RAM. ROBOTICS AND MECHATRONICS

IMPROVING THE AUTONOMOUS NAVIGATION OF A CARE ROBOT BY FOLLOWING SOCIAL NORMS IN A CARE ENVIRONMENT

D. (Dharanish) Nambinayakanahalli Hanumantha

MSC ASSIGNMENT

Committee:
dr. ir. J.F. Broenink
C. Heemskerk, Ph.D (HiT)
dr. ing. Englebienne

October, 2023

053RaM2023
Robotics and Mechatronics
EEMCS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

UNIVERSITY OF TWENTE. | TECHMED CENTRE
UNIVERSITY OF TWENTE. | DIGITAL SOCIETY INSTITUTE

Summary

This thesis discusses the challenge faced in the Netherlands, caused by experiencing an increasing shortage of human resources to provide care for the growing population of individuals aged 65 years and above. To tackle this challenge, HIT is developing robot ROSE, a semi-autonomous care robot capable of performing basic and generic tasks in care facilities. However, the default navigating solution for ROSE to move around in the care facility while continuously interacting with humans regularly leads to awkward and unsafe situations. As a solution, this study aims to create a socially aware navigation system that considers social norms to make robots more socially acceptable in public spaces.

Primary social norms are identified by analysing human and robot behaviour in an indoor environment, and a new navigation design has been developed. Direct social norms are staying on the right side while navigating a corridor and respecting personal space. Additionally, norms were derived from analysing socially unacceptable robot navigation behaviour, which included not scaring people around intersections and avoiding forbidden zones.

The new navigation system design's performance has been evaluated using objective and subjective metrics. Objective metrics include the robot's path length to destination, number of recoveries, and time to destination. Subjective metrics include a survey among test subjects exposed to various simulated scenarios. The objective metrics indicate that socially aware navigation is more effective than default navigation behaviour, significantly reducing the number of recoveries and unpredictable behaviours. Socially aware navigation does take a bit longer time to reach the destination. The survey results indicate that the new navigation design is more socially acceptable than the default navigation behaviour.

To further enhance the robot's social acceptability and effectiveness in care facilities, the study recommends improving human-robot interaction space, considering human actions, incorporating verbal interaction, conducting real-life testing, and utilising learning algorithms.

This study contributes to social robotics by providing a simple approach to a new navigation design that considers social norms, making robots more acceptable and adaptable in public spaces.

Contents

1 Introduction

1.1 Context

In the next 20 years, the population aged 65 years and older in the Netherlands will nearly double (Kennisplein Zorg voor Beter, 2022). The increase in the older population requires more resources to care for them. This demand is overwhelming the care facilities as there is already a shortage of human resources (DutchNews, 2022). To improve the quality of care facilities, the Netherlands's health minister has called for a transformation in the long-term care sector by 2040. The minister is particularly interested in using technology to assist older people in caring for themselves for as long as feasible (Ministerie van Volksgezondheid, 2022). Robotics has become more prevalent in recent years as technology has advanced. This has created several new use cases, such as human-robot interaction, autonomous navigation, and artificial intelligence (Dzedzickis et al., 2021). Therefore, creating healthcare robots for seniors that enable them to lead inclusive but independent lifestyles in their homes may benefit elderly care.

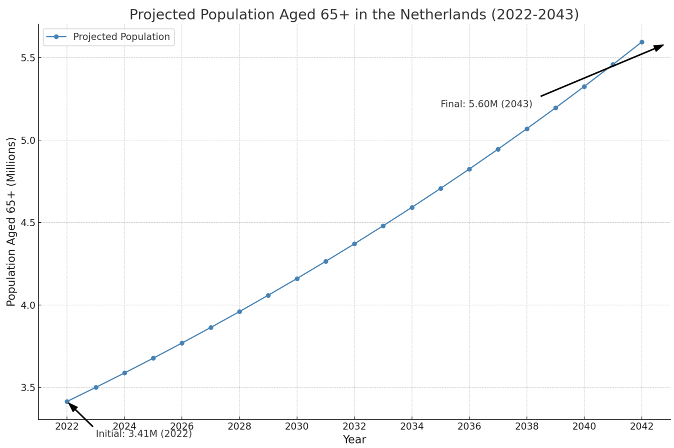


Figure 1.1: Projected Population of Individuals Aged 65 and Over in the Netherlands (2022-2043)

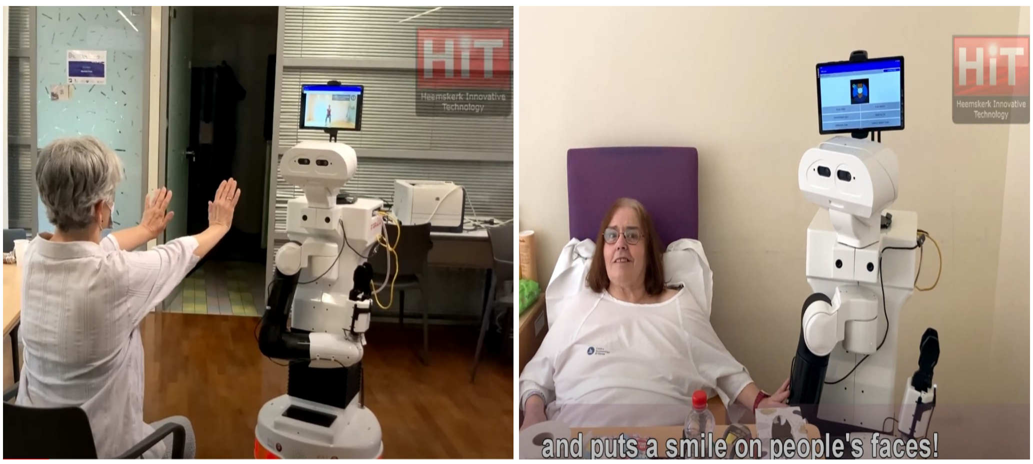


Figure 1.2: Robot ROSE helping patients in a care facility (HIT, 2022)

Under the name ROSE robot, Heemskerk Innovative Technology (HIT, 2021) is developing a semi-autonomous care robot solution. The robot is expected to perform basic and generic tasks autonomously. In contrast, for more complex tasks and dexterous manipulation, the robot is controlled by a remote operator via haptic telemanipulation (master-slave system) using a remote cockpit, as seen in Figure 1.3. The current company business model includes telemanipulation as a paid service for the customer, and the rest of the robot operation is autonomous. The autonomous services include navigating around the human environment efficiently and effectively.

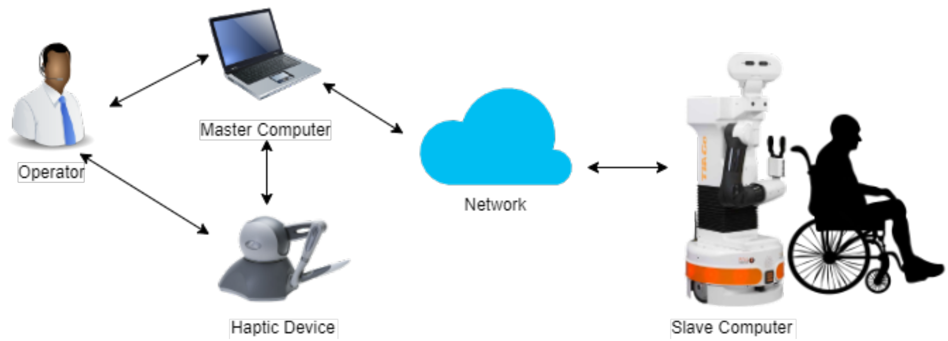


Figure 1.3: The architecture of ROSE the robot system. A remote operator will be able to control the care robot (via haptic telemanipulation)

Service robots are gaining popularity and attention in the care field due to this ageing society and the ever-increasing healthcare costs. The strong points of a hardware platform with limited autonomy, combined with the critical abilities of a remote human operator, are expected to lead to a breakthrough in the practical applicability of service robots.

However, the existing robot cannot be called 'mature' enough to be considered an authentic 'care product' because human awareness and socially acceptable navigation mean more than navigating the shortest route to reach the destination. The primary function of the care robot involves working around humans. This requires the robot to know the social norms, like respecting an individual's personal space. The current robot can navigate around with the knowledge of obstacle avoidance, but there is a need for further improvement to make the robot socially acceptable.

1.2 Related work

The field of robotics is advancing rapidly, especially in human-robot interaction contexts such as organisational and healthcare settings. Researchers have studied how workflow, social dynamics, and environmental factors influence robot technology effectiveness and acceptance in the workplace. For instance, researchers have emphasised that robots must navigate social environments effectively by interacting with humans appropriately while performing tasks (Mutlu and Forlizzi, 2008).

However, navigating dynamic human environments poses significant challenges because robots must understand complex factors such as human behaviour, social norms, and conventions . Sophisticated approaches are needed for a robot to abide by social rules while avoiding obstacles. The author proposed a unified social-aware navigation framework that requires carefully selecting components, including a global planner, local planner, prediction model, and suitable robot platform. This framework can help robots make informed decisions about their path and interactions with humans while adhering to social norms (Chik et al., 2016).

Trust is crucial in the acceptance of using robots (Kuipers, 2018; Pinker, 2012), which is why several methods have been proposed for enhancing the trustworthiness of robot navigation across different scenarios. For example, Che et al. (2020) introduced "social navigation", which uses explicit communication like speech or gestures and implicit communication like facial expressions to navigate through an environment while interacting with humans. Furthermore, proxemics theory has been used to develop algorithms for human-aware path planning in dynamic environments where multiple agents need to move around each other safely while maintaining appropriate personal space based on cultural context or individual preferences (Rios-Martinez et al., 2014).

To develop effective agent systems that can operate in various social contexts, it is crucial to understand technical challenges and the specific context itself (Mellema et al., 2020). Jeong et al. (2022) propose a method for managing traffic for multiple automated mobile robots in a crowded public place using a layered cost-map approach. The authors highlight how considering social factors such as human behaviour patterns and norms when designing algorithms can be critical when deploying navigational systems into these situations.

Human-aware navigation frameworks like those developed in Karageorgos (2017) have also been suggested as potential solutions for addressing issues associated with navigating dynamic human environments. The author introduced novel care-robot navigation that considers human occupants' presence and behaviour within domestic settings. This approach combines sensor data from cameras or microphones and machine learning techniques, allowing predictive modelling of movement patterns. The result helps improve autonomous movement around obstacles without colliding or being too close to people.

Previous studies have utilised standardised scales and customised questions to measure people's perceptions of a robot's sociability. For example, Vega et al. (2019) used three questions to evaluate how a mobile robot interacts with individuals: "Is the robot's behaviour socially appropriate?", "Is the robot's behaviour friendly?" and "Does the robot comprehend the social context and interaction?" However, the most effective way to measure sociability is still unresolved, despite a consensus on evaluation metrics for navigation and trajectory similarity (Gao and Huang, 2022).

The Perceived Social Intelligence (PSI) scale, introduced by Barchard et al. (2020), is a tool used to evaluate the social intelligence of robots. It assesses 20 aspects of robotic social intelligence, such as social competence, social awareness, social insensitivity (reversed), and strong social skills. The Social Competence (SOC) scale consists of four items. These scales enable researchers to assess the social abilities of robots, which is crucial in enhancing robot-human interactions. Moreover, it has been discovered that robots utilizing socially aware navigation planners are perceived to have higher social intelligence, as measured by PSI, than those that use traditional navigation planners (Honour et al., 2021).

1.3 Project Goals and Problem Statement

The related work section of this project highlights the increasing interest in human-robot interaction and the challenges associated with navigating dynamic human environments. Social and human-aware navigation frameworks have been suggested as potential solutions to overcome these challenges. Trust in robots and understanding social norms have been identified as crucial factors for their acceptance and effectiveness. This project aims to develop a more socially accepted care robot in terms of navigation behaviour. The care robot ROSE is designed to assist healthcare staff with daily caring for patients in a care facility. The robot must continually interact with humans while navigating around the facility. If the robot does not know how to navigate correctly in the given environment with humans, it raises concerns about its social acceptability. A well-behaved care robot must be able to localise itself within the environment, plan a path to a goal, follow the way while avoiding dynamic obstacles, and follow social norms while navigating the facility. When people and robots navigate the same physical place, standards must be established and respected to create social order. This graduation assignment focuses on integrating social norms with motion planning to improve the robot's social behaviour.

In general, a well-behaved care robot must be able to:

1. localises itself within the environment by estimating the robot's position and orientation on a static map,
2. plan a path to a goal,
3. follow the path while avoiding dynamic obstacles and
4. follow social norms while navigating around the facility.

1.3.1 Current status of ROSE capabilities

The robot encountered several difficulties while testing navigation tasks with care. When trials with Robot ROSE were conducted in a real-life care environment. The robot's behaviour was not always socially acceptable, which caused concern among the nurses and patients. For example, the robot would cut corners, startling the nurse, or become stuck in a congested corridor and fail to recover socially appropriately. Additionally, the robot would sometimes travel behind the nurse's desk, making it difficult for the nurse to move around freely. Furthermore, the robot would approach people too closely, making them uneasy. In conclusion, the testing of navigation tasks with care Robot ROSE conducted in a real-life care environment revealed several difficulties, including socially unacceptable behaviour, corner cutting, congestion issues, and invasion of personal space (HIT, 2023, 2022).

Figure 1.4: Still image from the end user test, where the robot exits the room and enters the corridor with high speed.

1.4 Research Questions

Standard navigation by moving between places along the shortest route is not sufficient for a care robot to be socially accepted. The improvements needed include making the robot understand its work environment and follow fundamental social norms without causing unexpected social behaviour. Give humans more interaction space while simultaneously respecting their personal space. Therefore, the main research question is:

> *RQ.1 How to make a care robot's navigation more socially acceptable by following social norms in a care environment?*

The thesis aims to include social norms for the robot. Following social norms increases the social acceptability and predictability of robots' behaviour. To understand how to work with humans, the robot should understand how humans function in an environment following social norms. Hence, the following sub-questions help in defining the main question further.

> *SRQ.1 What social norms should a care robot follow while navigating around the care facility?*

Sub-research question 1 helps to analyse the current robot condition and to identify the required social norms,

> *SRQ.2 How to implement social norms on a care robot?*

Sub-research question 2 helps to understand how the social norms can be implemented at the design level on a care robot.

> *SRQ.3 How to measure the effectiveness of a care robot following social norms in a real environment using objective and subjective measures?*

Lastly, Sub-research question 3 helps analyse robots' effectiveness in following social norms regarding subjective and objective measures.

1.5 Report organisation

The thesis report is structured as follows. Chapter 2 offers a comprehensive background on social navigation. In Chapter 3, the translation of social norms to robots navigating in natural care environments is analysed based on end-user tests caused by default robot navigation behaviour. Chapter 4 introduces a new concept that focuses on navigating in a socially acceptable manner. This includes the implementation of social norms such as staying on the right-hand side, defining workable areas and no-go zones, and improving human-robot interaction space. Chapter 5 presents a series of experiments based on real-world scenarios where social acceptability is measured by a combination of metrics and an evaluated survey, followed by a discussion. Finally, Chapter 6 concludes this report by summarising the findings and providing recommendations.

2 Background

2.1 Social behaviour and Human-Robot relation

Social behaviour is how humans behave differently depending on their surroundings. It includes acts directed towards others and addresses the importance of social interaction, culture, ethics, interpersonal relationships, politics, and conflict. Social norms determine the acceptability of behaviour and are governed by various forms of social control. Humans are encouraged to follow specific standards and exhibit behaviours that are acceptable or unacceptable based on society or culture.

Social norms are implemented to discourage individuals from exploiting vulnerability, violating trust, and thus preventing cooperation. Norms provide us with an expected understanding of how to act and help to keep society orderly and predictable. Some social norms are so firmly embedded in our minds that we don't even consider them; we do what is expected. Breaking social norms can occasionally result in awkward or uncomfortable situations.

Understanding and following expectations is an aspect of respecting those around us. By knowing and adhering to the etiquette rules for behaviour, we contribute to a pleasant, organized and even safer world. In our society, these social norms guide people's actions. They help set expectations about how others should behave, enabling us to plan our actions effectively. Moreover, violating these norms often leads to consequences. The context-dependent nature of patterns highlights the importance of norms in maintaining order and cooperation within our communities.

As robots become more integrated into our society, it becomes essential to recognize the role of norms in human-robot interactions. Robots interacting with humans must be programmed to align their behaviour with standards. This is particularly critical in healthcare settings where patient comfort and safety are paramount. Therefore, when designing robots, it is crucial to strive for behaviours that resemble those of humans and make adherence to norms a part of human-robot interaction. By understanding and complying with these expectations, robots can ensure that their actions are perceived as acceptable.

2.2 Translating social norms to robots navigating rules in a care environment

Translating social norms to robot navigation within a caring environment holds crucial importance, as it is pivotal in confirming their social acceptability. The social norms in question, namely, maintaining a safe distance between individuals, respecting personal space, exhibiting kindness towards older people, avoiding accidental collisions, and adhering to a right-sided walk in corridors, are essential in preserving harmony and order in human interactions.

