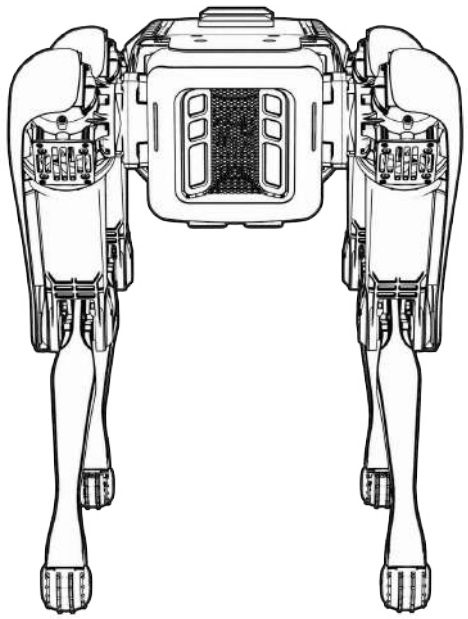
The Yellow Retriever

An AI-powered robotic dog assistant for efficient lost item retrieval



Team 18

Lars Bosma	[redacted]
Denniz Goren	[redacted]
Andrei-Carlo Papuc	[redacted]
Joop Sluimer	[redacted]
Jelle van Rijn	[redacted]

Monday 21st August, 2023

Case: TNO

TU Delft

Contents

Abstract

In a typical home environment objects are frequently misplaced and lost. Some humans need assistance in retrieving lost objects, in cluttered environments, because they might not be able to do it themselves, without damaging objects in the environment or other humans. To tackle this problem Team Yellow Retriever together with Spot the agile Boston Dynamics robot will work for a 8 week period, in 4 sprints, making use of agile project management. The devised solution consists of four modules: perception, mapping, navigation and decision handling, together with a finite state machine to handle interactions between the different modules. With the solution, Spot was able to move around in an unknown environment, detect both objects and people according to a task defined upon initialization, and deliver the found object to the corresponding human. Given more time, the solution could have been improved, such that the integration of the entire task works equally well as all the individual modules. The solution proposed by Team Yellow Retriever is a feasible solution to the problem statement, that currently partially works, but given more time for development Team Yellow Retriever is confident in the solution's ability to handle the defined task.

MEET THE TEAM

In this chapter the members of team Yellow Retriever will introduce themselves and highlight their strengths and weaknesses regarding the project. The team agreements on meetings and team roles are also shown in this chapter.

1.1 Member introductions

Joop Sluimer: I was assigned the HRI role. I believe my strengths are focused on organisation, taking initiative and having good communication skills to ensure we progress as planned. I also am very comfortable with Python which could be very useful for the project. My strengths according to the Team Roles Test [4] are ”diplomat”, ”actualiser” and ”coach.” Due to my initiative skills and also having a strong urge to finish work on time I was assigned the additional role of Communications Director, ensuring that individuals maintain contact with the group and notify the other of their progress.

Jelle van Rijn: I was assigned one of the two perception roles. More specifically, I am responsible for the detection of people. I am fairly confident with Python and I have a special interest in computer vision. My soft skill strengths lay in analysis and keeping the broad picture in mind. I scored high on ”actualiser” and ”judge-appraiser” on the Team Roles Test [4]. For this reason, I was assigned the role of Risk Mitigation Officer.

Lars Bosma: I was assigned the other perception role; the object detection specialist. I believe my greatest strengths to be coding in Python and applying finishing touches at the end of team projects. The Team Roles Test [4] indicated that I am an ”expert” with regard to thinking-oriented roles, meaning that I tend to immerse myself in a field that aligns with my interests. Regarding action-oriented roles, I am a ”finaliser,” which means that I am prepared to take the extra step to make sure the final work is as perfect as it can be. For this reason, my role in the team, apart from perception specialist, is Quality Assurance Officer.

Denniz Goren: I was assigned one of the two planning and navigation roles. I enjoy working together on projects as a group and can work well with different personalities. My passion lies in coding, and I am always eager to dive into a project and discover innovative solutions. My Team Roles Test[4] results indicated that I possess a balanced mix of skills and traits, with no particular role dominating the others. However, my strengths as an ”actualiser” and ”finaliser” stood a bit out, which led me to take on the role of Systems Engineer for this project.

Andrei-Carlo Papuc: I was assigned the role of planning and navigation engineer. I have some experience and a strong interest in the areas of optimal-control, reinforcement learning, and computer vision. Alongside my technical abilities, I always try to be a team player and work well with others. Based on the Team Roles test [4], I scored high on the ”connector” and ”coach” points, therefore, this led for me to take on the role of Resource Officer.

Figure 1.1: Team Yellow Retriever logo, created with Craiyon [2].

1.2 Team agreements

Our expectations for the course are well aligned, everyone is eager to dedicate the highest effort possible and also put in the necessary time. Given our very varying schedules we agreed to meet up every Tuesday and Wednesday afternoon, but also expect work to be done outside of sessions. The meetings represent time allocated to discuss progress and possibly resolve any encountered issues. Furthermore, we agreed that everyone will try to support each department when needed, yet the person assigned the role will hold the final responsibility of their own department.

After meeting our client in week 1, it was clarified that the motion control specialist role was redundant for this project. Therefore, Jelle, who was in charge of motion control at first, was assigned a secondary perception specialist role. We have decided that the two roles will be human and object perception specialists due to the task being slightly different yet equally important throughout the project.

The chosen project management framework will be Agile. This project will run across four sprints in which requirements, plans and results will be continuously evaluated such that the team can respond quickly to changes or the clients' feedback.

In order to give every member of the team equal experience with leadership, the decision was made that the team leader role will rotate on a weekly basis, see Table 1.1. The team leader will set out the goals for the sprint and aid in the task assignment between the team members if needed. The team leader is also responsible for mitigating any potential conflicts within the group. This will be in the form of addressing certain issues and guiding for conversion in case of misaligned visions regarding the project.

Table 1.1: Team leader role rotational schedule.

Week #	Team Leader
Week 2	Lars Bosma
Week 3	Andrei-Carlo Papuc
Week 4	Joop Sluimer
Week 5	Denniz Goren
Week 6	Jelle van Rijn
Week 7	Lars Bosma
Week 8	Andrei-Carlo Papuc

PROJECT GOAL

During this project, we are going to work on a case for TNO [11] , an independent scientific research organisation in the Netherlands. TNO currently works on novel robotic applications, that arise from developments in this realm. One of those new developments, that TNO works with, is Spot the robot dog from Boston Dynamics. TNO aims to use the robotic dog for the detection and retrieval of objects to specific humans and requested us to make the robot more intelligent, in order for it to be able to do this task efficiently.

In the following section, the problem is formally defined and the project plan is laid out according to the chosen project management framework. Moreover, a market research is conducted and all stakeholders are considered from numerous points of view. A small number of solutions that arise from brainstorming are analysed in relationship to the client SWOT, and an operational scenario is presented to better put into perspective the end product.

2.1 Problem Definition

The client has listed a set of requirements for the capabilities of this system, which are presented in Table 2.1. Each requirement has a unique identifier that will be used moving forward to refer to that requirement specifically. In order to ensure that requirements can be verified, each requirement is also accompanied by one or more verification methods. The different options include test (T), analysis (A), inspection (I), and demonstration (D) because they are the primary requirement verification techniques.

Table 2.1: Requirements and needs of the client summarised.

ID	Requirement description	Verification Method
STH.01	The system shall make use of the robot Spot[1], from Boston Dynamics.	D
STH.02	The system shall be able to operate autonomously.	T,D
STH.03	The system shall comply with all relevant governmental legislations.	I
STH.04	The system shall require no more frequent maintenance than once per month.	I
STH.05	The system shall not reduce Spot's operational lifetime.	A
STH.06	The system shall be able to differentiate between multiple humans when scanning its environment.	T
STH.07	The system shall interact with humans without inflicting any harm.	I,D

Once the initial requirements and needs of the client have been cleared, the following problem definition can be formulated:

- *In a typical home environment, objects are frequently misplaced and lost. Some humans need assistance in retrieving lost objects, in cluttered environments, because they might not be able to do it themselves, without damaging objects in the environment or other humans.*

2.2 Project Plan

Both a Work Breakdown Structure (WBS) and Gantt chart were created for this project for the purpose of organisation and attaining a better overview of all upcoming tasks. These can be found in Appendix Chapters B and C.

[-1] See https://www.bostondynamics.com/products/spot
[0] See https://www.bostondynamics.com/products/spot
[1] See https://www.bostondynamics.com/products/spot

2.3 Vision

When defining the main problem, in parallel brainstorming sessions frequently occurred. While looking at the list of requirements, and the problem definition, multiple solutions were thought of to solve the same problem in different ways.

The following concepts were generated and are present in the list below:

- Spot would first find the object and lead the person to the object by grabbing their hand using the gripper
- Spot would find the object and bring it to the person in a fully autonomous way
- Spot finds the object given a voice command and then would guide the person using voice control to traverse the cluttered environment
- Spot searches for the object and alerts when it is found, and communicates its position relative to the user

Market & SWOT analysis
As mentioned in the course manual [12], a SWOT analysis is a great approach to determine spaces in the market that can be filled, as well as the risks that could come when taking the system to market. The analysis will identify: strengths, weaknesses, opportunities and threats of the client to which the system at hand is applicable.

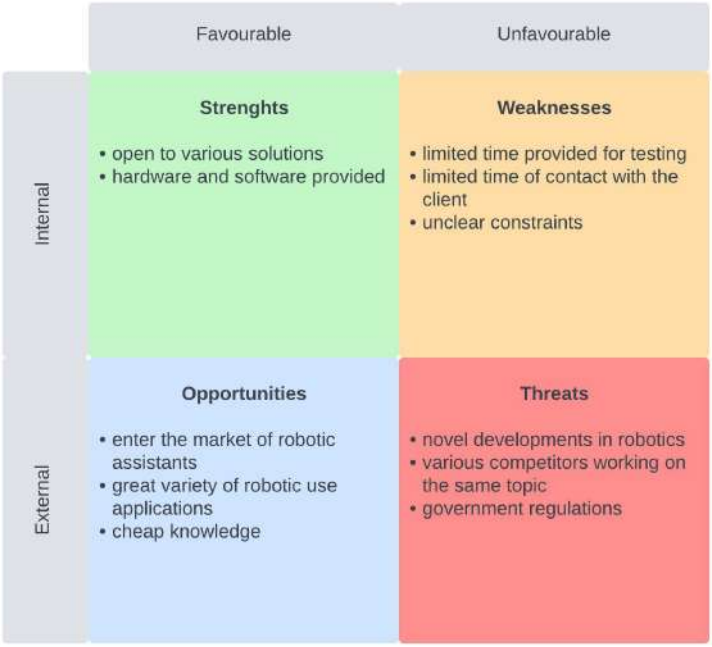


Figure 2.1: SWOT analysis of the client[2].

Compared to competitors, Spot has an advantage because it is very versatile. It is able to reach places other robots might not. Our solution is a solution for a simple problem, but it could have potential in dangerous environments, such as nuclear power plants or environments where people cannot leave their place. In these scenarios, it is extra important that Spot is able to find and bring the object autonomously. Having a solution that is able to do so will benefit TNO because it shows the potential for these types of environments. The versatility of Spot has another benefit, since it can perform other tasks with the object. This makes it more financially viable.

[2]Graphic created with Lucid.app

Ethics, safety and sustainability
For this project, we want to assure that the robot complies with regulations with regards to privacy and safety. We hope to achieve this by only collecting data with consent of the recorded and by making sure no data is stored or processed. Furthermore, our aim is to be transparent in how our software manages the data. The collision avoidance that is implemented by Boston Dynamics is already a tremendous benefit when it comes to safety. However, we would like to look further into additional safety measures and dive deeper into the limitations that the on-board systems have. Regarding sustainability, we want to make the robot as energy efficient as possible in performing its tasks. We hope to achieve this by adding this aspect to its cost function, with respect to its path planning and localisation of objects and humans.

Operational vision
After considering all stakeholders and based on the idea generation, the following operational story is presented to better visualise our vision for this project.

> I, an employee at company X, am working on an approaching deadline and do not have much time for anything else. Unfortunately, I have lost my lunchbox. Due to the pressing matter of this deadline I have no time to go look for it. There is a solution though, I know that we have Spot on our floor and I can tell Spot that I have lost my red lunchbox and it needs to be retrieved. Spot leaves my workstation looking for my lunchbox, while I continue working. After some time I see Spot approaching my workstation again with my lunchbox in its gripper.

2.4 Operational scenarios

Table 2.2: High level operational scenario

ID	Mission Phase	Description
S1	Startup Phase	The operator turns on Spot, allowing it to power all the subsystems and calibrate the body
S2	Manage Task	The operator tells Spot the object it needs to retrieve and to which human to deliver it.
S3	Scan & Search Object Phase	Scan the room and look for the desired object. Detect locations of humans while also moving around the room. Human search of second priority.
S3-1	Scan & Search Object Failed	Announce an infeasible task and return to the origin.
S4	Pick Object	Spot moves to the location of the object. Operator manually operates the gripper to pick up the desired object, as proposed by the client and as object detection using the gripper's camera is not required. Spot then returns to a default pose to continue scanning.
S5	Search Human Phase	If the human has not been previously identified in S3, look for the human to deliver the object to.
S5-1	Search Human Failed	If the correct human was not found at all, Spot returns to the origin with the object in hand. Drops the object at the origin. The operator is notified of the object's new location.
S6	Deliver Object	Spot moves to the location of the human. Human manually operates the gripper to place the object in front of the human.
S7	Return to origin	Once the object has been retrieved to the correct human, return to origin for a new task.
S8	Shut down	Once there are no more tasks to be completed, turn off Spot.
S9	Emergency stop	If any unwanted behaviour occurs, Spot stops its current action and the operator is notified.

With the vision laid out in great detail, and given basic operational scenarios from the view of a stakeholder, operational phases of the product can be described. Table 2.2 presents a high level overview of the operational scenarios from the initialisation of the robot, to searching and fetching the lost item, to shutting down. Additional subphases such as **S3-1** and **S5-1** represent backup scenarios to previously infeasible or faulty scenarios.

2.5 Nodes

Taking inspiration from the operational scenarios, a list of nodes needs to be developed to achieve the desired functionality of the robot. The following high level overview of operational nodes is presented in Figure 2.2, these will be developed in *ROS* [5].

The nodes are divided into multiple colours representing the different responsible modules, these being: mapping, navigation, decision handler and perception. Mapping will be responsible for taking Spot's depth camera and transforming the data into a pointcloud map of its environment. This created map will be used to allow Spot to explore more of the environment and also eventually search for the object and human together with perception. Navigation is responsible for moving Spot around its environment together with the created map to calculated points in the world for efficient space exploration. Decision handler is responsible for the state machine of Spot, that decides what Spot should do at each moment given results from different modules. It will also be responsible for initialising the user-defined task and the graphical user interface that the user will interact with the most. ROS actions will be used for communication between the different modules. Finally, perception will be responsible for object and human detection making use of Spot's cameras. The detections will also be used to move Spot through the world in case the defined task can be completed by delivering the object to the correct human. The information of the object and human location, obtained through the perception module, will be sent to the decision handler in order to move to the corresponding locations. Once the correct object has been delivered to the defined human at the initialisation, the task is completed.

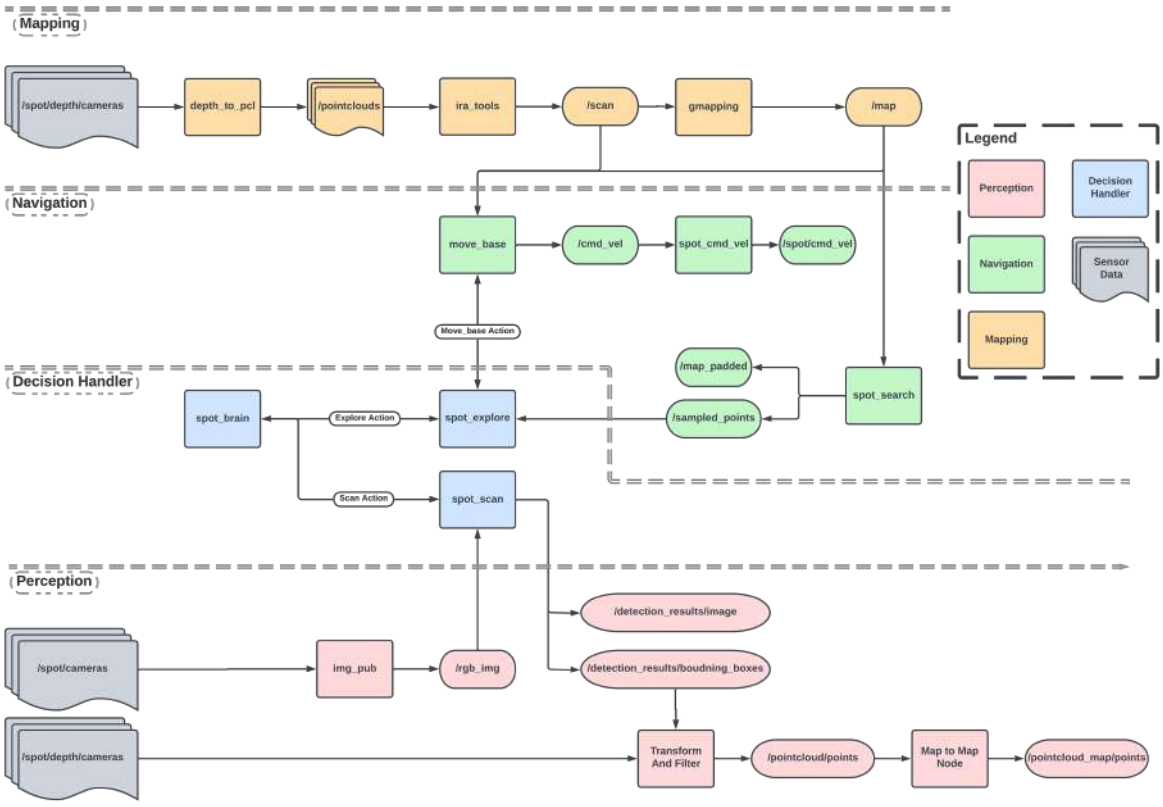


Figure 2.2: An overview of the packages, nodes and topics.

