries such as arm or base movement speed and would require a risk assessment before being used commercially in the vertical farm environment.

This system is required to not only harvest the strawberries, but to do so autonomously, as such a navigation module is required to move along a row of fruit. There are many po-

tential solutions to this, including the use of GPS or SLAM systems, as such, determining the viability of a navigation method is a vital part of this system. This type of adaptability requires a new methodology for a modular, configurable, loosely coupled system architecture, to enable these adaptability and configurability, as such, a solution to this objective is proposed as the first chapter of this research.

1.2.2 Multi-Robot Navigation and Control

Virtually all robotic systems incorporate multiple components depending on their application, for example automated warehouse robotic system requires a navigation method to move around its environment and components such as a gripper and carrying plate for moving items. Various sensors are required to dynamically detect and avoid obstacles and to target and locate items to be moved. These functionalities require the coordination of multiple components, operating around a single frame of reference. This can be summarised as a navigation and control method.

Agricultural environments present unique challenges rarely found in other applications. The terrain the systems are required to operate in is often unstructured and undulating, resulting in accumulating error and drift in navigation systems. The systems are often required to operate around other machinery and humans, whilst avoiding damage to themselves and the environment around them.

In harvesting fruit, the position and elevation of the fruit bushes is often dynamic, unstructured and occluded and atmospheric factors such as weather conditions all need to be taken into account. This presents many navigation and control challenges, the multiple systems

need to be combined to operate dynamically together, on varying terrain and with differing target positions for strawberry harvesting. The second chapter of this research proposes a novel navigation and control method for an agricultural environment to enable this functionality.

1.2.3 Performance and Optimisation Improvements

A potentially prohibitive factor of autonomous robotics is their cost. When a system is developed for automated manufacturing, the initial cost can be relatively high, but this is often offset by the systems improved efficiency and productivity. The risk of system failure is also considerably lower as these applications are often static.

When used in warehouse or industrial automation, these systems are often in structured relatively static environments, further reducing the risk of damage to the system and its surroundings. Agricultural environments are considerably different and pose many additional risks to the system and its environment, these all contributing to their economical viability. Along with reducing the cost of these systems, performance and optimisation improvements and additional functionality can all assist with offsetting the economic risks of investing in these systems.

The initial development of the system is as a proof of concept, to determine whether the proposed solution to autonomous fruit harvesting is capable of this functionality. Once this type of system is developed and evaluated, potential improvements can be made. The systems modular design and configurability also enable these improvements to be integrated with the existing system, without redesigning it overall. This chapter looks at potential improvements that could be made by reducing component costs and additional functionality such as metric collection and environment monitoring.

1.3 Methodology and Contributions

The methodology followed for this research was to first propose a software architecture incorporating the components of the system. Once this had been established, to then develop this and implement the components of the system, to have a minimum viable robot for laboratory and farm testing. As the intended environment of the robot is seasonal, much of its development and evaluation was completed in a laboratory environment.

The navigation software would initially be simulated using a custom developed virtual representation of the robot in an agricultural environment, to ensure the initial navigation software would function as expected before being tested in a laboratory and the vertical farm.

This approach was chosen as to ensure that the system could be evaluated in a farm environment when available, but so that development could continue when this was not possible.

This research methodology is separated into the following sections:

- Configurable Software Architecture for an Agricultural Robot
- Strawberry Picking Positional Control
- Single Board Computer for a Strawberry Picking Robot

These individual sections of this thesis each providing a solution to the objectives of this research.

1.3.1 Configurable, Modular Software Architecture for an Autonomous Strawberry Harvesting System

The software architecture of the system, although developed for a strawberry harvesting robot, also had the potential applicability of being a generalised solution for automated har-

vesting. This involved designing the software architecture in a way as to enable configurability and adaptability. The software architecture would was designed to enable additional modules, components or sensors to be easily added or removed, without redesigning the system. The network used by the perception system for example could be changed to another crop or an additional component or sensor could be added for another task if required. This was enabled by ensuring the system was loosely coupled and compatible with various software modules. This section contains the following robotics research contributions:

- A configurable, modular software architecture for an autonomous fruit harvesting system.
- The software design and implementation of the custom built, modular system.
- Initial evaluation and proof of concept of the system in a laboratory setting.
- Initial evaluation of a SLAM algorithm in a vertical harvesting agricultural environment.

This first set of objectives being to enable modular system components and software to be combined, then allowed for the navigation and control software to be developed.

1.3.2 Strawberry Picking Positional Control Movement Strategy and Implementation

The systems navigation and control is to enable the system to efficiency move to and between strawberries autonomously, whilst harvesting them. This providing the following research contributions:

- 1. The development and integration of the modular configurable system architecture.

- 2. Novel navigation and control method for the robotic arms and base in a vertical facility.

- 3. The experimentation and evaluation of the system in a laboratory environment.

- 4. The experimentation and evaluation of the system in the agricultural environment.

The final section of this research was to determine and implement potential system improvements and optimisation, whilst also aiming to reduce the cost of the system.

1.3.3 System Improvements and Performance Optimisation

The systems performance and optimisation improvements focused on reducing the cost of the system components used, whilst adding additional functionality, this providing the following research contributions:

- 1. A comparative analysis of instance segmentation networks on a Laptop, Jetson Nano and Jetson Xavier is conducted to ensure the single board computer is capable of running the existing system in an agricultural environment.

- 2. A 3D instance segmented environment reconstruction system is proposed.

These sections detailing the proposed method implementation, evaluation and results for each of the objectives of this research.

1.4 Overview of Thesis

This thesis is separated into 3 main research chapters, a literature review and conclusion, a brief synopsis of each section is as follows:

1.4.1 Chapter 2: Literature Review

This chapter gives a theoretical background to my thesis, with a detailed exploration of existing technologies and research in autonomous robotics in general and in an agricultural environment. This review includes navigation techniques that may be applicable to the system such as the use of GPS, inertial navigation systems and simultaneous localisation and mapping as well as control methods for robotic systems for static and autonomous applications.

This section also contains a review of computer vision techniques that may be applicable for improvements to the system, including semantic and instance segmentation networks and environment reconstruction systems.

1.4.2 Chapter 3: Configurable Software Architecture for an Agricultural Robot

The first main research chapter of my thesis details the software architecture of the system, the design considerations for the specific components of the system, the architectures implementation and initial proof of concept experimentation and evaluation of the potential navigation methods in the vertical farm environment.

1.4.3 Chapter 4: Strawberry Picking Positional Control

Chapter 4 includes the development, implementation and experimentation of the navigation and control software for strawberry picking and positional control.

The systems movement strategy is first defined, before the systems position and component

control methods are detailed, including the parallel operation of the multiple components of the system. The experimentation and performance analysis of the system is then shown in the laboratory and agricultural farm environments.

1.4.4 Chapter 5: Single Board Computer for a Strawberry Picking Robot

Chapter 5 proposes potential improvements and optimisation methods for the system, including a method of reducing the system component's costs and of introducing additional functionality using a potential replacement for the robots perception system, a 3D environment reconstruction method.

This includes a comparative analysis of instance segmentation networks on different single board computers. It is first conducted to determine if these are capable of replacing the existing perception system, this followed by the initial development of a proposed replacement strawberry perception system with additional functionality.

1.4.5 Chapter 6: Conclusion and Future Work

This final chapter gives a summary of the research in this thesis, the contributions made and the academic publications generated from this research.

The potential applications of the systems developed are also discussed, with final comments on future work that could continue from this research.

Chapter 2

Literature Review

Robotics is a multidisciplinary area of research, complex systems are often composed of many sensors and components working together as part of a system overall. Research and continuous improvements in autonomous robotics are enabling additional and new capabilities for these systems in an increasing number of industrial and commercial applications. For a robot to be autonomous and able to move around an environment, it requires a number of systems, the first of these being navigation capabilities, of which there are many varying methods and approaches. A control method for the individual components and their combined functionality is also required, along with various sensors to perceive its environment, dependent on the robotic systems intended use and the environment it will be operating in.

This research is focused on a navigation and control system for an autonomous strawberry harvesting robot, as such, the current state of the art in agricultural robotics, navigation techniques and multi-robot control methods are of particular interest. This system uses an existing perception system to detect target fruits using a camera sensor, an additional use of

the robot may be in metric collection and environment monitoring, as such, techniques such as semantic and instance segmentation methods are also explored to determine the state of the art in these fields and their potential applicability to this system as another method of perceiving its environment.

2.1 Robotic Systems and Applications

Autonomous robotics are becoming increasingly applicable to a vast array of environments and tasks in many industries, with significant advancement in recent years. Many labour intensive, repetitive and often dangerous tasks can now be carried out by robots such as drones, unmanned ground vehicles and robotic arms, with these systems often improving efficiency, productivity and safety in the areas they are applied.

Autonomous robots are becoming increasingly viable in extreme environments or for hazardous tasks such as search and rescue [7] [8] [9], explosive ordinance disposal [10] and geographic mapping in inaccessible or inhospitable environments [11] [12]. Along with these more specific uses, they are also commonly found in industrial and commercial settings such as warehouse management [13], manufacturing automation [14], automated drone delivery [15] and for the purpose of this system, in agricultural settings [1] as shown in Fig.2.1. [2] as shown in Fig.2.2. [16] [17] [18].

2.1.1 Agricultural Robotics

Agri-Tech has been an active research field for a number of decades, as far back as the early 1900's, a frequent contribution being automated driving systems utilizing GPS [19] [16] [1] [20] [21]. These systems have often required human supervision if not direct involvement. But with improving autonomous unmanned robotics technology, many new applications are

now becoming feasible and actively seeing development.

Figure 2.1: RASberry based on Thorvald II System [1].

The application of autonomous robots in agriculture has many benefits, through increased productivity with continuous, efficient harvesting and reduced waste with pest detection and removal to name a few, these becoming increasingly relevant with current global issues such as labour shortages, climate change and a reduction in viable agricultural land. There have been a number efforts for GPS assisted, remote controlled agricultural robots, such as seen in [16], this system is used for ploughing, seeding and leveling farmland. The system proposed for pumpkin harvesting [22] is another interesting method, this system using Bluetooth and an application for its control. Although viable in large, open, outdoor environments, for systems requiring millimetre precision such as fruit or vegetable harvesting and to operate indoors, GPS is often too inaccurate, unreliable if not entirely unavailable. Remote controlled methods also require human supervision or operation, this, although useful in reducing labour requirements, would not be an entirely autonomous solution. There are a number of autonomous research efforts into fruit harvesting [2] as shown in Fig.2.2 [23] [24] [25].
A cost effective fruit harvesting system has been developed using an Arduino, robotic arm and vision system for detecting and reaching fruit [25]. This method although able to reach a fruit does not have navigational capabilities. The orange harvesting autonomous robot is

able to detect and move toward a fruit tree in order to harvest the fruit [23]. This method uses a ZED camera to determine the centre point of a tree and uses these coordinates to move toward it, before the robotic arm then harvests the fruit. The Thorvald II [2] system as shown in Fig.2.2 is an example of a mechanically modular agricultural robot, designed to be re-configurable for many different environments. This system again often relies on GPS which can not be guaranteed in its environment, or human input for control. RASberry [1], a system based upon Thorvald also using GPS as well as LiDAR for navigation, operates in ground based poly-tunnels, with its localisation based upon a combination of GPS and LiDAR data for movement.

Figure 2.2: Thorvald II System [2].

In recent years there has been significant interest in the harvesting of fruits such as strawberries, one such example capable of this task in a polytunnel [17] is a following further improvement of a previous system [18], with a modified, cable driven gripper. This system uses a robotic arm and mobile base for harvesting strawberries in ground based static fruit trays, navigating using a 2D LiDAR, this also based upon the Thorvald II [2] robot.

This task, although receiving considerable interest in recent years, is yet to have a perfect solution, many systems are needed to operate together, without GPS, in unstructured, dynamic and often undulating terrain environments as seen in the vertical strawberry farm,

whilst maintaining the precision required to harvest strawberries, without damaging them and efficiently as to be an economically viable human replacement.

2.2 Navigation Methods

Navigation systems for autonomous robotics has been an active field of research for decades [26], with many differing techniques proposed dependent on the environment. Global Navigation Satellite Systems (GNSS) such as GPS are predominantly used in outdoor environments, ideally with a line of sight to multiple satellites to operate and are often unusable in occluded or indoor environments. As such, alternatives such as dead reckoning, using artificial or natural beacons and simultaneous localisation and mapping have been and continue to be of significant interest. There is as of yet, no absolute replacement solution for the capabilities offered by GPS, although techniques such as SLAM in particular are becoming increasingly capable, combined with machine learning. These may soon be capable of providing the same coverage and reliability in indoor, as GPS in outdoor environments.

Navigation in an agricultural type of environment often has the the challenges found in urban or indoor environments, along with many others, a lack of structure and often varying terrain and dynamic obstacles being of significance. The environments often being large and open, with few easily determinable reference points can lead to rapidly accumulating error in position estimation. This drift if unaccounted for can quickly and dramatically reduce the systems localisation accuracy.

A common solution to this in outdoor environments has again been found in GPS, this having

long been a solution for the automated traversing of farm environments [19]. This system is able to navigate in outdoor environments but it may not always have the accuracy required in all settings, for example if traversing under tree cover or into a large structure, and is not a reliable option for this system. This navigation solution and GPS overall would not be suitable for this systems navigational requirements as the facility is predominantly in an indoor, closed environment, and requires centimetre precision in a confined area when traversing throughout each row of vertical hanging baskets in order for the arms to operate as intended and without damaging the fruit or vertical growing system.

2.2.1 Dead Reckoning and Sensor Fusion

An early technique in navigation systems that can be used in place of GPS is dead reckoning, the estimation of a robots position as it moves, based on previous sensor data. This method can use various sensors such as wheel odometry, LiDAR measurements or an inertial measurement unit (IMU), a combination of gyroscopes and accelerometers, to determine and track its pose. These systems are self contained, requiring no external input for pose estimation and are, in ideal environments, precise in tracking a systems position initially. As a robot moves, many small errors from sources such as wheel or track slippage, can accumulate rapidly. This accumulating error can lead to drift, a robots estimated pose according to its sensor readings differing from its actual position. An IMU alone for most applications is insufficient. A method to combat this can be through combining these sensors with additional data sources to improve their pose estimation, by periodically having their absolute pose updated. The dead reckoning method has been a navigation technique researched for many years, with considerably more advanced solutions now available, such as those found in sensor fusion methods.

The use of combining data from multiple sensors is an interesting research area for navigation systems and a potential replacement for GPS based systems, improving sensor accuracy and the use of filter techniques for pose estimation, such as the variants of Kalman filter, can enable this method to be reasonably effective. As this method can operate in GPS denied areas, it can be particularly applicable to agricultural environments, with active research into systems using this method [3] in Fig.2.3., this utilizing an extended Kalman filter for pose estimation, using wheel odometry, IMU and camera sensor data in an agricultural environment.

Figure 2.3: Visual, Wheel Odometry, IMU Robot [3].

2.2.2 Artificial and Natural Beacon Based Navigation

A potential solution considered for these requirements was through natural or artificial beacons [27] [28]. This type of navigation uses artificial or natural way points such as landmarks, coloured markers or WiFi to determine and update a systems location. This type of way point system may not require prior knowledge of the environment, but may need constant maintenance in a dynamic agricultural setting. This would not be suitable as the interior environment itself is dynamic and frequently changing in regard to the position of the vertical growing baskets. This may render natural or artificial landmarks or coloured markers unreliable as there visibility would not be guaranteed with the frequently changing position of the vertical hanging baskets, with WiFi also being unreliable if not unavailable in this environment and not providing the positional accuracy required to operate safely and effectively in the vertical rows.

The terrain being soil, dust and plants would also effect the visibility of ground based beacons, reducing this and requiring both time and labour to maintain. Using artificial or natural beacons as way points for predetermined path planning would also not be a viable solution for this navigation system as the vertical growing facility is frequently changing its layout, the hanging baskets are fixed, moving vertically only, but are free hanging and may sway or move, not necessarily returning to their exact initial position.

2.2.3 Simultaneous Localisation and Mapping

A potential solution for the navigation system in this research is that of a combination of global simultaneous localisation and mapping (SLAM) for larger, less absolute accuracy

dependent areas, and a precise local path planning controller to keep the robotic arms in operational range.

SLAM is a necessity for many applications in automated robotics, for a robot to know its local or global location in an unknown environment it requires a map, to develop a map it requires its location. SLAM systems use their sensors to generate a map of an environment whilst localising to this map. These systems accuracy in location estimation relative to their environment can be effected by many external influences, accumulating error from drift as with dead reckoning or inaccurate sensor readings can quickly and dramatically effect the reliability of its determined location, this also effecting the accuracy of a map generated. With these challenges, there have been many various techniques and methods explored, with intriguing results and improving system's navigation capabilities. SLAM has been an active research area for many years with a wide variety of differing methods developed using multiple combinations of sensors [29] [30] [31] [32] [33], including LiDAR with efforts such as GMapping [29] and Google Cartographer [30] and RGB-D camera based systems such as ORB-SLAM [34] [35] and RTABMap [36]. These systems use sensory information to determine a robot's pose whilst keeping track of its position as it moves through an environment, simultaneously localising and updating the map as it moves throughout an area.

A relatively successful approach to this problem has been in the use of occupancy grid based methods [37] [38], using odometry readings and a laser scanner for example to gradually generate a map of an environment based on the likelihood of a grid cell being occupied or empty. This method could often face problems from accumulating errors, an issue that can be countered by using filtering techniques as with other methods, such as the Kalman and

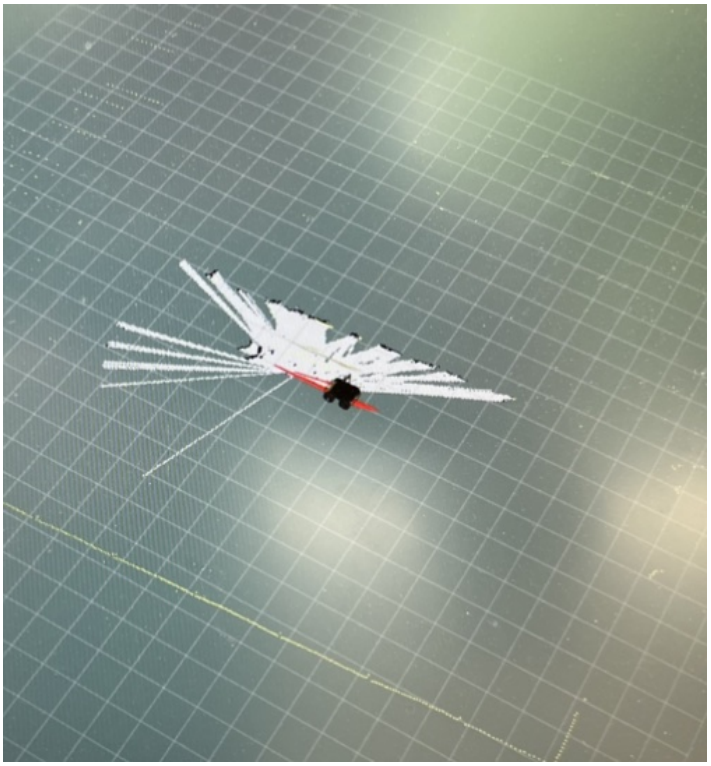

Figure 2.4: GMapping SLAM with Mobile Robot.

extended Kalman filter [39] [40] or the ICP algorithm used in Robust 3D SLAM [41]. With the lowering cost and improving capabilities of 3D laser scanners, LiDAR based point cloud techniques have also been developed for SLAM, the data collected from a laser scanner can be used to generate a 3D point cloud rendering of an environment, with accurate depth measurements obtained from landmarks and objects [42], [43] [44] [45]. There have also been approaches taken using camera sensors in visual SLAM methods, for example, with stereo based SLAM methods [46], [47] taking two images from a precisely calibrated stereo camera and determining the disparity between these two images to determine the depth from an object or environmental feature. This can then find distances from objects to be used as beacons that, through triangulation or trilateration, for example can be used to determine the systems location in an environment whilst also mapping this environment. Along with stereo vision, monocular cameras have also proven to be effective for real-time simultaneous local-

isation and mapping [48] [49] [34] [35] [50] [51]. A monocular camera can also be used to determine distance in images by taking two images and determining the disparity between two pixels in a corresponding image, the depth of objects and environmental features in an image being determined using this method.

As with previous methods, sensor fusion techniques such as combining a laser and camera sensor for simultaneous localisation and mapping have also been research areas of interest [52], for example in [53] using a camera to detect landmarks, and a laser to determine the distance as well as the RGB-D camera methods [54] [55]. The problem of localising to and mapping an unknown environment is still an active area of research, with the lowering cost and improving performance of 3D laser scanners. Their use in SLAM systems combined with a monocular or stereo camera could assist in the systems accuracy, coalescing the detail inherent in an image with the accuracy of a lasers distance measurements. A system combining these sensors could then also be used for purposes such as 3D environment reconstruction as the required data to generate this would have already been collected as RGB images and point cloud data.

2.3 Multi-Robot Control Methods

A robotic base on its own may find use in exploration tasks but to be useful in many environments requires others systems to be integrated, such as robotic arms, end effectors, dependent on its intended application. The combination of multiple robotic systems requires various components to operate cohesively. Complex systems often need to operate in the same coordinate reference frame to ensure they do not conflict, whilst also potentially en-

able multiple individual systems to operate in the same environment. The software libraries using robot operating system (ROS) are ideally suited for this task, enabling multiple components and sensors to function together in the same reference frame.

2.3.1 Robot Operating System

Robot Operating System (ROS) is a collection of libraries facilitating the use of multiple robotic components and sensors. ROS enables individual components and sensors of a system to be developed and used as nodes that can communicate with each other through one main server, in a publisher, subscriber model as shown in Fig.2.5. [56] [57] [58] This system enables rapid development and prototyping of robotic systems using this model, whilst also having various tools such as simulation and robot model development tools, Gazebo and RViz, respectively, available for testing software.

ROS also has a method for combining multi-system robots in its transformation tree structure, this enables multiple components to operate in the same coordinate frame by linking them to a single base component of which the other components of the system use as a reference point. This would enable a robot comprised of, for example, one robotic base and multiple robotic arms to all operate using one coordinate system, instead of accounting for each components seperate coordinate transformations when the system moved. The requirement of multiple systems operating together with various sensors for this research make the use of ROS ideal.

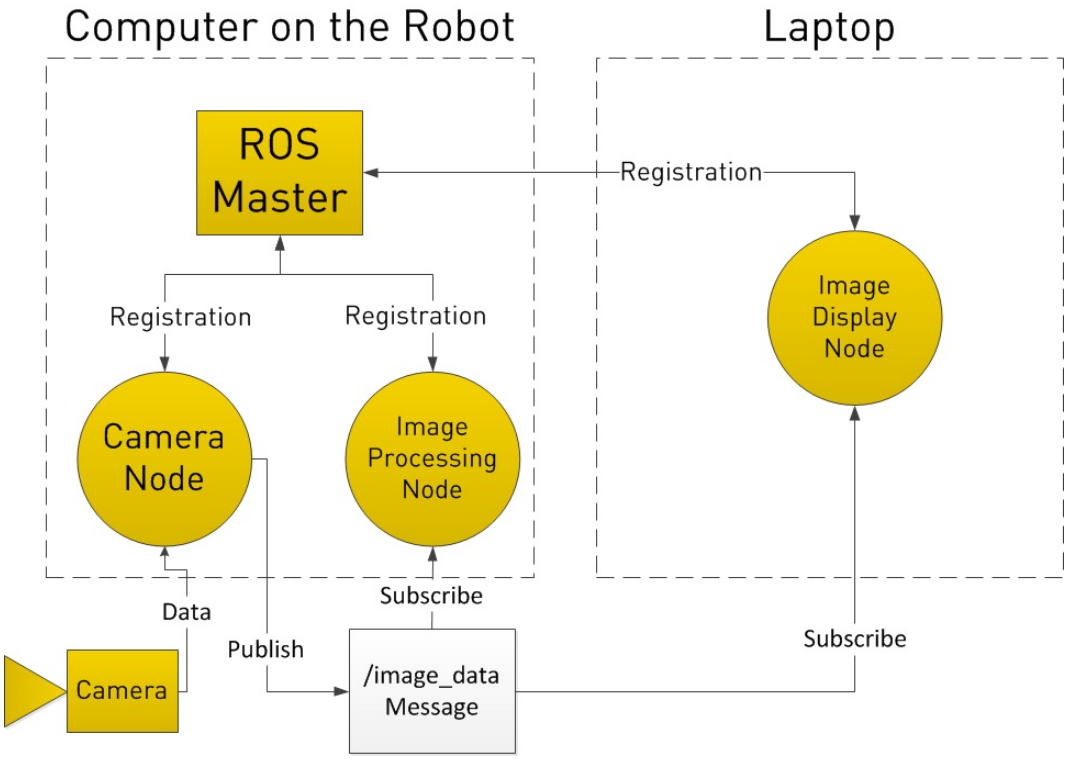

Figure 2.5: ROS Structure, Clearpath Robotics [4].