The infringement of these social norms can lead to severe repercussions. For instance, if a robot intrudes upon an individual's personal space, it may provoke discomfort or even panic, leading to an unfavourable opinion of the robot. Likewise, the inability to demonstrate compassion towards older people can culminate in them experiencing abandonment and being underappreciated.

In a care environment where individuals are already vulnerable, ignoring social norms can be harmful. For instance, if a robot suddenly halts in the middle of a corridor, it may unintentionally collide with someone, leading to physical harm, which can be particularly detrimental to older people. Moreover, if a robot disregards the right-sided walk-in corridors, it may cause traffic congestion, resulting in delays and frustration among individuals. Consequently, training robots to comply with social norms is crucial to guarantee their safe and efficient integration into the care environment.

By comparing the above situations with the list of social norms followed by humans in Section Section 2.1, we can understand which general social norms need to be considered for robot navigation inside the care facility. These social norms can be translated to improve robots' navigation.

Social Norms	Translating to robots navigation
Stay right	Stay right in narrow spaces
Do not invade other's personal space	Navigate in robots workable space
Be kind to people	Navigate around respecting personal space
Do not stop suddenly or block the path	When stuck in a narrow space, call for help

Table 2.1: This table outlines robot navigation rules based on social norms

Patient comfort and safety are of the highest importance in a healthcare facility. The robot's actions must be predictable and secure to avoid potential patient or personnel harm. The table outlines how social norms can be translated into robot navigation rules. From the testing conducted in a real care environment, as mentioned in Section 1.3.1, it is clear that the robot encountered some difficulties in adhering to these rules. To ensure the safety and comfort of patients and personnel in a healthcare setting, it is essential that these issues are addressed and solutions are devised that make the robot more socially acceptable.

2.3 Proxemics

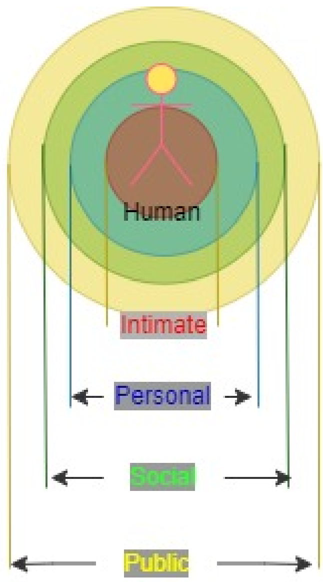


Figure 2.1: The four zones of Proxemics space

Proxemics studies how people use space and distance in social interactions. Hall et al. (1968), an American anthropologist, popularized the term in the 1960s. Hall proposes that people communicate and interact with others through different zones of personal space. Namely,

- Intimate space (closer than 0.45m)
- Personal space (0.45m to 1.2m)
- Social space (1.2m to 3.65m)
- Public space (further than 3.65m)

Hall also suggested that different cultures have different norms for personal space usage. For example, people in some cultures may stand closer together during conversations, whereas people in others may maintain a greater distance. He also claimed that different zones of personal space are used by different people in different situations. When interacting with strangers, for example, people may use a larger personal space than when interacting with friends or family. Robots and autonomous agents can be designed to better understand and interact with humans in social settings by understanding proxemics.

2.4 Navigation 2 Architecture

The Navigation2 (Nav2) is a production-grade, high-quality navigation framework for mobile robots. It has been adopted by over 50 companies worldwide and is regarded as the professional successor to the ROS Navigation Stack. Nav2's primary goal is to ensure the safe movement of a mobile robot, enabling it to complete complex tasks in various environments and robot kinematics. The system can move from one point to another, complete intermediate poses, and execute tasks such as object following (Macenski et al., 2020, 2022).

Nav2 employs behaviour trees to orchestrate numerous independent modular servers, creating intelligent and customized navigation behaviour. These servers can compute a path, control effort, recovery, or any other navigation-related task. Communication between these servers and the behaviour tree (BT) occurs over a ROS interface, such as an action server or service. A robot can use multiple behaviour trees to perform various unique tasks.

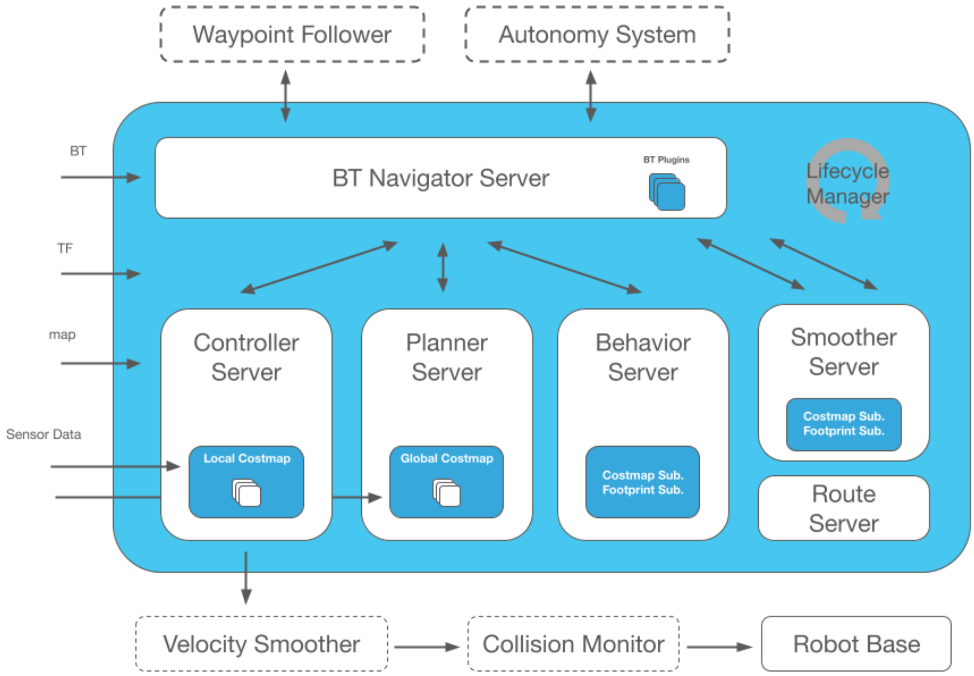


Figure 2.2: Nav2 architecture comprised with multi-functional servers. (Macenski et al., 2020)

The Nav2 framework, as shown in Figure 2.3, requires TF transformations, a map source for the Static Costmap Layer, a BT XML file, and any relevant sensor data sources as inputs. It then generates valid velocity commands for the motors of a robot. Nav2 supports major robot types, including holonomic, differential-drive, legged, and ackermann (car-like) base types with circular and arbitrarily shaped robots. It's worth noting that having multiple plugins for controllers, planners, and recoveries in each of their servers with matching BT plugins is possible. This can be used to create contextual navigation behaviours.

The Table 2.2 shows the functionalities of Nav2.

Component	Description
Map Server	Loads, serves, and stores maps.
AMCL	Localises the robot on the map.
Nav2 Planner	Plans a path from Point A to B around obstacles.
Nav2 Controller	Controls the robot as it follows the path.
Nav2 Smoother	Smoothens path plans to be more continuous and feasible.
Nav2 Costmap 2D	Converts sensor data into a costmap representation of the world.
Nav2 Behavior Trees and BT Navigator	Builds complicated robot behaviours using behaviour trees.
Nav2 Recoveries	Computes recovery behaviours in case of failure.
Nav2 Waypoint Follower	Follows sequential waypoints.
Nav2 Lifecycle Manager	Manages the lifecycle and watchdog for the servers.
Nav2 Core	Provides plugins to enable custom algorithms and behaviours.
Collision Monitor	monitor raw sensor data for imminent collision or dangerous situations.
Velocity Smoother	Guarantees dynamic feasibility of commands

Table 2.2: Navigation2 Components

2.4.1 Nav2 Costmap 2D

The robot's perception of its environment is facilitated by utilising a costmap, which serves as a structured 2D grid comprising cells, each assigned an appropriate cost value to denote its state—whether unknown, free, occupied, or inflated. This costmap is the foundation for various essential robotic tasks, including global path planning and local control adjustments.

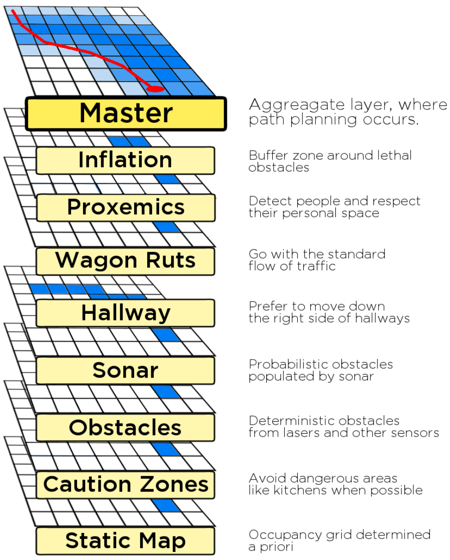


Figure 2.3: A stack of costmap layers, showcasing the different contextual behaviors achievable with the layered costmap approach.(Lu et al., 2014)

Costmap layers, developed by Lu et al. (2014), play a pivotal role in populating the costmap with data gathered from diverse sensors, including LIDAR, RADAR, sonar, depth cameras, and more. These layers empower developers to tailor the integration of data, enabling functions like obstacle detection, rule-based costmap modifications, and the real-time buffering of data into both 2D and 3D representations. In addition, the costmap relies on an occupancy grid representation to facilitate path planning and obstacle avoidance, further enhancing its accuracy.

The typical costmap layers are :

- Static Layer: This layer acquires a static map provided during initialization and integrates the associated occupancy information into the costmap.
- Obstacle Layer: Continuously updated, the obstacle layer employs raycasting techniques based on 2D laser scans to identify and mark empty spaces as free or occupied, enabling dynamic obstacle avoidance.
- Inflation Layer: Responsible for inflating costs related to lethal obstacles in the costmap, this layer employs exponential decay and convolution to expand cost values from occupied cells, ensuring safe robot navigation. Refer Section A.1 for more information on cost values.

2.4.2 Nav2 Planner

In ROS2, a planner server is an action server that hosts a collection of algorithm plugins to perform specific tasks. Its primary function is to generate a valid and potentially optimal path from the robot's current position to the desired destination. The planner has access to a global environmental representation and buffered sensor data to aid this process. Nav2 provides several planners that cater to different requirements, such as shortest path, complete coverage path, and paths along predefined or sparse routes. Different types of planners for different types of robots are shown below:

Planners	**Description**
NavFn Planner	A navigation function using A* or Dijkstras expansion, assumes 2D holonomic particle
Smac Planner 2D	A 2D A* implementation Using either 4 or 8 connected neighborhoods with smoother and multi-resolution query
Theta Star Planner	An implementaion of Theta* using either 4 or 8 connected neighborhoods, assumes the robot as a 2D holonomic particle
Smac Hybrid-A* Planner	A SE2 Hybrid-A* implementation using either Dubin or Reeds-shepp motion models with smoother and multi-resolution query. Cars, car-like, and ackermann vehicles. Kinematically feasible.
Smac Lattice Planner	An implementation of State Lattice Planner using pre-generated minimum control sets for kinematically feasible planning with any vehicle imaginable. Includes generator script for Ackermann, diff, omni, and legged robots.

Table 2.3: Different planner plugins available in Nav2

2.4.3 Nav2 Controller

The local planners, also known as controllers, use the representation of the surrounding environment to calculate the possible control efforts required by the robot base to move forward. The controller moves the robot in space and computes a feasible path at regular intervals of updates. Controllers can be programmed to navigate along a path, dock with a charging station using sensors in the odometric frame, board an elevator, or interact with a tool. The different controller plugins available are:

Controllers	Description
DWB Controller	A highly configurable DWA implementation with plugin interfaces
TEB Controller	A MPC-like controller suitable for ackermann, differential, and holonomic robots.
Regulated Pure Pursuit	A service / industrial robot variation on the pure pursuit algorithm with adaptive features.
MPPI Controller	A predictive MPC controller with modular and custom cost functions that can accomplish many tasks.
Rotation Shim Controller	A shim controller to rotate to path heading before passing to main controller for tracking.

Table 2.4: Different controller plugins available in Nav2

2.4.4 Nav2 Recoveries

The Recovery Behaviour Server is responsible for maintaining a fault-tolerant system. The goal of recoveries is to deal with unknown or failure conditions of the system and autonomously handle them. The current Nav2 stack supports the following recovery plugins:

Recoveries	Description
Spin	Rotate behavior of configurable angles to clear out free space and nudge robot out of potential local failures
Back Up	Back up behavior of configurable distance to back out of a situation where the robot is stuck
Wait	Wait behavior with configurable time to wait in case of time-based obstacle like human traffic or getting more sensor data
Drive On Heading	Drive on heading behavior with configurable distance to drive.
Assisted Teleop	AssistedTeleop behavior that scales teleop commands to prevent collisions.

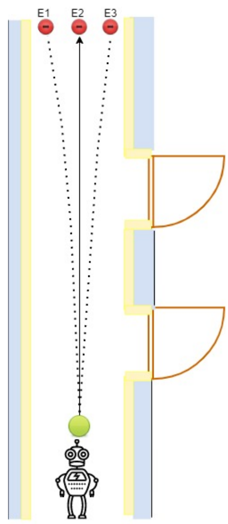
Table 2.5: Different recovery behaviour plugins available in Nav2

3 Analysis of current robot navigation behaviour

Considering a typical care facility ground plan, the robot has to navigate through different regions. These regions typically include the main hallway of the building, lobby, nurse station, storage room, and many patient rooms. By default, the robot behaves precisely the same in all the regions mentioned. For the robot, these regions are just spaces in its navigational map and all other things are considered obstacles, and the robot navigates in this map following the shortest path.

The care facility is generally populated with old people, and not everyone will be comfortable with the robot helping them. This can be due to a lack of experience or interaction with the robot in the past. In general, for a human to be comfortable, they should be able to know how the robot behaves. It would be difficult to explain the workings of the robot to everyone in a facility rather the robot should learn the social norms. Learning about these rules makes the robot behave more like a human while navigating around, and these behaviours are more predictable to the humans and create less discomfort around the robot. The following sections give an analysis of the current robot's behaviour in different scenarios,

3.1 Understanding areas in the environment



[A]

Figure 3.1: Illustration of default navigation system choosing the shortest path to destination

A socially awkward scenario that would be addressed in the project is caused by the robot not understanding the specific meaning and function of the different regions. Nurse stations, corridors, general areas, patient rooms, and storage rooms are some of the common areas in every care facility. Each area has its unique work function, which can be defined by the activity done in the area. For example, a nurse working in a care facility knows how to change his/her behaviour according to the area she is in. The nurse will be more careful with respect to the patient's personal space when inside the patient's room. Similarly, if a visitor of the care facility goes into a storage room or behind the nurse's desk, it would not be acceptable to the nurses. This would create a socially awkward scenario because the visitor would be considered an intruder in the nurse's working environment. The robot can also experience this exact problem.

By default, the robot's navigation system takes the shortest path to its destination with the most minor obstruction in its way. However, not all the shortest paths are socially acceptable.

Figure 3.1 shows an illustration of the default robot's behaviour. The green marker indicates the starting location, and the red marker indicates the ending destination. E1 / E2 / E3, anyone can be the destination at a time. The robot's global navigation path changes depending on the endpoints. The default planner plans the shortest route in the global map to the destination, avoiding obstacles as they come along.

The disadvantage of always taking the shortest path is illustrated with the help of Figure 3.2. The figure shows a scenario of the robot's default navigation behaviour. The robot plans the shortest route to the destination. But is it an acceptable path? The robot cuts the corner and will scare a person approaching through the corridor while navigating to the destination, the robot travels behind the nurse's desk. This behaviour is not socially acceptable to the nurse working there. The robot does not plan its path to take into consideration the function of the nurse's desk. When it reaches the corner, the obstacle gets detected by the robot's LIDAR sensor, and the robot changes its path to avoid the obstacle. The robot corrects its path by choosing the least obstructed path to the destination. The robot does not always go behind the desk sometimes it also takes the expected path, where the robot goes close to the desk and then makes a turn. The robot's behaviours show that it's unpredictable and does not consider the social factors that affect its social acceptability.

