Study on Implementation and Navigation of Autonomous Mobile Service Robots

松本, 耕平

https://hdl.handle.net/2324/4784642

出版情報：Kyushu University, 2021, 博士（工学）, 課程博士
バージョン：
権利関係：

Study on Implementation and Navigation of Autonomous Mobile Service Robots

Kohei Matsumoto

A dissertation submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

Department of Advanced Information Technology,
Graduate School of Information Science and Electrical Engineering,
Kyushu University

February 2022

Abstract

To solve various social problems, such as the declining birthrate, aging population, and depopulation of rural areas, autonomous mobile service robots are expected to assist or replace tasks such as transportation, security, agriculture, and nursing care. The implementation of an autonomous mobile robot requires an appropriate combination of hardware and software, and in particular, it is essential to incorporate three elements: localization, environmental perception, and path planning. In addition, for realizing an advanced service robot, the communication environment for sending commands to the robot from a remote location and for processing and utilizing the information obtained from the sensors mounted on it is a necessary element. Furthermore, service robots are expected to play active roles in environments where people typically live, and they need to operate safely and efficiently in a crowded environment. In the research presented in this dissertation, I focus mainly on the localization and communication environment of robot implementation and develop a system using the Quasi-Zenith Satellite System (QZSS) and the 5th-generation mobile communication system (5G). For navigation, I propose a method to generate optimal actions for a robot by learning changes in pedestrian behavior. Specifically, I propose the following: (1) a tour guide robot using the QZSS and a co-experiencing robot using 5G in a theme park and (2) a navigation method for a mobile robot in a dynamic environment using deep reinforcement learning based on predictive state representation (PSR). The details are described below.

(1) The QZSS is the latest global navigation satellite system (GNSS) operated by Japan, which enables centimeter-level positioning with only a single module using an original error correction method called CLAS. In this study, I compare the accuracy of

the QZSS with an existing high-precision GNSS, implement the QZSS on a guidance robot, and conduct experiments in a real theme park environment. In contrast, 5G is a recently developed communication system, which enables higher capacity wireless communication than conventional systems. Using 5G, I develop a system that transmits a 360-degree 4K video as a robot experience to a remote location with high speed and low latency, and allows users to co-experience it using the head-mounted display. I also describe experiments of this co-experiencing robot system in a real environment.

(2) The behavior of pedestrians may change under various influences, and therefore, the behavior of their robots may also be affected. The method proposed in this study is based on predictive state representation (PSR). It can predict the changes in an environment after the action of an agent in a time series, to deal with the changes in the behavior of pedestrians due to the influence of the action of the robot. In addition, I propose a new PSR model using a graph convolution structure to deal with the change in the pedestrian behavior caused by the influence of other pedestrians, which cannot be considered in the conventional PSR model. Moreover, I show by experiments that the proposed model is more effective than the conventional model. In addition, I propose two methods for integrating PSR states to manage the case in which the number of pedestrians differs between training and testing, and compare the performance of both methods by experiments.

Acknowledgements

This dissertation would not have been possible without the guidance and assistance of several individuals who advised me, provided relevant experience, and contributed their time and effort to prepare and complete the related studies.

First, I would like to thank my supervisor, Prof. Ryo Kurazume, for his constant and tremendous advice, encouragement, and support. The interactions with him developed many of the key concepts provided in this dissertation, and without his constructive criticism and suggestions, this work would not have been the same. As I attempted to overcome every obstacle to complete this research, Prof. Kuratsume answered all my queries. Owing to his support and guidance, I have been able to grow both personally and academically.

I am also grateful to my chief examiner, Assoc. Prof. Qi An, sub-chief examiner, Prof. Kenji Tahara, and my advisory committee member, Prof. Yusuke Ota, for their insightful comments.

I would also like to thank Prof. Ken'ichi Morooka, Assist. Prof. Akihiro Kawamura, and Assist. Prof. Shoko Miyauchi for their guidance, encouragement, and astute suggestions. Their advices were essential for completing my research and made me aware of new concepts and weaknesses in my studies. I am also grateful to all labmates I have met during the past five years. I am very thankful to my colleagues Asuka Egashira, Ukyo Katsura, Kouhei Kiyoyama, Naoki Setoguchi, Ryosuke Tatara, and Yumi Tsutsumida. They have encouraged me with their kindness and friendship from the past to the present.

In addition, this dissertation is partially supported by JST SPRING, Grant Number JPMJSP2136.

Finally, I would like to thank everyone who has contributed directly or indirectly to my studies and life, particularly my family. I am genuinely grateful to my father Manabu, my mother Kiyoko, my brother Yuki, my paternal grandparents Sadao and Yasuko, and my maternal grandparents Masayuki and Sueko Maeda. I would also like to thank all important people whose names I could not mention in this relatively short text.

Contents

List of Figures

List of Tables

1

Introduction

1.1 Background

Even though the society is growing, various social problems have become increasingly severe in recent years. For example, a shortage of labor is being caused by the declining birthrate, aging population, and depopulation of rural areas. Various entities have developed countermeasures, and machines and information technology have increased task efficiency. However, to solve this problem more effectively, it is necessary to “substitute” the tasks. Autonomous robots can realize this substitution. Autonomous robots are already active as the foundation of modern society in environments with limited conditions, such as factory automation. However, despite their high demand in traffic, security, agriculture, and nursing care, they are not as active in many human living and working environments as in industrial fields. The general rule for many such tasks is that they require locomotion. Consequently, to substitute these tasks, robots that move autonomously to perform tasks, or “autonomous mobile robots,” are

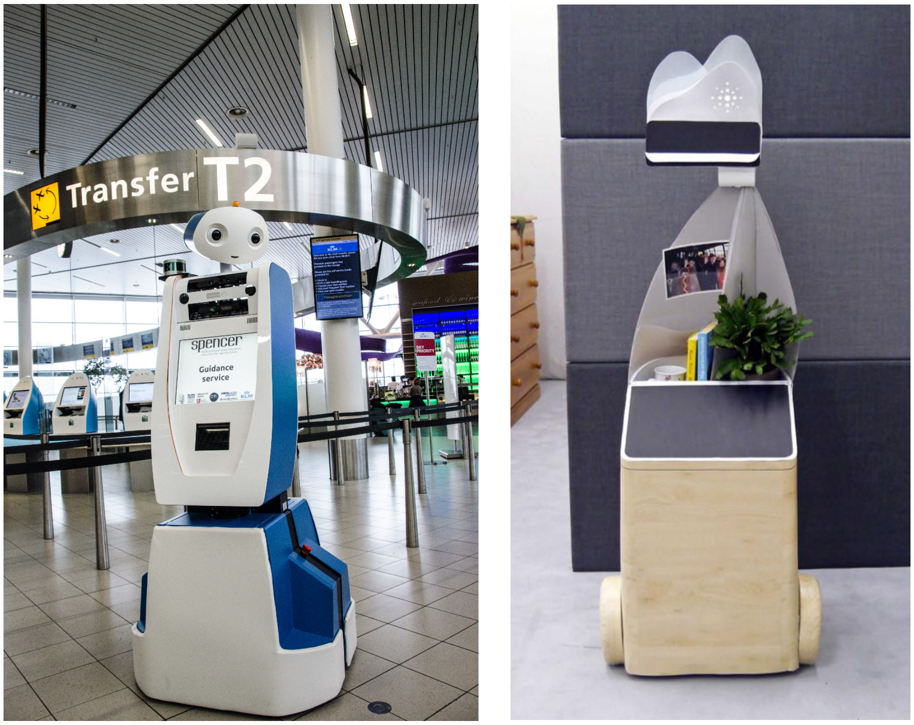


Figure 1.1: Examples of autonomous mobile service robots. Left figure shows SPENCER (Source: `http://www.spencer.eu/press.html`, 2021). Right figure shows SYMPARTNER (Source: `http://www.sympartner.de/fotos`, 2021).

needed instead of industrial robots that work at fixed points. Examples of research on service robots using autonomous mobile robots include SPENCER [2] and SYMPA-RTNER [3].

SPENCER is a robot that provides guidance at airports. This robot does not remain fixed and shows a user the route and the destination; it moves by itself to help a traveler make connections. With the development of society, the volume of air transportation is increasing, and in turn, airport schedules are becoming more complex, and the probability of the changes and delays is rising. At hub airports, in particular, there are a lot of people who are flying for the first time or who do not know the local language very well, and for them, making connections can be a challenging task. SPENCER robot can guide them from the arrival gate to passport control conveniently and efficiently.

SYMPERTNER is a robotic companion that lives with elderly, assisting them in their daily life and communicating with them. This robot is intended to provide support according to specific scenarios, such as morning and evening routines. Such a companion robot must interact with a user and actively approach the user to provide support, enjoyment, and cognitive and motor stimulation to enrich the daily life of the user. Therefore, it is necessary for robots to autonomously move to an appropriate position when performing services in a home environment.

In addition to the above two robots, an autonomous personal mobility scooter [4] to assist the mobility of people and an autonomous surveillance robot [5] to monitor indoor areas have been proposed. When such robots are fully introduced into the real world, the changes that will be caused in our lives are impactful.

As an example, let us consider a trip abroad. When we travel to an unfamiliar location, the first difficulty we face is commuting from the airport. When visiting an airport for the first time, there are many things that may be confusing, such as the nearest exit to the destination and deciding to choose a train or a bus. In such a case, a guide robot can help us move smoothly. In addition, there is a possibility that the location we want to visit may be inaccessible by public transportation. However, if we walk around in an unfamiliar location, we may become lost. In such cases, an autonomous personal mobility system can help us reach our destination without any concerns. When traveling abroad, security is a significant concern. Depending on the location, we may encounter factors that threaten our safety, such as theft or street thugs. A much safer trip can be achieved if surveillance robots patrol areas to prevent crimes, and if a crime does occur, they can rapidly contact the police. In addition, such robots will bring us food, talk to us, and provide entertainment to make our trip more interesting in restaurants and hotels.

These are only a few examples of how our lives will be enriched with the introduction of autonomous mobile service robots. Autonomous mobile service robots have the potential to realize such wonderful evolution of society without increasing the number of personnel.

1.1.1 Components of autonomous mobile robot

Although the above four robots are developed to perform completely different tasks at sites of completely different scales, they are similar in that they are equipped with various sensors to achieve their objectives and a software system configured to appropriately utilize the sensor suite. In particular, “localization,” “environmental perception,” and “path planning” are implemented as the basis for system realization, although their detailed implementation differs. These three elements are essential for implementing an autonomous mobile robot and also present very significant challenges.

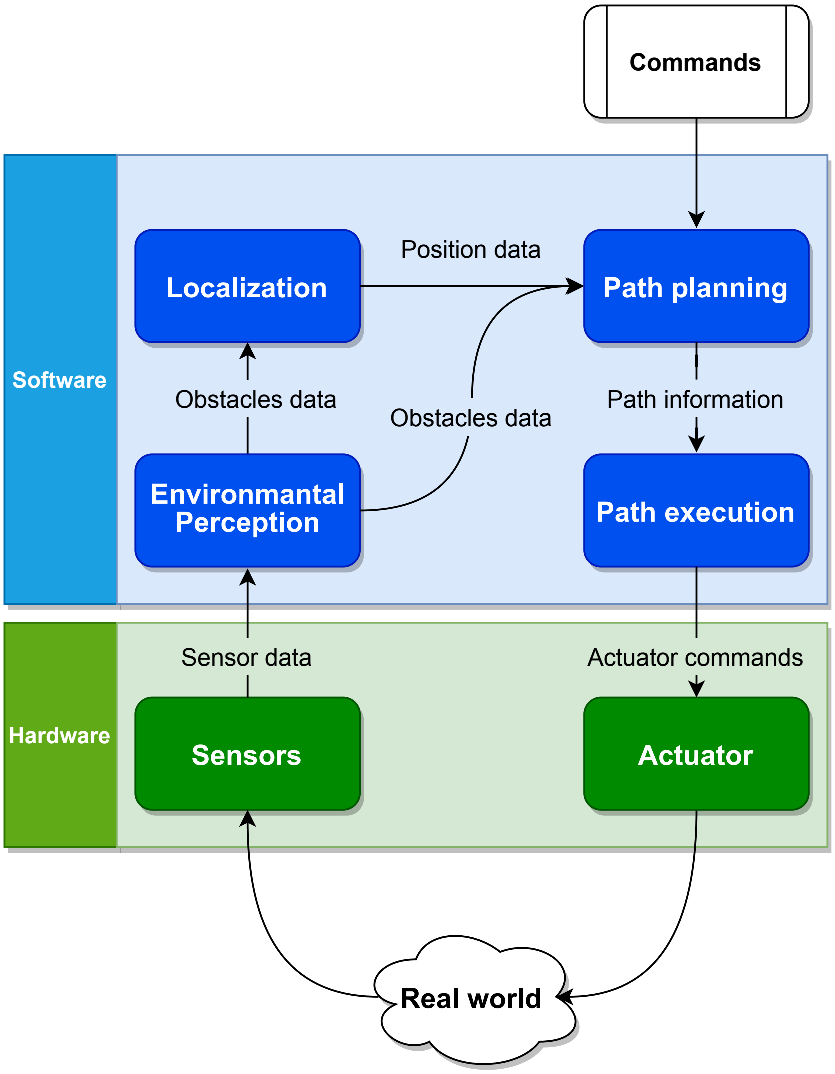


Figure 1.2: Framework of autonomous mobile robots.

The framework of autonomous mobile robots is shown in Figure 1.2. Such a robot observes the surrounding environment using sensors and obtains information. Environmental perception is achieved from the sensor information, and obstacles are recognized. The position of the robot itself is obtained using the sensor data. A path planner generates the path to achieve an objective from the obtained obstacle information and the position information of the robot and sends a command to the actuator to follow this path, finally the robots move.

Localization is a fundamental function of a robot to recognize its position. However, the degree of difficulty significantly varies depending on the environment and the task. It is difficult to obtain position information in daily life environments in which many people enter and exit.

Environmental perception is essential to recognize obstacles; however, our living environment contains static and dynamic obstacles. Without understanding the types of objects and deciding which obstacles to focus on, a robot may stay in a location based on prioritization to avoid collisions.

Path planning is a relatively simple function if only a path to a destination needs to be planned. However, when there are multiple pedestrians, a robot needs to consider the changes in the environment to plan a path, and planning a safe and efficient path is a difficult task.

These elements are the basis for realizing an autonomous mobile robot; however, the problem becomes more complex when considering the actual execution of the service. When performing advanced processing, the processing load may be extremely large for a robot to manage alone. Even if a robot can process such tasks independently, there are problems such as the decreased operating time due to the increased power consumption.

To solve this problem, it is necessary to support the processing of a robot with a remote server instead of the processing on the robot itself. However, this requires the communication environment to exchange information in real time. The communication environment is also vital for sending commands to the service robot and using the data acquired by the robot for service.

To realize these elements, system integration of a robot is crucial. Hardware, such as various sensors, and software to use them, must be combined appropriately to realize the required elements. In the next section, I will describe the components of an autonomous mobile robot in detail. I will also present some of the basic methods used in the proposed system.

1.1.2 Localization

It is necessary for a robot to automatically move to the position required to complete its tasks. To move to a target position, it is necessary to accurately know the difference between the target position and the position of the robot itself, and therefore, “localization” in which the robot recognizes its own position is important.

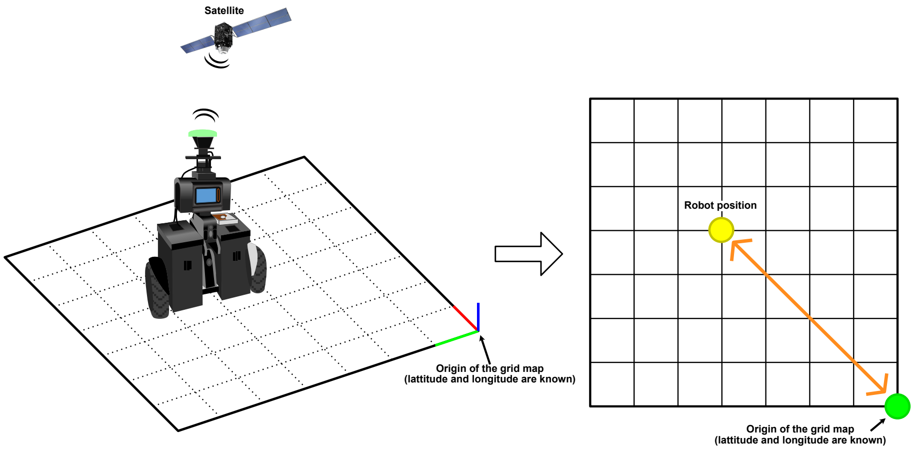

Satellite
Origin of the grid map
(lattitude and longitude are known)
Robot position
Origin of the grid map
(lattitude and longitude are known)

Figure 1.3: Conceptual diagram of localization using GNSS on grid map. Green circle indicates origin of the map where latitude and longitude are known, and yellow circle indicates position of robot. Yellow circle shows position of robot. Orange line represents difference between two points, and by determining this from difference in latitude and longitude, position of robot in form of grid map can be obtained.

The localization problem can be classified into two types: global and local. A local method is intended to compensate the travel distance errors during robot navigation. The initial position of a robot must be known, and if the robot loses its position, it is typically not recoverable. In a global method, the position of a robot can be identified without prior knowledge of its location. Specifically, the global method can deal with the problem of a kidnapped robot, where a robot is transported to an unknown position, and it can be recoverable with a large error that occurs unexpectedly. Initially, global localization appears to be a universal method; however, it is not perfect owing to the problems faced by sensors.

Typical sensors used in global methods include cameras, laser range finders, and a GNSS. The performance of cameras and laser range finders can be degraded by the occlusions caused by obstacles that are not assumed beforehand, and a GNSS can be degraded by the occlusions caused by obstacles and multipath. In practice, a combination of these two methods provides more robust localization.

In my implementation, I mainly use a GNSS for localization. However, to combine the system with the route generation method described below, it is necessary to convert the positioning information into the location information on a grid map instead of using it directly. Therefore, the latitude and longitude of the origin of the map are obtained in advance, and the position information on the map is calculated by computing the difference from it.