2.4 Semantic and Instance Segmentation

Semantic and instance segmentation is the continuation of research in object detection and recognition. Semantic segmentation is the process in which every pixel in an image is labelled and pixel-wise classified. These pixels are then grouped together as individual objects by having the same label, for example people and cars. Improving on this, instance segmentation is the process of detecting each individual object in an image, for example each individual person or car. This is achieved by identifying each individual object in a pixel-wise classified image.

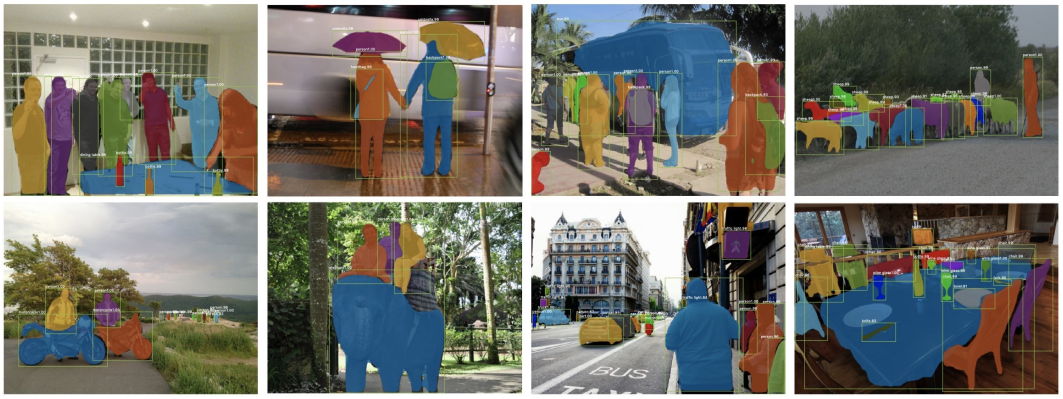

Figure 2.6: Instance Segmentation using Mask R-CNN [5].

2.4.1 Camera Based Methods

There has been significant progress in this field using camera sensors, with many differing approaches. A number of various approaches to this problem inspired by the "Fully Convolutional Networks for Semantic Segmentation" [59], the first fully convolutional neural network for semantic pixel-wise labelling. This initial work leading to improvements such as FCAN [59], ICNet [60], DeepLab [61] and ERFNet [62]. Recently, the focus has shifted toward improved inference speed for real-time performance, whilst maintaining and improving accuracy in semantic segmentation. The BiSeNet [63] and BiSeNetv2 [64] methods propose separating a network into a spatial branch and a semantic branch, this improving the inference speed of semantically segmenting an environment, whilst ensuring accuracy loss is kept at a minimum using an RGB-D camera sensor as shown in their results. Along with this research, approaches have been taken in using the data acquired by a camera sensor for semantic segmentation to also be used in 3D reconstruction as seen in, [65] proposing a dual use for their CNN, both semantically labelling and reconstructing an environment, and "Dense RGB-D Semantic Mapping with Pixel-Voxel Neural Network" [66]. These methods utilise both monocular images, and RGB-D data collected respectively to semantically seg-

ment and reconstruct an environment.

Object detection and recognition to semantic segmentation has now progressed to identifying and labelling individual instances of objects in an environment, instance segmentation. Although relatively new, this is also an active area of research, with advancements such as MID-Fusion [67], a system capable of 3D reconstructions of objects in an environment, whilst also instance segmenting them using an RGB-D camera sensor. This has also been attempted using a recurrent neural network (RNN) [68] and achieved impressive results at the time it was published along with ReScan [69] and "Volumetric Instance-Aware Semantic Mapping and 3D Object Discovery"[16] for indoor environment object instance segmentation. The more widely used CNN approach has led to the current state of the art in camera based instance segmentation in Mask R-CNN [5]. Mask-RCNN [5] is based upon previous work in RCNN [70], Fast-RCNN [71] and Faster-RCNN [72].

RCNN, Fast-RCNN, Faster-RCNN and RPN

R-CNN [70] uses selective search to find regions in an image that may contain an object. It does by looking for regions of interest (ROI). A ROI is an area in an image of similar pixels, colour or texture for example, that are grouped together in regions because of their similarity. These boxes are then passed to a convolutional neural network for feature extraction using layers of filters, instead of the input image with no initial attempt at identifying objects, speeding up the process, before then in R-CNN being passed through a support vector machine (SVM) layer to determine and output a classification. Fast RCNN [71] was the next successor to this. R-CNN [70] is computationally expensive as through selective search, a large number of regions were selected before then individually being passed through the

CNN for classification. Fast R-CNN speeds this process up by first passing the input image through a CNN to generate the ROI's using selective search on the feature map generated by the CNN.

A feature map is the output produced after an image passes through a layer of filters, with each filter designed to detect a specific feature, for example to detect a straight line, in a convolutional neural network, there can be anything from 10's to 100's of filters in a layer to detect these features. It then continues through the network to be classified through an ROI pooling layer and finally an activation function layer resulting in a predicted classification. The ROI pooling layer enables a list ROI's to be checked against one feature map, instead of having to check each ROI individually.

This is done by downsampling the regions into smaller sub samples of equal size, i.e. summarised versions of the features detected in the input. This downsampling can cause loss of data, an issue resolved using ROI align. The next improvement to Fast R-CNN is Faster R-CNN [72], Faster R-CNN [72] is a real time object detection network introducing the method of regional proposal networks (RPN). RPN's further speed up this process by being more computationally efficient in how the regions of interest are found. RPN's generate a prediction of an object in an image being an object, given the proposed region meets a defined threshold.

An input image is first passed through a CNN to generate a feature map. The feature map is then scanned over to generate a series of anchors. Anchors are boxes of various sizes generated across the entire previously output feature map, often overlapping to cover as much of the image as possible. These anchors detect first the background or foreground of an image, before then refining there alignment over the potential object. The most likely of these ob-

jects is determined by the intersection over union (IoU) of all of the predicted regions, if the IoU reaches a certain threshold, the region is considered an ROI, removing unlikely ROI's and keeping the ones with the highest prediction of containing an object. These proposed ROI's are then passed to an ROI pooling layer which enables a single feature map containing all of the proposed regions of interest to be passed to a CNN for classification instead of individual regions of interest, before then to an activation function layer to be classified.

Mask-RCNN

Mask R-CNN [5] is built on Faster R-CNN [72] and is the current state-of-the-art in semantic segmentation using images. Mask R-CNN is a significant milestone in image recognition as it is capable of real time semantic segmentation and introduces the concept of ROI align. Mask R-CNN, similar to Faster R-CNN, first passes an input image through a CNN to generate a feature map as an output. After this, the feature map is then passed through an RPN to detect regions of interest to be passed through the convolutional network. The RPN generates proposed regions of interest, with a foreground or background class, along with a refined alignment over the potential object. In the next stage of Mask R-CNN [5], the proposed regions are passed through the network to further refine the object detection using ROI align, the process of ROI align has the same function of reducing ROI size to all be smaller and equal, but losing less data in the process than in ROI pooling, a more solution. The final stage of Mask R-CNN is the addition of segmentation masks, image masks enable certain parts of an image to either be visible or not by changing the pixel values, depending on the values of the mask. In this stage, a mask is generated for each ROI, for example removing the background to reveal just the object that is being detected, the masks generated in Mask R-CNN are low resolution, these are then scaled up for inferencing, with the networks output

being the instance segmented predicted class of an object.

2.4.2 Laser Based Methods

There has been significant interest in research in using laser sensors for semantic and instance segmentation in recent years as the cost of these has reduced dramatically, whilst the performance has improved. Earlier research in this area can be found in "Fast Semantic Segmentation of 3D Point Clouds with Strongly Varying Density" [73], SegCloud [**?**] and a fusion of image and point cloud [74] for semantic segmentation developed by R. Zhang, et al, as well as PASS3D [75], an outdoor semantic segmentation method using a LiDAR scanner performing comparably to SqueezeSeg [76] and PointSeg [77]. These along with the methods proposed in RangeNet++ [78], PointSeg [77], OREOS [79], MultiModal [80] and DeepTemporalSeg [81] also demonstrate accurate semantic segmentation capabilities in outdoor scenes. This work in semantic segmentation has subsequently then led to research in instance segmentation using laser scanners. This field has considerably less research than can be found when using RGB and RGB-D camera sensors or semantic segmentation using a laser scanner, although there has been a number of efforts proposed. A substantial contribution has been made through [82] using a LiDAR scanner to detect instances of objects. The authors also provide a new point cloud dataset that is publicly available. This is a substantial contribution as a large limitation to research in this area is the availability of labeled datasets for training point cloud based networks. The current state of the art in point cloud segmentation and instance segmentation is PointNet [83] and PointNet++ [6].

Point Cloud Segmentation

The task of semantic and instance segmentation using point cloud data, although an active field of research is relatively new compared to that of RGB and RGB-D based techniques. The difficulty with point cloud data is that it is an unstructured set of data points, this can make determining objects in 3D difficult when points may not necessarily be related, although close together. A recent breakthrough in this field has been found in PointNet [83], and the current state of the art PointNet++ [6]. There can be thousands of inputs in any given set of point cloud data, the general structure of point cloud data being unordered means there can be N! permutations of this data. To use this data for object detection and semantic segmentation, it needs to be invariant to these permutations, to represent the same structure if rotated or translated for example.

PointNet

PointNet [83] itself takes point cloud data directly as an input, and outputs a classification label for an object, or per point classification for semantic segmentation. The architecture of the PointNet network consists of three components, each point at first is processed independently having a feature transformation applied, before then a max pooling layer and generating a global feature vector of the inputs. The input data in the form of coordinates is mapped to a space. The result of this mapping is a global feature vector of per point features. To ensure the order of the points does not have an effect, the authors make use of max pooling to downsample and reduce the dimensionality of the feature vector. The idea being, once a global feature vector has been generated from the data set, through max pooling the data will show the features regardless of transformation for example translation or rotation. This is important, as with RGB image classification, the object being detected is 2D, where as in

point cloud object detection, the object being detected may be formed of points that from one angle seen unrelated, but from another represent part of a structure, as distance between points are a fundamental part of the structure in point clouds. The global feature map is then passed back to the initial per point features and linked with its respective global feature point generated from the max pooling layer, this gives the new per point features local and global information. The final stage of this network is to apply an affine transformation matrix to the input points. This is achieved by a mini-network resembling the larger PointNet network. The network is fully connected and, using per point feature extraction and max pooling, the affine transformation matrix generated from this network is then applied to the input points to correct for geometric distortions that may occur, generating a more accurate representation. PointNet is capable of 3D object classification, 3D part segmentation, and semantic scene segmentation, and is the basis for PointNet++ [6].

PointNet++

PointNet++ [6] is an extension of PointNet, with an added hierarchical structure of set abstraction levels, these levels are comprised of a Sampling layer, Grouping layer and PointNet layer as shown in Fig.2.7. The Sampling layer selects a set of points, defining the centre of a region, points near this centre are then grouped together in the Grouping layer, and finally the PointNet layer uses the same process as the PointNet network to determine feature vectors from these sampled and then grouped data sets. The Sampling layer uses Iterative Farthest Point (FPS) to determine the subsets. This gives a better coverage of the entire point set than rather than random sampling, as through random sampling centre points may not be as evenly distributed across the entire dataset. The points near these central points are then grouped together in the Grouping layer in varying densities as the number of points may be

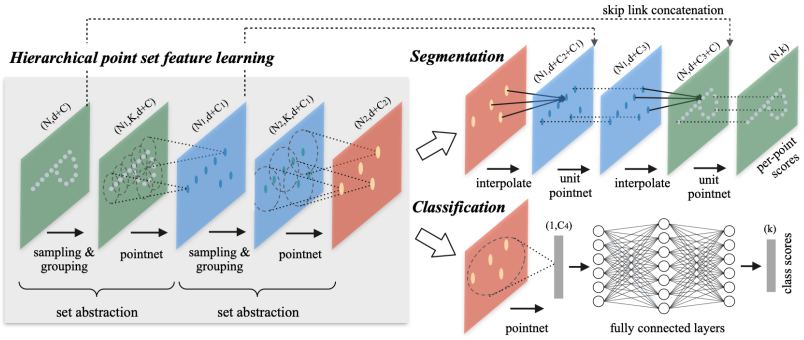

Figure 2.7: The Hierarchical PointNet++ Feature Learning Architecture [6].

different in different regions. The PointNet layer then applies the PointNet process to the input from these subgroups to output local region feature vectors. These local region features may vary in density, an object in a region for example will have a higher point density in the set than an empty region. This could cause an issue for generalising feature learning, in more dense or sparse areas of data.

The authors of PointNet++ attempt to solve this problem by looking for larger features in sparse data, and finer features in dense data through a process they have named density adaptive PointNet layers. This process learns to combine features from different scales and different point densities. To combine features from different point densities and scales, the authors give two solutions. The first being multi-scale grouping (MSG) shown in Figure 2.8.A., where features at different scales from every local region feature vector that were previously obtained, are captured and are then grouped together. The network is trained to optimise this multi- scale feature grouping by randomly dropping input points, which they call random input dropout, from each set of points. This gives the network various sparsity training data. This solution can be computationally expensive however as there are often a large number of central points. The second option is multi-resolution grouping (MRG)

shown in Figure 2.8.B. This compares two feature vectors and uses the most reliable vector based on its point density. This is done by joining two vectors of points together at a given set abstraction level. The first is from the feature obtained in a local region by processing all of the points in the region through a PointNet [83] at the given level. The second vector is generated by joining all of the sub region features from the level below a given level. This joined set of two vectors can then be used depending on the point density of a region, the vector chosen is determined by a weight given depending on the requirements. If the density is low, the first vector could be less accurate because of sampling deficiency, in which

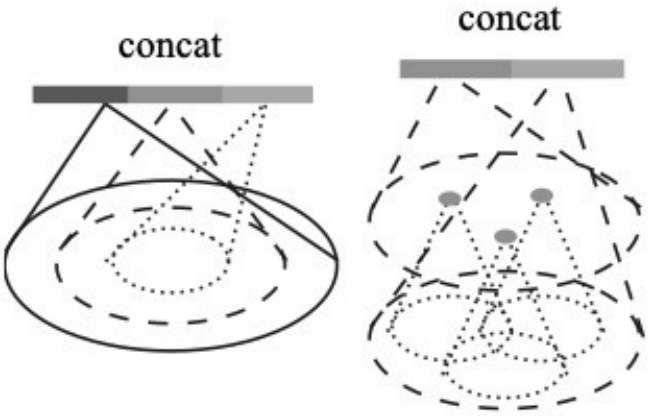

Figure 2.8: A: Multi-Scale Grouping (MSG). B: Multi-Resolution Grouping (MRG) [6].

case the second vector could be used, if the point density is high, the first vector may be more accurate as it can show finer details in higher resolutions. This solution was found to be more computationally efficient as the density adaptive layers enable the first vector to be used for higher density regions and the second vector to be used for lower density regions for feature extraction. The final part of the PointNet++ network is to output classification and segmentation predictions. The segmentation part of this network joins points together based on distance and using skip links, to skip between levels before these grouped points go through a PointNet network, this process is repeated for feature extraction from the initial

set of points. For classification the feature vectors go through a fully connected network using the rectified linear unit activation function (ReLU) to update the points for classification. This extension to the PointNet network adds a hierarchical grouping method to the data set, extracting features from different levels of different sub sets of the initial data set. This improves the performance of the network significantly, with PointNet++ having a lower error rate than PointNet without the hierarchical structure. PointNet extracts features from the initial set, generates a feature vector before passing this back to the initial points to extract features globally, PointNet++ instead uses a hierarchical approach to generate sub groups of points to extract local features, and applies this repeatedly for global data set classification and segmentation.

2.4.3 Laser and Camera Fusion Networks

There has also been a significant amount of research focused on RGB-D camera based systems, camera sensors with a depth element taken by determining the distance between two corresponding images, a method which has seen substantial success. Tt may be possible to further improve this by utilising both a camera and laser sensor separately. There has been a number of proposals at combining a laser and camera sensor for this purpose, including "PointFusion for 3D Object Bounding Box Estimation" [74] [84], "Real-time probabilistic fusion of sparse 3D LiDAR and dense stereo", "Label Propagation from ImageNet to 3D Point Clouds" [85] and "Depth Aware CNN for RGB-D Segmentation" [86] for indoor environments. The most successful of these methods being LDLS [87] based on Mask-RCNN, capable of instance segmentation in outdoor environments. This method incorporates Mask-RCNN for images combines this with point cloud data obtained using a 3D laser scanner to generate an instance segmented 3D point cloud of an outdoor environment. This method

has proven successful at segmenting people, cars and pedestrians in a 3D point cloud with improved accuracy over previous methods.

2.5 3D Environment Reconstruction Techniques

A potential additional use of the sensor data obtained by the system in this research could be in generating a 3D reconstruction of its environment. There has been a number of methods for environment reconstruction, mainly focused around camera sensors such as RGB-D cameras, methods such as PanopticFusion [88] StaticFusion [89], DynaSLAM [90], MaskFusion [91], and ReFusion [92].

MaskFusion uses Mask R-CNN for semantic segmentation to assist in mapping and tracking multiple objects. The authors note that it is however limited to reconstruction, tracking and recognition as it can only recognise objects that the Mask R-CNN network had been trained on, not accounting for errors in classification, it also does not handle dynamic, moving or small objects efficiently, which can lead to lowering its performance.

ReFusion [92] is capable of localising to and reconstructing a static environment using an RGB-D camera. This method has proven to frequently surpass current state of the art systems in static environments and also has comparable results on dynamic environments. It is noted that dynamic environments are more challenging without a semantic segmentation algorithm available, the authors also provide a dataset used for their work.

2.6 Datasets for Agricultural Environments

The use of a semantic or instance segmentation network in an agricultural environment would be reliant on the dataset a network has been trained with for its classification accuracy. There are a number of labelled datasets available for image based training such as

the KITTI dataset, MSCOCO, NYU Depth and the TUM RGB-D Dataset.

There is however far less data available for 3D point cloud classification networks currently, with Semantic KITTI and Semantic3D being two of the main datasets available. The performance of a network would be dramatically effected if it were trained on images of buildings and cars for example, then used in an agricultural environment for fruit harvesting.

A systems robustness and ability to perform efficiently in this type of environment requires many different training examples, often specifically for the type of fruit bushes to be classified. Whilst this is becoming more widely available for image based classification, it is still a time consuming problem and more so for a point cloud based network.

A potential solution to this could be through synthetic data. TensorFlow, an open source machine learning library enables the ability to synthesise images that can then be used as training data, for example, if 1000 images of a specific type of fruit bush in an agricultural environment were taken, these images can be transformed, moving an object in an image to different locations in the image, in this example, 1000 images could be used to generate 4000 images, this could potentially be useful for this system.

2.7 Research Novelty

The system proposed in this research is designed to operate in an indoor vertical growing facility, with no rigid ground based structure nor guarantee of GPS being available.

The accuracy of GPS if available would also be insufficient for the system to move between each narrow row as it finishes harvesting in each of them or to move along each row to autonomously harvest the fruit in hanging baskets of varying and frequently changing height.

It will also be required to operate entirely without assistance from a human, as such remote controlled methods would not be applicable.

The system will be comprised of a custom built robotic base with two robotic arms and multiple sensors as shown in Fig.2.9. Additional software modules such as the navigation and control system, sensor repositioning throughout the systems development or additional component integration may be required, as such, a modular and configurable architecture will be required. This would also enable generalisation of the system to different crop harvesting and environments. This will also ensure rapid prototyping and evaluation in a laboratory and agricultural environment will be possible, without the need to redesign the entire system if a new component is added or removed, as such a novel system architecture capable of these requirements is needed for the system.

The system once developed may also be capable of additional functionality such as metric collection to determine potential harvest yield, fruit quality and environment monitoring. The data required for this is often obtained as navigation data, with it potentially possible to use this data for this additional functionality, using classification techniques such as instance segmentation and a 3D reconstruction method.

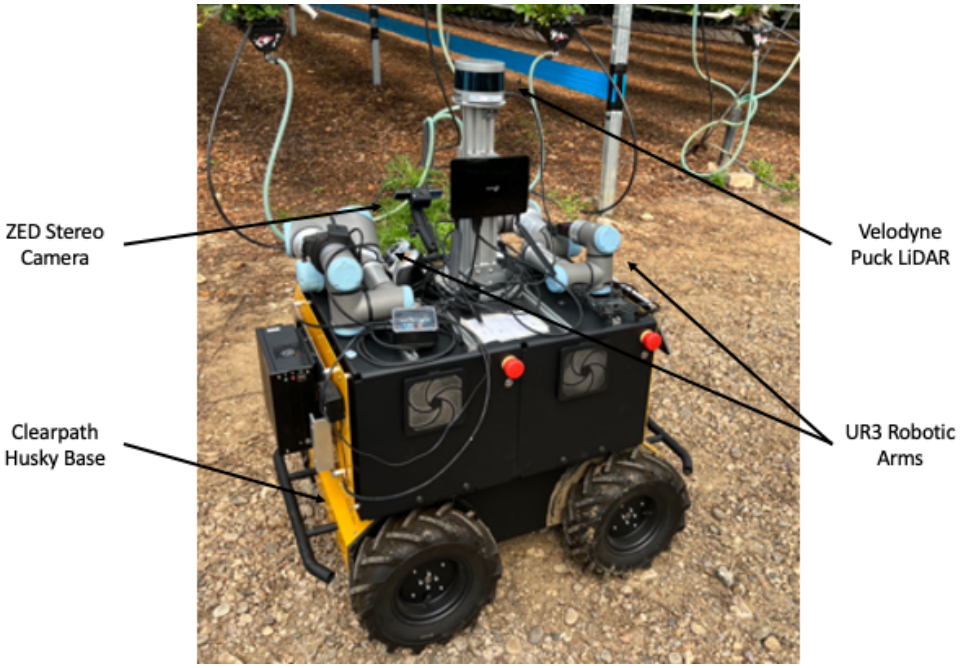

Figure 2.9: Robotic System for this Research.

Chapter 3

Configurable Software Architecture for an Agricultural Robot

3.1 Introduction

The agricultural sector has seen considerable interest in the use of robotics and intelligent systems over recent years, with advancements in the capabilities and reduced cost of unmanned ground vehicles and their peripherals, such as LiDAR and stereo camera sensors and robotic manipulator arms. With these advancements, many new applications are seeing active research and becoming commercially viable. There is nonetheless still much research and development required in this area for the realisation of autonomous agricultural robotic systems. The agricultural industry may benefit dramatically from the use of such technologies, this becoming increasingly of interest and importance with national and global issues such as labour shortages and climate change, reducing viable agricultural land and rising populace demand. An application of robotics in agriculture is that of autonomous fruit harvesting, an often time consuming and laborious task. An autonomous solution for this

task has the potential to greatly assist with these challenges in both reducing waste whilst increasing crop yield, alleviating labour shortages through harvesting twenty four hours a day, whilst facilitating new methods of agriculture such as vertical farming to utilize available land as efficiently as possible. Robotic systems are often associated with laboratory or factory settings such as warehouses, automated production and assembly lines, commercial settings such as drone delivery services and specialised uses in the medical or space sectors. These environments are often structured and purpose built for these systems. For use in an agricultural setting a system would be required to be adaptable and robust, able to tolerate environmental conditions typical of agricultural settings such as soil and dust, whilst maintaining precision in prolonged, repetitive tasks and in varying terrain.

This chapter introduces a novel software architecture, describing the design, development and initial evaluation for a configurable, modular robotic system capable of autonomously harvesting strawberries in a vertical growing facility. The following sections first establishes the hardware the system is comprised of, followed by a description of the software architectural design, development and implementation. The section is concluded with the modular software implementation description and the initial experimentation of the system as a proof of concept for both navigation and control capabilities.

3.1.1 Objectives and Contributions

The objectives and contributions of this chapter are the design, development and implementation of a configurable, modular software architecture for an agricultural robotic fruit harvesting system.

This chapter includes:

- A modular, configurable software architecture, a conceptual and practical architecture for autonomous strawberry harvesting and agricultural fruit harvesting.