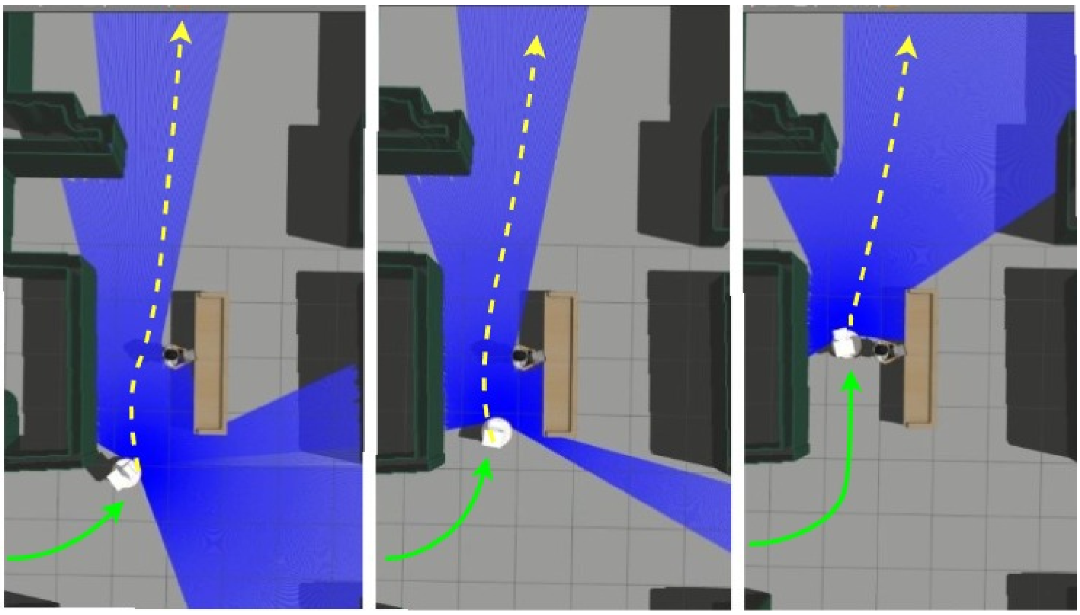


Figure 3.2: Default navigation choosing the shortest path to destination leading to a socially awkward situation with the robot passes directly behind the nurse, not colliding but entering the personal space, because the robot is regarded as an intruder in the nursing station .

3.2 Interaction in narrow corridor

The second type of socially awkward situation is caused by the robot not being aware of human presence and intentions. Such a situation is also observed between people navigating in different directions in a corridor, an intersection, or a pedestrian walkway.

Figure 3.3 illustrates human behaviour when navigating in a corridor. Figure 3.3.[A] shows two people following the social norm of staying on the right side, [B] shows a socially awkward scenario where both people take some time to correct their course, and [C] shows the least

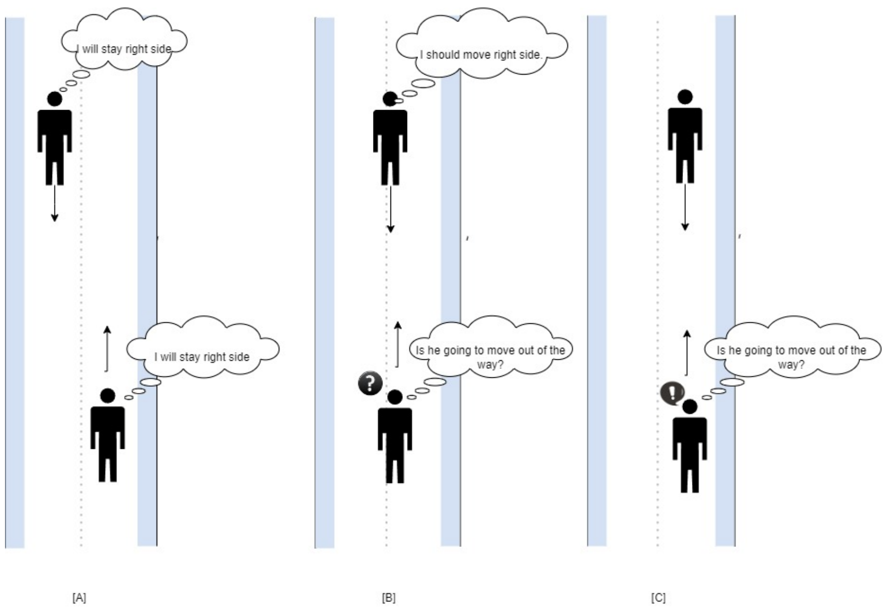


Figure 3.3: Illustration of two people walking in opposite directions in a corridor. Sometimes this leads to an awkward situation where people hesitate on which side to pass each other.

socially accepted situation where one person walks in the opposite direction, creating more confusion in the environment. The situation will become less predictable and more socially awkward behaviour in comparison with [A].

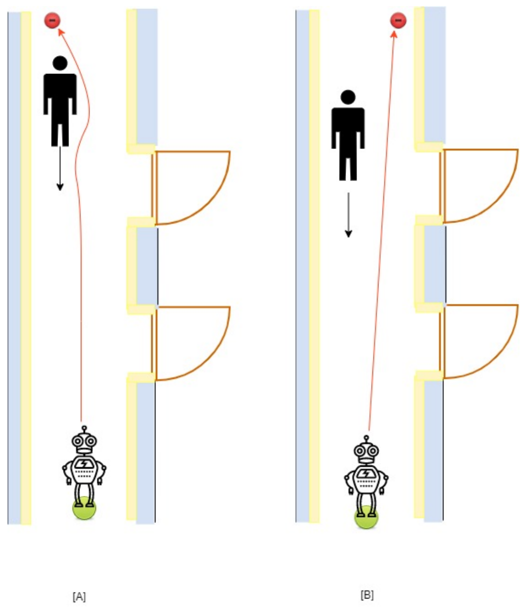


Figure 3.4: Illustration of human and robot navigating in the corridor. Depending on the destination, the shortest navigation may cause a socially awkward situation

When this social awkwardness arises between people, it gets rectified soon. People respect the personal space of others because of trust, assuming that other people would also do the same. The following illustration Figure 3.4 describes the same scenario with a human and a robot navigating in a corridor. By default, the robot has no particular knowledge of what social norms are. The robot's navigation behaviour changes based on the destination.

In Figure 3.4, the destination is behind the human or towards the left of the corridor. Considering the robot starts from the middle of the corridor, it takes the shortest route to the destination. While navigating, the robot tends to stay toward the left side of the corridor. When it detects a person (obstacle), it corrects its path and moves slightly toward the right to avoid a collision.

Humans still consider this situation awkward because the robot navigates too close to the human. On the other hand, if the destination is towards the right side of the corridor, the robot, by default, moves from the middle to the right, leaving more space for the person on its left to pass. This destination-dependent navigation behaviour is not easily predictable by normal people and hence creates a socially awkward scenario around people.

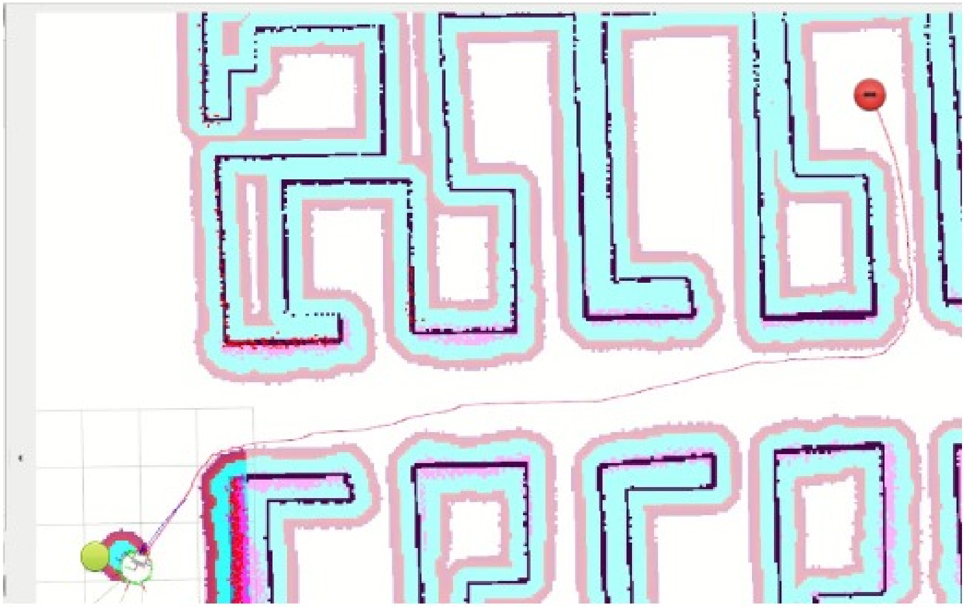


Figure 3.5: The above diagram shows a robot planning its path from the start location (green) to the end location in the room(red)

To analyze the above-mentioned situation, the below images give a better understanding of the problem. Figure 3.5 shows a situation where the robot navigates to a room through a corridor without any obstacles.

The robot again calculates the shortest route to the destination. The navigation path takes the robot in a diagonal path from the start of the corridor to the beginning of the room. This shows that the robot has no constraints while navigating in the corridor. Figure 3.6 shows the same behaviour in a different perspective where the robot navigates on the shortest route to the destination.

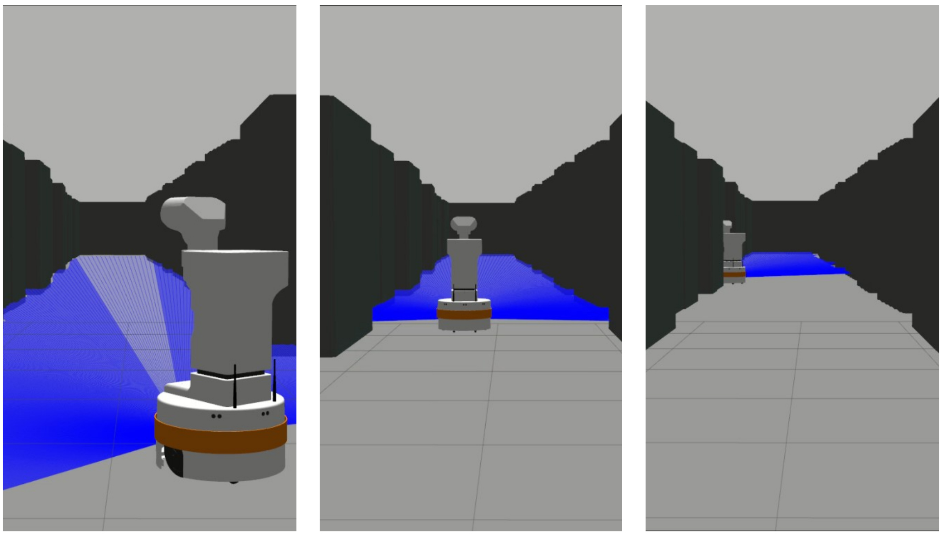


Figure 3.6: Robot navigating the shortest route through the corridor in simulated environment

3.3 Taking future interactions into account

The third scenario describes the robot's ability to scare people because it does not consider future interactions in the robot's environment. When people are about to cross or turn into an intersection, whether they are travelling by car or walking, they take into account future interactions, especially when they cannot see around a corner. They do this by slowing down and being more cautious of this potentially risky situation. They will check the surroundings and only move when it's safe. By default, the robot lacks this knowledge and only aims to find the shortest path to the destination.

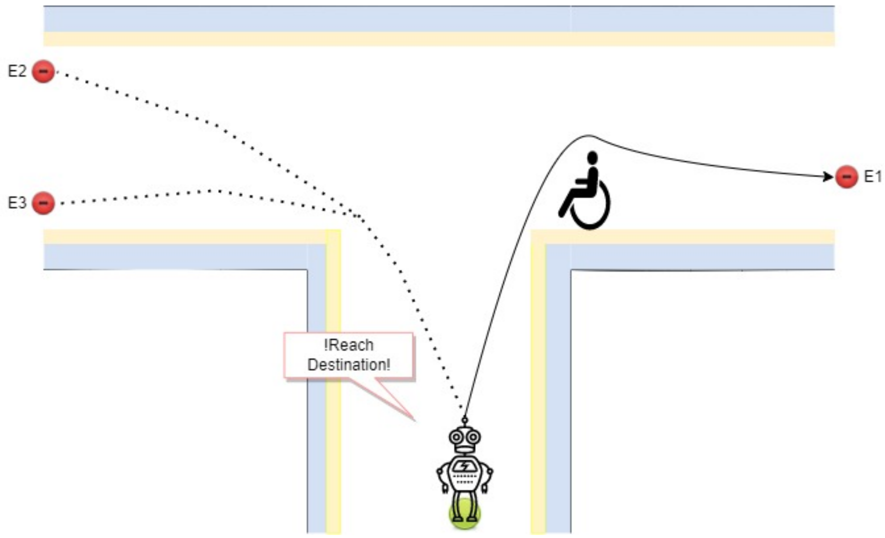


Figure 3.7: The path taken to the E1 destination does not collide with the person in the wheelchair, but it is socially awkward.

One common socially awkward scenario can be described with the help of the following illustration. By default, we know the robot takes the shortest route to the destination. Figure 3.7 is a situation where the robot is trying to go to E1, where a person is at the corner of the intersection. Before it reaches close to the end of the corridor, the robot does not see the person in a wheelchair. While navigating along the corridor, the robot doesn't see any obstacles and will reach the corner at maximum speed and start to turn the corner at the robot's maximum rotational speed. Then, the robot observes the person in the wheelchair and changes its course. Though it avoids collision, by moving so close to the person, people get scared, not just the person in a wheelchair.

With the default behaviour, the path taken to E2 is more socially acceptable to the person in a wheelchair as it unknowingly gives more space. However, the robot cuts the corner and may scare another person coming from E3. If the destination is E3, then the path is less socially acceptable because it is navigating on the wrong side of the corridor.

The above-mentioned scenarios are some of the basic situations where the robot fails to understand social factors that are required to work around people. In order to make it more socially acceptable, it is required to follow social norms like people. The improved work will help in mitigating the socially awkward problems faced by people while working with the robot.

4 Design implementation

This section of research will provide a detailed explanation of the design flow of the project. The main objective is to translate social norms into ROS2 navigation behaviour.

We will address the following questions:

- How to make the robot understand its workable environment?
- How to navigate in a narrow corridor while following social norms?
- How to handle intersections and corners?
- How to improve human-robot interaction in an indoor environment?

The research will provide a comprehensive answer to each of these questions, along with a discussion of the relevant social norms translation to ROS2 navigation workflow.

4.1 How to make the robot understand its workable environment?

Figure 4.1: Different work environment present in a typical care facility

By default, the robot does not know the differences between a room, corridor, nurse desk, etc. In order to make the robot understand the differences between the areas, we have to label the different areas on the map and to make the robot understand the difference between these areas. In this way, we can, for example, label the map with workable areas and non-workable areas for the robot. Note that, depending on the task to be performed by the robot, specific areas may change status and become workable or non-workable. For example, when the task is to clean the dining table, the robot is allowed to enter the dining table area. However, while navigating as a part of another task, the dining area is a forbidden zone. Another example is the area around the patient's bed; during an inspection task, it is a forbidden zone. While doing social interaction or entertainment tasks, the robot is allowed to get close to the patient.

Forbidden zones will force the robot to plan its path only within the workable area while navigating between two locations on the map. The forbidden zones are the areas where the robot cannot enter or plan its path. The forbidden zones here on the map are marked where the robot respects people's private space for various reasons: Respecting personal space around the bed in the patient's room, Nurse's desk, and dining area.

The fig.Figure 4.1 shows different zones marked on a map. The zones consist of all the places where the robot should avoid navigating unless the task explicitly requires the robot to enter the zone. These zones can define boundaries in patients' rooms around the nurse's station or any place where the robot should not enter. The boundaries are drawn on the map and then passed through the keep-out filter. The keep-out filter takes these boundaries and superimposes them on the original map. The new boundaries act as high-cost regions on the map; by default, the robot finds the shortest route, avoiding the high-cost regions.

4.1.1 Keepout Filter

The keepout filter implemented using the Navigation2 package in ROS is designed to prevent a robot from planning and executing paths that intersect with forbidden zones in the environment. This is achieved by defining polygons to represent the boundaries of these forbidden zones in the environment, which are specified in a YAML file and then used to build a representation of the environment, including the forbidden zones, when the Navigation2 stack is launched.

The filter utilizes an occupancy grid to represent the environment, where each cell of the grid is assigned a value based on its occupancy status. The values in the occupancy grid can be changed dynamically to reflect changes in the environment. When a forbidden zone is defined, the corresponding cells in the occupancy grid are assigned a high value to indicate that the robot should not enter these areas. The values assigned to these cells can be changed dynamically, for example, to allow the robot to enter a previously forbidden zone if the situation demands it.

As the robot navigates through the environment, the Navigation2 stack continuously updates its position and orientation. When the robot attempts to plan a path to a new location, the Navigation2 stack checks whether the path intersects with any of the forbidden zones. If it does, the path is rejected, and a new path is planned. This process continues until a valid path is found that avoids all the forbidden zones.