FUNCTIONAL ARCHITECTURE

This section continues on work provided in chapter 2. The subsections handle an extensive version of the operational scenarios, needs and functional requirements for these scenarios, a functional hierarchy tree and an activity diagram. This was all done to ensure code development in the future goes smoother, as many of the requirements have already been thought about and considered.

3.1 Extended operational scenarios

This section is an extension on the operational scenarios in section 2.4. Each scenario is extended by showing the preconditions, the triggering events, giving a description and by listing the postconditions, as can be seen in Table 3.1 through Table 3.11. The preconditions inform us on when a task is able to take place. The triggering event clarifies the event that actually makes Spot engage in the corresponding task. The description informs us what actually happens once the task has taken place. Finally, the postconditions specify the conditions that need to be satisfied for the task to be completed.

Table 3.1: Extended operational scenario S1

Operational Scenario:	**S1 Startup Phase**
Preconditions	1. Safe environment. 2. All object and humans are in static positions.
Triggering event	1. Operator manually presses the startup button.
Description	Spot will manually get started up by the operator. Spot's cooling fans will start making a lot of noise. During this time, Spot's hosted WiFi network is being initialised and its computers are booting. This takes about two minutes
Postconditions	1. Once Spot's cooling fans stops making a lot of noise, the operator is able to connect to Spot's hosted WiFi network and the Spot's front lights start flashing rainbow, the startup phase has been completed. 2. If any unwanted behavior is registered, continue to S9.

Table 3.2: Extended operational scenario S2

Operational Scenario:	**S2 Manage Task**
Preconditions	1. Startup phase has been completed. 2. Operator is ready to communicate with the robot.
Triggering event	1. Spot requests for its task to be initialised through a graphic user interface. 2. The operator communicates the desired object to be picked up and the corresponding person to deliver it to.
Description	The operator communicates which object needs to be picked up and delivered to which human through a graphic user interface. Spot shows the defined object and humans to the operator such that they can check that the task has been successfully communicated to Spot.
Postconditions	1. Spot has successfully communicated the task back to the operator. 2. The manage task phase is completed. 3. If any unwanted behavior is registered, continue to S9.

Table 3.3: Extended operational scenario S3

Operational Scenario:	S3 Scan and Search Object Phase
Preconditions	1. Manage task phase has been completed. 2. Spot is able to walk around.
Triggering event	1. Spot knows which object to look for and to whom it must be delivered. Also, the operator has requested Spot to scan the environment and look for the desired object.
Description	Spot will walk around and look for the desired object. Simultaneously, Spot will scan and map the environment whilst the object has not been found. In the case that a human/person gets detected, Spot will register this person's location in its mapping for future path planning and navigation. Spot will also register the color of the person's t-shirt or sweater. Once Spot detects the object, it walks towards it.
Postconditions	1. Once the object is within the range of the gripper, the scan and search object phase is completed. 2. If any unwanted behavior is registered, continue to S9.

Table 3.4: Extended operational scenario S3-1

Operational Scenario:	S3-1 Search Object Phase Failed
Preconditions	1. Spot has scanned and searched the entire environment and was unable to detect the desired object. 2. Spot is able to walk around.
Triggering event	1. Scan and search object phase has failed.
Description	Spot will announce that the scan and search object phase is infeasible and returns to its starting position (the origin).
Postconditions	1. After returning to the starting position, Spot will reinitialise the scan and search object phase if requested by the operator. 2. If any unwanted behavior is registered, continue to S9.

Table 3.5: Extended operational scenario S4

Operational Scenario:	S4 Pick Object
Preconditions	1. Scan and search object phase has been successfully completed. 2. Spot's gripper is functional. 3. The object is within the gripper's reach.
Triggering event	1. Scan and search object phase has been successfully completed. 2. The operator has requested Spot to pick up the desired object.
Description	The operator manually uses Spot's gripper to pick the object up. The operator does so by using the tablet. Once the object has been picked up, the gripper returns to its starting position.
Postconditions	1. Once the gripper has returned to its starting position, the task is completed. 2. If any unwanted behavior is registered, continue to S9.

Table 3.6: Extended operational scenario S5

Operational Scenario:	S5 Scan and Search Human Phase
Preconditions	1. Spot is holding the desired object in its gripper. 2. Spot is able to walk around.
Triggering event	1. The operator has requested Spot to locate the human corresponding to the object in the gripper.
Description	If the correct human was successfully located in S3, then Spot will walk towards this human. If no humans were found during S3, then Spot will continue scanning from its current position for humans in the same manner as that objects are searched for. If Spot finds a human, Spot will check if it is the right human, to whom the corresponding object must be delivered to, based on the color of the person's t-shirt or sweater. If it is not the right human, then Spot continues its search.
Postconditions	1. Once Spot has located and confirmed that Spot has found the right human, the scan and search human phase is completed. 2. If any unwanted behavior is registered, continue to S9.

Table 3.7: Extended operational scenario S5-1

Operational Scenario:	S5-1 Scan and Search Human Phase Failed
Preconditions	1. Spot has scanned and searched the entire environment and was unable to detect any humans. 2. Spot is able to walk around.
Triggering event	1. Scan and search human phase has failed.
Description	Spot has searched and scanned the entire room without locating the right human. It immediately returns to its starting position (the origin). Once it has reached this position, then it drops the object in his gripper, at the origin as well, and notifies the operator of the object's new location. The object has been located, but localisation of the corresponding person has failed.
Postconditions	1. The global task is terminated and the operator is notified and must intervene. 2. If any unwanted behavior is registered, continue to S9.

Table 3.8: Extended operational scenario S6

Operational Scenario:	S6 Deliver Object
Preconditions	1. Scan and search human phase is completed. 2. Spot's gripper is functional. 3. Spot is able to walk around.
Triggering event	1. The human has requested Spot to deliver the object to them.
Description	Spot moves towards the human and uses its gripper to place the object in front of the human, if the human has requested Spot to do so. The distance between the object and the human is $\tilde{3}0$ cm. Once the object has been placed, the gripper returns to its original position.
Postconditions	1. Once the gripper has returned to its starting position, the deliver object phase is completed. 2. If any unwanted behavior is registered, continue to S9.

Table 3.9: Extended operational scenario S7

Operational Scenario:	S7 Return to Origin
Preconditions	1. The object has successfully been delivered to the correct human. 2. Spot is able to walk around.
Triggering event	1. The object has been delivered and the operator requests Spot to return to its starting position.
Description	Spot will take the fastest route to its starting position (the origin).
Postconditions	1. Once Spot is back at its starting position (the origin), the task is completed. If any unwanted behavior is registered, continue to S9.

Table 3.10: Extended operational scenario S8

Operational Scenario:	S8 Shut Down
Preconditions	1. Either all tasks have successfully been completed or the operator deems it necessary to turn Spot off.
Triggering event	1. The operator manually presses the power off button.
Description	Spot will power down on his starting position (the origin), shutting off all hardware and software.
Postconditions	1. Once Spot makes no noise anymore and connection with the laptop is lost, the task is completed.

Table 3.11: Extended operational scenario S9

Operational Scenario:	S9 Emergency Stop
Preconditions	1. Something goes wrong in any of the preceding phases.
Triggering event	1. Unwanted behavior is registered.
Description	Spot will stop all actions and notify the operator.
Postconditions	1. The operator walks towards Spot to determine what is going and why it possibly went wrong.

3.2 Needs and functional requirements

Once the operational scenarios have been detailed, and the pre-conditions as well as the post-conditions have been established with the triggering event per phase, the needs and requirements can be explored. Each requirement must have a unique identifier per phase, and needs to follow the SMART criteria. Table 3.12 until Table 3.22 show the needs established for each operational phase.

Table 3.12: Requirements for operational phase S1

ID	Operational needs for S1 ”Start up phase”
R1_1	The system shall be started up within 5 minutes
R1_2	The system shall indicate when the system has started up

Table 3.13: Requirements for operational phase S2

ID	Operational needs for S2 ”Manage task”
R2_1	The system shall be able to understand which object is desired via the input of the human
R2_2	The system shall ask for confirmation about understanding the task
R2_3	The system shall be able to understand the confirmation of the human
R2_4	The system shall be able to understand which human belongs to the object

Table 3.14: Requirements for operational phase S3

ID	Operational needs for S3 “Scan & Search Object Phase”
R3_1	The system shall be able to scan the environment
R3_2	The system shall be able to map scanned obstacles within 2 meters in front of the robot
R3_3	The system shall be able to recognise the objects “screwdriver”, “apple” and “ball”
R3_4	The system shall be able to register the location of a recognised human
R3_5	The system shall be able to recognise humans within 2 meters in front of the robot
R3_6	The system shall be able to register the location of a recognised object
R3_7	The system shall be able plan a collision free path towards the registered object
R3_8	The system shall be able to follow the found collision free path

Table 3.15: Requirements for operational phase S3-1

ID	Operational needs for S3-1 “Search Object Phase Failed”
R3-1_1	The system shall be able to indicate when object searching has failed
R3-1_2	The system shall be able to plan a collision free path towards a base position
R3-1_3	The system shall be able to follow the found collision free path towards a base position

Table 3.16: Requirements for operational phase S4

ID	Operational needs for S4 “Pick Object”
R4_1	The system shall pick up objects which are of the class “screwdriver”, “apple” or “ball” manually if there are no potential collisions while performing the task
R4_2	The system shall be able to adjust its position if there is a potential collision

Table 3.17: Requirements for operational phase S5

ID	Operational needs for S5 “Scan & Search Human Phase”
R5_1	The system shall scan and search the environment
R5_2	The system shall recognise humans in front of the robot
R5_3	The system shall map scanned humans within 2 meters in front of the robot
R5_4	The system shall register the location of a recognised human
R5_5	The system shall recognise the color of the humans upper body clothing
R5_6	The system shall determine the correct human based on the detected color
R5_7	The system shall notify the user in what state it is

Table 3.18: Requirements for operational phase S5_1

ID	Operational needs for S5-1 “Scan & Search Human Phase Failed”
R5-1_1	The system shall notify the user that the scanning and searching has failed
R5-1_2	The system shall plan a collision free path towards the origin
R5-1_3	The system shall follow the found collision free path
R5-1_4	The system shall notify the user in what state it is

Table 3.19: Requirements for operational phase S6

ID	Operational needs for S6 “Deliver Object”
R6_1	The system shall plan a collision free path towards the desired human
R6_2	The system shall follow the found collision free path
R6_3	The robot shall place the object in its gripper $\tilde{3}0$ cm in front of the human
R6_4	The gripper shall return to its original position
R6_5	The system shall notify the user in what state it is

Table 3.20: Requirements for operational phase S7

ID	Operational needs for S7 ”Return to Origin”
R7_1	The system shall plan a collision free path towards the origin
R7_2	The system shall follow the found collision free path
R7_3	The system shall notify the user in what state it is

Table 3.21: Requirements for operational phase S8

ID	Operational needs for S8 ”Shut Down”
R8_1	The system shall notify the user it is shutting down

Table 3.22: Requirements for operational phase S9

ID	Operational needs for S9 ”Emergency Stop”
R9_1	The system shall stop all actions
R9_2	The system shall notify the user of the registered unwanted behavior

3.3 Functional hierarchy tree

The functional requirements identified are translated into high-level system functionalities. The functional hierarchy tree is given in Figure 3.1, describing a top down overview of all functionalities. Each operational phase is split into multiple functionalities, for example, **F2** Manage Task, is split into **F2.1** Define Object and **F2.2** Define Human.

Moreover, Table 3.23 up until Table 3.57 further describe in detail each of the functions present in the hierarchy tree. The tables further describe what each function does, including its inputs and outputs. Having such an overview speeds up development, and provides a clear oversight of what the final product needs to accomplish in terms of functionality.

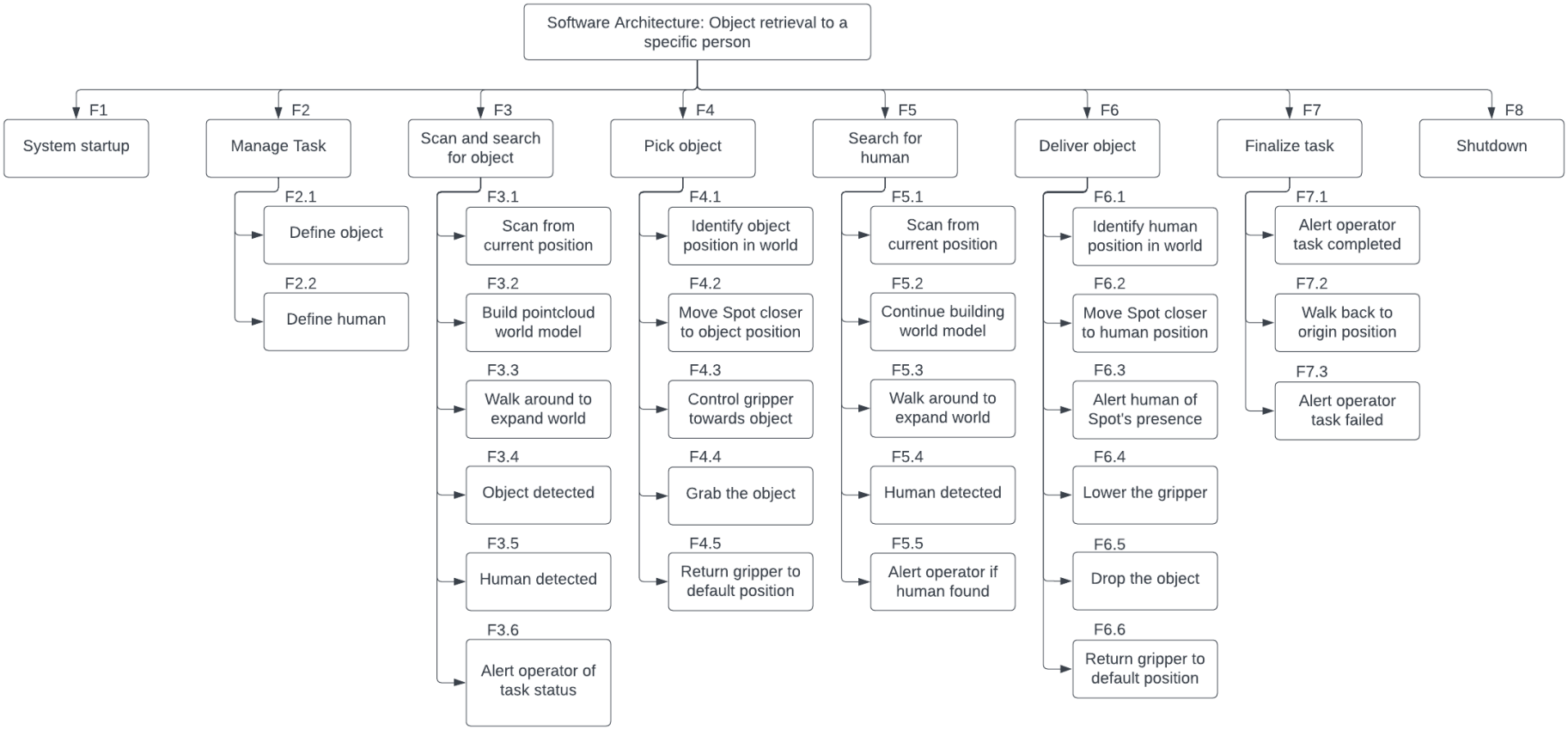


Figure 3.1: Functional hierarchy tree

Table 3.23: System startup

Label	Description
Name	System startup
Description	Spot manually gets started up by the operator
Inputs	Operator input to turn on Spot
Outputs	Active Spot
Parent function	Software Architecture
Child functions	

Table 3.24: Manage Task

Label	Description
Name	Manage task
Description	Initialisation of Spots task, define the object that has to be retrieved and also to whom to deliver the object to
Inputs	The operator can communicate which object needs to be picked up and delivered to which human.
Outputs	Communication of the current task for Spot to complete on the simulation, showing both the object and human. After confirmation Spot can continue
Parent function	Software Architecture
Child functions	

Table 3.25: Define object

Label	Description
Name	Define object
Description	Allow the operator to select one of multiple items to be chosen for Spot to be retrieved. Operator input of the chosen object.
Inputs	Spot communicates the chosen object, visually and audibly. Also publishes the chosen object information on a ROS topic.
Outputs	Spot confirms the chosen object, visually and audibly, and also publishes the chosen object information on a ROS topic.
Parent function	Manage Task
Child functions	

Table 3.26: Define human

Label	Description
Name	Define human
Description	Allow the operator to select one of multiple humans to be chosen for Spot to retrieve the previously chosen item to. Each human has features that are used to identify the human in the real world when scanning.
Inputs	Operator input of the chosen human
Outputs	Spot confirms the chosen human, visually and audibly, and also publishes the chosen human information on a ROS topic
Parent function	Manage Task
Child functions	