1.1.3 Environmental perception

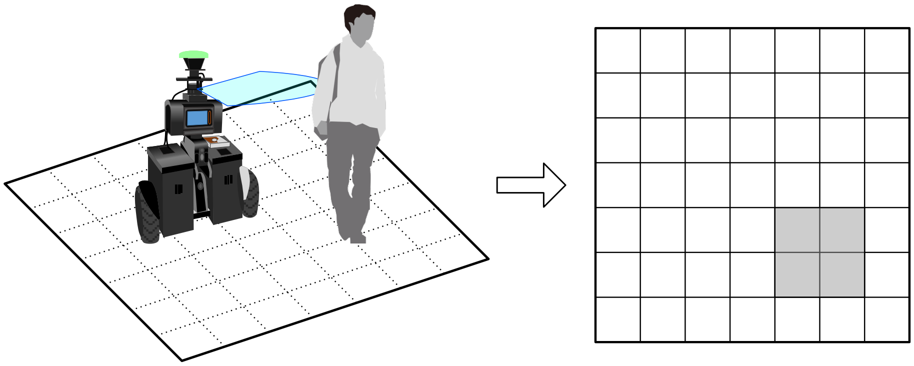


Figure 1.4: Conceptual diagram of object detection using range sensor (left side) and reflection to grid map (right side). Gray colored cell describes occupied cell by detected obstacle.

The environment is not typically configured for a robot, and various dangers can occur if a robot does not proactively understand its surroundings. In particular, object detection is necessary to avoid collisions between the robot and its surrounding objects. Because service robots moving in the daily life environment may collide with surrounding humans, the ability to avoid collisions is essential. The most basic object detection method is to use a range sensor. This method is simple; however, if a robot knows the distance to the object, it can plan its actions within a range where it will not collide with the object. Moreover, in a stationary environment, simply using a range sensor is frequently sufficient. One of the simplest methods for a robot to recognize the obstacles detected by the range sensors is to treat them as occupancy information in a grid map. Although it is important how obstacle information is reflected in the path planning, it is relatively easy to realize path planning that reflects the obstacle information by reflecting it in the map information, which is the basis for generating the path.

1.1.4 Path planning

Path planning is the final step for a robot to start moving autonomously and is an essential element. The generated path determines how a robot can move to a target position efficiently and safely. Compared to industrial manipulators, path planning for mobile robots is generally less complex. When considering the path planning for mobile robots, the degree of freedom is less than that for an industrial manipulator,

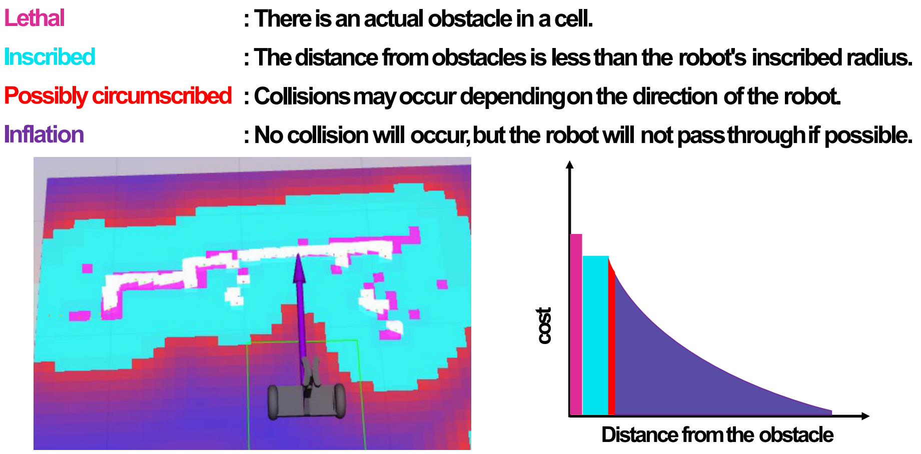

Lethal : There is an actual obstacle in a cell.
Inscribed : The distance from obstacles is less than the robot's inscribed radius.
Possibly circumscribed : Collisions may occur depending on the direction of the robot.
Inflation : No collision will occur, but the robot will not pass through if possible.
cost
Distance from the obstacle

Figure 1.5: Example of cost map in 2D map. The lower left figure shows an example of a cost generated on a map. The white area is the data measured by the distance sensor, and the cost is generated on the map based on this data. The cost varies with the distance from the obstacle, as shown in the lower right graph in the figure.

and in many cases, two dimensions are sufficient. In addition to simply generating paths, obstacle avoidance is important in path planning. In static environments, it is possible to deal with the existence of obstacles on a map by introducing a penalty. This is called a cost map and is commonly used in navigation using ROS and ROS2. Figure 1.5 shows an example of a cost map. However, in dynamic environments, it is not possible to deal with path planning simply by considering a simple cost map, and it is essential to predict the movement of dynamic objects and avoid them.

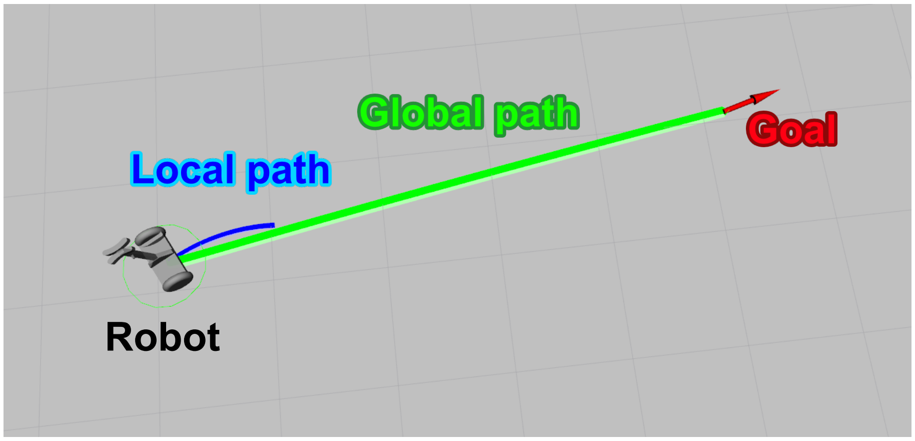
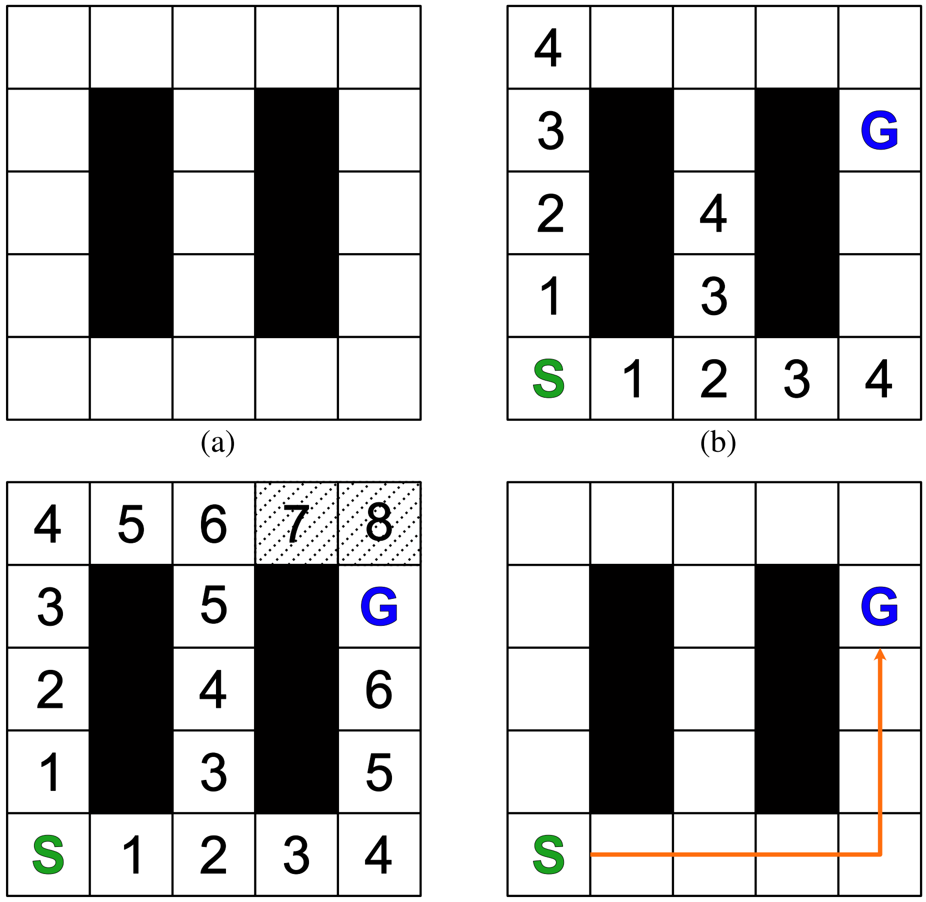


Figure 1.6: Examples of global and local path planning. The global path is generated first, and then the local path is generated with respect to it.



(a) (b) (c) (d)

Figure 1.7: (a)–(c) Process of calculating cost of Dijkstra's algorithm. Shaded areas are not actually explored in (c). (d) Generated path using Dijkstra's algorithm (orange line).

In real-world scenarios, I use a combination of methods that can generate fast, long-distance paths but do not consider robot speed constraints and other factors. Path generation methods are computationally expensive but consider the robot speed constraints, current speed, and risk of collision. They can be global and local path planning. Dijkstra's algorithm is frequently used as a global method, and the dynamic window approach (DWA) [6] is typically used as a local method. Figure 1.6 show examples of global and local path planning. Dijkstra's algorithm is a method for finding the shortest paths between nodes in a graph. Because the shortest path can be calculated in a linear time, it is possible to generate long-distance paths relatively rapidly. Using this method in a grid map, the shortest path from the current position of the robot to the target position can be generated. Figure 1.7 shows an example of path

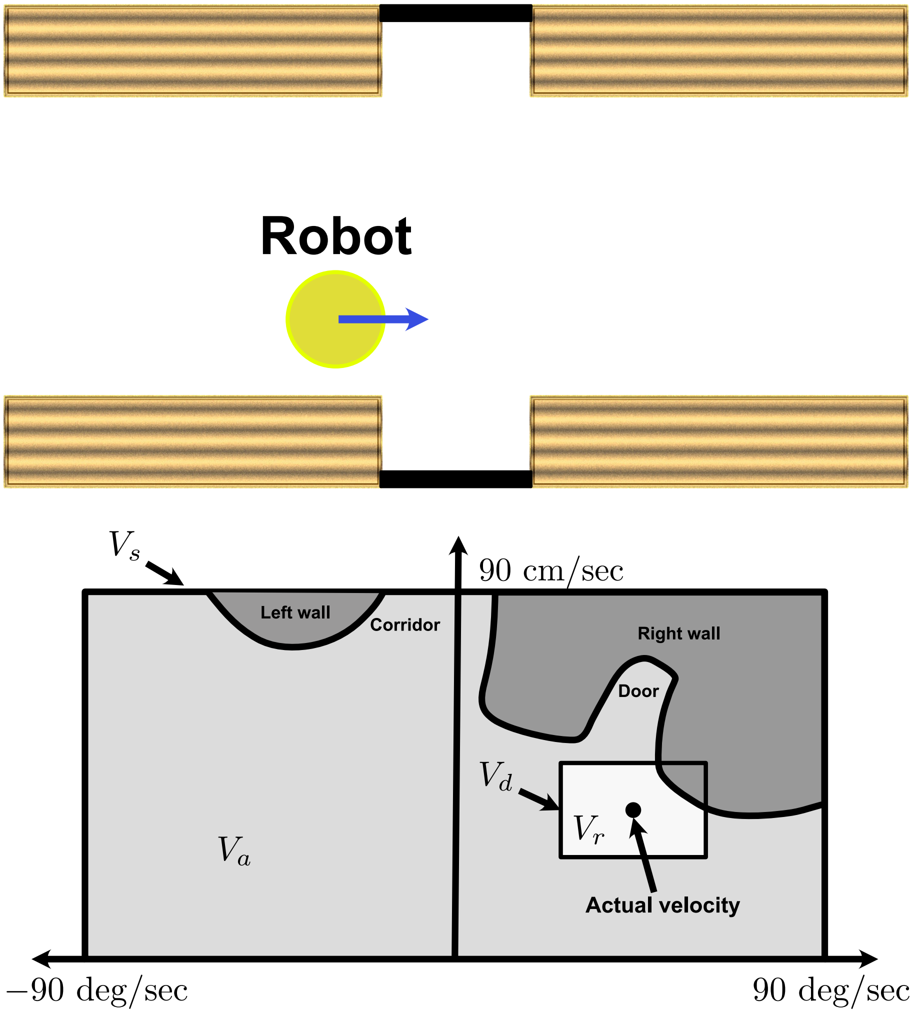


Figure 1.8: Conceptual diagram of robot moving in environment flanked by walls (upper side) and relation between velocity space and DWA for that scenario (lower side).

generation using the Dijkstra's algorithm. A DWA searches in the velocity space, with translational velocity v and rotational velocity ω as the axes, and calculates the speed at which it can reach the target position while avoiding possible collision areas. Figure 1.8 shows the relation between the velocity space of DWA for a scenario of a moving robot. In this figure, V_s is the maximum speed that the robot can take, V_a is the speed at which the robot does not collide with obstacles, and V_d is the dynamic window. V_a

and V_d are represented by equations (1.1) and (1.2), respectively.

$$V_a = \{v, \omega | \, v \leq \sqrt{2 \cdot \mathrm{dist}(v, \omega) \dot{v}_b} \land \omega \leq \sqrt{2 \cdot \mathrm{dist}(v, \omega) \dot{\omega}_b}\} \tag{1.1}$$

$$V_d = \{v, \omega | \, v \in [v_a - \dot{v}t, v_a + \dot{v}t] \land \omega \in [\omega_a - \dot{\omega}t, \omega_a + \dot{\omega}t]\} \tag{1.2}$$

where $\mathrm{dist}(v, \omega)$ is the distance to the nearest obstacle, $\dot{v}_b, \dot{\omega}_b$ is the acceleration that causes the collision, and v_a and ω_a are the actual velocities. Based on the above, the velocity space to be searched V_r is,

$$V_r = V_s \cap V_a \cap V_d \tag{1.3}$$

Among the trajectories obtained from this search space, the one that minimizes the objective function, G, is selected. In combination with global planning, the objective function is as follows.

$$G = \alpha \cdot \mathrm{dist}_{\mathrm{path}} + \beta \cdot \mathrm{dist}_{\mathrm{goal}} + \gamma \cdot \mathrm{cost} \tag{1.4}$$

where $\mathrm{dist}_{\mathrm{path}}$ is the distance from the endpoint of the trajectory to the global path, $\mathrm{dist}_{\mathrm{goal}}$ is the distance from the endpoint of the trajectory to the goal, and cost is the value of the largest cost along the trajectory. α, β, and γ are parameters to be set in advance.

1.1.5 Network environment

If autonomous mobile robots are to be operated as service systems, sending commands and monitoring them from remote locations for service management and operation are necessary, as shown in Figure 1.9. These requires communication, and it is essential to provide a wireless communication environment to prevent impeding of the free movement of the autonomous mobile robot. The most common wireless communication methods are the fourth-generation mobile communication system (4G) and Wi-Fi. 4G is mainly used for smartphones, and its well-developed infrastructure allows it to communicate over a wide area; however, it is frequently inferior to Wi-Fi in terms of the communication speed. On the other hand, Wi-Fi is faster than 4G, and if the communicating modules are connected to the same router, they can communicate with a low latency without using the Internet. However, owing to its limited communication range, the range of action of an autonomous mobile robot becomes narrower.

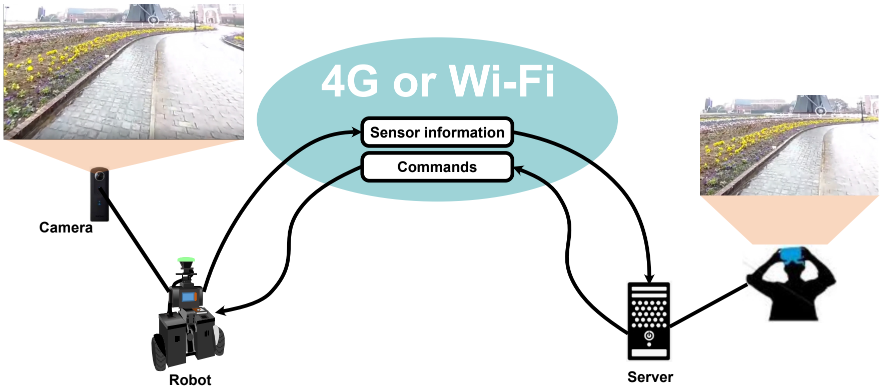


Figure 1.9: Conceptual diagram of communication for autonomous mobile service robots. Robot receives commands from server managing service and server receives image data for monitoring, and manager confirms scenario. The person on the right side is watching the image acquired by the robot through a VR head-mounted display as an example.

In this way, each method has its advantages and disadvantages, so it is necessary to build an appropriate network environment according to the services to be realized and the size of the data to be handled.

1.1.6 System integration for autonomous mobile robot

To realize an autonomous mobile robot that operates in the real world, the hardware for robot and system integration is essential. In particular, sensors for realizing the above three elements mentioned above, particularly object detection/tracking and localization, are essential. For object detection and tracking, it is crucial to measure the distance. Therefore, sensors such as two-dimensional (2D)/three-dimensional (3D)-LiDARs, depth cameras, and ultrasonic sensors are used. Concurrently, it is vital to measure the position and pose information of the robot for localization. For this purpose, optical trackers, beacons, inertial measurement units (IMUs), wheel encoders, and a GNSS are used. Figure 1.10 shows an example of the implementation of an autonomous mobile robot. In this example, the robot that uses a GNSS to estimate its position and a 2D-LiDAR to detect obstacles. The robot is also equipped with a Wi-Fi router to send commands from a distance and to monitor its status, and it communicates with a computer at a remote location.

