

Perception and Learning for Unified Map of Service Robots in Human-Centered Environments

Department of Mechanical Systems Engineering

Graduate School of Systems Design

Tokyo Metropolitan University

20962607

Mohamad Yani

June 2023

Abstract

Aging populations have become a serious issue in many countries. Recently, service robots have been developed to help older adults and people with disabilities live independently. Service robots use various types of sensory information to understand environmental conditions and interact with people. While the RGB-D data are important for localizing the object and manipulation, a 2D occupancy map is often sufficient for navigation tasks. It is a fundamental problem for a service robot to achieve high accuracy and detailed representation in the human-centered environment during manipulation and navigation tasks in real-time control. Moreover, the navigation tasks are usually challenging to deal with indoor furniture such as chairs and tables if the service robot relies only on 2D LiDAR. The 2D LiDAR only scans one slice of the environment and, as a result, the service robot often misses the big part of obstacles. Therefore, the service robot needs to ensure safety more efficiently without high computational costs. Furthermore, maintaining and representing the various types of information from the 2D occupancy map and the 3D object information is also essential for service robots to simplify access to multiple types of maps for diverse applications.

To address these challenges, this thesis introduces a unified map technique, which manages multiple modular map representations. This unified map enables online object mapping and updating using RGB-D data, providing 2D and 3D representations of the mapped objects. The unified map consists of three layers: the lowest layer comprises a 2D predicted occupancy map that specifies the global robot pose, the middle layer utilizes 3D object detection to build environmental knowledge and activity recognition framework using a graph neural network (GNN), and the highest layer integrates and maintains the various representations and information from the lowest and middle layers within a single module. The detected objects from RGB-D data are consolidated into 2D polygons and 3D representations using instance object segmentation with fast point cloud segmentation.

This thesis is organized as follows; Chapter 1 explains the social and technical background leading to this research objectives. The contribution of this thesis is also highlighted and discussed in this chapter. Chapter 2 presents an overview of service robots, visual and distance sensors, and intelligent technologies related with the proposed method. Furthermore, this chapter explains the previous works related with map building, manipulation, navigation, and human activity recognition.

Chapter 3 presents an investigation of the 2D predicted occupancy map method as the lowest layer to enable service robots to navigate safely in indoor environments using 2D LiDAR only. I investigated and demonstrated that the 2D predicted occupancy map enables all local planners to generate better collision-free paths using only 2D LiDAR instead of sensor fusion (RGB-D camera with 2D LiDAR) with a raw map. The proposed method was used for a sampling-based global planner using an unsupervised method.

Chapter 4 discusses the middle layer of a unified map based on 3D object detection information. The 3D object detection framework contains semantic information such as object types, colors, and positions. Using the various types of information on objects, I build a new framework to recognize the interaction between humans and their surrounding objects using GNN. This framework focused on the problem of classifying three activities where these activities were conducted in the same environment. The activity classification was learned from a 3D-detected object information corresponding to the human position. Next, human utterances were used to label the activity from the human and 3D object positions. The experimental results show that the proposed framework recognizes human activities more accurately than the deep layer aggregation (DLA) and X3D networks with conventional video datasets.

Chapter 5 proposes an integration method of two maps in the lowest and middle layers into a unified map to maintain multiple maps and object information simultaneously. The proposed framework can significantly reduce the complexity of interacting with different maps and can make it more accessible to robots. Some experiments are conducted to show the performance of the proposed frameworks to support daily human activity in human-centered environments with low computational costs.

Finally, chapter 6 summarizes the thesis, and discusses the limitation of the proposed framework and the future vision of this work.

Table of Contents

Abstract	i
Table of Contents	iii
List of Figures	vi
List of Tables	ix
1 Introduction	1
1.1 Background	1
1.2 The Challenge of Providing Accurate Information for Service Robot	2
1.3 Problem and Objectives	3
1.4 Contributions	4
1.4.1 Learning From 2D Occupancy Map and Partial Obstacle Representation ⁴	4
1.4.2 Learning From Human Activity and Object Interaction for Activity Detection	5
1.4.3 Implementation of Unified Map System for Daily Task Service Robot	5
1.5 Dissertation Scope	5
1.6 Brief Outline of Thesis	6
2 Literature Review	9
2.1 Recent Technology and Application of Service Robots	9
2.2 Machine Learning Theories and Applications for Robots	13
2.2.1 Map Representations	14
2.2.2 Robot Localization	16
2.2.3 Navigation	16
2.2.4 Theory of Artificial Neural Network	17
2.2.5 Activation Functions in Neural Networks	21

2.2.6	Batch Learning and Normalization	22
2.3	Related Works	25
2.3.1	Dynamic, Cluttered and Unseen Obstacle in Indoor Navigation Problem	25
2.3.2	Recent Research in Human Activity Recognition	27
2.3.3	Unified Map Concept in Service Robot Applications	29
3	Advanced Investigation on 2D Predicted Occupancy Map	30
3.1	Introduction	30
3.2	Integration System between Obstacle Prediction Network and ROS Navigation Stack	34
3.3	Kinematics and Local Planners Formulation	37
3.4	Experiment Setup	42
3.5	Result and Discussion	44
3.6	Summary	47
4	Activity Recognition From 3D Object-Oriented Map	48
4.1	Introduction	48
4.2	Human Activity Recognition	50
4.3	Human-Object Interactions to Recognize the Human Activity	51
4.4	Data Preparation	53
4.5	Graph Construction and Features	54
4.6	Graph Neural Network with ECC Graph Neural Network with Edge-Conditioned Convolution	55
4.7	Positional Data HOI and Scenarios	56
4.8	Comparison Method	57
4.9	Experimental Results	57
4.10	Training Results	59
4.11	Summary	61

5	Integration of 3D Object-Oriented Map and 2D Predicted Occupancy Map into Unified Map	63
5.1	Introduction	63
5.2	Unified Map Problem Formulation	64
5.3	Unified Map Framework	65
5.4	Experiments Setup	68
5.5	Results on Mapping and Obstacle Avoidance	69
5.6	Summary	70
6	Concluding Remarks	72
5.1	Conclusion	63
5.2	Future works	64
	References	75

List of Figures

Figure 1.1	: The structure of the thesis	7
Figure 2.2	: HSR Sensor and Control Model	11
Figure 2.3	: A generic path planning problem from node A to node E. The shortest path is A-B-C-D-E	14
Figure 2.4	: A grid map and its corresponding k-d tree	15
Figure 2.6	: Neural Network with one input, one output and two hidden layers	18
Figure 2.7	: The local gradient of loss for the input x and y can be computed by applying the chain rule.	20
Figure 2.8	: The LeNet-5 [38] is demonstrated to illustrate a typical architecture for a Convolutional Neural Network. It can identify integers on images. Two layers of Convolutional Layers, each followed by a subsampling Pooling Layer, are used to compute the input image. The final three layers are entirely interconnected in order to transfer the high-level characteristics to the final digit classification.	23
Figure 2.9	: On a 2-dimensional input of size $[4 \times 4]$, a max-pooling filter of size $[2 \times 2]$ - with a stride of 2 is employed. The output remains its highest value and its quantity is reduced by 4.[27]	25
Figure 3.1	: The human walking direction toward robot position is out of robot camera view. The robot changes the trajectory due to the human walking is detected by 2D LiDAR.	30
Figure 3.2	: Simulated home environment. The left part is the realistic simulated environment; the center part indicates the 2D recorded map obtained from laser observation; the right part shows the details of the complex obstacle that 2D laser rangefinders cannot fully detect.	31
Figure 3.3	: Initial step to build the predicted map using this method	32
Figure 3.4	: Raw map, predicted map and heat map from 3D environment of Small House and Large House.	34

Figure 3.5	: General ROS Navigation Stack (RNS) design for investigating the effectiveness of predicted map based on sensor input from HSR sensor.	35
Figure 3.6	: HSR base drive mechanism	38
Figure 3.7	: Elastic Band planner performance in static and dynamic obstacles	40
Figure 3.8	: General TEB planner architectures for robot navigation system	42
Figure 3.9	: Start, checkpoints and goal for the robot navigation test in SH and LH	44
Figure 3.10	: Example results using the raw map and conventional local planners. The robot cannot avoid by only using laser rangefinders	45
Figure 3.11	: The example results of using predicted map. We used laser rangefinders only. The predicted map has significant role to improve the navigation performance in real-world obstacle condition.	45
Figure 3.12	: The example results in failed navigation behavior using the predicted map and TEB Planners. The other planners also failed in this stage. The predicted map could not predict this type of chair. The leg part of the chair has a flat surface that is adhered to the floor.	46
Figure 3.13	: The example results from using 2 observation resources. We used X-Tion RGB-D camera and laser rangefinders. The robot can avoid a collision with sports equipment obstacles.	47
Figure 4.1	: Data collection system of proposed method and graph data representation	50
Figure 4.2	: Processing a subgraph in edge conditional convolution method	51
Figure 4.3	: The proposed GNN with ECC for the activity recognition	53
Figure 4.4	: Video dataset of eating, reading and working to train and test DLA and X3D networks as comparison methods.	57
Figure 4.5	: Experimental setup, collecting the dataset and visualizing the centre of the object into 3D Position.	58
Figure 4.6	: Confusion Matrix for Proposed Method, DLA and X3D	59
Figure 4.7	: Inferencing test by using HSR Head camera	60

Figure 5.1	: Unified map Framework. Integration from chapter 3 and chapter 4	65
Figure 5.2	: Examples of the object visualization in 2D occupancy map from object detector frame work	66
Figure 5.3	: Object visualization in 2D Predicted Map	67
Figure 5.4	: Comparison Result of Mapping in 3D environment.	69
Figure 5.5	: The comparison results in collision avoidance	70

List of Tables

Table 3.1	: Stages of testing configuration	36
Table 3.2	: Navigation parameters in RNS	43
Table 3.3	: Investigation and comparison results between 2D map prediction with RNS and conventional method	44
Table 4.1	: Scenarios of the data collection	56
Table 4.2	: Comparison results between the proposed method, DLA and X3D	58

Chapter 1

Introduction

1.1 Background

The aging population worldwide is on the rise and is predicted to continue to increase. According to a report in 2015, the number of people aged 60 years or older was close to 900 million, estimated to reach 2 billion by the year 2050 [1]. As a crucial asset to society, older adults need care in age-friendly physical and social environments. While healthcare facilities such as adult day care, long-term care, and nursing homes can provide the necessary healthcare, nutritional, social, and daily living support to the elderly, they may also result in the loss of independence and may be expensive. Consequently, older adults should age in place, the familiarity and security of their own homes. Encouraging older adults to engage in self-care activities independently can help them maintain their independence and provide them with a sense of accomplishment, enabling them to enjoy independence for longer [2]. Providing a physical environment that promotes active aging through innovative technologies such as smart homes and assistive robots can be an effective approach to achieving this goal. These technologies can assist older adults with their daily activities, help them maintain their independence, and enhance their quality of life. However, several challenges must be addressed to ensure that mobile service robots can be effectively applied in human-centered environments. Robots operating in environments with elderly or disabled individuals encounter people with varying needs who typically have limited experience with robots.

Moreover, robots that share their space with people must increase their adaptability and flexibility and consider human comfort and safety to be tolerated and accepted. Therefore, developing a mobile service robot that can navigate human-centered environments and interact with people safely and comfortably is a complex challenge requiring complete information and representation of the environmental condition to get a deeper understanding environment.

1.2 The Challenge of Providing Accurate Information for Service Robot

Modern intelligent and autonomous service robot applications need to comprehensively understand their environment beyond traditional 2D occupancy maps. Environmental understanding refers to the ability of a robot to perceive and comprehend its surroundings, including the environment's layout, the presence of obstacles, and the location of the robot itself. The environmental context knowledge is achieved through various sensors and algorithms that process sensor data, such as object detection, to construct a map of the environment. For service robots, a proper environmental understanding is crucial for navigation and interacting reasonably with the world. While 2D occupancy maps suffice for planar navigation, 3D information is necessary for any manipulation and can serve as a dataset for accurate behavior recognition. Therefore, many previous works constructed and maintained a unified map (2D and 3D maps) representation by using various modular frameworks [3]–[6]. Due to the modularity, extending the previous systems with additional task-related representations is easily possible. The previous system [3]–[6] demonstrates a potential application where the robot utilizes mapped representations of objects within its environment. Specifically, in this experiment [4], the robot aims to identify a requested cup by searching for all cups near a coffee machine. This information lets the robot plan a path to reach the desired cup based on the mapped 2D shapes. However, more than relying on the 2D shape representation is required when physically grasping the cup. In such cases, the robot can employ the 3D point cloud representation to enhance its ability to grasp the cup accurately.

From that example, a unified map offers clear advantages for intelligent service robots. However, their full potential has yet to be fully utilized, explored, and evaluated in many applications for service robot tasks in human-centered environments. This gap calls for additional research and development to bridge the current disparity and fully leverage the benefits that unified maps can offer in enhancing the capabilities and functionality of intelligent service robots.

1.3 Problem and Objectives

The possibility of low real-time performance is expected if the robot simultaneously applies several fusion sensor modules for navigation and manipulation in unified map. It is due to the computational process of a large amount of labeled data from object recognition and 3D map reconstruction. In previous research of unified map[], the service robot is expected to rely on fusion sensor for each task to ensure the safety during navigation and manipulation. Due to the modularity of unified map, each task can be individually dependent on one related sensor. For example, when the service robot has collected the target object position, the robot can reach the target position by only using 2D LiDAR. When the robot needs to grasp the target object, RGB-D camera can ensure the target object position and orientation. Nevertheless, if the robot only relies on horizontal 2D LiDAR during navigate the target object, as a result, most often miss the big part of obstacles. Consequently, the resulting 2D occupancy maps frequently may not accurately represent the occupied area in the environment. For instance, when the 2D LiDAR detect the table and chairs, the scanning result only shows the table and chair legs in a 2D occupancy map.

Therefore, to overcome these issues, this thesis presents development of unified map technique for managing several modular map representations to improve the safety and the various applicability. This unified map is based on three map layers. The lower layer is a 2D predicted occupancy map. The 2D predicted occupancy map is to ensure the robot safety during the navigation task and make the robot to use 2D LiDAR only when facing the unseen and cluttered obstacle. The global robot pose is specified by this layer. Then, in the middle layer is built by utilizing given 3D object detection to build the foundation of environmental knowledge and activity recognition framework using Graph Neural Network (GNN). The highest level is then integrated and maintained the various representations and information from the lower and middle layers of the map within a single module known as the unified map. The detected objects from RGB-D data are formed into 2D polygon and 3D representation in a single map using the instance object segmentation with a fast point cloud segmentation. This framework serves as an interface, unifying and simplifying access to multiple types of maps for diverse applications.

1.4 Contributions

This thesis presents an advanced learning and perception approach for mobile service robots that utilizes an efficient and accurate environmental information representation known as the unified map. The main objective of the unified map is to simplify and manage various types of information from different sensor data to enable robots to perform specific tasks such as recognition, navigation, and manipulation in human-centered environment. The main objective of the unified map is to simplify and manage various types of information from different sensor data to enable robots to perform specific tasks, such as recognition, navigation, and manipulation, more effectively. Furthermore, this thesis examines how the mobile service robot can learn and generate the important task from simple and efficient data and map representation. Further, this thesis provides the comparison result of the proposed framework with the related recent research topic with quantitative and qualitative result. This section outlines the thesis and summarizes its main contributions.

1.4.1 Learning From 2D Occupancy Map and Partial Obstacle Representation

Service robot requires exploring and performing the navigation task safely to reach the grasping target and ensure human safety in the indoor human-centered environment. An indoor home environment has complex obstacles such as chairs, tables, and sports equipment, which is difficult for robots that rely on 2D LiDAR. On the other hand, the conventional approaches overcome the problem by using 3D LiDAR, RGB-D camera, or fusing sensor data. CNN has shown promising results in dealing with unseen obstacles in navigation by predicting the partial obstacle from 2D grid maps to perform collision avoidance using 2D LiDAR only. Thus, Chapter 4 investigates and utilizes the proposed obstacle prediction network by [7] using preexisting 2D occupancy map in unified system. From this investigation, the limitation of obstacle prediction network is discovered from several experiment by using different local planners. To reduce the limitation, Chapter 5 discussed and proposed a deep reinforcement learning-based local navigation by integrating the unsupervised growing neural gas global planner approach to generate the roadmap navigation path based on uncertainty and incomplete information of the 2DPOM.

1.4.2 Learning From Human Activity and Object Interaction for Activity Detection

Human environments are designed and managed by humans for humans. Thus, adding robots to interact with humans and perform specific tasks appropriately is an essential topic in robotics research. In recent decades, object recognition, human skeletal, and face recognition frameworks have been implemented to support the tasks of robots. However, recognition of activities and interactions between humans and surrounding objects is an ongoing and more challenging problem. Chapter 3 deals with human activity recognition. By using 3-dimensional object information in the unified map, a human activity detection framework is presented. This study proposed a GNN approach to directly recognize human activity at home using vision and speech teaching data. Focus was given to the problem of classifying three activities, namely, eating, working, and reading, where these activities were conducted in the same environment. From the experiments, observations, and analyses, this proved to be quite a challenging problem to solve using only traditional convolutional neural networks (CNN) and video datasets. Moreover, human utterances were used to label the activity from the collected human and object 3D positions. The experiment, involving data collection and learning, was demonstrated by using human-robot communication to show the efficiency of the whole recognition system.

1.4.3 Implementation of Unified Map System for Daily Task Service Robot

To demonstrate the concrete application of the proposed system, Chapter 6 shows the manipulation and navigation task based on unified map and calculate the computational cost in the whole system. The experiment is conducted based on the daily activity scenarios of the human at home environment.

1.5 Dissertation Scope

The scope of the dissertation was:

- To build the unified map, the prerequisite systems should be built in advance. The 2D occupancy map is built from 2D laser scanner of the service robot. The robot is teleoperated in advance to collect the map in 3D simulated environment

and real-world environment. Then, the collected 2D occupancy map is improved by using additional framework to predict the unseen and cluttered obstacle. Next, the predicted 2D occupancy map is used to formulate the new path planning system and the result is investigated and evaluated by using static predicted obstacle and human walking.

- Furthermore, the 3D object-oriented map is also built by using existing method of deep learning-based object recognition with Xtion camera from head of service robot Toyota HSR. The deep learning-based object recognition is typically a list of bounding boxes and associated class labels that identify the objects detected in an image or video frame. Each bounding box is represented by its coordinates, typically the x and y coordinates of the top-left corner and the width and height of the box and is usually accompanied by a confidence score that indicates the algorithm's confidence in the accuracy of the prediction. The class labels associated with each bounding box represent the type of object detected, such as "bottle," "laptop," "cup," etc. Then, the ROS tf package is used in this framework to define the coordinate frames for the camera and the object of interest. The tf package allows to specify the position and orientation of each coordinate frame relative to the other. Then, Transform the 2D bounding box coordinates of the object into 3D coordinates using the camera's intrinsic and extrinsic parameters and the tf package. Further, the 3D position of the object frame is collected and used for building unified map and robot human-activity recognition teaching data.
- The unified system of the proposed method is conducted in 3D simulation by using ROS and Gazebo and in an office environment.

1.6 Brief Outline of Thesis

This dissertation is organized into six chapters. Chapter 1 gives the introduction, background, scope of this work and the contribution of this contribution. Chapter 2 presents the related works and literature reviews, as well as justifications relevant to this thesis.

Chapter 3 presents an investigation of the predicted 2D occupancy map approach to enable mobile service robots to navigate safely in indoor environments using 2D LiDAR only. I investigated and demonstrated that the 2-dimensional predicted occupancy map enables all local planners to achieve better collision-free

paths by using only 2D LiDAR instead of sensor fusion (RGB-D camera with 2D Lidar) with a raw map. This advanced investigation was then used to propose a sampling-based global planner using an unsupervised approach.

Chapter 4 deals with human activity recognition. By building the 3-dimensional object information framework, a human activity detection framework is presented in this thesis. The human activity recognition framework is based on human and object interactions. By utilizing the proposed framework, the service robot can learn to identify human activities within a home environment. Object localization data obtained from 3D object-oriented map framework are labeled using multi-modal communication, which provides the necessary activity data for teaching. The study focused on classifying three activities, namely eating, working, and reading, which were performed within the same environment. However, it was observed that this was a challenging task to accomplish using conventional 2-dimensional convolutional neural networks (2DCNN) and video datasets. To overcome these limitations, the

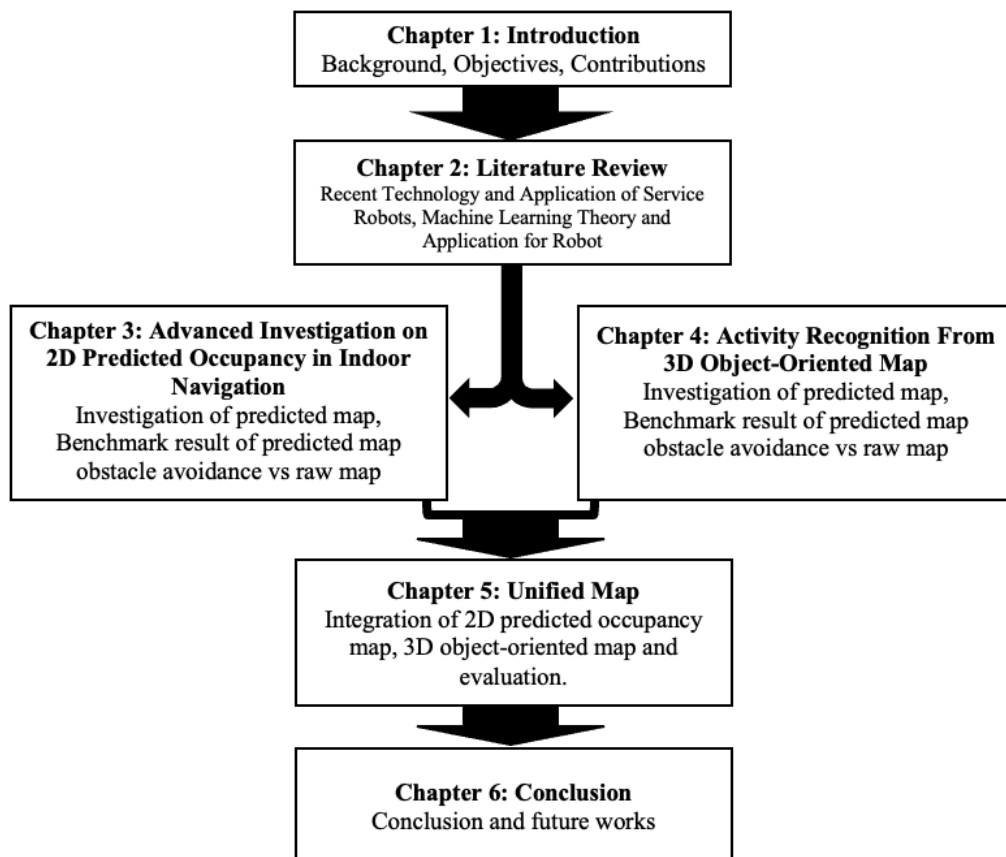


Figure 1.1: The structure of the thesis

activity data teaching was implemented using Graph Neural Network (GNN) to classify the human activity with improved accuracy and efficiency.

Chapter 5 integrate the 3D object-oriented map and 2D occupancy map into a unified map framework. The unified map is proposed to handle the information and the maintain multiple map and object information in a unified system to enable easy access and use in different applications. A unified system can significantly reduce the complexity of interacting with different maps, making it more accessible to robots. Some experiments are conducted to show the performance of the proposed frameworks to support human daily activity at real-home environment. The approaches in this thesis represent significant steps towards developing flexible, autonomous service robots that can be robustly and reliably applied in human-centered environments with low computational costs.

Finally, the chapter 6 summarizes the dissertation on the result of unified map implementation from the human-centered environment point of view. Several future directions and open problems are discussed in this chapter.

Chapter 2

Literature Review

This chapter offers a thorough and insightful overview of the latest developments in service robot technology and research, while also highlighting the related fundamental theories that underpin these advancements. Specifically, Chapter 2.1 delves into the most recent technological breakthroughs in service robots and their applications, while Chapter 2.2 focuses on presenting the foundational theories that support these innovations, including map representations, path-planning algorithms, and convolutional neural networks (CNN). Moreover, the critical evaluation of previous research works is discussed in section 2.3, which serves to highlight any potential gaps in the existing body of knowledge. By providing a comprehensive review of the existing research in this field, this chapter aims to identify areas where further investigation and development are needed, and ultimately contribute to advancing the field of service robotics.