Table 3.27: Scan and Search for object

Label	Description
Name	Scan and Search for object
Description	Spot has to scan the world around it using its cameras. From this a world map can be created, and both objects and humans can be attempted to be detected. Both the camera images and the map are used to determine the position of objects and humans, required for Spot to complete its task.
Inputs	Chosen object and human information, camera images from Spot
Outputs	The next step to be executed, published on a ROS topic. If the object is perceived, publish to topic Spot is ready for picking the object. If the object is not perceived, publish to topic to continue searching, by moving and scanning a new area.
Parent function	Software Architecture
Child functions	Scan from current position, Build point cloud world model, Walk around to expand world, Object detected, Human detected, Alert operator if human/object found

Table 3.28: Scan from current position

Label	Description
Name	Scan from current position
Description	From Spots current world position, all camera images have to be received and published on ROS topics.
Inputs	Signal to start scanning and retrieve all camera images
Outputs	Camera images to be used for processing, published on a ROS topic
Parent function	Scan and Search for object
Child functions	

Table 3.29: Build pointcloud world model

Label	Description
Name	Build pointcloud world model
Description	Camera images are obtained from the scan. Build a point cloud world model that can be used for depth perception and object/human detection.
Inputs	The camera images from the ROS topic
Outputs	World point cloud published on a ROS topic
Parent function	Scan and Search for object
Child functions	

Table 3.30: Walk around to expand world

Label	Description
Name	Walk around to expand world
Description	Using the world model areas that are insufficiently scanned can be determined. Spot moves to these areas to improve the accuracy of the world model.
Inputs	World model point cloud, signal to continue searching
Outputs	Audio confirmation that Spot continues exploring the space
Parent function	Scan and Search for object
Child functions	

Table 3.31: Object detected

Label	Description
Name	Object detected
Description	Processing both the camera images and world model using YOLO, objects can be detected. If the object to be retrieved is found, publish to a topic the image location of the found object.
Inputs	The camera images from ROS topics
Outputs	Camera location of object in image, published to a ROS topic
Parent function	Scan and Search for object
Child functions	

Table 3.32: Human detected

Label	Description
Name	Human detected
Description	Processing both the camera images and world model using YOLO, humans can be detected. If the human to return the object to is found, publish to a topic the image location of the found object.
Inputs	The camera images from ROS topics
Outputs	Camera location of human, published to a ROS topic
Parent function	Scan and Search for object
Child functions	

Table 3.33: Alert operator of task status

Label	Description
Name	Alert operator of task status
Description	Audibly and visually tell the operator that an object/human has been found. In case that neither the human or the object is found several actions can be taken. With an incomplete world model, Spot can move around to scan more. With a complete model, Spot can notify the operator the task has failed, either quitting the task or attempt the task again.
Inputs	World model from ROS topic
Outputs	Visual and audio representation that task status (pass/fail)
Parent function	Scan and Search for object
Child functions	

Table 3.34: Pick object

Label	Description
Name	Pick object
Description	Using the camera location of the object, the actual world position of the object can be determined. Spot will have to move closer to this world position and then Spot uses its gripper to pick the object up. Once the object has been picked up, the gripper returns to its starting position.
Inputs	Camera image and location of object
Outputs	Object now in Spot's gripper
Parent function	Software Architecture
Child functions	

Table 3.35: Identify object position in world

Label	Description
Name	Identify object position in world
Description	Together with the world model and the camera location of the object, the world position can be determined. To be done either with the depth cameras or other perception techniques
Inputs	Camera image and location of object
Outputs	World position of object published on ROS topic
Parent function	Pick object
Child functions	

Table 3.36: Move Spot closer to object position

Label	Description
Name	Move Spot closer to object position
Description	The world position of the object is known. Spot has to move close enough to the position in order for the gripper to be able to grab the object.
Inputs	World position of object
Outputs	Ready signal for gripper to move towards object
Parent function	Pick object
Child functions	

Table 3.37: Control gripper towards object

Label	Description
Name	Control gripper towards object
Description	The world position of the object is known. Spot also in range of the object, the gripper moves close enough to the object to grab it.
Inputs	World position of object, ready signal from Spot being in range of object
Outputs	Ready signal for gripper to grab the object
Parent function	Pick object
Child functions	

Table 3.38: Grab the object

Label	Description
Name	Grab the object
Description	With the gripper in place, the gripper only has to close to grasp the object.
Inputs	Ready signal for grasping
Outputs	Physical object in Spot's gripper
Parent function	Pick object
Child functions	

Table 3.39: Return gripper to default position

Label	Description
Name	Return gripper to default position
Description	Object is in Spot's grippers. The gripper must return to its original position in order for Spot to continue searching for the human. Deliver the object to the human.
Inputs	Signal indicating the gripper has grabbed the object.
Outputs	Physical object in Spot's gripper now in the default position, ready to continue operating
Parent function	Pick object
Child functions	

Table 3.40: Search for human

Label	Description
Name	Search for human
Description	If the human has not yet been located, Spot has to continue scanning the world around it using its cameras. From this a world map can be updated, and the human can be attempted to be detected. Again, both the camera images and the map are used to determine the position of humans, required for Spot to complete its task.
Inputs	Chosen human information, camera images from Spot
Outputs	The next step to be executed, published on a ROS topic. If the human is detected and Spot has the object to be delivered in its gripper, prepare to move towards the human.
Parent function	Software Architecture
Child functions	

Table 3.41: Scan from current position

Label	Description
Name	Scan from current position
Description	From Spots current world position, all camera images have to be received and published on ROS topics.
Inputs	Signal to start scanning and retrieve all camera images
Outputs	Camera images to be used for processing, published on a ROS topic
Parent function	Search for human
Child functions	

Table 3.42: Continue building world model

Label	Description
Name	Continue building world model
Description	Camera images are obtained from the scan. Continue building the point cloud world model that can be used for depth perception and object/human detection.
Inputs	The camera images from the ROS topic
Outputs	World point cloud published on a ROS topic
Parent function	Search for human
Child functions	

Table 3.43: Walk around to expand world

Label	Description
Name	Walk around to expand world
Description	Using the world model areas that are insufficiently scanned can be determined. Spot moves to these areas to improve the accuracy of the world model.
Inputs	World model point cloud, signal to continue searching
Outputs	Audio confirmation that Spot continues exploring the space
Parent function	Search for human
Child functions	

Table 3.44: Human detected

Label	Description
Name	Human detected
Description	Processing both the camera images and world model using YOLO, humans can be detected. If the human to return the object to is found, publish to a topic the image location of the found object.
Inputs	The camera images from ROS topics
Outputs	Camera location of human, published to a ROS topic
Parent function	Search for human
Child functions	

Table 3.45: Alert operator if human found

Label	Description
Name	Alert operator if human found
Description	Audibly and visually tell the operator that an object/human has been found. In case that the human is not found several actions can be taken. With an incomplete world model, Spot can move around to scan more. With a complete model, Spot can notify the operator the task has failed, either quitting the task or attempt the task again.
Inputs	World model from ROS topic
Outputs	Visual and audio representation of task status (pass/fail)
Parent function	Search for human
Child functions	

Table 3.46: Deliver object

Label	Description
Name	Deliver object
Description	Using the camera location of the human, the actual world position of the human can be determined. Spot will have to move closer to this world position and then Spot uses its gripper to place the object near the human. Once the object has been placed, the gripper returns to its starting position.
Inputs	Camera image and location of human
Outputs	Object delivered to the defined human
Parent function	Software Architecture
Child functions	

Table 3.47: Identify human position in world

Label	Description
Name	Identify human position in world
Description	Together with the world model and the camera location of the human, the world position can be determined. To be done either with the depth cameras or other perception techniques.
Inputs	Camera image and location of object
Outputs	World position of object published on ROS topic
Parent function	Deliver object
Child functions	

Table 3.48: Move Spot closer to human position

Label	Description
Name	Move Spot closer to human position
Description	World position of the human is known. Spot has to move close enough to the position in order for the gripper to drop the object near the defined human.
Inputs	World position of human
Outputs	Ready signal for gripper to move towards human
Parent function	Deliver object
Child functions	

Table 3.49: Alert human of Spot's presence

Label	Description
Name	Alert human of Spot's presence
Description	Let the human know that Spot is now near the human ready to deliver the requested item. This will be done audibly.
Inputs	Ready signal done moving towards the human
Outputs	Let the human know audibly that Spot is ready to deliver the object
Parent function	Deliver object
Child functions	

Table 3.50: Lower the gripper

Label	Description
Name	Lower the gripper
Description	With the world position of the human and Spot also in range, the gripper is lowered to drop the object near the human.
Inputs	World position of object, ready signal from Spot being in range of human
Outputs	Ready signal for gripper to drop the object
Parent function	Deliver object
Child functions	

Table 3.51: Drop the object

Label	Description
Name	Drop the object
Description	With the gripper in place, the gripper only has to open to let go of the object.
Inputs	Ready signal for dropping
Outputs	The object now being in front of the human and the grippers being empty
Parent function	Deliver object
Child functions	

Table 3.52: Return gripper to default position

Label	Description
Name	Return gripper to default position
Description	Object has been dropper off and is not longer in Spot's grippers. The gripper must return to its original position in order for Spot to return to its original position.
Inputs	Signal indicating the object has been dropped and gripper can return to original position
Outputs	Object now in front of the human and Spot is ready to return to its origin to finalise the task
Parent function	Deliver object
Child functions	

Table 3.53: Finalise task

Label	Description
Name	Finalise task
Description	Upon completing the task, or failing the task and wanting another attempt. Spot will return to its origin, preparing to shutdown or reattempt the search.
Inputs	Signal indicating another attempt or task completed successfully
Outputs	Spot back at its original position
Parent function	Software Architecture
Child functions	

Table 3.54: Alert operator task completed

Label	Description
Name	Alert operator task completed
Description	Visually and audibly let the operator know that the task has been completed
Inputs	Signal indicating task completed successfully
Outputs	Visual and audible representation of task completed
Parent function	Finalise task
Child functions	

Table 3.55: Walk back to origin position

Label	Description
Name	Walk back to origin position
Description	For both a completed task or another attempt at the task, Spot will walk back to its origin.
Inputs	Signal indicating to walk back to its origin
Outputs	Spot back on its origin
Parent function	Finalise task
Child functions	

Table 3.56: Alert operator task failed

Label	Description
Name	Alert operator task failed
Description	Visually and audibly let the operator know that the task has failed. Give option for another attempt or quit the task.
Inputs	Signal indicating task failed
Outputs	Visual and audible representation of task failed, and options to continue or quit
Parent function	Finalise task
Child functions	

Table 3.57: Shutdown

Label	Description
Name	Shutdown
Description	Spot needs to shut down and stop any actions.
Inputs	Signal for shutting down
Outputs	Spot turned off
Parent function	Software Architecture
Child functions	

3.4 Activity diagram

Figure 3.2 depicts the activity diagram of our main operational scenario. It provides an overview of all the processes and how they flow into each other. We can see how each task has its own sub-tasks and what conditions need to be met in order for the next task to be set into motion.

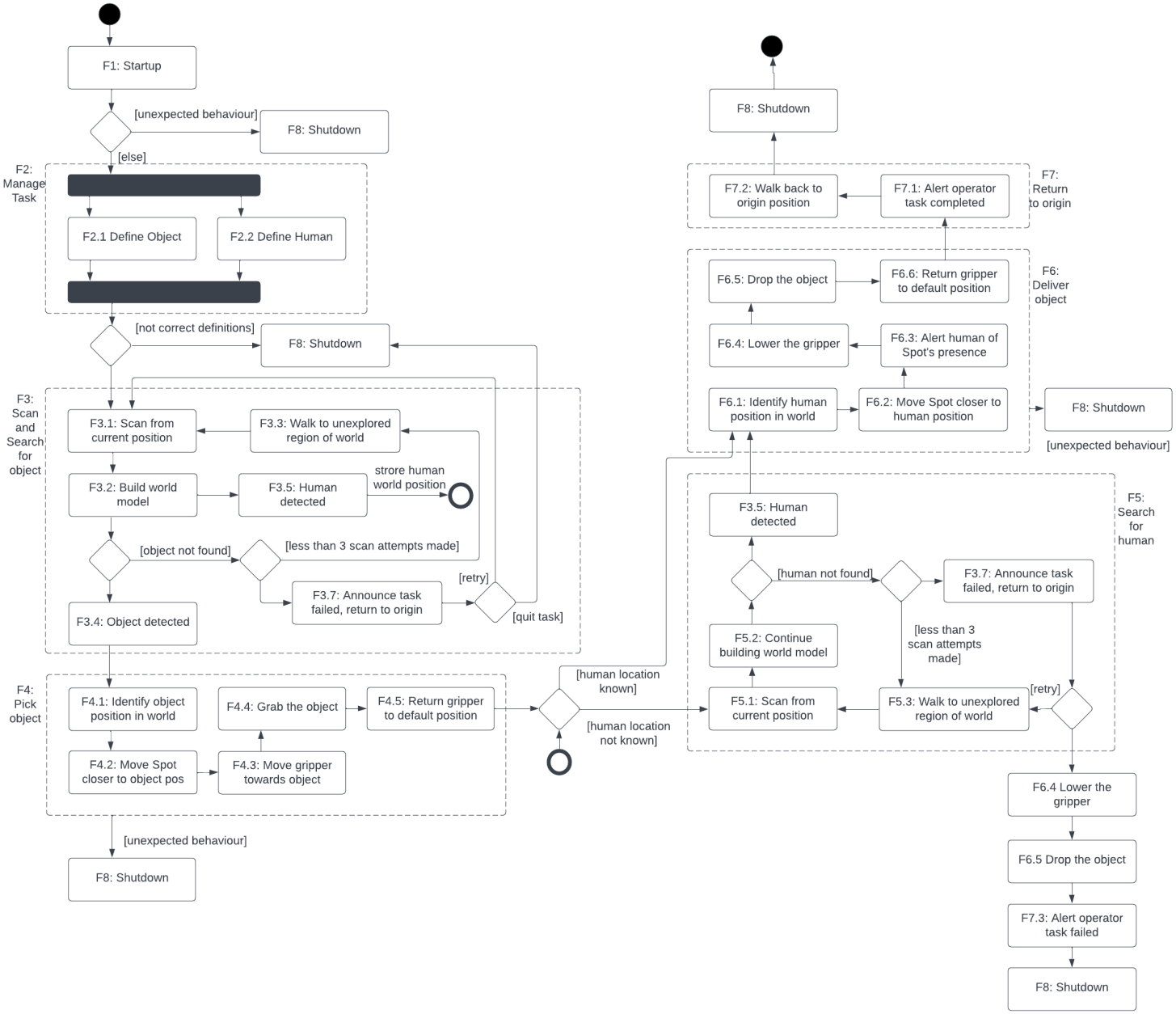


Figure 3.2: Activity diagram

FULL FUNCTIONAL DESCRIPTION OF THE ROBOT SOFTWARE

Previously the functional architecture was described for the operation of Spot to complete the task defined. Then, the robot software for the architecture actually had to be developed. This was done during the three week period of Sprint 3, where many hours were spent to develop the robot software. The tasks were divided and each member of the team got working on their respective part. From the first testing session with Spot, it became evident to the group that the simulation and the actual robot are in fact very different. The simulation having less cameras, no gripper arm, and missing some other features. Therefore, as a group the decision was made to focus more of the development of the software for the actual Spot robot, and less focused on tuning the software for the simulation.

4.1 ROS node graph

In the end, to show the achieved software architecture of the robot software for simulation developed by the group, the rqt_graph shown in Figure 4.1 is used. This shows the nodes and topics present during operation of the simulation environment. All nodes highlighted in light blue are created by the group. Due to the massive size of the node graph, there is no method of getting it more legibly, zooming in to 250% on the figure makes all nodes very legible. Some actions appear disconnected too, however, this is due to the nature of the finite state machine used to guide Spot through the operation. Further elaboration will be provided in subsection 4.2.3.