- A multi-robot system implementation, a method of combining multiple robotic systems to function together.

- An initial evaluation of the system as a proof of concept in a laboratory setting.

- SLAM experimentation in the laboratory and in the vertical farm, to determine the limitations of the SLAM algorithm in an agricultural environment, and prove the requirements of local positional control for fruit harvesting.

These objectives presented numerous challenges, with the system being required to operate in a dynamic, unstructured, natural environment, utilizing multiple sensors and components operating in parallel, whilst incorporating a perception system for detecting the fruit to harvest.

3.2 Hardware Description

The robot is entirely custom built, the robotic arms and base being pre-assembled with additional components specifically for the task of strawberry harvesting in a vertical growing facility added to the system.

An overview of the hardware components of the system can be seen in Fig. 3.1, categorised into three main aspects, the robotic base, robotic arms and sensors. These individual components communicate together to provide the systems functionality, for example the sensor data obtained by the stereo camera is used by the robotic arms and if required, the robotic base to move to and harvest a strawberry as shown in Fig. 3.1.

The configuration of the system's hardware is modular by design and can be modified and changed with sensors able to be added, replaced or removed dependent on the requirements or environment the robot is operating in.

This modular design has the additional capability of being adaptable to additional harvesting requirements with the capability of changing sensors and components if required, without having to redesign the entire system. The components of the system are detailed in the following sections.

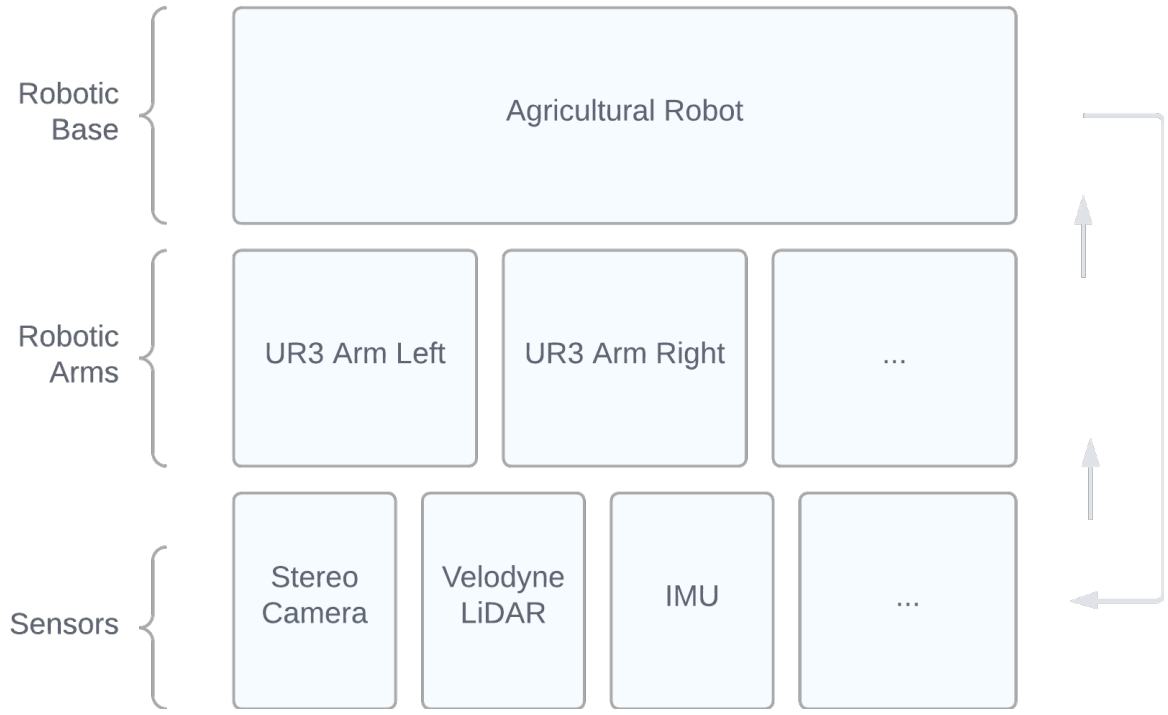

Figure 3.1: Overview of System Components.

3.2.1 Robotic Base

The robotic base is a custom built Clearpath Husky Unmanned Ground Vehicle as shown in Fig.3.2. The base unit is an outdoor all terrain field research robot, with a base plate size of 21.4in x 16.5in, a maximum payload of 70kg and speed of 1.0 m/s.

It is capable of carrying up to four robotic arms, a variety of sensors and has an internal

power supply enabling the system to be entirely wireless and self contained. The robot is compatible with Robot Operating System (ROS) and able to operate in various terrain making this an ideal prototyping platform for an agricultural robot. [4]

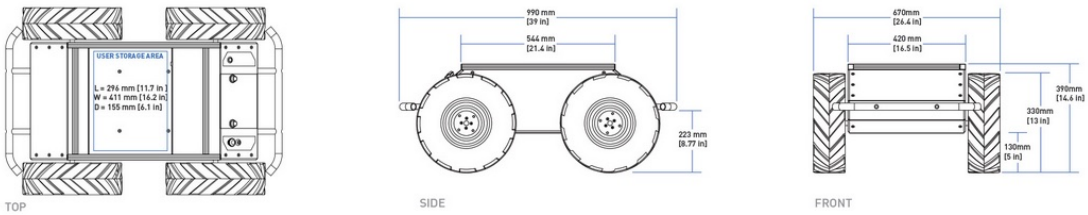

Figure 3.2: Husky Robotic Base Schematic. [4]

3.2.2 UR3 Robotic Arms

The robotic base was custom built to accommodate two universal robots UR3 robotic arms, Fig.3.3, these being pre-installed onto the top plate of the base.

The arms each have customised end effectors, designed for harvesting strawberries, with a total weight of 11.2kg each and a maximum payload of 3kg. They each have 6 degrees of freedom, with each joint capable of 360 degree rotation, except the wrist joint, this capable of infinite rotation. The base, shoulder and elbow joints are capable of a maximum rotational speed of 180 degrees per second, with the three wrist joints capable of up to 360 degrees per second.

This gives the robotic arms considerable flexibility and maneuverability to reach at numerous angles, ideal for this task where the strawberry stems may require the arms to maneuver into many different angles to be harvest-able.

Figure 3.3: UR3 Robotic Arm.

3.2.3 Components and Sensors

The robot as standard is capable of providing odometry readings through encoders in each of the four wheel motors, with 78,000 measurements per rotation. The additional sensors for perception and navigation required a platform for installation, this above the top plate of the robotic base to ensure they have the required height for their application and to facilitate the number of sensors required. In the center of the robot, a custom built mount was installed as a platform for the sensors to accommodate this, capable of mounting numerous sensors as required as shown in Fig. 3.4. The robot is equipped with three main additional sensors,

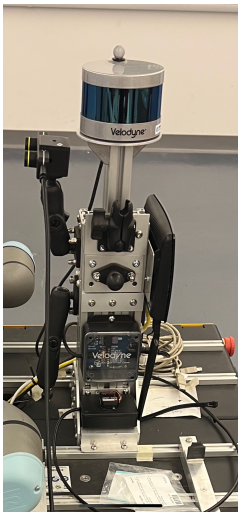

Figure 3.4: UGV Sensor Mount.

a Velodyne Puck 3D LiDAR sensor, a Redshift Labs UM7 inertial measurement unit and a ZED Mini stereo camera. A network switch was installed to connect the robotic arms and LiDAR sensor, with a router added as a wireless access point for the system, this capable of both Ethernet and wireless connectivity. Finally, the battery was replaced with a higher capacity variant to increase operational time with the additional sensors and components, this giving the system a battery life of 3-4 hours with this configuration. The full robotic system can be seen in Fig.3.5. The system's hardware components then required a method

Figure 3.5: ClearPath Husky UGV Custom Built.

of integration to be developed for their functionality, this solution needing to be modular and configurable to ensure the components are able to function together. The following section details the software architectural solution for this.

3.3 Configurable Software Architecture

The system's components required a novel software architecture to be designed for their integration and operation. As the system was a prototype to determine and evaluate the viability of automated robotic fruit harvesting, a modular, configurable architectural design was decided upon to enable rapid component addition, integration, movement and replacement.

The systems architecture design had a number of requirements, for example hardware components such as additional sensors may have been required to be added or replaced in the system, with functionality added for each component.

The position of the sensors may also have required moving through the systems development, for example the camera sensors position to ensure it did not obstruct or conflict with the robotic arms. The configurable architectural design facilitated these requirements, enabling rapid and dynamic changes to be made without the system overall needing to be redesigned.

The sensors and components of the system once installed required corresponding software modules to enable their functionality, this enabled through the software architecture design. The modular design of the system enabled these software components to be integrated, as well as functionality such as navigational capabilities and arm functionality to be added and removed without having to rebuild the system with each iteration. This design enabled rapid testing and adaptation of the system.

The software architecture's modular design, and through using ROS to integrate the individual components, enabled the system to support additional sensors and components with software written in multiple languages, such as Python, C and C++, this giving further flexibility to the system and its capabilities. The architecture is separated into four core components

with each containing multiple modules, the system overall being controlled through a user interface as shown in Fig. 3.6.

These main components are the Navigation Control, UR3 Arm Control and the Perception System, with the inter-operable functionality of the robotic base, arms and perception system combined in the Navigation and Control module of the system.

The main aspects of the system are loosely coupled, requiring only coordinates from the perception system as an input, this further facilitating the capability of components being able to be added or removed. For example the perception system can be replaced with another version trained for a different fruit, another arm could be connected to the system or a different navigation method.

3.3.1 Perception System

The perception system used for fruit detection was an existing system developed for this project. The systems architecture, arm control and navigation and control were developed to be integrated with the perception system, this used to provide the coordinates of the fruit to be harvested.

The perception system provides X, Y and Z coordinates to a network connected to each robotic arm, each network then calculating the joint angles required to harvest the fruit. These joint angles are then passed as an input to each robotic arm, these inputs then used for navigation control.

Figure 3.6: Software Architecture Overview.

3.3.2 Robotic Arm Control and Navigation Modules

The robotic arm control and navigation and control modules control the motion of the robot base and arms. The navigation and control class combines the functionality of the UR3 arm control, navigation control and perception system modules through multiprocessing, with the system overall controlled through a user interface.

The robot modules are loosely coupled to enable the addition and removal of components,

with the perception system providing coordinates for the fruit harvesting, these being the basis for the navigation control. With this modular design, the robotic arms, base or sensors can function independently of each other, for example if the left arm only is in range of a detected fruit, it would be able to harvest it or if the base is required to move to a new position to reach the detected fruit it can do so independent of the robotic arms.

The system, when the core components are interlinked, is designed to be led from the left arm, the coordinates from the perception system are passed to the left arm control, this leading the system with the left arm as the primary component for the robots direction and movement.

The right arm functions independently as a separate module, and is able to operate when it is in range of a fruit to be harvested, but will not overrule the left arm, this being the only difference between the left arm module and the right, or any additional arm module, with this functionality removed from the right arm.

The arm functionality is combined and controlled in the UR3 Arm control module, enabling additional arms to be connected, without having to redesign the system. The different functionality of the system is controlled through a user interface, enabling a user to access both independent module functionality, such as navigation control or each arm, as well as to begin and exit autonomous operation, and for manual control of the robotic base.

3.3.3 Module Integration and Multiprocessing

The modules of the system are individual, loosely coupled sets of functions for each component, for example the left and right robotic arms. All functionality of a module is contained within its respective class as one modular component for example with the left arms functions, except its additional control of the robotic base. These functions could be applied to

any number of robotic arms and will have the same functionality. The only dependence the class has is in receiving coordinates and joint angles to move to, these acquired from the perception system. The navigation and all other modules of the system are designed in this manner. The navigation class can function entirely independently or take an input from the perception system as a target to move to. The individual module functionality is combined with the rest of the system through the navigation and control class. This class contains multiple multi-threaded functions designed for different uses of the robot. The module integration overview is shown in Fig. 3.7.

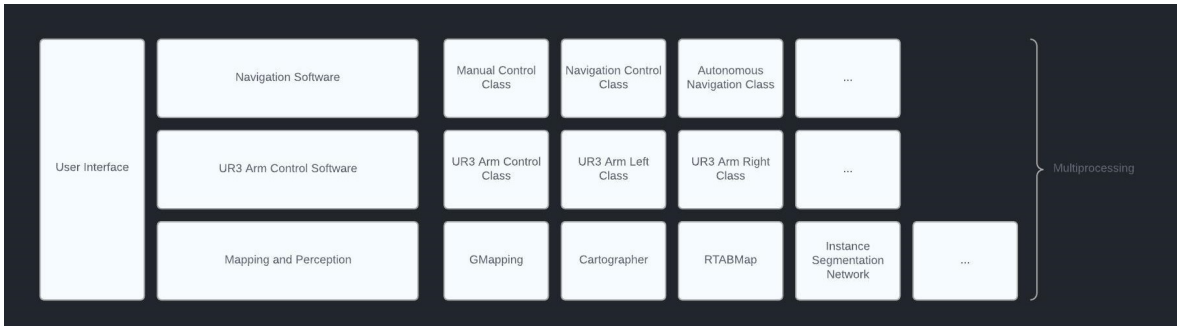

Figure 3.7: Module Integration.

3.4 Software Implementation

The software for the robot is comprised of multiple, independent modules as described, with each able to operate as an entirely stand alone function, or as part of the system overall, with additional functionality added as the modules are linked together.

This modularity is facilitated through ROS, using its transformation tree structure, Unified Robot Description Format (URDF) and launch files. The URDF and launch file composition, transformation tree and abstract module design is described in the following section.

3.4.1 ROS Node Structure, URDF and Launch Files

The navigation system is built using the Robot Operating System (ROS) software libraries. The robot being entirely custom built meant a new description of itself was developed and introduced to link each component and sensor in relation to the rest of the robot, this developed using ROS's transformation tree system.

This is required as each component and sensor would be considered to be the center of its respective transformation without a system to control this. The description ensures that each component transformation is following the same coordinate system as the robot moves, this through ROS's coordinate frames system.

The ROS system uses Unified Robot Description Format (URDF) files to create a virtual representation of a given system, these URDF files comprised of simulated visual components of a system, and enable each components position in a system to be defined, for example where the robotic arms are in relation to the rest of the robot, or where a sensor is mounted. The system operates with two coordinates systems, for the robotic base and arms, these at a 90 degree angle horizontally to each other and the robotic arms being at a 45 degree angle vertically to the rest of the robot. The URDF and subsequent transformation tree enables both systems to operate in the same coordinate frame. The ROS URDF file system enables a description of the robot to be written as a transformation reference system for the robot. This enables the sensors to operate around one single point of reference as the robot moves. This is useful for robotic systems as it enables multiple sensors to be added and to operate together in one coordinate system. The sensors can be defined in the URDF file including their location and orientation on the robot, from there when the robot moves. The sensors are transformed around a central defined point this titled as the base frame.

The launch file system enables all of the required components of the software to be launched simultaneously when the robot is powered on, including all components, sensors and configuration files. This system required a custom launch file to include the control configuration file, with ROS using Yet Another Markup Language (yaml) files for these configuration files. The control yaml file is a configuration file used to set the initial parameters of the robot, this launched through the control launch file, included with the main launch files when the system is started. The control file initializes the robotic base's initial parameters, for example the base frame of the transformation tree and the speed parameters of the robot. The configuration files can be used to set parameters for multiple components of the system, including the GMapping SLAM algorithm used in the systems evaluation, the results of this shown further in this chapter.

The individual sensors of the robot each have a corresponding launch file, collated into a single main system launch file, called when the system is started. This ensures all of the components of the system are launched when the system is powered on. The system was developed using both a simulated and the real system. There were two simulated versions developed, using ROS's RViz visualization software and the Gazebo simulation software. The first simulated version of the robot was the robotic base with a Velodyne Puck LiDAR sensor, as shown in Fig. 3.8 and 3.9, with and without the sensor mount. This was for developing the navigation software in a simulated environment. The second simulated version was a separate URDF and configuration file with a simulated version of the system incorporating the robotic arms. This was developed initially to test the robotic arm control in a simulated environment. This was not required and not used as the functionality of the robotic arms was tested instead in a real world environment, separate to the navigation system using the

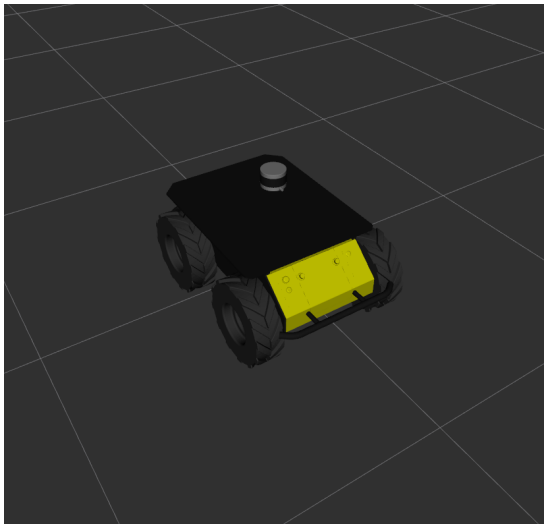

Figure 3.8: Simulated version of the Robot with Velodyne LiDAR Sensor.

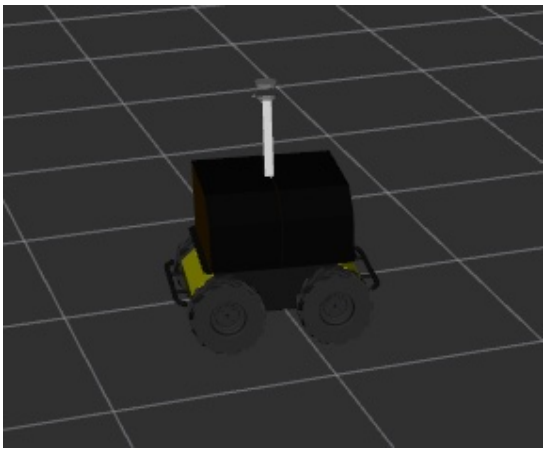

Figure 3.9: Simulated version of the Robot with the Custom Sensor Mount and Velodyne LiDAR Sensor.

perception system as the coordinate inputs.

3.4.2 Transformation Tree and Coordinate Frames

A transformation tree in ROS represents a coordinate system for all components in a robot. When the robot moves, all of the robots components are then transformed around a single point of reference, removing the need to keep track of each component. This system also enables rapid customisation for additional sensors and peripherals such as adding additional sensors, and enables multiple robots to operate in the same world environment using multiple maps. The transformation tree representing the components of this robot is shown in Fig.3.10, The coordinate system follows the ROS convention of:

- Earth Link:

  A global link that can enable multiple robots to operate using multiple maps.

- Map Link:

  A global frame of reference generated by the robots laser scanner.

- Odom Link:

  A local frame of reference obtained through the robots IMU.

- Base Link:

  A frame of reference related directly to the robot, a point on the robot to which all other components are linked.

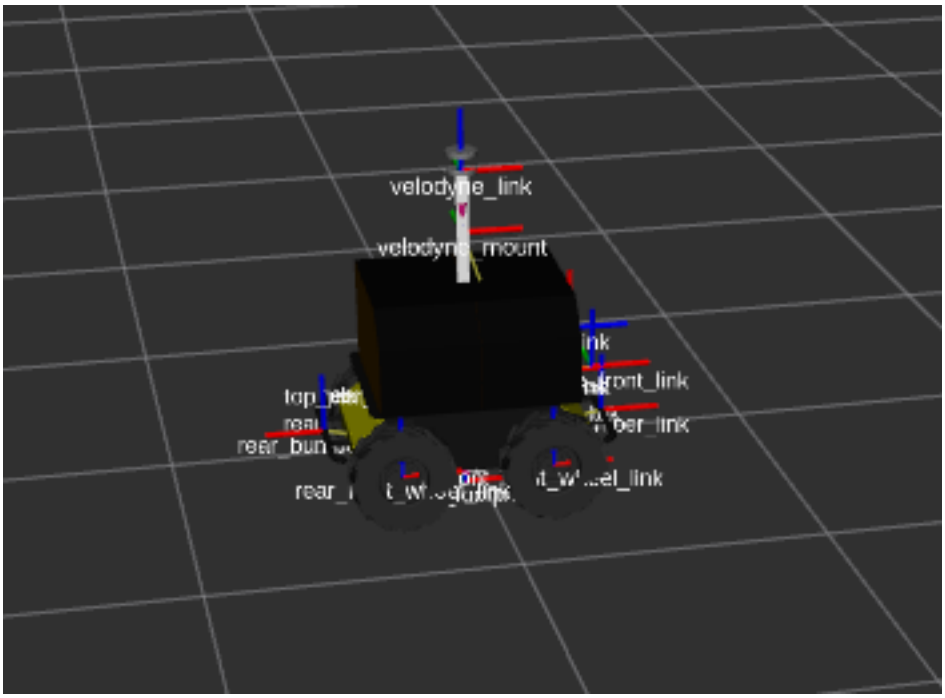

Figure 3.10: Visual Representation of the Robot with Transformation Links. The individual links of the system ensure the robot operates in the same coordinate frame.

The relevant links in the transformation tree for this robot can be see in detail in Fig. 3.11 and Fig. 3.12. The base link is the initial point of the transformation tree and linked to the center of the top plate of the mobile base. The wheels and the majority of the structural components of the system are linked below the base link, with the front and rear bulkheads linked to this. The velodyne mount is then linked to the center of the front bulkhead, with the Velodyne sensor and IMU linked to the velodyne mount. The robot and its sensors are initiated by a launch file when the robot is started, the custom description is passed to the

control launch file to initialize the connected sensors and the base launch file to initialize the required ROS nodes when the robot is started, giving it at an accurate representation of itself. The navigation software was developed initially using a simulated farm environment for the basic functionality before being tested and developed in a laboratory environment and the agricultural environment.

3.4.3 Software Component Integration

The navigation and control class contains the methods to combine the functionality of the system, for example the robotic base and left arm movement. In this class, there are various functions for the parallel operation of components in the system. Parallel operation in this context refers to the simultaneous operation of components, this through multiprocessing, using a CPU core for each operation, as opposed to sharing the computational load across multiple cores. The navigation and control class combines the separate modules of the system and passes these as functions to the user interface. The process threads from each module are called in this class in their respective functions. This modularity enables the base, one or both arms to function independently or in parallel, dependent on the robots operational requirements. With each robotic manipulator being controlled by a separate network and the network determining the joint angles based upon input from the perception system. The navigation control is designed to keep the system in the optimal position for the left arm to harvest the fruit, with the robotic base moving the system dependent on the coordinates provided for the left arm only to ensure this. The right arm will function provided it is in the correct position, but the base will not move to ensure the optimal position for the right arm. This is to ensure the coordinates sent to each arm do not conflict, resulting in the robotic base attempting to move into the optimal position for the left and right arms simultaneously.

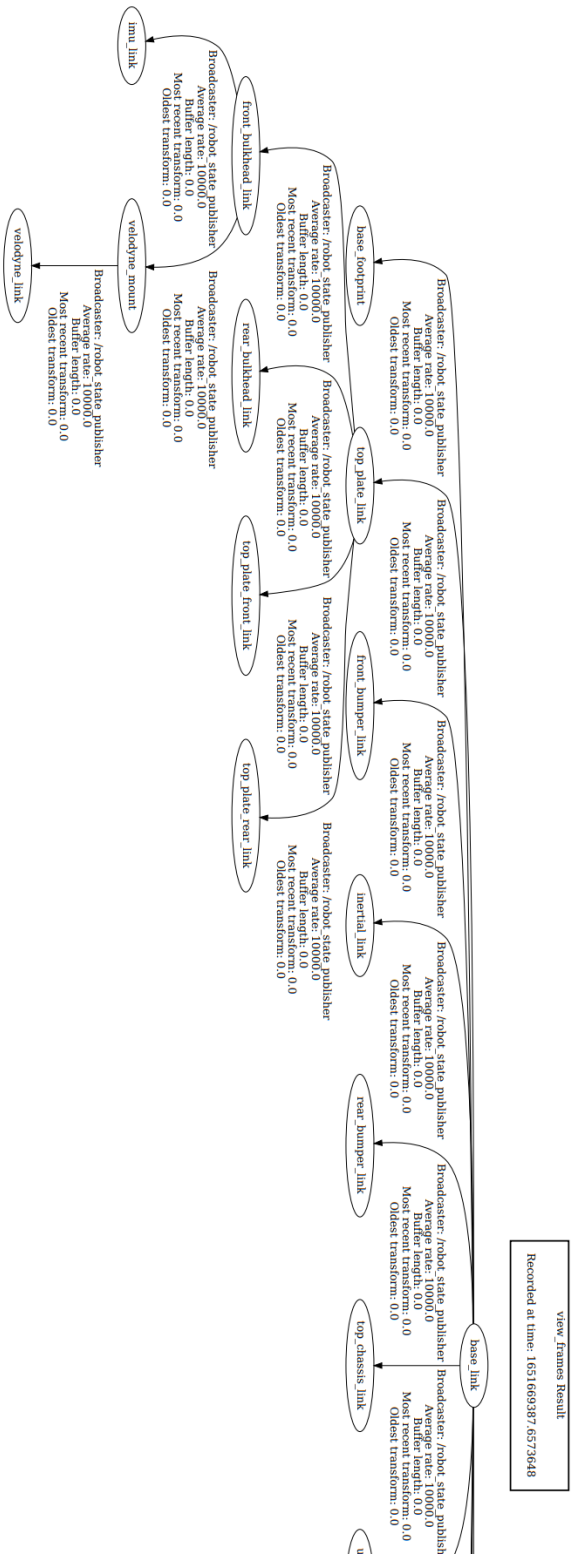

Figure 3.11: The transformation tree of the robot shows the individual links between the systems components. The Velodyne for example is linked to the Velodyne mount, this linked to the front bulkhead before the top plate, then the base link.