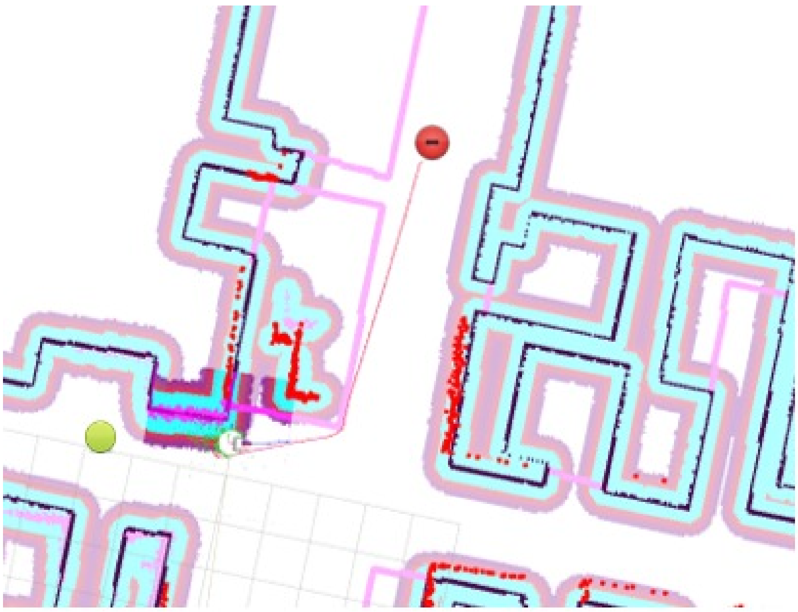


Figure 4.2: Improved navigation around forbidden zones

The keepout filter can be configured to operate in a variety of ways, depending on the specific needs of the application. For example, the keepout zone can be set to be a hard constraint, meaning that the robot is not allowed to enter the forbidden zones under any circumstances.

Alternatively, the keepout zone can be set to be a soft constraint, meaning that the robot is allowed to enter the forbidden zones if it is necessary to reach a goal location. In this case, the Navigation2 stack will attempt to find the path that minimizes the amount of time spent in the forbidden zones. The cost of a path is calculated based on the values in the occupancy grid, allowing for the use of cost maps to influence path planning.

Figure 4.2 shows how the robot plans its path around the forbidden zone. The robot always plans its path around the nurse's desk, irrespective of the endpoint. This way, the robot does not intrude on the workspace of other people.

4.2 How to navigate in a narrow corridor while following social norms?

The robot's behaviour in the corridor can be improved by making the robot follow social norms like staying on the right side of the corridor. Staying on the right-hand side can be implemented in a few methods like 'NavthroughPoses', 'Waypoint follower' and 'Keepout Filter'.

However, using the waypoint follower, the robot showed better behaviour than the other two methods. 'NavthoughPoses' is similar to a waypoint follower. The only difference is the robot does not stop after reaching the smaller goals. The keep-out filter, as explained in the previous section, helps in defining the workable area. The idea is to cover the left-hand side area according to the robot's direction of navigation, which comes with the disadvantage of giving narrow space for the robot to navigate around.

Figure 4.3: The image shows how a long path can be divided into smaller goals for the robot to achieve in order to reach the end location

The simple way for the robot to stay on the right side is by breaking down the long goals into smaller goals. The robot has to reach these small goals to reach the destination shown in Figure 4.3. Earlier in Section 3.2, we saw that the default navigation path in a corridor depends on the starting and ending location of the robot. Having smaller goals on the right side of the corridor guides the robot's ability to stay on the right-hand side of the corridor irrespective of the location of end goal.

4.2.1 Waypoint Follower

Staying on the right-hand side can be done by the ROS2 method Waypoint follower. The Figure 4.4 shows the working of the waypoint follower. Instead of giving one destination to the robot, it is given multiple destinations that it has to reach one after the other to reach the goal. In this method, the robot plans one path at a time and stops at every point for a second.

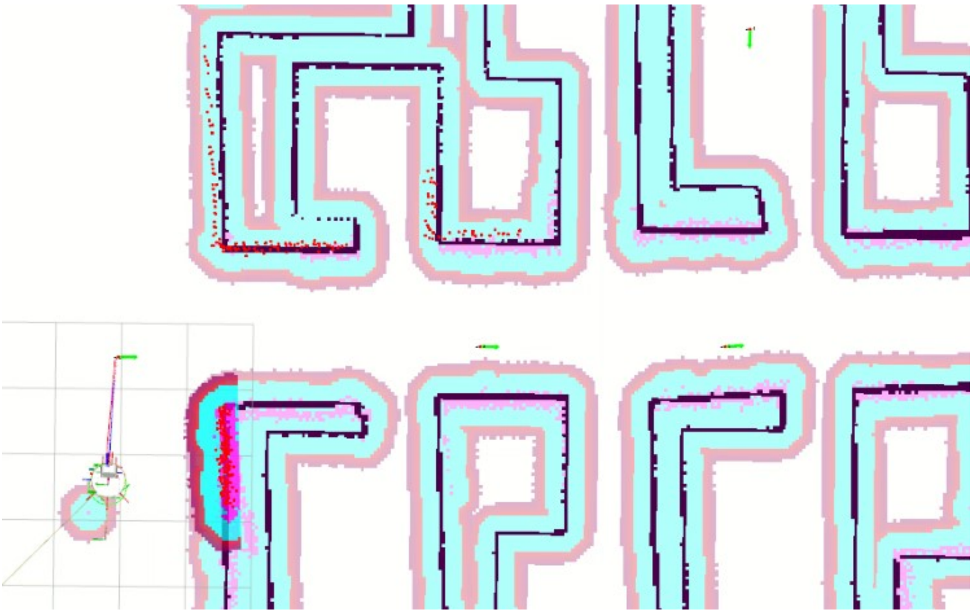


Figure 4.4: Improved navigation with waypoint follower. The green markers are shorter goals the robot has to follow to reach the destination. Starting from the left side, the robot follows the waypoint to reach the room on the top right

The *NavigateToPose* (NAV2, 2020)action plugin is designed to allow a robot to navigate to a goal. The plugin operates within the context of a Behavior Tree (BT), a hierarchical structure that is used to specify robot behaviours. The plugin is also responsible for generating a path that enables the robot to move from one pose to the other while avoiding any obstacles.

The implementation of the *NavigateToPose* plugin involves the creation of a BT XML file, which specifies the hierarchy of the BT and the parameters required by the plugin. The BT XML file is then loaded into the Navigation2 stack when it is launched. The plugin can be customized to meet specific application requirements by modifying the BT XML file. The BT XML file can be used to specify the target poses, the desired path planning algorithm, and other parameters such as the speed and acceleration of the robot. The plugin can also be combined with other action plugins to create more complex robot behaviours.

Nav2 waypoint follower is an extended application of *NavigateToPose* used to complete an orchestrated task. The package accepts an array of waypoints, which are sets of goals for one navigation cycle. The waypoint follower keeps track of these goals and gives feedback on the current index of the waypoint. The follower moves on to the next waypoint when the current waypoint fails and returns the lists of waypoints it was unable to complete (Macenski, 2020).

The waypoints are strategically placed between start and end locations. The number of waypoints varies according to the distance to the destination. The location of waypoints was followed to keep the robot on the right side of the corridor. The key locations are opposite lanes to the entrance of the room, entering/exiting the intersection and breaking down the long corridor routes into more minor routes approximately two rooms distance between each waypoint. Also, make sure the waypoints are not directly in front of the room, as shown in Figure 4.3. Adding more waypoints causes the robot to take more time to reach the destination.

4.3 How to handle intersections and corners ?

4.3.1 Speed Filter

The third type of socially awkward behaviour involves scaring people around the corners and intersections. This situation can be mitigated by reducing the robot's speed around these areas using a speed filter. The Speed Filter is an essential component of the costmap mentioned in Section 2.4.1, which calculates the maximum speed at each point on the robot's path using the occupancy grid and a filter mask. The implementation of the Speed Filter involves drawing a filter mask that annotates the map with requested zones, similar to the Keepout Filter. However, the OccupancyGrid mask values for the Speed Filter encode speed limits for the corresponding areas on the map (Nav2SpeedFilter, 2020).

To elaborate, the filter mask for the Speed Filter is created by assigning values to each cell in the OccupancyGrid map. These values represent the maximum speed that the robot can travel in the corresponding area. The higher the value, the faster the robot can move in that area, and vice versa. For example, a region on the occupancy grid with a high value, indicating an obstacle or obstruction, will be shaded with a darker colour. When the Speed Filter is applied, the higher mask values in that region will result in a lower speed limit for the robot in that area. Conversely, areas on the map with lower mask values, representing open and unobstructed spaces, will allow for higher speed limits.

During operation, the Speed Filter uses this filter mask to dynamically adjust the robot's maximum velocity based on the current location and environment. As the robot navigates through the map, the filter mask is continuously updated based on the current occupancy and velocity constraints. The speed limit at a particular point on the robot's path is calculated using the formula.

$$\text{speed\_limit} = \text{filter\_mask\_data} \cdot \text{multiplier} + \text{base} \tag{4.1}$$

, where the filter mask data represents the mask values from the occupancy grid for that point and based on the requirement, multiplier and base values are set. The multiplier is used to scale the mask values, and the base is added to the scaled mask values to determine the speed limit.

By adjusting the multiplier and base values, one can customize the speed limits for different areas on the map, ensuring safe and efficient navigation for the robot. This approach allows the robot to safely navigate through environments with varying speed limits, such as areas with narrow passages or obstacles. By restricting the robot's maximum speed in certain areas, the Speed Filter helps to prevent collisions and ensure smooth and efficient navigation.

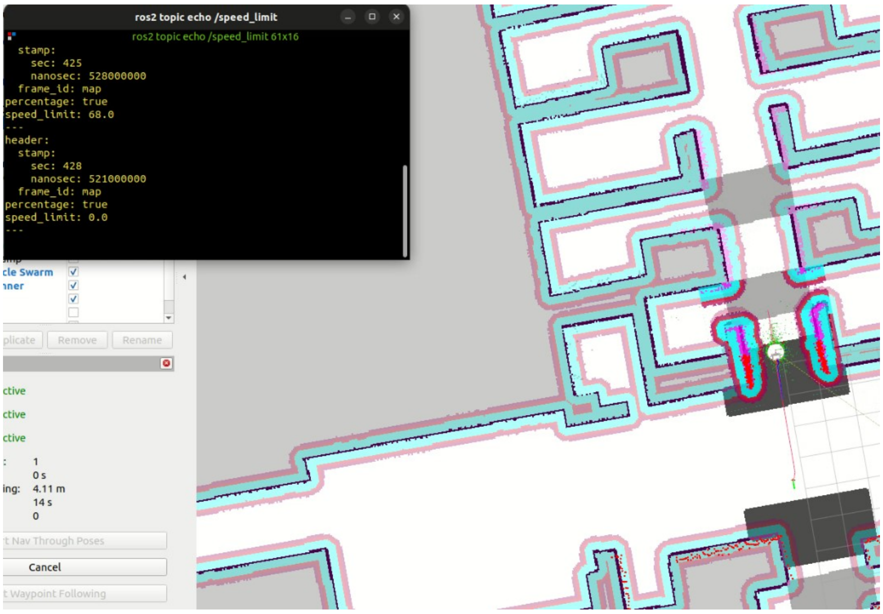


Figure 4.5: Speed zones marked on a real hospital map. Darker regions represent areas with the highest speed restrictions around the main intersection. Lighter regions represent areas with lesser speed restrictions around intersections of the room. The robot travel at 60 per cent of the maximum speed while navigating around an intersection. Where the darker region shows the area of reduced speed.

4.4 How to improve human-robot interaction in an indoor environment?

The fourth type of socially awkward behaviour is about respecting personal space. By default, the robot does not differentiate between the obstacles in its environment. Hence, it gives the same priority to all obstacles. So humans, desks, wheelchairs, and all other objects are treated as the same type of obstacle. Explicitly detecting humans and distinguishing them from other obstacles in the environment is the basis for different behaviours.

When detected, the robot can travel more cautiously based on proxemics theory (Rios-Martinez et al., 2014). Respecting people's personal space improves people's trust in robots while working in the same environment.

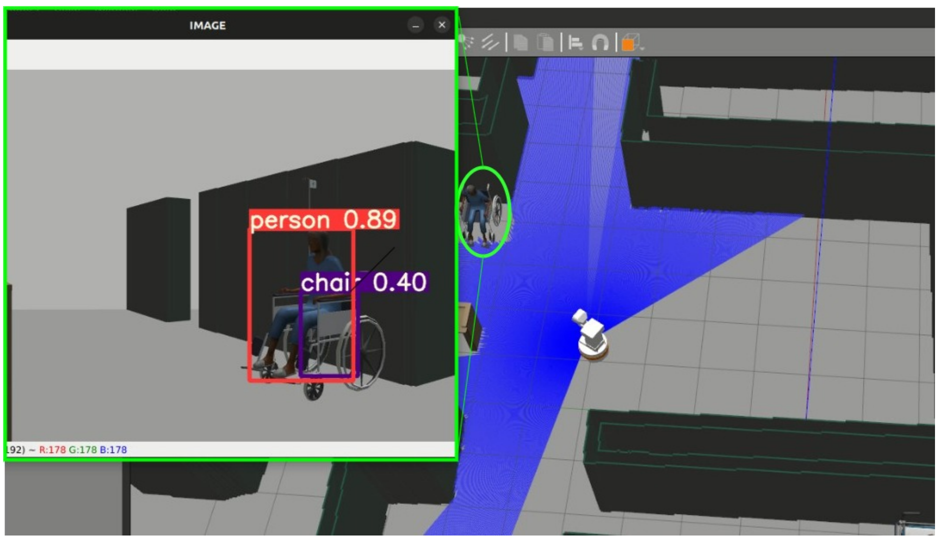


Figure 4.6: Human and object detection by the robot

Humans and objects are detected through the object recognition process. Yolo V5 software developed by Jocher et al. (2022) is used to recognize humans and objects as shown in Figure 4.6. Through the detected poses, the distance to the robot can be evaluated. However, in the simulation, the location of the model is already known. Hence, the position of the human model and objects can be identified in the simulation

When the robot detects humans in its surroundings, the distance is known. Based on the proxemic zones mentioned in Section 2.3, the robot restricts its speed based on the interaction distance. The robot travels at 80% of the original robot speed when in public space between 3.6m and 5m, 40% of the original speed when in social space, 20% when in personal space, and stops moving forward when in intimate space. Reducing speed when around the people reduces uncertain behaviour of the robot.

5 Experiment Design

The Experiment Design chapter outlines the methodology and hypotheses for the study to evaluate the effectiveness of an improved planner for care robots. The study uses mixed metrics to assess the planner's performance, including path length, Time to destination, number of backups, proxemics, and social comfort.

5.1 Methodology

Figure 5.1: Robot ROSE navigating in the corridor in a real-world hospital

The primary goal of this experiment is to evaluate whether introducing social norms leads to more socially aware navigation compared to default navigation behaviour. To what extent a care robot follows the social norms (using the new method) better than the default is assessed using standardized metrics, and the results are analyzed in two ways.

First, the robot's navigation behaviour is evaluated using primary odometry data, rendering metrics such as Time taken, distance covered, and collisions. Secondly, the robot's performance is evaluated using a survey by humans. Humans are presumed to provide a better assessment of social norms concerning safety and social acceptability. Together, the performance can be analysed both with subjective and objective measures.

Standard metrics that provide quantitative data for analysis are essential to evaluate socially aware navigation instead of default navigation behaviour. We can identify areas that require improvement by observing the robot's performance based on odometry data, such as the Time taken, number of recoveries and path length avoided during its path planning process.

Furthermore, analyzing human feedback through surveys provides a qualitative evaluation of the robot's behaviour in terms of safety and social acceptability. We can refine our interactions by understanding how people perceive robots' actions in their environment.

The test procedure involves simulation on a map of a real-life hospital Figure 5.1. In Figure 5.2 shows a map divided into three regions: A, B, and C. Where 'A' (A1, A2, A3, A4) represent all the regions that have patient rooms in it, 'B' (B1, B2) represents the general area of the hospital like the nurse station, dining area, waiting area, and robot charging station, and 'C' represents the corridor of the hospital. The robot will start navigating by selecting two known locations from the regions mentioned above.

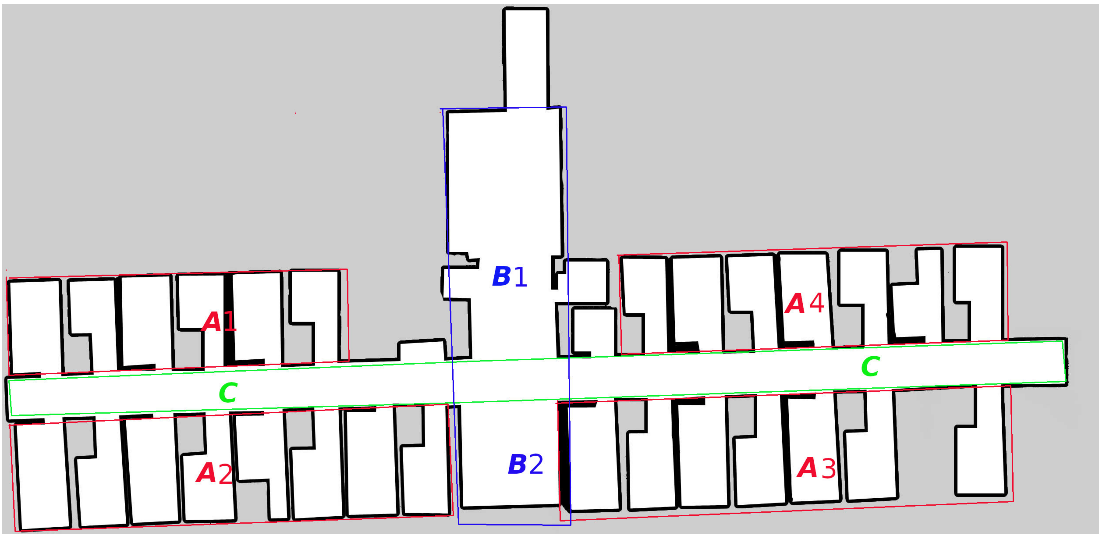


Figure 5.2: Dividing the map based on the separate work environment

For example, a start location from one of the rooms in the A2 region and the destination location in B1. Now, the robot has to safely exit the room and travel through the corridor and intersection to reach the destination in B1. While doing so, the robot goes through at least two of the potentially awkward social scenarios mentioned in the design section: navigating outside the non-workable area (F), staying on the right side of the corridor (C), or cutting down the speed while navigating near intersections (I), entering the room (E) and leaving the room (L). Figure 5.3 shows how these scenarios challenge the robot to navigate a healthcare facility. In each scenario, the robot is expected to show different behaviours based on the situation one is in.