2.1 Recent Technology and Application of Service Robots

Service robots are a sort of robot that is commonly utilized in non-industrial contexts. Frontline companies are progressively introducing robots [8]. Service robot research is gaining traction, and various definitions have been offered to describe service robots. For example, the International Federation of Robotics [9] specifies service robots as those "that perform useful tasks for humans or equipment, excluding industrial automation applications." They are defined by Wirtz et al. [10] as "system-based autonomous and adaptable interfaces that interact, communicate, and deliver service to an organization's customers." A service robot contains not only technological features for providing services, but also the capacity to interact with humans [11]. Service robots have advanced at a rapid pace, paralleling improvements in computer vision, speech recognition, sensors, and artificial intelligence. Sensor, navigation, and machine learning advancements are making robots smarter, more mobile, and less



Figure 2.1: Various Service Robots: Fetch Robot[15], Toyota HSR [19], PR2 Robot [17]

expensive for a broader range of tasks that are frequently performed in dynamic contexts, necessitating the ability to navigate across populated and sometimes restricted locations [12].

Service robots have numerous potential advantages, including increased productivity, constant service quality, and lower human expenses. Service robots enable businesses to instantly collect data from their surroundings, analyze the data on the fly, and respond to changing client needs. Intelligent robot wheelchairs, surveillance drones, teaching robots, therapy robots, entertainment robots, and self-driving autos are a few examples. According to the Gartner Hype Cycle 2019 for artificial intelligence [12] smart robots is on the rise and will reach a plateau in 5 to 10 years when the technology is widely adopted. Automation and labor substitution can help to offset some of the costs associated with the use of service robots [13].

The likelihood of the robot market expanding is quite promising. According to Markets and Markets [14] the service robotics industry is predicted to increase at a compound yearly growth rate of 22.6% from USD 37.0 billion in 2020 to USD 102.5 billion by 2025. According to a Brookings Institution survey [15], 52% of 2021 adult Internet users believe robots will perform most human activities, and 94% of those who have used robots claim they have enhanced corporate efficiency.

Industrial robots have been widely used in a variety of production jobs such as hazardous material handling, hazardous operations, and machine monitoring and operation. Service robots, on the other hand, are used for specific service functions. Service robots offer significant prospects to boost efficiency and cut expenses [11]. When COVID-19 is a serious concern to public health, consumers are more frequently



Figure 2.2: HSR Sensor and Control Model

presented with options for human and robot services in the hospitality industry, and they have a more positive view toward robot-staffed hotels [16]. With the increased usage of robots and artificial intelligence, researchers and practitioners have been debating the influence of robots on the labor market and economy because to their potential for substituting human jobs and labor [16].

Within the realm of robotics, various companies have made noteworthy contributions. For instance, Fetch Robotics, a renowned US-based robotics company, has developed an autonomous logistics support robot called "Freight." This self-propelled robot can handle a load of up to 100 kg, alleviating workers from transportation tasks and enabling them to focus on other essential duties such as picking and assembly work. The incorporation of autonomous intelligence allows Freight to navigate its environment, create maps, and adapt to changing surroundings. By embedding crucial information and rules in the map, such as movement priority routes and no-go areas, multiple robots can effectively share information and respond flexibly [13]. Another influential player in the robotics industry is Willow Garage, founded by Scott Hassan, an early architect of Google. Willow Garage's primary mission was to accelerate the development of robotics applications by establishing an innovative research lab. With the creation of the Robot Operating System (ROS), an open-source robotics framework, and the production of robots like the PR1 and PR2, Willow Garage aimed to provide a common development platform for personal robotics. The PR2 Beta Program, launched in 2010, provided free PR2 robots to 11 institutions worldwide, furthering the advancement of personal robotics [17].

Toyota, a prominent automobile manufacturer, has also made significant strides in robotics with the introduction of its Human Support Robot (HSR)[18]. Developed as part of the Partner family of robots, HSR addresses the growing need for long-term elderly care, particularly in countries like Japan. With its compact and maneuverable cylindrical body, HSR features a folding arm capable of retrieving objects from shelves, picking items up from the floor, and performing various tasks. The robot is extensively used for research in Toyota facilities and university labs in Japan and the United States.

Therefore, I used Toyota HSR in this research due to the availability and support from the university, and Toyota can make it easier to access the documentation and community resources for research. Additionally, being part of the Partner family of robots, the HSR benefits from Toyota's commitment to advancing robotic technology and its potential applications.

Toyota HSR Specifications: In this dissertation, I used The Toyota Human Support Robot (HSR) that equipped with a 4-DOF manipulator arm mounted on a torso, featuring prismatic and revolute joints, and a differential drive base[19]. Notably, the HSR possesses a revolute joint directly on top of its differential drive base, rendering it effectively omnidirectional. To facilitate manipulation, the joint space $\mathbf{q} \in \mathbb{R}^{10}$ is utilized, employing the depicted actuators in Fig. 2.1 and the specifications are shown in Table 2.1. Apart from \mathbf{q} , the HSR's end effector incorporates a parallel gripper with series elastic fingertips designed for grasping objects, with the fingertips boasting a maximum width of 135 mm.

$$\mathbf{q} = [q_{\text{head tilt}}, q_{\text{head pan}}, \dots, q_{\text{base roll}}]^T \quad (2.1)$$

Regarding perception capabilities, the HSR incorporates several sensors. For obstacle avoidance, the base-mounted UST-20LX 2D scanning laser is employed. Additionally, the HSR is equipped with the head-mounted Xtion PRO LIVE RGB-D camera for depth sensing and the end effector-mounted wide-angle grasp camera for object segmentation. The head camera's gimbal system consists of tilt and pan joints, providing 2-DOF movement, while the grasp camera moves in conjunction with the arm and wrist joints. Both cameras stream RGB images at a resolution of 640x480.

It is worth noting that the HSR's manipulation DOF encompasses a significant contribution from its mobile base. Although numerous planning algorithms excel when

dealing with high-DOF arms in stationary bases, the HSR's odometer errors accumulate during trajectory execution, which can result in missed grasps.

Table 2.1: HSR Specifications[18], [19]

Height	$\phi 430 \times 1,005 (\sim 1,350)$ mm
Weight	37Kg
Arm Length	600mm
Shoulder Height	340~1,030mm
Grasped Object	~1.2kg weight 130mm width
Maximum Velocity	0.8km/h

2.2 Machine Learning Theories and Applications for Robots

This section aims to provide a comprehensive explanation of the fundamental theories that form the backbone of this dissertation. By delving into these essential concepts, readers will gain a deeper understanding of the research studies presented within this work. The knowledge shared in this section serves as a foundation upon which the subsequent chapters build, allowing for a more nuanced exploration of the research topics. Sections 2.2.1 to 2.2.3 delve into the intricate details of map representations, path planning algorithms, and convolutional neural networks in the context of robotics applications. In Section 2.2.1, various map representations commonly employed in robotics are discussed. This includes an examination of grid-based representations, such as occupancy grids and cost maps, as well as graph-based representations, such as topological maps and visibility graphs. Exploring these different approaches helps understand the advantages and limitations of each representation, enabling informed decision-making in the context of robotic mapping. Moving on to Section 2.2.2, the focus is on path planning algorithms. The discussion covers classical algorithms like Dijkstra's algorithm, A* search, and probabilistic roadmaps (PRMs), along with more advanced techniques such as Rapidly-exploring Random Trees (RRT) and its variants. Understanding the underlying principles and trade-offs of these algorithms allows researchers and practitioners to effectively navigate the vast search space of possible paths and make informed decisions in robotics applications involving autonomous navigation and path planning. Finally, in Section 2.2.3, the role of convolutional neural networks (CNNs) in the realm of robotics

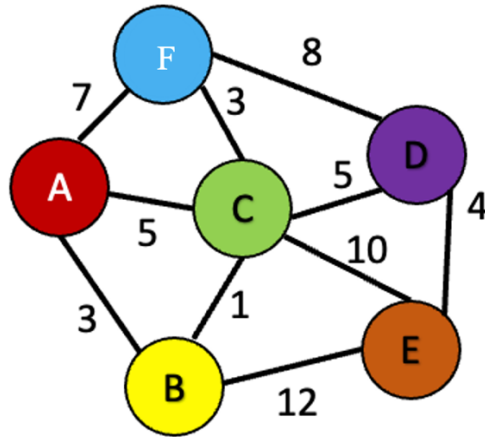


Figure 2.3: A generic path planning problem from node A to node E. The shortest path is A-B-C-D-E

is explored. With the rise of deep learning, CNNs have proven highly effective in tasks such as object recognition, localization, and semantic segmentation. The fundamental concepts of CNNs, including convolutional layers, pooling layers, and fully connected layers, are discussed. Moreover, the section explores how these networks can be trained and fine-tuned for specific robotics applications, such as visual perception and scene understanding.

2.2.1 Map Representations

To plan a path, the robot must digitally represent the environment. There are two distinct approaches: discrete and continuous approximations. In the discrete approximation, the map is divided into chunks of equal or varying sizes, such as a grid or hexagonal map, or even rooms in a building, which are referred to as topological maps. Discrete maps are well-suited for a graph representation, where each chunk corresponds to a node, and these nodes are connected by edges if the robot can navigate between them. An example of a topological map is a roadmap, where intersections serve as vertices and roads as edges, labeled with their respective lengths (Figure 2.3). From a computational standpoint, a graph can be stored as an adjacency matrix. On the other hand, a continuous approximation involves defining inner boundaries (obstacles) and outer boundaries, usually in the form of a polygon, while paths can be encoded as sequences of points represented by real numbers. Despite the memory advantages

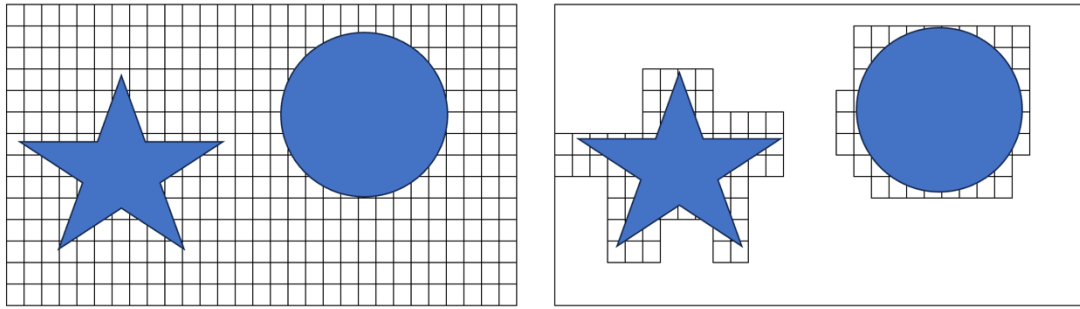


Figure 2.4: A grid map and its corresponding k-d tree

offered by continuous representations, discrete maps remain the prevalent choice in the field of robotics.

The most commonly used map for identifying obstacles is called the 2D occupancy map. In this map, the environment is divided into squares of a specific size, such as 1cm x 1cm, and obstacles are marked on these squares. A probabilistic occupancy grid map can also assign a probability value to each square, indicating the likelihood of containing an obstacle. This is especially useful when there is uncertainty in the robot's position while sensing obstacles. However, grid maps have drawbacks in terms of their high memory requirements and the time it takes to process large numbers of vertices in the data structure. To address this, an alternative approach is to use a *k-d tree* structure to store the grid map. A *k-d tree* divides the environment recursively into k pieces, such as four pieces for $k = 4$. Each piece is further divided into four sub-pieces until the desired resolution is achieved. This tree structure allows for efficient storage, as not all vertices need to be divided into the smallest possible resolution. Only areas with obstacles require further division.

In addition to discrete representations, there are also combinations of discrete and continuous representations. For example, roadmaps used in GPS systems are stored as topological maps, which include GPS coordinates for each vertex. These maps may also incorporate overlays of aerial and street photography or even 3D point clouds stored in an octree, a type of 8-dimensional tree. These diverse map representations are utilized at various stages of the path planning process based on their specific characteristics and suitability.

2.2.2 Robot Localization

Adaptive Monte Carlo Localization (AMCL) [20], which utilizes the Particle Filter, is employed to address the problem of localization in this work. In this approach, each particle represents a potential solution for the robot's position. The Particle Filter undergoes the following steps iteratively:

Do

1. A particle is randomly selected from the previous distribution and its position is adjusted based on the physical system.
2. The particle is placed in the discretized state space, and if it occupies an empty bin, the number of non-empty bins (denoted as k) is incremented.
3. The particle's weight is determined by how well it aligns with the recent sensor data. Particles that closely match the sensor data receive higher weights than those that deviate to a lesser extent.
4. The sample size bound M_x is adjusted based on the number of non-empty bins k . As the level of agreement among particles increases with smaller k , the final sample size (denoted as n) decreases.

This iterative process continues until the sample size reaches or exceeds the specified bound $n \geq M_x$. As the iterations progress, the particles gradually converge towards the most probable position of the robot. In step 3, the correspondence between the sensor data and the particle's environment is estimated by comparing them to the global map. However, it's important to note that AMCL may encounter challenges in crowded and dynamic areas, as people can obstruct significant environmental features, making it difficult to establish a clear correspondence.

2.2.3 Navigation

The navigation system relies on the Navigation Stack within ROS [21]. This stack includes a global planner responsible for calculating the most efficient long-distance routes from point **A** to point **B**, utilizing a global costmap. The costmap is a grid-based representation of occupancy, where each cell value indicates the likelihood of being in an unsafe position. The global costmap primarily relies on the pre-existing global map of the environment but also incorporates sensor data. This allows for the detection of new objects through sensors, which can then be integrated into the global

costmap, even if they are not initially included in the map of the world. In this section, some basic path-planning algorithms and problem will be discussed.

Dijkstra's algorithm. One of the earliest and simplest algorithms for finding paths is Dijkstra's algorithm[22] (Dijkstra, 1959). The algorithm starts at the initial node, marking the direct neighbors of that node with the cost to reach them. It then iteratively selects the node with the lowest cost and updates the costs of its adjacent nodes if a lower cost can be achieved through that node. This process continues until the goal node is reached, and the robot can follow the edges with the lowest cost.

In Figure 2.3, Dijkstra's algorithm would initially assign costs of 3, 5, and 7 to nodes B, C, and F, respectively. It would then explore the edges of node B, which has the lowest cost. During this exploration, it discovers that node C can be reached in fewer steps ($3+1 < 5$), so the cost of node C is updated to 4. To fully evaluate node D, Dijkstra's algorithm considers the remaining edge and assigns a cost of $3 + 12 = 15$ to node E.

The node with the lowest cost now becomes node C (cost 4). Node E is then updated to a cost of 14 (smaller than 15), and node V is assigned a cost of $4 + 5 = 9$, while node F remains at a cost of $4 + 3 = 7$. Despite finding two paths to the goal, with one being better than the other, the algorithm continues because there are still unexplored nodes with overall costs lower than 14. Continuing the exploration from node D leads to the discovery of a shortest path A-B-C-D-E with a cost of 13, and no more nodes remain to be explored.

Since Dijkstra's algorithm will not stop until no node has a lower cost than the current cost to the goal, it can be certain that it will find the shortest path if one exists. This property makes the algorithm complete. The algorithm explores nodes in a manner that resembles a wave front emanating from the starting vertex and eventually reaching the goal. However, this process can be highly inefficient, especially when exploring nodes far from the goal. This inefficiency becomes apparent when additional nodes are introduced to the left of node A in Figure 2.3

2.2.4 Theory of Artificial Neural Networks

This section provides a concise overview of the important fundamental ideas pertaining to neural networks, with a particular emphasis placed on convolutional neural networks and graph neural networks. In addition, this part explains the

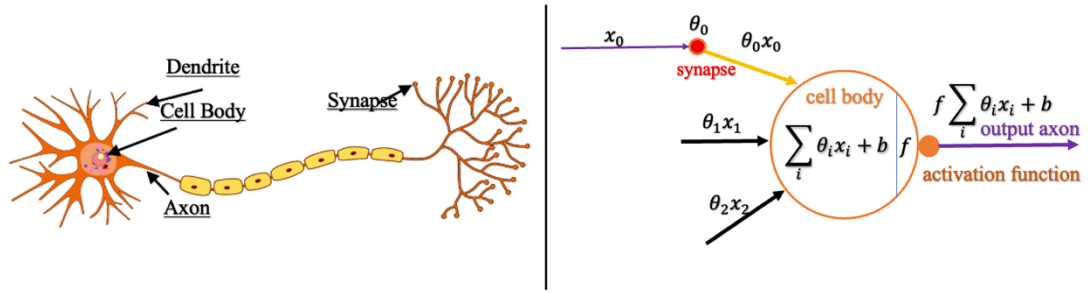


Figure 2. 5: Artificial neuron (right) and organic neuron (left). The artificial neuron simulates the dendrites as the weighted inputs and processes the sum using an activation function. [23] fundamental learning process, regularization, activation function, batch learning, and normalization in artificial neural networks. Additionally, the basic graph neural networks for human activity recognition were studied in this part.

Artificial Neural Networks: The biological neuron present in the animal brain serves as the model for the artificial neuron. Each neuron in the human brain, which has over 100 billion neurons and processes sensory data including vision, touch, and sound, gets several inputs from neighboring neurons known as dendrites. By processing these inputs, the neuron determines whether a specific action potential has been reached. If this threshold is surpassed, the neuron "fires" through its singular output, known as an axon. The output generated by the axon is then transmitted to all subsequent interconnected neurons.

The artificial neuron, also known as a perceptron, simplifies the modeling of biological neurons. In this simplified representation, each artificial neuron possesses n input connections. The neuron processes these inputs by calculating the weighted

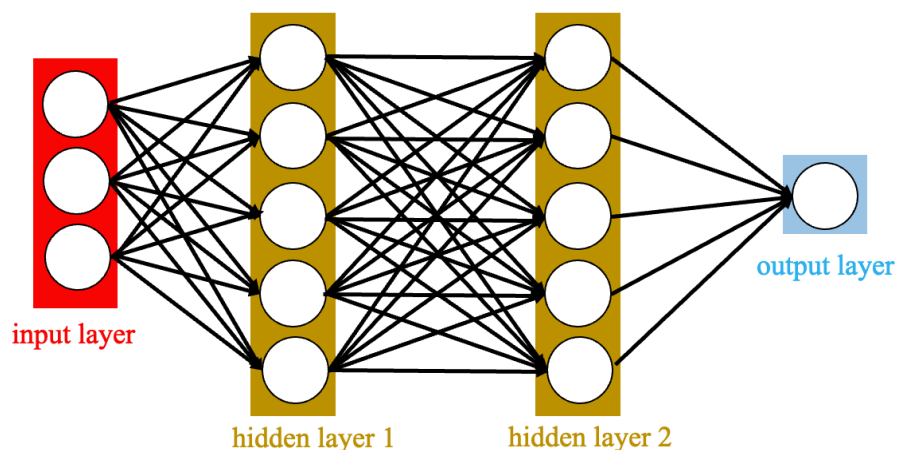


Figure 2.6: Neural Network with one input, one output and two hidden layers

sum, incorporating a bias term, and subsequently applying an activation function: $f(\sum \theta_i x_i + b)$ [23]. Figure 2.4 illustrates an artificial neuron and the biological neuron.

Neural networks approximate a nonlinear function $f(x)$ by chaining together numerous neurons. The parameter set θ , which comprises all weights θ_i of all neurons, must be changed such that the Neural Network produces the best possible function approximation. The process of determining an appropriate parameter set is known as learning. The artificial neurons are organized in layers using a feedforward Neural Network. The neurons in the layers are linked in a forwarding manner. There are no connections between neurons that are fed back to preceding neurons. Each network includes one input layer for processing raw input data and one output layer for storing the approximated outcome. Between those two levels, there may be one or more hidden layers where significant computation occurs. A fully connected, feedforward deep neural network can be seen in Figure 2.6.

Learning Process: The learning process's purpose is to identify a parameter configuration that produces the best possible function approximation. The real result Y of a certain input X is presented in supervised learning and may be utilized to adjust the parameters. It is a looping process that includes the following steps.

1. **Forward Pass :** The input X passes into the network, and the predicted result $Y_{pred} = f(X, \theta)$ is obtained.
2. **Loss :** The loss $L(\theta)$ is computed when the estimated output Y_{pred} is compared to the actual result Y . The loss function is chosen based on the learning task. The relevant loss functions are given below:

- Mean-square-error [24]. It is generally used and calculate the L2-distance between Y_{pred} and Y .

$$L(\theta) = \frac{1}{2} \sum_{i=1}^n (Y_i - Y_{pred,i})^2 \quad 2.1$$

- Logistic Loss function [24]. The logistic loss function penalizes correctly classified elements that have low confidence. Still, samples that are incorrectly classified receive penalties more severely.

$$L(\theta) = \sum_{i=1}^n \log(1 + \exp(-Y_i \cdot Y_{pred,i})) \quad 2.2$$

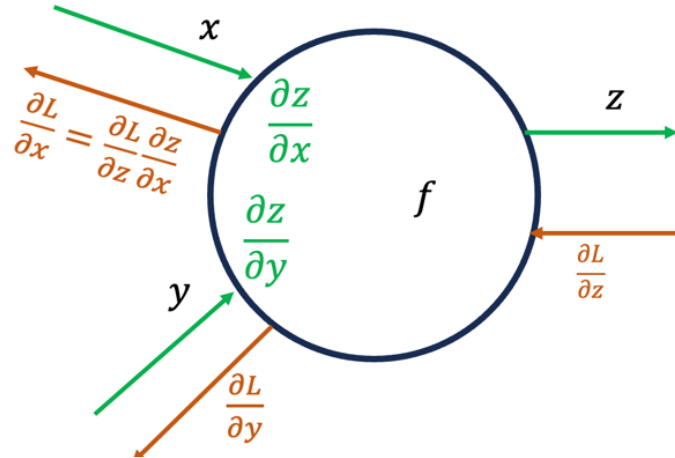


Figure 2.7: The local gradient of loss for the input x and y can be computed by applying the chain rule.

Back-propagation: The calculation of the global gradient of loss $\nabla L(\theta)$ is performed and subsequently propagated in a backward manner throughout the neural network. Initially introduced by [25], the back-propagation algorithm contributes local loss gradients to all hidden neurons. The fundamental principle underlying this algorithm is the chain rule, which enables the computation of derivatives for composite functions by multiplying their respective local derivatives. Specifically, when considering a function $y = g(x)$ and $z = f(g(x))$, the derivative $\frac{\partial z}{\partial x}$ can be determined using equation 2.3, as explicated in [26].

$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y} \frac{\partial y}{\partial x} \quad 2.3$$

The chain rule is applied to facilitate the propagation of the global gradient loss $\frac{\partial L(\theta)}{\partial \theta}$ backward through the network. This propagation occurs in the reverse direction of the forward pass. As depicted in Figure 2.7, a neuron characterized by the function $z = f(x, y, \theta)$ is examined. Its local derivatives denoted as $\frac{\partial z}{\partial x}, \frac{\partial z}{\partial y}$ and can be determined during the forward pass. During the back-propagation phase, the local gradient of loss is computed by multiplying the local derivative with the local gradient of loss from the connected neuron in the subsequent layer $\frac{\partial L}{\partial z}$. Consequently, individual local gradients of loss: $\frac{\partial L}{\partial x}, \frac{\partial L}{\partial y}$ are obtained for each input of the neuron, which is then further backpropagated to the primary neurons. When a neuron is connected to multiple neurons in the subsequent layer, the gradients are aggregated using summation. [23].

Update: The weights of all neurons undergo updates during the training process, and one commonly used optimization algorithm is stochastic gradient descent (SGD). SGD combines the principles of Batch Learning and gradient descent. Gradient descent modifies the weights in the opposite direction of the gradient of the loss function, aiming to bring the function approximation closer to the local minimum with each iteration. It is anticipated that, after a number of iterations, a local minimum will be reached. The update rule of gradient descent, as depicted in Equation 2.4, governs this process. The learning rate parameter α determines the pace at which the minimum is approached. If the learning rate α is set too high, there is a possibility that the minimum cannot be attained, as the magnitude of the steps taken will be excessively large and overshoot the target.

$$\theta_i \leftarrow \theta_i + \alpha \nabla_{\theta} L(\theta) \quad 2.4$$

2.2.5 Activation Functions in Neural Networks

There are three generally used activation functions: sigmoid, tanh, and ReLU, which will be explained in this chapter. The most widely used function is the ReLU function, which may be found expressed as an Equation in 2.5. It prevents the output from falling below zero in any circumstance. It has a non-saturating form, which makes it possible for the gradient to converge more quickly while being trained. An additional benefit is that it is a straightforward operation that requires less computation cost. [27]

$$\text{ReLU}(x) = \max(0, x) \quad 2.5$$

Equation 2.6 is shown the sigmoid function, which is still another type of activation function. It converts the value x that is given as input into a value between 0 and 1. Large negative values become 0 and large positive values become 1.

$$\text{sigm}(x) = \frac{1}{1 + e^{-x}} \quad 2.6$$

The sigmoid function is less popular because it has significant drawbacks. If the output of the neuron saturates at 0 or 1, the local gradient approaches zero. During back-propagation, the global gradient is multiplied by the local gradients, resulting in a product of zero. Eventually, the weights will stop changing, and the network will be incapable of effective learning. It is particularly problematic if the network is initialized

with weights that result in saturated outputs. Another drawback is that the sigmoid function output is not zero-centered. [27]

Equation 2.7 shows the tanh function. It centers the sigmoid function at zero. The disadvantage of saturation continues to exist.

$$\tanh(x) = 2 \operatorname{sigm}(2x) - 1 \quad 2.7$$

2.2.6 Batch Learning and Normalization

Batch Learning is based on the concept of processing a set of m training samples (mini-batches) as opposed to a single training example. The gradient is the average of all m training examples that have been processed. This can result in a more precise gradient with less variance, which can reduce training time. In addition, Batch Learning accelerates training when a graphics processing unit (GPU) is used. All training samples can be independently, or in parallel, processed. [27]

Batch normalization: By zero-centering and rescaling the data, batch normalization[28] applies normalization to the entire collection. The expected mean and variance of the normalized data are close to zero and one, respectively. In a batch H with m samples, each value h_i is normalized according to equation 2.9 over the entire batch. Mean (Equation 2.10) and variance (Equation 2.11) are calculated element-by-element for each spatial position in the entire batch. $\delta > 0$ is a minuscule number that prevents division by zero. Using Equation 2.11, the normalized value \hat{y}_i will be further processed. γ_{bn} and β_{bn} are batch normalization bn layer parameters that are learnt alongside the initial parameter set of the Neural Network. The additional learning dynamics enhance the expressive capacity of the network.

$$\mathcal{Y}_i = \gamma_{bn} \hat{y}_i + \beta_{bn} \quad 2.8$$

$$\hat{y}_i = \frac{h_i - \mu}{\sigma} \quad 2.9$$

$$\text{with } \mu = \frac{1}{m} \sum_{i=1}^m h_i \quad 2.10$$

$$\text{with } \sigma = \sqrt{\frac{1}{m} \sum_{i=1}^m (h_i - \mu)^2 + \delta} \quad 2.11$$

The application of batch normalization to both the input data and the output of any hidden layers has a regularizing effect on the learning process and precludes overfitting. Another benefit is that it reduces training time. It must be noted that batch normalization is only applicable when the precise position of the features is irrelevant and only their presence in the input data.