Probabilistic filters are also essential elements for integrating sensors into robots. Because it is difficult for a single sensor to acquire all information, installing sensors

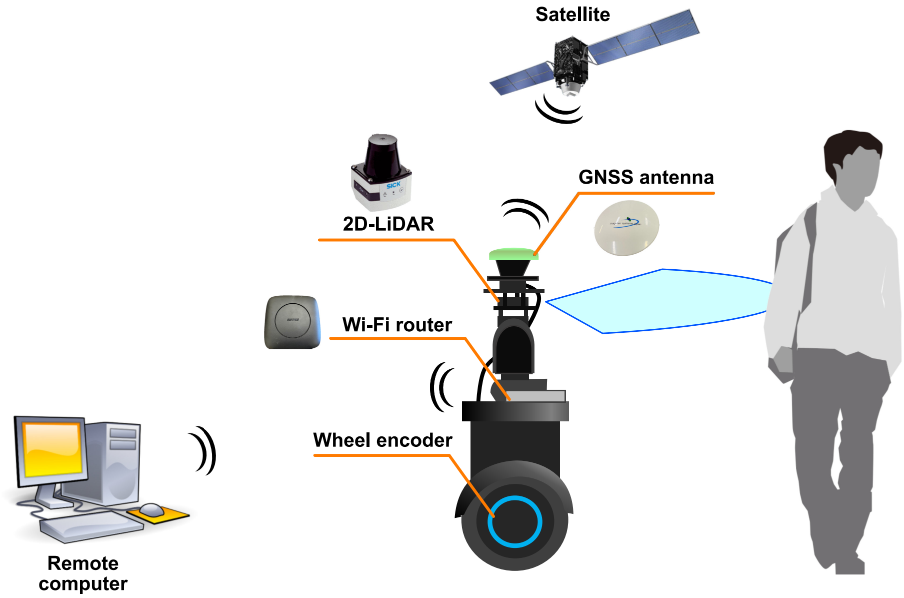


Figure 1.10: Example of implemented system for autonomous mobile robot.

that can acquire different information or sensors that can acquire the same information but have different characteristics is necessary for sensors to compensate for their shortcomings. Probabilistic filters are used for sensor fusion, which integrates information from multiple sensors. In particular, an extended Kalman filter (EKF) is frequently used to integrate multiple sensor information in localization. An ordinary Kalman filter deals only with linear systems, whereas an EKF can be applied to nonlinear systems because it uses a Taylor expansion for linear approximation. The position and velocity of a mobile robot over time can be described as a nonlinear dynamical system.

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}) + \mathbf{w}_{k-1} \tag{1.5}$$

where $\mathbf{x}_k$ represents the state of the system at time k and f is the nonlinear state transition function. In addition, $\mathbf{w}_{k-1}$ is the system noise, which follows a normal distribution. The state vector, $\mathbf{x}$, considers the system in a 2D space and has eight dimensions: position (x, y), attitude (yaw), velocity $(\dot{x}, \dot{y})$, angular velocity $(y\dot{a}w)$, and acceleration $(\dot{x}, \dot{y})$. Furthermore, the measurement equation can be expressed as follows:

$$\mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{v}_k \tag{1.6}$$

$\mathbf{z}_k$ is the measured value at time k and h is the nonlinear sensor model. $\mathbf{v}_k$ is the measurement noise following a normal distribution. Herein, I describe the processing flow of an EKF, which is divided into prediction and filtering steps. In the prediction step, the prior estimate, $\hat{\mathbf{x}}_{k-1}$, and the prior error covariance matrix, $\hat{\mathbf{P}}_k$, are obtained from the state estimate, $\mathbf{x}_{k-1}$, of the previous time. In the filtering step, the state estimate, $\mathbf{x}_k$, and the posterior error covariance matrix, $\mathbf{P}_k$, are obtained using the prior estimate and the measured values. The time update equation for the prediction step is expressed below.

$$\hat{\mathbf{x}}_k = f(\mathbf{x}_{k-1}) \tag{1.7}$$

$$\mathbf{F}_{k-1} = \left.\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_{k-1}} \tag{1.8}$$

$$\mathbf{H}_k = \left.\frac{\partial h(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_k} \tag{1.9}$$

$$\hat{\mathbf{P}}_k = \mathbf{F}_{k-1}\mathbf{P}_{k-1}\mathbf{F}_{k-1}^{\mathrm{T}} + \mathbf{Q} \tag{1.10}$$

where $\mathbf{F}_{k-1}$ and $\mathbf{H}_k$ are linear approximations of the nonlinear state transition function and the nonlinear sensor model, respectively. $\mathbf{Q}$ is the covariance matrix of the system noise. The time update equation for the filtering step is expressed below.

$$\mathbf{K} = \hat{\mathbf{P}}_k\mathbf{H}^{\mathrm{T}}(\mathbf{H}\hat{\mathbf{P}}_k\mathbf{H}^{\mathrm{T}} + \mathbf{R})^{-1} \tag{1.11}$$

$$\mathbf{x}_k = \hat{\mathbf{x}}_k + \mathbf{K}(\mathbf{z} - \mathbf{H}\hat{\mathbf{x}}_k) \tag{1.12}$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}\mathbf{H})\hat{\mathbf{P}}_k \tag{1.13}$$

Here, $\mathbf{K}$ is the Kalman gain and $\mathbf{R}$ is the measured covariance matrix. These two steps are repeated to integrate the data and estimate the state. As an example, when using a GNSS, an IMU, and wheel encoders to integrate information in an EKF for

Table 1.1: Example of configuration of sensors for localization using EKF ("✓" implies True and "-" denotes False).

Sensors	Configuration					
	x	y	yaw	$\dot{x}$	$\dot{y}$	$y\dot{a}w$
GNSS	✓	✓	-	-	-	-
IMU	-	-	✓	-	-	✓
Wheel encoder	-	-	-	✓	✓	✓

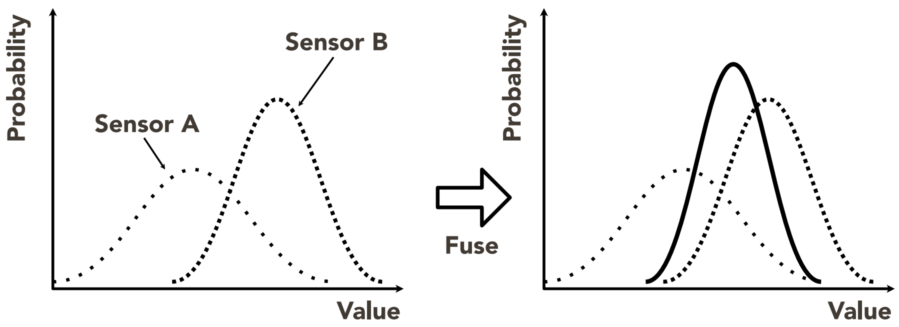


Figure 1.11: Conceptual diagram of multiple sensor data fusion. Bold line in right figure shows data after fusion.

localization, the configuration of the information that can be obtained from each sensor is summarized in Table 1.1. When the same information can be obtained from multiple sensors, it is synthesized according to the product of the normal distribution, as shown in Figure 1.11.

1.2 Research Aim

Implementation of an autonomous mobile robot is significant for it to act freely in the real world. In Sections 1.1.1–1.1.6, I described the elements required for an autonomous mobile robot. The overall implementation of the hardware configuration, software, and algorithms is an important issue in real-world robotics. My first aim is to implement actual service robots, a tour guide robot and a co-experience robot, using the components mentioned above of an autonomous mobile robot as well as advanced hardware and information technology. Tour guide robots are designed to replace human guides in theme parks, and their practical application is expected to solve labor shortage. Co-experiencing robots aim to provide people with mobility difficulties with an experience as if they were moving freely by sharing the information experienced by the robot on behalf of the human operator. I also conduct experiments in a target environment in which the robots perform the service.

For autonomous mobile service robots to perform tasks in human living environments, it is important that they can move smoothly and safely in the presence of people. The second aim is to address this issue using machine learning methods. In particular, by applying deep reinforcement learning, I propose a new method for mobile robot navigation in a dynamic environment by pedestrians. In such a problem, it is essential to predict how a person will move; however, the behavior of a person is influenced by the surrounding pedestrians and, of course, robots. I focus on the fact that the behavior of pedestrians is affected by the action of a robot.

1.3 Contributions and Outline

The studies in this dissertation focus on the implementation and navigation of an autonomous mobile service robot. In terms of implementation, a tour guide robot and a co-experience robot in a theme park are proposed using the latest hardware and information technology. For navigation, a navigation method based on deep reinforcement learning is proposed, considering that the behavior of a robot influences the behavior of pedestrians in a dynamic environment. The remainder of this dissertation is structured into two main parts.

Chapter 2 describes the development of the tour guide robot using the QZSS and a co-experience robot using the 5G in a theme park. The QZSS is the most advanced GNSS operated by Japan, and it enables centimeter-level positioning with only one set of modules using a unique error correction method called CLAS. In the study, compare the accuracies of the QZSS and existing high-precision GNSSs and show the implementation of the guide robot by experiments in an actual theme park environment. Concurrently, 5G is a mobile communication system developed worldwide in recent years, and it enables a higher capacity wireless communication than the conventional communication environment. Using 5G, I develop a system to share 360-degree 4K images as a robot experience to remote locations with high speed and low latency. I also show the experimental results of this co-experience robot system in a theme park environment.

In Chapter 3, a method for navigating mobile robots in dynamic environments using deep subject learning is proposed. The proposed method is based on predictive state representation (PSR), which can predict the changes in an environment after the action of an agent in a time series. In addition, a new PSR model is proposed to better deal with the dynamic environment caused by pedestrians. The effectiveness of this new model compared with that of the previous model is shown by experiments. In addition, two methods of integrating PSR states are proposed to deal with the case in which the number of pedestrians differs between training and testing. The performance of the two methods is compared by experiments.

Finally, Chapter 4 presents the conclusions and future challenges and prospects of the method proposed in this dissertation.

2

Development of Tour Guide and Co-experience Robot System

The service robot is an effective solution for various social problems, such as the problems of a super-aged society, labor shortages caused by depopulation in the countryside. Tasks that are expected to be replaced by service robots include surveillance, cleaning, and guiding. Recently, new and advanced technologies, which are important factors in realizing intelligent service robots, are being actively developed. The Quasi-Zenith Satellite System (QZSS) [7] and the 5th-generation mobile communication system (5G) are among these technologies. I intend to use these technologies and develop two types of service robot system, namely, a tour guide robot system and a co-experience robot system. Figures 2.1 and 2.2 show conceptual diagrams of these systems.

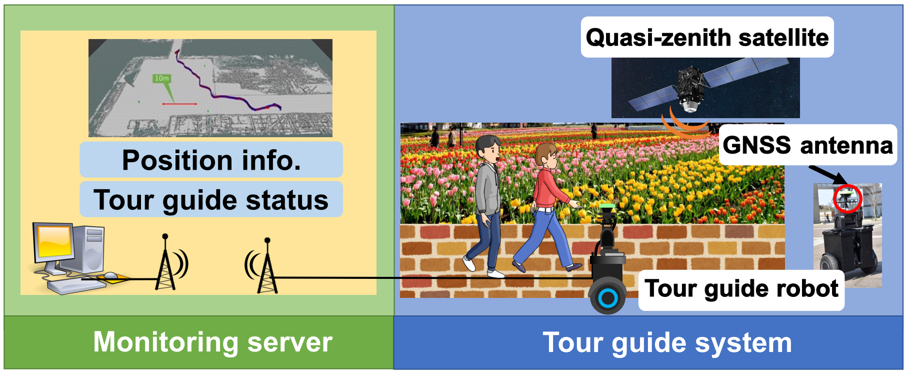


Figure 2.1: Conceptual diagrams of the tour guide system. The robot conducts a tour using the QZSS, and some information obtained from the robot is monitored from the server. The blue area on the right side shows a QZSS-equipped tour guide robot performing tour guide tasks, and the green area on the left side shows the robot's location and guiding status being monitored from a remote location.

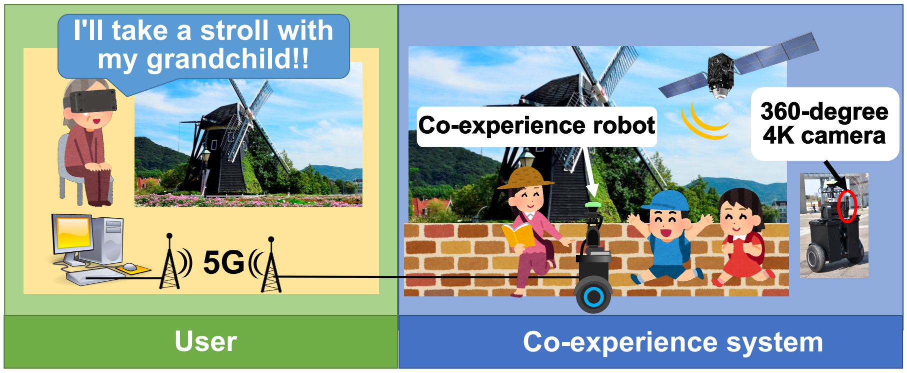


Figure 2.2: Conceptual diagrams of the co-experience system. The user can share an experience (360-degree 4K video) of the robot remotely through the 5G network. The blue area on the right shows a co-experiencing robot equipped with a 360-degree 4K camera moving around the theme park, and the green area on the left shows a person viewing the video acquired by the robot from a remote location via a 5G network.

2.1 Introduction

Localization is one of the most important and fundamental functions for an autonomous service robot like the tour guide robot. In an outdoor environment, a global navigation satellite system (GNSS), in particular, the Global Positioning System (GPS), is the most popular technique. However, the accuracy of a GNSS is approximately 10 meters, which is inadequate for navigation of an autonomous service robot.

Therefore, the real-time kinematic GNSS (RTK-GNSS) or a virtual reference station (VRS) that provides centimeter-class positioning is used for accurate navigation of an autonomous service robot, such as a personal mobility vehicle or a delivery robot.

A number of autonomous robot systems using the RTK-GNSS have been proposed [8, 9, 10, 11, 12].

In [8], the authors proposed a robust and precise localization system that achieves centimeter-level accuracy in diverse city scenes. In this system, the measurement of the RTK-GNSS, LiDAR, and IMU are synthesized in the sensor fusion framework using an error-state Kalman filter. In [9], the authors proposed a high-precision localization method by treating the global pose estimation problem as a pose graph optimization problem. Both the RTK-GNSS and wheel odometry are used as constraints of the pose graph. In [10], the authors proposed a sensor fusion method for 3D mapping and localization using multiple heterogeneous and asynchronous sensors. In this system, the authors first create an accurate prior map by ORB-SLAM [13] and LOAM [14] using a vehicle that has an RTK-GNSS sensor unit. After creating the prior map, the map is used for localization in the GPS frame of reference without the use of GPS, and thus localization in GPS-denied environments, such as tunnels or parking garages, is also performed. In [11], an integrated framework for underground 3D mapping using a mobile rover is proposed. This framework conducts 3D underground mapping based on ground penetrating radar (GPR) data. In this system, the RTK-GNSS is used for accurate geo-reference. In [12], the underwater localization system for an underwater mining vehicle (MV) and a surface launch and recovery vessel (LARV) was proposed. The LARV is used for supporting the MV. In this system, the RTK-GNSS is used for the localization of the LARV using RTKLIB [15].

The QZSS began operating on November 2018 in and around Japan. The QZSS provides high-accuracy position information and a localization error of less than 10 centimeters by using electronic reference points and four quasi-zenith satellites (QZSs). These satellites transmit signals not only for localization but also for error correction using the electronic reference points. Therefore, we do not need the base station for the RTK-GNSS, and thus the QZSS is easy to use for centimeter-class positioning, as compared with the RTK-GNSS. Accordingly, the QZSS is suitable for mobile robots that move over a wide area, such as the tour guide robot system proposed herein. Since the QZSS can be used without the communication between the two stations required for the RTK-GNSS, the QZSS is suitable for mobile robots that move over a wide area, such as the same proposed system.

In particular, due to the rapid development of communication technology, various systems for conducting service tasks using robots over networks have been constructed [16, 17, 5]. Regarding communication technology, 5G is being developed and introduced as a next-generation wireless communication technology that enables large-capacity data to be handled by wireless communication. Wireless communication is an indispensable technology for autonomous mobile service robots, and, by using 5G, more advanced services are possible. In the present study, I develop a co-experience system that allows us to experience the surrounding image and sound around the robot using a VR head-mounted display from a remote location by transferring the 360-degree 4K video taken by the robot using 5G. By using this system, for example, older people with lower-limb disabilities can enjoy experiences from a remote place like taking a stroll with a grandchild, using a robot, and due to the high-capacity communication of 5G, high immersion is realized.

The key elements, the QZSS, 5G, 360-degree 4K camera, and VR head-mounted display, have been developed individually. The main contribution of this study is the integration of these systems into a mobile service robot system that operates in the real world. I assume the following requirement specifications for target systems and construct systems to achieve them. TG1 to TG3 are the requirement specifications for the tour guide system, and CE1 and CE2 are the requirement specifications for the co-experience system.

Tour guide system

TG1. *Guiding the way while moving autonomously*
The target system in this study assumes a theme park. This environment consists of various features, such as a place surrounded by buildings and an open flower garden, this study is especially targeted at the open environment. In particular, localization has difficulty in such environments. Localization using cameras and LiDARs are not working well in open environments because of the difficulty in acquiring features. On the other hand, the GNSS has an advantage in open environments. However, the existing high-precision GNSS, RTK-GNSS, has a limitation for positioning, the communication range between base stations and mobile stations. Since theme parks are vast environments, robots need to be able to move over a larger area to provide tour guide services. It can be expected that situa-

tions where the park is crowded with pedestrians in the environments. The communication between base stations and mobile stations will be more restrained in such cases, and the limitations will be more severe. In order to solve this problem, I adopt the QZSS, which does not have such constraints.

TG2. *Explanation and announcement*
Explaining the facilities and attractions is an essential part of the guide service at theme parks. Since tour guides involve movement, it is necessary to make appropriate explanation according to where the robot is currently and where it will move. It is also necessary for the user to be able to know the current situation, such as whether the robot is stand-by or not. Furthermore, in situations where the environment is crowded with pedestrians, it may be necessary to encourage the surrounding pedestrians to give way. To achieve these, I implement a tour guide application that can manage the robot's current location, destination and state, and provide appropriate voice explanations and announcements.