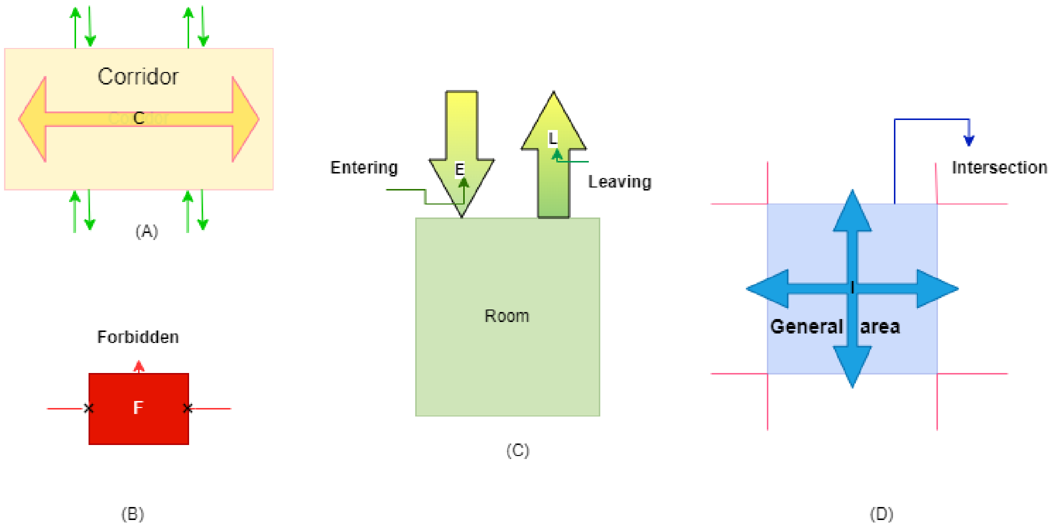


Figure 5.3: Building blocks for a map based on the separate work environment

Refer to Table 5.1 for all the scenarios. From all the scenarios in the table, the seven common scenarios are selected [L, C, E],[L, C, I, E], [L, C, I], [L, C, I, F], [I, C, E] [I, F], and [F, I, C, E]. It's worth noting that the default navigation behaviour of a social robot can also generate socially acceptable paths, but this is unpredictable because the default navigation algorithm does not consider social norms or conventions and will only prioritize speed and efficiency over social acceptability.

Start and end locations	**Scenarios**	**Start and end locations**	**Scenarios**
A1-A2	L, C, E	A2-A1	L, C, E
A1-A3	L, C, I, E	A3-A1	L, C, I, E
A1-A4	L, C, I, E	A4-A1	L, C, I, E
A1-B1	L, C, I, F	B1-A1	F, I , C, E
A1-B2	L, C, I	B2-A1	I, C, E
A2-A3	L, C, I, E	A3-A2	L, C, I, E
A2-A4	L, C, I, E	A4-A2	L, C, I, E
A2-B1	L, C, I, F	B1-A2	F, I, C, E
A2-B2	L, C, I	B2-A2	I, C, E
A3-A4	L, C, E	A4-A3	L, C, E
A3-B1	L, C, I,F	B1-A3	F, I, C, E
A3-B2	L, C, I	B2-A3	I, C, E
A4-B1	L, C, I, F	B1-A4	F, I, C, E
A4-B2	L, C, I	B2-A4	I, C , E
B1-B2	I, F	B2-B1	F, I
C - Narrow corridor, F - Forbidden area, I - Intersection, E- Entering room, L - Leaving room			

Table 5.1: Combinations of start and end locations and all the scenarios the robot has to go through to reach the destination.

5.1.1 Experiment Setup

The experiments were conducted on a laptop with the following specification

- **CPU** - Intel(R) Core(TM) i5-9300H CPU @ 2.40GHz 2.40 GHz
- **RAM** - 16.0 GB
- **GPU** - NVIDIA Geforce GTX 1650 4GB
- **OS** - Ubuntu 22.04.3 LTS (Jammy Jellyfish) / Windows 11 (Dual Boot)
- **ROS** - ROS2 Humble & Gazebo 11

5.1.2 Static and Dynamic Obstacles

To assess a robot's navigation ability, it's essential to simulate scenarios where it must navigate around people, including dynamic obstacles like moving individuals. Creating a diverse set of real-world scenarios can be achieved by combining static and dynamic obstacles shown in Figure 5.4 to evaluate the robot's ability to navigate complex environments. While static obstacles like furniture challenge the robot's ability to navigate fixed barriers, dynamic obstacles like moving people add a more complex challenge, requiring the robot to adapt randomly to changes in its environment.

Three particular conditions are tested with dynamic obstacles, which are Passing, Crossing, and Overtaking used to evaluate a robot's ability to navigate in an accepted social manner.

- **Overtaking** occurs when the robot needs to advance beyond a pedestrian walking in the same trajectory as the robot. In this circumstance, the robot must calculate the appropriate pace to pass the pedestrian without approaching too closely or causing the individual discomfort. The robot must also recognize any obstructions in the surroundings, such as

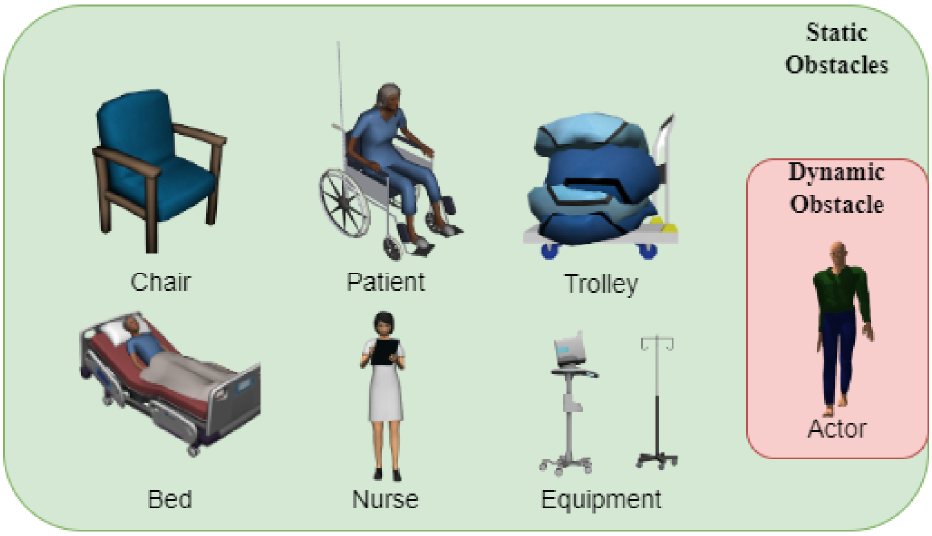


Figure 5.4: The set of obstacles used in the simulation consisting of both static and dynamic obstacles

other individuals or objects on the surface, and navigate around them without resulting in any disturbances.

- **Crossing** occurs when the robot must traverse a path where individuals or other obstacles are present. In this scenario, the robot must determine potential hazards, such as oncoming pedestrians, and wait until space clears before crossing. The robot must also be conscious of any other pedestrians crossing the exact path and modify its speed and trajectory to prevent collisions.
- **Passing** arises when the robot needs to pass a pedestrian walking in the opposite direction. In this event, the robot must establish a suitable distance and pace to surpass the pedestrian without inducing any discomfort or obstruction.

(a) Robot navigating in a crowded corridor with passing and overtaking

(b) Robot inside a patient room with static obstacles

Figure 5.5: (a) The robot navigating in a narrow corridor with a green shirt human model(dynamic model) and a nurse and patient in a wheelchair (static obstacles) (b) The robot navigating inside the room with static obstacles like bed and chair.

A few scenarios are recreated with dynamic and static obstacles to build a social navigation context for the robot. For example, the following scenarios involve a robot's navigation in a care facility. The first situation entails overtaking a resident walking slowly down a crowded hallway. The robot must determine the necessary distance and speed to overtake the resident without causing any discomfort.

The second scenario involves passing by a group of residents socializing in a narrow corridor. The robot must find a suitable pace and distance to pass by the group without impeding their movement or invading their personal space. Lastly, the third case involves crossing a path where a resident uses a walker. The robot must identify potential hazards, such as oncoming residents or staff members, and wait for a clear path before safely crossing to its intended destination.

To obtain a comprehensive understanding of the robot's navigation capabilities, a set of 7 common scenarios mentioned above with standard planning and improved planning are recorded to assess the robot's performance. For each scenario, three trials are conducted with a random set of dynamic and static obstacles in the simulation. A total of 42 unique scenarios were created at the end, with metrics and videos. Understanding the scenarios where standard planning fails can highlight areas for improvement in the robot's navigation capabilities through challenging environments.

5.1.3 Objective metrics

The following are the metrics used for evaluation:

- **Path length**: This measurement evaluates the distance the robot covers to reach its destination. The shorter path indicates that the robot navigates efficiently and avoids obstacles effectively, compared to a longer path, which indicates a less efficient path and obstacle avoidance. But in the case of social navigation, the shorter path does not necessarily have to be socially appropriate. The path length is computed as the Euclidean distance between the positions of the base frame(Robot's current location) in consecutive changes to get the distance travelled by the robot at each time step. Measuring the distance values over time to get the total path length travelled by the robot.
- **Time and Speed to Destination**: This measurement evaluates the robot's time to reach its target. A faster time to goal suggests that the robot is navigating more efficiently. However, it is essential to ensure that the robot's speed is socially acceptable and that it is not navigating at a higher speed that could be dangerous or threatening to the humans around it. When measuring the time it takes for the robot to complete a task, the clock starts at the initial pose and stops at the goal pose.
- **Number of backup/recovery times**: This evaluates how many times there was an error in navigating that required stopping, reversing, waiting, or recovering fewer recoveries. A higher number of backup or recovery times indicates that the robot struggles to navigate efficiently in the environment and around the obstacles. The number of backup/recovery times is calculated by counting the instances when the robot had to stop, reverse or wait during navigation and needed human assistance to get the robot out of being stuck.
- **Proxemics count**: This metric evaluation for testing social navigation can be beneficial in ensuring a robot's ability to approach and interact with humans without violating their personal space. Measuring the distance between the robot and the human reveals whether the robot respects personal space while navigating around. This metric indicates the comfort or respect people feel when around the robot. The proxemics metric is calculated by measuring the distance between the robot and the nearest human during navigation at regular intervals. When the distance is less than 0.5 meters between the human and the robot, the interaction can be defined as "close interaction". The number of close interactions evaluates the ability of the robot to navigate around humans safely. Depending on the robot-human distance, a count is incremented according to proxemic space every cycle it receives the message.

- **Social Comfort score** : This metric evaluates the ease people feel near the robot during its navigation. It can be assessed using feedback from the people regarding interactions with the robot and their perceptions of the robot's behaviour.

5.1.4 Subjective Metrics

Social comfort is an essential indicator of a robot's ability to navigate in a socially acceptable and safe manner. It measures the level of comfort and safety experienced by people around the robot during its navigation. This set of metrics is evaluated using the videos of robots' navigation following both default and improved navigation. A 5-point Likert scale survey gathers subjective data to evaluate the social comfort metric for a robot's navigation behaviour.

The participants are asked three questions to rate how

- ***safe***,
- ***socially appropriate***,
- and ***human-robot interaction***

they felt the robot's behaviour on a scale of -2 to +2, where -2 indicates "very unsafe /unpredictable/ inappropriate" and +2 indicates "very safe/predictable/ appropriate". The average score of the survey responses can provide a quantitative measure of the perceived social comfort of the robot's navigation behaviour. Utilizing this set of metrics in a survey containing questions designed to obtain answers related to safety and social acceptability, we can gather valuable insights into the robot's performance (Vega et al., 2019).

The survey was conducted either online or offline for the sake of participants. 6 participants answered the survey questions by going through all 42 scenarios. Every participant was asked to answer a question about familiarity with care robots. Two participants had expertise at a technical level, three participants had a typical understanding towards care robots, and one responded that they were new to the study. Every participant went through the random order of 14 unique scenarios containing seven from each improved and default method.

Analyzing the resulting data can help determine to which extent the robot's behaviour is socially acceptable. Based on the results, conclusions and recommendations are made to enhance the robot's navigation capabilities. Overall, incorporating static and dynamic obstacles and utilizing metrics and videos in a survey can help create a comprehensive understanding of a robot's navigation capabilities in social environments.

5.2 Hypotheses

This study uses mixed metrics to evaluate the effectiveness of an improved planner for care robots. These metrics include path length, Time to destination, Average speed, number of backups, proxemics, and social comfort.

The following hypotheses have been developed based on the research objectives:

1. Path length: The improved planner is expected to produce longer paths, up to 30% more than the default planner, due to its prioritization of safety over efficiency.
2. Time to destination: The improved planner is expected to take longer to reach the destination than the default planner, as it may adopt a more cautious approach of reduced speed to prevent potential collisions and follow social norms.
3. Number of recoveries: The improved planner is expected to require fewer backups or interventions from a human operator to complete the task, indicating improved autonomy and reliability.
4. Proxemics: The improved planner is expected to show more respect to human interaction space than the default planner, as it may prioritize safety and trust over efficiency.

5. Social comfort: The improved planner is expected to be safer, more socially appropriate, and more predictable than the default planner, leading to improved comfort and trust in the autonomous system.

6 Results

The following can be analysed from the hypotheses mentioned in Section 5.2 and comparing the results using objective and subjective metrics.

6.1 Objective metrics

6.1.1 Path Length

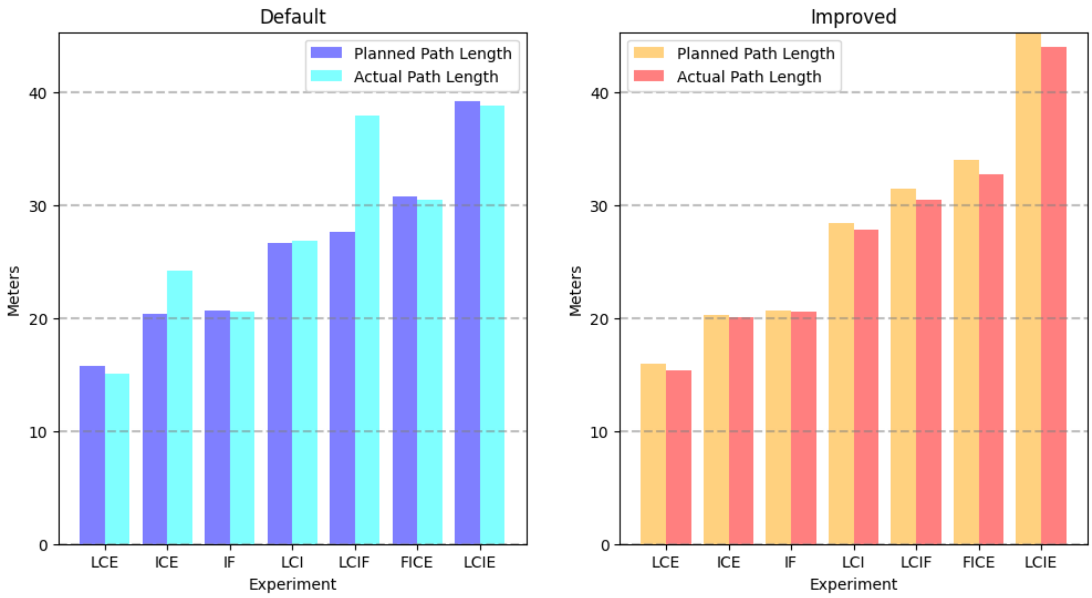


Figure 6.1: Mean Planner path length and Distance Travelled (in meters) across all the default and improved planner experiments.