2.2.7 Convolutional Neural Networks

Convolutional Neural Networks are modeled after the brain's receptive field, which processes sensor input data and is sensitive to specific stimuli, such as visual system boundaries. They efficiently manage large input data and are therefore extensively used in modern approaches in the fields of Computer Vision, such as object detection [29]–[32] or image segmentation [33]–[37].

Figure 2.8 depicts the LeNet-5[38] image recognition system. It provides the typical architecture of Convolutional Neural Networks, which consists of Convolutional Layer stacks followed by a subsampling Pooling Layer. Typically, the final concealed layers of a network are entirely interconnected to compute the network's final low-dimensional output. In the early phases of a network, it is presumed that low-level features such as edges and corners are learned, while in later layers, these features are combined to form high-level features.

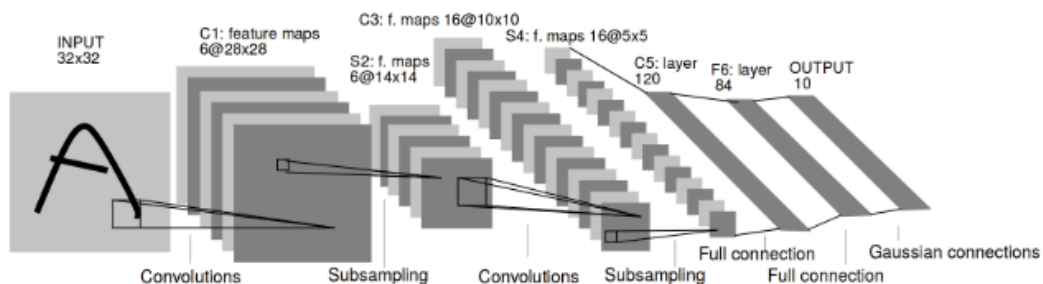


Figure 2. 8: The LeNet-5 [38] is demonstrated to illustrate a typical architecture for a Convolutional Neural Network. It can identify integers on images. Two layers of Convolutional Layers, each followed by a subsampling Pooling Layer, are used to compute the input image. The final three layers are entirely interconnected in order to transfer the high-level characteristics to the final digit classification.

Convolutional Layer: The Convolutional Layer extends the discrete convolution operation, which applies a square filter f of size $[m \times m]$ with $m = 2k + 1$ to an input matrix g at position $[x, y]$ by computing the dot product. The discrete convolution procedure is illustrated by the Equation 2.12

$$h[x, y] = f * g[x, y] = \sum_{u=-k}^k \sum_{v=-k}^k f[u, v]g[x - u, y - v] \quad 2.12$$

One neuron in a Convolutional Layer is represented by a $[m \ m \ d]$ -sized filter. The neuron's weights consist of the filter values and a bias b . The filter shifts the input matrix g by a depth of d and produces an output $h[x, y]$ for each $[x, y]$ position. Note that the depth d of the input matrix g and the filter f is identical. Zero padding can be used to produce an output h that has the same dimension as the input g . The zero-padding operation extends the input matrix g by $(m - 1)/2$ rows or columns with zero values on each side. The Convolutional Layer is comprised of various filters (= neurons). All filters with the same size but distinct filter values are applied to the same input g to generate an activation map. The layer's final output is an array of all activation maps. Commonly, the filter is shifted by a constant stride S over the input, so that every S th position of the input is convolved. It reduces the amount of data in the subsequent layer.

The number of weights in a Convolutional Layer is kept low compared to a fully-connected Layer, and the layer's computation is more efficient. Typically, the filter size is maintained considerably smaller than the input data, resulting in the detection of small, low-level features. Small filters necessitate fewer parameters as well as fewer convolution operations. In addition, the filter is applied to various input locations owing to filter shifting. This is based on the premise that identical features can present in various locations of the input and be detected by the same neuron. Feature detection with Convolutional Layers is therefore translation-invariant.

Pooling Layer : The Pooling Layer performs subsampling in the width and height spatial dimensions by applying a downsampling filter to the input. Max- and average-pooling filters are common pooling filters. At max-pooling, a $[m \times m]$ -sized filter is slid over the input, leaving only the highest value in the output. Figure 2.9 provides an example: A $[2 \times 2]$ -filter is shifted with a 2-dimensional step over the 2-

dimensional input. The output size is a quarter of the input. In average-pooling, the filter calculates the average of each position $[x, y]$ and its companions.

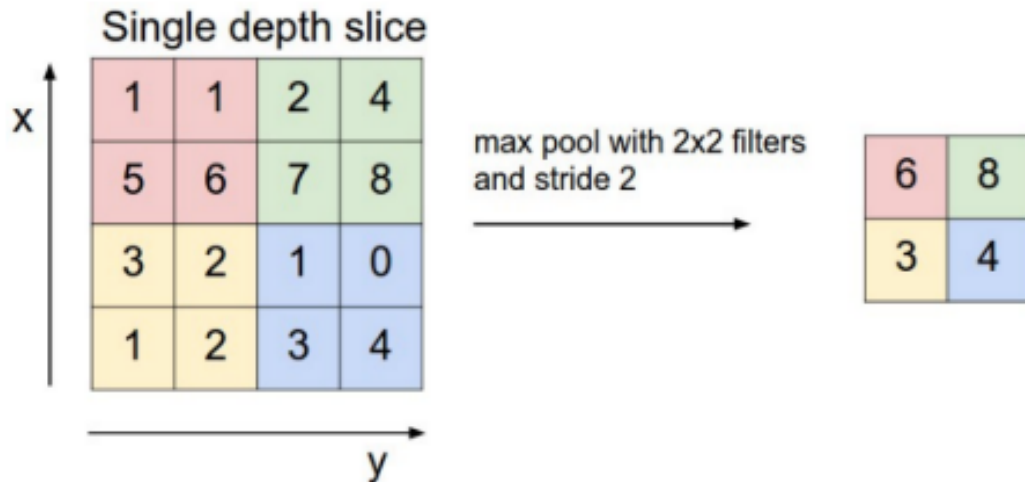


Figure 2. 9: On a 2-dimensional input of size $[4 \times 4]$, a max-pooling filter of size $[2 \times 2]$ - with a stride of 2 is employed. The output remains its highest value and its quantity is reduced by 4.[27]

Pooling reduces data capacity, resulting in an increase in network efficiency. It is useful when the precise position of a feature is not important, but rather its presence in the input.

2.3 Related Works

2.3.1 Dynamic, Cluttered and Unseen Obstacle in Indoor Navigation Problem

Mobile manipulation robot tasks are typically specified by end-effector and base navigation subgoals. Hence, navigation has the main contribution to the mobile manipulation tasks, such as reaching and placing the target object.

According to [39], [40] typical ROS navigation approaches consists of a combination of global and local planner approaches. Global planner techniques like prospective field strategies, cell decomposition, roadmaps, and Dijkstra require a complete environment model. The global method calculates a whole trajectory from the beginning place to the target position. These methods cannot induce reactive

avoidance behavior in the robot's surroundings. In this scenario, integrating local planner methodologies [41] can improve navigation performance.

The ROS-based navigation system often uses the Trajectory Rollout [42] and the Dynamic Window Approach (DWA) [41]. A 2D occupancy map planner controls a mobile robot. Local planners use a 2D occupancy map to compute a mobile robot's path from start to finish. DWA controls discretized space differently from Trajectory Rollout. The DWA local planner evaluates just the collision-free area following the current state, whereas the Trajectory Rollout simulates all subsequent states [42].

Elastic Band (EBand) and Time Elastic Band (TEB) are other local planners [43], [44]. The EBand planner specifies a subset of the maximum free space location in a configuration using the bubble area and the robot can move more freely [43]. The simplified robot model and 2D occupancy map data produce the bubble. The Bubble band analyzes obstructions and internal pressures to decrease bubble-to-bubble force. EBand planners generate collision-free, deformable trajectories. It continuously deforms the calculated route to avoid obstructions. The mobile robot can react to a sudden movement. TEB works like EBand. It minimizes time expense rather than energy [44].

Deep reinforcement learning have made indoor learning-based robot navigation more popular [45]–[50]. Guldering et al. [9] used PedSim[51] ROS Navigation stack plugins to avoid active human walking and static obstacles like walls and corridors in their learning-based local planner. 2D LiDAR only detects one slice of the surroundings and sometimes misses large items. Thus, 2D occupancy maps frequently misrepresent the robot workspace's free area. These works [39], [45], [52] suggest choosing and preparing obstacles based on the robot's capabilities and sensor restriction. To acquire a comprehensive environment representation, the navigation system can use an RGB-D camera, 3D LiDAR, or sensor fusion [53],[54]. However, this program might drain power and slow navigation performance.

Lundell et al. [55] used a fully convolutional network (FCN) autoencoder to estimate laser rangefinder scanning with actual obstacle distances from 2D laser scans. Their later result [56] unified the processed laser scans into occupancy maps with uncertainty approximation. The actual obstacle distances data is collected for training with a 3D camera measurement. The authors presented that their strategy could avoid collisions with obstacles in real-world scenarios. This method depends on an additional RGB-D camera for making the training examples, which is discarded when

implementing the navigation in realistic scenarios. Moreover, this method cannot be implemented directly in a different type of 2D LiDAR angle and range data.

Kollmitz et al. [7] proposed obstacle prediction networks for predicting unseen obstacle shapes in 2D map created with 2D LiDAR. Their strategy uses a fully convolutional neural network (FCN) trained from collision events recorded with a bumper. The results confirm that the trained FCN on a simulated collision dataset can predict and segment the unseen obstacles in a 2D map. They also demonstrate that the implementation of this method can be further enhanced by combining new obstacle examples collected during real-world applications.

The comparison result that indicates the advantages and limitations of obstacle prediction networks [7] with other various types of obstacle and local planners and sensor configuration has yet to be discussed in recent papers or in original papers. Therefore, we initiate to investigate the efficiency between this method and the conventional method for robot navigation in a 3D simulated environment. Since our robot has multiple sensors and cameras for manipulation and navigation, the combination sensor for the navigation task will spend more computation time. However, the computation cost is not our focus in this work. We focused on which method can enhance the robot's navigation efficiently and avoid collision in different types of objects that are difficult to detect by using 2D LiDAR only. Using the Kollmitz et al.[7] result, we can rely on 2D LiDAR only for navigation and decrease its limitation with the recent result of neural networks. We evaluated this result with RNS and various conventional local planners such as TEB Planner [44], DWA Planner [41], EBand Planner [43], and a global planner.

2.3.2 Recent Research in Human Activity Recognition

Recognizing human activities using graph convolutional neural networks has attracted the attention of many robotics and computer vision researchers in recent years. Activity recognition is required for homecare robots that are taking care of children, the elderly, or persons with disabilities. With this implementation, the inference of human activities using perceptual information plays an important role in human-robot interactions, smart surveillance, and content-based video analysis. A lot of work has been conducted toward predicting human activities in 2D and 3D images and videos where the overall technique observes and correlates with HOI. [57]–[60]. The principal

method for predicting HOI is extracting visual characteristics from instance detectors and spatial knowledge to instantiate multi-streams of deep neural networks. Each stream includes detected human and contextual objects. The last step is designed for inference application. The work of [57] presented the forecasting of a human activity by predicting the possible trajectory movement towards a targeted object from RGB and depth sensors data. Using this strategy, the predicted trajectory for the ongoing action can be visualized. However, the camera should be set up at a certain distance and height to avoid the broader field of view that may lead to occlusions and poor activity prediction. Wang *et al.* [61] proposed a fully-convolutional approach that predicts the interactions between human-object pairs from RGB images. The network can predict the context of human activities by localizing the interaction points from the object, human and pairwise streams.

GNN has been utilized to predict human activities from HOI by extracting the image features into graph structures. Qi *et al.* [58] used a graph parsing neural network to detect HOI and predict human activity from various RGB datasets. Morais *et al.* [59] introduced asynchronous-sparse interaction graph networks, which are constructed from the temporal structure and content label of human-object interaction activities. This method uses a graph attention network model for HOI detection in the RGB dataset. Their approach involves the construction of nodes and edges from visual features. An adjacency matrix defines the structure and properties of the network, and is updated by a weighted sum of the messages from the other nodes. Finally, for interaction inference, a node readout function is employed. Simonovsky and Komodakis [62] proposed the edge-conditioned convolution (ECC) GNN, a spatial domain operation on graph signals in which filter weights are conditioned on edge labels and dynamically formed for each input sample. It was demonstrated that this strategy could generalize the traditional convolution on graphs if edge labels are suitably chosen, and this claim was empirically tested on MNIST. Moreover, this method was also evaluated for point cloud classification, achieving a new state-of-the-art performance on the Sydney dataset. The current work was inspired by [62], and their model was used for the proposed method. A 3D object and human pose of point clouds data were used to construct the edge features for the graph data by transforming the centre of the 2D bounding boxes of the YOLO and Mediapipe face detection frameworks into the 3D point cloud by utilizing the ROS library features. The data was labelled using human speech when the robot questioned the human to learn the HOI.

In the application for this study, a 3D object and human pose data were used to construct the graph data by transforming the centre of the 2D bounding boxes into a 3D point cloud by utilizing the ROS library features. The data was labelled using human utterances in a human-robot communication scenario for the data collection. The data was collected during the ongoing activity, and the robot asked the human, "What are you doing now?". Then, the uttered answer would be specified as an activity label. After collecting the data, a graphical representation was constructed for the training process.

2.3.3 Unified Map Concept in Service Robot Applications

Modern intelligent and autonomous robotic applications often demand a more comprehensive understanding of the environment than what can be derived from traditional occupancy grid maps. A prime example is the task of autonomous semantic exploration, where a robot must label objects in its surroundings while navigating independently. To tackle this task effectively, the robot needs to maintain several types of maps. Firstly, an occupancy map is necessary for navigation purposes. Additionally, an exploration map is needed to keep track of the areas that have already been visited. Lastly, a semantic map is essential for recording the locations and labels of objects in the environment.

While three-dimensional (3D) information is indispensable for manipulation tasks, two-dimensional (2D) information often suffices for navigation tasks [63]. However, the occupancy map in this system frequently fails to accurately represent the obstacle areas in the environment. For instance, when the 2D LiDAR detect tables and chairs, the resulting 2D occupancy map only shows the legs of these objects.

As the number of required maps increases, applications face the challenge of managing and dealing with various map representations, which can become burdensome. Therefore, it is crucial to create and maintain an accurate representation of the environment while ensuring safety during navigation.

Chapter 3

Advanced Investigation on 2D Predicted Occupancy Map

3.1 Introduction

Service robots are frequently operated in designed environments for and occupied by humans. Especially in countries with low birthrate issues, such as Japan and many other developed countries, service manipulation robots such as Toyota HSR are possible solutions to improve the quality of life (QOL) for aged people in the home environment or the public area [18]. These environments usually consist of various complex objects such as chairs, tables, and sports equipment with a narrow workspace. At the same time, we expect to increase the robot's adaptability in various environments and ensure the user's safety and comfort.

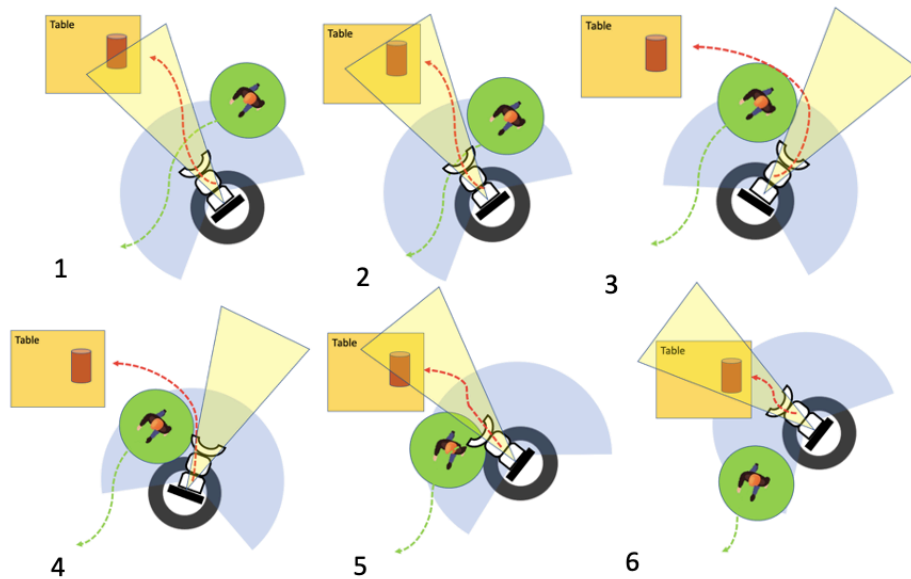


Figure 3.1: The human walking direction toward robot position is out of robot camera view. The robot changes the trajectory due to the human walking is detected by 2D LiDAR.

The service manipulation robot generally can be combined with the manipulation and navigation task as whole-body manipulation or decomposed for

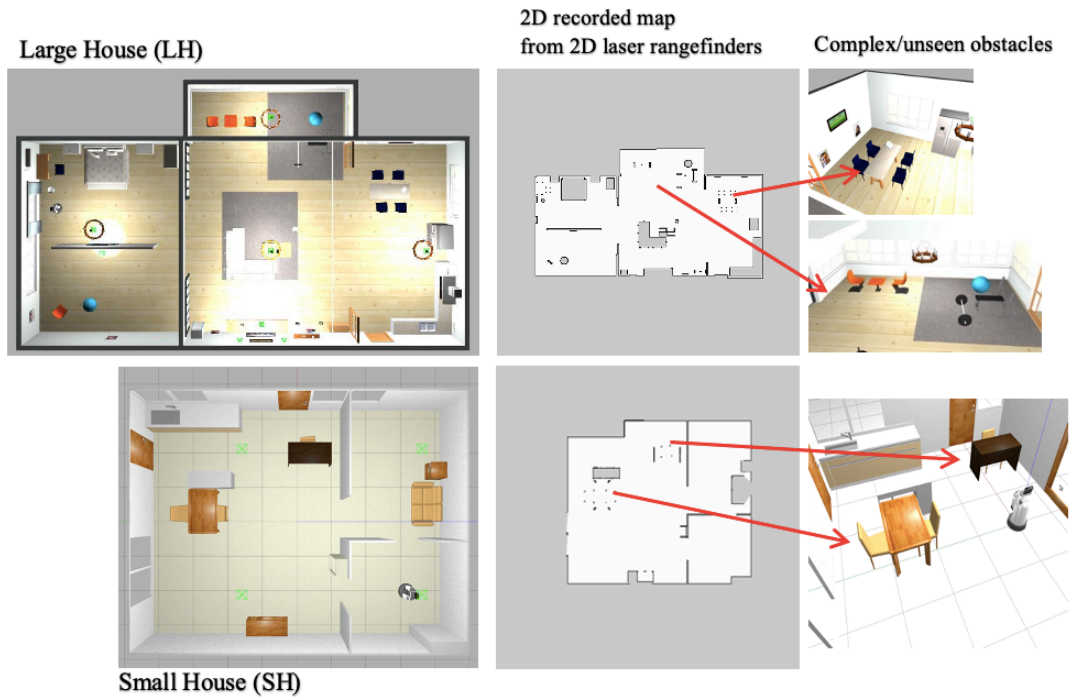


Figure 3.2: Simulated home environment. The left part is the realistic simulated environment; the center part indicates the 2D recorded map obtained from laser observation; the right part shows the details of the complex obstacle that 2D laser rangefinders cannot fully detect.

different purposes and needs [19], [64]–[66]. The decomposed task of the service manipulation robot is typically defined by a set of mobile base navigation and arm manipulation subgoals. Thus, navigation significantly enhances the arm manipulation tasks, such as picking, placing, pulling, and pushing. Particularly, Toyota HSR has a shorter arm [18] and less degree of freedom (DoF) than other domestic service manipulation robots such as Fetch [67], Tiago[a], and PR2 [17]. Moreover, the compact body of HSR is more suitable for navigating narrow and small environments such as a house, office, and hospital room. These abilities have been shown in [17]–[19], [67], [68], and the result shows that HSR always performs the base navigation task to reach the grasping object.

Toyota HSR has several sensors, such as 2D LiDAR to build the 2D map and sense the obstacle for navigation, IMU for localization, and several cameras, including an RGBD camera and a pair of RGB cameras in a stereo setup in the head part. However, the possibility of low real-time performance is expected if we simultaneously apply several fusion perception modules for navigation and manipulation. It is due to the computational process of a large amount of labeled data from object recognition and 3D map reconstruction. On the other hand, it is difficult for a service robot to detect

the surrounding approaching dynamic obstacle, such as a walking human toward the robot position where the walking direction is out of camera view such as in Figure 3. 1 description. Thus, the robot requires a more extensive range of sensors such as 2D LiDAR to generate a safer path a 2D map. Nevertheless, horizontal 2D LiDAR only scan one slice of the environment and, as a result, most often miss the big part of obstacles. Consequently, the resulting 2D occupancy maps frequently may not accurately represent the occupied area in the environment. For instance, when the 2D LiDAR detect the table and chairs, the scanning result only shows the table and chair legs in a 2D occupancy map. Therefore, it may produce the unsafe trajectory planning from global and local planners when the mobile manipulation robot performs the navigation task. Figure 3. 2 described the unseen obstacle or partially observed object examples viewed by 2D LiDAR in the human-centered environments that we obtained from Toyota HSR ROS package for SH and from AWS RoboMaker for LH. Many previous works [39], [45], [47], [48], [52], [69] have proposed DRL-based local planners and used the recorded 2D maps from 2D LiDAR to train the deep with static obstacles and reactive human walking. This trial-and-error strategy lets the robot learn static and dynamic obstacle avoidance behavior based purely on observations. The training process is accelerated with deep neural networks, and recent research showed excellent results in robot navigation. The purpose of using 2D LiDAR only in these navigation approaches is to reduce the computational cost during learning navigation and avoid active walking humans with simple static obstacles in indoor environments. However, the unseen obstacles or partially observed objects are not considered as the navigation task problem in these recent works [39], [45], [47], [48], [52], [69].