TG3. *Interface to respond to user requests*
It is also important to provide services in response to user requests. Especially for tour guides at theme parks, it is crucial to take requests without directly operating the terminal so that they can concentrate on sightseeing. I develop a system for accepting user requests through voice recognition to achieve a system that allows user to send requests to the robot in a casual manner while enjoying sightseeing.

Co-experience system

CE1. *Transmission of high resolutional video*
The target system in this study present the video acquired from the camera mounted on the robot to the user as the robot's experience. In order to share a higher quality experience, it is important to transmit the video with higher resolution. The mainstream wireless communication methods are the 4G and the Wi-Fi. However, The 4G is not suitable for large-volume communication, and the Wi-Fi has a large limitation in communication range. In order to cope with these limitations, I use 5G in this study.

CE2. *Presenting the transferred video to the user in an immersive way*
In order to create an immersive experience, we need a system that allows the user to recognize the video sent from the robot as if it were their own point of view. Merely displaying data acquired from an ordinary camera on an ordinary display does not provide a high level of immersion. In this study, I use the 360-degree 4K camera and the VR head-mounted display to achieve a high level of immersion.

In the rest of this section, I compare the positioning performance of the RTK-GNSS and the QZSS and verify the stability and accuracy of the QZSS in an outdoor environment. This is to show that the new module QZSS and its centimeter positioning with only a single module can be used for autonomous mobile robots. In addition, I introduce the configuration of a tour guide robot system using the QZSS and a co-experience system using the 5G network. Finally, I present an experiment for the tour guide system and co-experience system in a theme park.

2.2 Centimeter-class positioning by GNSS

For high-accuracy measurement using a GNSS, error correction is very important. Errors include the clock error of the satellite, the clock error of the receiver, the position error of the satellite, ionospheric delay, tropospheric delay, the effect of multiple paths, and the noise of the receiver [18] [19]. In this section, I explain the measurement procedure and the error correction system for the RTK-GNSS and the QZSS.

2.2.1 Real-time kinematic GNSS

The RTK-GNSS uses two modules: a base station and a rover station. Using these modules, the RTK-GNSS calculates the double difference of the carrier phase and achieves high-accuracy measurement. The double difference of the carrier phase is calculated using the carrier phase from two satellites to base and rover stations. Here, the carrier phase data arriving at the base station from satellite A and satellite B are denoted as (ϕ_r^A and ϕ_r^B), respectively. The double difference of the carrier phase $D\phi_{br}^{AB}$ is calculated as $D\phi_{br}^{AB} = (\phi_r^A - \phi_b^A) - (\phi_r^B - \phi_b^B)$. This calculation removes the clock errors of the satellites and the receiver. In addition, if the distance between the base and rover stations is less than a certain value, then the ionospheric delay and tropospheric delay can be removed. Furthermore, by using information on pseudo-ranges between multiple satellites and receivers, we can determine the integer ambiguities remaining as errors and thereby realize centimeter-class positioning. In the present study, I use MJ-2001-GL1 (Magellan Systems Japan Inc.; Figure 2.3) as an RTK-GNSS module in the experiment.

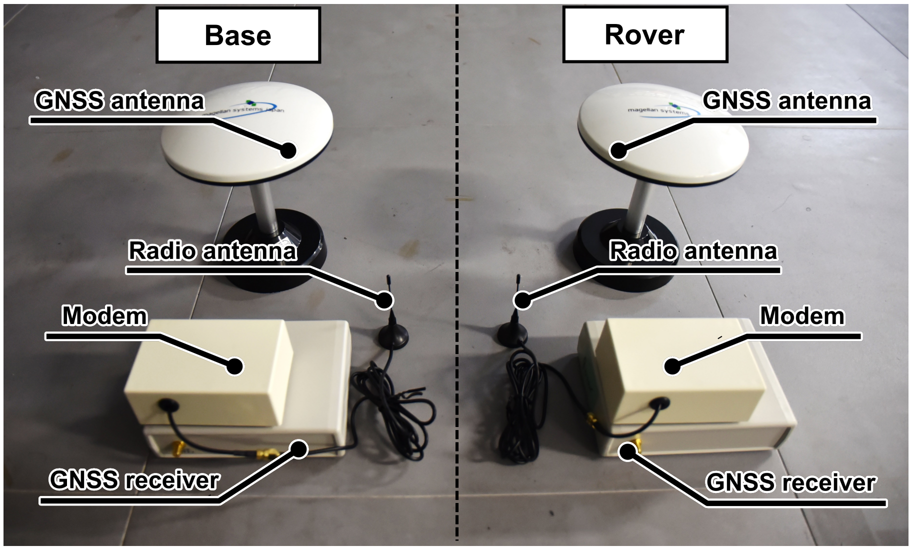


Figure 2.3: Real-time kinematic GPS module (MJ-2001-GL1, Magellan Systems Japan Inc.).

2.2.2 Quasi-Zenith Satellite System

The QZSS uses four quasi-zenith satellites, referred to as the "Michibiki" constellation, and began operating in November 2018 in Japan. Whereas the RTK-GNSS uses two sets of modules, the QZSS provides highly accurate positioning with only one module, consisting of an antenna and a receiver. As explained above, the RTK-GNSS uses the correction signal measured by the base station. On the other hand, the QZSS generates an error correction signal using observation data at electronic reference points placed very densely in Japan, and the correction is performed by transmission to the user terminal via the satellites. This correction method is referred to as centimeter-level augmentation [20] [21], and centimeter-class positioning has been realized in and around Japan. The quasi-zenith orbit is shown in Figure 2.4. This orbit is an asymmetrical trajectory, and each quasi-zenith satellite follows this trajectory in the course of one day. By constructing this quasi-zenith orbit with four satellites,

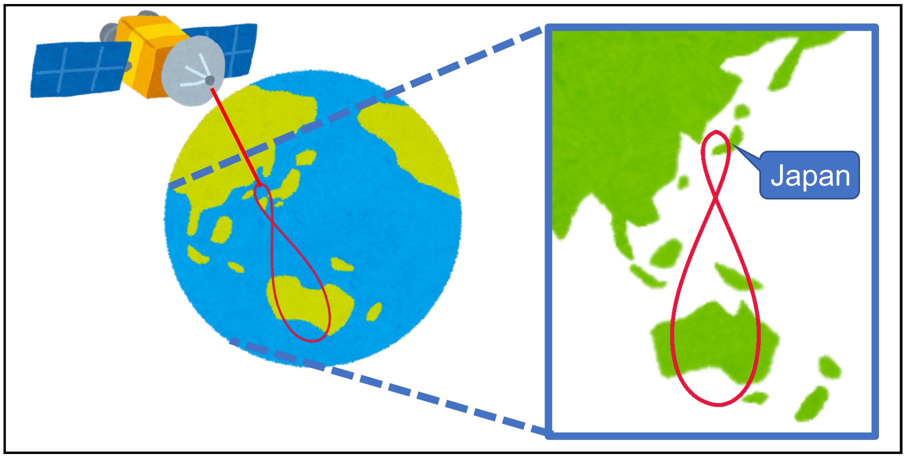


Figure 2.4: Quasi-zenith orbit. The red-colored trajectory is the quasi-zenith orbit [1]. The trajectory covers Japan and Australia.

and shifting their positions in time, a high elevation angle to at least one quasi-zenith satellite can always be obtained in and around Japan. In the present study, I used the QZSS module called AQLOC-V (Mitsubishi Electric Inc.; Figure 2.5) and MJ-3021-GM4-QZS-EVK (Magellan Systems Japan, Inc.; Figure 2.6). The AQLOC-V is used for experiment compareing performance of the RTK-GNSS and the QZSS and the first version of the developed robot system. The MJ-3021-GM4-QZS-EVK is used for the second version of the developed robot system because the compactness of the module is necessary. The robot systems are described in detail in the section describing the configuration of the developed robot system.

2.2.2.1 Centimeter-level augmentation service

The centimeter-level augmentation service (CLAS) is a unique function of the QZSS. The “Michibiki” constellation adopts a state space representation (SSR) method [22] for the CLAS and realizes centimeter-class positioning using the L6 signal, which is an auxiliary signal of quasi-zenith satellites. In the centimeter-class augmentation information generated at the control segment, a dynamic error model called the state space model (SSM) is used based on observation data of the electronic reference point network. Each error amount, such as the clock error, the satellite orbit error, ionospheric delay, tropospheric delay, and signal bias, is generated as an SSR.

The flow of the centimeter-level augmentation is shown in Figure 2.7. Based on the positioning information at the electronic reference point for which the latitude

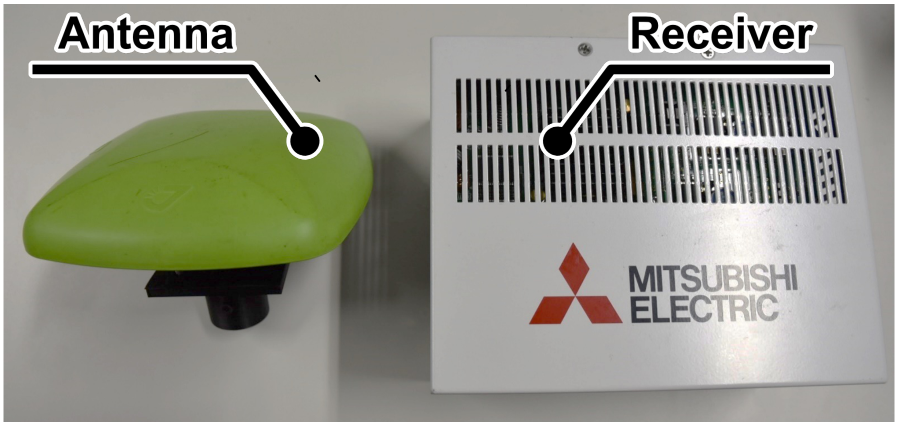


Figure 2.5: Quasi-Zenith Satellite System module AQLOC-V (Mitsubishi Electric Inc.). This module was used for the first version of the developed system.

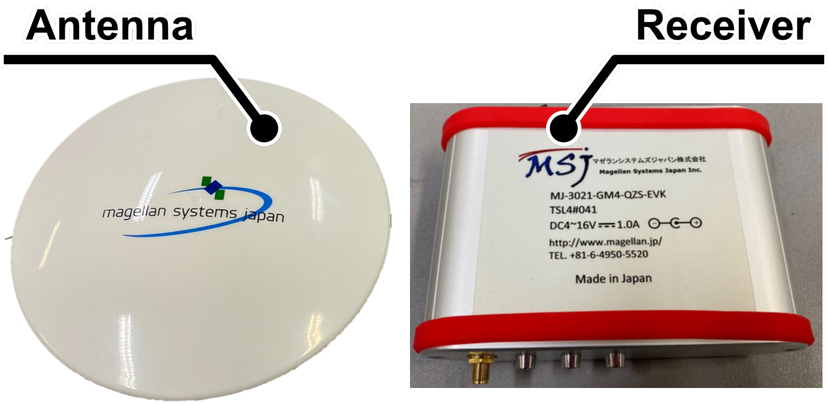


Figure 2.6: Quasi-Zenith Satellite System module MJ-3021-GM4-QZS-EVK (Magellan Systems Japan, Inc.). This module was used for the second version of the developed system. This module is more compact than the AQLOC-V.

and longitude are known, the correction information for removing the error is created at a facility called the monitoring station and transmits the information to the quasi-zenith satellites via the antenna of the tracking station. Then, by receiving the correction information simultaneously with the positioning signal on the user terminal side, centimeter-class positioning is realized.

2.3 Accuracy measurement experiments

In order to verify the measurement accuracy of the QZSS, I compared the positioning performance the RTK-GNSS and the QZSS in the stand-still state and in motion in

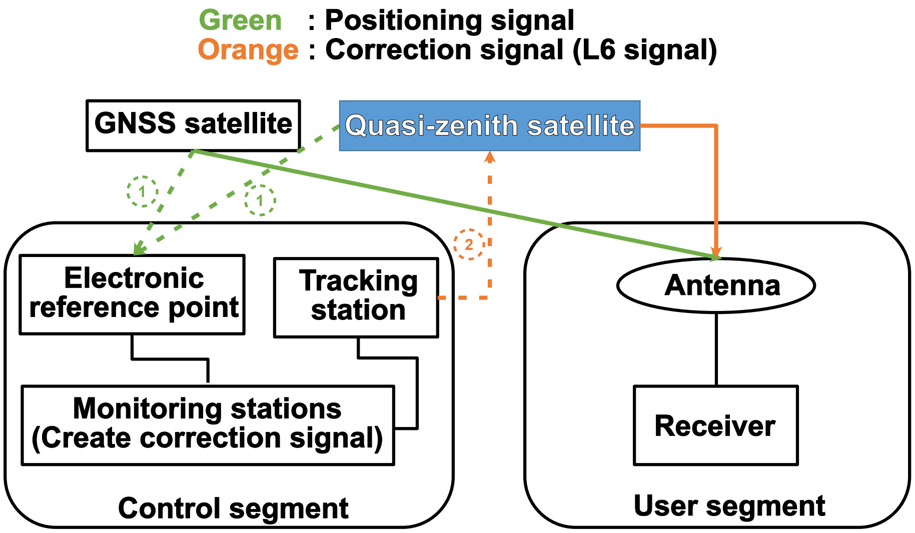

Green : Positioning signal
Orange : Correction signal (L6 signal)
GNSS satellite
Quasi-zenith satellite
1
1
2
Electronic reference point
Tracking station
Monitoring stations (Create correction signal)
Control segment
Antenna
Receiver
User segment

Figure 2.7: Flow of centimeter-level augmentation. The solid line represents the signal used by the user. The dashed line shows the flow of correction signals to the quasi-zenith satellites, and the numbers indicate the order of the signals.

an open-sky environment and in a partially obscured environment in which buildings block portions of the sky.

2.3.1 Measurement accuracy in the stand-still state in an open-sky environment

In this experiment, the precision of the RTK-GNSS and the QZSS were compared by the variance of positioning data from the average value in the stand-still state. I performed measurements for each module at the same location and compared about 3,000 data acquired during a 5 minute measurement for each module.

The results are shown in Figure 2.8. Based on these results, the RTK-GNSS can perform positioning more stably than the QZSS in the stand-still state. One reason for this is the difference in the mechanism of position information correction, i.e., that the base station is placed close to the rover station in the RTK-GNSS. However, the errors of the QZSS are less than approximately ±4 cm and satisfy most applications of autonomous service robots. A more detailed discussion will be presented in the following section for performance comparison of the RTK-GNSS and the QZSS.

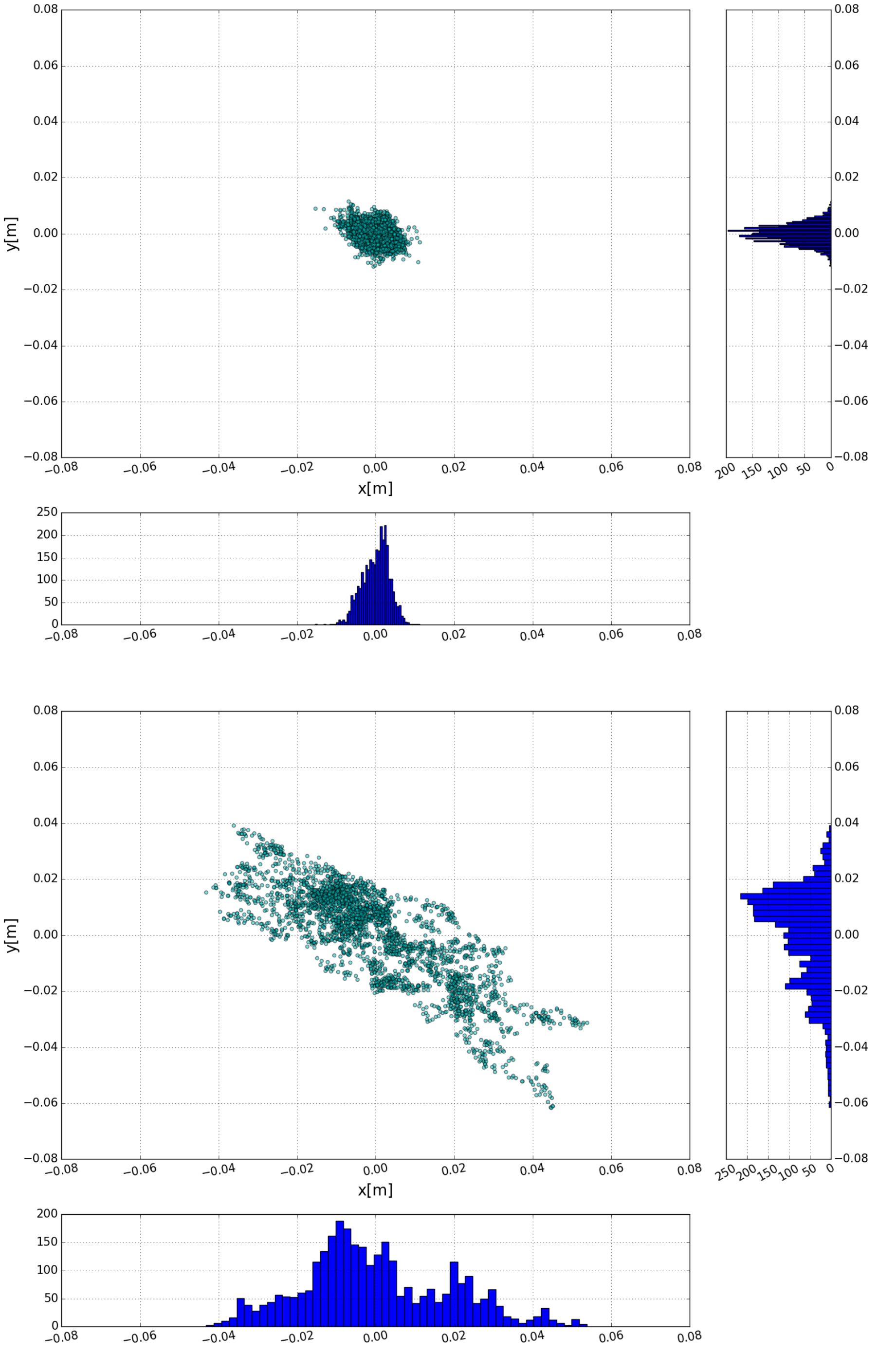


Figure 2.8: Distribution of positioning data of the RTK-GNSS (upper graph) and the QZSS (lower graph). The right-hand side panels and the lower panels of each graph show the histograms for each of the corresponding axes.