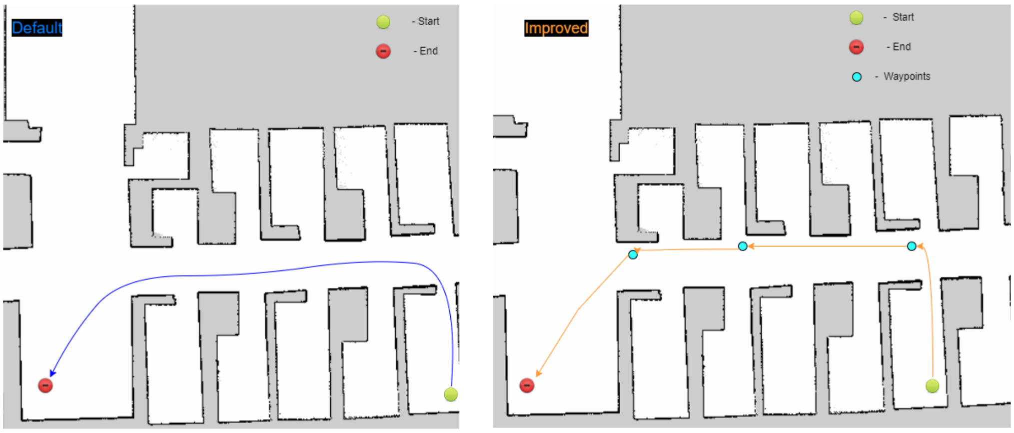
The Figure 6.1 presents a comprehensive plot illustrating the mean planned path length and actual path length (distance travelled) travelled by the default and improved navigation planner across various scenarios with different start and end locations. This graphical representation effectively demonstrates the impact of the improved method on the robot's navigation results.

Upon analyzing the graph, it becomes evident that the robot utilizing the improved method consistently follows the planned path across all scenarios. However, in the default method, the robot deviates from the initial path planned in two specific conditions, namely *ICE* and *LCIF*. The robot travels more in these scenarios than the initially intended path.

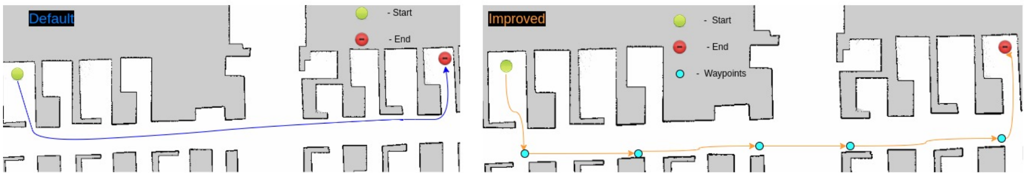
Interestingly, the default and improved planner exhibit similar characteristics in the IF scenario, where the robot successfully adheres to the planned path without significant deviations. However, the improved path length is slightly longer than the default method in scenarios *LCE*, *LCIF*, *FICE*, and *LCIE*.

According to the hypothesis, the improved planner is expected to travel more than the default planner by 30% in Figure 6.2. However, from the results, it was evident that this is not true entirely. By comparing the path length generated by both the planner in 3 experiments(scenarios), the default and improved path planner shows that the robot generates a planned path that is the same length in experiments *LCE*, ICE, and *IF*; however, in scenarios *LCIF* and *ICE* the default planner's planned path length differs from the actual path length travelled by the robot.

(a) Leaving - Corridor - Entering (LCE) scenario



(b) Leaving - Corridor - Intersection (LCI) scenario

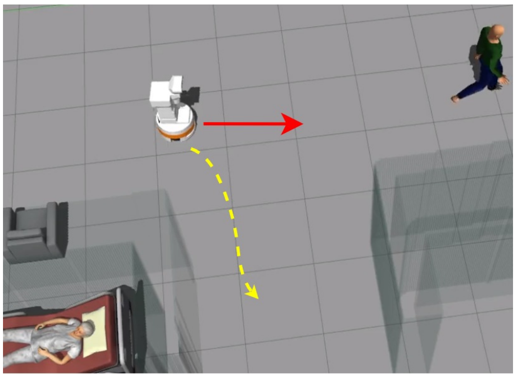


(c) Leaving- Corridor - Intersection - Entering (LCIE) scenario

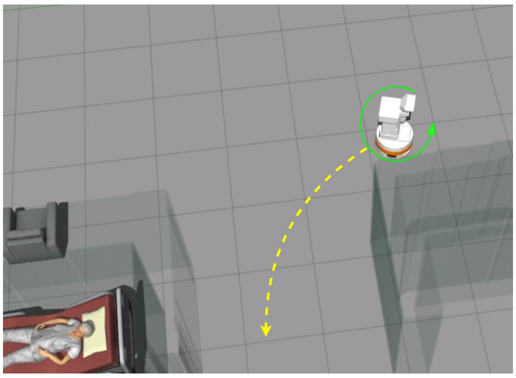
Figure 6.2: Distance comparison of the robot's travelled path from start to end location in three scenarios: LCE, LCI and LCIE. The image describes the path the robot took to reach the end goal. The difference in the distance between the start and end location changes the pattern of the travelled path

The distance between the start and end location increases in experiments *LCIF*, *FICE*, and *LCE*. When covering longer distances, the improved planner generates a path that is about 20% longer than the default planner on average. However, the improved planner ensures that the robot follows the path consistently, while the default planner often results in unpredictable robot behaviour.

In both cases, the robot went through a localization issue. The localization issue of the robot is one of the causes that would make its behaviour unpredictable, as seen in Figure 6.3. In that particular situation, the robot suffers from difficulty localizing itself in the environment when travelling at a higher speed and making a turn or entering a narrow to a broad region or vice versa.



(a) Robot must take a right turn and enter the corridor. But goes forward and misses the turn.



(b) Robot suffers localization issues where it goes further and recovers to correct itself

Figure 6.3: Robot suffers localization issue and causes to travel the extra distance to reach the goal.

6.1.2 Speed and Time

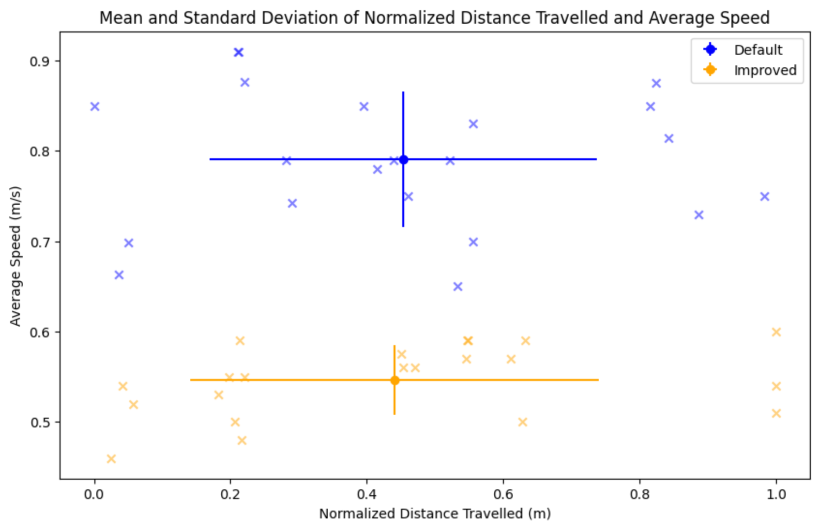


Figure 6.4: Normalized Distance Travelled vs Average Speed between Default and Improved navigation planner. The improved planner travels at a significantly lesser speed while covering almost the same distance

The Figure 6.4 compares the default, and improved methods regarding the normalized distance travelled based on average speed in all the experiments. The plot showcases the mean and standard deviation, represented by the cross, for both approaches. It shows the default planner is faster than the improved planner.

Additionally, the improved planner exhibits a slightly shorter distance travelled than the default approach when considering the overall performance. This indicates that the improved planner not only resulted in slower robot navigation but also led to a marginally reduced path length.

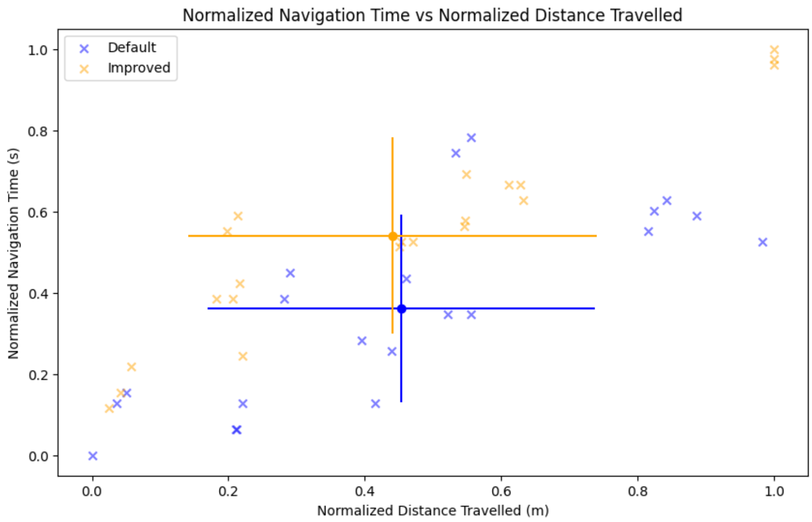


Figure 6.5: Normalized Distance Travelled vs Normalized Time between Default and Improved navigation method. The improved planner takes more Time to reach the destination.

The graph in Figure 6.5 illustrates the relationship between the normalised distance travelled and the normalized navigation time of the robot, similar to the previous plot. The plot shows that the improved planner significantly impacted the robot's navigation results. The robot, under the improved planner, travelled for a considerably longer time as compared to the default planner.

The second hypothesis is concerning the navigation time of the robot. From Figure 6.5, by the graph, it is evident that the improved planner takes more Time to reach the destination. The improved planner takes approximately 20% more time than the default planner. As expected, the improved planner slows down at multiple sections in the map and causes the robot to navigate at a lower speed than the default planner.

The graph in Figure 6.4 shows that the mean improved planner speed is approximately 25% slower than the improved planner. The default planner's average speed spread is higher than the improved planner because the robot reduces speed only when there is an obstacle over the path following or a narrow space with less room to travel. In the improved planner, the robot slows down when humans are present in its surroundings and navigating across the intersections.

6.1.3 Recoveries

During the robot's navigation from one point to another, it encounters static and dynamic obstacles in its environment. These obstacles posed a challenge for the robot, requiring it to make multiple corrections through the default recovery cycle. This recovery cycle aims to ensure the robot can rectify any issues it encounters while following its designated path or updating its knowledge of the surroundings.

Interestingly, the bar graph depicted in Figure 6.6 visually represents the total number of recoveries across all the experiments. From this chart, we can observe that the improved planner

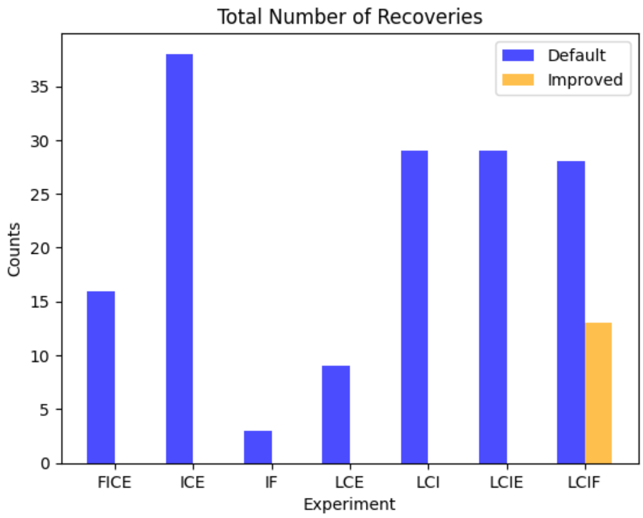
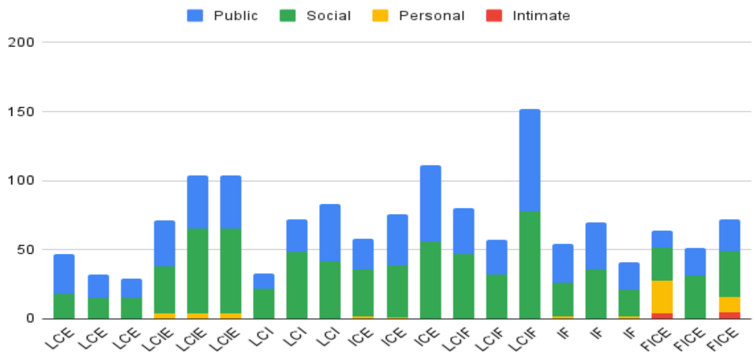


Figure 6.6: Bar chart representing the total number of recoveries recorded during the experiment. The improved planner shows a significantly lower number of recoveries.

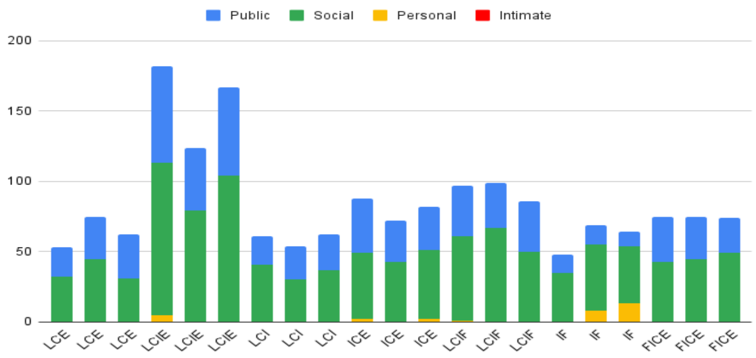
significantly reduces the number of recoveries required. This improvement in navigation results highlights the enhanced safety and efficiency of the robot's operations.

The number of recoveries of the robot gives an overall picture of the robot's behaviour. The Figure 6.6 shows the division of the total number of recoveries by both planners over all the experiments. The improved planner shows a significantly lower number of recoveries, and the hypothesis on the number of recoveries is true. While most of the recoveries were observed due to localization problems faced by the default planner, the improved planner also suffered from localization issues. Still, due to its slow speed and longer navigation time, the improved planner recovered well compared to the default planner. This was the critical difference observed while observing the navigation behaviour of the robot navigating with varied speeds.

6.1.4 Proxemics count



(a) Default planner Proxemic counts



(b) Improved planner Proxemic counts

Figure 6.7: Proxemic counts for each trial of the scenario in a stacked bar chart. The proxemic spaces are labelled according to the above colour scheme.Default planner's proxemic counts show that the improved planner has a lower number of intrusions in the personal space and none in the intimate space

Figure 6.7a and Figure 6.7b provide a comprehensive analysis of the proxemic counts, which offer valuable insights into the robot's proximity to humans throughout the experiment.

Upon careful examination, it is evident that both graphs exhibit significant similarities in terms of the observed proxemic counts. However, two noteworthy exceptions deserve attention. In the FICE scenario, where the default planner was utilized, the robot intruded into intimate space distance around the human.

Conversely, a slight increase in personal space counts can be observed in the IF scenario where the improved planner was implemented. This suggests that while displaying improved planning capabilities, the robot may have, at times, encroached upon personal space by moving slowly.

Further analyzing two proxemic zones, Intimate and Personal, is essential in understanding robots' interaction with humans. The Figure 6.8 is a graph representing normalized counts to normalized distance travelled. Which gives a scale of count per distance travelled(count/m). The default planner does enter the intimate space twice in a situation, and the improved planner avoids intimate space. But In personal space, the improved planner has a higher count than the default planner.

In Figure 6.9a, the default planner gets stuck behind a nurse in FICE. In this particular situation, the robot gets stuck in the intimate space of the nurse. Also, in both the planners, the obstacle inflation radius was set to 0.45m. This inflation radius sets high costs around all the obstacles in the environment, which causes the robot to avoid them. But in the mentioned scenario of

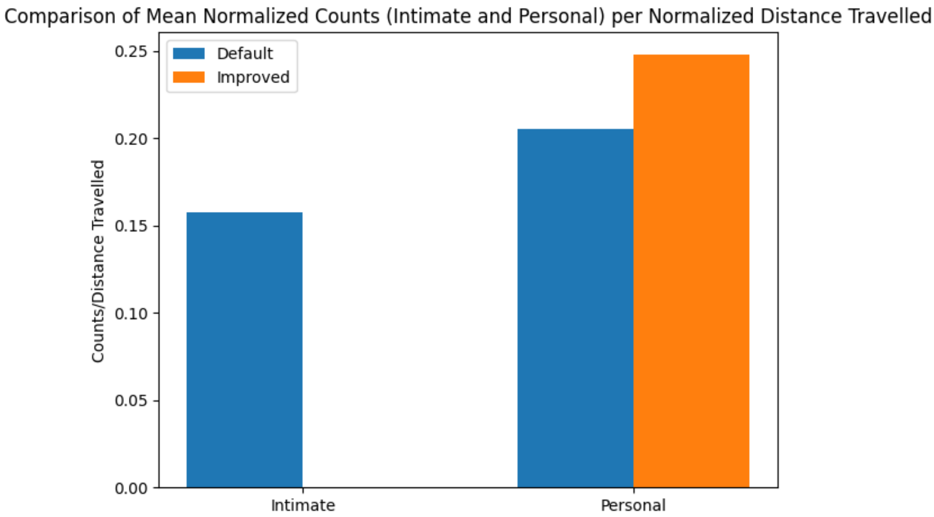


Figure 6.8: Comparision of Normalised Proxemic counts over normalised distance travelled across all the experiments between default and improved, shows that the default planner makes the robot enter intimate space around the human

the default planner, while navigating at a higher speed, it enters the inflation layer in a narrow space behind the nurse.