Based on this report [7], the tiny or incomplete obstacle representation in a 2D occupancy map can trap the robot when an attempt to avoid the collision. Moreover, the recent result from [7] has shown an outstanding solution to solve the unseen obstacle problem in several indoor environments without inflation layer or safety layer parameter tuning using the feature of ROS Navigation Stack. Since all the local planners depend on the 2D occupancy map information to find the collision-free path,

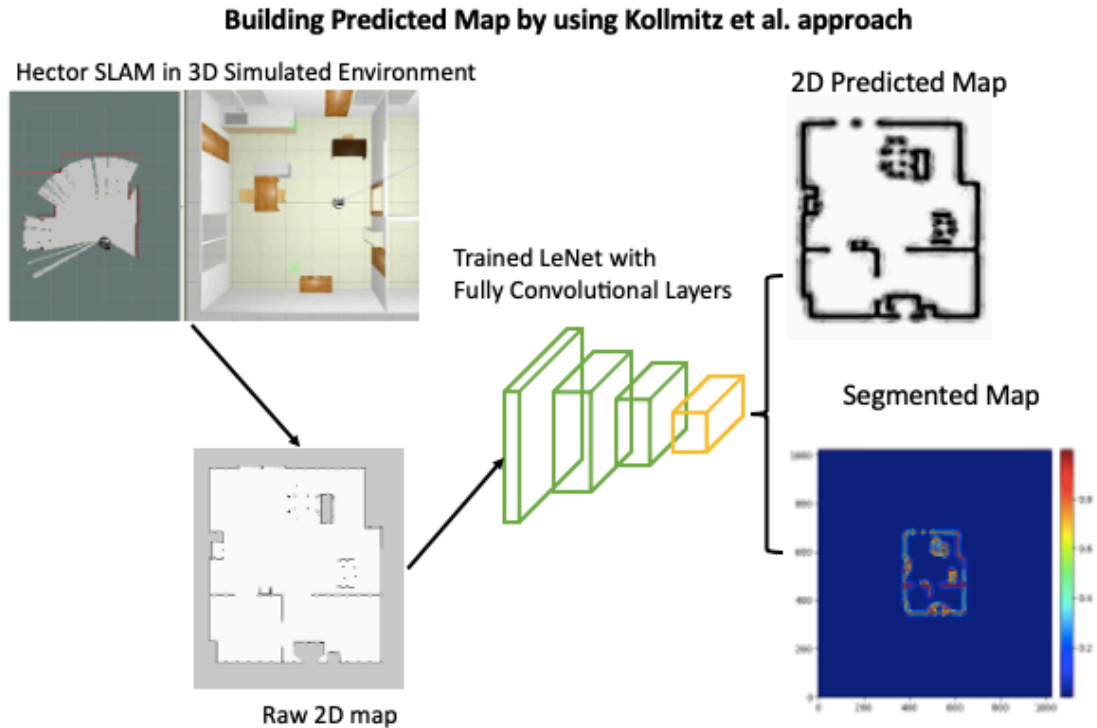


Figure 3.3: Initial step to build the predicted map using this method

combining the ROS navigation stack with this work [7] is essential to perform obstacle avoidance more efficiently using 2D LiDAR only. However, to apply and utilize this method in different types of robots such as Toyota HSR, we should conduct further investigation and evaluation to measure the effectiveness and limitation with different kind of obstacles and existing global and local planners.

We make the following contributions:

- We present an investigation and comparison study for possible solutions to the unseen obstacle problem in the service robot (Toyota HSR) navigation.
- We compare obstacle prediction network results with a raw map and sensor fusion (obstacle data acquired through the combination of the RGB-D camera and 2D LiDAR) and various conventional local planners in simulated environments to find its methods' upper limit and efficiency.

Since the DRL local planner uses the recorded map to avoid static obstacles, this investigation can be considered as a future work application on the learning-based local planner. The paper is arranged as follows; Section II explains the related work of the investigation method. Section III describes the proposed integration navigation

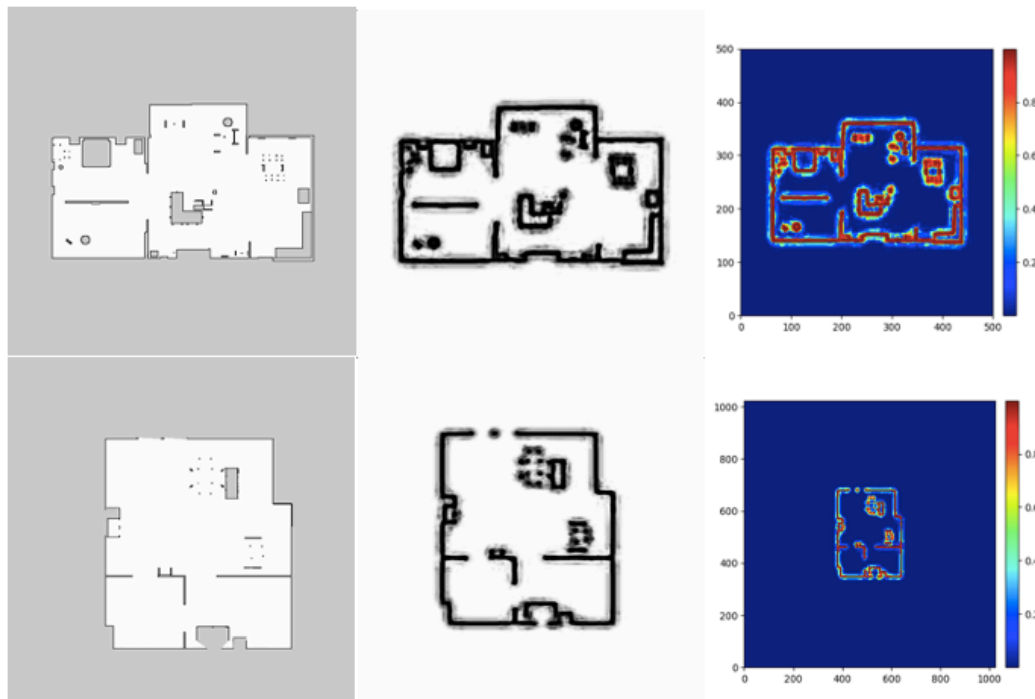


Figure 3.4: Raw map, predicted map and heat map from 3D environment of Small House and Large House.

system obstacle prediction network results. Section IV explains the experimental setup and defined parameters that we used in the evaluation. Section V shows the evaluation result and discussion. Finally, conclusions and future work suggestions are presented in Section VI.

3.2 Integration System between Obstacle Prediction Network and ROS Navigation Stack

In this section, we explain each step to investigate and evaluate the recent result of obstacle prediction networks and how to integrate it for local navigation in a 3D simulated world using Toyota HSR. First, we will explain how to obtain the predicted map using this method [7]. Afterward, the predicted map can be used for the RNS system for navigation tasks. Robotic systems depend on accurate information about the environments they interact, particularly during navigation. In the previous works [45], [50], [52], [70], the recorded 2D map is used for DRL agents and the traditional local planner methods to learn or perform the local navigation, avoid dynamic and static obstacles, and find the best path to reach the goal. The recorded map (raw map), such as in Figure 3. 2, generally can be obtained by using Grid mapping [71], Hector-SLAM

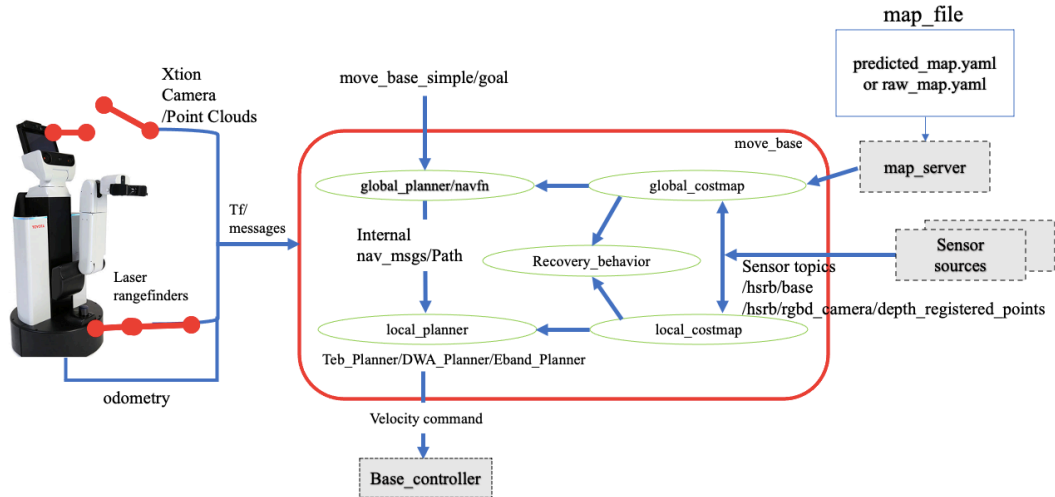


Figure 3.5: General ROS Navigation Stack (RNS) design for investigating the effectiveness of predicted map based on sensor input from HSR sensor.

[72] or Google Cartographer [73] to record the map from a real-world environment or 3D realistic simulation. First, we use the Kollmitz et al. trained architectures [7] to predict the unseen obstacle in our 2D recorded map that we collected from Hector SLAM by teleoperating the robot. The 2D map prediction is used to perform the navigation task. Figure 3.3 illustrates the process of how we obtained the predicted map by using Kollmitz et al. architectures. The authors use the LeNet architectures with a fully convolutional model to get shorter training process time and better accuracy. This model's dataset uses SceneNet [74], consisting of a 3D environment with 52 indoor scenes with various room layouts and furniture. The authors also used the simulated version of the service robot equipped with a sensitive force sensor that records the collision examples. The authors drove the robot to a random position in a simulated environment until they received the collision signal. Then collision data from the sensor is saved as the dataset. They divided the dataset into 34 rooms for training, 9 rooms for testing, and 9 rooms for validation.

To get the predicted 2D map, we inserted the raw maps that we obtained from Hector-SLAM into a pre-trained neural network for collision prediction and segmentation. The segmentation result of a small house (SM) and large house (LH) can be seen in Figure 3.4. The predicted map indicates the table legs and chair legs, and free space among them can be segmented as obstacle marks by using the trained networks in both map. The heat maps result shows the data visualization representing the magnitude of a free-collision and collision area as colors in two dimensions. In Figure 3.4, the raw map on the left side detects that the table and chair legs look like a

Tabel 3.1: Stages of testing configuration

Global planner /Nafvn			
TEB Planner/DWA Planner/Eband Planner			
	Sensor	Map	Environment
Stage 1	laser Only	raw	
	laser + RGBD	raw	SH
	laser Only	Predicted map	
Stage 2	laser Only	raw	
	laser + RGBD	raw	LH
	laser Only	Predicted map	

tiny dot in the 2D occupancy map. On the other hand, the predicted map in the center visualized that the several small dots around the table are merged, becoming a new obstacle footprint. For more explanation about the details of the network, architectures can be read in the original paper. To evaluate the effectiveness of the predicted map in local navigation, we designed the integration system based on RNS such as Figure 3.5. A brief explanation of nodes from the integration system is as follows:

- **Global planner:** NavFn is a global planner node in RNS that drives on a 2D occupancy global costmap by using the Dijkstra strategy to achieve the goal point using the global map. This path message uses general type navigation msgs/Path.msg, which includes the waypoints without the orientation.
- **Odometry:** In this system, we used wheel odometry which is used to create the ego-motion measurements from the starting point of the vehicle.
- **Global and local costmap:** The local and global costmap nodes represent the robot of the vehicle to an obstacle using the robot geometry [34]. The 2D map has the obstacles inflated by a safety boundary where the robot should not allow entering. Moreover, this map is used by local and global planners to adjust the path based on the distance from the robot to the obstacles. We use the very small value of the inflated layer in each costmap to measure the effectiveness of the predicted map.

- **Input/Sensor Usage:** The integration of RGB-D camera and 2D LiDAR with the raw map will be compared to 2D LiDAR only with predicted maps. We also provide the experiment of the navigation with 2D LiDAR only and with a raw map to know the robot is difficult to avoid the table, chair, and sports equipment such as in SH and LH 3D environments.
- **Transformation messages (tf) :** tf message is a robot transform message in the ROS package which manages the position and orientation between various sensors connected to an HSR. In this case, the reference point of Odometry is at the center of the robot base. So, a TF between the Odometry (Odom), robot mobile base (base_link), and X-Tion camera and 2D LiDAR is established using TF library in ROS.
- **Recovery behavior:** this node provides the simple recovery motion to clear the space in the costmaps by rotating the robot 360 degrees.

3.3 Kinematics and Local Planners Formulation

This section discusses about HSR service robot kinematics and the applications of velocity-based local planner that calculates the optimal collision-free both in 2D raw map and in predicted map. In HSR, a dual-wheel caster drive mechanism is used for mobility [75]. The scheme of this mechanism is described in Figure 3.6. J is denoted as Jacobian form in Equation 3.1 based on radius of the wheel r and the angle from the pivot axis that connected to the top table is θ_h . Moreover, w is denoted as the treat of the wheels and the distance from the axle center to the center axis is denoted by h .

$$J = \begin{bmatrix} \frac{r}{2} \cos \theta_h - \frac{r_h}{w} \sin \theta_h & \frac{r}{2} \cos \theta_h + \frac{r_h}{w} \sin \theta_h & 0 \\ \frac{r}{2} \sin \theta_h + \frac{r_h}{w} \cos \theta_h & \frac{r}{2} \sin \theta_h - \frac{r_h}{w} \cos \theta_h & 0 \\ \frac{r}{w} & -\frac{r}{w} & 1 \end{bmatrix} \quad 3.1$$

The velocity of the right and left wheels, as well as the rotation axis $\omega_R, \omega_L, \omega_h$ is calculated using Equation 3.1 from the velocity of the body v_x, v_y, v_θ as shown in Equation 3.2.

$$\begin{bmatrix} \omega_R \\ \omega_L \\ \omega_h \end{bmatrix} = J^{-1} \begin{bmatrix} v_x \\ v_y \\ v_\theta \end{bmatrix} = [v, w] \quad 3.2$$

This mechanism is a holonomic one since J maintains its full rank regardless of the condition it is in. As a result, it can generate speed in any direction.

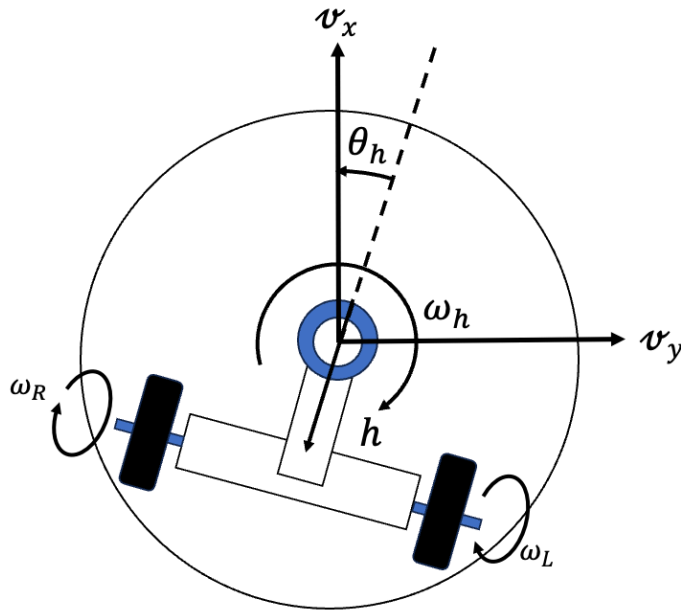


Figure 3.6: HSR base drive mechanism

Dynamic Window Approach (DWA) Planner: DWA[41] is a velocity-based local planner that determines the best collision-free ('admissible') velocity for a mobile service robot to achieve its goal. It converts a desired goal x_g, y_g, θ_g into a velocity command v_x, v_y, v_θ for HSR. The key objectives are to compute a valid velocity search space and to determine the best velocity. The search space is built from the set of velocities that result in a safe trajectory such as enabling the robot to stop before colliding, given the set of velocities that the robot can attain in the next time slice given its dynamics ('dynamic window'). The ideal velocity is used to optimize the robot's clearance, velocity, and heading closest to the desired goal. It is simpler to explain using the following pseudo-code:

Algorithm 1 Dynamic Window Approach Algorithm

```
1: Start DWA (robotPose( $x_0, y_0, \theta_0$ )), robotGoal ( $x_g, y_g, \theta_g$ )
2:   DesiredVelocity( $v_x, v_y, v_\theta$ ) = calculateV(robotPose,robotGoal)
3:   laserscan = readScanner()
4:   allowable_v = generateWindow(robotV, robotModel)
5:   allowable_w = generateWindow(robotW, robotModel)
6:   for each v in allowable_v ( $v_x, v_y$ )
7:     for each w in allowable_w( $v_\theta$ )
8:       dist = find_dist(v,w,laserscan,robotModel)
9:       breakDist = calculateBreakingDistance(v)
10:      if (dist > breakDist)
11:        heading = hDiff(robotPose,goalPose, v,w)
12:        clearance = (dist-breakDist)/(dmax - breakDist)
13:        cost=costFunction(heading,clearance,abs
14:          (desired_v( $x_g, y_g, \theta_g$ )- v( $x_0, y_0, \theta_0$ )))
15:        if (cost > optimal)
16:          best_v = v
17:          best_w = w
18:          optimal = cost
19:      set robot trajectory to best_v, best_w
20:   END
```

Elastic Band Planner: Elastic Band is a type of obstacle avoidance method first proposed by Quinlan and Khatib [76] for use in robotics. The advantage of this method is that the navigation behavior of the mobile robot is not fully determined at the planning level, but local navigation behavior does not limit the performance required to reach the desired position. By deforming the path when changes in the environment are detected, this method avoid the cost of recalling the path planner; the robot can respond in real-time to data obtained from 2D LiDAR. Nevertheless, while performing local navigation behaviors, the method can maintain a collision-free path to the desired goal. An elastic band is visualized by a finite series of bubbles constructed from configurations or via points for the mobile robot. Figure 2.9 describes the elastic band planner in static and dynamic obstacles. This method assesses the condition that the bubbles at consecutive via points overlap to ensure that a collision-free path can be generated between the via points. If the path remains inside the bubbles, it will be collision-free. Obviously, a straight line between the via points will satisfy this

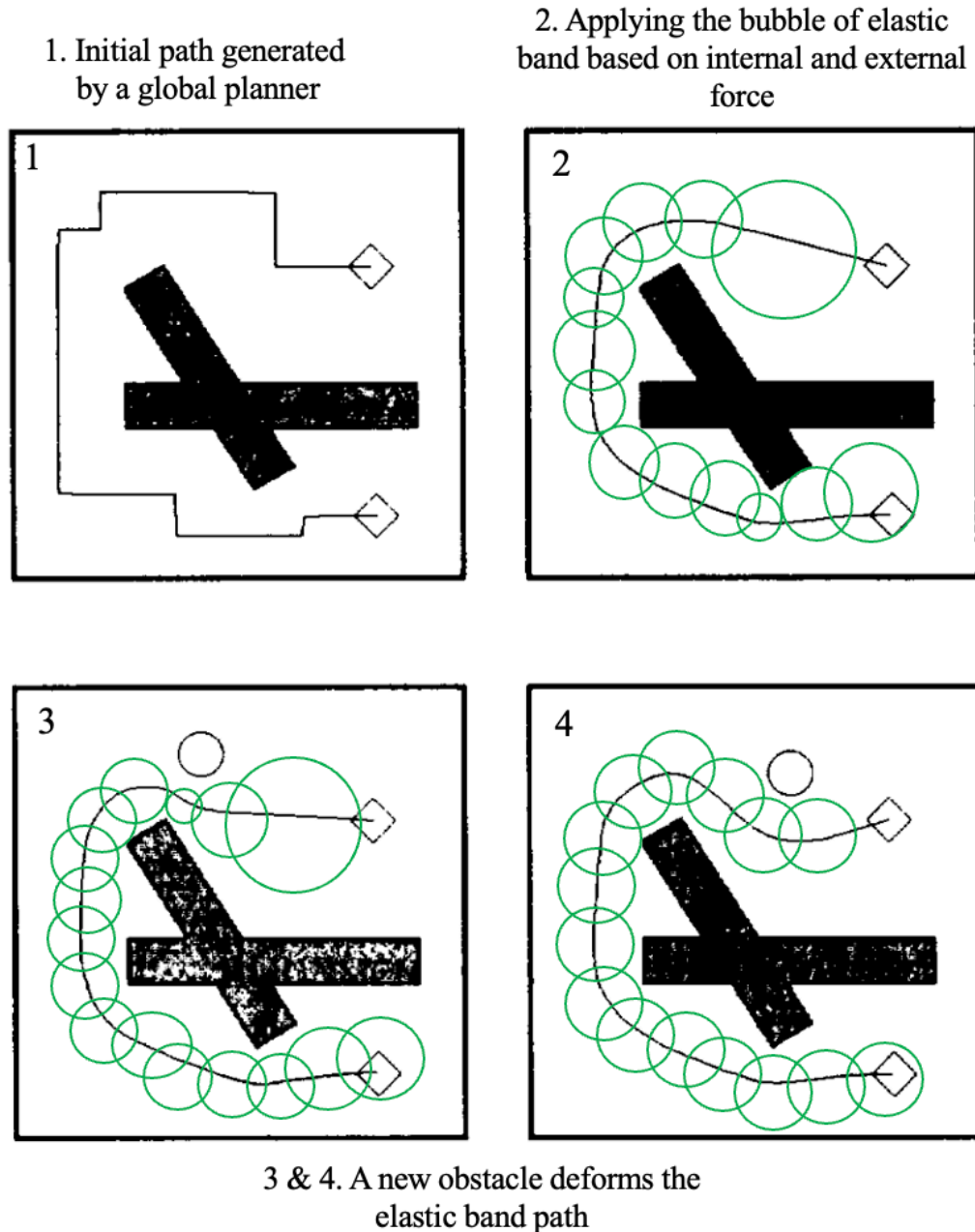


Figure 3.7: Elastic Band planner performance in static and dynamic obstacles

requirement for the circular bubbles. However, by selecting more complex curves, this method can significantly enhance the path from a control point of view. It is simpler to explain using the following pseudo-code:

Algorithm 2 Elastic Band Algorithm

```
1: eb[0] = path.start
2: i = 0
3: for each point along path
4:   if dist(eb[i],point)> c
5:     eb.add(point)
6:     while robot not at path.goal
7:       for each bubble b in eb
8:         f_e = b.pos - closestObstacle(b)
9:         f_i = (b.pos - eb[b.i - 1].pos) + (b.pos - eb[b.i + 1].pos)
10:        b.integrate(f_e+f_i, b.vel) //update velocity from forces
11:        b.dampen(b.vel) //filter state
12:        b.integrate(b.vel, b.pos)
13:        dist = abs(b.pos - eb[b.i + 1].pos)
14:        if dist > c
15:          eb.add(midpoint(b, eb[b.i+1]))
16:        if dist < c_min
17:          eb.remove(b)
18:        tmp_goal = eb.closestbubble(path.robot + lookahead) //set goal from
        robot position
```

Timed-Elastic Band (TEB) Planner: TEB planner is an improved Elastic band planner for locally optimizing the robot's trajectory with respect to trajectory execution time, separation from obstacles, and compliance with kinodynamic constraints at runtime. The TEB planner optimizes robot trajectories by subsequent modification of an initial trajectory generated by a global planner. The objectives considered in the trajectory optimization include the overall path length, trajectory execution time, separation from obstacles, passing through intermediate waypoints, compliance with the robot's dynamic, kinematic, and geometric constraints, and dynamic obstacles. It also allows efficient online motion planning of robots. Figure 3.8 shows the general robot system with TEB planner.

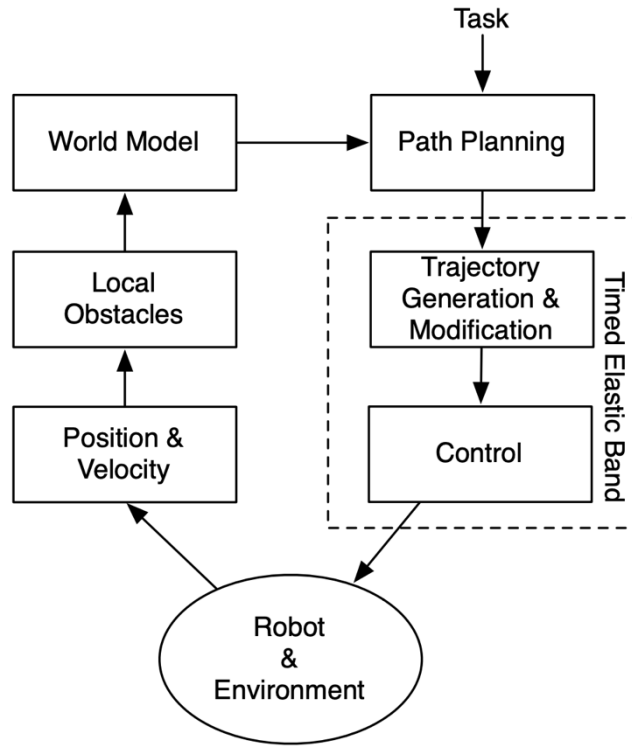


Figure 3.8: General TEB planner architectures for robot navigation system

3.4 Experiment Setup

To test and evaluate the approach from [7], we formulate a simple test where the robot navigates from the initial point to several checkpoints and a final goal point. Fig. 6 describes the robot navigation point in each environment. We designed the checkpoints based on the unseen obstacle position in the 3D environments to evaluate how the robot performance avoids the unseen obstacle using different combinations of sensors, maps, and local planners.

The testing scenario is divided into 2 stages such as in Table 3.1. In each scenario, several combinations are configured in the RNS system and 3D environment to investigate and analyze the performance of robot navigation with different types of unseen obstacles. The navigation parameters in RNS are defined in Table 3.2. We used small values on inflation radius to evaluate the effect of the 2D predicted map and 2D LiDAR only, navigation with sensor fusion and raw map and raw map with 2D LiDAR only in the experiments. The inflation radius is a safety distance parameter around the obstacles. Suppose the value of the inflation radius is more than 0.1. In that case, the

2D LiDAR can neglect the narrow free space between the table and chair legs. Also, sometimes robot is difficult to enter a narrow space.