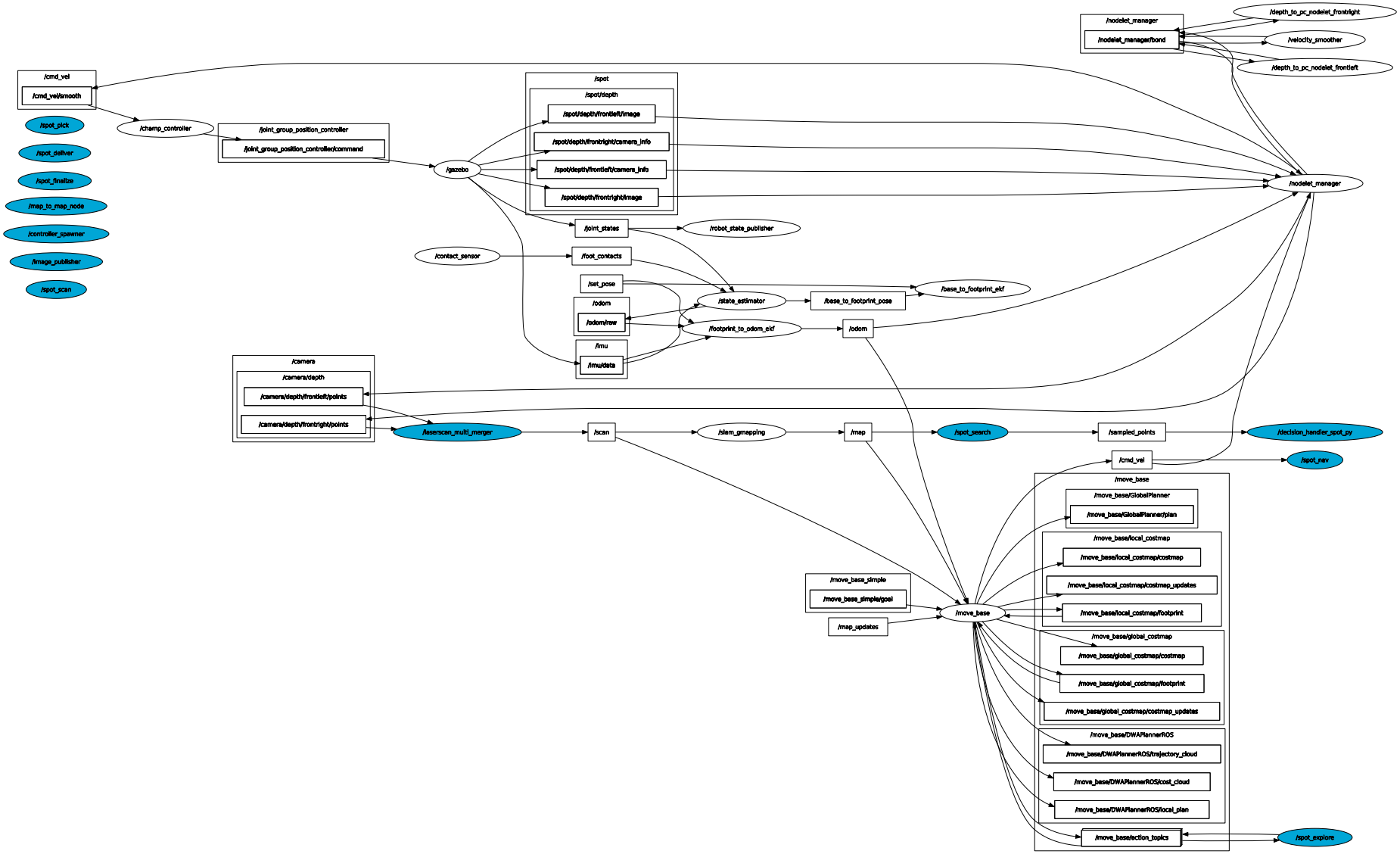


Figure 4.1: Rqt_graph of the simulation

4.2 Overview of the implemented robot behaviour

The entire robot software architecture for the simulation per module will be discussed in this chapter. The software architecture is divided into the perception, the planning and navigation, and the decision handler module. Figures and diagrams are used to help elucidate the operation of the module.

4.2.1 Perception

For perception, the primary task was to make a detection model that can process 2D images as fast as possible. The detection algorithm in question is the You Only Look Once version 5 (YOLOv5) model. This model is trained on the Common Object in Context (COCO) dataset. This dataset consists of eighty different object classes; including people, balls and apples. In order to enable YOLOv5 to detect the additional screwdriver, the perception specialists extended the model by gathering data and manually annotating the screwdrivers, which were used to train the model again using a guide [3].

The perception package consists of six functional self-made nodes, one modification of an existing Python script, five self-made simulator nodes, and four launch files; two for the simulation and two for the actual robot. All nodes are either publisher nodes or subscriber nodes, or both. For the simulation, two different launch files were created. The simulation recording launch file, record_sim.launch, launches the nodes that are necessary for YOLOv5 to perform predictions on incoming camera images from Spot's front left camera. Additionally, it saves unannotated images as well as annotated images. These images are shortly afterwards manually deleted for privacy reasons. The nodes used for this launch file are not directly necessary for the functionality of the robot regarding perception tasks, but are used for visualisation and debugging purposes. An examples can be seen in Figure 4.6b.

The functional launch file for the simulation, transform_to_3D_sim.launch, launches all the functional nodes for the perception task. The self-made functional nodes are the /detector_sim_node, /transform_and_filter_sim_node and map_to_map_node. The rqt_graph for this is illustrated in Figure 4.2.

The detector_sim node takes the RGB camera images from Spot's front camera and rotates this skewed image to make it suitable for our object detection models. Detection is performed on these images using Yolov5 for the humans, apples and balls. Our custom model is used for screwdrivers. The list of found detections are published in a list of boundingboxes.

Using boundingboxes from the detector_sim_node we can extract where the object is located in the world frame in the transform_and_filter_node. This is done by applying tranforming the depth camera pointcloud to the RGB camera frame and then using the uvs method to find the depth points corresponding to the center point of the boundingboxes (closest euclidean distance in the uv plane).

Lastly we transform this found point to the world mapping in the map_to_map_node by transforming our point to the world frame to locate the found object in the 2D occupancy gridmap.

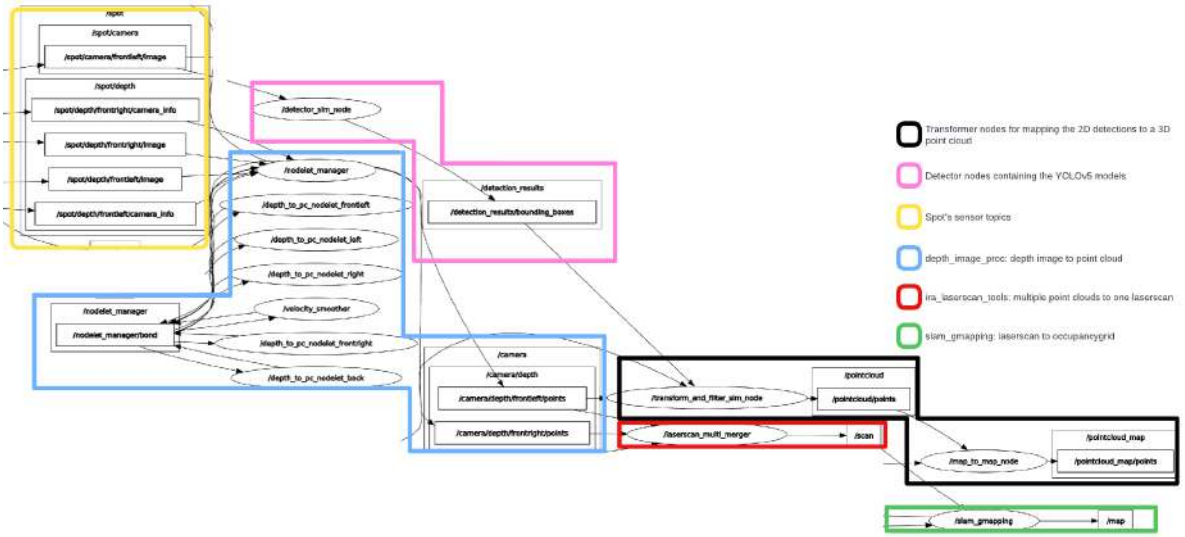


Figure 4.2: Rqt_graph of the perception pipeline

4.2.2 Planning and Navigation

In order for the robot to map the obstacles in its environment and to find free space for its navigation tasks, a 2D occupancy grid had to be created. This required the coupling of the sensory data to odometry data, so that the map could become larger as the robot would explore. There was only a slight problem that was highlighted by the client: the odometry is very imprecise and the drift from ground through becomes larger with time and distance travelled. Therefore, it was early on decided to find a solution for the SLAM (Simultaneous Localisation and Mapping) problem, which would solve the mapping task and the self-localisation within this map. The slam_gmapping [7] packages solves this SLAM problem using a Rao-Blackwellised particle filter for the learning of grid maps. All the particles represent a hypothesis of the robot's pose and map and the odometry is taken into account as an initial guess. These particles are then fused together, resulting in a map and pose for the robot.

The slam_gmapping package was, however, designed for long-range laser scanners, something that Spot neither possesses in reality or in simulation. Therefore, a pipeline needed to be created from the sensor's of Spot to laser scan data. Initially it was assumed that pointcloud data would be available for this task, since that was available in simulation. After the initial testing however, it became evident that this was not the case and that Spot only provides depth image data. This data therefore had to be converted to pointcloud data and was done using the depth_image_proc [6] package.

Afterwards the ira_laser_tools[8] package was significantly modified in order to output laser scans from multiple pointclouds. The ira_laser_tools package was originally designed to merge together multiple laser scans and output them as one. This package was modified to firstly merge all the pointclouds (2 in simulation, 5 on the actual robot) and then to create a laser scan of this merged pointcloud. This laser scan could then be used by the slam_gmapping package to create the gridmap. The visualisation of this pipeline can be seen in the rqt_graph in Figure 4.3.

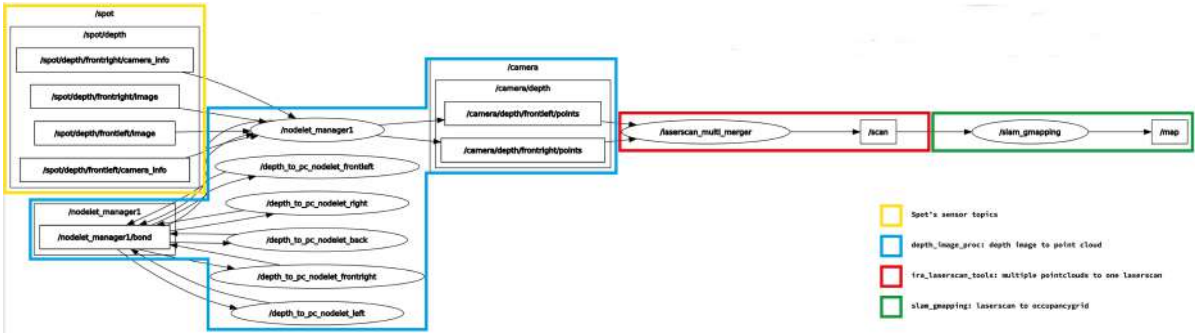


Figure 4.3: rqt_graph of the mapping pipeline

With an accurate mapping and clear absolute positioning of the robot in the map frame, the navigation part can be taken care of. To navigate around a cluttered environment, Spot needs to perform SLAM and navigate around the mapped area in a careful manner. To do this, both a global and local planner are being used to estimate an obstacle free path in the map frame. This is done by employing the move_base[10] package, which has a built in global and local planner. This package also conveniently supports action-based communication which adds robustness and ease of integration together with the finite state machine (FSM). In Figure 4.4, one can see how the move_base package is connected to the other topics.

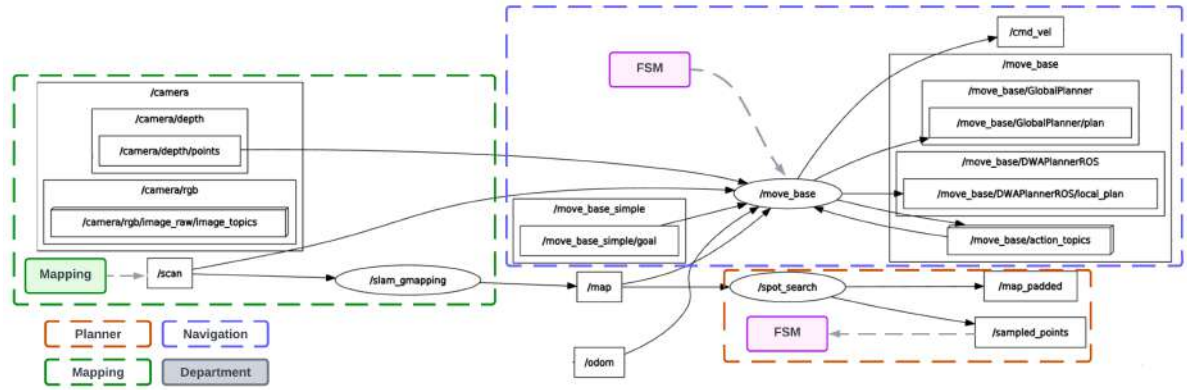


Figure 4.4: rqt_graph of the planning pipeline

The move_base package subscribes to the map topic which is constantly updating by the mapping pipeline, as the robot moves around and gathers more scans of the room. Moreover, the move base node also takes into account the scans from depth cameras, alongside the already mapped free space. This is useful for the case when humans or other dynamic obstacles appear in front of the robot, as it tells the local planner to recalculate the desired path. Even though, spot has integrated local obstacle avoidance, by using a local planner in the navigation pipeline, a more controlled movement is achieved. The global planner is using an RRT algorithm to calculate the desired path to a specific pose in the map frame. That is the (x, y, z) position of the robot and it's orientation $(pitch, roll, yaw)$. Figure 4.10 shows the planner in action.

The planner part in Figure 4.4 is represented by the orange outline containing the spot_search node. This node has the task of sampling exploration frontiers in the free space. In the free space, the frontiers are sampled from 1000 random points, which maximise the distance between them. These are then posted to the topic of sampled_points which is picked up by the FSM to command spot to explore. Each time the robot finishes exploring the frontiers new ones are sampled until the object and human are found.

The padded free space (map_padded topic) is computed by the spot_search node, by applying convolutions the map with the $[1, 0, 1]$ kernel for repeated times. The result is an erosion of the free space

marked by gmapping, thus, the robot has an extra safety margin away from the obstacles. This padded free space is then used by the frontier sampling algorithm.

4.2.3 Decision handler

In order to connect all the different functions a decision handler had to be created, this was done in the form of a Finite State Machine (FSM). The FSM together with many different action servers would control the operation of Spot, guiding it through the many different tasks. The decision handler can also be referred to as the Spot brain.

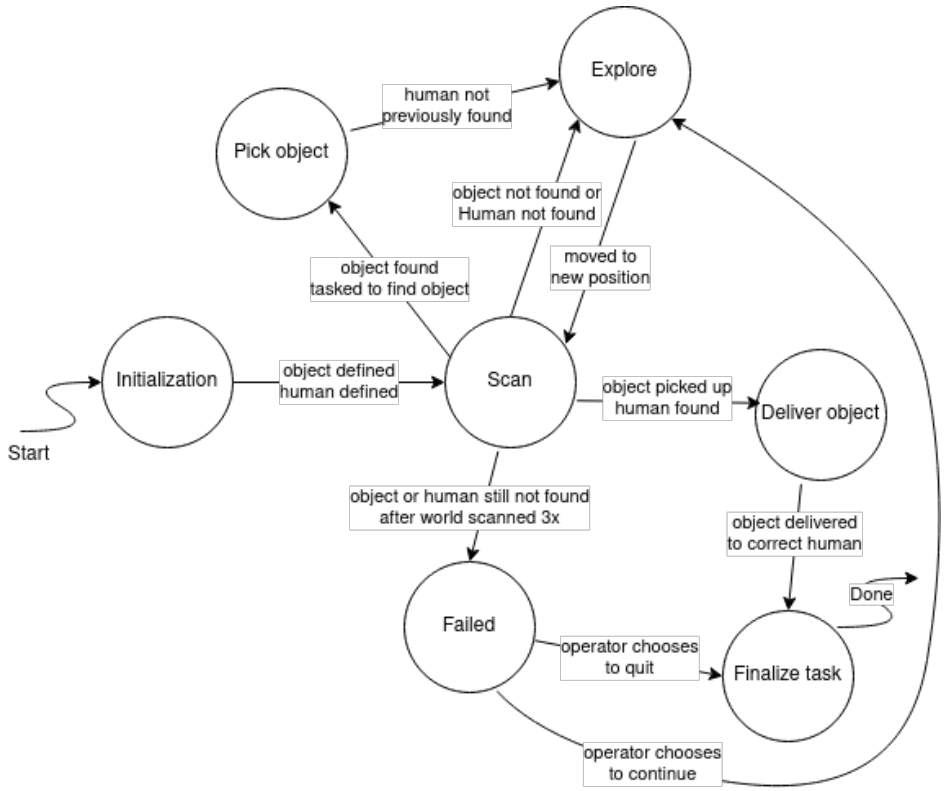


Figure 4.5: The FSM of Spot's Decision handler

Compared to a lot of the other subsystems the decision handler does not have a great presence in the rqt_graph, despite its importance. The decision handler makes use of the actions with custom messages to start different operations. Due to the sequential nature of the FSM it is not working on multiple actions at the same time. Thus showing some of the action servers as disconnected in the graph. Simply due to the action server being initialised but only being needed at the specific state of the FSM. The most noteworthy action nodes are the 'scan' and 'explore' actions, which are to be explained later in this section.

Starting Spot's decision handler with its respective launch file, starts all action servers and all other nodes used described by the previous sections. The first stage is initialisation, initializing variables, nodes and actions, but also the task. Task initialisation is done with the Graphical User Interface (GUI) as explained in subsection 4.3.3, which no longer required its own node. The GUI is controlled within the decision handler. As soon as a selection for object and human is made, the chosen selection is stored to be used throughout the rest of the FSM. The GUI changes what is shown depending on the current FSM state or due to a result obtain from one of the states. Upon defining the task Spot has to complete, Spot can start scanning to try and look for the object and human.

Upon entering the 'scan' state, the decision handler will make use of a ROS action to send the current task, whether to look for the object or human specifically, and the object and human IDs. This information

together with the perception nodes previously described in subsection 4.2.1, will then try to locate the objects or humans. Depending on the result of perception, if the object is correctly detected Spot will go into the 'pick' state to pick up the object. If the human is detected before the object is found, the information on the human location is stored to be used once the object has been detected, and Spot will go into the 'explore' state in order to find the object. If neither the object or human is found Spot will always go to the 'explore' state.