2.3.2 Measurement accuracy in motion in an open-sky environment

In-motion experiments were conducted by the RTK-GNSS and the QZSS equipped in mobile robots. I compare the values measured by the RTK-GNSS and the QZSS and the true values measured by a robotic total station (GPT-9005A, TOPCON, Inc.). The measurement accuracy and frequency of the robotic total station are approximately ±7 mm and 1.7 Hz, respectively. The latitude, longitude, and orientation of the robotic total station were measured using prism poles and the QZSS (Figure 2.10).

Figure 2.9 shows the experimental environment, which is a square space of 18 m × 18 m, and the orange, green, and blue circles in Figure 2.9 indicate the initial position of the mobile robot, the position of the robotic total station, and the position of the prism pole, respectively. In this experiment, the maximum linear velocity of the robot was set to 0.1 m/s for stable measurement using the total station. In order to track the position of the mobile robot in motion by the robotic total station, a prism is mounted between the GNSS antenna and the mobile robot, as shown in Figure 2.11. The movement speed of the mobile robot was set to 0.1 m/s in the experiment.

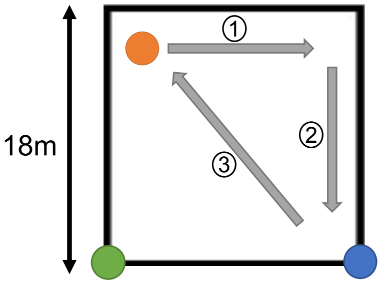


Figure 2.9: Experimental conditions.

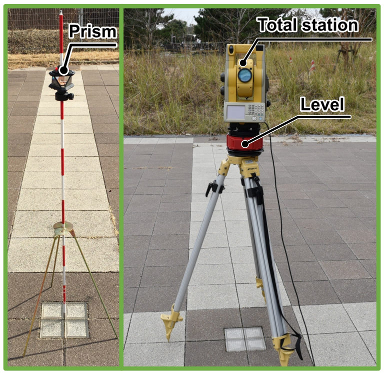


Figure 2.10: Prism pole (left window) and robotic total station (right window).

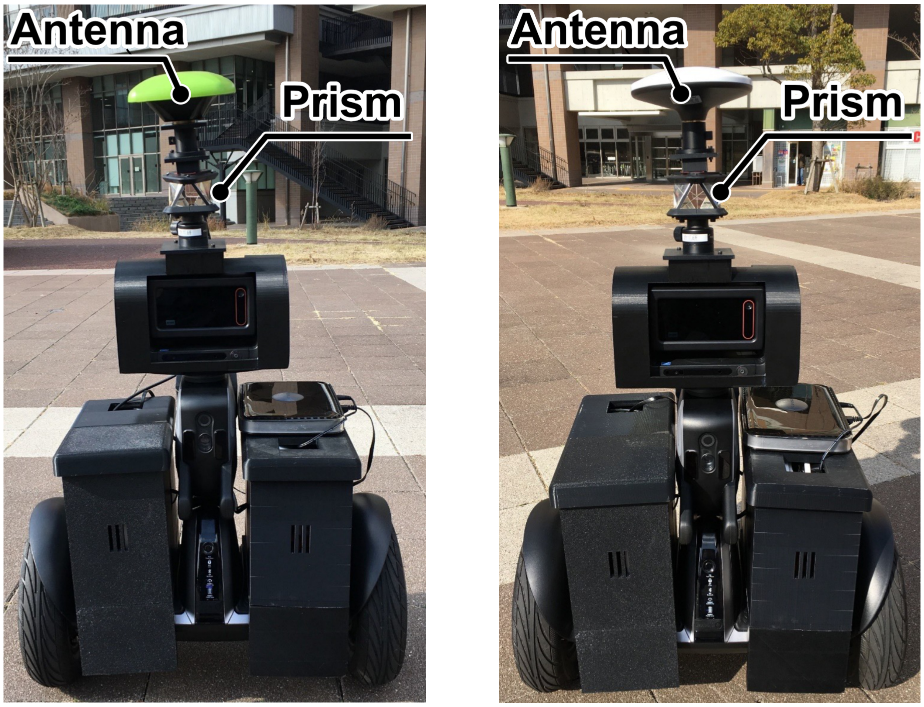


Figure 2.11: Setup of the GNSS antenna and prism for the comparing experiment. The left-hand side describes the configuration for the QZSS, and the right-hand side describes the configuration for the RTK-GNSS.

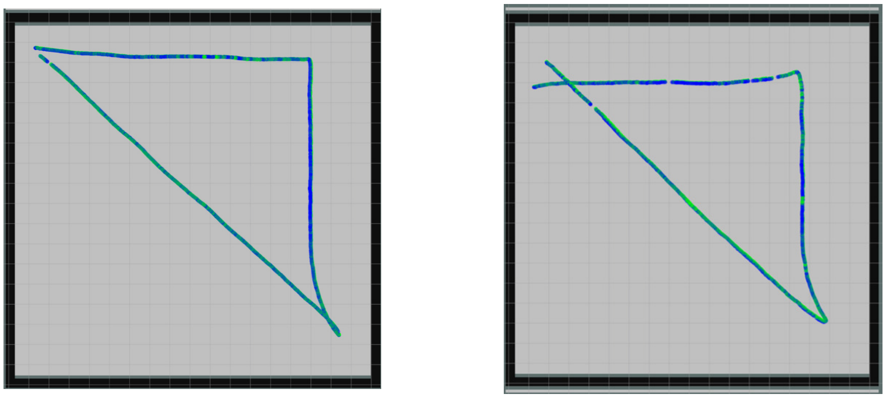


Figure 2.12: Measured trajectories. Green lines indicate the trajectories obtained using the robotic total station, and blue lines indicate the trajectories obtained using the RTK-GNSS (left figure) and the QZSS (right figure).

The trajectories measured by the RTK-GNSS, the QZSS, and the robotic total station are shown in Figures 2.12. In these figures, the green lines indicate the trajectories obtained using the robotic total station, and blue lines indicate the trajectory obtained using the RTK-GNSS or the QZSS.

The maximum value (MAX), root mean square (RMS), and standard division (SD) of the differences between the positions measured by the GNSS and the robotic total station are shown in Table 2.1.

Table 2.1: Differences between the positions measured by each GNSS module and the robotic total station (in meters).

	RTK-GNSS	QZSS
MAX	0.085	0.115
RMS	0.032	0.043
SD	0.014	0.018

Based on this results, the accuracy of the RTK-GNSS is slightly higher than that of the QZSS. A more detailed discussion is presented in Section 2.3.4.

2.3.3 Experiment in a partially obscured environment

In this experiment, I run the robot along a route that is close to higher-rise buildings and compare the positioning accuracy and stability of the RTK-GNSS and the QZSS. The results are shown in Figure 2.13. The fixed solution is a good result and the independent solution is a bad result. The fixed solution provides positioning results with high reliability and accuracy, and the independent solution yields positioning results with low reliability and accuracy.

Based on these results, the QZSS maintains a fixed solution along most of the route and performs stable measurements, even when the robot passes near high-rise buildings. On the other hand, the RTK-GNSS becomes unstable in some cases. One reason for this is that the QZSS uses the quasi-zenith satellites placed at the quasi-zenith orbit and observed with a high elevation angle from the GNSS antenna. I repeated the

Table 2.2: Fixed, float, and unstable rates for each GNSS module (in percentage).

	RTK-GNSS	QZSS
Fixed	34.0	92.2
Float	50.4	7.35
Unstable	15.6	0.50

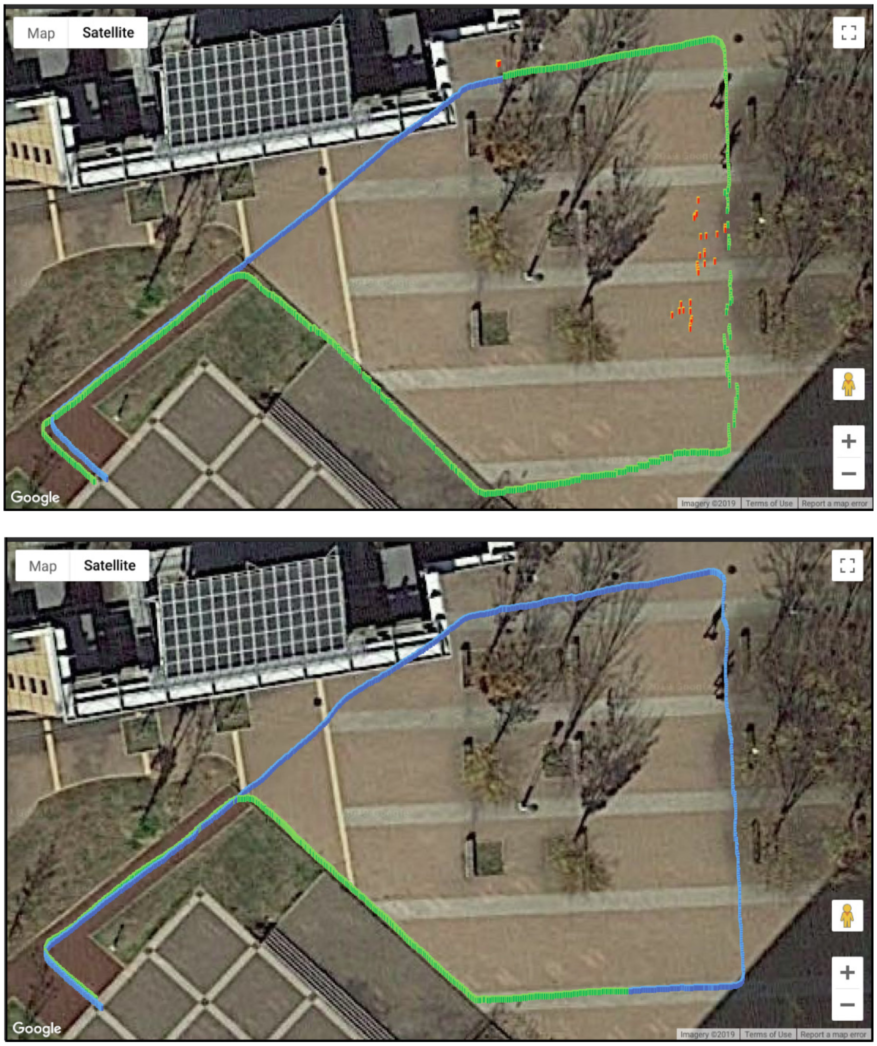


Figure 2.13: Measured trajectory by the RTK-GNSS (upper side) and the QZSS (lower side) are described by markers. The blue and green markers indicate the fixed and float solutions, respectively, and the orange markers indicate independent solutions.

experiment 10 times in the environment. Table 2.2 shows the fixed rate, float rate, and unstable rate for the RTK-GNSS and the QZSS by percentage of the total measurement data (about 10k data for each module). . These measurement types are discriminated by the module.

2.3.4 Performance comparison of the RTK-GNSS and the QZSS

The performances of the RTK-GNSS and the QZSS are shown in Table 2.3. As a result of the experiments, it can be seen that the RTK-GNSS is more accurate than the QZSS. The reason for this is thought to be the difference of the mechanism of position information correction. Correction information in the RTK-GNSS is created using the observation data at the base and the rover stations that are on line. On the other hand, as mentioned above, the QZSS uses the PPP-RTK (precise point positioning RTK) method, which is a model-based technique called a SSR. In the PPP-RTK method, the error is decomposed into the error in satellite clocks, a small variation in the orbit, tropospheric delay, ionospheric delay, etc., and these factors are estimated at the electronic reference points in the CLAS of the QZSS. However, the PPP-RTK method cannot take into account the real-time state change of errors and therefore cannot handle, for example, a sudden change in ionospheric conditions.

As demonstrated by the results of the experiment described in performance comparison section, the positioning accuracy does not differ greatly between the RTK-GNSS and the QZSS, and both techniques satisfy most requirements for applications of autonomous service robots. Although the RTK-GNSS is slightly more accurate than the QZSS, the RTK-GNSS requires two modules, and accurate positioning requires acquisition of the correct position of the base station. If the latitude and longitude of the base station are not accurately known, it takes a long time to obtain the accurate latitude and longitude by the GNSS. I have to place the base station for a certain period of time and collect data repeatedly. In addition, since communication between the base station and the rover station is required, the GNSS can only be used within the range in which such communication is possible.

On the other hand, since the QZSS can perform centimeter-class positioning with a single device, we do not need to consider an initialization procedure or the available range. Moreover, the QZSS can perform more stable positioning, even near buildings, because the QZSS uses satellites placed in a quasi-zenith orbit that are observed with

Table 2.3: Statistics of the RTK-GNSS and the QZSS.

	RTK-GNSS	QZSS
Accuracy (stopping)	Very good (fixed)	Good
Accuracy (moving)	Very good (fixed)	Good
Stability	Not good	Very good
Number of modules	2	1
Initialization	Measurement of base position	None
Measurement range	In communication range between modules	Not limited (around Japan)

a high elevation angle from the GNSS antenna. At any time, at least one quasi-zenith satellite can always be observed in and around Japan. Consequently, I conclude that the QZSS is more suitable for a centimeter-class positioning system for autonomous service robots. With respect to the overall accuracy, the RTK-GNSS has a higher performance, but the difference is approximately several centimeters, which is not a large difference when considering the position identification of the robot. On the other hand, the QZSS is superior with respect to the stability of measurement, the number of required modules and preparations, the limits of the measurement range, and convenience. Overall, I conclude that the QZSS is better for robot position identification.

2.4 5G network

In this study, I'm using a 5G environment provided by NTT DOCOMO, Inc., in the Huis Ten Bosch theme park. However, there are difficulties in using the network system for the proposed robot system. The components of the network system and how to solve these problems in the proposed system are introduced hereinafter. The configuration of the network system is described in Figure 2.14. Preliminary experiments revealed that wireless communication at approximately 20–40 Mbps was possible with this configuration. Note that the upper limit of the communication speed was set at 40 Mbps because of the limitation of CPU power of the onboard computer. Theoretically, the 5G network is capable of communication speeds of 20 Gbps. On the other hand, the average speed of the LTE (4G) network, which is the current standard for outdoor wireless communication, was 16 Mbps (Aterm MR05LN, NEC, Corp.) by actual measurement. Since the 360-degree camera in the system requires a maximum of about 56 Mbps to transmit 4K images, the 5G network system is suitable for the proposed application. The details of the application will be described later.

2.4.1 Components of the network system

The network system has a control PC, a system server, the docomo Open Innovation Cloud (dOIC), and 5G routers. The control PC is used to control the robot. The ROS-based controlling system and 360-degree 4K camera system are implemented in the PC. (These systems are described in detail in the section on the configuration of the developed robot system.) The system server is used for monitoring the information of the robot. In addition, we can control the robot remotely from the system server.

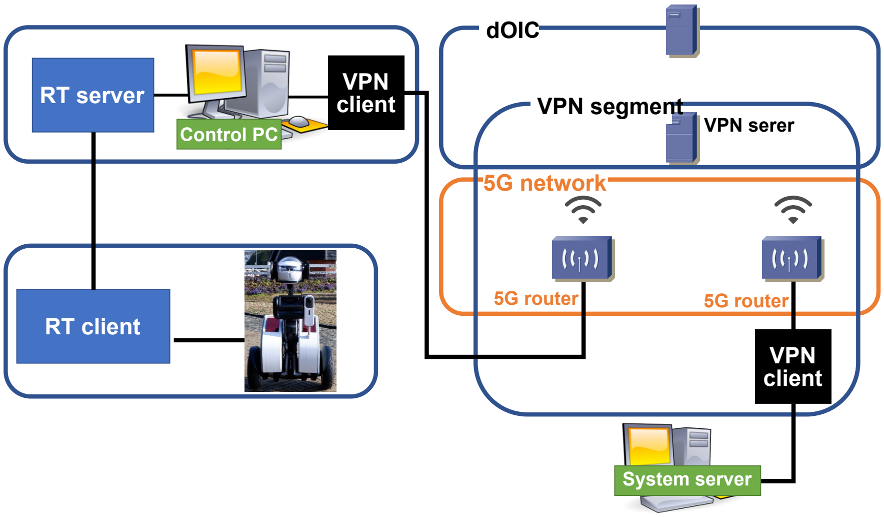


Figure 2.14: Configuration of the 5G network. The control PC and the server PC are connected to a 5G router. Each router is communicating through a VPN network. The reverse tethering system is adopted between the control and the control PC.

The dOIC is a cloud service provided by NTT DOCOMO, Inc. with multi-access edge computing (MEC) features, such as low latency and high security, which are required in the 5G era. This is achieved by building a cloud platform on the facilities within the DOCOMO network. We can use virtual machine instances and virtual networks. In addition, we can use some assets developed by NTT DOCOMO, Inc., such as an image processing API, in the future. The 5G router was also provided by NTT DOCOMO, Inc. The router has a LAN port and Wi-Fi system for connecting to the router. Moreover, 5G routers can communicate wirelessly with other routers by 5G radio waves through the base station for 5G.

2.4.2 Solutions for the problems in the network system

There are two problems related to using the network system for the robot system. The first problem is related to the 5G router, and the second problem is related to an android device mounted in the robot. The details of these problems are as follows:

1. The modules connected to 5G routers cannot communicate directly with other modules connected to other routers on the network, because of the functional limitation of the 5G router.

2. The interface for controlling the base robot in the proposed system is an android device mounted in the robot. Therefore, the android device also needs to connect

to the 5G network system. Therefore, the android device also needs to connect to the 5G network system. However, the current android device cannot connect to the 5G network directly and needs to be connected to the 5G network via a 5G router. Besides, the wired connection of the Android device is functionally disabled.

In order to solve the first problem, I adopt a virtual private network (VPN) communication system. The VPN communication system realizes the situation that private networks are virtually connected. Using this system, the modules connected to the 5G routers can communicate with other modules connected to other router networks. The VPN server is implemented in dOIC, and the VPN client is implemented in each control PC and server PC using SoftEther VPN [23].