Tabel 3.2: Navigation parameters in RNS

Costmap Common Parameters	
Parameters	Values
Radius	0.25
Laser_scan_layer :	
Topic:	hsrb/base_scan
Clearing	True
Obstacle_range	2.0
Raytrace_range	3.0
RGBD__layer :	
Topic:	/head_rgbd_sensor/ depth_registered_poi nts
Clearing	True
Obstacle_range	6.0
Raytrace_range	10.0
inflation radius	0.1
cost scaling factor	10.0
Global costmap parameters	
update frequency	10
static map	True
Local costmap parameters	
update frequency	10
rolling window width	true
height resolution	0.025
publish frequency	1.0
static map	false

We evaluate the efficiency of local planners with a different type of 2D map by calculating the path length and the average time to reach the goal. In addition, we define the percentage of the success rate of each episode based on achieved checkpoints(cp). If the robot cannot reach a cp1 in less than 2 minutes, the navigation process to reach cp1 will terminate and attempt to the next cp.

We used a laptop with intel® Core TM i7-8750 CPU @ 2.20GHz processors with eight cores, 16GB RAM, NVIDIA GeForce RTX 2070 graphics card, and Ubuntu 18.04 LTS 64bits operation system. The Toyota HSR is used for the development of

this project [18]. It has an omnidirectional dual-wheel caster drive that can move in any direction on a 2D plane and sensors used for 2D and 3D cameras.

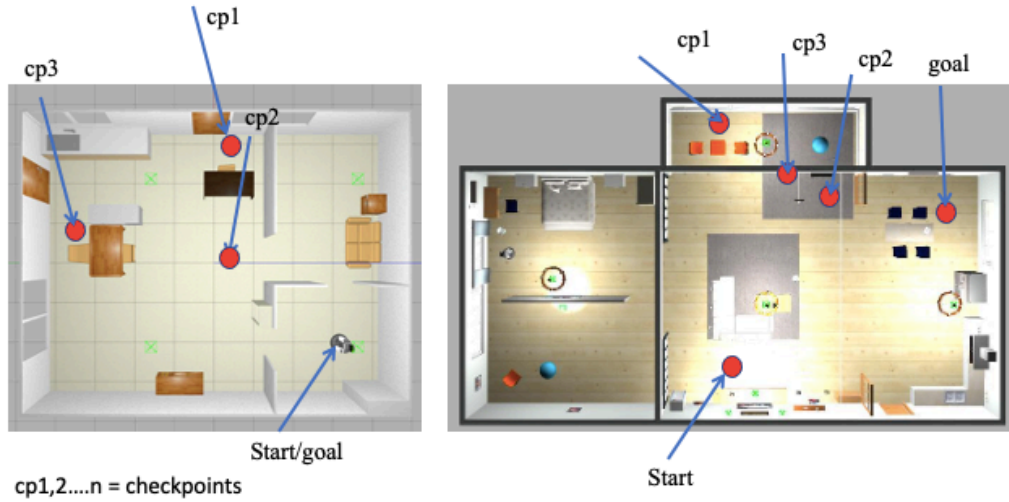


Figure 3.9: Start, checkpoints and goal for the robot navigation test in SH and LH

3.5 Result and Discussion

From 2 stages, 10 experiments were carried out for each local planner and map, environment, and sensor combination. 18 subsets of tests were conducted, totaling 180 navigation experiments in simulated environments. The primary purpose of these investigations is to evaluate the robot’s navigation behavior in unseen obstacles, maps, and environments using 2D LiDAR only and sensor fusion. The raw map and 2D LiDAR results indicate that the local planner could not generate a collision-free path when facing the table, chair, and sports equipment. On the other hand, the predicted map with 2D LiDAR only and sensor fusion with the raw map can still manage the difficult obstacle. In this experiment, we conduct different rules for this configuration.

Tabel 3.3: Investigation and comparison results between 2D map prediction with RNS and conventional method

	TEB + Laser				TEB + 2 Obs				DWA + Laser				DWA + 2 Obs				Eband + Laser				Eband + 2 Obs			
	Raw Map		Predicted Map		Raw Map		Raw Map		Predicted Map		Raw Map		Raw Map		Predicted Map		Raw Map		Raw Map					
Evaluation	SH	LH	SH	LH	SH	LH	SH	LH	SH	LH	SH	LH	SH	LH	SH	LH	SH	LH	SH	LH	SH	LH		
Average Time (minute)	1:00	1:00	2:41	4:21	2:39	4:34	1:00	1:00	3:41	4:16	4:01	4:20	1:00	1:00	3:11	4:52	3:10	4:57						
Average Path Length (m)	8.7	7.9	36.5	24.4	36.4	25.36	8.7	7.9	31.5	25.21	19.44	25.23	8.7	7.9	36.8	25.33	21.34	25.56						
Total Success Rate(%)	0	0	100	75	100	75	0	0	75	75	40	75	0	0	100	75	60	75						

2 Obs = RGB-D camera + 2D laser rangefinders

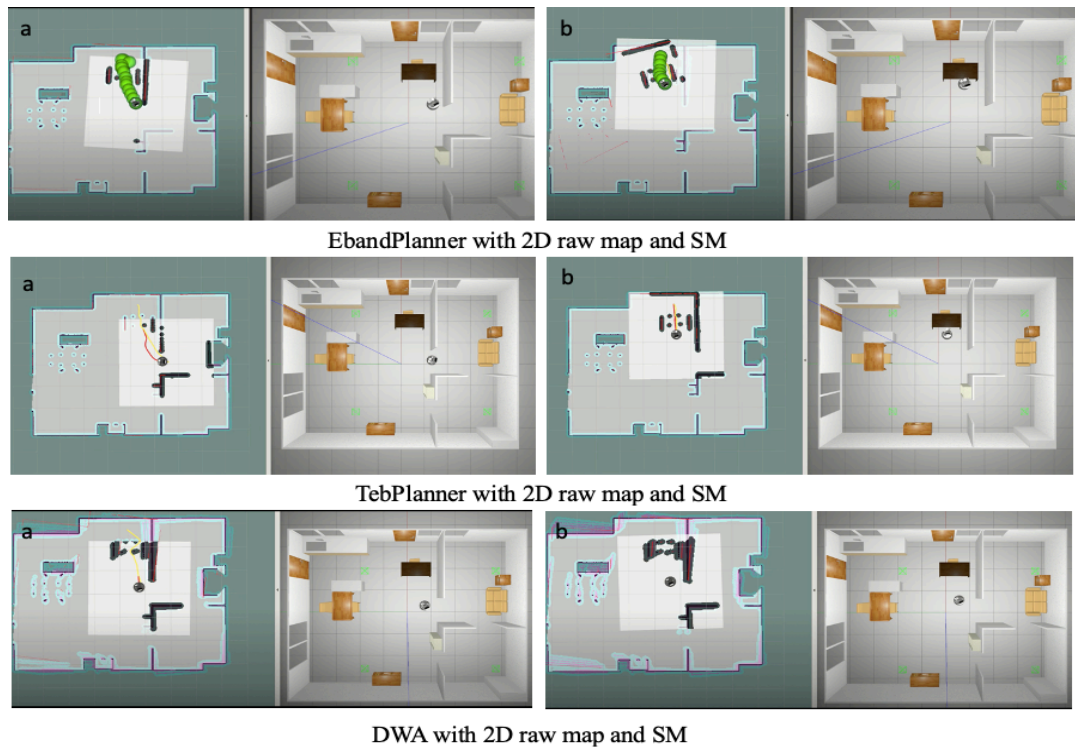


Figure 3.10: Example results using the raw map and conventional local planners. The robot cannot avoid by only using laser rangefinders.

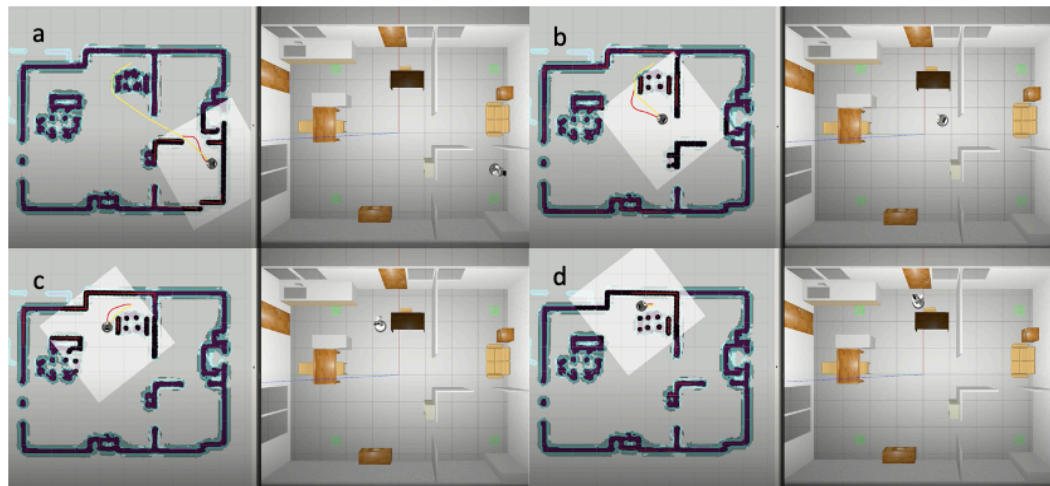


Figure 3.11: The example results of using predicted map. We used laser rangefinders only. The predicted map has significant role to improve the navigation performance in real-world obstacle condition.

We terminate the simulation when the robot hits an obstacle or is freezing for more than a minute. It is difficult for the local planners to pass the table and chair safely by using raw map configuration such as in the Figure 3.10. However, the trial-error parameter tuning in inflation layers can be used to find the appropriate inflation layer using the raw map and 2D LiDAR only. Nevertheless, this method is time-consuming and



Figure 3.12: The example results in failed navigation behavior using the predicted map and TEB Planners. The other planners also failed in this stage. The predicted map could not predict this type of chair. The leg part of the chair has a flat surface that is adhered to the floor.

tedious due to should be tried every time the map and environment change. Table 3. the presented the quantitative result from the navigation experiment. As we expected, all the local planners with laser and the 2D predicted map outperformed in each environment, TEB Planner and EBand Planner managed a 100% success rate in SH and 75% in LH. In the predicted map, the local planners could find the collision free path easily by only using the 2D LiDAR due to the unseen obstacle has already predicted by the obstacle prediction networks. When the robot faces the chair, table, and sports equipment, the tiny footprint represented in the raw map can be merged into the predicted map. Therefore, the free-collision area between tiny obstacles is become new obstacle footprint that make the local planner to generate safer collision free path. The example behavior of the robot using the predicted map is shown in Fig. 8. DWA planner with the predicted map only attains a 75% success rate in SH due to a freezing behavior when the robot approaches the *cp1* in SH but can safely avoid the obstacle in another unseen obstacle. It is due to the limitation of DWA Planner when facing the narrow space, as explained in [46]. All the local planners with a predicted map could not reach the maximum success rate in the LH environment. It is due to the mobile base colliding with the flat surface of the chair's leg, as shown in Figure 3.12. This type of chair is hard for the robot to avoid collision by only using the 2D LiDAR. Therefore, after the robot hits the chair, the robot will move to the following *cp* to finish the mission. Figure 3.13. is an example of the robot avoiding collision with sports equipment objects using the 2D LiDAR and RGB-D camera. When we used the conventional sensor fusion method, the detected free space between the wall and the table from the camera could be considered an alternative path for the robot.

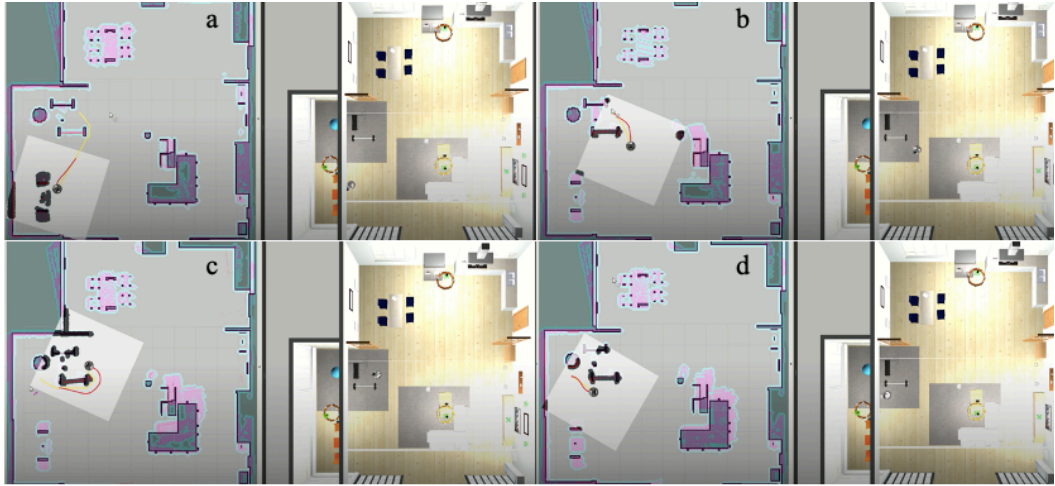


Figure 3.13: The example results from using 2 observation resources. We used X-Tion RGB-D camera and laser rangefinders. The robot can avoid a collision with sports equipment obstacles.

Nevertheless, the robot cannot pass the narrow space due to its size. Therefore, some local planners could not achieve the maximum success rate due to this limitation. TEB Planner achieves the best results among 3 conventional sensor fusion approaches with a success rate of 100 % in SH and 75% in LH. While EBand Planner has second best performance and reached a success rate of 60% in SH and 75% in LH. DWA Planner has only reached a 40% success rate in SH and 75% in LH. Overall, the path length for each local planner with the highest success rate did not significantly differ. It is because we used the static obstacle only.

3.6 Summary

This paper presented an investigation and comparative study of the approach to solving unseen or partially observed obstacles using only 2D LiDAR. We compare the result of Kollmitz et al. [7] with the conventional method to perform the navigation and avoid a collision from an object such as a table, chair, and sports equipment that is difficult for 2D LiDAR to fully observed. The conventional method uses sensor fusion, such as a combination of RGB-D camera and 2D LiDAR. All methods were evaluated using the RNS in the simulated 3D environment. All the local planners with 2D LiDAR and the 2D predicted map had achieved significant performance in each environment than only using the conventional method with raw map. Since the learning-based (DRL) local planner uses the recorded map to avoid static obstacles, the 2D predicted map could be considered a future work application for the learning-based local planner.

Chapter 4

Activity Recognition From 3D Object-Oriented Map

4.1 Introduction

The problem of a large elderly population and a low birth rate has become a serious social issue in several nations. Thus, developing a homecare robot to do work is being considered as a potential solution to the problem of the increase in aging societies across the globe. Many homecare robots are equipped with various sensors, such as RGB-D cameras, 2D LiDAR, stereo cameras, and arms, to understand a situation and carry out physical tasks in a dynamic environment. Furthermore, the Robot Operating System (ROS) is designed as a software system architecture to manage sensor inputs and outputs in many robotics applications. Many researchers are interested in developing a learning model for understanding the context of the environment from object detection and human behavior estimations [77]–[80]. This capability affects the level of interaction with the environment and the action that is performed. For instance, a homecare robot at home is expected to take the initiative to help in the daily household chores, such as performing tasks in the kitchen to prepare food or wash the dishes, etc. Then, the question arises: What information will the robot need to help humans in their daily activities? Some of that information is defined below:

- The robot must obtain information on the objects in the home environment, including the class, position, and orientation of the objects.
- The robot needs to know the position of the human and recognize the activity when the human interacts with several objects at home.

However, it is no easy task to obtain this high level of ability in homecare robots, such as understanding human activity from semantic information and human-object interaction (HOI). Many efforts have been made to recognize human activities using images and videos with 2D and 3D visual information [81]–[84]. Semantics and the context of a situation are usually used for classification, which involves typical HOI

[85]–[87]. However, most of their frameworks require massive, labelled datasets and much training time to achieve a high level of accuracy.

Recently, studies on the analysis of graphs using machine learning have been gaining more attention because of the outstanding expressive power of graphs. For instance, graphs can be used to denote a large number of applications across different scopes, including the extraction of topologies and geometries from 3D object detection or point cloud [88]–[90], natural science problems [91], [92], pharmaceutical research [93], [94] and other areas [95]. Graph computation, which is a unique form of data structure for supervised and unsupervised learning strategies, focuses on tasks, such as clustering, link prediction, and classification. Graph neural networks (GNNs) are deep learning-based approaches in the graph domain. Due to their effective performance, GNNs have lately become a widely used strategy for the analysis of graphs. Therefore, formulating these interactions into a graph representation based on the detected object and human features will be of significant help to the robot in learning human activities and deciding on the appropriate tasks.

In this paper, an efficient learning method was proposed to predict daily human activities in an indoor environment. The approach used a GNN, which was trained based on activities recorded directly from the Toyota HSR (Human Support Robot). The activities were labelled directly from human speech. The real-time YOLOv3 object detection [96] and MediaPipe [97] face detection frameworks were implemented using an Xtion RGB-D camera from the HSR head to obtain the object class, geometry, and human face position in a real indoor environment. Human activity is a huge topic in robotics. Therefore, the aim of this paper is to create a method to learn and predict three specific activities, namely, eating, reading, and working, from utterances. With this method, the robot can be trained to predict daily human activities in real-world applications using ROS environment. A human activity dataset recorded directly from a human-robot communication (HRC) system was also introduced. To summarize, the contributions of this work are as follows:

- A GNN model was proposed that utilized the prior information from HOI and HRC to classify human activities, especially eating, reading, and working, in a very similar environment.

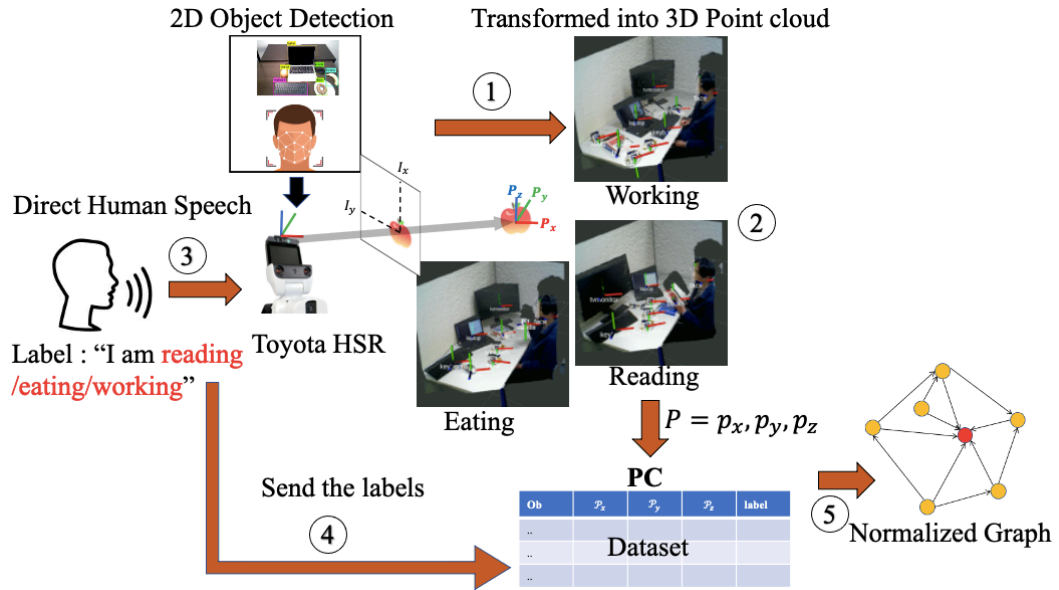


Figure 4.1: Data collection system of proposed method and graph data representation

- The proposed method can efficiently construct the labelled dataset using human-robot communication. When the robot detects a human doing an activity with some objects, the center of the detected 2D bounding box object is transformed into a 3D position. Then, the human can ask the robot to collect the dataset of the current activity and train the teaching data using utterances.
- The method was applied using a Toyota HSR robot with an ROS environment system in a real-world indoor environment. The proposed method can tackle the problem of activity recognition in a very similar environment, which is difficult to classify using conventional 2D CNN.

The rest of the paper is arranged as follows. Section 4.2 discusses the method and explains the related works and the GNN strategy, along with explanations of the network configuration and various features. Following this, section 3 presents an evaluation of the approach in a real-world indoor environment, as well as a comparison with multiple dataset combinations. The conclusions are given in section 4.

4.2 Human Activity Recognition

Human activity recognition (HAR) is one of the most difficult tasks in robotics and computer vision since it requires assigning a label to each activity. HAR can be divided into three types: (1) by vision, (2) by sensors, and (3) by waves/radio. (1) Vision

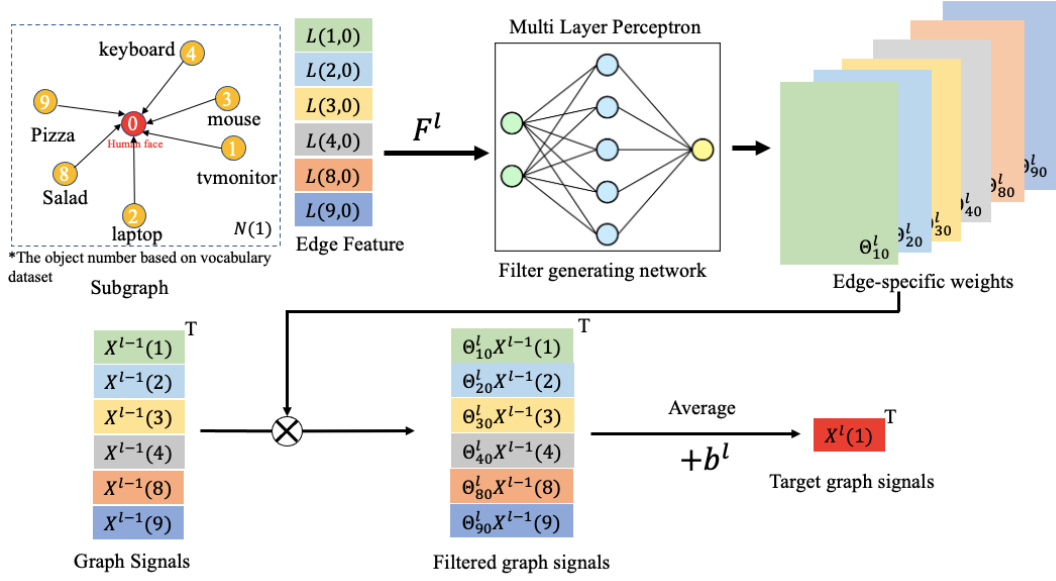


Figure 4.2: Processing a subgraph in edge conditional convolution method

typically refers to a camera that utilizes red, green, and blue (RGB) colours from a video to identify a human activity [4], [22]. Nevertheless, RGB is just a 2D vision, and does not indicate how far the object is. Therefore, a depth camera was added to the vision to make it RGB-D [23], [24]. By having depth, the 3D position of an object or human can be efficiently recognized, and the HAR can be realized using the 3D pose and visualization [10]. (2) Sensors refer to devices that do not use RGB-D data [25], such as mobile phone accelerometers and gyroscopes. The data is numeric and reliable, but the drawback of this method lies in the coverage issue, where the individual is required to always carry the device. (3) Finally, waves/radio is the latest technology that utilizes Wi-Fi signals to predict human activity [26]. The recognition coverage is more extensive and there are no problems with privacy. However, this method is still new and requires more extensive research to make it more reliable.

4.3 Human-Object Interactions to Recognize the Human Activity

Recognizing human activities using graph convolutional neural networks has attracted the attention of many robotics and computer vision researchers in recent years. Activity recognition is required for homecare robots that are taking care of children, the elderly, or persons with disabilities. With this implementation, the inference of

human activities using perceptual information plays an important role in human-robot interactions, smart surveillance, and content-based video analysis. A lot of work has been conducted toward predicting human activities in 2D and 3D images and videos where the overall technique observes and correlates with HOI. [57]–[60]. The principal method for predicting HOI is extracting visual characteristics from instance detectors and spatial knowledge to instantiate multi-streams of deep neural networks. Each stream includes detected human and contextual objects. The last step is designed for inference application. The work of [57] presented the forecasting of a human activity by predicting the possible trajectory movement towards a targeted object from RGB and depth sensors data. Using this strategy, the predicted trajectory for the ongoing action can be visualized. However, the camera should be set up at a certain distance and height to avoid the broader field of view that may lead to occlusions and poor activity prediction. Wang *et al.* [61] proposed a fully-convolutional approach that predicts the interactions between human-object pairs from RGB images. The network can predict the context of human activities by localizing the interaction points from the object, human and pairwise streams.