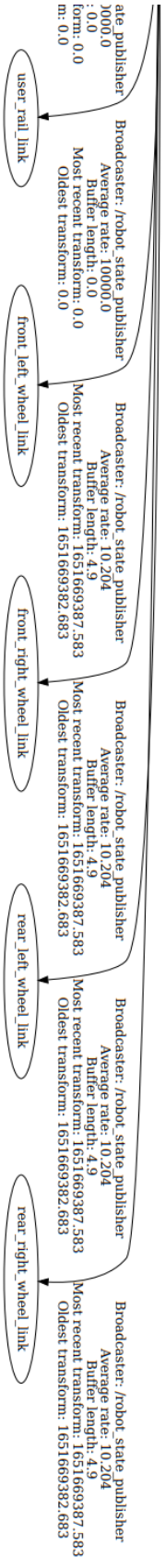

Figure 3.12: Transformation tree detailing the robotic base wheel links.

The LiDAR sensor and all other sensors the system contains follow this modular approach, to connect a sensor and module to the system would require the following:

- 1. Connect the required sensor to the system.
- 2. Add this sensor to the URDF file, including defining its position in x, y, z coordinates for the transformation tree.
- 3. Add the sensor launch file to the main robotic system launch file.
- 4. Add the sensor script to its respective module to take an input or give an output.

The system could have a separate additional camera sensor mounted parallel to the current sensor on the robotic base, with two additional robotic arms mounted parallel, and these would be able to function. An abstract view of this can be described as, the limit of the number of sensors and components is the size of the base plate and power supply required. The global SLAM software uses the laser scanner to generate a 2D topological map of the environment for autonomous navigation throughout the facility. The GMapping algorithm was used as as a proof of concept for the navigation system, to determine its capability and viability in, at first a laboratory setting before an agricultural environment.

3.4.4 UI and System Control

The user interface enables control of the robot, with options for all functionality of the system. This includes manual and automated control of the robotic base, the control of each arm individually, controlling each gripper and launching the perception system. The user interface is terminal based, with a graphical user interface considered that could be used through a tablet. An overview of the user interface is shown in Fig. 3.13.

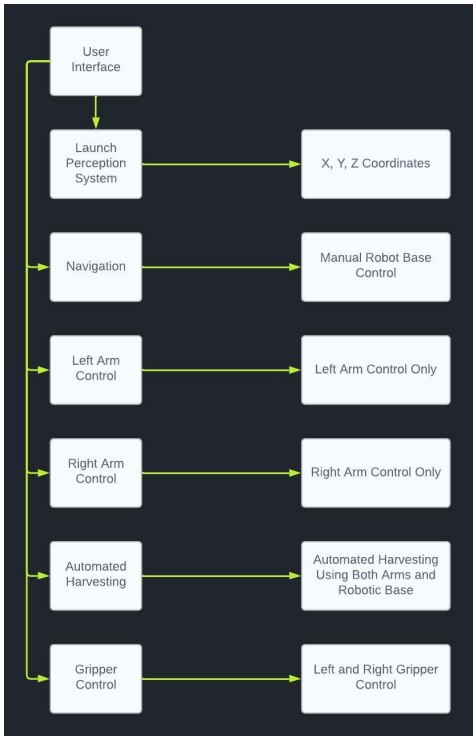

Figure 3.13: User Interface.

3.5 Initial Proof of Concept and Evaluation

The initial proof of concept of the system was to determine the software architecture functions as expected, enabling all of the integrated components to operate, and to determine through experimentation the viability of a SLAM algorithm such as GMapping to be used as part of a navigation solution, in this type of environment. There are a number of existing potential solutions that would fundamentally not be viable for this application. A robots basic navigational capabilities, such as wheel odometry only would be far too inaccurate over prolonged distances and GPS would be unusable in this setting as the robot is required to operate on uneven ground and in a partially indoor environment. The use of odometry would be too inaccurate, with drift and slippage accumulating significant errors in a large environment, and GPS again not giving the accuracy required to maintain a specific distance

from the fruit bushes. The bushes are also naturally uneven, a straight path through the environment would also not give the required accuracy throughout the environment. The robot is required to move throughout the farm environment and identify specific rows to traverse between whilst maintaining a constant distance between the robotic arms and the fruit plants, this whilst ensuring it does not damage the fruit or lose its position to ensure the perception system can operate. This requires a novel navigation system to be developed based on both global and local path planning requirements. A global requirement is to ensure the robot travels to the correct row, this may be methodically, one after another or decided ahead of time by a drone or other robotic system at another location in the global environment. In order to navigate to the required location, the system must constantly update its perception of the area as the vertical hanging baskets frequently move position whilst it is operating and the density of plants in the baskets also frequently changes, this meaning a pre-existing map may not accurately reflect the current environment. When the vertical baskets move, areas below them previously occupied are no longer.

3.5.1 Natural and Artificial Beacons for Path Planning and SLAM

Natural and artificial beacons may also have been an option for this task, they would again present their own challenges in this setting as the environment, although topologically reasonably static, the interior of the environment by its nature changes frequently, with the vertical bushes being of varying density's in the environment. Using a beacon based approach would also require considerable maintenance to ensure they are visible at all times and from all required view points for the robot to accurately maintain its position in each row of the vertical bushes. The option of using a SLAM algorithm for this objective, at first may seem to be suitable system. A 2D SLAM method is used as part of this system for the

topological map of the environment, but it would not give the required accuracy in localised regions of the setting. A 3D SLAM approach, although able to map localised regions with reasonable precision, would require the laser or camera sensor used to be mounted in a way that the fruit bushes do not distort its view. If the sensor is positioned too high or low on the robot, it would not be able to determine the position required, or may give incorrect distance measurements rendering the system unable to complete its objective. A view of the vertical growing facility is shown in Fig. 3.14. The navigation system is required to ensure the

Figure 3.14: Vertical Farm

robot keeps within a set distance of the plants in order for the perception system to operate. If the robot moves too far from the fruit or too close, it may lose its view and be unable to function, or damage the fruit or growing system itself. To ensure the robot stays within these parameters, a novel solution for local path planning and following is also a requirement. With the fruit plants in the environment between each row varying in density, as such the laser sensor was tested in between a row to determine if it was capable of detecting the plants whilst travelling through the row. The navigation system and robotic arms are also required to be dynamic and adaptable to obstacles in the environment. The farming facility having many employees, potentially other robotic systems and the vertical hanging baskets

changing position, requires the robot to be able to detect and avoid these obstacles as they move, or to ensure the robotic arms do not collide with the vertical hanging baskets or other obstacles. The initial experiments conducted were to ensure the robot base and arms functioned in parallel, the laser scanner and inertial measurement unit added to the robot were configured correctly, and that the SLAM algorithm was capable of mapping the laboratory, before then using the robot in an agricultural setting.

3.5.2 Simulation and Laboratory Experimentation

The robot base was simulated using the RViz simulation software and ROS, this for the navigation controllers development using the LiDAR sensor as shown in Fig. 3.15.
This allowed for the controller to be developed and experimented with for the mobile base in a laboratory setting before being evaluated in the agricultural facility. The robots description developed was used to give an accurate representation of the actual robot in the simulation.

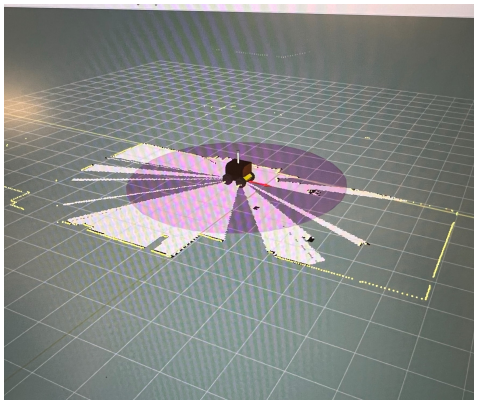

Figure 3.15: Velodyne Lab Outline

The initial laboratory experiments proved to be successful and the system functioned as required, with both arms and the robot base operating in parallel, and the sensors and robot description all functioning correctly. The robot navigational capabilities were then evaluated in the agricultural environment using the GMapping SLAM algorithm.

The experiments for the robotic system in a laboratory were to evaluate the system overall before experimentation in the agricultural environment. The software required both arms and the mobile base to function in parallel, this was at first ensured in a controlled environment. The agricultural environment can be simulated by adding a horizontal row of fruit bushes, to evaluate its ability to operate with both arms and the mobile base in parallel and to determine the systems configuration including all sensors.

This initial experimentation data can also be compared against standard odometry, beacon based and a SLAM method of navigation and experimental results obtained from the agricultural facility.

The initial evaluation of the Gmapping SLAM algorithm can be seen in Fig.3.16, Fig.3.17, this proved that the laser sensor had been configured correctly for 2D laser scanning and functioned with the system overall.

The parallel functionality of the robotic base and arms were evaluated, with the base and each arm able to function independently and in parallel.

Each sensor was also checked to ensure they were functioning, along with basic movement commands for the robot, including its ability to navigate to a given coordinate. The system in its entirety proved to be successfully integrated.

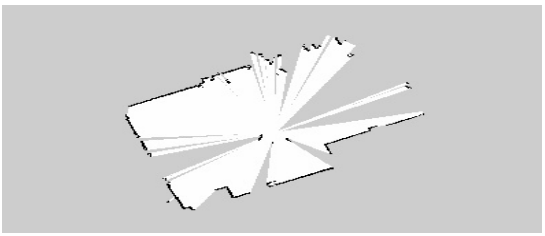

Figure 3.16: Velodyne Lab Outline

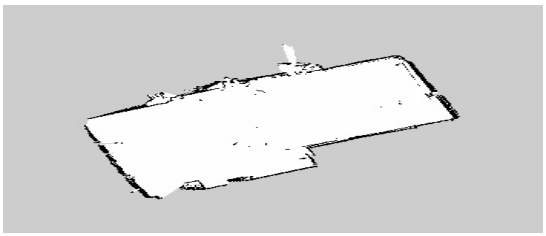

Figure 3.17: Velodyne Lab Outline

3.5.3 Mapping and Navigation in an Agricultural Environment

The experiments conducted were to evaluate the system's mapping capabilities in a large, open environment. The facility is 40m x 40m, with translucent boundaries which could have effected the laser sensors ability to determine them.

The facility is also mostly empty other than the vertical hanging baskets. This can also effect the mapping capability in a large environment as the system would have few environmental objects to detect as it moves assisting in the map generation.

This can be seen in Fig.3.18, Fig.3.19. The area mapped in this evaluation was 10m x 10m, the boundary detected in Fig.3.18, being a line of tape covering the majority of one row of the fruit bushes.

The system was able to detect the boundary at the edge of the facility relatively well despite the translucency, but as expected encountered significant drift whilst moving over this distance.

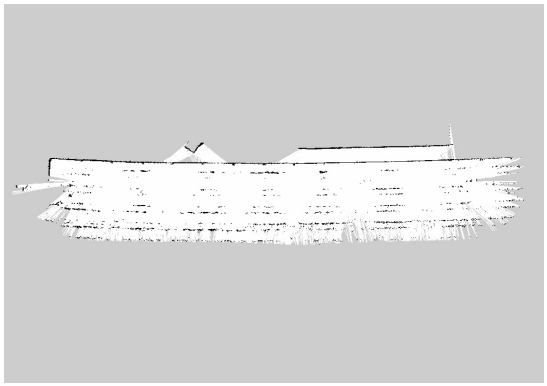

Figure 3.18: Facility Boundary

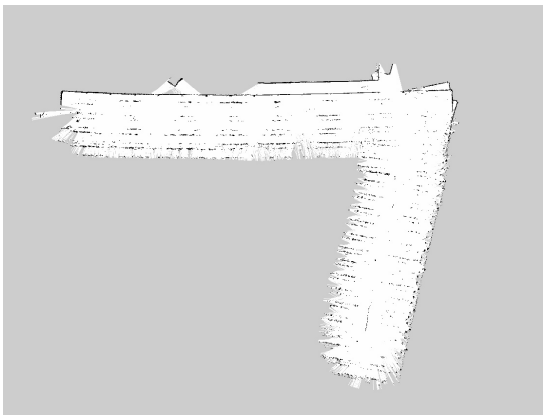

Figure 3.19: Facility Boundary

The system was then evaluated in determining and traversing rows in the environment. The laser was able to detect the rows but as seen in Fig.3.20, Fig.3.21, the laser sensor can not entirely detect the fruit bushes as a boundary.

This is because they are not very dense and this is not precise enough to enable the robotic arms functionality with centimetre precision. The robot also encountered significant drift when traversing the rows as with the general environment, this because of the terrain.

This drift was apparent within the first 1-3 metres of the robot bases movement, identifying further a need for a feedback control method of localised path planning to correct for this drift to ensure the robot is both in operating range of the fruit bushes and to ensure it does not damage the vertical hanging baskets. With this experiment the robot began at the start of the row and traversed 20m to the end and back.

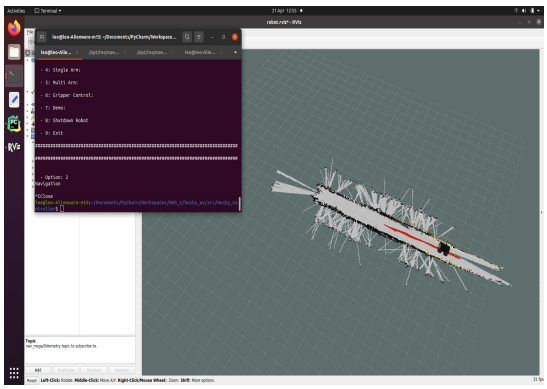

Figure 3.20: Navigating a row 1.

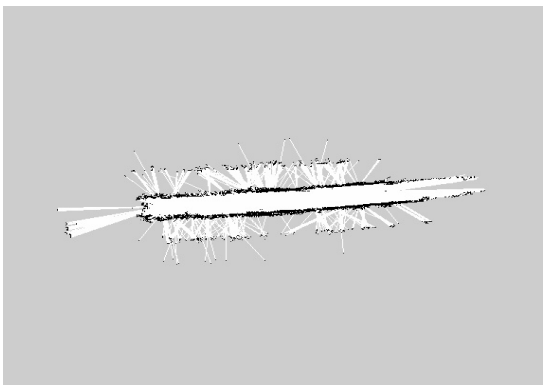

Figure 3.21: Navigating a row 2.

As can be seen in Fig.3.22, Fig.3.23, the robot as it reached the end of the row and traversed to the second was able to detect this, but within 1m of traversing the row, the boundary density was too transparent and the GMapping algorithm merged to the first row.

Whilst the robot traversed the row, GPS data was also collected to determine its accuracy in the facility as shown in Fig.3.24. Whilst traversing each row in one direct, GPS was able to track its location with reasonable accuracy, but when the robot turned around and traversed back it quite dramatically lost its location.

This proves that GPS would not be accurate enough for this application as the arms are required to be within centimetre accuracy of the fruit bushes, and also to ensure the robot

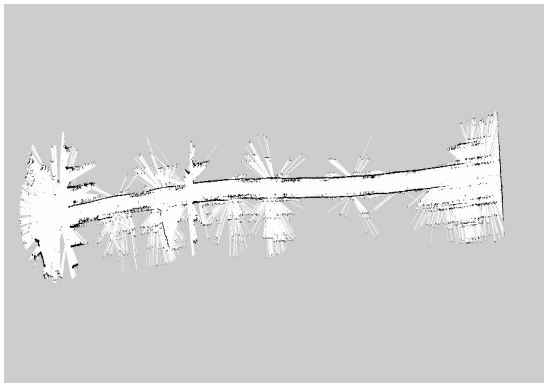

Figure 3.22: Navigating a row 3.

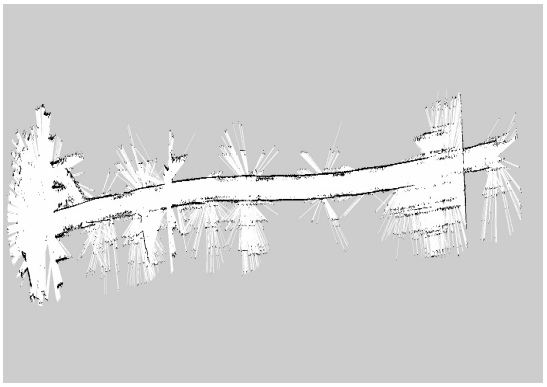

Figure 3.23: Navigating a row 4.

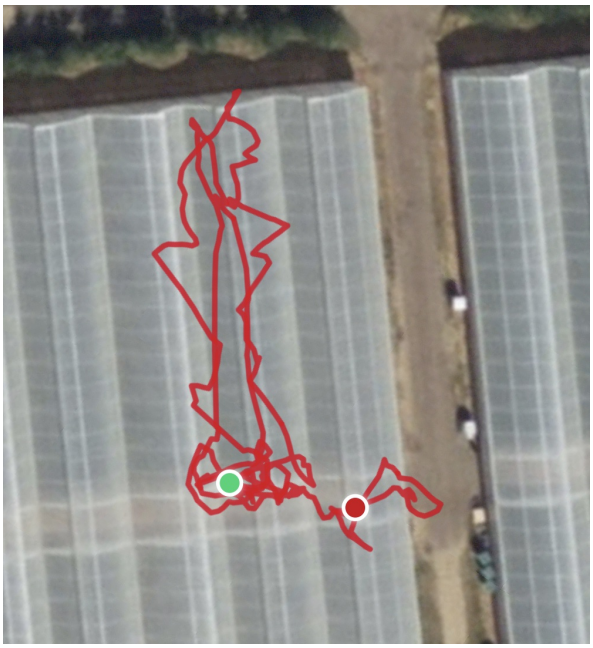

Figure 3.24: Overlay of GPS with Robot Navigation.

does not damage the facility. The system was evaluated again, this time ignoring a row to assist the SLAM algorithm in determining a second row as shown in Fig.3.25. This enabling a second row to be determined, but the drift was also dramatic in this experiment. The rows were detected relatively well, but the drift can be seen quite dramatically in each row as each is a straight line to the end of the facility.

As can be seen in Fig.3.25, Fig.3.26, Fig.3.27, Fig.3.28, the SLAM algorithm also shifted the map forward part way through the first row.

This shows the first row as longer than the second although they are equal length. This could be because of inaccurate sensory information from the odometry, IMU or laser, or a limit to the SLAM algorithms capabilities in this type of environment. The algorithm was able to determine the entrance to each row it passed as shown in Fig.3.27, and again able to determine the boundary of each row relatively well given the type of environment and the accumulating error as it traversed each row.

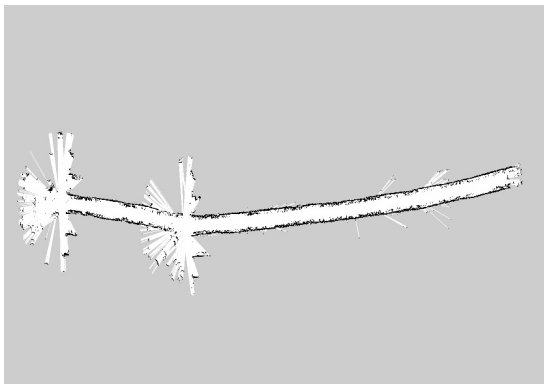

Figure 3.25: Farm Row Navigation 1.

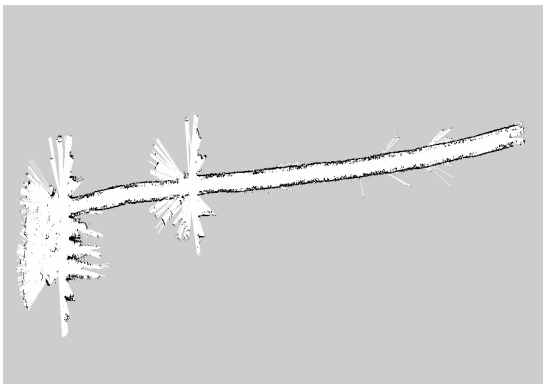

Figure 3.26: Farm Row Navigation 2.

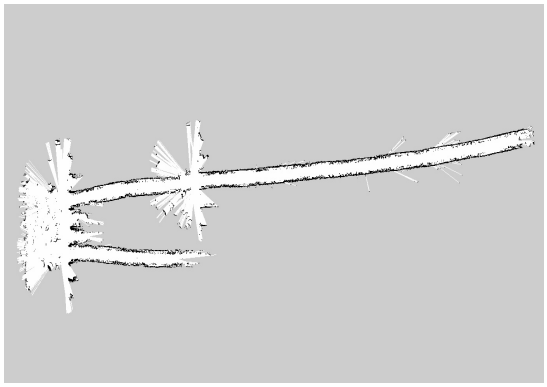

Figure 3.27: Farm Row Navigation 3.

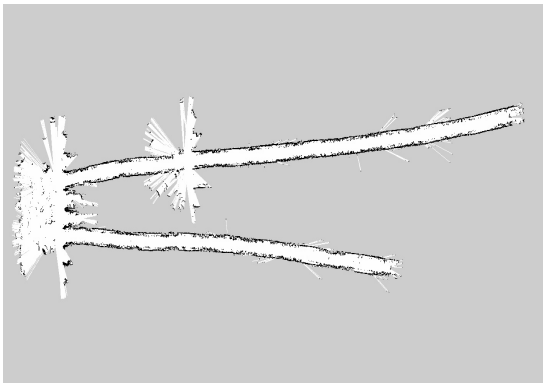

Figure 3.28: Farm Row Navigation 4.

The GMapping SLAM algorithm was capable of determining both row entrances and the rows as it traversed them reasonably well in the experiments conducted, but the algorithm as shown in Fig.3.25 and Fig.3.26, was incapable of determining the row boundaries with the required accuracy for the system. As can be seen in Fig.3.27 and 3.28, the boundary of a row

of fruit is detected and reasonably distinguishable, however the system would be required to traverse the row without colliding with or damaging either the fruit or infrastructure. The granularity of the fruit bushes as detected by the algorithm did not provide a sufficient level of detail to ensure the system would be capable of avoiding this, with the actual fruit bushes not distinguishable by the algorithm.

The robot also encountered considerable drift and accumulating error whilst traversing the rows and the environment with this type of terrain. This experimentation with the GMapping SLAM algorithm and the robot controller not having any form of error correction, along with the GPS accuracy, proved a navigation controller capable of accurate path following would be required.

3.6 Conclusion

This chapter demonstrates the design, implementation and initial experimentation and evaluation of a novel modular, configurable software architecture for an autonomous robotic fruit harvesting system. The proposed configurable software architecture is used for strawberry harvesting in this chapter, but could be generalised to other complex robotic systems. Under this architecture, a ground mobile base, multiple arms and multiples sensors are able to be integrated into one entity, this is a very generic combination for agricultural robotic systems, or other industries. The next chapter details the navigation and control software of the system, its design, implementation, experimentation and evaluation in a simulated, laboratory and agricultural setting.

Chapter 4

Strawberry Picking Position Control

4.1 Introduction

The use of technology in agriculture for food production and harvesting has been of increasing interest over the past decade, with a rising global population comes an increasing food requirement. These systems such as robotics may be one solution to this problem. The use of automated robotics in agricultural environments can solve many current issues. Their ability to operate 24 hours a day could dramatically increase productivity and harvesting yield, with minimal human labour requirements, whilst reducing food waste. This solution has many challenges to be overcome to enable automated robotics to operate and complete tasks in these environments.