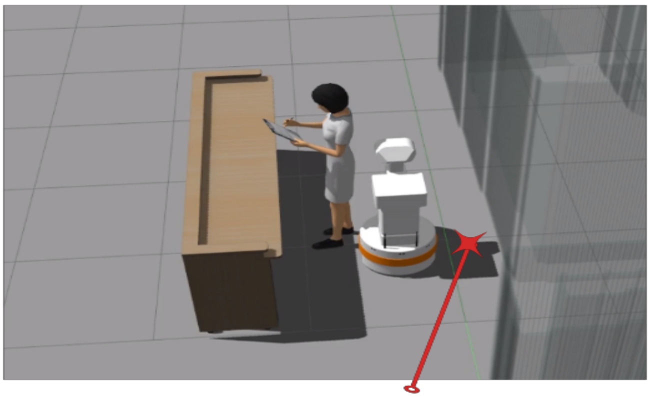
The robot entering intimate spaces is not easy because of the inflation layer around the objects, which influences the robot to avoid it. Still, in the above situation, it got stuck behind the nurse, took a few recoveries and reached the destination. The improved planner had a high proxemic count for personal space for one particular scenario IF.

Figure 6.9b shows the improved planner moving slowly in front of the person in the wheelchair. The main difference between the two planners is the speed around humans. The improved planner is expected to stay around humans for a long time due to the reduction of speed based on the proxemic distance.

However, when navigating around dynamic humans on the map, the robot slowed down and managed the human interaction well, as it allowed humans to move ahead of the slow robot. The robot took more Time to pass around them, cautious of static humans in the environment. Compared with the default planner, which does not differentiate between humans and other obstacles in the environment, it maintains the same speed throughout the navigation.

From the hypothesis mentioned, the improved planner does better in respecting the intimate space; at the same time, it spends more time around static humans, which is helpful in a few cases to avoid unpredictable movements that happen at higher speeds.

(a) Improved planner slowly going in front of the person in the wheelchair does not intrude on intimate space but takes longer time to pass in front of the wheelchair



(b) Default planner taking a shorter route.Leading to the unacceptable intrusion of intimate space

Figure 6.9: The above images show the scenarios where both the planners scored high proxemic score

6.2 Subjective Metric

Figure 6.10 represents the plot of the social comfort score based on the survey answers. The mean score for each survey question respective to the scenario is calculated. The improved method is consistent in the mean positive scale over all the questions, and the default method has the mean values in the mean negative scale.

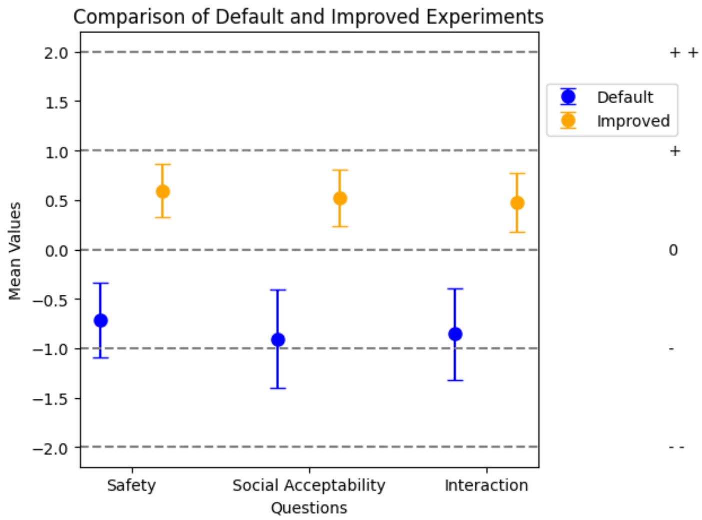


Figure 6.10: Comparision of mean social comfort score between Default and Improved planner.

The hypothesis stays true for social comfort as seen in the survey result graph Figure 6.10. The improved planner performed better. People felt the improved planner was safer and showed socially acceptable robot behaviour compared to the default planner. Some scenarios in the improved planner also got a negative score for the survey questions. For example, in Figure 6.9a, this particular situation was slowing down the robot in front of the person in the wheelchair while moving close. A few people answering the survey did not like this, as they felt the robot should be more aware of a person sitting in a wheelchair than the person standing or walking.

7 Conclusions and Recommendations

7.1 Conclusions

The main objective of this study was to investigate how robot navigation can be made more socially acceptable in care environments. The study achieved this by analyzing human-robot interaction and default navigation behaviour. The primary research question was: *"How to make a care robot's navigation more socially acceptable by following social norms in a care environment?"*

Through literature investigation, two primary social norms were identified, leading to the development of new and improved navigation designs for care robots. At first glance, these social norms may seem quite obvious: keeping to the right and respecting personal space.

However, further analysis revealed that implementing and understanding these norms is not always straightforward. An additional social norm identified during analysis was the need to avoid frightening or scaring people around intersections and corners.

To implement these social norms into a care robot, various strategies were employed, including the use of keepout filters to avoid forbidden zones, following waypoints to stay on the right side of narrow spaces, applying speed filters to be cautious at intersections and corners, and improving human-robot interaction by respecting proxemic space.

The result was a new, socially aware navigation system that outperformed the default navigation behaviour in terms of both subjective and objective measures. In objective metrics, the socially aware navigation system showed a significant reduction in the number of recoveries needed, although travel speed was slower, and more time and distance were required to navigate cautiously. However, the time required to reach the destination using the default navigation system was almost as long due to the time needed to recover from awkward situations.

Subjective measures from surveys showed that the improved navigation system resulted in safer and less socially inappropriate behaviour. Although the default navigation system did produce some socially acceptable scenarios, the improved navigation design outperformed it in most situations.

The study presented in this report has a few limitations: The social norms identified and implemented were not exhaustive and may not be universally applicable across different cultures or environments. The evaluation was conducted in a controlled environment, which might not entirely reflect the complexities and unpredictability of real-world settings. Future research should explore other relevant social norms. Furthermore, it is recommended to test the socially aware navigation system proposed in this thesis in diverse real-world environments.

7.2 Recommendations

The following further improvements can be made to improve the social acceptability of the robot

- **Enhance Personal Space Awareness:** Currently, the robot does not create extra space around the humans apart from the inflation layer radius. Creating high costs around the detected human in the environment will make the robot respect the personal space of humans better. Predicting the future state of these models would help the robot to navigate better so that the robot can avoid being too close to humans while navigating.
- **Consider Human Actions**: Creating high costs around humans based on the action they are performing, such as standing, walking, sitting and lying down, can also make the robot understand the human interaction space better.

- **Incorporate Verbal Interaction**: Other social factors like verbal interaction with humans will improve the social acceptability of the robot. When navigating in the facility, using verbal commands by the robot would help the people understand what it is doing.
- **Real-Life Testing**: Testing improved methods on the robot in real life is recommended to understand the actual robot performance in real life around humans.
- **Utilise Learning Algorithms**: Further development can use learning algorithms to teach social norms to the robot to follow in a closed environment. Learning how robots should behave in a care facility while navigating the corridor, intersection, and workable environment will make the robot more acceptable.

A Appendix

A.1 Inflation of costmap cells

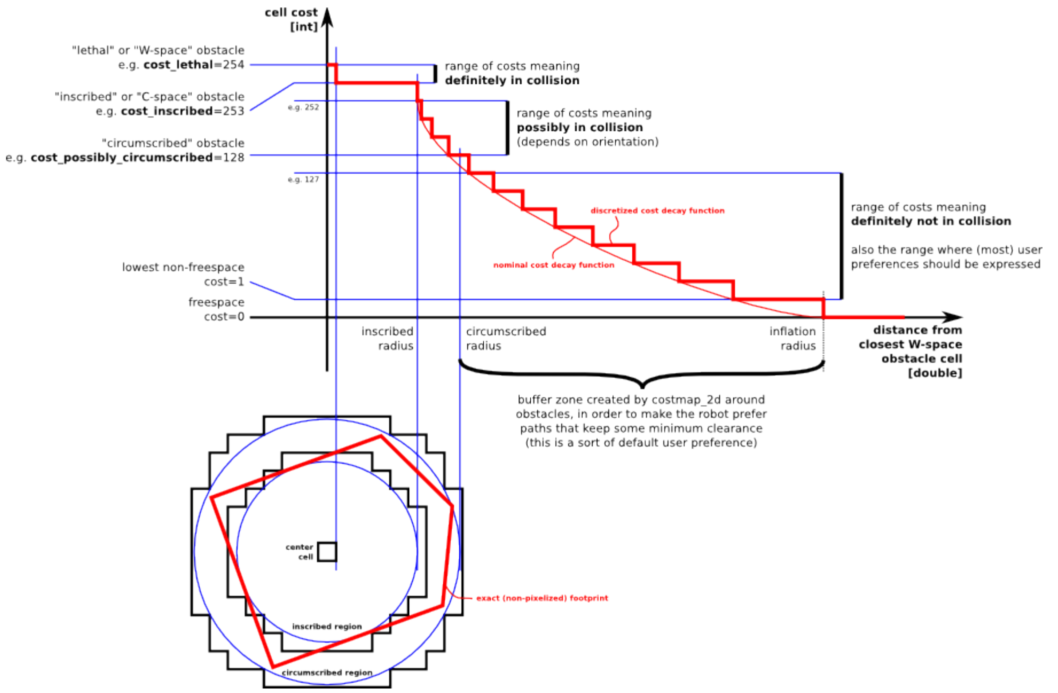


Figure A.1: Assignment of Inflation Layer costs

Inflation is a process that redistributes cost values from occupied cells to nearby cells. As the distance from the occupied cell increases, the cost values decrease. For this purpose, five specific symbols are defined for costmap values related to a robot.

The "Lethal" cost symbol signifies an actual obstacle present in a cell. If the robot's center is in that cell, it will collide with the obstacle.

The "Inscribed" cost symbol denotes that a cell is less than the robot's inscribed radius away from an obstacle. If the robot's center is in a cell that is at or above the inscribed cost, it will collide with an obstacle.

The "Possibly circumscribed" cost symbol is similar to "Inscribed." However, it uses the robot's circumscribed radius as a cutoff distance. If the robot's center lies in a cell at or above this value, it depends on the robot's orientation whether it will collide with an obstacle or not. The term "possibly" is used because it may not be an obstacle cell but some user preference that puts that particular cost value into the map. For example, if a user wants to express that a robot should avoid a particular area of a building, they may inset their costs into the costmap for that region independent of any obstacles.

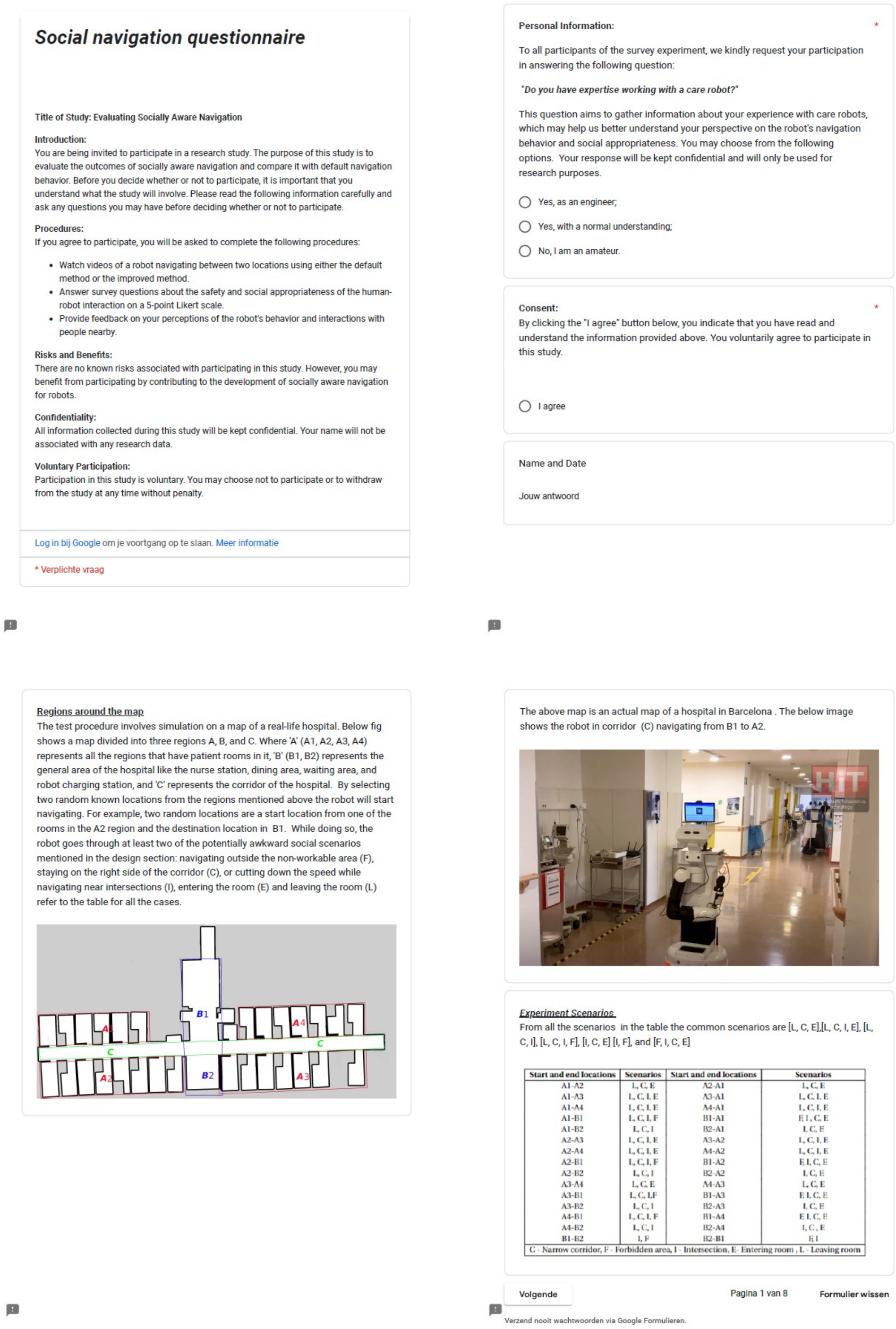
The "Freespace" cost symbol assumes zero cost, meaning that there is no obstacle preventing the robot from going there.

The "Unknown" cost symbol signifies that there is no information about a given cell. The user of the costmap can interpret this as they see fit.

All other costs are assigned a value between "Freespace" and "Possibly circumscribed" depending on their distance from a "Lethal" cell and the decay function provided by the user. The rationale behind these definitions is to enable planner implementations to decide whether or not they care about the exact footprint. We provide enough information so that they can incur the cost of tracing out the footprint only when it is necessary.

A.2 Survey

The below images show the online survey questionnaire form.



Social navigation questionnaire

Title of Study: Evaluating Socially Aware Navigation

Introduction:
You are being invited to participate in a research study. The purpose of this study is to evaluate the outcomes of socially aware navigation and compare it with default navigation behavior. Before you decide whether or not to participate, it is important that you understand what the study will involve. Please read the following information carefully and ask any questions you may have before deciding whether or not to participate.

Procedures:
If you agree to participate, you will be asked to complete the following procedures:

- Watch videos of a robot navigating between two locations using either the default method or the improved method.
- Answer survey questions about the safety and social appropriateness of the human-robot interaction on a 5-point Likert scale.
- Provide feedback on your perceptions of the robot's behavior and interactions with people nearby.

Risks and Benefits:
There are no known risks associated with participating in this study. However, you may benefit from participating by contributing to the development of socially aware navigation for robots.

Confidentiality:
All information collected during this study will be kept confidential. Your name will not be associated with any research data.

Voluntary Participation:
Participation in this study is voluntary. You may choose not to participate or to withdraw from the study at any time without penalty.

Log in bij Google om je voortgang op te slaan. Meer informatie

* Verplichte vraag

Personal Information: *

To all participants of the survey experiment, we kindly request your participation in answering the following question:

"Do you have expertise working with a care robot?"

This question aims to gather information about your experience with care robots, which may help us better understand your perspective on the robot's navigation behavior and social appropriateness. You may choose from the following options. Your response will be kept confidential and will only be used for research purposes.

- Yes, as an engineer;
- Yes, with a normal understanding;
- No, I am an amateur.

Consent: *
By clicking the "I agree" button below, you indicate that you have read and understand the information provided above. You voluntarily agree to participate in this study.