GNN has been utilized to predict human activities from HOI by extracting the image features into graph structures. Qi et al. [58] used a graph parsing neural network to detect HOI and predict human activity from various RGB datasets. Morais et al. [59] introduced asynchronous-sparse interaction graph networks, which are constructed from the temporal structure and content label of human-object interaction activities. This method uses a graph attention network model for HOI detection in the RGB dataset. Their approach involves the construction of nodes and edges from visual features. An adjacency matrix defines the structure and properties of the networks and is updated by a weighted sum of the messages from the other nodes. Finally, for interaction inference, a node readout function is employed. Simonovsky and Komodakis [62] proposed the edge-conditioned convolution (ECC) GNN, a spatial domain operation on graph signals in which filter weights are conditioned on edge labels and dynamically formed for each input sample. It was demonstrated that this strategy could generalize the traditional convolution on graphs if edge labels are suitably chosen, and this claim was empirically tested on MNIST. Moreover, this method was also evaluated for point cloud classification, achieving a new state-of-the-art performance on the Sydney dataset. The current work was inspired by [62], and

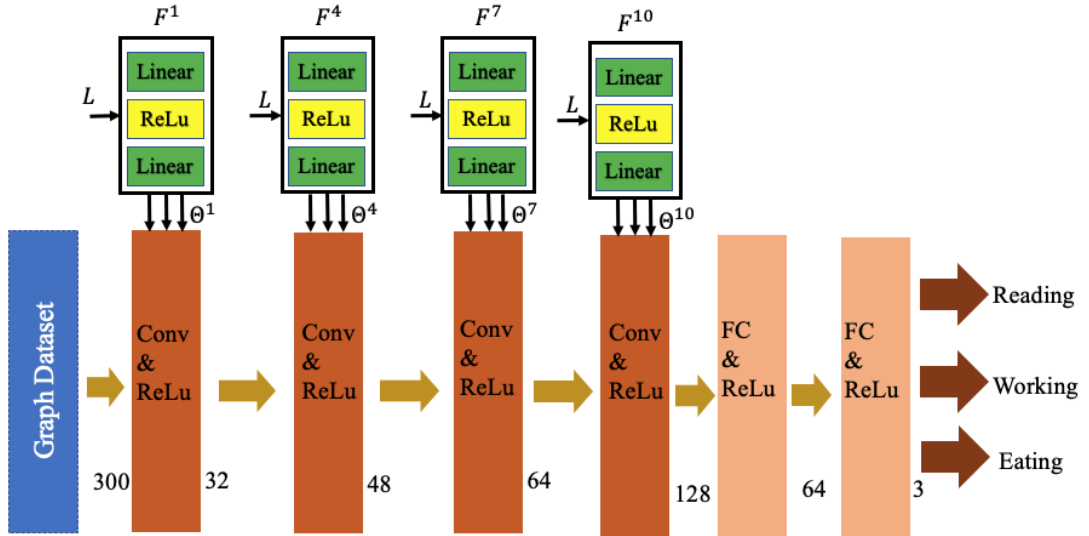


Figure 4.3: The proposed GNN with ECC for the activity recognition

their model was used for the proposed method. A 3D object and human pose of point clouds data were used to construct the edge features for the graph data by transforming the center of the 2D bounding boxes of the YOLO and Mediapipe face detection frameworks into the 3D point cloud by utilizing the ROS library features. The data was labelled using human speech when the robot questioned the human to learn the HOI. In the application for this study, a 3D object and human pose data were used to construct the graph data by transforming the center of the 2D bounding boxes into a 3D point cloud by utilizing the ROS library features. The data was labelled using human utterances in a human-robot communication scenario for the data collection. The data was collected during the ongoing activity, and the robot asked the human, "What are you doing now?". Then, the uttered answer would be specified as an activity label. After collecting the data, a graphical representation was constructed for the training process.

4.4 Data Preparation

This section explains the approach for predicting specified human activities at home from a homecare robot system in real-world applications. The dataset was pre-processed and collected from the YOLO and MediaPipe face detection frameworks, and each detected object and activity were classified using direct communication with the Toyota HSR robot. The activity labels were specified into 3 categories: eating,

reading, and working. Next, a graph convolutional neural network proposed by [62] was used. The data collection process for the approach is shown in Fig. 1, where the dataset from the YOLO object detection and MediaPipe face detection frameworks was collected using an Xtion RGB-D camera from the Toyota HSR head.

To obtain the 3D position of each detected object and a human face, we transform the center of the bounding boxes of each detected object and a human face by using the point cloud data type service in the ROS environment and transform it into 3D real-world position by using tf2 of ROS library [98]. Afterward, the object and human position can be constructed in a graph representation. During the labeling process, our robot only has a Japanese voice system. Therefore, we conduct the human-robot communication for label collection in Japanese such as in this video [https://youtu.be/WI8v1_UJwCc]. We used Google Cloud Speech-to-Text frameworks [URL : <https://cloud.google.com/speech-to-text>] to recognize the human voice and convert it to text. We stored the collected dataset on the server PC for graph construction and training.

4.5 Graph Construction and Features

In this paper, we represented the HOI containing the relationships between human positions and the surrounding objects as a graph form $G = (V, E)$, where V is the set of $|V| = N$, and N denoted as the centroid of 3D detected object $p \in P$. Meanwhile, E is a set of edges with $|E| = M$, where M is the number of edges. Then, we specify each detected object to graph signal by $X^0(i) = X_p(p)$. Then, we connect each node i to all nodes j by directed edge (j, i) . In our works, we specify an edge from j to i by calculating the distance of each number of centroids from the detected object by $d = P_j - P_i$ where d represented in cartesian coordinates as 3D edge label vector $L(j, i) = (d_x, d_y, d_z)$ and $L: E \rightarrow \mathbb{R}^{M \times C}$ is feature matrix with features number of each edges C . Afterwards, the node embedding is constructed from the name of the YOLOv3 object class by creating the vocabulary dataset, which encodes the object names with their IDs. The ID is an integer (index) that identifies a word's position in the vocabulary dataset such as shown in the subgraph in Fig. 2. The vocabulary dataset to construct the node embedding is arranged as follows:

{"face": 0, "tvmonitor": 1, "laptop": 2, "mouse": 3, "keyboard": 4, "book": 5, "soup": 6, "sandwich": 7, "salad": 8, "pizza": 9, "cup": 10}

4.6 Graph Neural Network with ECC Graph Neural Network with Edge-Conditioned Convolution



A GNN can solve many complex problems, especially for activity classifications [60], social network recommendations [95], and predicting chemical and molecular properties [62]. In this paper, a GNN was used with the ECC by [62] to process the normalized graph structure data. This approach was used [62] to perform the graph classification to predict human activities, such as eating, working, and reading. The entire architecture was made up of 4 ECC layers and 2 fully-connected (FC) layers. The 4 ECC layers were used to aggregate the information from each of the human interactions and objects, while the 2 FC layers were used to perform the final recognition. There are three aggregations (or message-passing) approaches in the ECC, which are "add", "mean", and "max", where "add" is to calculate the total neighbours' features, "mean" is to calculate the average, and "max" is to use the maximum features during the message-passing. In this approach, the "mean" for all the ECC layers was used to obtain stable features. The overview of the GNN architecture is shown in Fig. 3. Moreover, the Cross-Entropy Loss function was utilized in the proposed GNN, as well as the Pytorch Geometric package [99], a python library dedicated to the GNN. The graph neural network can be described in the following form [62], [100] as:

$$\begin{aligned} X^l(i) &= \frac{1}{|N(i)|} \sum_{j \in N(i)} F^l(L_{ji}; w^l) X^{l-1}(j) + b^l \\ &= \frac{1}{|N(i)|} \sum_{j \in N(i)} \Theta_{ji}^l X^{l-1}(j) + b^l \end{aligned} \quad (4.1)$$

where $X^l(i)$ denotes the node embedding corresponding to the i -th vertex in layer l , and $|N(i)|$ is the total number of neighborhood node of i -th node. Each layer l includes a multi-layer perceptron as a filter generating network F^l with learnable w^l and bias b^l that implements aggregation between nodes i and j with edge

Table 4.1: Scenarios of the data collection

Activities		laptop	tvmonitor	keyboard	mouse	book	salad	sandwich	soup
Working	A	Object Always use	Object sometimes use	Object sometimes use	Object sometimes use	Object sometimes use			
	B	Object Always use	Object Always use	Object Always use	Object Always use	Object sometimes use			
Eating	A	Object sometimes use	Object sometimes use			Object sometimes use	Object Always use	Object sometimes use	Object sometimes use
	B	Object sometimes use	Object sometimes use				Object Always use	Object Always use	Object Always use
Reading	A					Object Always use			Object sometimes use
	B	Object Always use		Object sometimes use	Object sometimes use	Object Always use			

 Object Always use
 Object sometimes use

embeddings L_{ji} . The computed $X^l(i)$ are used to train two FC networks for final prediction. We illustrated the ECC and the overall architecture of the proposed method in Fig. 2 and Fig. 3 respectively.

4.7 Positional Data HOI and Scenarios

To decrease the overfitting of small dataset, we collect the dataset by makes the data slightly different from one to another group of class of data. During the dataset collection, we specified 2 scenarios (A and B) for each activity and asked the subject to follow the scenario A and B in each specified task such as eating, reading, and working. To describe more detail about this scenarios, Table 1. describes the scenarios of human-object interaction for data collection. We decided to use 8 objects for 3 activities. We collect the $P_0 = [p_x, p_y, p_z]$ from the human face and object position during human interaction with an object. When the data reached 1000 for each A and B scenario, the utterance was used to label the collected dataset. The human utterance is converted into text using Google speech-to-text [URL : <https://cloud.google.com/speech-to-text>]. This automatic speech recognition can accurately convert more than 125 languages. In total, there are 6000 data to train the GNN. Afterward, we use the collected dataset to transform it into graph data

representation. We used 80% of the dataset for training the GNN, while 20% was for testing the networks.

4.8 Comparison Method

In this paper, the proposed method for the recognition of eating, reading, and working in a similar environment was compared with the DLA [101] and X3D networks [102] approach using different datasets. These two architectures commonly use an image or video dataset to make an image or video classification. Therefore, only 6 videos were collected for each specified activity, where 4 videos were for training and 2 for testing. The video dataset was conducted with a slightly different scenario of object and environment, such as described in Fig. 4. Moreover, the video was labelled manually in these methods without a speech-to-text recognition framework.



Figure 4.4: Video dataset of eating, reading and working to train and test DLA and X3D networks as comparison methods.

4.9 Experimental Results

This section will discuss the experimental setup and several results. The object and human face detection was carried out using MediaPipe and YOLOv3. The centre of the object and face detection was converted into the 3D position using the ROS tf library, as shown in Fig. 5. The subjects were asked to use several objects to perform the specified tasks, such as eating, reading, and working. The Toyota HSR would ask

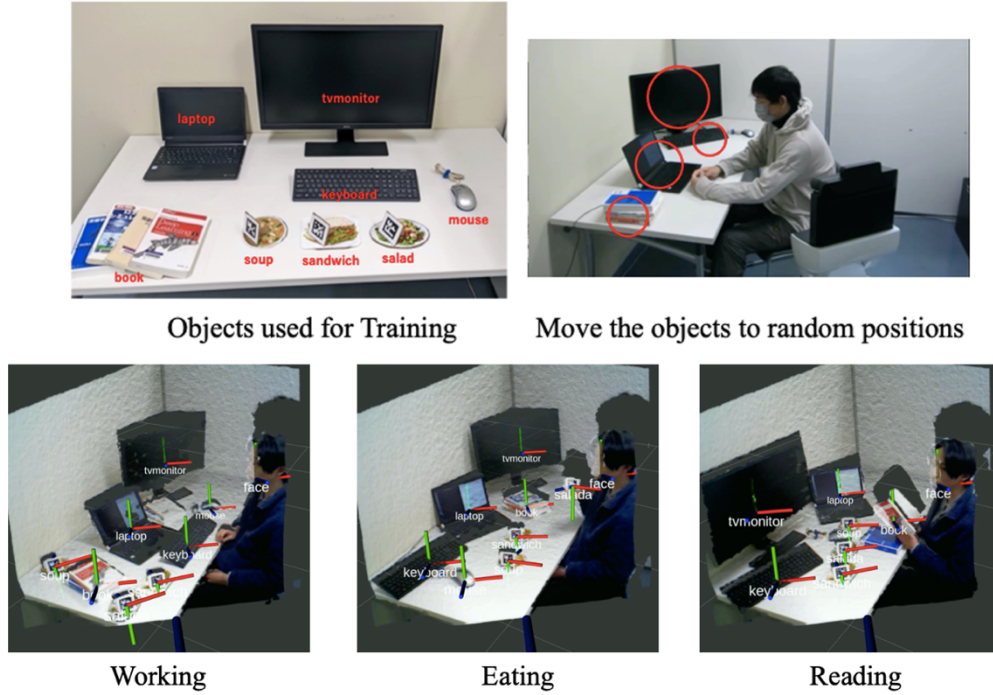


Figure 4.5: Experimental setup, collecting the dataset and visualizing the centre of the object into 3D Position.

what kind of activity was being done, and the subject would answer by speech. The robot would collect the detected object and face position with the utterance information. To visualize the brief experimental setup and the results in greater detail, the experimental example, data collection and inferencing were uploaded to this URL: [https://youtu.be/WI8v1_UJwCc]. In the proposed method, 100 epochs were used to train 4000 positional datasets, and the proposed architecture was tested with 2000 positional graph datasets. However, by using DLA and X3D, 50 epochs were used to train the collected videos. Generally, video classification frameworks are trained with huge datasets with a high number of epochs, such as by using UCF101[103] and kinetics dataset [104]. However, this study had its limitations in terms of the dataset collection and computational cost.

Table 4.2: Comparison results between the proposed method, DLA and X3D

Network Architecture	Total Training Data	Total Testing Data	Epoch	Training Time(seconds)	Training Loss	Avg Training Accuracy (%)	Avg Testing Accuracy (%)
GNN with ECC(Ours)	4000 positional objects	2000	100	100.346	0.0145	99.789	91.26
DLA	4 videos	2 Videos	50	4030.784	0.306	99.96	22.67
X3D	4 videos	2 Videos	50	129.456	7.187	91.00	64

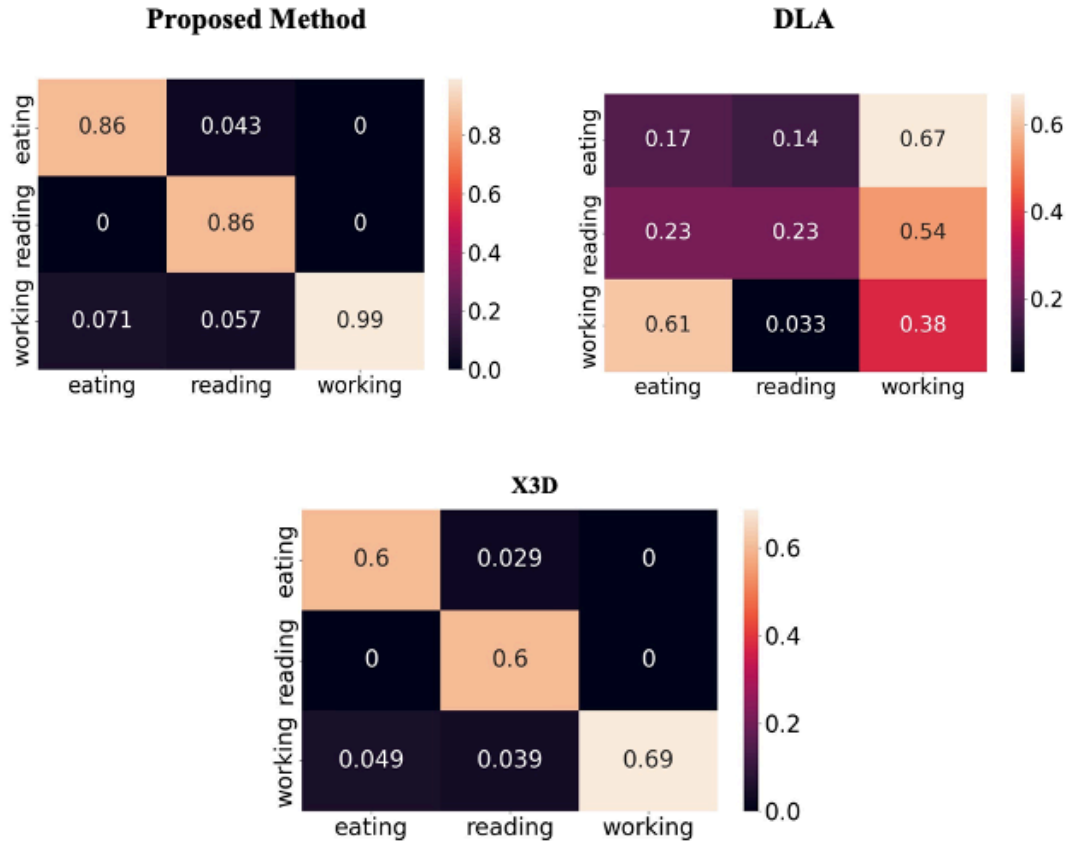


Figure 4.6: Confusion Matrix for Proposed Method, DLA and X3D

4.10 Training Results

This section presents the results of the GNN with ECC by using positional datasets. Table 2 gives a comparison of the efficiency of the proposed method with the DLA and X3D network architectures. The details of the DLA architecture are presented in [<https://github.com/kuangliu/pytorch-cifar>] and those of the X3D network are presented in [<https://github.com/facebookresearch/SlowFast>]. It can be seen from Table 2 that the proposed method can efficiently recognize reading, eating, and working from the dataset collection in general. The positional dataset was collected and labelled in csv file format. Therefore, less time was spent on processing this collected dataset than on training the video dataset as only the 3D position of the object had to be processed with 3 discrete classifications. However, the object and face positions had to be extracted from the YOLOv3 and MediaPipe frameworks beforehand. However, it took the longest time to train the 4 videos using the DLA architecture, among other methods. 2D CNN, which commonly requires many datasets and takes a longer time

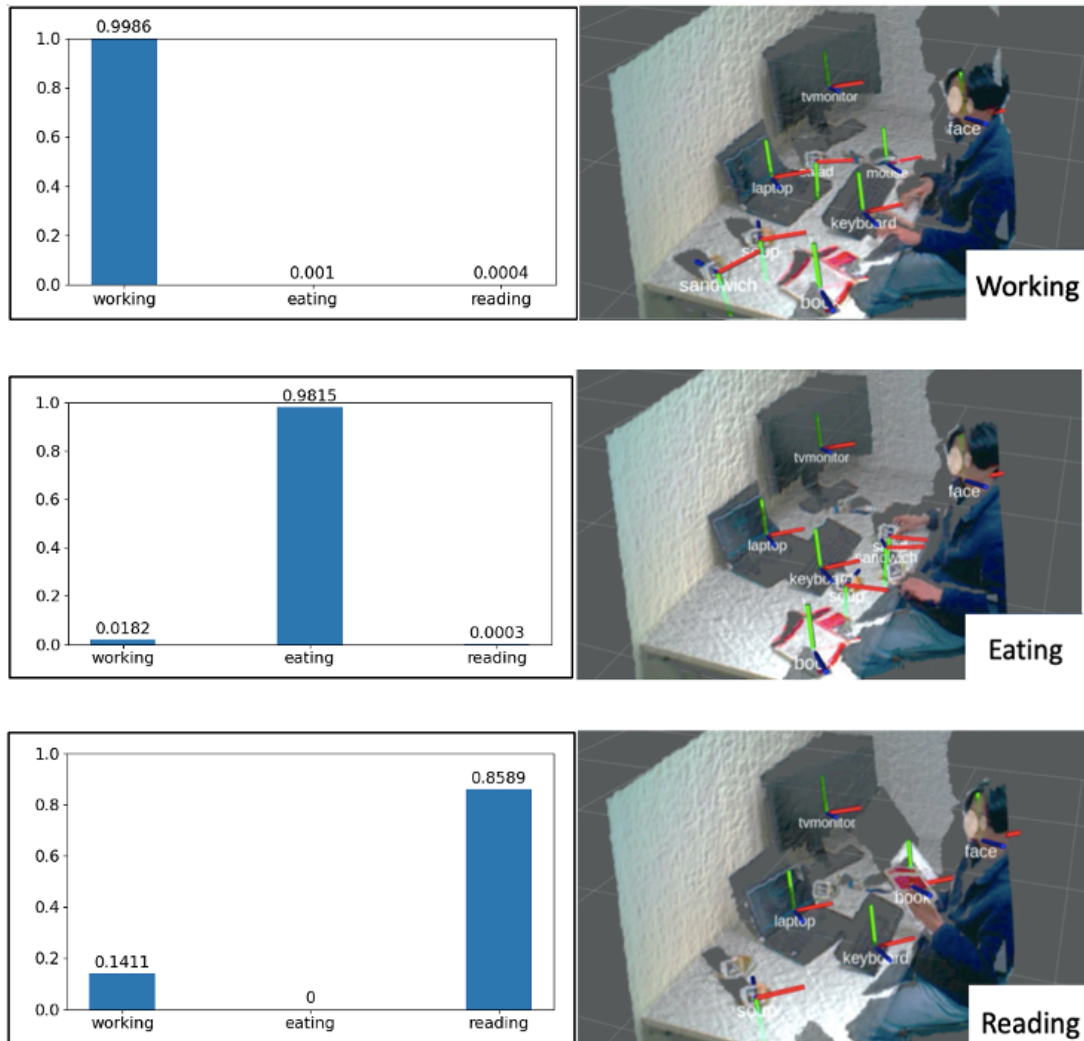


Figure 4.7: Inferencing test by using HSR Head camera

to achieve high accuracy, was used in the DLA architecture. In the X3D, a similar dataset as in the DLA was used, but the X3D showed a better performance with a shorter training time and higher accuracy than the DLA. The outstanding performance of the 3D CNN in X3D architecture is discussed in [102]. Feature extraction of the object and face detection was not required to classify the activity in X3D. The video only had to be resized for the dataset pre-processing,

such as in the DLA. It could be concluded from the results that different datasets and tools have different training times and accuracy scores. Nevertheless, the proposed method outperformed the other methods, where the GNN with ECC was able to detect eating, reading, and working based on 3D position data, and achieved a shorter training time. The confusion matrix for each tool is shown in Fig. 7. The X3D network may achieve a similar result as the proposed method if the parameters are tuned and the number of epochs is increased. However, the duration of the training time may have to

be increased. This will be investigated in a future work. By using this method, the robot will always be able to detect activities accurately while the objects and humans are in a 3D global position. Therefore, this method can be used directly for navigation targets or manipulation targets based on the current situation. For the inferencing test using an HSR camera, the trained proposed network was tested for the detection of working, eating, and reading activities, as depicted in Fig. 7.

4.11 Summary

In this paper, a human activity prediction using a GNN with ECC model was proposed based on the concept of human-object interactions from pragmatic research on human-robot communication. To obtain a homecare robot that can help with human activities, the proposed model can teach a robot to recognize human-specific activities using utterances and object information. It was also shown that the proposed system can recognize the tasks for each specified activity and detect the object related to the selected activity. From the viewpoint of the graph with ECC layers, it was assumed that the human activity consisted of the positional relationship with objects around the target human and the meaning of those objects. The object position was obtained by combining object recognition and face detection, and it was converted into node embedding using a distributed representation of words. ECC layers was used to construct the graph representation for the prediction model. Since human prediction requires instruction, the system was configured with human speech instructions to start the data collection and training. Through experiments to determine the performance of the prediction model, it was confirmed that the model could achieve an accuracy of 91% in the testing stage. The proposed method was also compared with the DLA and X3D networks. The proposed method had a lower training time as only the positional dataset collected from human interactions with objects was used to extract the features of the 2D object and face detections. However, the limitation of the proposed method only used the human 3D position. To detect more complex activities and discriminate different activities with a similar object, we need to use the whole human body movement dataset, such as the skeleton data. For example, the robot detects a human holding a book and standing in front of a bookshelf. This activity can be classified as reading or arranging the books on a bookshelf in our proposed method. Therefore, the

skeleton dataset representing a particular activity with a similar object should be considered to discriminate the different activities with a similar object in future work. Thus, the model can generate the robot task based on the current situation and environment.

Chapter 5

Integration of 3D Object-Oriented Map and 2D Predicted Occupancy Map into Unified Map

5.1 Introduction

Maps are crucial in various mobile robotics applications, providing vital information for navigation and understanding the environment. Traditional research in this domain has primarily focused on geometric map representations, such as grid maps, which have been extensively studied and widely implemented. However, as the field of robotics progresses, there is a growing interest and significance placed on mapping not only geometric but also semantic information. The exploration and integration of semantic information into maps have gained increasing attention within the robotics researchers' community. This emerging trend is driven by the recognition that incorporating semantic understanding enhances the robot's ability to interact with and interpret its surroundings more effectively. Semantic maps enable robots to recognize and comprehend objects, scenes, and other contextual cues within their environment by going beyond purely geometric information.

Including semantic information in maps provides a valuable foundation for advanced robotic functionalities, such as object manipulation, human-robot interaction, and task planning. Robots can better understand and reason about their surroundings by incorporating semantics into maps, leading to more intelligent and context-aware behaviors. This integration improves the overall performance and capabilities of mobile robots and enables them to adapt and cater to specific user needs in various application domains.