The nature of agricultural areas is often dynamic, structurally variable and hazardous. An automated system would be required to operate with humans in the vicinity, navigating difficult terrain and maintaining accurate functionality such as the task of fruit harvesting. Agricultural environments such as found in a vertical growing facility have a general overall structure, including a defined perimeter, the interior environment however is dynamic, with the

vertical growing baskets changing position, the terrain often being undulating and comprised of soil or other shifting material and often without GPS as found in the environment used in this research.

This type of environment would require an automated system to be capable of dynamically adapting to its environment, for example whilst moving along a vertical hanging basket row, and to be capable of accurately harvesting fruit without damaging any infrastructure or the fruit itself. The system would also need to detect any humans or obstacles in the environment to avoid whilst harvesting.

This chapter is a description of the development, implementation and evaluation of an automated system for agricultural fruit harvesting as shown in Fig.4.1. First the objectives and contributions are described, detailing the methodology and outcome objectives of the system. This followed by the implementation of the system, how the system was built and its implementation within the architecture previously described. The final section of this chapter details the experimentation and evaluation of the system in a simulated, laboratory and real world environment.

4.1.1 Objectives and contributions.

The objective of this chapter is to develop and evaluate a novel method of autonomous strawberry picking positional control and navigation in a large, unstructured, agricultural environment for the purpose of fruit harvesting using two robotic arms.

The application of the system being to harvest strawberries from fruit bushes presents a number of interesting and complex challenges. The environment consists of open, uneven terrain, with the fruit bushes being in vertical rows throughout the facility. The terrain and environment the system is required to operate in means methods such as artificial or natural beacons,

Figure 4.1: Robotic System in Vertical Fruit Row.

or natural or prior installed line following are not practical as the rows are frequently changing position, with varying fruit bush density and with differing degrees of visibility along a fruit row.

A line following method would require considerable maintenance and would be inaccurate by the nature of the ground being both uneven and shifting, potentially covering any sort of colour based line for a line following method to follow. Magnetic paths to follow would also be unsuitable for this environment as, although the ground layout does not frequently change, the shape and density of the fruit bushes hanging vertically would render this often inaccurate if not unusable. The facility being partially indoors would also render GPS inaccurate, if not unusable as demonstrated in chapter 3.

A natural or artificial beacon based navigation method would be unsuitable in this type of environment as the vertical fruit bush density constantly changing may cover artificial QR

codes, or natural beacons. For this type of navigation to be feasible, it would require constant maintenance. It would also not be able to be generalised to different facilities without considerable human intervention, a consideration of this system, such as updating the QR codes or ensuring the path the robot is traversing is kept clear. This meant a new method would need to be developed for the system to travel along a row of fruit, harvesting the strawberries as it moves.

This chapter details the implementation of the configurable, modular software design in chapter 3. The system development and implementation is first described before experimentation and performance analysis of the system in a laboratory environment and the vertical farm. The section is comprised of the following contributions for this research:

- 1. The development and integration of the modular configurable system architecture.
- 2. Novel navigation and control method for the robotic arms and base in a vertical facility.
- 3. The experimentation and evaluation of the system in a laboratory environment.
- 4. The experimentation and evaluation of the system in the agricultural environment.

The system's design was to be modular and configurable with a potential to be adaptable for many agricultural environments. This implementation is in regard to a vertical growing strawberry farm as described in the following sections.

4.2 Overview and Requirements

The system as shown in Fig.4.2, is comprised of two main aspects, a perception system and a navigation controller: The system requires each component to function. The perception

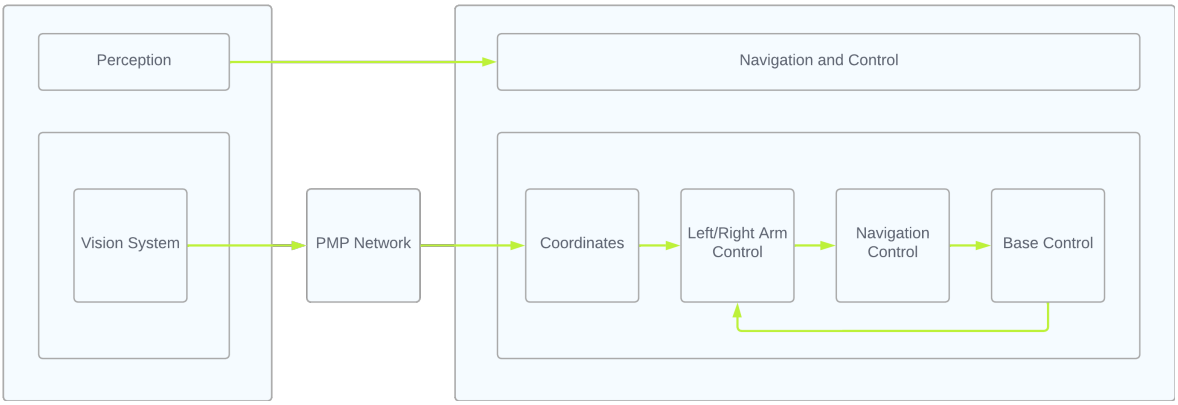

Figure 4.2: Perception and Navigation Components.

method is a separate system to this research, and provides target coordinates for each strawberry to harvest.

As a series of individual inputs, these target coordinates are then used by the strawberry picking position and navigation control for both left and right robotic arms and mobile base. This gives two coordinates systems, one for the robotic base and arms and one for the perception system, named Robot Coordinates and Camera Coordinates respectively.

- Robot Coordinates: Xr, Yr, Zr
- Camera Coordinates: Xc, Yc, Zc

These two coordinate systems are the X,Y and Z coordinates of the robot and camera as shown in Fig. 4.3, providing a bounding box in which the robotic arms can operate, without conflicting or damaging themselves. The perception system's output is the Xc, Yc and Zc coordinates and joint angles for each arm.

These are the inputs taken by the position and navigation control method. The perception system passes these coordinates to the PMP network for each robotic arm, a part of the perception system, that generates the joint angles to the detected fruit to be harvested. The picking position and navigation control uses these coordinates and target joint angles to de-

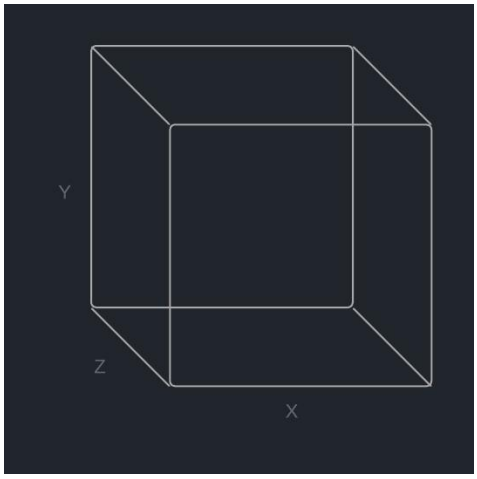

Figure 4.3: Visualisation of Operating Region for the Robotic Arms.

termine whether a given strawberry is reachable by the robot in its current pose, if the robotic base is required to move, and to determine the most optimal position of the robotic base for harvesting the fruit.

The optimal position for harvesting the fruit was determined through experimentation as described in this chapter. If the strawberry is in a suitable position for harvesting, the robotic arm or arms will operate and harvest the fruit. If the robotic base is required to move, the coordinates from the perception system are passed to the position and control methods and the robot will move left or right, dependent on which direction will bring the robotic base to its most optimal position for the arms to function. Once the fruit has been harvested, the robot will continue moving left along a fruit basket harvesting strawberries autonomously until a new fruit is detected through the coordinates received from the perception system and the process repeats. This continues until there is no fruit detected or by stopping the system. The system is controlled through a user interface as shown in Fig.4.4, all operational modes including the perception system can be launched from this menu. The UI allows optional control of manual navigation, left and right arms individually, gripper control and an autonomous operation mode of the robot.

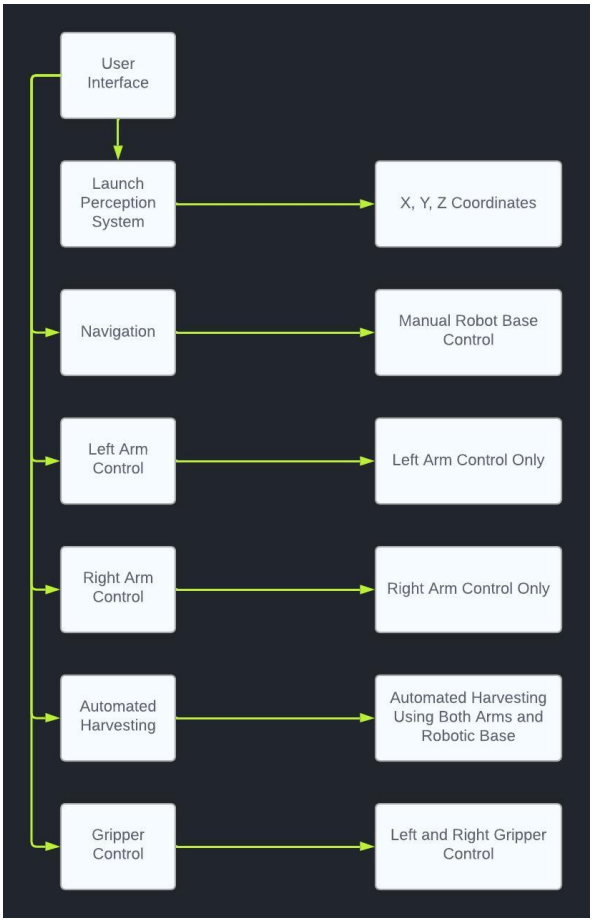

Figure 4.4: User Interface.

4.3 Navigation and Control

The navigation and control is separated into two main components, a navigation class for basic movement of the robot and a navigation control class that is directly related to the left arm class. This combines the functionality of the left and right arms with the robotic base for the system to operate at the same time. The robot is driven forward by the requirements of the left arm as shown in Fig.4.5, the system repeatedly checking and updating the target position against the robot base's current position and correcting for this until the left arm is in an optimal position to harvest the fruit, this based upon the Zr and Zc coordinates being within or equal to 500mm to 800mm from the target strawberry to harvest.The system was

designed to follow the left arm specifically to ensure there are no conflicting base movement parameters, for example the base attempting to move left and right at the same time to ensure the optimal range for the left and right arms, and to ensure the left and right arms do not conflict and damage each other. This also ensures no strawberries are missed by the system as it moves. If a fruit is not detected by the left arm as the robotic base moves along the row, it may be detected by the right arm.

4.3.1 Movement Strategy

There were various movement strategy's considered to use for the systems functionality. The system could have been developed to attempt to harvest a strawberry as soon as one had been detected by either arm, left then right arm driven or to only attempt to harvest a strawberry if both arms were in an optimal position amongst other methods. The system was required to operate with both arms and base functioning at the same time, this meant a strategy would be needed that enabled either robotic arm to operate, without conflicting left and right base movements. The most appropriate movement strategy for this system was decided to be to follow the left arm. The perception system provides two separate sets of coordinates, one for the left arm and one for the right, the robot is designed to move left along a fruit bush, prioritising the left arm. If a strawberry is detected and reachable by the left arm without the base movement, the arm functionality will attempt to harvest the strawberry. If the left and right arms are within range, they will both attempt to harvest the strawberries, if the left arm is not in optimal range, the base will move to position the left arm. If the right arm is not in optimal range and there is no strawberry detected for the left arm, the base will move to position the right arm. The base will move left or right to ensure the optimal position for either arm, but will prioritise the left arm, if there are strawberries detected for both arms

and neither are in an optimal range, the base will move to position the left arm. This strategy also meant that as the robot moved left, if the left arm had missed a strawberry, the right arm may be able to harvest it as it moved along the fruit bush, this also ensured that the robot continued moving to unharvested areas of the fruit bush, as the left side of the robot was continuously moving toward a new area.

4.3.2 Robotic Base Movement, Arms and ROS

The robotic base will continue moving left with the left arm pushing the system forward whilst coordinates for fruit to harvested are obtained. The system will continue moving autonomously until no strawberries are detected by the left arm as shown in Fig.4.5. A second manual controller was also developed for the system, to enable manual control of the robot throughout the vertical farm in the initial testing of the system. The robot requires 3

Figure 4.5: Robotic Base Movement.

distinct components to operate in parallel, the robotic arms, base and the perception system. Both arms also require individual networks to run in parallel. This is completed through multiprocessing, combining multiple system modules to function at the same time. The order of operation is as follows:

- Left and Right target coordinates are obtained.
- Left and Right coordinates are passed as inputs to the respective control functions.

- The mobile base moves to the robot's optimal position according to the left arms coordinates.

- The fruit is harvested and the process repeats until there are no further fruit detected.

The individual arms are controlled through separate left and right arm classes and combined in a main arm control class. The parallel operation of the arms is defined in this class, this also combining the navigation control class and its respective functions, to be passed to the UI.

4.3.3 Position and Position Correction

The position and positional correction of the system is designed to ensure the left arm is always in the optimal position to harvest the fruit and to ensure the robot continues moving along the fruit row without conflicting directions to the robotic base that may occur if two targets are received at the same time. This is controlled through multiple conditions as shown in Fig 4.10. The positional correction behaviour of the robotic base automatically moves the base to a new position if the coordinates to a fruit are out of optimal or unreachable range of the robot. The robotic base will move left or right dependent on which direction would move the robotic arms closer to the required position. The target for both left and right arms are repeatedly updated in the software until a strawberry is detected. Once a fruit has been identified for harvesting by either the left or right arms, these target coordinates are then repeatedly updated and checked in each robotic arms control software as shown in Fig.4.6, whilst the robot is moving, until the robotic arm to be used is in the optimal position to harvest the fruit. If the robotic base is in an optimal range of the base, this determined through the coordinates received being within the optimal parameters, the robotic arm will attempt

to correct its position three times through wrist movements. If the fruit is still detected, it is considered not harvestable and the robotic base will move again until it is in an optimal position.

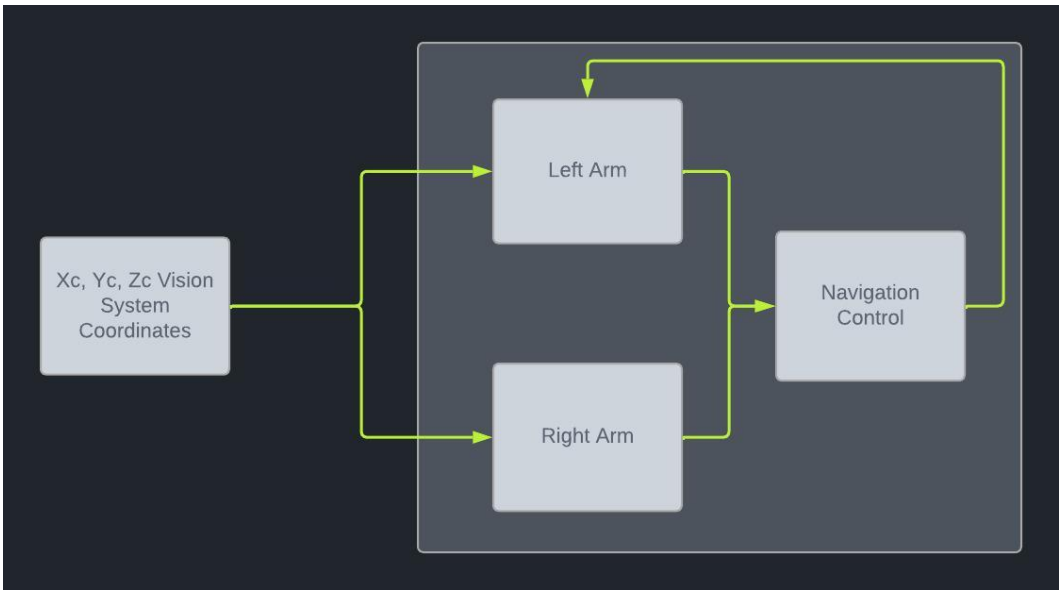

Figure 4.6: Coordinates to Left and Right Arms and Navigation Control.

4.3.4 Left Arm Control

The left arm control method is shown in Fig.4.10, this takes the coordinates Xc, Yc, Zc. A snapshot of these coordinates is taken and stored as Zr, Yr and Xr at specific points in the software. These are taken before the robotic arms function or the base moves from the perception system and are then used to check if the robot base is in the correct position. The ideal position of the fruit for the robotic arms was determined through experimentation, at first the robotic arms had considerable constraints imposed, able to operate in a very small area to ensure they did not conflict or damage each other. The robotic arms have a maximum operating range that would extend across each arms operational area, to ensure they did not conflict, the distance and width required to harvest the fruit was measured and correlated to the input received from the perception system.

The ideal range for the robotic arms to be positioned in front of the target fruit, defined as Y Robot (Yr) and Z robot (Zr), was determined through measurement and experimentation to be 76mm for the Yr coordinate and 500mm to 800mm for the Zr coordinate, this being a minimum Zr=500mm and maximum Zr=800mm to the target fruit. The Y camera (Yc) value was obtained from the perception system, with Yr being the target value. The Yc value was repeatedly updated as the robot moved. As such, when the robotic arm functionality began, the Yr value was set as an additional control method along with the Zr and Zc values, to ensure this value did not update if the strawberry became occluded when the arm was moving. This essentially took a snapshot of its last and targeted location. The Zr minimum and maximum values are also defined at multiple points in the software after the robotic base had moved or before a robotic arm function as additional constraints, to ensure the robotic arms would not function if they are out of the ideal operational range. This was initially determined by measuring the distance from the arms to the strawberry detected, reducing the robotic arms speed, attempting to harvest detected fruit and stopping the robotic arms if they appeared to have any unexpected behaviour. Any range above or below these values can cause unexpected behaviour and potential damage to the robotic arms or surrounding infrastructure, for example the robotic arm rotating 180 degrees at its shoulder joint as shown in Fig.4.7 or a collision with a vertical fruit basket. These constraints would then prevent this behaviour, with multiple constraints added in the initial development of the system and reduced as the systems development progressed.

The robotic base is constantly attempting to move the arms to an ideal position when it is operating, to ensure the Yr and Zr values are in their ideal operating range. When a fruit has been detected the robotic arms will not function until the Zr coordinate is equal to or

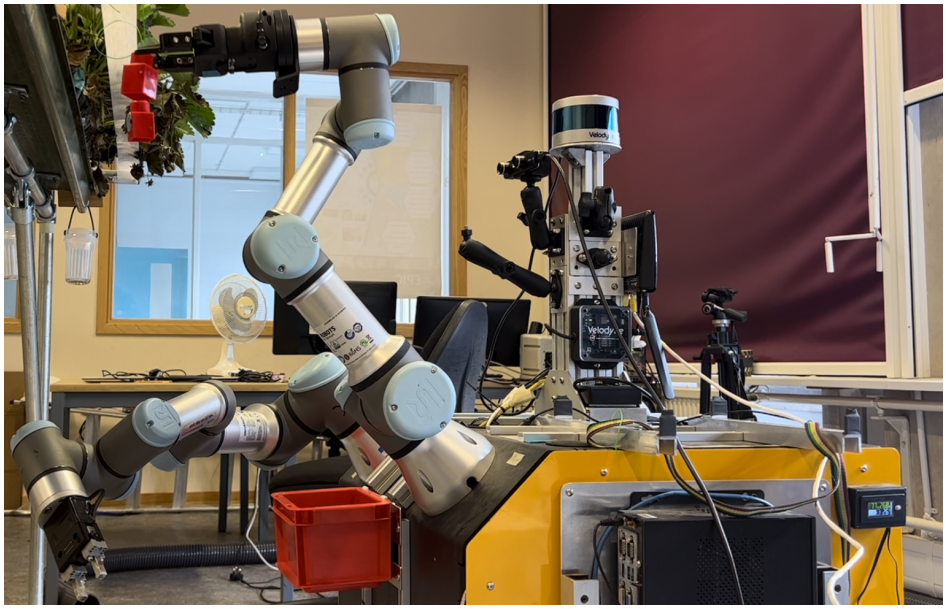

Figure 4.7: Robot Arm Error without Constraints.

greater than 500mm or less than or equal to 800mm and the target Yr value of 76mm has been reached. As the robot moves left or right, the Yc and Xc values are updated with respect to the robot's Yr and Xr coordinates and a snapshot of the coordinates are taken as with the Zc, Zr coordinates. The left or right movement of the robotic base is determined by the Yc input value. A target value Yreq is first calculated before being checked against the current Yc of the robot.

$$Yc - Yr = Yreq \tag{4.1}$$

This would provide the required movement distance in mm (Yreq) to position the robotic arm in front of the strawberry to be harvested, this value is then set as Y position (Y pos) to be used to track the distance moved by the robotic base. The Yc value is then set as Ycurrent (Ycur), If the Yreq value is greater than the Ycur value, the robot will move left, if it is lower, the robot will move right.

$$Yreq > Ycur \tag{4.2}$$

$$Yreq < Ycur \tag{4.3}$$

The coordinates are repeatedly updated as the robot moves to determine if an ideal harvesting position, Yr, Zr has been reached.

The robots position is then checked again once an ideal position has been reached to ensure the fruit is still detected and harvestable as shown in Fig.4.8. If it is not the robotic base will continue to move and update its position until it is in an ideal range. The distance moved, Y position (Ypos) is updated each second as the robot moves until the robotic base has travelled the calculated required distance. The left or right arm will then attempt to harvest

Figure 4.8: Robotic Arm Harvesting Strawberry.

the strawberry detected once the Yr, Zr and Zc coordinate are within the defined range as

shown in Fig.4.9. If the fruit is still detected after this first attempt and the robotic base is still in an ideal range, the arm adjustment, error correction behaviour is triggered.

The left or right arm's fifth or wrist axis is adjusted left, right and vertically to correct for where the fruit stem is located as shown in Fig.4.9. The method attempts three different corrections. If the fruit is not detected, it is determined to be harvested and the method begins again. If the fruit is still detected it is determined to not be harvestable, the robot base will move position again and the process will repeat.

The Xr and Xc coordinates are also obtained and updated as the robotic base moves, and

Figure 4.9: Robotic Arm Harvesting Target Position

could be used to move the robot further or closer to the vertical hanging basket if required.

The systems ability to navigate to given X, Y coordinates was tested in the laboratory but was not required for testing the system.

When evaluating the system's picking and positional control, this was not required as the robotic base and arm length was wide enough to reach the vertical fruit hanging baskets from its furthest or closest positions whilst in the row in regard to the systems X distance from the fruit bushes.

The system also had the capability of using the Velodyne laser for object detection to stop if an object or fruit basket became too close to the system to prevent injury or damage, this determined to be 70cm from the center of the robot, where the Velodyne laser is mounted. This functionality was also tested in the laboratory. The fruit baskets are often within this 70cm range, as such the laser obstacle detection was directed infront of and behind the robot, ignoring the distance to the fruit basket either side of the system.

The left and right arms will continue to function with the robotic base moving left along a fruit row until no fruit is detected or is considered not harvestable, this enabling autonomous picking and positional control of the system.

4.3.5 Right Arm Control

The right arm control operates in the same way as the left arm as shown in Fig.4.11, but does not move the robotic base. The arm follows the same control method, this through checking if a fruit is detected. If a fruit is detected, the Zr and Zc coordinates are checked to ensure the robotic arm can reach the fruit.