On the other hand, I used the reverse tethering (RT) system for solving the second problem. The basic tethering system is used for the PC connected to the network to which mobile devices, such as an Android system, are connected. Reverse tethering simply realizes the reverse version of tethering, which means that mobile devices can connect to the network to which the PC is connected. By using RT, I connected the robot Android device to the 5G network to which the control PC was connected. The server and client systems for RT are implemented using SimpleRT [24]. The RT server was running on the control PC, and the RT client was running on the Android device.

2.5 Configuration of the developed robot system

As mentioned above, I confirmed that the QZSS can perform centimeter-class positioning with a simple and easy-to-use system consisting of a single module. In addition, I verified the 5G network system has a high communication speed. In this section, I present two types of service robot systems using the QZSS and the 5G network. One is a tour guide robot system that mainly uses the advantages of the QZSS, and the other is a co-experience system that uses the advantages of the 5G network. The QZSS is suitable for outdoor mobile robots in wide area because it does not require multiple modules and communication between base and rover stations, which are required for the RTK-GNSS. In addition, the co-experience system can share experiences using 360-degree video, and the high communication speed using the 5G network is suitable for the transmission of 4K high resolution images. Figures 2.15 and 2.16 show the developed robot systems.

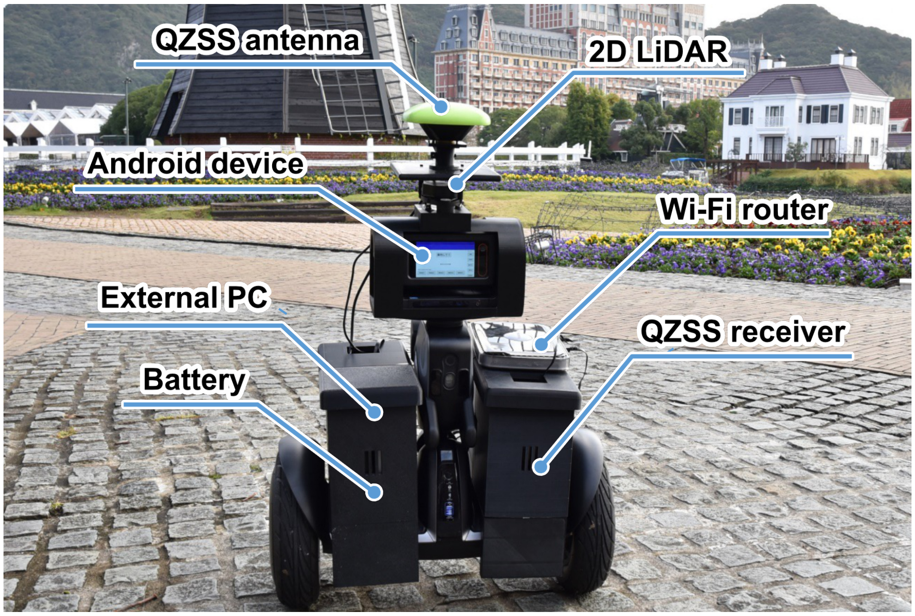


Figure 2.15: Qurin: the first version of the robot system. This robot uses a standard Wi-Fi network.

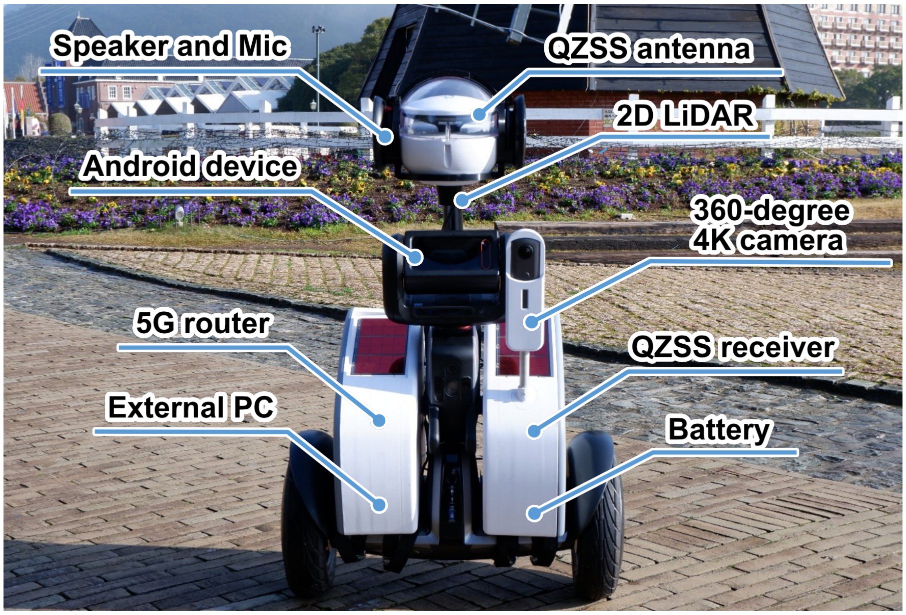


Figure 2.16: Quriana: the second version of the robot system. This robot uses a 5G network.

2.5.1 Tour guide robot system

The tour guide robot system aims to perform guiding at theme parks. It moves automatically and guides the guests to the requested goal with voice announcements.

2.5.1.1 Hardware configuration of the tour guide robot system

As a mobile platform, I used Loomo (Segway, Inc.), which is an inverted two-wheeled robot, controlled from an Android terminal. I equipped Loomo with 2D LiDAR (LDS-01, ROBOTIS, Inc. for Qurin and TiM581, SICK, Inc. for Quriana) and the QZSS external sensors. The 2D LiDARs are used to detect obstacles. In addition, a battery, an external PC (Intel NUC), a Wi-Fi router for communication between Loomo and a PC were mounted on Qurin. Instead of a Wi-Fi router, Quriana has a 5G router and was mounted with a 360-degree camera, a speaker, and a microphone.

2.5.1.2 Navigation system

As software, a navigation system and the tour guide application were installed. The navigation system is based on the ROS Navigation Stack. Each component of the navigation system, localization, collision avoidance, and path planning is explained below.

- Localization: Position information obtained by the QZSS and the velocity information measured by the wheel encoder are integrated by the extended Kalman filter (EKF) in the robot_localization package [25]. The QZSS alone is not enough to estimate the robot's orientation, so it is needed to combine it with information from the wheel encoders. The EKF estimates the pose (position and yaw angle) and the velocity (linear and angular) of the robot.
- Collision avoidance: Using the data measured by 2D-LiDAR, the robot stops when a pedestrian is detected within a certain range.
- Path planning: The shortest path (global path) to the destination is planned using the Dijkstra method, and an optimal route (local path) to avoid obstacles is generated by the dynamic window approach along the global path.

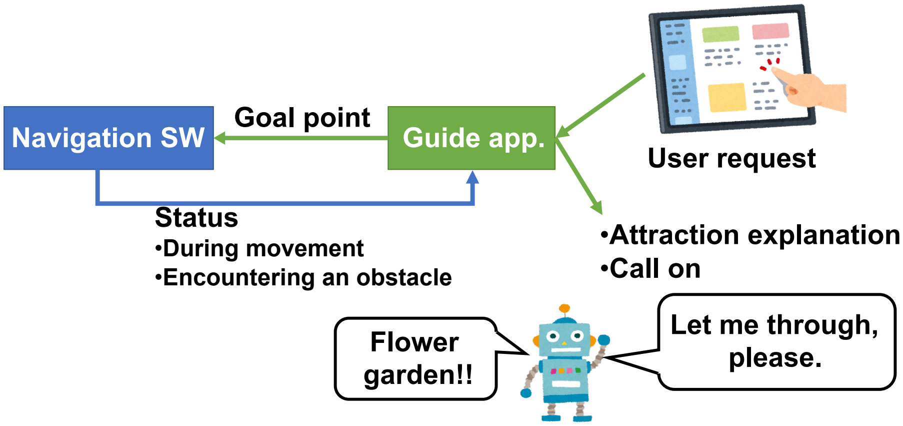


Figure 2.17: Tour guide application.

2.5.1.3 Tour guide application

As shown in Figure 2.17, the tour guide application is implemented on the Android terminal. This application sends the goal information to the navigation system in response to a request from the user and receives the current status of the robot. The status includes information such as whether the robot has reached the goal or an obstacle has been detected, and guide information for an attraction is provided by voice according to the location of the robot.

2.5.1.4 Voice recognition system

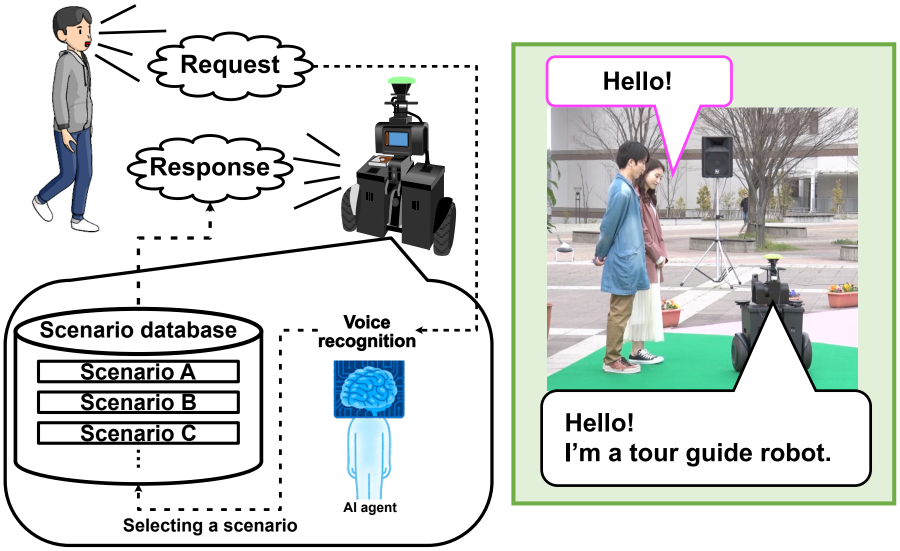


Figure 2.18: Conceptual diagram of the voice recognition system. The system responses with selecting scenario according to voice recognition results.

The speech recognition system is implemented using the DOCOMO AI Agent API [26]. As shown in Figure 2.18, it is possible to have a conversation according to a predetermined scenario. The tour guide robot system uses the speech recognition system to accept user requests and answer questions.

2.5.2 Co-experience system

The co-experience system is a system in which users share experiences with robots through a network. This system allows us to present a 360-degree 4K video to the user in an immersive manner, as if the situation around the robot were the situation around the user. In a theme park environment in particular, using this system makes it possible to visit a theme park from a remote location via the proposed robot system.

2.5.2.1 Hardware configuration of the co-experience system

In the co-experience system, a 360-degree camera is used to capture the field of view of the robot. The 360-degree camera used is the Theta V (Ricoh Co., Ltd.) shown in Figure 2.19, and it is possible to acquire 360-degree video with 4K quality. The view of the robot is presented to the user using a VR head-mounted display. By viewing the video captured by the 360-degree camera on the VR head-mounted display, the user can observe the surroundings of the robot as if the field of view was his/her own. The VR head-mounted display used in this system is the Oculus Rift (Oculus VR Inc.) shown in Figure 2.20.

Figure 2.19: Theta V.

Figure 2.20: Oculus Rift.

2.5.2.2 Software configuration of the co-experience system

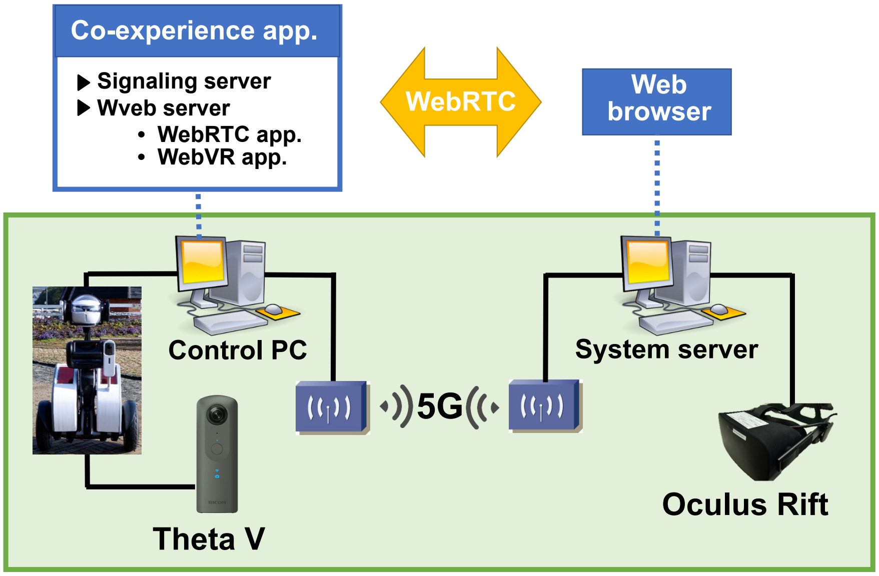


Figure 2.21: Software configuration of the co-experience system.

Figure 2.21 shows the software configuration of the co-experience system. This system uses WebRTC [27] to communicate in real-time between the robot and the user. The WebRTC system is implemented as an application that runs on a Web browser using the JavaScript API, and by accessing the Web server, the application can be accessed via the network. In addition, A-Frame [28] is adopted to realize a VR application using Oculus Rift. Note that VR applications can be implemented on a Web browser using A-Frame. This VR application is also incorporated into the application on the Web server.

2.6 Experiment for tour guide robot system

2.6.1 Collision-avoidance experiment

I conducted a collision-avoidance experiment to confirm whether the collision avoidance system and the voice announcement of the tour guide application work well. Figure 2.22 describes the results of the experiment. Figure 2.22(i) shows a scene in which pedestrians approach from the direction of movement of the robot. Figure 2.22(ii) shows a scene in which the robot stops in front of a pedestrian and is providing an

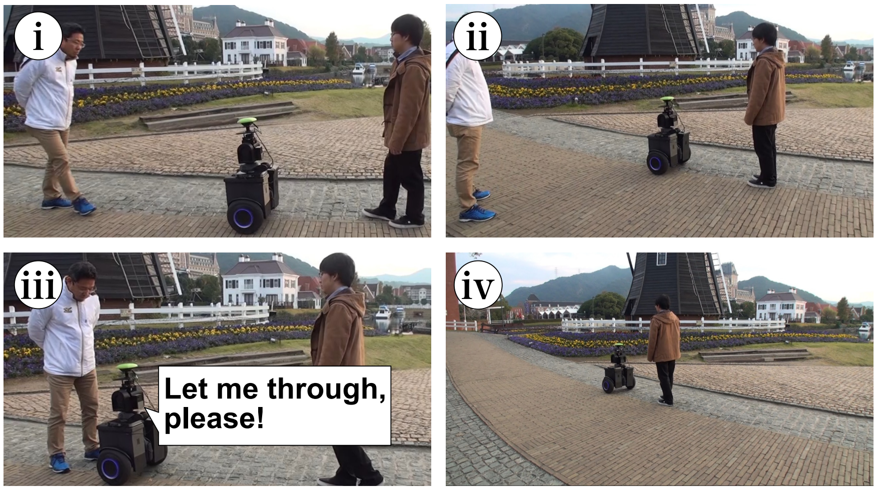


Figure 2.22: Collision-avoidance experiment.

announcement by voice. Figure 2.22(iii) shows a scene in which the pedestrian moves out of the way according to the voice. Figure 2.22(iv) shows a scene in which the robot restarts because the path has been cleared.

I confirmed that the collision avoidance system works well based on the results of the experiment.

2.6.2 Guided tour experiment

I conducted a guided tour experiment to confirm the performance of the developed system at the Huis Ten Bosch theme park in Japan. The environment and the procedure of the experiment are shown in Figure 2.23. The robot moves from point ① to point

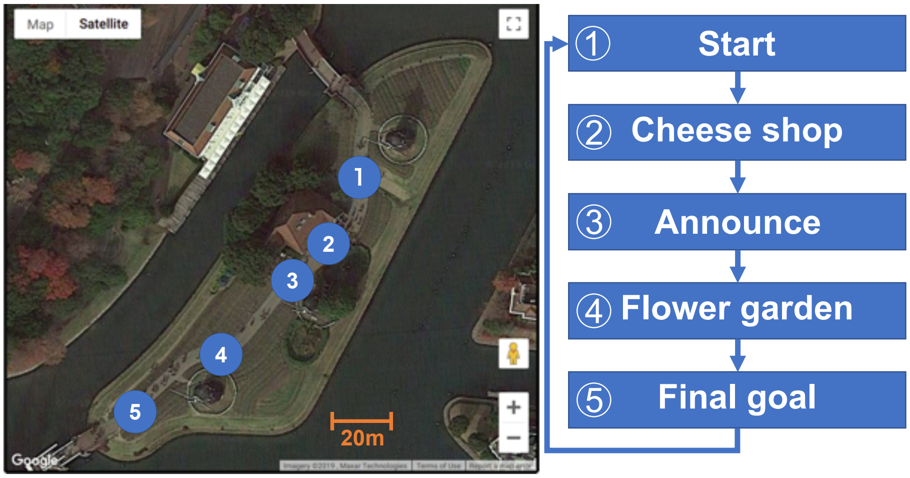


Figure 2.23: Environment for the guided tour experiment. The blue circles describe the target points of the guided tour, and the flowchart describes the flow of the experiment and each attraction.