To achieve accurate semantic maps for navigation purposes, extensive research has primarily focused on two approaches: object-based SLAM algorithms [105] and comprehensive 3D world reconstruction with semantic annotations [106]. However, there has been a noticeable increase in interest in maps that prioritize objects specifically [3], [107]–[109]. This shift is primarily driven by the practical advantages

of concrete object instances, which can be easily accessed and utilized in an object-based approach.

In many existing approaches that generate object-oriented maps, a common characteristic is the inclusion of a 3D object reconstruction alongside its corresponding position within the map. While including 3D information is crucial for tasks involving object manipulation, navigation tasks often require only 2D information, which is generally sufficient [63].

5.2 Unified Map Problem Formulation

Generally, Maps show objects location in the environment. In this section, I will discuss the problem formulation of unified map by combining 2D predicted occupancy map and 3D object instances data. Let the map parameter is defined by $\mathcal{M} = (P, O, \varepsilon, c, s)$:

- P is a position of spatial location for each object.
- O denotes as object, which generally includes a $v \in V$, where V is set of possible values and position $p \in P$;
- ε is an object entity $\varepsilon \subseteq O$, following the object definition O .
- a “content” function $c : P \rightarrow O$ that, given a position, returns a set of objects at that position.
- a “search” function $s : V \rightarrow P$ that, given a value, returns the positions of objects having that value.

In here, I use 2D predicted occupancy map \mathcal{M}_{pd} as a general representation of a grid map. As a result, each $p_w \in P_w$ is represented as a row and column pair in a \mathcal{M}_{pd} ; technically, $P_w = \{(r, c) \mid r, c \in \mathbb{N}\}$. Each position relates to a cell $o_w \in \varepsilon_w$, which has the object representation $O = (v, p)$, where $v \in [0, 1]$ denotes the probability of the occupancy map. It is clear from this formulation that there is no simple way to get the extent of an item in grid maps for objects spanning several cells because each cell is treated individually. Object-oriented maps, on the other hand, should record identified items together with their extent. As a result, grid representation is inappropriate for these maps, and polygonal maps should be used instead. So, in object-oriented maps, an object $o_\sigma \in \varepsilon_\sigma$, is defined by its label and occupied area, i.e., $O = \{(v_\sigma, A_\sigma)\}$, with $v_\sigma \in V_\sigma$ and $A_\sigma \subseteq P_\sigma$, where V_σ is the set of all class labels. In this situation, the position definition relates to coordinates in a frame of reference,

therefore $P = \{(x, y) \mid x, y \in \mathbb{R}\}$. Because the map representations differ, an application that has to manage information from various type of maps would require distinct routines to handle the different types of representations independently. Ideally, the application should only interface with a "master" map that contains all environmental knowledge derived from other maps. In this research, i present a system for providing uniform access to maps from various types of maps that I call Unified Map. The unified map framework is described in Figure 5.1.

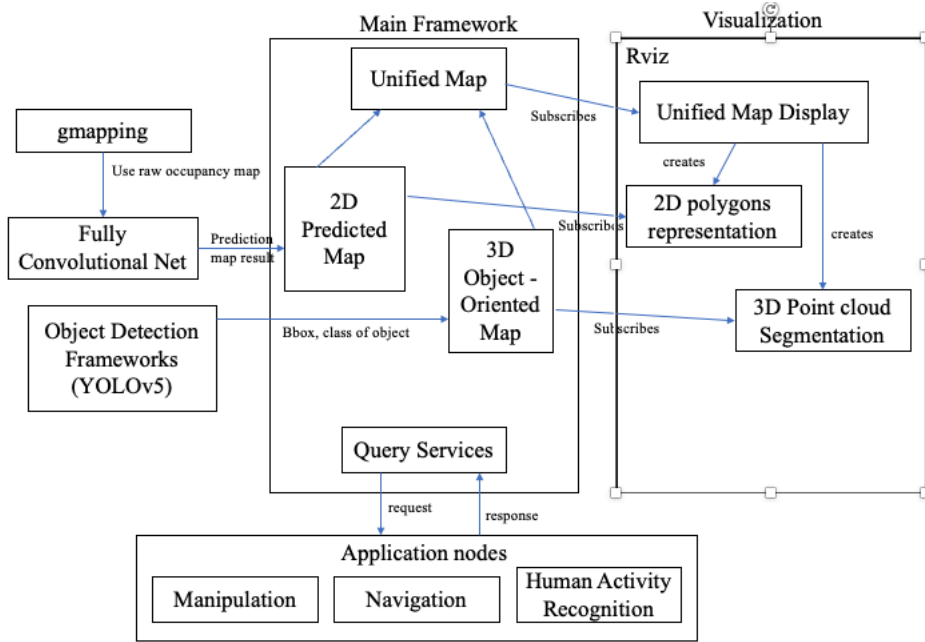


Figure 5.1: Unified map Framework. Integration from chapter 3 and chapter 4

5.3 Unified Map Framework

Let's specify a hypermap as $\mathcal{M}_u = (P_u, V_u, B)$, where P_u and V_u are the framework's definitions of position and value and $B = \{(\mathcal{M}_i, \mathfrak{t}_i, \mathfrak{u}_i)\}_{i=1}^N$ respectively consists of the N maps \mathcal{M}_i and the interface functions \mathfrak{t}_i and \mathfrak{u}_i . The framework uses a function $\mathfrak{t}_i: P_u \rightarrow P_i$ to translate from its own position definition to that of the i -th layer, and a function $\mathfrak{u}_i: V_u \rightarrow V_i$ to translate from the framework value definition to that of the i -th layer.

The "content" and "search" parts of the framework make use of the \mathfrak{t}_i and \mathfrak{u}_i functions as follows: A collection of layers $I = \{i_1, i_2, \dots, i_m\}$ at position $p_u \in P_u$ and is specified as:

$$\hat{c}(p_u, I) = \{c_i(\mathbb{t}_i(p_u))\}_{i \in I} \quad 5.1$$

Instead, the "search" functions allow one to seek for the location of objects in a collection of layers I that have the value V_u , and is defined as:

$$\hat{S}(V_u, I) = \{\mathbb{t}_i^{-1}(S_i(\mathbb{w}_i(v_u)))\}_{i \in I} \quad 5.2$$

where \mathbb{t}_i is \mathbb{t}_i^{-1} inverse function.

The unified and its layers are kept on the map server node. In addition to offering services that let other nodes query the map for data, it can store and load unified map files. Furthermore, the metadata of the unified is published, as is each layer's content that is published to a subject. The rviz plugin's unified map display has the ability to subscribe to the metadata topic. For the layers that subscribe to the relevant subjects, it automatically builds the required displays. The global position definition $P_u = \{(x, y) \mid x, y \in \mathbb{R}\}$ is used for spatial searches. There are other options for area inquiries in addition to point queries. A list of points can be used to represent a basic polygon as the area. Strings are the global value type V_u . The framework uses the content function to obtain values from the layers when a service for a spatial query is performed, whereas the search function is employed when a service for a value service is queried. Following the formalism described in next section, the occupancy and semantic layers are represented as a grid map \mathcal{M} and a polygonal map \mathcal{M}_u , respectively. The layer implementation manages all conversion operations.

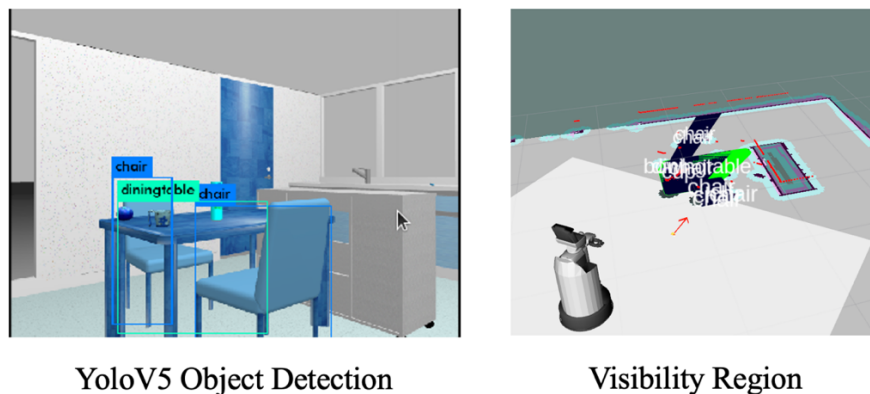


Figure 5.2: Examples of the object visualization in 2D occupancy map from object detector frame work

Polygonal object in 2D occupancy map: While incremental mapping for 2D occupancy maps is a well-studied problem, less research has been conducted on mapping for polygonal maps. According to the preceding section, a semantic object

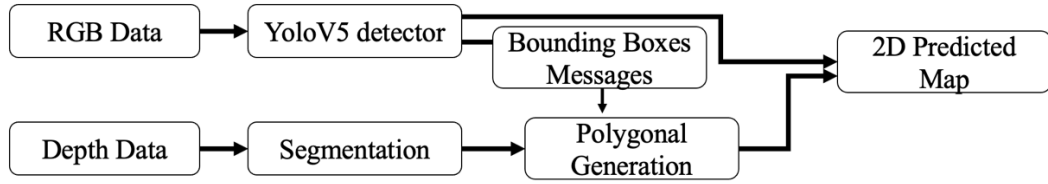


Figure 5.3: Object visualization in 2D Predicted Map

consists of a label v_σ and an area A_σ . The area is defined by a set of polygon's vertices. During the process of mapping, new sensor data must be incorporated so that the area of this polygon can be estimated incrementally. This presents three primary difficulties: 1) after each new reading, the labeled objects in the scene must be placed on the map with an estimate of their coverage area; 2) when a portion of the environment is observed again after some time, new readings must be integrated with the estimate of the area of already known objects; 3) confidence in the existence of objects must be updated whenever an already mapped area is observed, and eventually, objects with low confidence must be removed from the map.

The evidence gathering process for semantic mapping can be divided into three phases: object detection, area generation, and map construction. Figure 5.3 depicts the information transfer between the camera and the map generator. A RGB-D camera is used as the mapping sensor. A commercially available deep learning algorithm performs object detection on the RGB image. Figure 5.3 depicts how, for each detection, the algorithm generates bounding boxes around the object. Occasionally, these bounding boxes may incorporate portions of the scenery. Consequently, the detected pixels are transmitted to corresponding points in a point-cloud generated from the depth image of the RGB-D camera, and the background is eliminated using a segmentation algorithm.

Then, to ascertain the object's area on the map, the object's point-cloud cluster is projected onto the plane of the map frame. The transformation between the camera frame and the map frame must be known, so a localization is conducted on the existing occupancy layer. As an object's area, the convex hull is computed from the projected cloud. The region is forwarded to the map generator.

The generator searches the map for analogous regions of the same class. The similarity is determined by calculating the Jaccard index of the new areas and any overlapping map area. Calculating the Jaccard index between areas A and B is as follows:

$$J_{cr}(A, B) = \frac{|A \cap B|}{|A \cup B|} \quad 5.3$$

If the index exceeds a predetermined threshold, it is presumed that the areas belong to the same object. If an object is determined to be a potential match for the new area, the area is added to the object's area list and the probability of its existence is increased. A new object is created if not. If the existence probability exceeds a predetermined threshold, the object is included in the map.

The average centroid of all collected areas is calculated to determine the area to display for the object. The region whose centroid is closest to the mean is selected as the optimal region and depicted on the map. In order to eliminate falsely detected objects, the existence probability must be decreased if no object is detected.

For this purpose, the camera's field of view is utilized: for each object on the map within the field of view of a sensor measurement that has not received new evidence, the probability of its existence is decreased. The visibility region is computed by projecting the entire camera point cloud onto the (x, y) plane of the map frame and then computing the convex hull.

5.4 Experiments Setup

In this section I will discuss the experiments setup of unified map in human-centered environment. The performance of unified map is compared to the Zaenker et al. [3] approach. Our approach is not only focus on mapping and polygonal construction, but I also focus on efficiency and safety during navigation in unified map. I test this framework on home-like environment such as in Chapter 3. I use the SH environment and 2D predicted and raw map to compare the unified map framework and Zaenker et al. [3] framework. I carried out 2 experiments, mapping and obstacle avoidance. I used the Toyota HSR and collected RGB-D data with an Asus Xtion Pro mounted at a height of 1.35m and in an angle of 36° from the horizontal facing down detect the surrounding object during mapping and obstacle avoidance. In the experiment of obstacle avoidance, I use 2D LiDAR only and TEB planner to show the performance that the proposed method has significant contribution and better performance.

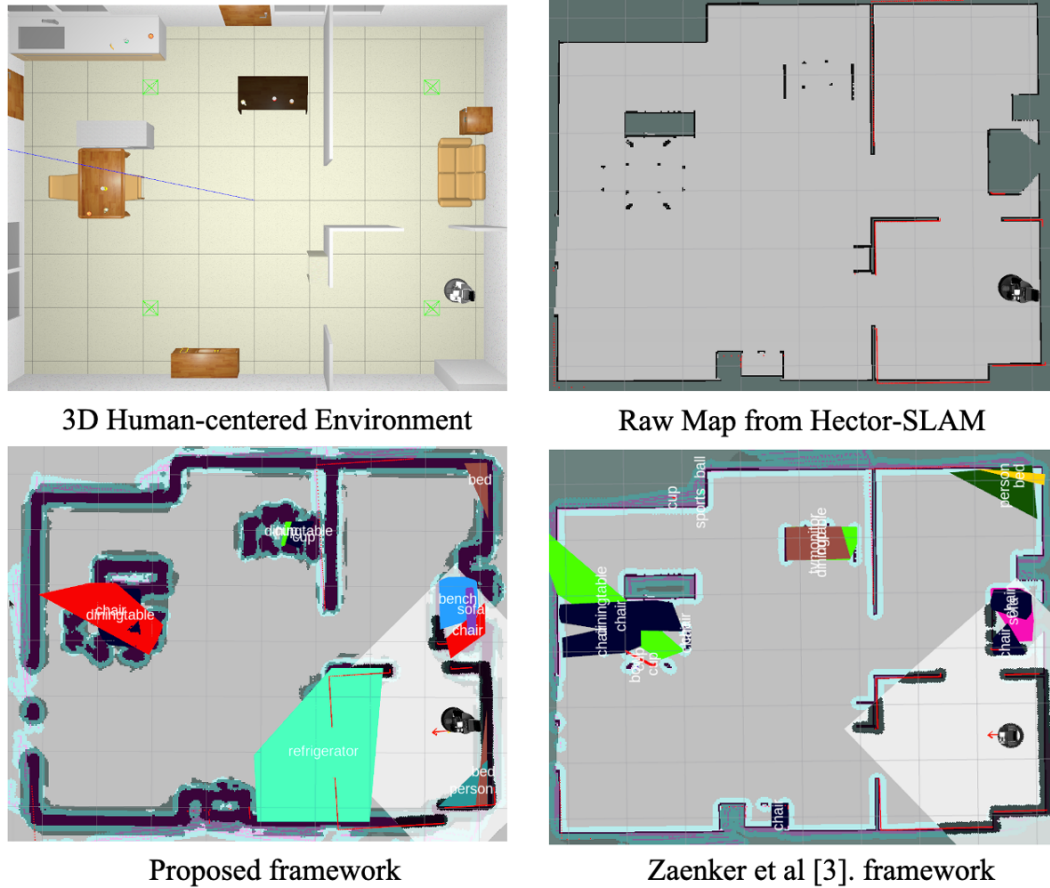


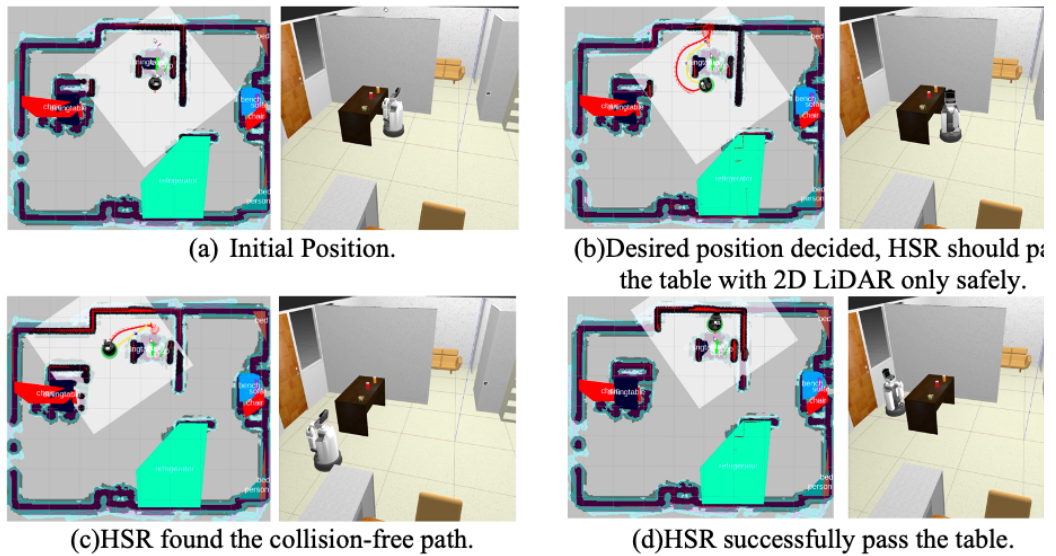
Figure 5.4: Comparison Result of Mapping in 3D environment.

5.5 Results on Mapping and Obstacle Avoidance

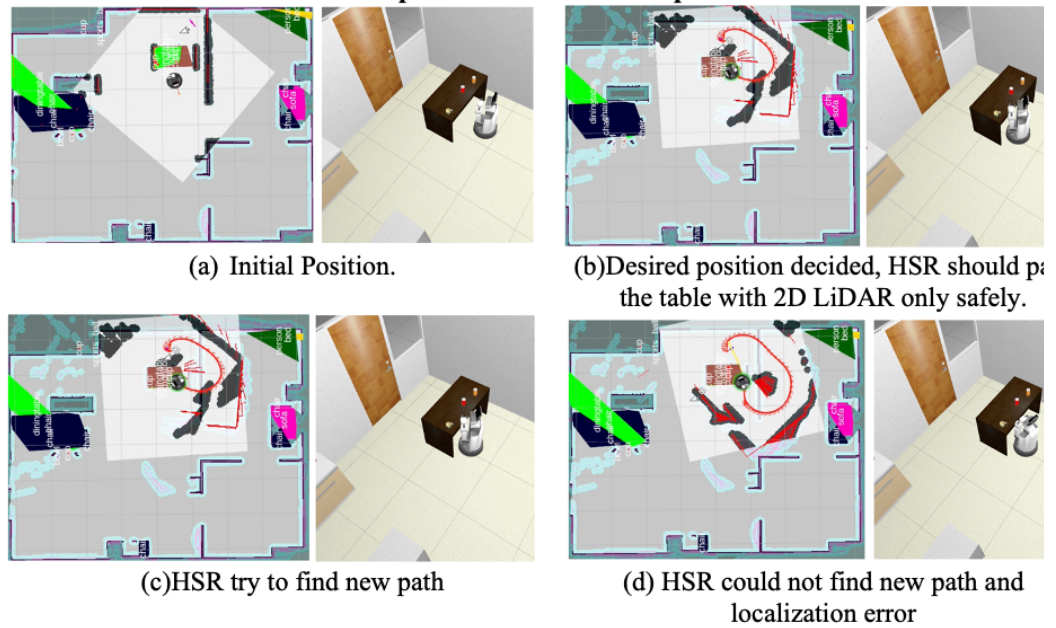
In the mapping result, HSR's RGB-D camera can detect both tabletop and ground objects. In the proposed framework, I used YOLOv3 to detect the adjacent object, whereas the framework developed by Zaenker et al. [3] employed Yolact. The detection of visualization in Zaenker et al. is marginally more precise than in the proposed framework. For example, in a table with no chairs, polygonal forms such as the table dimensions can be generated. However, in the proposed framework, the polygonal form in the 2D predicted occupancy map only depicts a small table size. Due to an occlusion issue in the object detector, the camera generates the refrigerator information despite the fact that there is no refrigerator in the larger room section.

In terms of performance in obstacle avoidance, the proposed method outperforms the Zaenker et al. [3] framework. In this investigation, the 2D predicted map plays the most significant role. The Zaenker et al. [3] framework utilizes only the conventional map from gmapping. The detected object only displays its size, color,

class, and shape, while the visualization that covers unoccupied cells in the occupancy map is not represented as an obstruction. Consequently, this infrastructure is required for camera usage during obstacle avoidance.



Unified Map in collision avoidance performance



Zaenker et al [3] framework in collision avoidance

Figure 5.5: The comparison results in collision avoidance

5.6 Summary

In this chapter, the unified map is constructed using 3D object-oriented data and a 2D map of predicted occupancy. Using polygonal map visualization in 2D predicted maps and obstacle avoidance, the performance of the proposed framework is evaluated.

I compare the proposed framework to the framework developed by Zaenker et al. [3] The results are depicted in Figures 5.4 and 5.5, which show that the proposed framework excels in obstacle avoidance using only 2D LiDAR and that the mapping performance is competitive despite the framework's slightly less accurate object record and visualization.

Chapter 6

Concluding Remarks

6.1 Conclusion

In this thesis, I have presented a novel method to unified map by utilizing modular object representations. Our approach differs from previous works in the way the created map is represented, as it allows for the incorporation of multiple representations within a single framework. This flexibility enables us to capture a richer and more comprehensive understanding of the environment. One key contribution of our approach is the 2D predicted occupancy map. The 2D predicted occupancy map can make the robot rely on 2D LiDAR only during navigation in 3D real-world environment and dynamic obstacle. Additionally, i have introduced a new applicability of the unified map for human activity recognition. The data of unified map can be used for service robot to understand the semantic representation more deeply by using GNN.

To evaluate the effectiveness of our approach, we conducted four real-world experiments. The results obtained from these experiments demonstrated significant improvements in performance when compared to an existing framework. Various metrics were employed to assess the quality and reliability of the mapping outputs, and our approach consistently outperformed the comparative framework. Furthermore, we evaluated the runtime performance of our system, specifically focusing on its capability for online mapping while the robot is in motion within the environment. The evaluation results highlighted the efficiency and effectiveness of our approach in generating real-time mapping results, indicating its practical applicability for dynamic environments.

In summary, our research presents an innovative approach to online semantic mapping by employing modular object representations. Through the incorporation of multiple representations, likelihood calculations, and object refinement steps, our approach enhances the accuracy, robustness, and runtime efficiency of the mapping process. The conducted experiments validate the efficacy of our approach, showcasing its superiority over existing frameworks.

6.2 Future works

In the future works of the unified map, a key objective is to integrate lifelong mapping techniques into the existing framework. Lifelong mapping refers to the continuous updating and prediction of the 2D occupancy map in real-time, using only a 2D LiDAR sensor. This approach offers several advantages, including the ability to dynamically adapt the map based on evolving environmental conditions and the elimination of the need for additional sensor inputs.

To further enhance the capabilities of the unified map, the visualization of 2D object polygons will be incorporated through the utilization of an RGB-D camera. This integration will enable the map to capture and represent not only the spatial occupancy information but also the geometric characteristics of detected objects. By incorporating object polygons, the resulting map becomes more informative and visually interpretable, allowing for a more comprehensive understanding of the environment.

One significant advantage of this proposed approach is the direct usability of the resulting map within the unified map framework. Unlike traditional methods that require extensive preprocessing steps to refine and align the map data, the integration of lifelong mapping and object polygon visualization eliminates the need for such preprocessing, simplifying the overall mapping process. Consequently, the unified map can be readily utilized in various robotic applications without significant delays or complex data transformations.

Furthermore, in order to exploit the rich information available in the environment, the structural elements of the scene, such as architectural components, will be leveraged to refine the map's accuracy and detail. These elements provide contextual information that can significantly improve the map's understanding of the environment. By incorporating this additional information, the map becomes more robust, enabling more accurate localization, object detection, and scene understanding.

Moreover, the map information will be employed for room classification tasks. The objective is to enable the robot to categorize different types of rooms based on the objects detected within them. This classification capability is particularly relevant in human-centered environments, where distinct types of rooms, such as kitchens, bedrooms, living rooms, bathrooms in residential settings, or workspace areas in office environments, hold specific functional characteristics. By analyzing the objects present

within a given room, the robot can infer the room's purpose, facilitating context-aware decision-making and task execution.

Overall, the future development of the unified map aims to integrate lifelong mapping, object polygon visualization, and the utilization of structural elements to create a comprehensive and efficient mapping framework. This framework not only reduces the preprocessing burden but also enables accurate room classification, empowering robots to navigate and interact intelligently in a wide range of environments.