The robot base will also move to correct the right arms position, provided there is no coordinate from the left arm, this to prevent conflicting base movement position. The coordinates are then updated again to determine if the fruit has been harvested. If a fruit is still detected

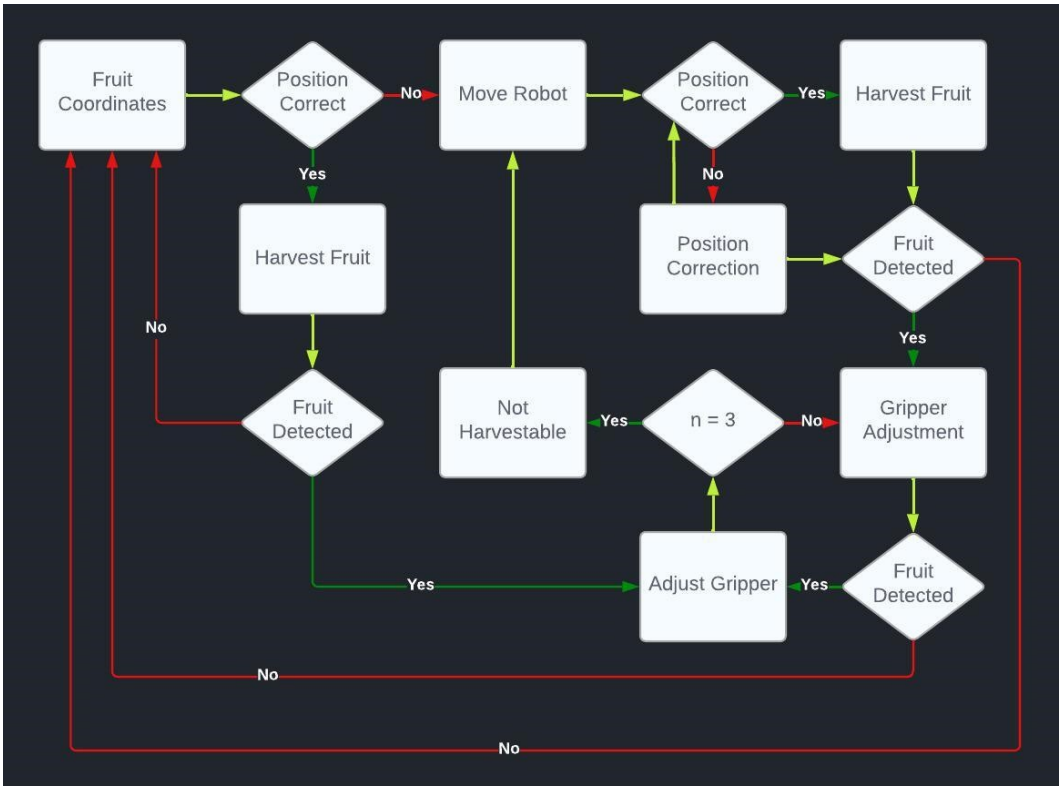

Figure 4.10: Left Arm Control Method.

and the arm is in an optimal position with the Zr and Zc values of 500mm to 800mm, the system will attempt to harvest the fruit in three different arm positions as with the left arm. The coordinates will repeatedly update to determine if the fruit has been harvested. If it is detected after the third arm positional movement, the fruit is determined to not be harvestable. If the fruit is not harvestable by the right arm, the system will stop and wait for the left arm to move the base to a position where this method will repeat, this to ensure the base is consistently moving along the row and not given conflicting left and right target coordinates.

4.3.6 Automated Fruit Harvesting

The autonomous harvesting option as shown in Fig.4.4, in the UI, is the continuous and normal operational state of the robot, to enable continuous harvesting of a row of fruit. The system was initially evaluated in a laboratory environment as shown in Fig.4.12, for both

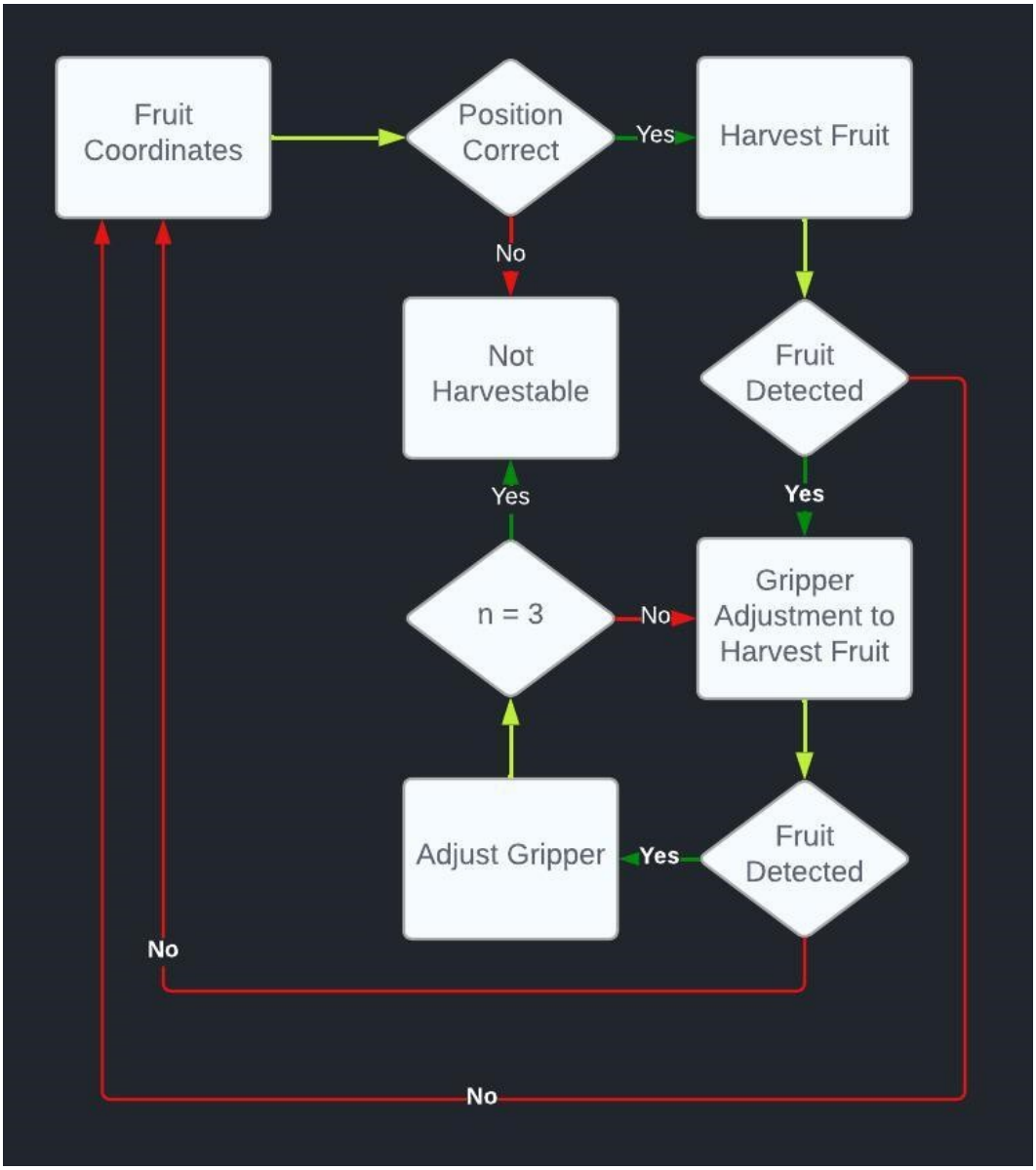

Figure 4.11: Right Arm Control Method.

individual and combined arm and base functionality.

The robot was then evaluated in the vertical growing farm as shown in Fig.4.13, there were a number of further obstacles to overcome in this setting. For example additional factors such as wind, lighting conditions, wet terrain and obstacles in the robots path.

The evaluation and performance analysis of the system shows first laboratory experimental conditions followed by the vertical growing farm. The system's performance was determined

based upon its ability to successfully move to and harvest strawberries autonomously.

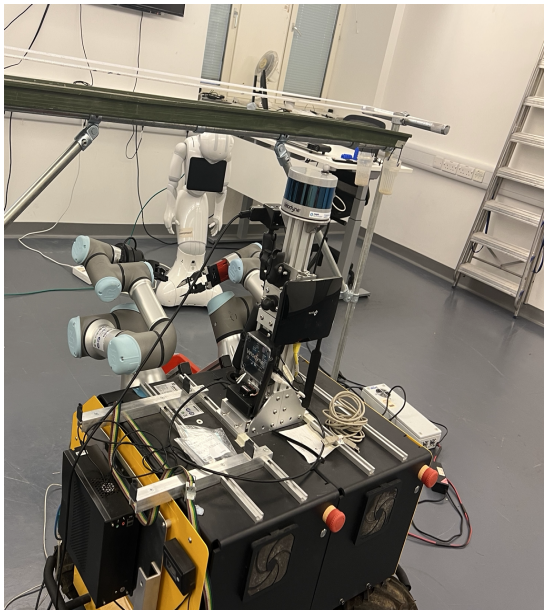

Figure 4.12: Robotic System in the Laboratory.

Figure 4.13: Robotic System in the Farm.

4.4 Experimentation and Performance Analysis

The robotic system was extensively tested and improved in simulation for the navigation as in chapter 3, a laboratory setting for the initial testing and improvement of the control method and in the vertical farm to prove the concept of the system.

The laboratory setting used at first a representation of a strawberry, this followed by a to scale replica of a section of the vertical hanging baskets and real strawberries.

The system was then tested in the vertical growing farm and its performance was evaluated before improvements being developed and implemented and the evaluation repeated whilst reducing the constraints of the robotic arms to improve the performance of the system.

The laboratory and farm experimentation, evaluation and performance analysis was sepa-

rated into a number of tests to establish the robots functionality, these in the laboratory and vertical growing farm as listed below:

Laboratory:

- Initial testing for base movement and arm control with a plastic strawberry representation.
- Autonomous harvesting with arms and robotic base using a real strawberry bush.

Veritcal Growing Farm:

- Initial testing in the vertical growing farm for base movement and the left robotic arm.
- Arm error correction with real strawberry bush's for occluded or difficult to reach strawberry's.
- Autonomous harvesting with the robotic base and robotic arms in a row of the vertical farm.

4.4.1 Laboratory Experimentation and Evaluation

The laboratory experimentation and evaluation enabled the year round continuous development and improvement of the system. The position and control method was tested for its ability to repeatedly detect, move to and harvest a fruit, its speed and harvest rate.

The system was capable of harvesting fruit continuously at a rate of one strawberry every 10 - 30 seconds, dependent on the visibility of the fruit and whether it harvested the fruit in the first attempt or after subsequent attempts with the error correction.

The perception system categorised the fruit as easy, medium and hard in regard to visibility.

The system was capable of harvesting all of the strawberries categorised as easy that were detected and was able to harvest fruit with both arms. The left arm gripper functioned as expected throughout the evaluation of the system. The right arm gripper was damaged in an unrelated experiment and was unable to open and close.

With this, if the arm reached the strawberry and was able to position the stem in a 2cm gap between the left and right parts of the gripper cutter, the strawberry was considered to be harvested. The right arm, although unable to cut the stem with the broken gripper, was able to reach the detected fruit with all of the strawberries categorised as easily visible.

The following experimental results show the system evaluation in the laboratory for both robotic arms in Fig. 4.14 to 4.16 using a representation of a strawberry.

The evaluation of the system with real strawberries is then shown in Fig.4.17 to 4.19. The system was repeatedly improved and evaluated in the laboratory before being evaluated in the vertical farm environment to ensure it functioned as expected and not cause damage to the vertical hanging baskets, environment or to the robot itself.

4.4.2 Picking and Position Control with Representative Fruit in a Laboratory

The system is shown to move along the vertical row and picks the three representative strawberries autonomously. The system at first moved left to position itself for the left most strawberry, this also enabled the right strawberry to be detected and was also harvested.

The system then moves right to position itself for the final left strawberry and was able to harvest this. With the system capable of autonomously harvesting representative strawberries, it was then evaluated with real strawberries in a replicated version of the vertical growing farm. The following sequence of Figures show this functionality, at first a left and right

strawberry are detected, the robotic base then moves left to position the left arm optimally in front of the fruit as shown in Fig.4.14. The right arm is also in a correct position to harvest a strawberry after this movement and both arms operate as shown in Fig.4.15. This enables two strawberry's to be harvested, the final strawberry is then detected by the left arm, for this the robotic base moves right to position itself as shown in Fig.4.16, the left arm then functions again and harvests the left strawberry. This was completed using a metal frame

Figure 4.14: Laboratory Picking 1.

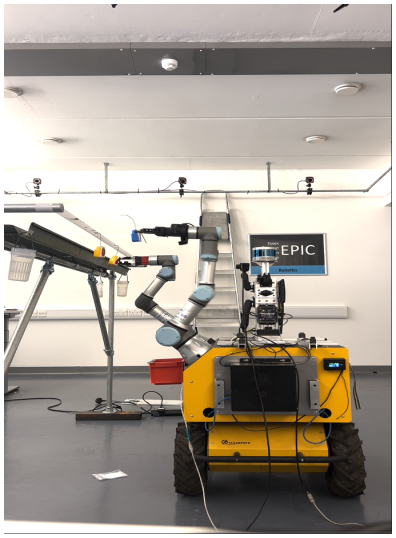

Figure 4.15: Laboratory Picking 2.

Figure 4.16: Laboratory Picking 3.

set to the height of the vertical hanging baskets and using real strawberry plants that were to be harvested. The system was capable of repeatedly harvesting strawberries categorised as easy and frequently capable of harvesting strawberries categorised as medium difficulty to harvest. The strawberries categorised as hard were intentionally ignored by the system as these were considered not to be harvestable without human intervention.

4.4.3 Picking and Position Control with Strawberries in a Laboratory

The laboratory experimentation with the real strawberries is shown in Fig.4.17 to Fig.4.19. This proved to be successful with each easy categorised strawberry repeatedly harvested,

medium strawberries again frequently harvested, and strawberries categorised as difficult ignored. The robotic base first moved from its initial position right to the optimal position for

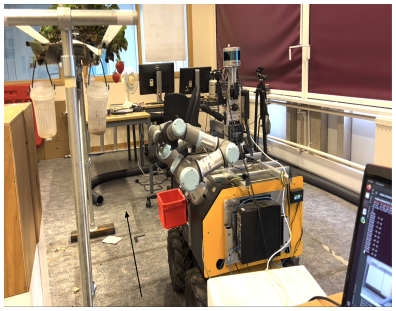

Figure 4.17: Laboratory Picking Strawberry 1.

Figure 4.18: Laboratory Picking Strawberry 2.

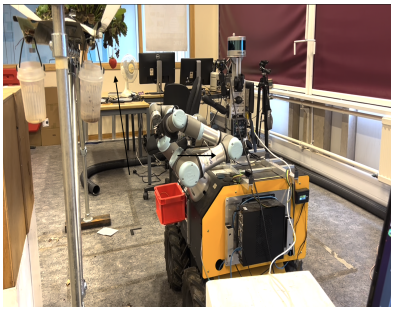

Figure 4.19: Laboratory Picking Strawberry 3.

the left arm as shown in Fig.4.17 and Fig. 4.18., the right arm also detects a strawberry it is able to harvest and both arms operate. The right arm gripper is damaged in this experiment and although reaches the target correctly but is unable to open and close to harvest the fruit correctly, the left arm functions as expected and harvests the strawberry as shown in Fig.4.19. The initial laboratory evaluation proved the system was capable of autonomous strawberry harvesting in ideal conditions and capable of repeatedly demonstrating this. The next evaluation would be in the vertical farm environment, this having a number of additional challenges to evaluate the performance of the system.

4.4.4 Vertical Growing Farm Evaluation and Performance Analysis

The system was then evaluated at the vertical growing farm as shown in Fig.4.20 to Fig.4.23. This presented a number of expected and unexpected challenges. The increased density of the fruit bushes and narrower operating environment were expected and accounted for in the development of the system, being considerations for the initial constraints of the robotic arms and base. An unexpected yet considerable challenge in this environment was wind. The vertical farm is indoor, and testing was often in warm conditions without wind, as such

this was not initially accounted for. The error correction behaviour of the system was able to accommodate for most windy conditions and enabled the system to continuously harvest the fruit, this demonstrating the robustness of the system. The system performed as expected in these conditions. If weather conditions were able to move the fruit bushes enough to change the fruit detected from easy to medium or hard, the system was not as capable of harvesting the strawberries. If the fruit was categorised as easy in the vertical farm, the system was able to maintain a 100 percent harvest rate in all experimental attempts.

4.4.5 Picking in Vertical Farm

The system was able to repeatedly harvest each of the strawberries categorised as easily visible, as long as their stems were not considerably occluded, this is shown in Fig.4.20 to 4.23. This was also in a considerably more challenging environment in regard to fruit bush density and terrain, this showing the systems precision and accuracy in maintaining an optimal robotic base and arm position. The system proved to be capable of harvesting the strawberries in the farm environment as in the laboratory environment. The considerably denser fruit bushes and uneven terrain meant that more strawberries were being categorised as medium or hard in regard to visibility and the easy categorised strawberries were sometimes more occluded, requiring a particular angle to be harvested. This required the method of error handling and correction. If a strawberry was categorised as easily visible, but the stem was occluded, the error capability was able to account for this and the fruit was able to be harvested.

Figure 4.20: Vertical Farm Strawberry Picking 1.

Figure 4.21: Vertical Farm Strawberry Picking 2.

Figure 4.22: Vertical Farm Strawberry Picking 3.

Figure 4.23: Vertical Farm Strawberry Picking 4.

4.4.6 Picking Error Correction in Vertical Farm

The error correction method is shown in Fig. 4.24 and functioning at the vertical farm in Fig.4.25 to 4.28 for the left robotic arm, with each being capable of this behaviour and able to harvest strawberries that are categorised as easily visible. The robotic arm first attempts to harvest the strawberry and misses, in the second attempt the robotic gripper rotates right 5 degrees and is able to harvest the strawberry. If the harvesting attempt was not succesful, the robotic gripper would rotate to its initial position, then 5 degrees left, with a third rotation back to its initial point and moving 10 degrees downward in order to attempt three different gripper positions that may improve its ability to harvest the strawberry. If these three

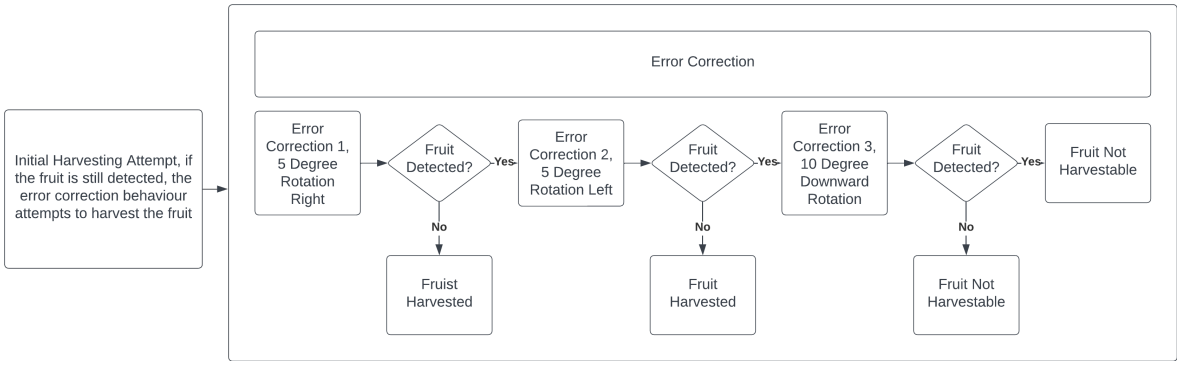

Figure 4.24: Error Correction Behaviour.

positions did not harvest the fruit, the optimal position of the arm would be checked again and the base move if required, the arm functionality repeat, or if the strawberry detected was not categorised as medium or hard in regard to visibility, the system would move to the next strawberry detected categorised as easily visible. The error correction behaviour accounted

Figure 4.25: Vertical Farm Error Correction 1.

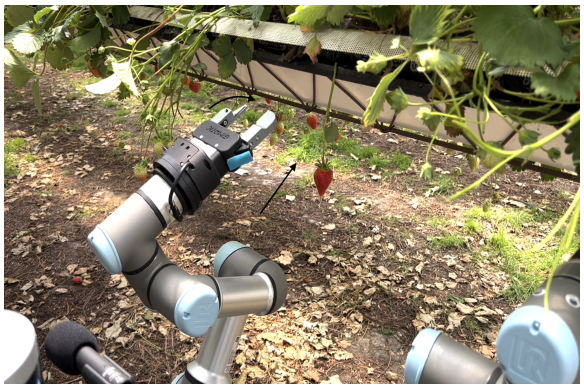

Figure 4.26: Vertical Farm Error Correction 2.

Figure 4.27: Vertical Farm Error Correction 3.

Figure 4.28: Vertical Farm Error Correction 4.

for occluded strawberry stems that were categorised as easily visible and enabled the system to harvest all strawberries detected as easily visible. The final evaluation of the system was with autonomous harvesting.

4.4.7 Picking and Position Control in a Vertical Farm

The system was then evaluated with autonomous harvesting in a row of the vertical farm as shown in Fig.4.29 to 4.32. The robot first moves left to position the left arm in an optimal position for harvesting the detected strawberry, the strawberry is then harvested as shown in Fig.4.29 to Fig.4.30. The robotic base then moves left to position the arm in the optimal position to harvest the next strawberry detected as easily visible, the arm functionality then harvests the strawberry as shown in Fig.4.31 to Fig.4.32. The system was capable of continuously detecting and autonomously harvesting strawberries, moving along the row whilst strawberries categorised as easily visible were detected, the system was tested along 10m of the vertical fruit row and able to harvest all of the strawberries categorised as easily visible. This experimentation and evaluation proving the concept and capabilities of the system. The system was evaluated and improved in the laboratory and vertical growing farm continuously to improve its harvesting accuracy with it able to harvest all strawberries categorised as easily visible.

The mobile base, although capable of moving at 1.0 m/s, was set at a speed of 0.1m/s throughout this experimentation. The robotic arms were also constrained to an arm acceleration and speed of 4 radians per second, with delays between arm function execution at first set at 2.0 seconds to ensure the system functioned as expected. The delays were then lowered to 1.0 seconds to increase the system's harvesting efficiency and performance improvement. The system was able to harvest a strawberry at a rate of 1 per 9 seconds and tested 10-

Figure 4.29: Vertical Farm Autonomous Harvesting 1.

Figure 4.30: Vertical Farm Autonomous Harvesting 2.

Figure 4.31: Vertical Farm Autonomous Harvesting 3.

Figure 4.32: Vertical Farm Autonomous Harvesting 4.

40 times on multiple occasions in both a laboratory environment and the vertical farm. This evaluation and performance analysis proving the concept of the autonomous picking position and navigation control system.

4.5 Conclusion

This chapter detailed the development and implementation of the system, the functionality of the left and right arms and robotic base, and the combination of this functionality.

The evaluation and performance analysis of the system showed the robot to be capable of autonomously harvesting strawberries in a laboratory and vertical farm environment, able to harvest all strawberries categorised as easily visible.

The arm speed was increased and constraints reduced throughout the development and evaluation of the system as its capabilities and accuracy were improved. It was capable of error correction for strawberries of increasing difficulty and of harvesting strawberries with the left and right arms, with the robotic base capable of navigating to the optimal position for the require arm.

This experimentation and evaluation proved the concept of a configurable, modular system architecture and strawberry picking positional and navigational control for an autonomous fruit harvesting robot.

The next section details a potential replacement for the perception system, with an instance segmented 3D environment reconstruction system, on a small, light weight device, incorporating this position and navigation system.

Chapter 5

Single Board Computer for a Strawberry Picking Robot

5.1 Introduction

The effective use of an agricultural fruit harvesting system would require it to operate over large areas for a prolonged period of time, as such, reducing the power requirements of the system would be beneficial in enhancing operational time and productivity. There are a number of components whose power consumption can not realistically be reduced without affecting the functionality of the system, such as the robotic base and arms. The system computer used for navigation, control and perception is the most appropriate power intensive component that can be changed. This would also have the additional benefit of reducing the overall cost of the system, whilst also increasing the operational time of the system between the battery charges.

The system had currently been controlled by a laptop with a maximum battery life of 45 minutes, without being connected to a mains power supply. If this were to be replaced with a

lower powered embedded system such as a single board computer(SBC), this would be quite drastically improved.

Along with reducing the systems power consumption, its efficiency and productivity could be further improved by incorporating additional functionality into the perception system, by replacing this system with a method capable of obtaining metrics such as harvesting yield and crop quality, or to enable new functionality such as remote crop monitoring for mould or pest detection. This additional functionality could be realised through a 3D instance segmented environment reconstruction of the robots environment, obtained by using largely the same data collected by the robot as it moves throughout the rows of the farm.

The modular design of the system architecture would enable any additionally required sensors, for example a second camera sensor and additional software to be implemented with the existing components and without having to redesign any of its other aspects, essentially with a plug and play methodology.

In order to replace the onboard computer, the candidate replacement single board computers would first need to be evaluated to ensure they are capable of running the system's existing components, such as the navigation and control software and the perception system.

This chapter proposes a method to achieve these tasks by first evaluating three instance segmentation networks based upon their capabilities to detect strawberries in an agricultural environment using a single board computer, to determine whether the boards are capable of running the networks with the higher resolution images required to detect the detailed features of a strawberry. The two single board computers are compared with the currently used laptop based on their real time inference speed performance. The final section of this chapter introduces a method of 3D environment reconstruction for the system.

Figure 5.1: The Robotic System.