The 'explore' state is responsible for making sure Spot moves to try and complete the task, this can either be done by turning in place or moving to a different location in the world's free space using the exploration frontiers mentioned in subsection 4.2.2. Due to the detection currently only making use of one camera, in order to scan all around Spot, Spot at first does a 360 degree turn in place in 30 degree turn intervals. After each interval, return to the 'scan' state to try and find the object or human. If the object or human, depending on if Spot needs to find the object or human, has still not been found Spot will move to a new location in the world's free space. This process of doing a 360 degree turn then moving to a new place is done 3 times before Spot enters the 'failed' state, the operator is asked if Spot should continue trying to complete the task using the GUI, if yes Spot will try another new location, otherwise, the task is determine as failed. Spot will then have completed its task and finalise operation.

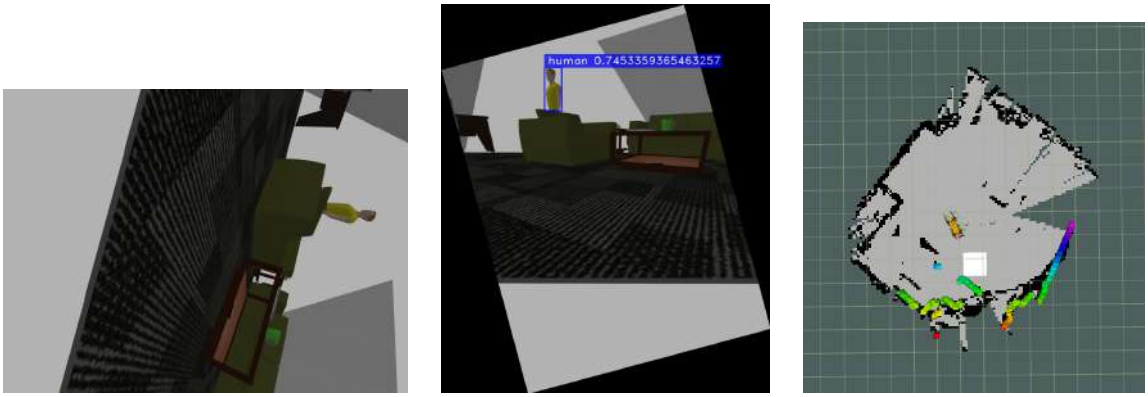
Finally, the remaining states: 'pick object', 'deliver object', 'failed' and 'finalise task', are states that are not used within the simulation. Due to the simulation version of Spot not having an arm. The action nodes responsible for picking and delivering the object to the human are not active in the rqt_graph. Furthermore, the failed and finalise task nodes currently only complete the FSM of Spot, with not much added functionality. The 'failed' state, as previously mentioned, asks the operator if Spot should continue searching for the missing object or human. Otherwise, Spot will enter the 'finalise task' state where Spot is eventually turned off. Theses nodes were deemed as less important when trying to develop a working Spot operation, thus together with the time constraints present, other tasks like perception, navigation and integration took priority during development.

4.3 Recorded behaviours

This section is responsible for showing the main behaviours of the robot in simulation, corresponding to the three specialisation within the group: Perception, Planning and Navigation, and HRI.

4.3.1 Perception

Figure 4.6 shows the pipeline of a raw output image of spot. The image first gets rotated and annotated and then mapped into the 3D world map (as a white box).

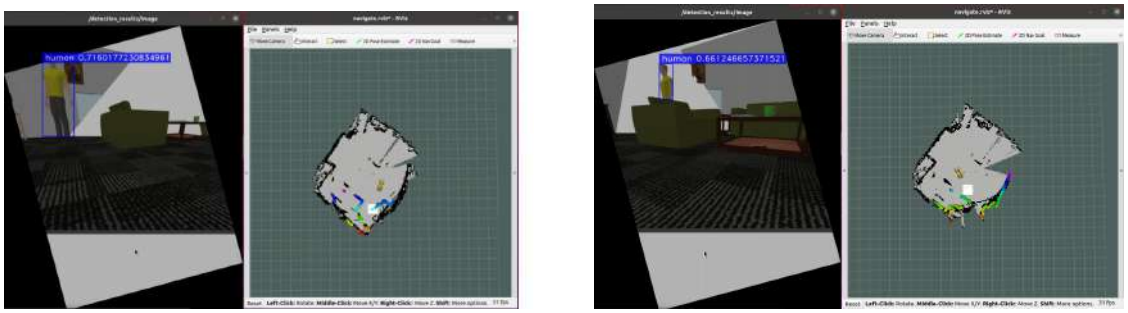


((a)) Raw RGB image output **((b))** Annotated camera image **((c))** Annotation mapped to world map

Figure 4.6: An example of the perception pipeline

Figure 4.7 shows two examples of human detection. We can see that the mapping is not always accurate. The left image depicts a good mapping, but the right image does not. The human is mapped

too close to Spot. This will need some debugging.



((a)) Good mapping

((b)) Worse mapping

Figure 4.7: Examples of good and worse mapping

4.3.2 Planning and Navigation

Figure 4.8 depicts all the different conversions of the sensory data. The white points on the right are from the original depth image. This depth image is than transformed to a pointcloud and can be seen on the left in black. Afterwards these merged pointclouds are scanned by a laser scan, which is visualised with the rainbow boxes.

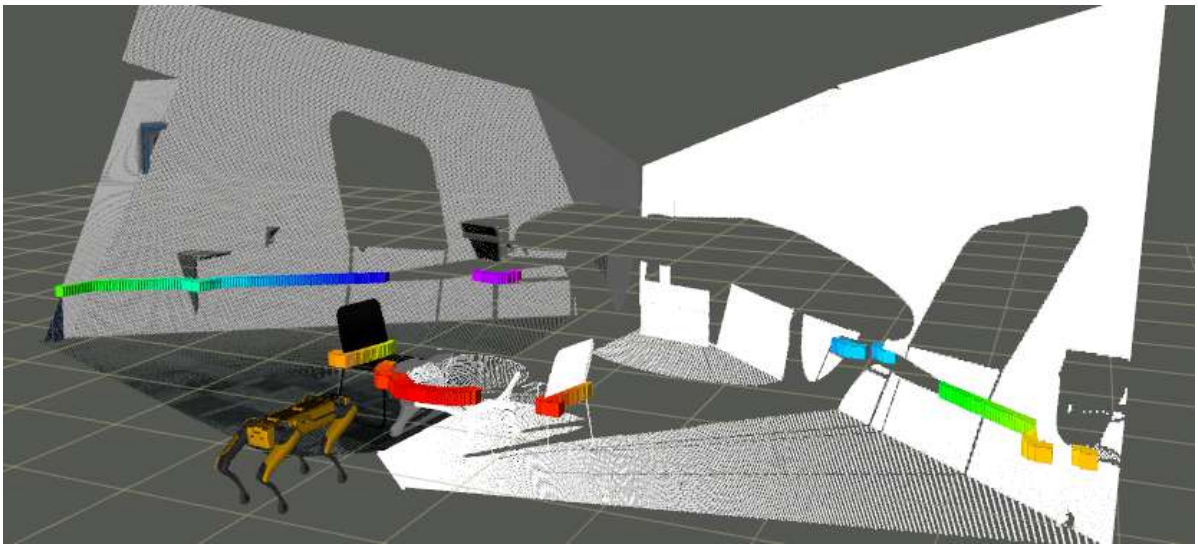


Figure 4.8: The transformation of the sensory data

Figure 4.9 shows the map that is generated by the slam_gmapping package. The laser scan bounces of the obstacles, which transferred to the map and depicted in black. The free space that is found is depicted in grey.

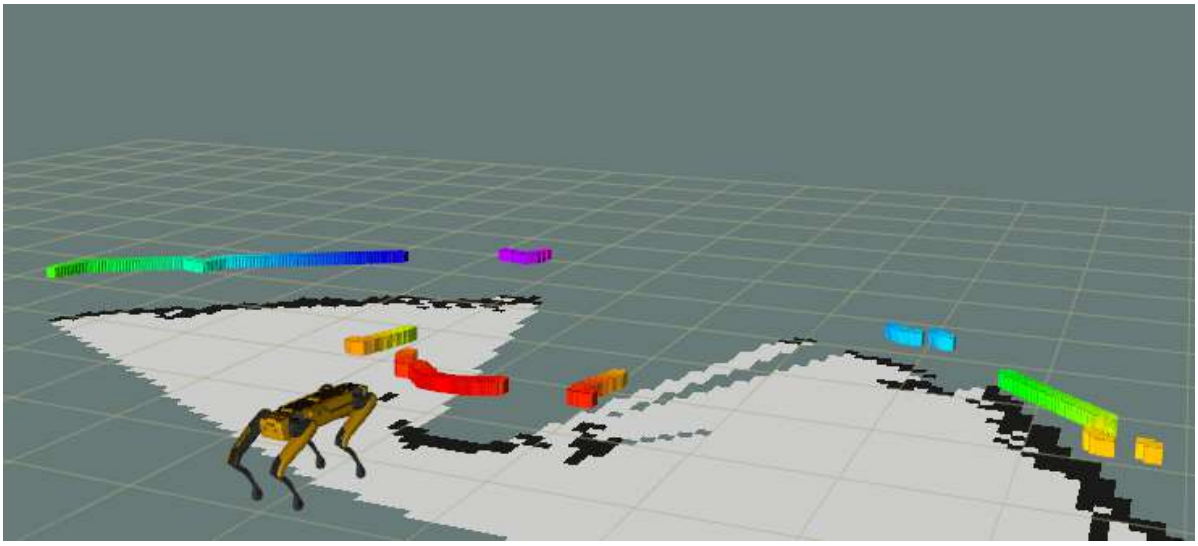


Figure 4.9: The transformation of the sensory data

In Figure 4.10 the map is represented by three different colors. The black dots represent occupied space by obstacles, the gray space represents free space calculated by the mapping package, and the white space is padded free space. Moreover the red dots represent the sampled frontiers in the padded free space.

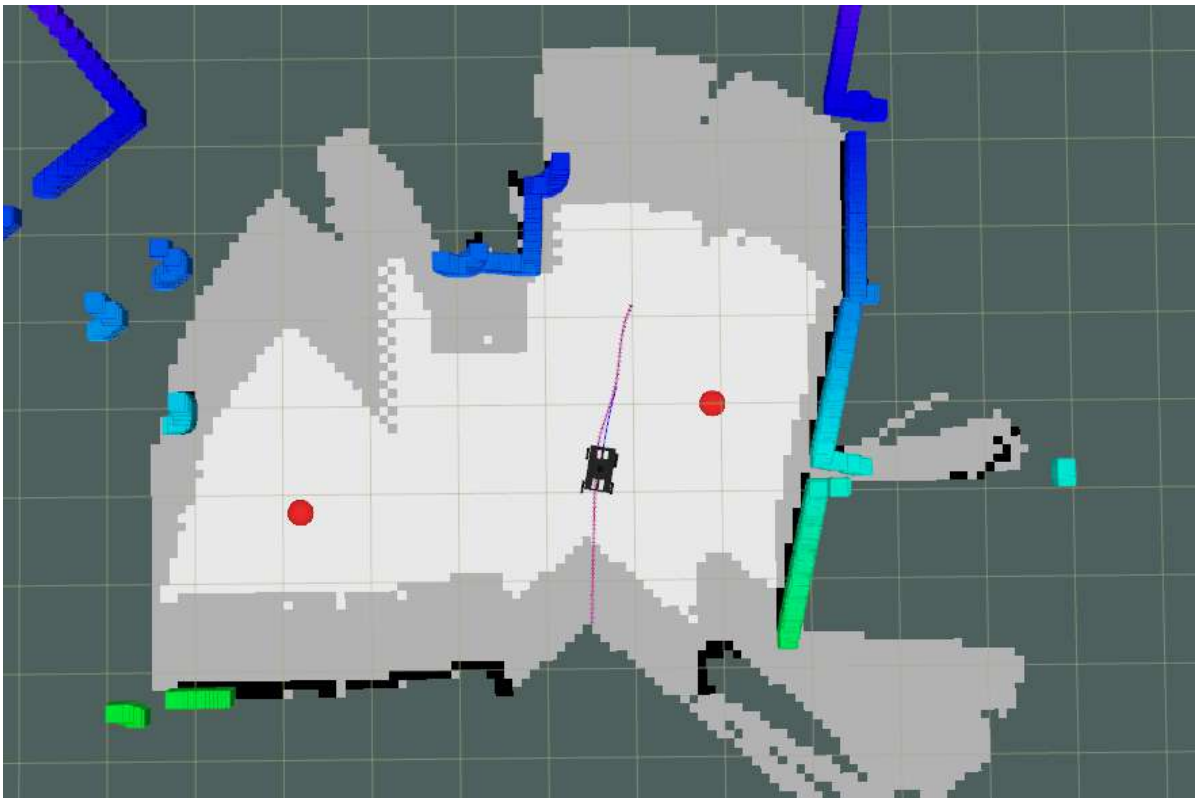


Figure 4.10: Robot moving across the map using RRT global and local planner to a new point while also padding free space (white space). Red points represent exploration frontiers sampled in free padded space.

4.3.3 Human Robot Interaction

For human robot interaction (HRI), the main task was to alert the operator of the current task status and allow the operator to define the task for Spot to complete. Initially, this was done using a ROS action client and server that ask the operator for terminal inputs, entering the corresponding ID of different tasks. This quickly became an unpleasant experience. The terminal shows many different values and information for all the different topics used during operation, which only becomes more when Spot was actually used too. Therefore, text to speech and a GUI were created for the HRI too.

Firstly, to allow the operator and everyone in Spot's surrounding know of Spot's presence and current task, text to speech was used to audibly inform of the different tasks. The different texts that are used can be seen in Table 4.1.

Table 4.1: Different speech texts used during Spot operation

Functional phase	Text used
F1 System startup and F2 Manage Task	”Hello my name is Spot, I am ready for my task”
F3.1 Scan from current position	”Looking for the object”
F3.3 Walk around to expand world	”Exploring free space”
F3.4 Object detected	”Object found”
F4 Pick object	”Picking up object”
F5.1 Scan from current position	”Looking for the human”
F3.5/F5.4 Human detected	”Human found”
F6 Deliver object	”Delivering the object”
F7.1 Alert operator task completed	”Finished with the task, getting ready to power off”
F7.3 Alert operator task failed	”Failed to complete task”

The text is played as soon as Spot enters the respective functional phase corresponding also to the FSM states described in subsection 4.2.3. However, this alone was insufficient for a good user experience for the operator as most of the interaction done with Spot was still done through the terminal, therefore, it was decided that creating a Graphical User Interface (GUI) was necessary too. The GUI, similarly to the text to speech updates at the different FSM states. The GUI allows the operator to interact with buttons and also better represents the current phase and the defined task. The following images in Figure 4.11 show different screens of the GUI. Note that the image shown in Figure 4.11(e) and Figure 4.11(h) are placeholder images from the simulation, these update with the most recent camera image. These images are taken from perception when either the object/human is detected and allows the operator to confirm the detection. If the detection is correct, the operation will continue as planned, otherwise Spot will continue to try and locate the correct human. Furthermore, in the upper right corner of all images the defined object and human are shown to remind the operator of the current task. The images show a few different combinations of objects and human shirts.

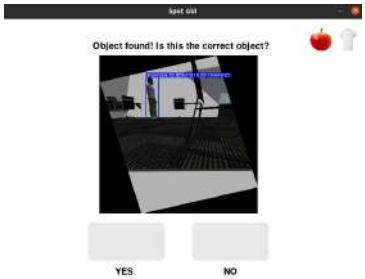
The clear difference that the addition of the GUI made for the user experience when interacting with Spot can be seen in Figure 4.12. Figure Figure 4.12(a) shows the terminal based initialisation that was initially developed. At this point this was the only feature developed inside the decision handler, and thus didn't do anything else. Figure 4.12(b) shows the same terminal based initialisation but now with the addition of the perception, planning and navigation nodes. The text is quickly lost due to all the other information also published on the terminal, making it very hard to read and simply an unpleasant experience. Finally, Figure 4.12(c) shows the GUI used at a similar stage of initialisation with all the other nodes implemented. The GUI in action can also be seen in this video, taken during one of the test session with Spot, both the GUI and text to speech feature can be seen or heard. The GUI was made to allow for a better user experience when interacting with Spot to complete the task, which the GUI succeeded in, and in the future could also be something that users could use from their phone if development was continued.

((a)) F1 System startup

((b)) F2.1 Define object

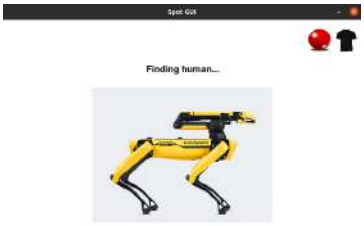
((c)) F2.2 Define human

((d)) F3.1 Scan from current position

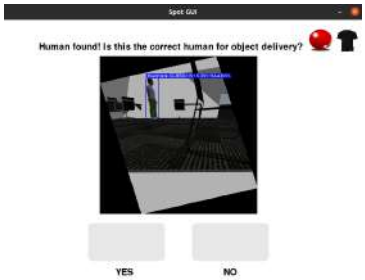


((e)) F3.4 Object detected

((f)) F4 Pick object



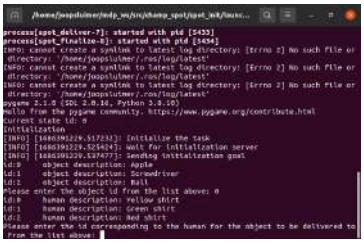
((g)) F5.1 Scan from current position



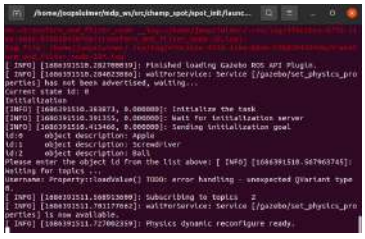
((h)) F3.5/F5.4 Human detected

((i)) F6 Deliver object

Figure 4.11: Nine different GUI screens shown during operation, note some figures may look similar but show different texts for the different states



((a)) Terminal initialisation without other nodes implemented



((b)) Terminal initialisation with other nodes implemented, the screen fills up with more text soon after this moment

((c)) GUI initialisation

Figure 4.12: The difference visualised that adding the GUI made for the user experience

SUMMARY OF REAL ROBOT RESULTS

As previously mentioned in chapter 4 the group decided to focus development on the actual Spot robot rather than the simulation, due to the large difference between the simulation and real robot and also to ensure a good final product can be delivered. This chapter focuses on the differences between the software of the actual robot and the simulation.