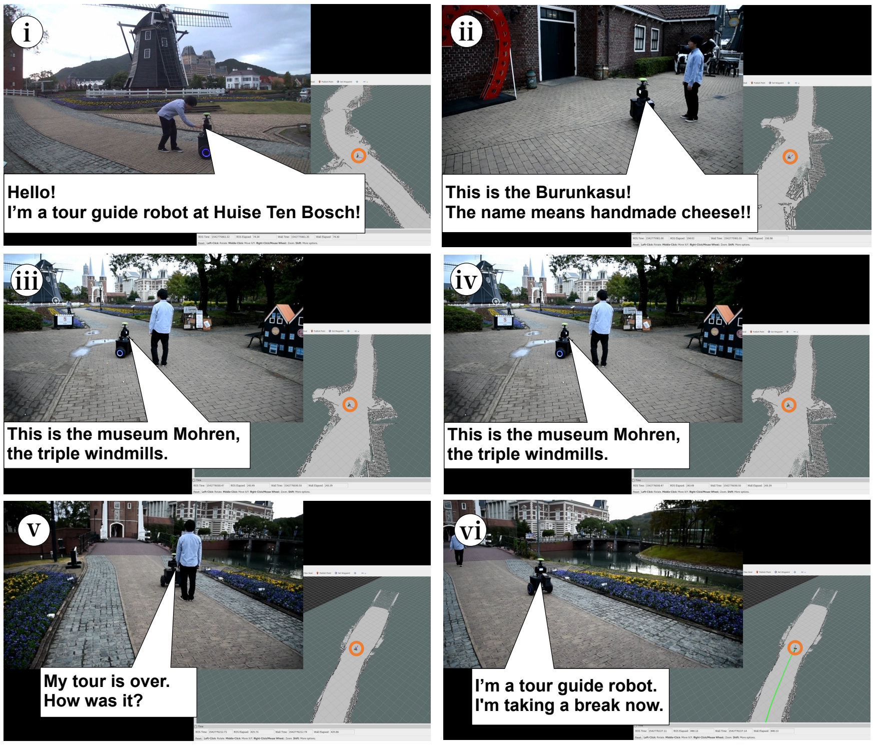


Figure 2.24: Guided tour experiment.

⑤ and explains the attraction at each point by voice. The total distance traveled by the robot is approximately 130 meters, and the robot returns to the initial point, i.e., point ①, after arriving at point ⑤ automatically, as shown in Figure 2.24.

Figure 2.24(i) shows the robot start the tour at the start position. In Figure 2.24(ii), the robot arrives at the cheese shop, which is the first target point, and gives a description of the shop. Figure 2.24(iii) shows the robot arriving at the windmill, which is the second target position. The robot provides a description of the windmill and the history of the Netherlands. In Figure 2.24(iv), the robot arrives at the third target point, the flower garden, and provides a description of the types of flowers. In Figure 2.24(v), the robot arrives at the end point and announces the end of the tour. Figure 2.24(vi) shows the robot returning to the start position after the tour is over. Guided tour experiments were conducted seven times in total, and six of the tours were successfully performed as planned. The reason for the failure is that the measurement of the QZSS became unstable in the area where buildings and trees were closely placed around the robot. However, this does not occur often, and, thus, if we plan the tour route carefully, then the developed system is quite practical as a tour guide system for an outdoor theme park.

2.6.3 Experiment using the voice recognition system

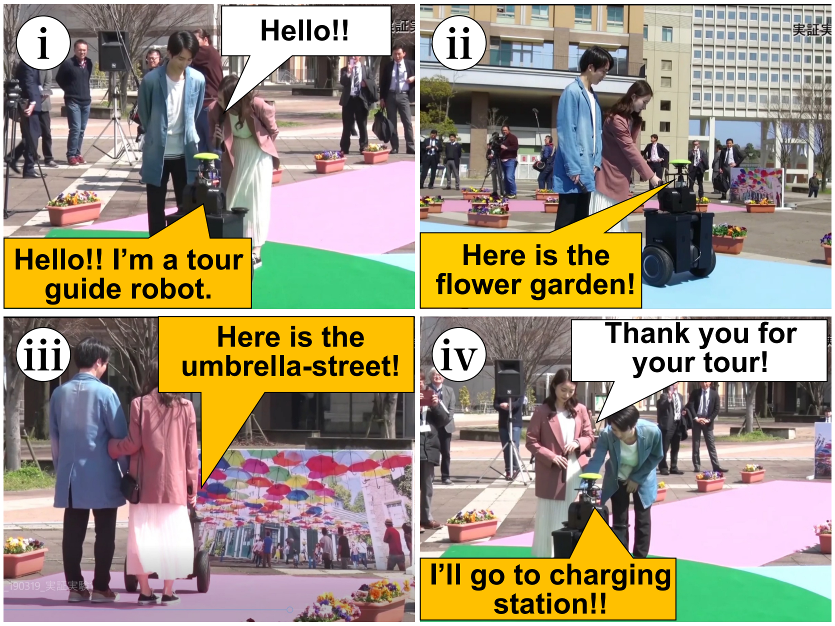

i
Hello!!
Hello!! I'm a tour guide robot.
ii
Here is the flower garden!
iii
Here is the umbrella-street!
iv
Thank you for your tour!
I'll go to charging station!!

Figure 2.25: Experiment with the voice recognition system.

I conducted the tour guide demonstration with the voice recognition system. The system used in this experiment consists of the proposed tour guide robot system, and the AI-based voice interaction system. The voice commands are transferred to the cloud-based AI system in real time. As shown in Figure 2.25, the robot properly guided the guests to several sights by voice command.

Figure 2.25(i) shows a greeting scene. Figure 2.25(ii) shows a scene in which the robot explains the first attraction. Figure 2.25(iii) shows a scene in which the robot explains the second attraction. Figure 2.25(iv) shows a scene in which the robot answers a question from a user.

Based on the experiment, I confirmed that the system can conduct tour guide tasks corresponding voice requests.

2.7 Experiment using the 5G network

I conducted two types of experiments with the 5G network system: a tour guide experiment and a co-experience experiment. The environment of these experiments is

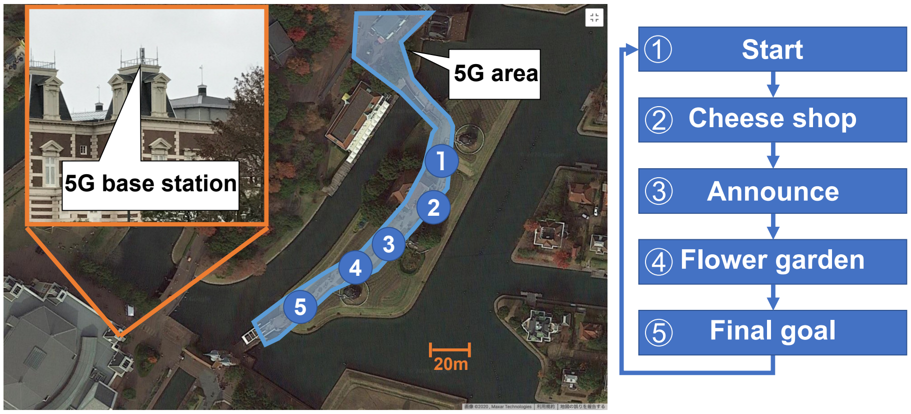


Figure 2.26: Environment for experiments with the 5G network system. The blue area describes the 5G area, and the window at top left shows one of the base stations of 5G.

shown in Figure 2.26. In the environment, the 5G usable area is configured by two base stations.

2.7.1 Guided tour experiment

I conducted a guided tour experiment to confirm the performance of the developed system. This experiment is also conducted at the Huis Ten Bosch theme park. The

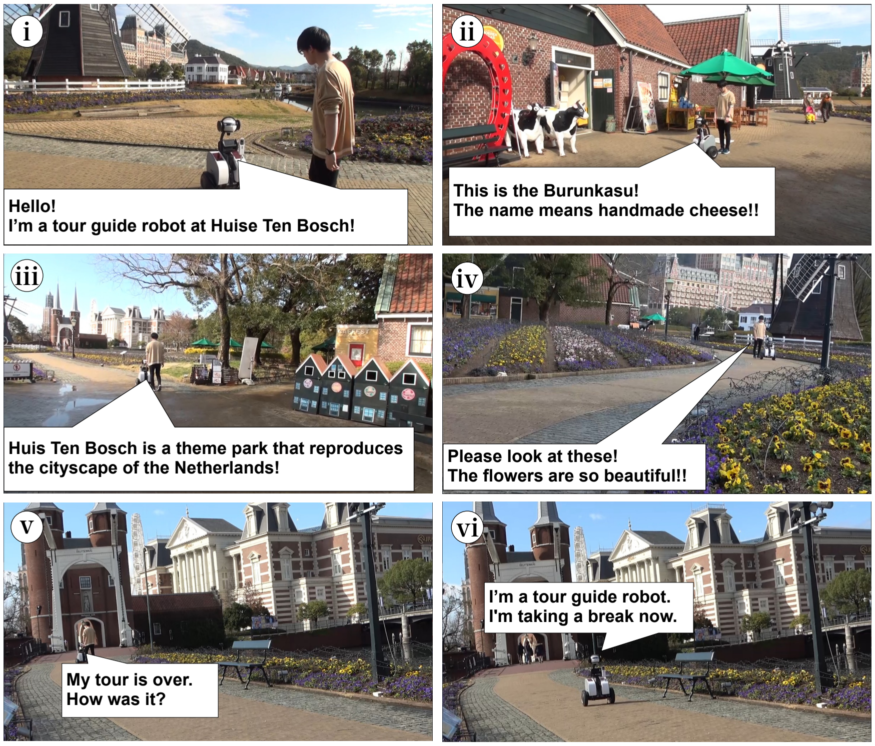

i
Hello!
I'm a tour guide robot at Huise Ten Bosch!
ii
This is the Burunkasu!
The name means handmade cheese!!
iii
Huis Ten Bosch is a theme park that reproduces
the cityscape of the Netherlands!
iv
Please look at these!
The flowers are so beautiful!!
v
My tour is over.
How was it?
vi
I'm a tour guide robot.
I'm taking a break now.

Figure 2.27: Guided tour experiment using the 5G network.

environment and the procedure of the experiment are shown in Figure 2.27.

Figure 2.27(i) shows the robot start the tour at the start position. In Figure 2.27(ii), the robot arrives at the cheese shop, which is the first target point, and gives a description of the shop. Figure 2.27(iii) shows the robot is introducing the theme park and navigating to the next point. In Figure 2.27(iv), the robot arrives at the second target point, the flower garden, and provides a description of the types of flowers. In Figure 2.27(v), the robot arrives at the end point and announces the end of the tour. Figure 2.27(vi) shows the robot returning to the start position after the tour is over.

2.7.2 Co-experience experiment

Using the developed system, I confirmed the operation of the co-experience system in the 5G network area at Huis Ten Bosch. The maximum communication distance in this experiment was about 190 meters. The user wears a VR head-mounted display, as shown in Figure 2.28, and could experience the visual field of the robot and could see the image shown in Figure 2.29. I confirmed that the robot experience could be shared from a remote location using the system developed by this experiment.

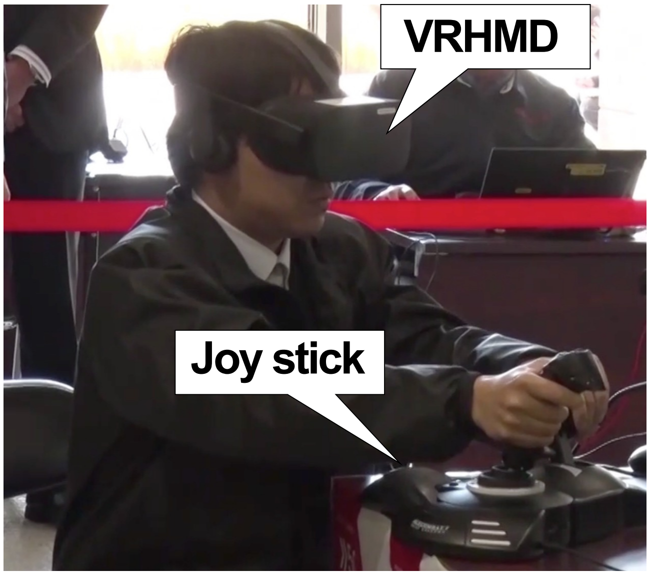


Figure 2.28: User of the co-experience system. The user is viewing 360-degree 4K video sent via the 5G network using VR head-mounted display (VRHMD).

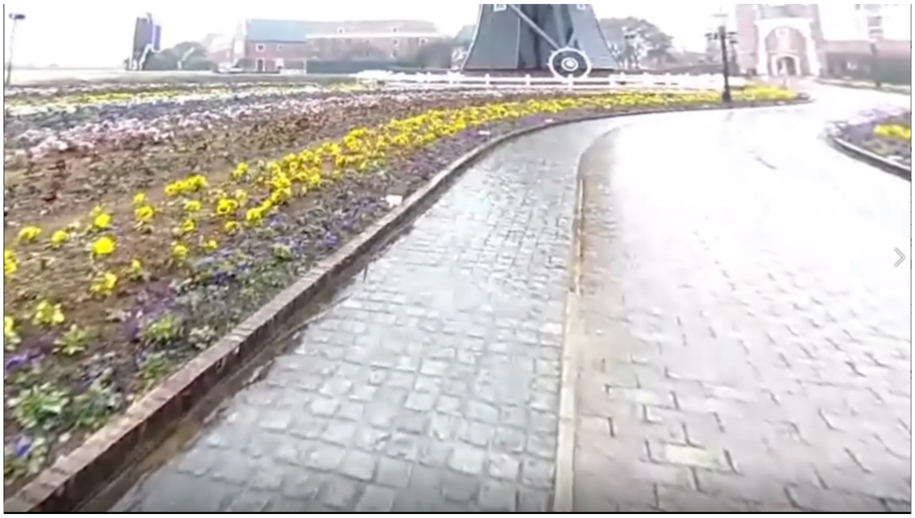


Figure 2.29: View presented by the co-experience system.

2.8 Conclusion

In the present study, I developed a tour guide robot system and a co-experience system using new technologies: the QZSS and 5G. The elemental technologies and devices used to realize the systems in this study have been developed individually. I constructed a service robot that can operate in the real world by integrating them.

In order to confirm the effectiveness of using the QZSS for the tour guide robot, the performance of the QZSS was examined, and its accuracy and stability were verified by a centimeter-class positioning system for autonomous service robots. I compared the accuracy of the QZSS and the RTK-GNSS, and I think that the positioning accuracy of both systems is sufficient for the tour guide application if a stable GNSS signal is obtained. Experiments conducted at the theme park show that the tour guide robot successfully traveled 130 meters repeatedly and acted as a guide to the attractions using the QZSS. The centimeter-class positioning service was started November 2018 and, to the best of my knowledge, this study is the first to use the CLAS of the QZSS for autonomous service robots. The QZSS can perform centimeter class positioning by only one module, so the developed robot system is more useful than common RTK-GNSS-based mobile robot systems.

I also developed a co-experience robot system that allows us share the experience of the robot. The proposed system uses a 5G network system for transporting a 4K video stream of the experience of the robot. As mentioned in section “5G network”, the communication speed of outdoor wireless network by LTE is not suitable for the proposed application, and I believe that 40 Mbps or more is necessary for co-experience application using 360-degree 4K video.

The implementation and results of this study for the requirement specification are summarized below. As described above, I achieved the required specifications presented at the beginning of this chapter.

TG1. *Guiding the way while moving autonomously*
I developed an autonomous mobile robot system based on Navigation Stack, one of the ROS packages. In particular, by using the QZSS and combining it with a wheel encoder to estimate position and orientation, I was able to achieve autonomous mobile robot in an open environment. Since this system uses the QZSS, there are no limitations on high-precision positioning such as the communication range between base stations and mobile stations in conventional GNSS. This is a huge advantage for tour guides in vast and potentially crowded environments such as the theme park. I conducted experiments using the developed system in the theme park environment, which is the target of this study, and confirmed that the system can move autonomously.

TG2. *Explanation and announcement*
I implemented a tour guide application that can manage the robot's current location, destination and state, and provide appropriate voice explanations and announcements. It makes voice explanations when it arrives at a facility or attraction in the theme park and voice announcements according to state of the robot. This application uses the status information of the robot obtained from the navigation system. This status includes information on whether the robot has arrived at the goal or detected obstacles. Through experiments in collision avoidance and guided tours, I confirmed that the application is capable of providing appropriate explanations and announcements based on the robot's position and state.

TG3. *Interface to respond to user requests*
A voice recognition system has been implemented to respond to user requests. This system is developed using a DOCOMO AI Agent API, which can respond to user requests according to predetermined scenarios. The system can start tour guide by requesting the destination location of the guidance through a voice request. This realized a system that allows user to send requests to the robot in

a casual manner while enjoying sightseeing. I confirmed its operation by our guide demonstration with voice requests.

CE1. *Transmission of high resolutional video*
By using 5G and dOIC, a high-capacity and low-latency communication environment has been established, and a system that can distribute 360-degree 4K camera video to remote locations has been implemented. We confirmed the operation of the system by conducting an experiment in which 360-degree 4K video was transmitted over a distance of up to 190 meters, which is impossible with conventional wireless communication systems.

CE2. *Presenting the transferred video to the user in an immersive way*
I developed an immersive system that allows users to perceive the robot's point of view as if it were their own, by presenting 360-degree 4K video acquired via the 5G network to the user using an HMD. The system is implemented as an application that runs on a web browser using A-Frame.

In the future, I intend to improve the stability of the developed tour guide robot system by combining sensors including not only on-board sensors, such as 2D LiDAR and cameras, but also ambient sensors embedded based on the concept of the informationally structured environment [17]. In addition, pedestrian detection and tracking are also important functions for a safe and efficient autonomous robot system, and I intend to implement these functions and develop a practical tour guide robot system. Alternatively, with respect to the co-experience system, usability is an important factor. I'm planning to develop a hybrid system of automatic and manual control that will realize comfortable remote control by supporting manual commands with an autonomous system.

Moreover, I also need to work on improving our path planning methods. Since an environment such as the theme park is crowded with many people, path planning methods are very important for robots to move safely in such a dynamic environment. I'll discuss this topic in the next chapter.

3

Learning-Based Navigation with Predictive State Representation

3.1 Introduction

Mobile robot navigation in dynamic environments is an important element for service robots that provide services in spaces inhabited by humans. Handling this task requires the robots to understand the behavior of pedestrians. However, pedestrian behavior may change depending on their intentions and other factors that depend on particular environments. Thus, it cannot be easily modeled in advance with sufficient accuracy, particularly if the road shape and attractions in the environment (*e.g.,* stores or scenery) are considered. Furthermore, the behavior of the robot may affect the pedestrian behavior.