5.1.1 Objectives and Contributions

The objectives of this chapter are to improve power consumption and efficiency of the system. The two methods identified to achieve this are by reducing the systems power consumption and through incorporating additional capabilities such as metric collection. Currently, the system has a terminal based user interface and graphical interface displaying the perception system, these functions are power and computationally expensive.

The autonomous operation of the system does not require a visual output to ensure the system will function correctly as there are a number of redundancies to ensure it will only attempt to harvest fruit if it is able to reach it as described in the previous chapter. As such, the most appropriate component to replace of the system would be the computer used to control it. This chapter explores the options available to achieve this through the following contributions:

- 1. A comparative analysis of instance segmentation networks on a Laptop, Jetson Nano and Jetson Xavier is conducted to ensure the single board computer is capable

of running the existing system in an agricultural environment.

- 2. A 3D instance segmented environment reconstruction system is proposed.

This chapter at first introduces two single board computers, detailing their specification to determine their suitability as a replacement for the currently used laptop, whether the boards are compatible with the base software of the system such as their operating system and software library compatibility. This is followed by initial experimentation with the selected SBCs to determine if they are capable of detecting strawberries in an agricultural environment, before a comparative analysis of three instance segmentation networks on the laptop and the single board computers.

The comparative analysis of these systems would also be to determine their real time performance in comparison with the currently used laptop, to ensure they would be a viable replacement and able to run the existing system. The final section proposes a 3D instance segmented environment reconstruction method including the initial evaluation of a 3D camera based SLAM algorithm.

5.2 Single Board Computer for a Fruit Harvesting System

A single board computer (SBC) is a small, low powered single circuit computer, comprised of a microprocessor, input/output and memory, with some of these systems containing additional components such as a graphical processing unit dependent on their intended application. They are compatible with various operating systems, software libraries and languages, often used as a controller for a wide variety of applications across many industries from factory controllers to satellites and particularly in robotics.

5.2.1 Robotic System Software Compatibility

The system chosen for this application is required to run a variety of software components, the navigation and control software and three separate networks, one for each of the robotic arms and the perception system.

SBCs are often developed for specific controller applications. This system requires many software components to function at the same time. As such, to be viable replacement for the laptop used, a relatively computationally powerful SBC would be needed, capable of running the various software components of the system, whilst maintaining real time performance for the perception system.

The navigation and control software was developed specifically to be compatible with Ubuntu 20.04, using Python 3.7 and ROS Noetic. This was to ensure future compatibility throughout the system's development as there were multiple aspects to be integrated as the robots development progressed. The perception system required NVIDIA CUDA to run, a toolkit developed by NVIDIA to enable a graphics processing unit (GPU) to be used for general purpose processing, this only compatible with NVIDIA hardware. This system also required Tensorflow, an open source machine learning, training and inference software library.

These requirements meant an SBC with an NVIDIA GPU able to run CUDA would be needed, of which there were a number of options available. The two boards selected were a Jetson Nano and Jetson Xavier NX based upon their low battery requirements and relatively high computational power for their size and cost, whilst compatible with all of the requirements of the system.

Along with these software requirements, the SBC would also need up to 4 I/O ports and 1 Ethernet port for the system existing and potentially additional sensors, for example if a

second camera was added.

5.2.2 Jetson Nano Single Board Computer

The first board selected was a Jetson Nano, this SBC has a Quad Core Arm Cortex-A57 CPU, 4GB of 1600MHz DDR4 Memory, 4 USB 3.0 ports, 1 Ethernet port and a 128 Core GPU, whilst only requiring a 5V power supply. This SBC is compatible with NVIDIA CUDA,

Figure 5.2: Jetson Nano.

ROS Noetic and Python 3.7, making this an ideal candidate SBC for evaluation.

5.2.3 Jetson Xavier Single Board Computer

The second board selected was the Jetson Xavier NX, this has a 6 Core NVIDIA Carmel ARM V8.2 64bit CPU, 8GB 1600MHz DDR4 Memory, 4 USB 3.0 ports, 1 Ethernet port and a 384 Core GPU. Include that the ZED2 Camera was used. This SBC, although considerably

Figure 5.3: Jetson Xavier.

higher cost than the Jetson Nano, has the same number of I/O ports, also requiring a 5V power supply but with double the memory and 3x the GPU cores. It is also compatible with NVIDIA CUDA, ROS Noetic and Python 3.7, this being selected as the second board for evaluation.

5.3 Instance Segmentation Network Selection and Methodology

The instance segmentation networks selected for evaluation were Mask RCNN, YOLACT and YOLACT Edge. These networks were chosen based upon their inference speed rather than their classification accuracy, as they were required to run on a single board computer. The YOLACT Edge being the most recent of these was also claimed to be capable of running in real time on a SBC. The networks would first be tested to ensure they were capable of running on a SBC using a densely populated image of cars and people, items the network models were trained to detect.

This is followed by compact images of fruit to determine if the models used would be capable of classifying these before their comparison.

The networks would then be compared using a number of images of strawberries taken from the vertical farm, these being relatively high resolution to ensure the strawberries were clearly visible in detail. After this initial evaluation, the networks would then be compared for their frame rate speed on the laptop, Jetson Nano and Jetson Xavier using 400 images from the KITTI 2015 Dataset and 400 images taken of a real world environment to determine their real time performance with differing resolution images.

5.3.1 Instance Segmentation Network Comparative Analysis

The comparative analysis was conducted using 400 images from the KITTI Stereo 2015 dataset using the laptop, an Alienware M15 with an NVIDIA 2080 RTX GPU, the Jetson Nano and the Jetson Xavier, with each network tested using the same dataset.

The laptop was able to run MASK RCNN using these images at 7.9 FPS, YOLACT at 13.65 FPS and YOLACT Edge at 22.34 FPS and was able to instance segment the images as shown in Fig.5.4-Fig.5.11. The Jetson Nano was unable to run MASK RCNN, or YOLACT Edge, but obtained 0.88 FPS with YOLACT using the same dataset.

The Jetson Xavier was also unable to run the Mask RCNN network, but managed to achieve comparative speeds for YOLACT and YOLACT Edge in this experiment at 2.27 and 2.56 FPS, respectively, this slow processing speed may have been because of the image size being 1242x375. The Jetson Nano, although able to run the YOLACT network, only able to perform at 0.88 FPS, was far below the FPS performance required, as such this platform would not be suitable for the system.

The Jetson Xavier, although unable to run the Mask RCNN network, was able to process the same dataset at 2.27 FPS using the YOLACT network and 2.56 FPS using the YOLACT Edge network, although this speed was far slower than required, these results are summarised in Table 5.1.

The YOLACT Edge network being determined to have the highest FPS of the three networks evaluated on the laptop was then also evaluated using a set of 400 images containing household items, at a lower resolution of 672x376 as shown in Fig.5.12 and Fig.5.13. The image processing speed was considerably faster with these images at 29.73 FPS determining the

Figure 5.4: KITTI Dataset 1.

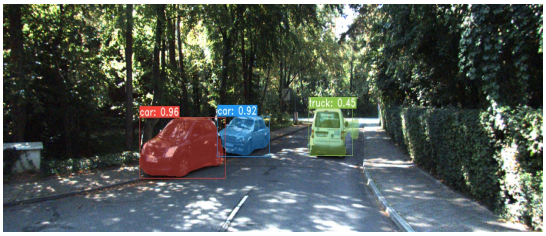

Figure 5.5: KITTI Dataset 1 Instance Segmented.

Figure 5.6: KITTI Dataset 2.

Figure 5.7: KITTI Dataset 2 Instance Segmented.

Figure 5.8: KITTI Dataset 3.

Figure 5.9: KITTI Dataset 3 Instance Segmented.

Figure 5.10: KITTI Dataset 4.

Figure 5.11: KITTI Dataset 4 Instance Segmented.

Network Comparative Analysis KITTI Stereo 2015 Dataset			
Hardware	Mask RCNN	YOLACT	YOLACT Edge
Jetson Nano	n/a	0.88 FPS	n/a
Jetson Xavier	n/a	2.27 FPS	2.56 FPS
RTX 2080	7.9 FPS	13.65 FPS	22.34 FPS

Table 5.1: Network Comparative Analysis.

Figure 5.12: Image Dataset 1.

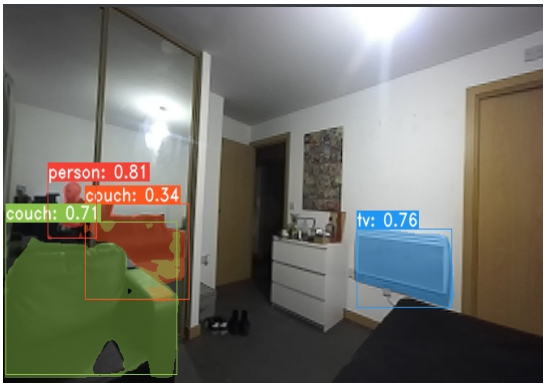

Figure 5.13: Image Dataset 1 Instance Segmented.

resolution had a reasonable impact on the FPS obtained.

5.3.2 Network Inference on a Single Board Computer

The YOLACT Edge network being the fastest of the three networks evaluated was also tested to determine its inference capability on a variety of images to be potentially used as part of a 3D instance segmentation environment reconstruction system.

The Jetson Xavier was also used in this evaluation to determine if it would be capable of processing images taken in the vertical growing farm. The datasets the models used for each network were trained on included cars, people, animals and household items as shown in Fig.5.14 to Fig.517, this having been ran on the Jetson Xavier. With the YOLACT Edge

Figure 5.14: Image of Cars before Instance Segmentation.

Figure 5.15: Instance Segmented Image of Cars.

Figure 5.16: Image of People before Instance Segmentation.

Figure 5.17: Instance Segmented Image of People.

network able to classify these images, the next set of images were of densely populated to fruit to ensure the model could detect these objects as shown in Fig.5.18 to Fig.5.21.

Figure 5.18: Image of Fruit before Instance Segmentation.

Figure 5.19: Instance Segmented Image of Fruit.

Figure 5.20: Image of Fruit before Instance Segmentation.

Figure 5.21: Instance Segmented Image of Fruit.

These were also able to be classified, although the network began to infer objects incorrectly as shown in Fig.5.19 and was unable to detect certain smaller or occluded fruits as shown in Fig.5.21. This is being of relevance as the network would be required to detect strawberries, these often appearing as a similar size in images. The misclassification may also have been because of the dataset the model used was trained on. This would require a new model to be trained for the network to be used in the vertical farm.

After these initial images, the next set were of strawberries taken at the vertical farm as shown in Fig.5.22 to 5.27. These images contained mixed results, with some of the strawberries able to be detected as can be seen in each figure, but some were misclassified. This is because of the dataset it was trained on as some of the fruit in the image are detected, with bounding boxes clearly drawn around the fruit. The network overall was able to segment some of the fruit using the Jetson Xavier. The misclassified fruit in these image may be resolved with a model trained specifically for these objects. To improve this a dataset taken from the vertical farm may also increase the classification accuracy of the network. The Jetson Xavier also being compatible with the navigation and control software, this being considerably less computation intensive, may be a potential candidate to replace the laptop used for the system. The next experimentation would be to determine the speed at which the laptop and boards could run the networks.

5.3.3 Comparative Analysis Evaluation

The network performance could be attributed to an optimisation issue for the single board computer, the network model used, this having been trained on the MS COCO dataset, the dataset resolution or a combination of these. The network FPS results maybe improved by training the network with an appropriate dataset for the type of environment it is inferring,

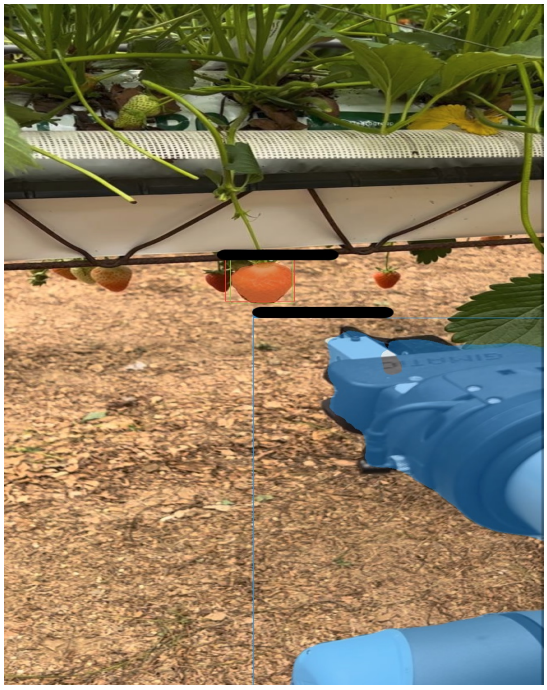

Figure 5.22: Instance Segmented Image of Strawberry 1.

Figure 5.23: Instance Segmented Image of Strawberry 2.

Figure 5.24: Instance Segmented Image of Strawberry 3.

Figure 5.25: Instance Segmented Image of Strawberry 4.

this requiring further experimentation. The single board computers were determined in this evaluation, although compatible with the navigation and control software, were not able to

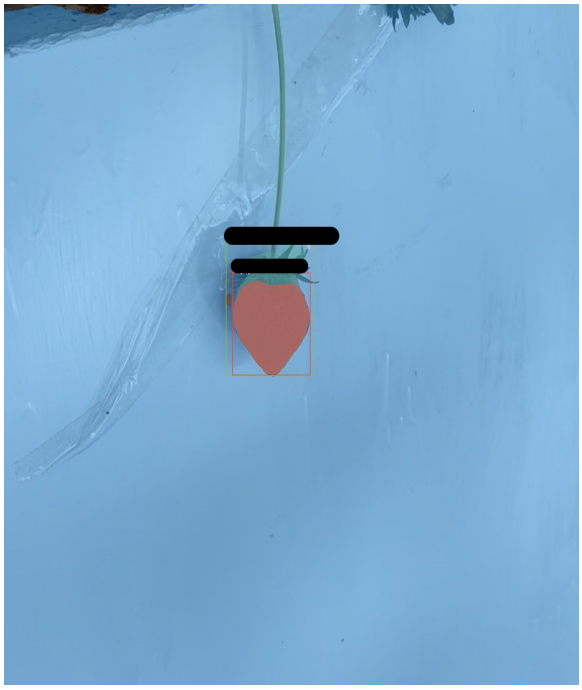

Figure 5.26: Instance Segmented Image of Strawberry 5.

Figure 5.27: Instance Segmented Image of Strawberry 6.

run any of the instance segmentation networks in real time, but provide a potential option as a replacement for the system. With the trend of available lightweight network models, it is possible to use a single SBC for this robotic system.

5.4 3D Environment Reconstruction for a Fruit Harvesting System

The final aspect of this research was to develop a method of 3D environment reconstruction to be used for metric collection and monitoring as the robot harvested the fruit in the farm. The methodology for the reconstruction system was to use a stereo camera to obtain RGB-Depth values from an image sequence in conjunction with a 3D SLAM algorithm. The system would have unmodified images to enable monitoring of the farm and instance segmented images to enable individual fruit to be detected.

An image sequence would be taken with their individual time stamps recorded before being processed as an unmodified and instance segmented version, separated into individual RGB values.

The system would also obtain LiDAR measurements as shown in Fig.5.28, that could then be mapped to the pixel values for each frame. They would then be recombined pixel-wise as

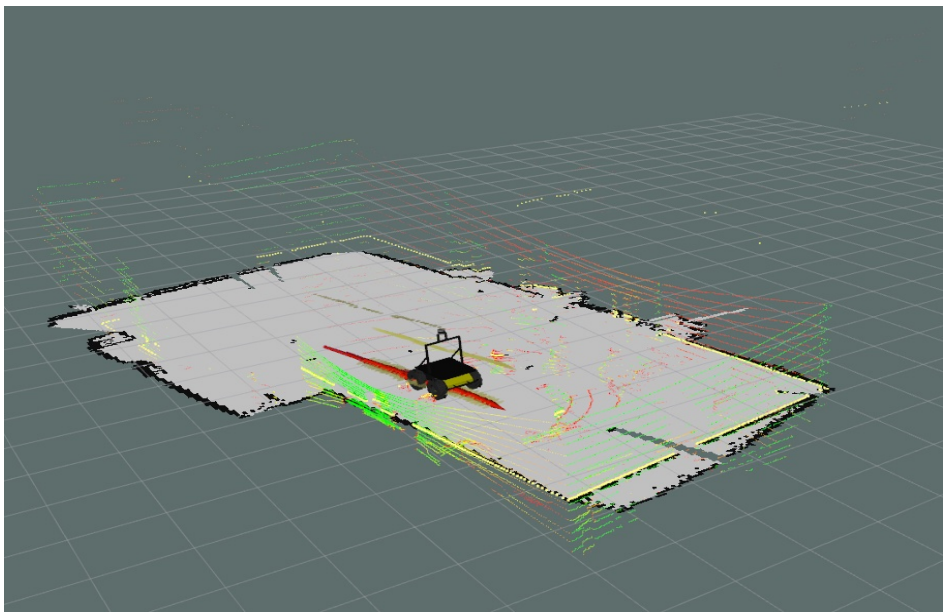

Figure 5.28: Lab LiDAR Map.

the robot moved, using LiDAR distance measurements corresponding with the image frames, the depth map from these images as shown in Fig.5.29. The instance segmented RGB value data, LiDAR measurements and depth map data would then be combined with the robots x, y, z pose data obtained by the SLAM algorithm, this represented in Fig.5.30, and linked to the images by their timestamps. The system as with the navigation and control was also intended to be developed to be modular and compatible with the rest of the robotic fruit harvesting system, with individual modules such as the camera sensor or reconstruction system able to be integrated.

Figure 5.29: Lab Depth Map.

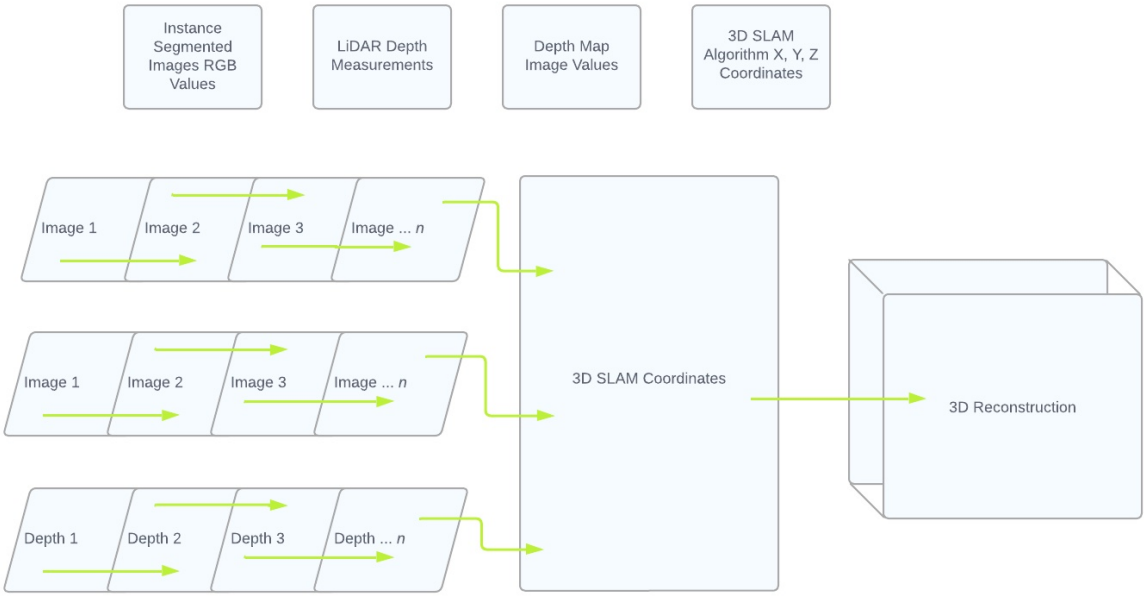

Figure 5.30: 3D Environment Reconstruction Representation.

5.4.1 Laser and Camera SLAM for Location Data

The RTABMap stereo camera 3D SLAM algorithm was first implemented to be used to obtain Cartesian coordinates as part of the 3D reconstruction system. The algorithm was first implemented and tested moving around a room in a square and the coordinates obtained out-

put to a CSV file as shown in Fig.5.31 - Fig.5.32.

This was the initial start of the development of this system. A stereo camera sensor would be used to obtain a sequence of time stamped images, these then instance segmented. The SLAM algorithm location data would then have been combined with the instance segmented images and depth map data from the RGB-D camera as part of an instance segmented 3D environment reconstruction system.

The software to obtain the RGB values was developed for this system, with an image sequence able to be processed, instance segmented and the RGB values output to be combined with the location data from the SLAM algorithm. This would have been integrated with the robotic system.

The system would have used the 3D reconstruction system to generate an overlay as well

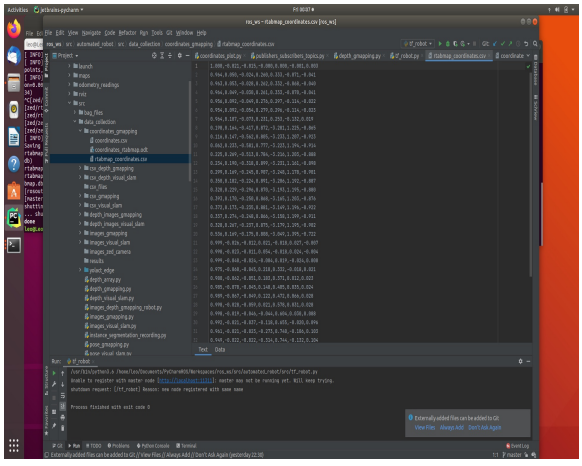

Figure 5.31: Location Data using RTABMap.

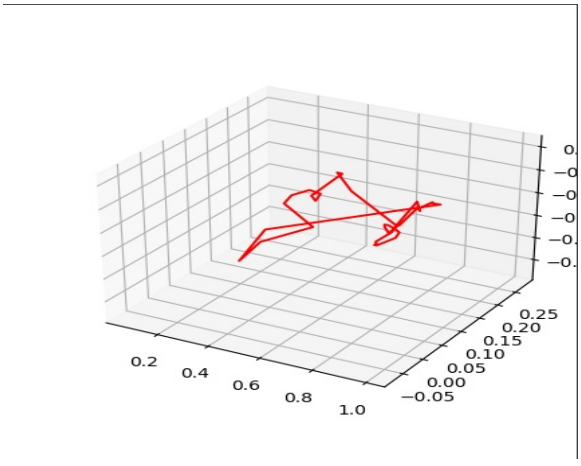

Figure 5.32: X, Y, Z Data Obtained from RTABMap.

as a 3D view of the environment, this could be used for monitoring the environment as the robot moves along with obtaining metric data.

5.5 Conclusion

This chapter's objectives were to explore potential single board computer options to replace the laptop used for the robotic fruit harvesting system, to reduce its power requirements and cost and improve the systems efficiency. A method of 3D environment reconstruction was also proposed and the initial implementation begun.

The two SBCs evaluated, although capable of running the navigation and control software, were not capable of running the instance segmentation networks at an FPS required for the system. This may have been because of the model used for the networks inference or the image resolution of the dataset used for the experimentation. The Jetson Xavier was able to run a network at 29.73 FPS with a Small image size. This is a promising result.

To resolve this, a model could be trained using images from the vertical farm, specific to this application, to potentially improve the image processing speed. The image resolution could have also been reduced as the dataset images used for the comparison were relatively high resolution.

The single board computers were compatible with the navigation and control software and would have been a suitable replacement for this, but they would have also been required to run two deep learning networks for the robotic arms. As such, the SBCs selected for this were determined not to be viable replacements for the laptop used for the systems control.

Chapter 6

Conclusion and Future Work

6.1 Research Summary

The purpose of this research was to determine whether an autonomous fruit harvesting could be developed for a vertical farm environment and to evaluate its performance. To answer this question, a novel modular, configurable software architecture was proposed and developed, with an initial navigation method evaluated in the first of the three main chapters. The software architecture proved to be extremely useful and enabled the configurability required to develop the system as well as for its interoperable functionality.

The second chapter contained the movement strategy, navigation and control implementation and evaluation of the system. The system was capable of autonomous operation, using the robotic base to position itself for the two robotic arms to harvest the fruit. The system was capable of harvesting the strawberries highly accurately, in multiple laboratory and vertical farm environment experiments. A problem the system did encounter was with varying weather conditions such as wind. This was an unexpected environmental factor as the vertical

farm is indoors. The navigation system would also require further improvement to navigate out of a row and into another, especially the row directly next to it, as shown in chapter 4. The algorithm used in this environment was unable to determine every row entrance, often detecting every other row. This may have been because of the lack of structure and large size of the environment or the sensor resolution with this algorithm being too low, instead determining two close row entrances to have been the same or practically indistinguishable. A potential solution in further work could be to use a camera based SLAM system for absolute navigation or the proposed 3D environment reconstruction system.