5.1 Functional graph

Similarly to before, to show the achieved software architecture of the real robot is shown in Figure 5.1. All nodes highlighted in light blue are created by the group, where a majority were also present in the simulation version of the rqt_graph. Once again the zooming in to 250% on the figure makes all node very legible.

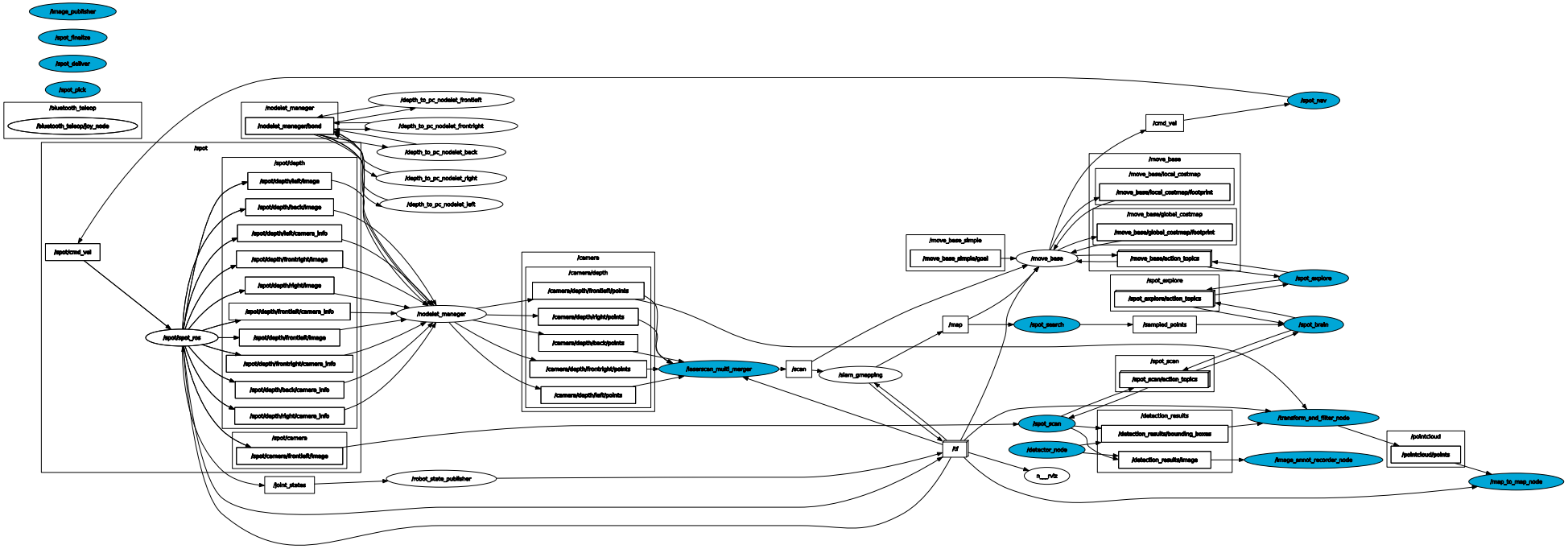


Figure 5.1: Final obtained node graph of the software on the real Spot robot

5.2 Recorded behaviours

Given that the entire robot software architecture was previously already described in section 4.2, this section will focus more on the differences between the recorded behaviour between the simulation environment and real robot.

5.2.1 Perception

For the real robot's environment, the use of all three objects was possible; apples, balls and screwdrivers. In the simulation, the YOLOv5 algorithm was only tested on humans. The 2D object detections performed satisfactory in some situations, but unsatisfactory in others. In general, the detection performance on balls and humans was excellent; the best confidence scores were 92.1% and 89.6% respectively. Balls consistently received high confidence scores throughout a test run for nearby balls. As the camera-to-object distance increased, the confidence score decreased. Humans received slightly lower confidence scores compared to nearby balls. However, YOLOv5's performance was a lot more consistent as the camera-to-object distance increased and partially occluded humans were still recognised. These detections are illustrated in Figure 5.2.

((a)) Detection on tennis ball with 92.1% confidence.

((b)) Detections on two humans with 76.9% and 89.6% confidence.

Figure 5.2: The best detections on balls and humans, respectively.

Unfortunately, YOLOv5 performed unsatisfactory regarding the detection of apples and screwdrivers. Apples were occasionally misclassified once the camera-to-object distance exceeded a certain threshold. This is most likely due to the similar shape between apples and balls, in addition to the fact that the apple used for testing had a homogeneous colour. Moreover, when apples were detected, the YOLOv5's confidence score never exceeded 70.0%. YOLOv5's best successful apple detection and example of a misclassified apple are illustrated in Figure 5.3. Screwdriver detections had an even lower performance than apple detection. YOLOv5 was extended manually on a dataset containing circa 1100 screwdrivers. This underperformance is most likely a consequence of an insufficient amount of representative data. The dataset of 1100 screwdrivers contained many different scenarios; e.g. screwdrivers in a toolbox, screwdrivers hanging on a wall and screwdrivers being held in someone's hand. The highest confidence score of a successful screwdriver detection is 80.5%. YOLOv5 produced numerous false positives and false negatives, which made it hard to tune the detection threshold. A low threshold resulted in false positives composed of beams of light or thin metal beams, whereas a high threshold would result in more false negatives where actual screwdrivers would not get recognised. YOLOv5's most successful

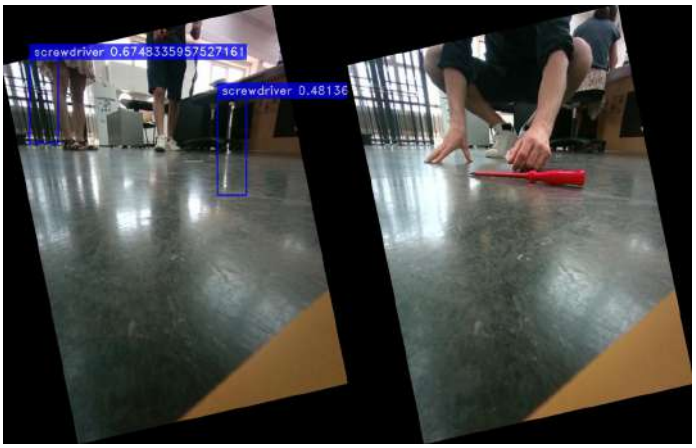
detection and examples of false positives as well as false negatives are illustrated in Figure 5.4.

((a)) Detection on an apple with 69.6% confidence.

((b)) Misclassified an apple as a ball.

Figure 5.3: A successful and unsuccessful detection of an apple, respectively.

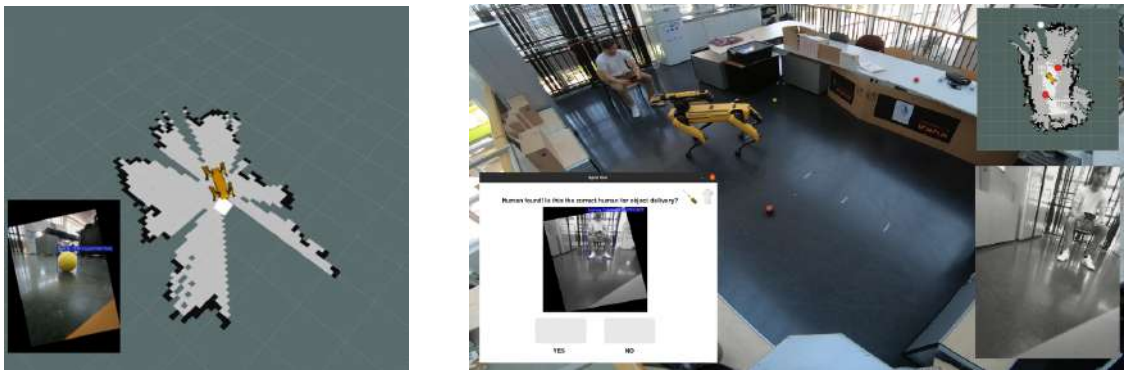
((a)) Detection on a screwdriver with 80.5% confidence.



((b)) Examples of unsuccessful screwdriver detections. Left illustrates false positives and right illustrates a false negative.

Figure 5.4: A successful detection of a screwdriver and two unsuccessful detections, respectively.

After attaining 2D detections of objects using the real robot, these had to be converted into a pointcloud for the occupancy gridmap. The same correlation between accuracy and object-to-camera distance could be seen with mappings of 2D detections; the greater the distance, the lower the accuracy. This created a margin of error. However, this margin of error becomes less salient at greater distances. The idea of having the location of the object or human in the occupancy map, was to implement this into Spot's navigation strategy. Spot would have walked towards this location in increments. While walking towards it, Spot would get closer and would update this location in its mapping, thus increasing the accuracy of the gridmap location. Examples of a mapping of a nearby 2D detection and a mapping of a relatively distant 2D detection are shown in Figure 5.5.



((a)) 2D mapping of a nearby 2D detection of a ball.

((b)) 2D mapping of a relatively distant 2D detection of a human.

Figure 5.5: Examples of 2D mappings of nearby and distant 2D detections, respectively.

Finally, when constantly updating the gridmap with 2D detections, we found that the 3D location rotated with Spot, with the same angular velocity. This is a consequence of the detection being relative to Spot's pose. For future mappings, this will not prove to be problematic as Spot will only perform detections and mappings when in a stationary state. In general, there was no great difference between the behaviour of the software on the real robot and the simulation. The only noticeable distinction is that the 2D detections of humans exhibited slightly higher accuracies.

5.2.2 Planning and Navigation

In Figure 5.6, the results of the implementation of the mapping package on the real robot are visualised. It can be seen that the depth image of all five cameras are used for the conversion to pointclouds and laser scans, as opposed to the two front facing camera's in the simulation. This resulted in a much faster way of mapping the environment, where often a useful map was produced instantly after start up. In simulation, a similar result would only be acquired after doing a full 360 degree spin.

Figure 5.6: Mapping in the testing environment

Although the performance of the mapping was good, there were some differences between the simulation environment and the real world testing environment that needed to be taken into account. The biggest being that there were no physical walls in the real world testing environment. This resulted in depth measurements from outside the testing environment, that were caused by gaps in the enclosure of the testing grounds. These gaps were for example between the storage containers and the rods of the railing, as shown in the bottom part of Figure 5.6. The steps taken to mitigate the influence of these differences on the actual occupancy grid can be read in the debugging report in section 5.3.

After implementing these changes described in the debugging report, the autonomous navigation within the desired free space could be performed. There were no big differences in the performance of the exploration compared to simulation and satisfactory results were obtained for this implementation. The robot would autonomously navigate to the frontier points and perform the 360 spin in order to find the objects and humans. During this phase the map would get updated and new frontier points would be found where the robot could pursue the exploration task.

5.2.3 Human Robot Interaction

All the previously implemented features for a better user experience worked independently of a working real robot, thus the GUI and text to speech feature provided the same experience as in the simulation environment. However, now that real humans are present during the operation some important things have to be considered. Due to an individuals safety and privacy reasons, all images used during perception must not be saved, or deleted once operation has completed. This has yet to be implemented, but is of a high priority if development were continued. Similarly, as per Figure 4.11(h), the GUI shows the user an image of the detected human for confirmation. Again for the safety and privacy of humans in Spot's presence during operation, the face of the human should be blurred such that all humans maintain their right to privacy.

5.2.4 Integration

Even though everything worked well individually both in simulation and on the real robot, upon trying to integrate all the sub-modules together the performance drastically decreased. Not only were there issues with package dependencies by working on multiple different laptops and different packages having different dependencies that were unable to be resolved. The performance of the perception, planning and navigation modules decreased. During the execution of the task, Spot was unable to make any detections at all. Even after proving more time for the scan to be complete in the decision handler, no successful detections were outputted. Furthermore, often times Spot would fail to move, which occasionally would be caused by the move_base action failing and thus not sending a position that Spot should move to. Finally, when using everything together it was clear that the map created represented the actual space much worse than compared to when ran alone. Often times the map would not update properly or replicate the correct size of the room it was tested in. This all was most likely due to limited computational efficiency, thus leading to all modules performing worse. An attempt was made at making ROS run on multiple computers making use of ROS master/slave APIs [9], having a ROS master launch the decision handler and two other computers run perception and mapping respectively. However, this also had little success as Spot would fail move even more frequently, most likely due to the move_base action failing more often on a slave computer. In the end if more time was available the integration issues could have probably been solved.

5.3 Debugging report

Throughout the short period of working together with the robot, many differences between the simulation environment and real robot were noticed. Due to this difference most of the group's focus was put on developing a working real robot rather than in the simulation.

Perception

For detection the system is subscribed to one of the front cameras. In the simulation this was an RGB image, but during the first test it was observed that this topic is a greyscale image. This is not optimal because Yolov5 is trained on RGB images. Spot does have RGB cameras, but the images are converted before they get published to the image topic. To solve this we used an example python file from Boston Dynamics which can be used to save RGB images. We modified it such that we had a subscriber node which publishes RGB images.
When testing the full implementation on the real robot a weird error occurred. The error suggested something was wrong with the pointcloud message type, but only occurred when a simple numpy opperation was aplied to the object. The error was caused by a difference in the names of transformation matrices in the simulation and in the real world. It was a very small but frustrating bug because the traceback pointed to a different piece of code than where the actual bug was.

Planning and Navigation

During testing, the biggest difference in performance was observed with the quality of constructing a map of the environment. Parameters of the package gmapping, which handles the transition from robots scans to an occupancy grid map, required fine-tuning. As explained in subsection 5.2.2, space behind obstacles which defined the testing environment was being categorised as free space. A way to circumvent this issue, was to reduce the maximum scanning depth and height of the point clouds.

Another occurring issue had to do with commanding the robot to move to a certain position. As explained in subsection 4.2.2, the move_base package takes care of global and local planning of the robot in the map frame. This package sends messages to the ”cmd_vel” topic, which as tested in the simulation, made the robot able to receive the messages and move accordingly. However, during testing Spot was not able to move to its commanded position and so an extra node had to be developed to forward the messages to Spot’s specific topic, which is ”/spot/cmd_vel”. Ultimately this node has been given the name ”spot_cmd_vel”, and after developing it, spot was able to function correctly.

INDIVIDUAL REFLECTION

This chapter focuses on the importance of reflection as a critical skill for engineers. All the team members share the personal goals that they hope to achieve over the course of the project using the SMART method. During the project, everyone reflects on their goals, work and role in the team through peer evaluations and sprint retrospectives in order to enhance their reflection skills.

6.1 Learning Goals

Joop
For this project I want to gain more experience working on a project with a client in mind, but also to work with a physical robot. I would like to actively take part in the two group meetings per week, and make sure that everyone else in the group also feels included and can take active part in discussions. From this project I expect to learn lots about working with a client, which I can use in the future working in the robotics industry, and also learn from my fellow project members.

After having worked with the actual Spot robot I do feel more comfortable and also enjoyed working with the actual robot. We as a group definitely managed to work together at least twice a week. Everyone was communicating well and having an active role in development. Due to the very short time given for the project I don't believe that I have learned much about working with a client, two presentations to show results is not a lot. The fact that there was so little time in general made the project very stressful for me, working a lot of long days to get Spot working within the time frame.

Jelle
During this project I want to bring my computer vision skills into practice. Having learned a lot about the subject during the last year, this is the perfect opportunity to bring my newly learned skills into practice. Moreover, I would like to learn how to work together with a client, since this is something I haven't done before. Lastly I'm excited to work with a group where everybody has their own responsibility, because this is the way in which the industry works.

Lars
During this project, I am most keen to learn how to program an actual robot and implement my coding into it. I am very confident in my coding skills, but I believe that they can be improved even more. Additionally, I am very curious to experience how it is to work in a team with an actual company as my team's client. I believe that this project will help me achieve both goals, as it seems to be very representative for the type of work for a robotics expert.

Denniz
Over the course of the project, my primary goal is to gain expertise in collaboratively working with individuals in various roles, as one would in a real-world software project. I am keen on becoming better at working together on coding projects, where a lot depends on bringing together the work of different team members. Additionally, I am motivated to become more proficient in working with physical robots and gaining hands-on experience.

Now that we're in the last days of the project, I am happy to say that I achieved more from my initial goals than I envisioned when I set them. I learned a tremendous amount in terms of working together as a team and collaboratively creating a software architecture. Working with individual git branches, following timelines and giving each other feedback thought me a lot in becoming a better team member. I also enjoyed working with a physical robot and was pleasantly surprised at how well we as a team we're able to implement our software on a real robot, even though it was the first time for all of us. It made see how much we learned in only one year.

Andrei
Throughout the duration of the project, my specific goal is to gain expertise in the programming language and tools that we're using for the project, as well as to develop a strong understanding of working the Agile way. I am most enthusiastic about applying my current knowledge of cognitive robotics in the field of planning and navigation for solving a challenging problem. Moreover, I expect to learn a lot from my different colleagues and their domain of work by exchanging ideas, collaborating and integrating our work towards an end-product.