- I agree

Name and Date

Jouw antwoord

Regions around the map

The test procedure involves simulation on a map of a real-life hospital. Below fig shows a map divided into three regions A, B, and C. Where 'A' (A1, A2, A3, A4) represents all the regions that have patient rooms in it, 'B' (B1, B2) represents the general area of the hospital like the nurse station, dining area, waiting area, and robot charging station, and 'C' represents the corridor of the hospital. By selecting two random known locations from the regions mentioned above the robot will start navigating. For example, two random locations are a start location from one of the rooms in the A2 region and the destination location in B1. While doing so, the robot goes through at least two of the potentially awkward social scenarios mentioned in the design section: navigating outside the non-workable area (F), staying on the right side of the corridor (C), or cutting down the speed while navigating near intersections (I), entering the room (E) and leaving the room (L) refer to the table for all the cases.

The above map is an actual map of a hospital in Barcelona . The below image shows the robot in corridor (C) navigating from B1 to A2.

Experiment Scenarios
From all the scenarios in the table the common scenarios are [L, C, E],[L, C, I, E], [L, C, I], [L, C, I, F], [I, C, E] [I, F], and [F, I, C, E]

Start and end locations	Scenarios	Start and end locations	Scenarios
A1-A2	L, C, E	A2-A1	L, C, E
A1-A3	L, C, I, E	A3-A1	L, C, I, E
A1-A4	L, C, I, E	A4-A1	L, C, I, E
A1-B1	L, C, I, F	B1-A1	F, I, C, E
A1-B2	L, C, I	B2-A1	I, C, E
A2-A3	L, C, I, E	A3-A2	L, C, I, E
A2-A4	L, C, I, E	A4-A2	L, C, I, E
A2-B1	L, C, I, F	B1-A2	F, I, C, E
A2-B2	L, C, I	B2-A2	I, C, E
A3-A4	L, C, E	A4-A3	L, C, E
A3-B1	L, C, I,F	B1-A3	F, I, C, E
A3-B2	L, C, I	B2-A3	I, C, E
A4-B1	L, C, I, F	B1-A4	F, I, C, E
A4-B2	L, C, I	B2-A4	I, C , E
B1-B2	I, F	B2-B1	F, I
C - Narrow corridor, F - Forbidden area, I - Intersection, E- Entering room , L - Leaving room			

Volgende　　Pagina 1 van 8　　Formulier wissen

Verzend nooit wachtwoorden via Google Formulieren.

Figure A.2

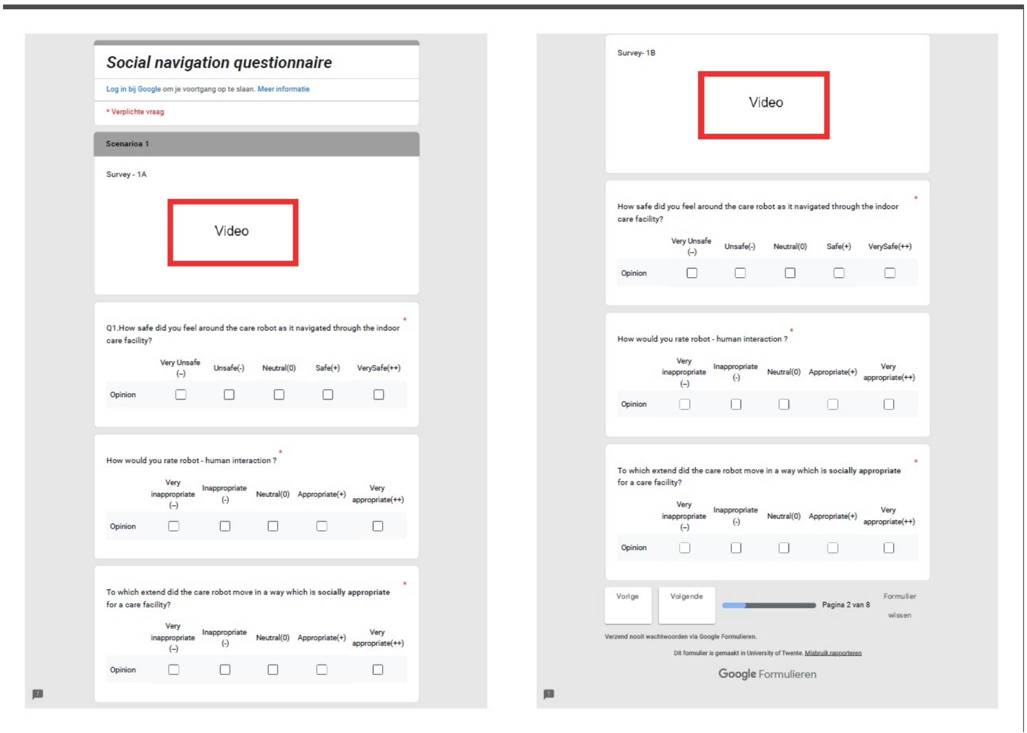


Figure A.3

A.3 ROS2 parameter settings

The below yaml file gives the settings for each server

A.3.1 Controller Server

```
controller_server:
  ros__parameters:
    use_sim_time: ${use_sim_time}
    controller_frequency: 20.0
    min_x_velocity_threshold: 0.001
    min_y_velocity_threshold: 0.5
    min_theta_velocity_threshold: 0.001
    failure_tolerance: 0.3
    progress_checker_plugin: "progress_checker"
    goal_checker_plugins: ["general_goal_checker"]
    controller_plugins: ["FollowPath"]
    speed_limit_topic: "/speed_limit"

    # Progress checker parameters
    progress_checker:
      plugin: "nav2_controller::SimpleProgressChecker"
      required_movement_radius: 0.5
      movement_time_allowance: 10.0
    # Goal checker parameters
    general_goal_checker:
      plugin: "nav2_controller::SimpleGoalChecker"
      xy_goal_tolerance: 0.25
      yaw_goal_tolerance: 0.25
      stateful: True
    # DWB parameters
    FollowPath:
      plugin: "dwb_core::DWBLocalPlanner"
      debug_trajectory_details: True
      min_vel_x: 0.0
```

```
      min_vel_y: 0.0
      max_vel_x: 1.00
      max_vel_y: 0.0
      max_vel_theta: 1.00
      min_speed_xy: 0.0
      max_speed_xy: 1.00
      min_speed_theta: 0.0
      acc_lim_x: 2.5
      acc_lim_y: 0.0
      acc_lim_theta: 3.2
      decel_lim_x: -2.5
      decel_lim_y: 0.0
      decel_lim_theta: -3.2
      vx_samples: 20
      vy_samples: 5
      vtheta_samples: 20
      sim_time: 1.3 #1.7
      linear_granularity: 0.05
      angular_granularity: 0.025
      transform_tolerance: 0.2
      xy_goal_tolerance: 0.25
      trans_stopped_velocity: 0.25
      short_circuit_trajectory_evaluation: True
      stateful: True
      critics: ["RotateToGoal", "Oscillation", "BaseObstacle", "GoalAlign",
    "PathAlign", "PathDist", "GoalDist"]
      BaseObstacle.scale: 0.02
      PathAlign.scale: 32.0
      PathAlign.forward_point_distance: 0.1
      GoalAlign.scale: 24.0
      GoalAlign.forward_point_distance: 0.1
      PathDist.scale: 32.0
      GoalDist.scale: 24.0
      RotateToGoal.scale: 32.0
      RotateToGoal.slowing_factor: 5.0
      RotateToGoal.lookahead_time: -1.0
```

A.3.2 Global Costmap

```
global_costmap:
  global_costmap:
    ros__parameters:
      update_frequency: 2.0
      publish_frequency: 1.0
      global_frame: ${global_frame}
      robot_base_frame: ${base_link_frame}
      use_sim_time: ${use_sim_time}
      robot_radius: ${robot_radius}
      resolution: 0.05
      track_unknown_space: true
      plugins: ["static_layer", "obstacle_layer", "inflation_layer"]
      filters: ["keepout_filter","speed_filter"]
      speed_filter:
        plugin: "nav2_costmap_2d::SpeedFilter"
        enabled: true
        filter_info_topic: "/speed_filter_info"
        speed_limit_topic: "/speed_limit"
      keepout_filter:
        plugin: "nav2_costmap_2d::KeepoutFilter"
        enabled: true
        filter_info_topic: "/keepout_filter_info"
      obstacle_layer:
        plugin: "nav2_costmap_2d::ObstacleLayer"
```

```
          enabled: true
          observation_sources: scan
          scan:
            topic: ${scan_topic}
            max_obstacle_height: 2.0
            clearing: True
            marking: True
            data_type: "LaserScan"
            raytrace_max_range: 8.0
            raytrace_min_range: 0.0
            obstacle_max_range: 5.0
            obstacle_min_range: 0.0
        static_layer:
          plugin: "nav2_costmap_2d::StaticLayer"
          map_subscribe_transient_local: True
        inflation_layer:
          plugin: "nav2_costmap_2d::InflationLayer"
          cost_scaling_factor: 2.0
          inflation_radius: 0.45
        always_send_full_costmap: True
```

A.3.3 Local Costmap

```
local_costmap:
  local_costmap:
    ros__parameters:
      update_frequency: 5.0
      publish_frequency: 2.0
      global_frame: ${odom_frame}
      robot_base_frame: ${base_link_frame}
      use_sim_time: ${use_sim_time}
      rolling_window: true
      width: 4
      height: 4
      resolution: 0.05
      robot_radius: ${robot_radius}
      plugins: ["voxel_layer", "inflation_layer"]
      filters: ["keepout_filter"]
      keepout_filter:
        plugin: "nav2_costmap_2d::KeepoutFilter"
        enabled: true
        filter_info_topic: "/keepout_filter_info"
      inflation_layer:
        plugin: "nav2_costmap_2d::InflationLayer"
        cost_scaling_factor: 2.0
        inflation_radius: 0.45
      voxel_layer:
        plugin: "nav2_costmap_2d::VoxelLayer"
        enabled: True
        publish_voxel_map: True
        origin_z: 0.0
        z_resolution: 0.05
        z_voxels: 16
        max_obstacle_height: 2.0
        mark_threshold: 0
        observation_sources: scan
        scan:
          topic: ${scan_topic}
          max_obstacle_height: 2.0
          clearing: True
          marking: True
          data_type: "LaserScan"
          raytrace_max_range: 5.0
```

```
        raytrace_min_range: 0.0
        obstacle_max_range: 4.0
        obstacle_min_range: 0.0
    static_layer:
      plugin: "nav2_costmap_2d::StaticLayer"
      map_subscribe_transient_local: True
    always_send_full_costmap: True
```

A.3.4 Planner server

```
planner_server:
  ros__parameters:
    expected_planner_frequency: 20.0
    use_sim_time: ${use_sim_time}
    planner_plugins: ["GridBased"]
    GridBased:
      plugin: "nav2_navfn_planner/NavfnPlanner"
      tolerance: 0.5 #0.5
      use_astar: false
      allow_unknown: true
```

Bibliography

(2023), Heemskerk Innovative Technology (HIT).
https://www.youtube.com/watch?v=ro0th41b7ZY

Barchard, K. A., L. Lapping-Carr, R. S. Westfall, A. Fink-Armold, S. B. Banisetty and D. Feil-Seifer (2020), Measuring the perceived social intelligence of robots, *ACM Transactions on Human-Robot Interaction*, **vol. 9**, ISSN 25739522, doi:10.1145/3415139.

Kennisplein Zorg voor Beter, z. (2022), Cijfers.
https://www.zorgvoorbeter.nl/veranderingen-langdurige-zorg/cijfers-vergrijzing

Che, Y., A. M. Okamura and D. Sadigh (2020), Efficient and Trustworthy Social Navigation via Explicit and Implicit Robot-Human Communication, *IEEE Transactions on Robotics*, **vol. 36**, pp. 692–707, ISSN 19410468, doi:10.1109/TRO.2020.2964824.

Chik, S. F., C. F. Yeong, E. L. M. Su, T. Y. Lim, Y. Subramaniam and P. J. H. Chin (2016), A Review of Social-Aware Navigation Frameworks for Service Robot in Dynamic Human Environments, *Journal of Telecommunication, Electronic and Computer Engineering*, **vol. 8**, pp. 41–50.

DutchNews (2022), Nursing homes call on relatives to help take care of frail elderly.
https://www.dutchnews.nl/news/2022/07/nursing-homes-call-on-relatives-to-help-take-care-of-frail-elderly/

Dzedzickis, A., J. Subaciute-Zemaitiene, E. Šutinys, U. Samukaite-Bubniene and V. Bučinskas (2021), Advanced Applications of Industrial Robotics: New Trends and Possibilities, *Applied Sciences*, doi:10.3390/app12010135.

Gao, Y. and C. M. Huang (2022), Evaluation of Socially-Aware Robot Navigation, *Frontiers in Robotics and AI*, **vol. 8**, ISSN 22969144, doi:10.3389/frobt.2021.721317.

Hall, E. T., R. L. Birdwhistell, B. Bock, P. Bohannan, A. R. Diebold Jr, M. Durbin, M. S. Edmonson, J. Fischer, D. Hymes, S. T. Kimball et al. (1968), Proxemics, **vol. 9**, no.2/3, pp. 83–108.

HIT (2021).
https://heemskerk-innovative.nl/

HIT (2022), Projectsandro.
https://www.youtube.com/watch?v=cA8mtkqo-zw&t=7s&ab_channel=HeemskerkInnovativeTechnology%28HIT%29

Honour, A., S. B. Banisetty and D. Feil-Seifer (2021), Perceived social intelligence as evaluation of socially navigation, IEEE Computer Society, pp. 519–523, ISBN 9781450382908, ISSN 21672148, doi:10.1145/3434074.3447226.

Jeong, S., T. Ga, I. Jeong, J. Oh and J. Choi (2022), Layered Cost-Map-Based Traffic Management for Multiple Automated Mobile Robots via a Data Distribution Service.
http://arxiv.org/abs/2207.08902

Jocher, G., A. Chaurasia, A. Stoken, J. Borovec, NanoCode012, Y. Kwon, K. Michael, TaoXie, J. Fang, imyhxy, Lorna, Yifu), C. Wong, A. V, D. Montes, Z. Wang, C. Fati, J. Nadar, Laughing, UnglvKitDe, V. Sonck, tkianai, yxNONG, P. Skalski, A. Hogan, D. Nair, M. Strobel and M. Jain (2022), ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation, doi:10.5281/zenodo.7347926.
https://doi.org/10.5281/zenodo.7347926

Karageorgos, D. (2017), Human-Aware Autonomous Navigation of a Care Robot in Domestic Environments.

Kuipers, B. (2018), How can we trust a robot?, *Communications of the ACM*, **vol. 61**, pp. 86–95, ISSN 15577317, doi:10.1145/3173087.

Lu, D. V., D. Hershberger and W. D. Smart (2014), Layered costmaps for context-sensitive navigation, in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 709–715, doi:10.1109/IROS.2014.6942636.

Macenski, S. (2020), WaypointFollower.
https://index.ros.org/p/nav2_waypoint_follower/

Macenski, S., T. Foote, B. Gerkey, C. Lalancette and W. Woodall (2022), Robot Operating System 2: Design, architecture, and uses in the wild, **vol. 7**, no.66, p. eabm6074, doi:10.1126/scirobotics.abm6074.
https://www.science.org/doi/abs/10.1126/scirobotics.abm6074

Macenski, S., F. Martin, R. White and J. G. Clavero (2020), The Marathon 2: A Navigation System, in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, doi:10.1109/iros45743.2020.9341207.
https://doi.org/10.1109%2Firos45743.2020.9341207

Mellema, R., M. Jensen and F. Dignum (2020), Social rules for agent systems.
http://arxiv.org/abs/2004.12797

Mutlu, B. and J. Forlizzi (2008), Robots in organizations: The role of workflow, social, and environmental factors in human-robot interaction, in *2008 3rd ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pp. 287–294, doi:10.1145/1349822.1349860.

NAV2 (2020), NavigateTopose.
https://navigation.ros.org/behavior_trees/trees/nav_to_pose_recovery.html

Nav2SpeedFilter (2020), Nav2SpeedFilter.
https://navigation.ros.org/tutorials/docs/navigation2_with_speed_filter.html

Pinker, S. (2012), *The Better Angels of Our Nature: Why Violence Has Declined*, Penguin Books.

Rios-Martinez, J., A. Spalanzani and C. Laugier (2014), From Proxemics Theory to Socially-Aware Navigation: A Survey, **vol. 7**, no.2, pp. 137–153, doi:10.1007/s12369-014-0251-1.

Vega, A., L. J. Manso, R. Cintas and P. Núñez (2019), Planning Human-Robot Interaction for Social Navigation in Crowded Environments, Springer Verlag, pp. 195–208, ISBN 9783319998848, ISSN 21945357, doi:10.1007/978-3-319-99885-5_14.

Ministerie van Volksgezondheid, W. e. S. (2022), Nieuw Programma Ouderenzorg: Meer Digitaal, Meer Thuis en Meer Eigen Regie.
https://www.rijksoverheid.nl/actueel/nieuws/2022/07/04/nieuw-programma-ouderenzorg-meer-digitaal-meer-thuis-en-meer-eigen-regie