The third chapter was to look at reducing the cost of the system using a single board computer. This was to replace the laptop that was being used to reduce the systems cost and improve its power efficiency. This required testing multiple single board computers with three different classification networks, to determine if these were capable of running the navigation and control software as well as the perception system. Although these boards were unable to run these networks at the required speed, they were capable of running the networks. They were also compatible with the navigation and control software. With improving classification algorithm techniques and the increasing capabilities of single board computers, it may have been possible to have used a higher specification SBC to replace the laptop, or, with further work, to enable additional capabilities using the proposed 3D environment reconstruction system.

The system overall was able to autonomously harvest strawberries in a vertical growing environment, efficiently and highly accurately in ideal operating conditions.

Figure 6.1: Autonomous Robotic Fruit Harvesting System.

6.2 Research Contributions

This thesis contains the following contributions to the field of autonomous robotics:

Configurable Software Architecture for an Agricultural Robot

A configurable, modular software architecture for an autonomous fruit harvesting system.

The software design and implementation of the custom built, modular system.

An initial evaluation and proof of concept of the system in a laboratory setting with this architecture enabling continuous, rapid development and experimentation.

An initial evaluation of a SLAM algorithm in a vertical harvesting agricultural environment. This included determining the limitations of a SLAM algorithm in an agricultural environment and proved the requirements of a local position control method.

Strawberry Picking Position Control

The development and integration of the modular, configurable system architecture.

A novel navigation and control method for the robotic arms and base in a vertical farm, capable of autonomously harvesting strawberries in this environment.

A method of moving along a row of dense and sparsely arranged strawberries that could be applied to many other agricultural tasks

Developed and implemented a method of error correction to improve the systems harvesting capabilities.

Developed, implemented and proved an Autonomous Robot is capable of fruit harvesting in a vertical farm environment.

The experimentation and evaluation of the system in a laboratory and vertical farm environment demonstrated the system was capable of repeatedly harvesting strawberries categorised as easy by the perception system, at a rate of up to one every 9 seconds, as well as frequently capable of harvesting medium categorised strawberries.

Single Board Computer for a Strawberry Picking Robot

A comparative analysis of instance segmentation networks on a Laptop, Jetson Nano and Jetson Xavier was conducted to ensure the single board computer is capable of running the existing system in an agricultural environment.

Determined the limitations of current single board computers with the perception system.

A 3D instance segmented environment reconstruction system was proposed.

6.3 Academic Publications

ICAC 2022: “Novel Software Architecture for an Autonomous Robotic Fruit Harvesting System”

6.4 Potential Applications

The system developed for this research continued to be evaluated in the vertical farming environment, with further development to the system beyond the scope of this research. Autonomous agricultural robots capable of operating in indoor environments for repetitive, time consuming tasks could be extremely beneficial to the agricultural industry, but also to society with increased efficiency and productivity whilst reducing labour requirements.

These type of systems could also operate in fleets of multiple robots operating together. The technology involved in this systems development can also be applied to other environments, the configurability and modularity of the system enabling components to be easily changed and integrated for its required application.

6.5 Future Work

There are many potential aspects of future work for this system, in agriculture, the ability to forecast crop yield, avoid or remove damaged fruit and detecting pests is often an important, time consuming task, frequently carried out by a human, that may not be very reliable as it is often based upon conclusions from smaller samples that are then extrapolated for entire crops or parts of a farm.

These tasks reliability could be greatly improved by a system be capable of monitoring its environment and collecting metric data as it moves throughout the farm. This could be through counting the amount of strawberries harvested in a given time to predict crop yield, the quality of the fruit in regard to which rows are ripe and which are not ready for harvesting and to assist with pest detection to avoid the plant becoming damaged, potentially detecting these sooner than would normally be noticed.

This additional functionality could be added to the system by implementing a new perception system with the capability to perceive its environment in 3D. A 3D environment reconstruction system could use the data obtained by the system as it navigates through the rows in a farm by incorporating an additional front facing stereo camera and using this image data as it moves to generate a 3D representation of the fruit plants. By integrating a LiDAR sensor with this system, this could potentially provide an accurate virtual representation of the environment, enabling automatic detection and monitoring of the crops for these metrics.

The cost and power consumption of the system and its components is also an area that may be the focus of future work. This system could be extended to a fleet a robots to cover a much larger area in a shorter amount of time, this may become prohibitively expensive with the systems current components. The laptop is one aspect of the system that could be replaced, the single board computers evaluated in this research were not capable of running the entire system, but with improving technology and higher specification single board computers reducing in cost and with the capabilities of light weight instance segmentation networks improving, this may be a potential area for future research.

The navigational capabilities are also aspect of the system that could see future work. The system could potentially use a 3D reconstruction of its environment, incorporating a camera SLAM algorithm to navigate, using this improvement for metric as well as navigation. For the system to navigate around the entire farm environment or for the use of a fleet of these robots, the system could generate a map that is used by itself and any other robots in the same environment. This would assist with obstacle detection and avoidance and with path planning for the fleet of robots to ensure they are not operating in the same rows.

A drone could also be incorporated with the system using the new perception system to determine rows in the farm with the highest yield and directing harvesting robots to these rows, enabling these to be found considerably faster than if the system were to navigate to these on the ground.

The systems architecture was also developed to be generalised to many different environments in its modular, configurable design, by changing the components and incorporating a perception system for a given task, the system could be used to harvest additional types of fruit, remove damaged or mouldy fruit, distribute pesticides in specific locations and many other tasks requiring precise articulation and accuracy.

Bibliography

[1] P. Johan From, L. Grimstad, M. Hanheide, S. Pearson, and G. Cielniak, "RASberry - Robotic and Autonomous Systems for Berry Production," *Mechanical Engineering*, vol. 140, no. 06, pp. S14–S18, 06 2018. [Online]. Available: https://doi.org/10.1115/1.2018-JUN-6

[2] L. Grimstad and P. From, "Thorvald II - a modular and re-configurable agricultural robot," 07 2017. [Online]. Available: https://doi.org/10.1016/j.ifacol.2017.08.1005

[3] Y. Yan, B. Zhang, J. Zhou, Y. Zhang, and X. Liu, "Real-time localization and mapping utilizing multi-sensor fusion and Visual-IMU-Wheel odometry for agricultural robots in unstructured, dynamic and GPS-denied greenhouse environments," *Agronomy*, vol. 12, no. 8, 2022. [Online]. Available: https://www.mdpi.com/2073-4395/12/8/1740

[4] Clearpath Robotics, "Husky UGV - Outdoor Field Research Robot by Clearpath." [Online]. Available: https://clearpathrobotics.com/husky-unmanned-ground-vehicle-robot/

[5] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2980–2988.

[6] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," *Advances in neural information processing systems*, vol. 30, 2017. [Online]. Available: https://doi.org/10.48550/arXiv.1706.02413

[7] D. Drew, "Multi-agent systems for search and rescue applications," *Current Robotics Reports*, vol. 2, 03 2021.

[8] Y. Liu and G. Nejat, "Robotic urban search and rescue: A survey from the control perspective," *Journal of Intelligent Robotic Systems*, vol. 72, 11 2013.

[9] C. Sampedro Pérez, A. Rodríguez Ramos, H. Bavle, A. Carrio, P. de la Puente, and P. Campoy, "A fully-autonomous aerial robot for search and rescue applications in indoor environments using learning-based techniques," *Journal of Intelligent Robotic Systems*, vol. 95, pp. 1–27, 08 2019.

[10] S. Deng, H. Cai, K. Li, Y. Cheng, Y. Ni, and Y. Wang, "The design and analysis of a light explosive ordnance disposal manipulator," in *2018 2nd International Conference on Robotics and Automation Sciences (ICRAS)*, 2018, pp. 1–5.

[11] A. H. Zakaria, Y. M. Mustafah, J. Abdullah, N. Khair, and T. Abdullah, "Development of autonomous radiation mapping robot," *Procedia Computer Science*, vol. 105, pp. 81–86, 2017, 2016 IEEE International Symposium on Robotics and Intelligent Sensors, IRIS 2016, 17-20 December 2016, Tokyo, Japan. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050917302259

[12] C. Wong, E. Yang, X.-T. Yan, and D. Gu, "Autonomous robots for harsh environments: a holistic overview of current solutions and ongoing challenges," *Systems Science & Control Engineering*, vol. 6, no. 1, pp. 213–219, 2018.

[13] M. Dhouioui and T. Frikha, "Intelligent warehouse management system," in *2020 IEEE International Conference on Design Test of Integrated Micro Nano-Systems (DTS)*, 2020, pp. 1–5.

[14] S. Noh and J. Park, "System design for automation in multi-agent-based manufacturing systems," in *2020 20th International Conference on Control, Automation and Systems (ICCAS)*, 2020, pp. 986–990.

[15] V. R. Miranda, A. M. Rezende, T. L. Rocha, H. Azpúrua, L. C. Pimenta, and G. M. Freitas, "Autonomous navigation system for a delivery drone," *Journal of Control, Automation and Electrical Systems*, vol. 33, pp. 141–155, 2022.

[16] I. Ünal and M. Topakci, "Design of a remote-controlled and GPS-guided autonomous robot for precision farming," *International Journal of Advanced Robotic Systems*, vol. 12, no. 12, p. 194, 2015. [Online]. Available: https://doi.org/10.5772/62059

[17] Y. Xiong, C. Peng, L. Grimstad, P. J. From, and V. Isler, "Development and field evaluation of a strawberry harvesting robot with a cable-driven gripper," *Computers and electronics in agriculture*, vol. 157, pp. 392–402, 2019.

[18] Y. Xiong, Y. Ge, L. Grimstad, and P. J. From, "An autonomous strawberry-harvesting robot: Design, development, integration, and field evaluation," *Journal of Field Robotics*, vol. 37, no. 2, pp. 202–224, 2020.

[19] K. Shaik, E. Prajwal, S. B., M. Bonu, and B. V. Reddy, "GPS based autonomous agricultural robot," in *2018 International Conference on Design Innovations for 3Cs Compute Communicate Control (ICDI3C)*, 2018, pp. 100–105.

[20] R. Moeller, T. Deemyad, and A. Sebastian, "Autonomous navigation of an agricultural robot using RTK GPS and Pixhawk," in *2020 Intermountain Engineering, Technology and Computing (IETC)*, 2020, pp. 1–6.

[21] Y. Ding, L. Wang, Y. Li, and D. Li, "Model predictive control and its application in agriculture: A review," *Computers and Electronics in Agriculture*, vol. 151, pp. 104–117, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0168169917315296

[22] K. D. Sowjanya, R. Sindhu, M. Parijatham, K. Srikanth, and P. Bhargav, "Multipurpose autonomous agricultural robot," in *2017 International conference of Electronics, Communication and Aerospace Technology (ICECA)*, vol. 2, 2017, pp. 696–699.

[23] K. A. M. Almendral, R. M. G. Babaran, B. J. C. Carzon, K. P. K. Cu, J. M. Lalanto, and A. C. Abad, "Autonomous fruit harvester with machine vision," *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, vol. 10, no. 1-6, pp. 79–86, 2018.

[24] H. Zhou, X. Wang, W. Au, H. Kang, and C. Chen, "Intelligent robots for fruit harvesting: Recent developments and future challenges," *Precision Agriculture*, vol. 23, no. 5, pp. 1856–1907, 2022.

[25] S. M. Mangaonkar, R. Khandelwal, S. Shaikh, S. Chandaliya, and S. Ganguli, "Fruit harvesting robot using computer vision," in *2022 International Conference for Advancement in Technology (ICONAT)*, 2022, pp. 1–6.

[26] "Localization strategies for autonomous mobile robots: A review," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 8, Part B, pp.

6019–6039, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1319157821000550

[27] H. Hu and D. Gu, "Landmark-based navigation of industrial mobile robots," *Industrial Robot: An International Journal*, vol. 27, 07 2004.

[28] R. Madhavan and H. Durrant-Whyte, "Natural landmark-based autonomous vehicle navigation," *Robotics and Autonomous Systems*, vol. 46, pp. 79–95, 02 2004.

[29] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with Rao-blackwellized particle filters," *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[30] B. xu, Y. Fu, C. Zhang, and Z. Liu, "Research of cartographer laser SLAM algorithm," 11 2017, p. 104. [Online]. Available: 10.1117/12.2292864

[31] N. Habibie, A. M. Nugraha, A. Z. Anshori, M. A. Ma'sum, and W. Jatmiko, "Fruit mapping mobile robot on simulated agricultural area in Gazebo simulator using simultaneous localization and mapping (SLAM)," in *2017 International Symposium on Micro-NanoMechatronics and Human Science (MHS)*, 2017, pp. 1–7.

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

[33] Z. Zhao, Y. Zhang, L. Long, Z. Lu, and J. Shi, "Efficient and adaptive lidar–visual–inertial odometry for agricultural unmanned ground vehicle," *International*

Journal of Advanced Robotic Systems, vol. 19, no. 2, p. 17298806221094925, 2022. [Online]. Available: https://doi.org/10.1177/17298806221094925

[34] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, no. 5, p. 1147–1163, Oct 2015. [Online]. Available: http://dx.doi.org/10.1109/TRO.2015.2463671

[35] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017. [Online]. Available: https://doi.org/10.48550/arXiv.1610.06475

[36] B. M. F. da Silva, R. S. Xavier, T. P. do Nascimento, and L. M. Goncalves, "Experimental evaluation of ROS compatible SLAM algorithms for RGB-D sensors," in *2017 Latin American Robotics Symposium (LARS) and 2017 Brazilian Symposium on Robotics (SBR)*, 2017, pp. 1–6. [Online]. Available: 10.1109/SBR-LARS-R.2017.8215331

[37] O. Ozisik and S. Yavuz, "An occupancy grid based SLAM method," in *2008 IEEE International Conference on Computational Intelligence for Measurement Systems and Applications*, 2008, pp. 117–119.

[38] A. Pfrunder, P. V. K. Borges, A. R. Romero, G. Catt, and A. Elfes, "Real-time autonomous ground vehicle navigation in heterogeneous environments using a 3d lidar," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 2601–2608.

[39] J. Weingarten and R. Siegwart, "EKF-based 3D SLAM for structured environment reconstruction," in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005, pp. 3834–3839.

[40] D. M. Cole and P. M. Newman, "Using laser range data for 3D SLAM in outdoor environments," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, 2006, pp. 1556–1563.

[41] M. Tomono, "Robust 3D SLAM with a stereo camera based on an edge-point icp algorithm," in *2009 IEEE International Conference on Robotics and Automation*, 2009, pp. 4306–4311.

[42] D. Droeschel and S. Behnke, "Efficient continuous-time SLAM for 3D LiDAR-based online mapping," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 5000–5007.

[43] F. Moosmann and C. Stiller, "Velodyne SLAM," in *2011 IEEE Intelligent Vehicles Symposium (IV)*, 2011, pp. 393–398.

[44] J. Weingarten and R. Siegwart, "3D SLAM using planar segments," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006, pp. 3062–3067.

[45] P. Kim, J. Chen, and Y. K. Cho, "SLAM-driven robotic mapping and registration of 3d point clouds," *Automation in Construction*, vol. 89, pp. 38 – 48, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0926580517303990

[46] C. Brand, M. J. Schuster, H. Hirschmüller, and M. Suppa, "Stereo-vision based obstacle mapping for indoor/outdoor SLAM," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, pp. 1846–1853.

[47] D. Schleicher, L. M. Bergasa, M. Ocana, R. Barea, and M. E. Lopez, "Real-time hierarchical outdoor SLAM based on stereovision and GPS fusion," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 3, pp. 440–452, 2009.

[48] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.

[49] O. Brock, J. Trinkle, and F. Ramos, *Improving Localization Robustness in Monocular SLAM Using a High-Speed Camera*, 2009, pp. 183–190. [Online]. Available: https://ieeexplore.ieee.org/document/6284860

[50] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 834–849.

[51] R. Li, S. Wang, and D. Gu, "DeepSLAM: A robust monocular SLAM system with unsupervised deep learning," *IEEE Transactions on Industrial Electronics*, pp. 1–1, 2020.

[52] S. Wen, M. Sheng, C. Ma, Z. Li, H.-K. Lam, Y. Zhao, and J. Ma, "Camera recognition and laser detection based on EKF-SLAM in the autonomous navigation of humanoid robot," *Journal of Intelligent and Robotic Systems*, vol. 92, pp. 1–13, 10 2018.

[53] Sheng Fu, Hui-ying Liu, Lu-fang Gao, and Yu-xian Gai, "SLAM for mobile robots using laser range finder and monocular vision," in *2007 14th International Conference on Mechatronics and Machine Vision in Practice*, 2007, pp. 91–96.

[54] S. Yang, X. Yi, Z. Wang, Y. Wang, and X. Yang, "Visual SLAM using multiple RGB-D cameras," in *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2015, pp. 1389–1395.

[55] S. Li and D. Lee, "RGB-D SLAM in dynamic environments using static point weighting," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 2263–2270, 2017.

[56] R. Reid, A. Cann, C. Meiklejohn, L. Poli, A. Boeing, and T. Braunl, "Cooperative multi-robot navigation, exploration, mapping and object detection with ros," in *2013 IEEE Intelligent Vehicles Symposium (IV)*, 2013, pp. 1083–1088.

[57] M. Köseoğlu, O. M. Çelik, and Pektaş, "Design of an autonomous mobile robot based on ros," in *2017 International Artificial Intelligence and Data Processing Symposium (IDAP)*, 2017, pp. 1–5.

[58] N. Jain, A. K. Gupta, and P. Mathur, "Autonomous drone using ros for surveillance and 3d mapping using satellite map," in *Proceedings of the Second International Conference on Information Management and Machine Intelligence*, D. Goyal, A. K. Gupta, V. Piuri, M. Ganzha, and M. Paprzycki, Eds. Singapore: Springer Singapore, 2021, pp. 255–266.

[59] Y. Zhang, Z. Qiu, T. Yao, D. Liu, and T. Mei, "Fully convolutional adaptation networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[60] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia, "Icnet for real-time semantic segmentation on high-resolution images," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[61] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *CoRR*, vol. abs/1606.00915, 2016.

[62] E. Romera, J. M. Álvarez, L. M. Bergasa, and R. Arroyo, "Erfnet: Efficient residual factorized convnet for real-time semantic segmentation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 1, pp. 263–272, 2018.

[63] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "BiSeNet: Bilateral segmentation network for real-time semantic segmentation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018. [Online]. Available: https://doi.org/10.48550/arXiv.1808.00897

[64] C. Yu, C. Gao, J. Wang, G. Yu, C. Shen, and N. Sang, "BiSeNet v2: Bilateral network with guided aggregation for real-time semantic segmentation," 2020. [Online]. Available: https://doi.org/10.48550/arXiv.2004.02147

[65] Y. Bai, L. Fan, Z. Pan, and L. Chen, "Monocular outdoor semantic mapping with a multi-task network," 2019. [Online]. Available: https://doi.org/10.48550/arXiv.1901.05807

[66] C. Zhao, L. Sun, P. Purkait, and R. Stolkin, "Dense RGB-D semantic mapping with pixel-voxel neural network," 2017. [Online]. Available: https://doi.org/10.48550/arXiv.1710.00132

[67] B. Xu, W. Li, D. Tzoumanikas, M. Bloesch, A. Davison, and S. Leutenegger, "Mid-fusion: Octree-based object-level multi-instance dynamic SLAM," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 5231–5237.

[68] B. Romera-Paredes and P. H. S. Torr, "Recurrent instance segmentation," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., 2016.

[69] M. Halber, Y. Shi, K. Xu, and T. Funkhouser, "ReScan: Inductive instance segmentation for indoor rgbd scans," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[70] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-based convolutional networks for accurate object detection and segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 1, pp. 142–158, 2016.

[71] R. Girshick, "Fast R-CNN," 2015. [Online]. Available: https://doi.org/10.48550/arXiv.1504.08083

[72] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," 2016. [Online]. Available: https://doi.org/10.48550/arXiv.1506.01497

[73] T. Hackel, J. D. Wegner, and K. Schindler, "Fast semantic segmentation of 3d point clouds with strongly varying density," *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. III-3, pp. 177–184, 2016. [Online]. Available: https://www.isprs-ann-photogramm-remote-sens-spatial-inf-sci.net/III-3/177/2016/

[74] D. Xu, D. Anguelov, and A. Jain, "PointFusion: Deep sensor fusion for 3d bounding box estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[75] X. Kong, G. Zhai, B. Zhong, and Y. Liu, "Pass3d: Precise and accelerated semantic segmentation for 3d point cloud," *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Nov 2019. [Online]. Available: http://dx.doi.org/10.1109/IROS40897.2019.8968296

[76] "SqueezeSeg: Convolutional Neural Nets with Recurrent CRF for Real-Time Road-Object Segmentation from 3D LiDAR Point Cloud, author=Bichen Wu and Alvin Wan and Xiangyu Yue and Kurt Keutzer, year=2017, url=https://doi.org/10.48550/arXiv.1710.07368."

[77] Y. Wang, T. Shi, P. Yun, L. Tai, and M. Liu, "Pointseg: Real-time semantic segmentation based on 3D LiDAR point cloud," 2018. [Online]. Available: https://doi.org/10.48550/arXiv.1807.06288

[78] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, "Rangenet ++: Fast and accurate lidar semantic segmentation," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 4213–4220.

[79] L. Schaupp, M. Bürki, R. Dubé, R. Siegwart, and C. Cadena, "Oreos: Oriented recognition of 3d point clouds in outdoor scenarios," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 3255–3261.

[80] J. Jeong, T. S. Yoon, and J. B. Park, "Multimodal sensor-based semantic 3D mapping for a large-scale environment," 2018. [Online]. Available: https://doi.org/10.48550/arXiv.1802.10271

[81] A. Dewan and W. Burgard, "DeepTemporalSeg: Temporally consistent semantic segmentation of 3d lidar scans," 2020. [Online]. Available: https://doi.org/10.48550/arXiv.1906.06962

[82] F. Zhang, C. Guan, J. Fang, S. Bai, R. Yang, P. H. S. Torr, and V. Prisacariu, "Instance segmentation of LiDAR point clouds," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 9448–9455.

[83] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," 2017, pp. 652–660. [Online]. Available: https://doi.org/10.48550/arXiv.1612.00593

[84] W. Maddern and P. Newman, "Real-time probabilistic fusion of sparse 3D LiDAR and dense stereo," 10 2016, pp. 2181–2188. [Online]. Available: 10.1109/IROS.2016.7759342

[85] Y. Wang, R. Ji, and S. Chang, "Label propagation from ImageNet to 3D point clouds," in *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 3135–3142.

[86] W. Wang and U. Neumann, "Depth-aware CNN for RGB-D segmentation," 2018. [Online]. Available: https://doi.org/10.48550/arXiv.1803.06791

[87] B. H. Wang, W.-L. Chao, Y. Wang, B. Hariharan, K. Q. Weinberger, and M. Campbell, "LDLS: 3-D object segmentation through label diffusion from 2-D images," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, p. 2902–2909, Jul 2019. [Online]. Available: http://dx.doi.org/10.1109/LRA.2019.2922582

[88] G. Narita, T. Seno, T. Ishikawa, and Y. Kaji, "PanopticFusion: Online volumetric semantic mapping at the level of stuff and things," 2019. [Online]. Available: https://doi.org/10.48550/arXiv.1903.01177

[89] F. Li, M. Ding, J. Takamatsu, and T. Ogasawara, "Static 3D map reconstruction based on image semantic segmentation," in *2018 15th International Conference on Ubiquitous Robots (UR)*, 2018, pp. 583–585.

[90] B. Bescos, J. M. Facil, J. Civera, and J. Neira, "DynaSLAM: Tracking, mapping, and inpainting in dynamic scenes," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, p. 4076–4083, Oct 2018. [Online]. Available: http://dx.doi.org/10.1109/LRA.2018.2860039

[91] M. Runz, M. Buffier, and L. Agapito, "MaskFusion: Real-time recognition, tracking and reconstruction of multiple moving objects," in *2018 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2018, pp. 10–20. [Online]. Available: 10.1109/ISMAR.2018.00024

[92] E. Palazzolo, J. Behley, P. Lottes, P. Giguère, and C. Stachniss, "ReFusion: 3D reconstruction in dynamic environments for RGB-D cameras exploiting residuals," 2019. [Online]. Available: https://doi.org/10.48550/arXiv.1903.01177