6.2 Peer Feedback Received and Peer Feedback Given

Joop
From the first buddycheck I received a lot of positive feedback, which was good to know. I was already aware of the one time I did not properly communicate my availability to someone I was working together with, and made sure that it did not happen again afterwards. For the second buddycheck, I received the feedback that I could've provided an opinion more often on other modules, which I agree with. I tried working with the other modules too but possibly could've done more in terms of suggesting ideas.

My peers agreed with the feedback and did not get defensive. I believe that everyone was already doing well, so giving feedback to improve was quite challenging. I have learned from the peer feedback workshop that giving more positive feedback works well when giving feedback on how to improve.

Jelle
In the first buddycheck it was made clear that I should be more outspoken about my opinions. This is something I know i do too little in the early stages of a project as I tend to analyse problems a lot before I try to form a strong opinion. This might be due to a slight fear of being wrong, but it is definitally not a good trait. After the first buddycheck I tried to open up more about my opinions.

In the second buddycheck this tip did not come forward again which I think is a good sign. During the second buddycheck the most important critique was that I share my progress too little sometimes. I understand where this comes since I mostly discussed things with my perception colleque only.

Overall I think the team was open to critique the entire project, because we saw it as opportunities to improve. Critique was always valid and the person receiving the feed understood.

Lars
The feedback from both buddychecks helped me improve myself. My team members view me as a hard worker and as a leader, at some times. I agree with both of them; I enjoy organising the team in order to work towards the goals we have envisioned. Furthermore, what stood out to my team members was that I volunteer quite often and that my communicative skills are excellent. This I also agree with; I never mind doing extra work and I try my best to always keep my team up to date with my progress and availability.

Thankfully, my peers also provided feedback on which I could improve myself. This involves asking for help when needed, keeping a better overview of what needs to be done and thinking of back-up solutions in case primary solutions do not work out as planned. I have to agree with all of these points for improvement. Occasionally, I get lost in all the other work I am doing, resulting in a tendency to become unstructured and losing sight of what exactly needs to be done. In order to prevent this from happening, I organised my agenda, always kept my to-do list up to date and started asking for help when my agenda got too full. Additionally, I also started to prepare back-up solutions in case my primary solutions did not work the way I had hoped.

My team members reacted grateful for the feedback they received. Occasionally, individuals asked for some explanation on some feedback they received, but were always in agreement with what they received. I think I can improve my manner of giving feedback by adding some more explanation or by giving an example in order to clarify what exactly is meant. Furthermore, I noticed that positive feedback encourages continuation of that behaviour.

Denniz
I found the buddycheck peer evaluation system to be a very valuable tool throughout this project, as it allowed me to work towards improving my skills as a team player. I am grateful for the well-written and kind feedback provided by my colleagues. Their input not only motivated me for the project as a whole but also compelled me to actively address the areas they highlighted.

During the first round of feedback, I learned that while my interaction with and attitude towards my team members were generally positive, I lacked in demonstrating leadership and taking initiative within the team. I considered this feedback seriously and viewed it as an opportunity for personal growth. In the subsequent weeks, I made a conscious effort to show more initiative and take on a leadership role where possible.

I believe the changes I implemented in my behavior were well received by my team members, as reflected in the significantly improved feedback during the second round. The score for my leadership and initiative increased by more than a full point, and I am very grateful of the written feedback provided by my teammates.

I took pleasure in enhancing my ability to provide effective feedback. I found that my colleagues were consistently receptive to receiving feedback and that they recognised it's value in achieving better project outcomes and personal development. Overall, I consider buddycheck to be a highly useful tool in helping me achieve the goals I set at the start of this project.

Andrei
The buddycheck peer evaluation allowed me to get a better an unbiased review into how I am perceived as a collaborator in an engineering team. My peers feedback for me allowed to think deeply about my actions and daily interactions, and to improve on them where needed.

Regarding the positive feedback I received, I am glad to know that my teammates recognise my ability to maintain a good overview of tasks and guide the team accordingly. It is encouraging to hear that they value my perspectives and find them interesting. Additionally, the feedback highlighting my leadership skills and good planning abilities affirms that my efforts to improve in those areas have been successful. I appreciate the recognition for being well-informed, open to collecting ideas, and willing to seek help when needed.

As for the negative feedback, I take it seriously and reflect on how it aligns with my self-perception. The feedback about having a tunnel vision and potentially not fully listening to others is valuable. It prompts me to be more conscious of my listening skills and ensure that I give due attention and consideration to the viewpoints of my teammates. The feedback about being more proactive suggests that there is room for improvement in terms of taking initiative and going beyond the assigned responsibilities. Similarly, the feedback about communication highlights an area where I can focus on enhancing my skills to ensure effective and clear communication within the team.

After giving feedback to my peers, I found that they were generally receptive and appreciative of the feedback I gave them. They recognised the value of receiving constructive criticism for their personal development and the improvement of project outcomes. I did not encounter defensiveness or disagreement from my peers.

However, I believe there is always room for improvement in how I deliver feedback to my peers. I can strive to ensure that my feedback is not only constructive but also delivered in a supportive and empathetic manner. By focusing on providing specific examples and offering suggestions for improvement, I can further enhance the effectiveness of my feedback and foster a positive and growth-oriented environment within the team.

6.3 Transferable Skills

Joop
Throughout the project I tried to make sure to work well together with the rest of the team, and everyone was working well together too. We communicated and planned well within our groups of all the tasks that had to be done, which very much helped with teamwork. Most of my coding development was done alone, but in the end working together with all the different disciplines to make sure it all connected together went smoothly too. We changed leadership roles too often during the project, which was a good way of working on leadership. I made sure to prepare before meetings for all the tasks we had ahead of us, and made sure everyone took part too. Everyone in the group was responsible with their assigned tasks and worked hard, meaning much leadership in my opinion wasn't necessary. In terms of entrepreneurial thinking, it was developed subconsciously. Most of the time when something was not working, we quickly changed our plans to make the best out of the situation. I like working in a team and don't mind a leadership role either, both I can consider part of my strengths. For weakness, it was mainly motivation, I made sure to still take initiative but upon received very negative feedback it was hard to find the motivation to continue all the reporting and other required tasks. Luckily, programming was a lot more fun and helped with the motivation.

Jelle
Transferable skills are a big part in a project like this. Good team work is important, but can be hard with a high workload. When the workload is too high often problems get split up along team member which can be time efficient, but also is at expense of critical thinking of the group as a whole. The teamwork with my perception colleague was really good, but as stated in the peer feedback, updates to the rest of the team were missing sometimes. This improved towards the end of the project when everything came together. Regarding leadership I know I am not a natural leader, which was confirmed during this project. It does not feel natural for me to take the lead, but that was not really necessary. The group worked really well without always having a formal leader. What was new during this project was the separation of tasks. Everyone having their own speciality was a fun new experience and it made me notice that I enjoy thinking about the broad picture a lot too. Especially the multidisciplinary solutions where you think about how to connect everything. I learned a lot in this aspect, because your part does not only have to work, but it also needs to work together with the other parts. Entrepreneurial thinking came into play when talking about risk assessment and mitigation and during planning. Because of a tight schedule, not everything we wanted to do was possible and thus we had to focus on the most crucial aspects. I enjoyed thinking about the constraints of the project and analysing what was most important.

Lars
Regarding my own development throughout the project, I believe that I have made progress in teamwork, leadership, and strategic multidisciplinary problem-solving. I feel that I still lack skills with respect to entrepreneurial thinking. The feedback I received regarding teamwork and leadership, taught me to be more structured, even during very busy weeks, so that the process of reaching the ultimate goal goes more smoothly. Moreover, the feedback taught me to ask for help when struggling with a problem; being able to ask for help is one of the main advantages of working in a team. With regard to strategic multidisciplinary problem-solving, I feel that I have made some improvement. When working on perception, I noticed that with every problem I was able to solve, my expertise in that field grew. Furthermore, I simultaneously learnt more about the fields of path planning and navigation and of decision making. At the final stage of this project, we had to combine all fields, thus we all had to understand each others' parts to a certain extent. When integrating all fields together, I noticed that I understood almost every error we encountered and that my debugging skills were not only useful for perception related bugs, but also helped debugging in the other fields as well. However, besides all these improvements, I did notice that I was not very useful in entrepreneurial thinking. I had my mind set on developing the software for Spot, rather than thinking of background and relevance of this software. Throughout the project, my interests proved to lie with the problem-solving and managing sections. I enjoy working in a team and I like to organise the manner in which we work together. Entrepreneurial thinking, however, proved to be one of my weaknesses and I do not believe that this type of thinking peaked my interest.

Denniz
I believe I learned a lot in terms of personal development over the course of the project. Like I mentioned before in section 6.2, I made great progress in teamwork and leadership. When I analyze my development in entrepreneurial thinking, I believe the format of this project was excellent to grow in this field. We had to come up with an innovative real life application for product based of a few requirements of our client. This thought us for the first time during a project that the product not only has to work, but also has to fulfil some demand gap and has to provide a pleasant user experience.

Furthermore, working with ROS both in simulation and real life, thought me to be very adaptive and to seek for opportunities that are already out there. In making the occupancy grid, where a whole pipeline of sensor transformations had to be performed, I learned that there is great benefit in looking at what already has been implemented, in stead of always creating something from scratch. It was very nice to see the vast amount of packages that available within the ROS ecosystem and to use packages that were created by others. These packages were already highly optimised sometimes, where it would suffice for me to make a few adaptations or to write a piece of software as a step in between the stages of the pipeline.

I also enjoyed working together with different disciplines within a team. It was nice that everyone was responsible for the specific parts of the software and to discuss the specific wants and needs of the disciplines in regards to each other. An example of this was that we as planning and navigation had to align the way we would store the location information of objects in the map with detections from the software of our perception colleagues.

Overall I really gained a lot of experience from the project as a whole and feel great affinity towards all of the different transferable skills. It showed me that I enjoy this way of working and that I would want to pursue a future career where this way of working would be required, for example at a start-up.

Andrei
Throughout the course, I have had the opportunity to develop and reflect upon several important skills, including teamwork, leadership, entrepreneurial thinking, and strategic multidisciplinary problem-solving.

In terms of teamwork, I have learned the value of effective collaboration and maintaining positive interactions with my team members. The peer feedback I received acknowledged my ability to guide the team and keep a good overview of tasks. However, I also received feedback about the potential tunnel vision and the need to fully listen to others. This feedback has made me more mindful of being open to different perspectives and actively engaging with my teammates' ideas.

Regarding leadership, I have made conscious efforts to demonstrate initiative and take on leadership responsibilities. The positive feedback I received about leadership and good planning indicates that these efforts have been fruitful. It is an area I find fulfilling and enjoyable, and I aim to continue developing my leadership skills.

In terms of strategic multidisciplinary problem-solving, the course has provided opportunities to tackle complex problems from various angles. While the feedback received did not explicitly address this skill, the overall project experience has allowed me to enhance my ability to analyze problems holistically, consider different perspectives, and develop well-rounded solutions.

Through the course, I have identified both strengths and weaknesses in these areas. My strengths include maintaining a good overview of tasks, demonstrating leadership and planning skills, and being receptive to ideas and seeking help. However, I have also identified areas for improvement, such as being more proactive, improving communication, and ensuring that I fully listen to others.

Overall, I am personally interested in leadership, entrepreneurial thinking, and strategic multidisciplinary problem-solving. These areas align with my passion for innovative approaches, taking initiative, and making a meaningful impact. The course has allowed me to identify and develop these interests further while also highlighting areas where I can continue to grow and enhance my skills.

6.4 Agile, Specialist Roles, and Other Team Roles

Joop
Within the given time frame we did not have time to implement Agile project management, we worked very hard to get a working version of the task running, but then there was no time to iteratively change the design with customer feedback. We received some feedback during the initial client presentation, that was considered in the project design, but getting a working project was top priority. Regarding my HRI specialist role, at the start of the development I was very much not sure what to do, given that a lot of it depended on results from other packages that had to be created. So I took responsibility for the initialisation and the state machine. The initialisation was done initially with terminal command but given that it felt like insufficient for my HRI role, I decided to make a GUI for the operator to use too. I felt like I was responsible within my role to make sure everything that the operator might need to see is shown. From this role I learned to be creative with solutions for a given task with the creation of a GUI with many features , given the limited and slightly vague role of HRI specialist.

Jelle
The agile project management was not a big part of our process, but some aspects were subconsciously applied . I personally experienced this course as quite a high workload which meant we were working from deadline to deadline. This did not leave a lot of time to reflect extensively on each sprint. Of course we did reflect on what work was done, but not always actively in an agile way. It is something that is quickly overlooked when a lot needs to be done. The course did have some guidance towards working in an agile way. For example by having intermediate reports. It was not always pleasant to have more deadlines, but it definitely helped to know if we were on the right track, which I liked. As for my specialist role, perception, I really enjoyed the task at hand. Together with Lars we divided the perception tasks, but in the meantime kept working together. I was mostly responsible for extending YOLOv5 to make it able to detect screwdrivers and for converting the detections into custom bounding box objects for the other subsystems to use. Actually seeing the screwdrivers being detected on the input images was really satisfying. I learned a lot about applying theory of computer vision into practice, which was one of my goals.

Lars
We were unsuccessful in applying Agile project management regarding the software on the final end product. We were only able to improve the software during the third and fourth sprint, but had no time to iteratively improve the actual design using customer feedback. During the second sprint we received feedback from the client (TNO), but we were not able to implement all of it due to lack of time. We planned on improving the software according to the ideas and feedback from the client once we had created a working end product. Despite this, I still enjoyed my role within the team; perceptionist. My perception partner and I enjoyed learning how to extend YOLOv5 in order to gain the ability to detect 'unseen' [1] objects. Our task was clear, but simultaneously very difficult because we had to learn how to extend an existing detection model and at the same time had to ensure that this model attained a high performance. My partner and I were able to implement a working detection model that attained moderate performance. I believe the task could have been executed with the current performance level, however, we did need quite some time to get it working the way it does. My perception partner and I worked together on mostly everything. However, I was responsible for creating or preparing all the nodes within the ROS package and having them communicating together properly. Additionally, I was also responsible for mapping the 2D detection to the 2D occupancy grid. From my role, I learnt exactly what I was aiming to learn; how to create software and implement that into a real robot.

Denniz
In this project, I had the opportunity to explore different project management methodologies, including Agile. Initially, I felt that Agile might be slightly overkill for our project due to the short timeline and the alignment of our team members' capabilities and vision. Reflecting at the end of each sprint proved challenging due to the high workload and looming deadlines. However, we still had moments of reflection through feedback from our supervisors and each other.

[1] Meaning that YOLOv5 has not been trained to detect this particular object class.

Although we didn't fully adhere to the structure of Agile and found it somewhat unnecessary at times, I appreciate the working principles it offers for projects with longer timelines and diverse team backgrounds. Agile can help keep everyone within the team focused on the bigger picture and aware of the work others are doing. The concept of working in sprints with predefined deliverables at the end of each one, helps maintain engagement.

Regarding my team roles as part of the Planning & Navigation team and serving as the Systems Engineer, I faced challenges in integrating these fields. While my primary focus was on ensuring a well-functioning solution within the Planning & Navigation role, the responsibility for creating the system's state machine was delegated to our HRI specialist. However, I still gained valuable insights from the project in this area. I enjoyed engaging in discussions on how different tasks could be separated and effectively combined to achieve a cohesive system. I often worked together with the different specialties and thought along with them on how to achieve our goals.

Andrei
I believe our introduction to Agile development has been unsuccessful, even though we strove to separate tasks between sprints. One key mistake we kept on doing was implementing the feedback per sprint as a last task, before finishing the next sprint.

One way to improve my Agile way of working is to have more clear separation of tasks and relate to the initial planning made, on what needs to be done in a specific week. Moreover, being more proactive and organising short progress meetings would be more helpful to convey the daily struggles and to also potentially get external ideas.

As a resource officer, my role would typically involve managing and coordinating the resources required for the project. This included ensuring that the necessary tools, equipment, materials, and human resources are available and allocated effectively. In this role, I discovered the importance of resource management and the impact it has on project progress and success. I believe I developed skills in organizing and optimizing resources, ensuring their availability when needed, and resolving any resource-related challenges.

6.5 Other Key Reflection Conclusions

Joop
I had a hard time managing the time this course required especially since I was also doing 2 other courses at the same time, but am proud of myself for having managed to do so.

Jelle
I would like to reflect on my last learning goal of section 6.1 which I have not addressed yet. I wanted to learning about working with a client. I realised that I can get caught up too much in the technical and practical parts of a project, where that should not always be the case. I realised that not everything is just about technical stuff, but the way you present your idea is also very important. Even if you have a really good idea and a really good implementation of that idea, it does not matter as much if you cannot convince your client of your approach.

Lars
During this course, I noticed at the start that the workload was heavy. When my team and I were creating the actual software, I made deadlines for myself. In the beginning, the deadlines I set for myself were a bit too optimistic, which led to me having to make long hours and trade sleep for more study hours. Noticeably, the amount of sleep I lost, had a negative impact on the quality of my work. From this, I learnt that I should plan better ahead and always assume extra time is needed.

Denniz
One key reflection that I had, is that I would have liked the project to be slightly longer. I believe that we where at the brink of getting a fully functioning product and that we would have managed to implement everything if we had one or two extra weeks. On the other hand, this thought me that also an unfinished product has valuable parts to show to the world and I learned that this also requires a certain way of selling the product.