References

- [1] H. M. Do, M. Pham, W. Sheng, D. Yang, and M. Liu, “RiSH: A robot-integrated smart home for elderly care,” *Rob Auton Syst*, vol. 101, pp. 74–92, Mar. 2018, doi: 10.1016/J.ROBOT.2017.12.008.
- [2] J. Secker, R. Hill, L. Villeneuve, and S. Parkman, “Promoting independence: But promoting what and how?,” *Ageing Soc*, vol. 23, no. 3, pp. 375–391, May 2003, doi: 10.1017/S0144686X03001193.
- [3] T. Zaenker, F. Verdoja, and V. Kyrki, “Hypermap Mapping Framework and its Application to Autonomous Semantic Exploration,” *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, vol. 2020-Septe, pp. 133–139, 2020, doi: 10.1109/MFI49285.2020.9235231.
- [4] N. Dengler, T. Zaenker, F. Verdoja, and M. Bennewitz, “Online object-oriented semantic mapping and map updating,” *2021 10th European Conference on Mobile Robots, ECMR 2021 - Proceedings*, 2021, doi: 10.1109/ECMR50962.2021.9568817.
- [5] K. P. Sivananda, F. Verdoja, and V. Kyrki, “Augmented Environment Representations with Complete Object Models,” Mar. 2021, [Online]. Available: <http://arxiv.org/abs/2103.07298>
- [6] J. L. Matez-Bandera, D. Fernandez-Chaves, J. R. Ruiz-Sarmiento, J. Monroy, N. Petkov, and J. Gonzalez-Jimenez, “LTC-Mapping, Enhancing Long-Term Consistency of Object-Oriented Semantic Maps in Robotics,” *Sensors*, vol. 22, no. 14, Jul. 2022, doi: 10.3390/s22145308.
- [7] M. Kollmitz, D. Buscher, and W. Burgard, “Predicting Obstacle Footprints from 2D Occupancy Maps by Learning from Physical Interactions,” *Proc IEEE Int Conf Robot Autom*, pp. 10256–10262, 2020, doi: 10.1109/ICRA40945.2020.9197474.
- [8] F. Mcleay, V. S. Osburg, V. Yoganathan, and A. Patterson, “Replaced by a Robot: Service Implications in the Age of the Machine,” *journals.sagepub.com*, vol. 24, no. 1, pp. 104–121, Feb. 2021, doi: 10.1177/1094670520933354.
- [9] “Introduction into Service Robots”, Accessed: May 16, 2023. [Online]. Available: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=55890.
- [10] J. Wirtz *et al.*, “Brave new world: service robots in the frontline,” *Journal of Service Management*, vol. 29, no. 5, pp. 907–931, Nov. 2018, doi: 10.1108/JOSM-04-2018-0119/FULL/.
- [11] D. Belanche, L. V. Casaló, C. Flavián, and J. Schepers, “Service robot implementation: a theoretical framework and research agenda,” *Service Industries Journal*, vol. 40, no. 3–4, pp. 203–225, Mar. 2020, doi: 10.1080/02642069.2019.1672666.
- [12] “Top Trends on the Gartner Hype Cycle for Artificial Intelligence, 2019.” <https://www.gartner.com/smarterwithgartner/top-trends-on-the-gartner-hype-cycle-for-artificial-intelligence-2019> (accessed May 16, 2023).

- [13] M. Decker, M. Fischer, and I. Ott, “Service Robotics and Human Labor: A first technology assessment of substitution and cooperation,” *Rob Auton Syst*, vol. 87, pp. 348–354, Jan. 2017, doi: 10.1016/J.ROBOT.2016.09.017.
- [14] “Service Robotics Market Size, Global Industry Trends Forecast, Opportunities 2028.” <https://www.marketsandmarkets.com/Market-Reports/service-robotics-market-681.html> (accessed May 16, 2023).
- [15] “Brookings survey finds 52 percent believe robots will perform most human activities in 30 years.” <https://www.brookings.edu/blog/techtank/2018/06/21/brookings-survey-finds-52-percent-believe-robots-will-perform-most-human-activities-in-30-years/> (accessed May 16, 2023).
- [16] S. (Sam) Kim, J. Kim, F. Badu-Baiden, M. Giroux, and Y. Choi, “Preference for robot service or human service in hotels? Impacts of the COVID-19 pandemic,” *Int J Hosp Manag*, vol. 93, Feb. 2021, doi: 10.1016/J.IJHM.2020.102795.
- [17] W. Garage, *PR2 User Manual*. 2012.
- [18] T. Yamamoto, K. Terada, A. Ochiai, F. Saito, Y. Asahara, and K. Murase, “Development of the Research Platform of a Domestic Mobile Manipulator Utilized for International Competition and Field Test,” *IEEE International Conference on Intelligent Robots and Systems*, pp. 7675–7682, 2018, doi: 10.1109/IROS.2018.8593798.
- [19] T. Yamamoto, K. Terada, A. Ochiai, F. Saito, Y. Asahara, and K. Murase, “Development of Human Support Robot as the research platform of a domestic mobile manipulator,” *ROBOMECH Journal*, vol. 6, no. 1, 2019, doi: 10.1186/s40648-019-0132-3.
- [20] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, “Monte Carlo Localization: Efficient Position Estimation for Mobile Robots.”
- [21] M. Quigley *et al.*, “ROS: an open-source Robot Operating System.” [Online]. Available: <http://stair.stanford.edu>
- [22] “Graph Theory with Applications to Engineering & Computer Science”.
- [23] R. Gldenring, “Applying Deep Reinforcement Learning in the Navigation of Mobile Robots in Static and Dynamic Environments.”
- [24] S. Shalev-Shwartz and S. Ben-David, “Understanding Machine Learning.”
- [25] “Learning representations by back-propagating errors ”.
- [26] Y. Bengio, I. J. Goodfellow, and A. Courville, “Deep Learning,” 2015.
- [27] “CS231n Convolutional Neural Networks for Visual Recognition.” <https://cs231n.github.io/> (accessed Jun. 16, 2023).
- [28] S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.”
- [29] J. Redmon and A. Farhadi, “YOLOv3: An incremental improvement,” *ArXiv*, pp. 10–11, 2018.
- [30] W. Kehl, F. Milletari, F. Tombari, S. Ilic, and N. Navab, “Deep learning of local RGB-D patches for 3D object detection and 6D pose estimation,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2016. doi: 10.1007/978-3-319-46487-9_13.
- [31] D. O. Melinte, A.-M. Travediu, and D. N. Dumitriu, “Deep Convolutional Neural Networks Object Detector for Real-Time Waste Identification,” *Applied Sciences*, vol. 10, no. 20, p. 7301, 2020, doi: 10.3390/app10207301.

- [32] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield, “Deep Object Pose Estimation for Semantic Robotic Grasping of Household Objects,” no. CoRL, pp. 1–11, 2018, [Online]. Available: <http://arxiv.org/abs/1809.10790>
- [33] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, “PointNet: Deep learning on point sets for 3D classification and segmentation,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017. doi: 10.1109/CVPR.2017.16.
- [34] W. Jing, W. Zhang, L. Li, D. Di, G. Chen, and J. Wang, “AGNet: An Attention-Based Graph Network for Point Cloud Classification and Segmentation,” *Remote Sens (Basel)*, vol. 14, no. 4, pp. 1–18, 2022, doi: 10.3390/rs14041036.
- [35] S. Caelles, K. K. Maninis, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. Van Gool, “One-Shot video object segmentation,” *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 5320–5329, 2017, doi: 10.1109/CVPR.2017.565.
- [36] J. Long, E. Shelhamer, and T. Darrell, “Fully Convolutional Networks for Semantic Segmentation,” Nov. 2014, [Online]. Available: <http://arxiv.org/abs/1411.4038>
- [37] Y. Hu, J. Hugonot, and P. Fua, “Segmentation-driven 6D Object Pose Estimation”.
- [38] Y. Lecun, L. Eon Bottou, Y. Bengio, and P. H. Abstract|, “Gradient-Based Learning Applied to Document Recognition.”
- [39] F. de A. M. Pimentel and P. T. Aquino-Jr, “Evaluation of ROS Navigation Stack for Social Navigation in Simulated Environments,” *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 102, no. 4, 2021, doi: 10.1007/s10846-021-01424-z.
- [40] E. Marder-Eppstein, E. Berger, T. Foote, B. Gerkey, and K. Konolige, “The office marathon: Robust navigation in an indoor office environment,” *Proc IEEE Int Conf Robot Autom*, pp. 300–307, 2010, doi: 10.1109/ROBOT.2010.5509725.
- [41] D. Fox, W. Burgard, and S. Thrun, “The Dynamic Window Approach to Collision Avoidance,” *IEEE Robot Autom Mag*, vol. 4, no. 1, pp. 23–33, 1997, doi: 10.1109/100.580977.
- [42] B. P. Gerkey and K. Konolige, “Planning and Control in Unstructured Terrain,” *ICRA Workshop on Path Planning on Costmaps*, 2008.
- [43] S. Quinlan and O. Khatib, “Elastic bands: connecting path planning and control,” in *[1993] Proceedings IEEE International Conference on Robotics and Automation*, Atlanta, GA, USA: IEEE Comput. Soc. Press, 1993. doi: 10.1109/ROBOT.1993.291936.
- [44] R. Christoph, F. Hoffmann, and T. Bertram, “Timed-Elastic-Bands for Time-Optimal Point-to-Point Nonlinear Model Predictive Control,” pp. 3352–3357, 2015.
- [45] R. Guldenring, M. Gornier, N. Hendrich, N. J. Jacobsen, and J. Zhang, “Learning local planners for human-aware navigation in indoor environments,” *IEEE International Conference on Intelligent Robots and Systems*, pp. 6053–6060, 2020, doi: 10.1109/IROS45743.2020.9341783.
- [46] U. Patel, N. Kumar, A. J. Sathymoorthy, and D. Manocha, “Dynamically Feasible Deep Reinforcement Learning Policy for Robot Navigation in Dense Mobile Crowds,” 2020, [Online]. Available: <http://arxiv.org/abs/2010.14838>

- [47] P. Long, T. Fanl, X. Liao, W. Liu, H. Zhang, and J. Pan, “Towards optimally decentralized multi-robot collision avoidance via deep reinforcement learning,” *Proc IEEE Int Conf Robot Autom*, pp. 6252–6259, 2018, doi: 10.1109/ICRA.2018.8461113.
- [48] S. Yao, G. Chen, Q. Qiu, J. Ma, X. Chen, and J. Ji, “Crowd-Aware Robot Navigation for Pedestrians with Multiple Collision Avoidance Strategies via Map-based Deep Reinforcement Learning”.
- [49] S. B. Banisetty, V. Rajamohan, F. Vega, and D. Feil-Seifer, “A deep learning approach to multi-context socially-aware navigation,” in *2021 30th IEEE International Conference on Robot and Human Interactive Communication, RO-MAN 2021*, 2021, pp. 23–30. doi: 10.1109/RO-MAN50785.2021.9515424.
- [50] K. Li, Y. Xu, J. Wang, and M. Q. H. Meng, “SARL* : Deep reinforcement learning based human-aware navigation for mobile robot in indoor environments,” *IEEE International Conference on Robotics and Biomimetics, ROBIO 2019*, no. December, pp. 688–694, 2019, doi: 10.1109/ROBIO49542.2019.8961764.
- [51] B. Okal, T. Linder, D. Vasquez, S. Wehner, O. Islas, and L. Palmieri, “Pedestrian Simulator,” 2018. https://github.com/srl-freiburg/pedsim_ros
- [52] L. Kästner, T. Buiyan, X. Zhao, L. Jiao, Z. Shen, and J. Lambrecht, “Arena-Rosnav: Towards Deployment of Deep-Reinforcement-Learning-Based Obstacle Avoidance into Conventional Autonomous Navigation Systems,” 2021, doi: 10.1109/iros51168.2021.9636226.
- [53] H. Darweesh *et al.*, “Open source integrated planner for autonomous navigation in highly dynamic environments,” *Journal of Robotics and Mechatronics*, vol. 29, no. 4, pp. 668–684, 2017, doi: 10.20965/jrm.2017.p0668.
- [54] K. Tobita, Y. Shikanai, and K. Mima, “Study on automatic operation of manual wheelchair prototype and basic experiments,” *Journal of Robotics and Mechatronics*, vol. 33, no. 1, pp. 69–77, 2021, doi: 10.20965/jrm.2021.p0069.
- [55] J. Lundell, F. Verdoja, and V. Kyrki, “Hallucinating Robots: Inferring Obstacle Distances from Partial Laser Measurements,” *IEEE International Conference on Intelligent Robots and Systems*, pp. 4781–4787, 2018, doi: 10.1109/IROS.2018.8594399.
- [56] F. Verdoja, J. Lundell, and V. Kyrki, “Deep network uncertainty maps for indoor navigation,” *IEEE-RAS International Conference on Humanoid Robots*, vol. 2019-Octob, pp. 112–119, 2019, doi: 10.1109/Humanoids43949.2019.9035016.
- [57] V. Dutta and T. Zielinska, “Prognosing Human Activity Using Actions Forecast and Structured Database,” *IEEE Access*, vol. 8, pp. 6098–6116, 2020, doi: 10.1109/ACCESS.2020.2963933.
- [58] S. Qi, W. Wang, B. Jia, J. Shen, and S. C. Zhu, “Learning human-object interactions by graph parsing neural networks,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11213 LNCS, pp. 407–423, 2018, doi: 10.1007/978-3-030-01240-3_25.
- [59] R. Morais, V. Le, S. Venkatesh, and T. Tran, “Learning asynchronous and sparse human-object interaction in videos,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 16036–16045, 2021, doi: 10.1109/CVPR46437.2021.01578.

- [60] Z. Liang, J. Liu, Y. Guan, and J. Rojas, “Visual-Semantic Graph Attention Networks for Human-Object Interaction Detection,” *2021 IEEE International Conference on Robotics and Biomimetics, ROBIO 2021*, pp. 1441–1447, 2021, doi: 10.1109/ROBIO54168.2021.9739429.
- [61] T. Wang, T. Yang, M. Danelljan, F. S. Khan, X. Zhang, and J. Sun, “Learning Human-Object Interaction Detection using Interaction Points,” vol. 1, 2020.
- [62] M. Simonovsky and N. Komodakis, “Dynamic edge-conditioned filters in convolutional neural networks on graphs,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017, pp. 29–38. doi: 10.1109/CVPR.2017.11.
- [63] P. Regier, A. Milioto, P. Karkowski, C. Stachniss, and M. Bennewitz, “Classifying Obstacles and Exploiting Knowledge about Classes for Efficient Humanoid Navigation,” in *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*, Beijing, China: IEEE, Nov. 2018. doi: 10.1109/HUMANOIDS.2018.8625036.
- [64] J. Kindle, F. Furrer, T. Novkovic, J. J. Chung, R. Siegwart, and J. Nieto, “Whole-Body Control of a Mobile Manipulator using End-to-End Reinforcement Learning,” 2020.
- [65] J. Bohren *et al.*, “Towards autonomous robotic butlers: Lessons learned with the PR2,” *Proc IEEE Int Conf Robot Autom*, pp. 5568–5575, 2011, doi: 10.1109/ICRA.2011.5980058.
- [66] F. Xia, C. Li, R. Martin-Martin, O. Litany, A. Toshev, and S. Savarese, “ReLMoGen: Integrating Motion Generation in Reinforcement Learning for Mobile Manipulation,” pp. 4583–4590, 2021, doi: 10.1109/icra48506.2021.9561315.
- [67] Melonee Wise, M. Ferguson, D. King, E. Diehr, and D. Dymesich, “Fetch & Freight: Standard Platforms for Service Robot Applications,” *Workshop on Autonomous Mobile Service Robots, held at the 2016 International Joint Conference on Artificial Intelligence*, pp. 2–7, 2016.
- [68] K. Blomqvist *et al.*, “Go Fetch: Mobile Manipulation in Unstructured Environments,” *arXiv*. 2020.
- [69] P. Singamaneni, A. Favier, and R. Alami, “Human-Aware Navigation Planner for Diverse Human-Robot Contexts,” 2021.
- [70] L. Kästner, T. Buiyan, X. Zhao, Z. Shen, C. Marx, and J. Lambrecht, “Connecting Deep-Reinforcement-Learning-based Obstacle Avoidance with Conventional Global Planners using Waypoint Generators,” 2021. doi: 10.1109/iros51168.2021.9636039.
- [71] G. G, S. C, and B. W, “Characterization and distributed electrothermal modelling of IGBT chip-application to top-metal ageing,” *IEEE transactions on Robotics*, pp. 23–34, 2007, doi: 10.1109/TRO.2006.889486.
- [72] S. Kohlbrecher, O. Von Stryk, J. Meyer, and U. Klingauf, “A Flexible and Scalable SLAM System with Full 3D Motion Estimation,” in *SSRR 2015 IEEE International Symposium on Safety, Security, and Rescue Robotics*, IEEE, Ed., Kyoto, Japan, 2016, pp. 0–5.
- [73] W. Hess, D. Kohler, H. Rapp, and D. Andor, “Real-time loop closure in 2D LIDAR SLAM,” in *Proceedings - IEEE International Conference on Robotics and Automation*, Stockholm, Sweden, pp. 1271–1278. doi: 10.1109/ICRA.2016.7487258.
- [74] A. Handa, V. Patraucean, S. Stent, and R. Cipolla, “Scenenet: An annotated model generator for indoor scene understanding,” *Proc IEEE Int Conf Robot*

- Autom*, vol. 2016-June, pp. 5737–5743, 2016, doi: 10.1109/ICRA.2016.7487797.
- [75] M. Wada, A. Takagi, and S. Mori, “A Mobile Platform with a Dual-wheel Caster-drive Mechanism for Holonomic and Omnidirectional Mobile Robots.”
- [76] S. Quinlan and O. Khatib, “Elastic Bands: Connecting Path Planning and Control.”
- [77] M. Yani, A. R. A. Besari, N. Yamada, and N. Kubota, “Ecological-Inspired System Design for Safety Manipulation Strategy in Home-care Robot,” in *2020 International Symposium on Community-Centric Systems, CcS 2020*, 2020. doi: 10.1109/CcS49175.2020.9231354.
- [78] H. Riaz, A. Terra, K. Raizer, R. Inam, and A. Hata, “Scene Understanding for Safety Analysis in Human-Robot Collaborative Operations,” *2020 6th International Conference on Control, Automation and Robotics, ICCAR 2020*, pp. 722–731, 2020, doi: 10.1109/ICCAR49639.2020.9108083.
- [79] A. Carolina and H. Silva, “Scene Understanding for Autonomous Robots Operating in Indoor Environments by,” 2021.
- [80] M. S. Ryoo, “Human activity prediction: Early recognition of ongoing activities from streaming videos,” *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1036–1043, 2011, doi: 10.1109/ICCV.2011.6126349.
- [81] S. Wan, L. Qi, X. Xu, C. Tong, and Z. Gu, “Deep Learning Models for Real-time Human Activity Recognition with Smartphones,” *Mobile Networks and Applications*, vol. 25, no. 2, pp. 743–755, 2020, doi: 10.1007/s11036-019-01445-x.
- [82] K. Li, J. Wu, X. Zhao, and M. Tan, “Real-Time Human-Robot Interaction for a Service Robot Based on 3D Human Activity Recognition and Human-Mimicking Decision Mechanism,” *8th Annual IEEE International Conference on Cyber Technology in Automation, Control and Intelligent Systems, CYBER 2018*, pp. 498–503, 2019, doi: 10.1109/CYBER.2018.8688272.
- [83] M. Latah, “Human action recognition using support vector machines and 3D convolutional neural networks,” *International Journal of Advances in Intelligent Informatics*, vol. 3, no. 1, pp. 47–55, 2017, doi: 10.26555/ijain.v3i1.89.
- [84] Y. A. Andrade-Ambriz, S. Ledesma, M. A. Ibarra-Manzano, M. I. Oros-Flores, and D. L. Almanza-Ojeda, “Human activity recognition using temporal convolutional neural network architecture,” *Expert Syst Appl*, vol. 191, no. March 2021, p. 116287, 2022, doi: 10.1016/j.eswa.2021.116287.
- [85] A. R. A. Besari, A. A. Saputra, W. H. Chin, Kurnianingsih, and N. Kubota, “Finger Joint Angle Estimation With Visual Attention for Rehabilitation Support: A Case Study of the Chopsticks Manipulation Test,” *IEEE Access*, vol. 10, no. September, pp. 91316–91331, 2022, doi: 10.1109/ACCESS.2022.3201894.
- [86] N. Khalid, Y. Y. Ghadi, M. Gochoo, A. Jalal, and K. Kim, “Semantic Recognition of Human-Object Interactions via Gaussian-Based Elliptical Modeling and Pixel-Level Labeling,” *IEEE Access*, vol. 9, pp. 111249–111266, 2021, doi: 10.1109/ACCESS.2021.3101716.
- [87] M. Hassan, P. Ghosh, J. Tesch, D. Tzionas, and M. J. Black, “Populating 3D scenes by learning human-scene interaction,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 14703–14713, 2021, doi: 10.1109/CVPR46437.2021.01447.

- [88] Y. Tian, L. Chen, W. Song, Y. Sung, and S. Woo, “Dgcb-net: Dynamic graph convolutional broad network for 3d object recognition in point cloud,” *Remote Sens (Basel)*, vol. 13, no. 1, pp. 1–20, 2021, doi: 10.3390/rs13010066.
- [89] L. Shi, S. Li, Q. Zheng, L. Cao, L. Yang, and G. Pan, “Maximum Entropy Reinforcement Learning with Evolution Strategies,” 2020.
- [90] S. A. Taylor, R. De Jong, T. Azevedo, M. Mattina, and P. Maji, “Towards Efficient Point Cloud Graph Neural Networks Through Architectural Simplification,” *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2021-Octob, pp. 2095–2104, 2021, doi: 10.1109/ICCVW54120.2021.00237.
- [91] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, and P. W. Battaglia, “Learning to Simulate Complex Physics with Graph Networks,” *ArXiv*, 2020.
- [92] I. W. McBrearty and G. C. Beroza, “Earthquake Location and Magnitude Estimation with Graph Neural Networks,” pp. 1–5, 2022.
- [93] P. Ruiz Puentes *et al.*, “Predicting target–ligand interactions with graph convolutional networks for interpretable pharmaceutical discovery,” *Sci Rep*, vol. 12, no. 1, pp. 1–17, 2022, doi: 10.1038/s41598-022-12180-x.
- [94] J. Xiong *et al.*, “Multi-instance learning of graph neural networks for aqueous pKa prediction,” *Bioinformatics*, vol. 38, no. 3, pp. 792–798, 2022, doi: 10.1093/bioinformatics/btab714.
- [95] W. Fan *et al.*, “Graph neural networks for social recommendation,” in *The Web Conference 2019 - Proceedings of the World Wide Web Conference, WWW 2019*, 2019, pp. 417–426. doi: 10.1145/3308558.3313488.
- [96] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016. doi: 10.1109/CVPR.2016.91.
- [97] C. Lugaresi *et al.*, “MediaPipe: A Framework for Building Perception Pipelines,” 2019.
- [98] T. Foote, “Tf: The transform library,” in *IEEE Conference on Technologies for Practical Robot Applications, TePRA*, 2013. doi: 10.1109/TePRA.2013.6556373.
- [99] M. Fey and J. E. Lenssen, “FAST GRAPH REPRESENTATION LEARNING WITH PYTORCH GEOMETRIC,” in *The International Conference on Learning Representations (ICLR)*, New Orleans, Louisiana, United States, 2019, pp. 1–9.
- [100] L. Wu, P. Cui, J. Pei, and L. Zhao, *Graph Neural Networks: Foundations, Frontiers, and Applications*. 2022. doi: 10.1007/978-981-16-6054-2.
- [101] F. Yu, D. Wang, E. Shelhamer, and T. Darrell, “Deep Layer Aggregation,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2403–2412. doi: 10.1109/CVPR.2018.00255.
- [102] C. Feichtenhofer, “X3D: Expanding Architectures for Efficient Video Recognition,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 200–210, 2020, doi: 10.1109/CVPR42600.2020.00028.
- [103] K. Soomro, A. R. Zamir, and M. Shah, “UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild,” no. November, 2012.
- [104] W. Kay *et al.*, “The Kinetics Human Action Video Dataset,” 2017.

- [105] C. Cadena *et al.*, “Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age,” *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, Dec. 2016, doi: 10.1109/TRO.2016.2624754.
- [106] Institute of Electrical and Electronics Engineers, *2020 IEEE International Conference on Robotics and Automation (ICRA) : 31 May-31 August, 2020. Paris, France.*
- [107] W. Li, J. Gu, B. Chen, and J. Han, “Incremental Instance-Oriented 3D Semantic Mapping via RGB-D Cameras for Unknown Indoor Scene,” *Discrete Dyn Nat Soc*, vol. 2020, 2020, doi: 10.1155/2020/2528954.
- [108] N. Sünderhauf, T. T. Pham, Y. Latif, M. Milford, and I. Reid, “Meaningful Maps With Object-Oriented Semantic Mapping,” Sep. 2016, [Online]. Available: <http://arxiv.org/abs/1609.07849>
- [109] Y. Nakajima and H. Saito, “Efficient object-oriented semantic mapping with object detector,” *IEEE Access*, vol. 7, pp. 3206–3213, 2019, doi: 10.1109/ACCESS.2018.2887022.