Andrei
Our unsuccessful separation of tasks led to not visualising the bigger picture and relating back to the overall vision of the project. Often times I have also felt frustrated with the performance of my tasks individually and strove to perfect them. I believe this could've been seen in the final presentation where the client was not specifically looking for technical details, but for a more convincing overall picture.

Conclusion

In conclusion this multidisciplinary project has been a very educational experience, encompassing different disciplines, but where teamwork was the key to prosper. This conclusion will give a comprehensive summary of our journey from creating our vision to actually bringing this vision into practise. The team's vision was simple: "Use Spot to minimise time spent on searching for lost items." To achieve this vision, Spot should be able to detect object and should be able to deliver them to the correct person. After the operational scenarios and requirements we analysed extensively, the functional hierarchy was designed. Spot should be able to recognise apples, ball and screwdrivers and it should be able to recognise people by the color of their shirts. To do so the robot needs to know where is it and where people and objects are. To obtain the objects and to deliver them a navigation system is needed. All of this has to be done in a safe and transparent way where it is clear what the robot's intentions are.

To detect humans and objects we used YOLOv5. This model is able to detect humans, apples and balls, but not screwdrivers. To solve this problem the YOLO model was extended by utilizing transfer learning, training on a thousand hand annotated images. The object could be mapped into 3D by combining the middle of the bounding boxes and their closest corresponding depth camera points and transforming these points into the world mapping.

To navigate in the free space a 2D occupancy grid was created. Spot's odometry could be imprecise and would drift. Therefore the simultaneous localisation and mapping problem was solved by using the Rao-Blackwellised particle filter called gmapping. Gmapping could be used by combining the depth camera images and transforming them into a pointcloud and then into laser scans. To navigate in our created mapping we used the move_base package, which has a built in global and local planner and is easy to integrate with the finite state machine. The planner samples random points in free space and uses the points with maximum distance between them as exploration frontiers.

The decision handler, implemented as a finite state machine (FSM), plays a crucial role in connecting various functions and controlling Spot's operations. The FSM progresses sequentially, with certain action servers becoming active only at specific states. Notably, the 'scan' and 'explore' actions are significant. The decision handler manages initialisation, task selection and states of other submodules. All of this is displayed in an easy to use graphical user interface.

Individually everything worked well both in simulation and on the real robot. However after integrating the sub-modules the performance was up our expectations. The performance of perception, planning and navigation decreased. This was likely due to computational limitations when running everything simultaneously. With more time and effort, these issues could probably have been resolved.

All in all the multidisciplinary project was an insightful and fun project where a lot was learned. The knowledge, skills, and experiences gained throughout this project will undoubtedly shape our future pursuits in the field of robotics engineering, paving the way for innovative contributions to the field.

References

[1] Boston Dynamics. *Spot User Guide R2.0*. [PDF]. Generation Robots. 2020. URL: https://www.generationrobots.com/media/spot-boston-dynamics/spot-user-guide-r2.0-va.pdf.

[2] Craiyon. *Crainyon Free online AI image generator*. https://www.craiyon.com/. Accessed 2023.

[3] *How to Train a YOLOv5 Model On a Custom Dataset*. https://blog.roboflow.com/how-to-train-yolov5-on-a-custom-dataset/. Accessed: 2023-05-25.

[4] IDRLabs.com. *Team Role test*. 2023. URL: https://www.idrlabs.com/team-role/test.php.

[5] ROS. *ROS - Robot Operating System*. https://www.ros.org/. Accessed 2023.

[6] ROS Wiki. *depth_image_proc*. http://wiki.ros.org/depth_image_proc. Accessed 2023.

[7] ROS Wiki. *gmapping*. http://wiki.ros.org/gmapping. Accessed 2023.

[8] ROS Wiki. *ira_laser_tools*. https://wiki.ros.org/ira_laser_tools. Accessed 2023.

[9] ROS Wiki. *Master*$_{S}lave_{A}PIs$. http://wiki.ros.org/ROS/Master_Slave_APIs. Accessed: 2023-05.

[10] ROS Wiki. *move_base*. https://wiki.ros.org/move_base. Accessed 2023.

[11] TNO. *TNO - Nederlandse Organisatie voor Toegepast Natuurwetenschappelijk Onderzoek*. https://www.tno.nl/nl/. Accessed 2023.

[12] TU Delft Robotics. *Multidisciplinary project – RO47007*. Delft University of Technology. Delft, Netherlands, 2023.

Change Log

Copy and paste all received feedback here and explain what you did with that feedback in the new version of the report. Next to this, report new additions and/or focus areas.

Date	Feedback or Update	Description
03/05	Update statement	Finalise for Sprint 1 Chapter 1, and start working on Chapter 2, Project Plan requires WBS and Gantt chart to be created and Vision chapter requires more detail.
05/05	Update statement	Finalise for Sprint 1 Chapter 2, add WBS and Gantt chart to appendix, everyone write Section 6.1 regarding personal learning goals.
09/05	Update statement	Determine with the group which operational scenarios there are and which nodes there will have to be.
10/05	Update statement	Implement operational scenarios and nodes into report.
12/05	Update statement	Based on the operational scenarios, extend these for section 3.1. After this, determine the needs and functional requirements for section 3.2.
12/05	Feedback statement	Implement all feedback from the previous report made during Sprint 1.
12/05	Update statement	Start making the functional hierarchy tree for section 3.3.
13/05	Update statement	Finish 3.2 and start making activity diagram in section 3.4.
15/05	Update statement	Each individual can voluntarily choose to update their learning goals in section 4.1.
9/6	Feedback statement	Implement majority of the feedback received for Sprint 2.
8/6	Update statement	Start writing section 4, of all the different sections of the software architecture
9/6	Update statement	Finish writing section 4, start on section 5 but this will have to wait until after the final session with the real robot.
9/6	Update statement	Each individual can voluntarily choose to update their personal reflection section in chapter 6.
16/6	Feedback statement	Implement the feedback received for Sprint 3.
19/6	Update statement	Finish writing final report. Reevaluate all previous reports and finish chapter 5 and 6.

Work Breakdown Structure

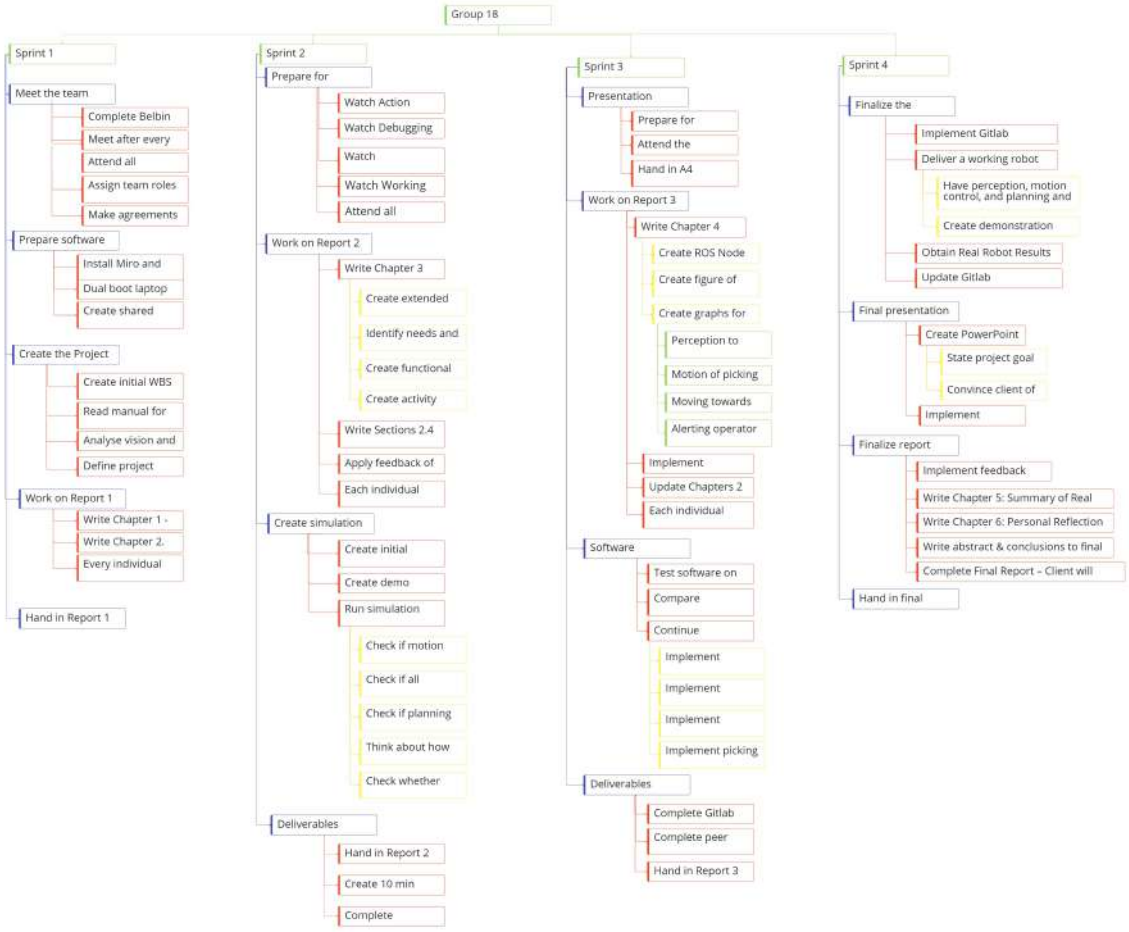


Figure B.1: Work Breakdown Structure of our project, text has scaled with zoom, for more detail look at https://miro.com/app/board/uXjVMRJioYA=/

Gantt Chart

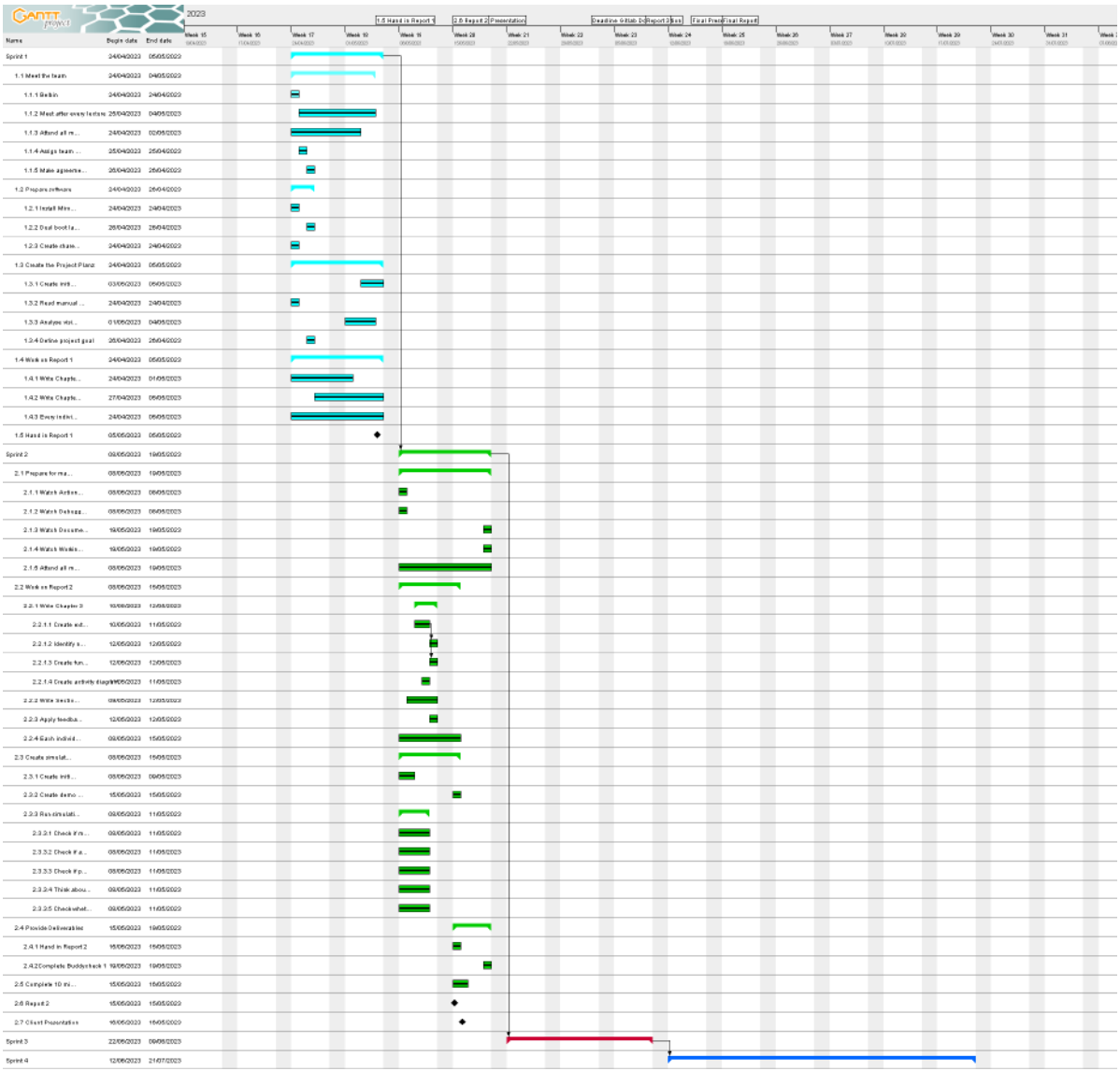


Figure C.1: Gantt chart of our project, with details displayed on sprints 1 and 2. Alternatively the Gantt chart can be downloaded from https://drive.google.com/file/d/1eEiQitDEZxzTy2Hst_xkkdlVyV4NHpA6/view?usp=sharing and opened with GanttProject https://www.ganttproject.biz/.

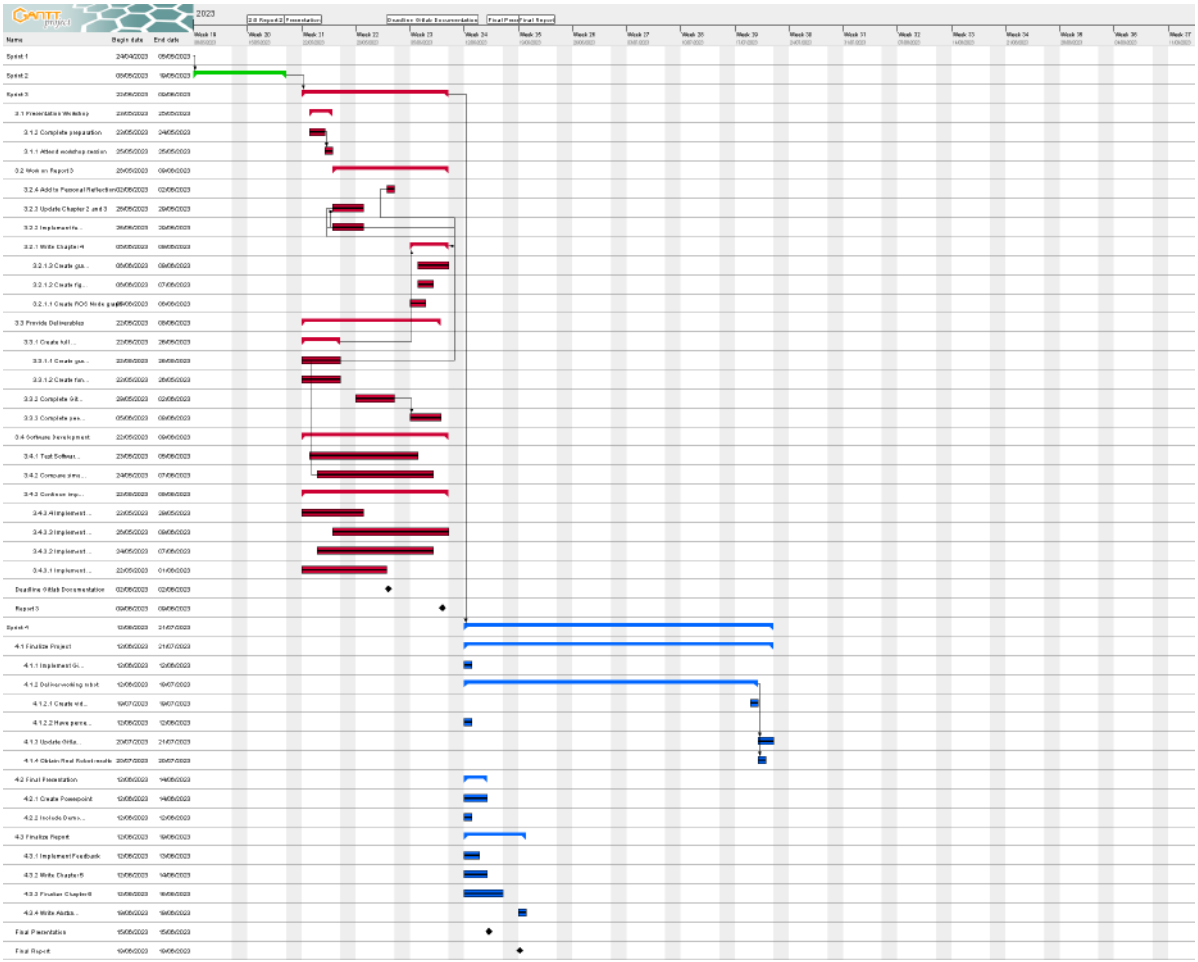


Figure C.2: Gantt chart of our project, with details displayed on sprints 3 and 4. Alternatively the Gantt chart can be downloaded from https://drive.google.com/file/d/1eEiQitDEZxzTy2Hst_xkkdlVyV4NHpA6/view?usp=sharing and opened with GanttProject https://www.ganttproject.biz/.