To address these issues, I propose a learning-based method based on predictive state representation (PSR) that considers the changes in pedestrian behavior caused by robot actions. In addition, to make PSR more adaptable to the dynamic environment caused by pedestrians, I propose a new PSR model that can consider the effect of pedestrians on each other. By using it, the proposed method can learn the behavior of pedestrians by observing their interactions with the environment and the changes in their behavior due to the robot's actions. Mobile robot navigation methods based on deep reinforcement learning (RL) have been actively studied in recent years [29, 30, 31, 32, 33]. However, the methods proposed so far mainly use predesigned models or do not specifically consider the changes in pedestrian behavior caused by robot actions. The difference between my method and existing methods is that my method consider both the effects of pedestrians on each other and the effects of the robot's behavior on the pedestrians' behavior in single model.

In this study, I apply a deep RL method based on the new PSR model to mobile robot navigation for considering the changes in pedestrian behavior caused by robot actions and other pedestrians. In addition, I consider the situation where the number of pedestrians changes between training and testing. The main contributions of this study are as follows:

- The application of a deep RL method using PSR for mobile robot navigation and the verification of the effects of the structure based on PSR for mobile robot navigation tasks;
- Proposing a new PSR architecture for mobile robot navigation in a dynamic environment with pedestrians considering the interactions among pedestrians and the verification of the effectiveness of the architecture;
- Proposing methods for integrating the states of the PSRs corresponding to each pedestrian for handling the change in the number of pedestrians between training and testing, and comparing these integrating methods.

3.2 Related works

Various methods based on deep RL techniques have been proposed. Everett *et al.* [29] proposed a method based on policy-based RL and long short-term memory. Chen *et al.* [30] proposed a pooling module for learning the relationships between the attention mechanism, changes in human behavior, and surrounding pedestrians; they also proposed a method to learn robot behavior, including human–robot and human–human

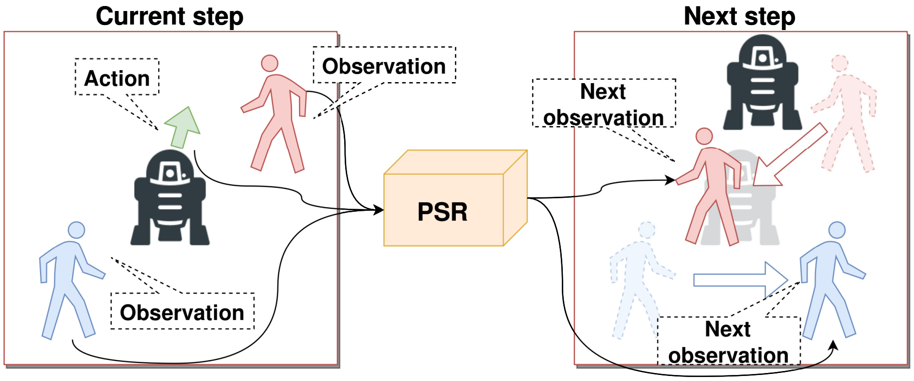


Figure 3.1: PSR modeling for mobile robot navigation in a dynamic environment with pedestrians.

interactions. Chen *et al.* [31] proposed an excellent navigation method that uses a graph convolutional network and attention mechanism based on human gaze. Chen *et al.* [32] used model-based deep RL methods and a graph convolutional network to achieve efficient navigation by encoding the interactions among agents. Liu [33] proposed a deep RL method that focuses on robot navigation in crowded real-world environments.

PSR [34] is used for modeling dynamics by considering partial observability. Various methods for utilizing PSR have been proposed. Boots *et al.* [35] proposed an extended PSR model based on the Hilbert space embeddings of distributions to manage infinite sets of continuous observations and actions. Hefny *et al.* [36] proposed a supervised learning method based on the kernel version of instrumental variable regression for PSR models. They also proposed an efficient PSR model for learning controlled dynamical systems using random Fourier features with a kernel function [37]. In addition, methods utilizing PSR for deep learning have been proposed recently. Venkatraman *et al.* [38] studied the use of PSR for representing the latent states of recurrent neural networks (RNNs). They observed that the proposed method could improve the performance and compensate for the disadvantages of RNN in probabilistic filtering, imitation learning, and RL. Hefny *et al.* [39] proposed a fundamental deep RL method, called the recurrent predictive state policy network, to utilize PSR.

In robotics, PSR has been applied in human–robot communication by utilizing its ability to model the results after performing actions [40]. It has also been applied in in-hand manipulation, which includes interactions with the environment [41]. This method extends PSR to address partial observability using a new kernel-based feature that integrates actions and observations. However, PSR has not yet been applied to mobile robot navigation in dynamic environments. In order to apply PSR to a dynamic environment with pedestrians, it is necessary to consider the effects of pedestrians on each other, but this can't be taken into account because PSR models basically focus only on the correspondence between the agent's actions and the changes in the environment. In this study, I propose a new PSR model to solve this problem.

3.3 Preliminaries

3.3.1 Predictive state representation

In PSR [34], the dynamics are modeled considering the interactions between an agent and the environment. The characteristic of PSR is that the state is expressed in a way that facilitates the prediction of the future. The state is represented by observations and actions. Therefore, values other than the fully observable elements do not need to be defined. In this section, I describe PSR in detail.

3.3.1.1 Original PSR

The basic concept of PSR is that, if the expected results of all possible tests are known, the dynamical system becomes completely understood. By expressing the state using observable information, partially observable dynamic systems can be modeled without prior knowledge. Consider a discrete system with a finite set of observations, $\mathcal{O} = \{o_1, o_2, \ldots, o_k\}$, and actions, $\mathcal{A} = \{a_1, a_2, \ldots, a_k\}$. The state representation of the system at time t is a vector composed of the probability of occurrence of a test based on the latest history. Each test is a sequence of actions and observations starting at time $t+1$; the history at time t is a sequence of actions and observations up to and including time t. The probability of success of a test, τ, of length m for history h, i.e., the probability of obtaining an observation sequence in τ when taking the sequence of actions in τ, is represented by $p(\tau \mid h) = p(h, \tau)/p(h) = \prod_{i=1}^{m} \Pr\left(o_i \mid h, a_i\right)$.

Knowing the success probabilities of some tests may help us understand those of other tests. Given a test set $\mathcal{T} = \{\tau_1, \tau_2, \ldots, \tau_k\}$, if there exists a function f_t whose prediction vector $p(\mathcal{T} \mid h) = \left[p\left(\tau_1 \mid h\right) p\left(\tau_2 \mid h\right) \ldots p\left(\tau_k \mid h\right)\right]$ satisfies $p(\tau_l \mid h) = f_t(p(\mathcal{T} \mid h))$ for any test τ_l, then $\mathcal{T}$ is called the *core test* and the prediction vector, $p(\mathcal{T} \mid h)$, represents the state of PSR.

3.3.1.2 Recurrent PSR

The original PSR model can only be applied to systems consisting of discrete observations and actions. Methods to make PSR compatible with continuous systems have been proposed [35, 37]. Herein, these methods are collectively referred to as recurrent PSR (RPSR). In RPSR, the state q_t is represented as a conditional distribution of future observations, $o_{t:t+k-1}$, conditioned by future actions, $a_{t:t+k-1}$. The state update of RPSR is performed in two steps.

- Extension: The linear map W_{ext} is applied to the state q_t to obtain the extended state, e_t. The extended state, e_t, is a conditional distribution of the extended observations, $o_{t:t+k}$, conditioned by the extended action, $a_{t:t+k}$:

$$e_t = W_{\text{ext}} q_t. \tag{3.1}$$

- Conditioning: For the action a_t and the observation o_t at time t, the known conditioning function, f_{cnd}, updates the state as follows:

$$q_{t+1} = f_{\text{cnd}}\left(e_t, a_t, o_t\right). \tag{3.2}$$

In discrete systems, q_t and e_t are represented by conditional establishment tables, and a Bayesian rule is applied by f_{cond}. To apply these to a continuous system, the Hilbert space embeddings of the distributions [35] and the kernel Bayes' rule [42] are used.

3.4 Approach

3.4.1 Application of PSR to mobile robot navigation in a dynamic environment with pedestrians

In this study, I applied a PSR-based deep RL method to mobile robot navigation in a dynamic environment with multiple pedestrians. The elements of mobile robot navigation and PSR can be associated as follows:

- Observations: The position and velocity data of N pedestrians and the robot are treated as observations. The observation data for each pedestrian and robot include vector $(p_x^i, p_y^i, v_x^i, v_y^i)$, where (p_x^i, p_y^i) represents the position and (v_x^i, v_y^i) represents the velocity of the ith pedestrian or robot.

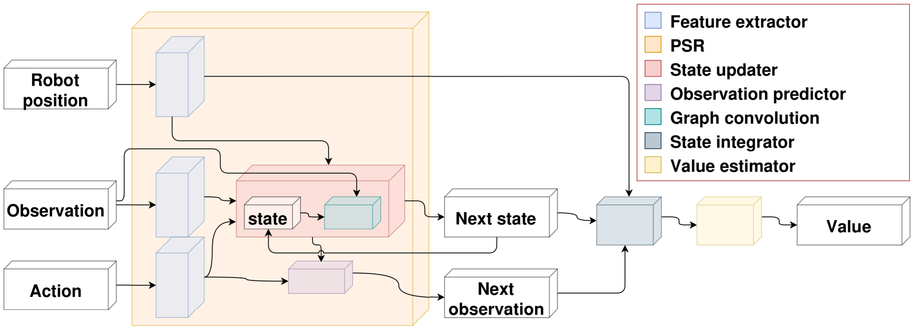


Figure 3.2: Architecture of the proposed method.

- Actions: In this study, I assumed a holonomic omnidirectional mobile robot and used a two-dimensional vector (v_x, v_y) consisting of the input velocity v_x in the x-axis direction and the input velocity v_y in the y-axis direction of the robot in a two-dimensional space.

The PSR used in this study becomes a model that can predict the position of the pedestrian and the velocity in the next step by inputting the command velocity of the robot. The conceptual diagram of the PSR model is shown in Figure 3.1.

3.4.2 Architecture of proposed method

The architecture of the proposed method is shown in Figure 3.2. The proposed method consists of a PSR consisting of a feature extractor, state updater, and observation predictor, as well as state integrator and value estimator.

- Feature extractor: Extracts features from each input datum using a nonstationary spectral kernel (NSK) function.

- State updater: Updates states via the state updater using the features extracted from the observation and action.

- Observation predictor: Predicts the next observations using the states features extracted from action.

- State integrator: Integrates the states of the PSR corresponding to each pedestrian and provides the integrated state to the value estimator.

- Value estimator: Estimates the value from the integrated state and robot feature.

3.4.3 Feature extraction using nonstationary spectral kernel

In the original RPSR model, kernel functions are used for feature extraction. In particular, RFF-PSR uses a stationary spectral kernel. The function is defined as

$$\phi(\boldsymbol{x}) = \sqrt{\frac{2}{D}} \cos\left(\boldsymbol{\Omega}^T \boldsymbol{x} + \boldsymbol{b}\right), \tag{3.3}$$

where $\boldsymbol{\Omega} = \{\omega_1, \omega_2, \cdots, \omega_D\}$ is the frequency matrix and $\boldsymbol{b} = \{b_1, b_2, \cdots, b_D\}$ is the phase vector sampled uniformly from $[0, 2\pi]^D$.

It should be noted that the stationary kernel is no longer applicable to complex tasks because of its locality. To solve this problem, NSK was proposed [43, 44, 45, 46]. NSK have been reported to be able to learn both local and global correlations between input spaces, and to be more capable of covering non-trivial input domains.

Stationary kernel, which is the source of the stationary spectral kernel function, is described as

$$\kappa\left(\boldsymbol{x}, \boldsymbol{x}'\right) = \int_{\mathbb{R}^D} e^{i\boldsymbol{\omega}^T(\boldsymbol{x}-\boldsymbol{x}')} s(\boldsymbol{\omega}) d\boldsymbol{\omega}, \tag{3.4}$$

where $s(\boldsymbol{\omega})$ is a non-negative probability density. On the other hand, non-stationary kernel $\kappa\left(\boldsymbol{x}, \boldsymbol{x}'\right)$ is described as Eq. 3.5 [43, 44].

$$\kappa\left(\boldsymbol{x}, \boldsymbol{x}'\right) = \int_{\mathbb{R}^D \times \mathbb{R}^D} \mathcal{E}_{\boldsymbol{\omega}, \boldsymbol{\omega}'}\left(\boldsymbol{x}, \boldsymbol{x}'\right) \mu\left(d\boldsymbol{\omega}, d\boldsymbol{\omega}'\right), \tag{3.5}$$

where

$$\begin{aligned} \mathcal{E}_{\boldsymbol{\omega}, \boldsymbol{\omega}'}\left(\boldsymbol{x}, \boldsymbol{x}'\right) = & \frac{1}{4}\left[e^{i\left(\boldsymbol{\omega}^T \boldsymbol{x} - \boldsymbol{\omega}'^T \boldsymbol{x}'\right)} + e^{i\left(\boldsymbol{\omega}'^T \boldsymbol{x} - \boldsymbol{\omega}^T \boldsymbol{x}'\right)}\right. \\ & \left. + e^{i\boldsymbol{\omega}^T(\boldsymbol{x}-\boldsymbol{x}')} + e^{i\boldsymbol{\omega}'^T(\boldsymbol{x}-\boldsymbol{x}')}\right], \end{aligned} \tag{3.6}$$

and $\mu\left(d\boldsymbol{\omega}, d\boldsymbol{\omega}'\right)$ is a Lebesgue-Stieltjes measure associated to some positive semi-definite spectral density function $s(\boldsymbol{\omega})$ with bounded variations. Using the Monte Carlo method for it, we obtain the NSK function as shown in Eq. 3.7.

$$\phi(\boldsymbol{x}) = \frac{1}{\sqrt{2D}}\left[\cos\left(\boldsymbol{\Omega}^T \boldsymbol{x} + \boldsymbol{b}\right) + \cos\left(\boldsymbol{\Omega}'^T \boldsymbol{x} + \boldsymbol{b}'\right)\right]. \tag{3.7}$$

I adopted NSK for feature extraction in each element of the PSR model.

3.4.4 Graph convolutional RPSR

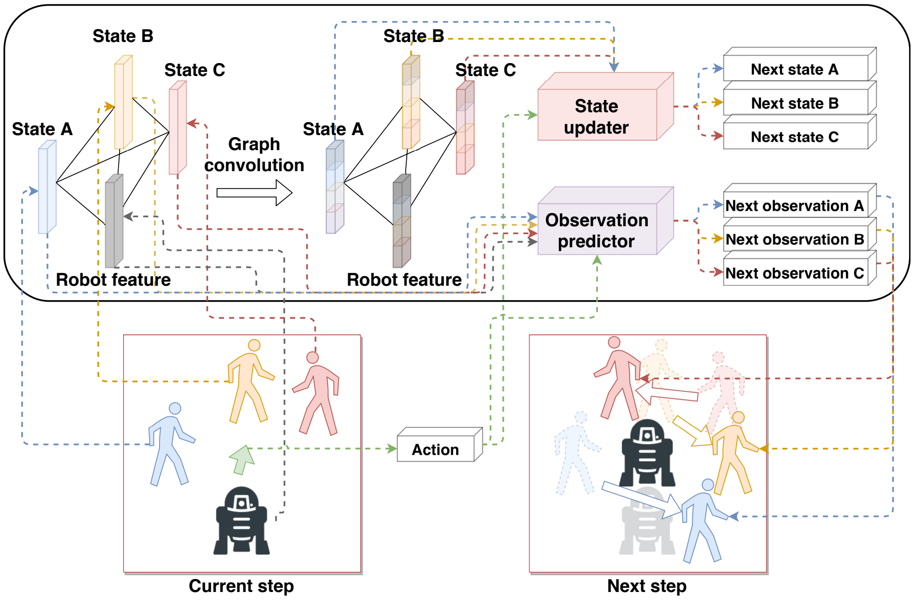


Figure 3.3: Procedure of GC-RPSR.

The basic RPSR model can consider the effect of the action of the agent on the environment. Through this feature, the effects of robot actions on each pedestrian can be considered. However, the RPSR cannot consider the effects of the interactions among pedestrians. For handling this problem, I propose graph convolutional RPSR (GC-RPSR).

This architecture has a graph convolution part in the state update phase in the RPSR for describing the relation effects among pedestrians. The state update process is shown in Figure 3.3. The kernel function for the adjacency matrix A_t is based on the inverse of the L_2 norm of the difference of each position [47]. This function is expressed in Eq. (3.8).

$$a_t^{ij} = \begin{cases} 1/\left\|p_t^i - p_t^j\right\|_2 & \text{if } \left\|p_t^i - p_t^j\right\|_2 \neq 0 \\ 0 & \text{Otherwise} \end{cases} \tag{3.8}$$

Finally, I use the normalized adjacency matrix. The normalization process is expressed in Eq. (3.9).

$$\tilde{A}_t = \Lambda_t^{-\frac{1}{2}} \hat{A}_t \Lambda_t^{-\frac{1}{2}}, \tag{3.9}$$

where $\hat{A}_t = A_t + I$ and Λ_t is the diagonal node degree matrix of A_t.

By using the graph convolution and Eqs. (3.1) and (3.2), the state update process of GC-RPSR is expressed as follows:

$$\tilde{q}_t = f_{gc}\left(q_t, \tilde{A}_t\right) \tag{3.10}$$

$$e_t = W_{\text{ext}}\tilde{q}_t \tag{3.11}$$

$$q_{t+1} = f_{\text{cnd}}\left(e_t, a_t, o_t\right), \tag{3.12}$$

where f_{gc} represents the function for graph convolution.

3.4.5 State integrator

An integration process is needed to construct input data for the value estimator from the state data corresponding to each pedestrian. I propose two methods for the integration. The first method uses graph convolution and the second method is based on an occupancy map.

3.4.5.1 Integration using graph convolution

This method uses graph convolution for integrating the states of the PSRs corresponding to each pedestrian. The value estimator only receives the robot feature to which graph convolution is applied. The kernel function to be used for the adjacency matrix is the same as the state update of the GC-RPSR expressed in Eq. (3.8). The conceptual diagram of this method is shown in Figure 3.4.

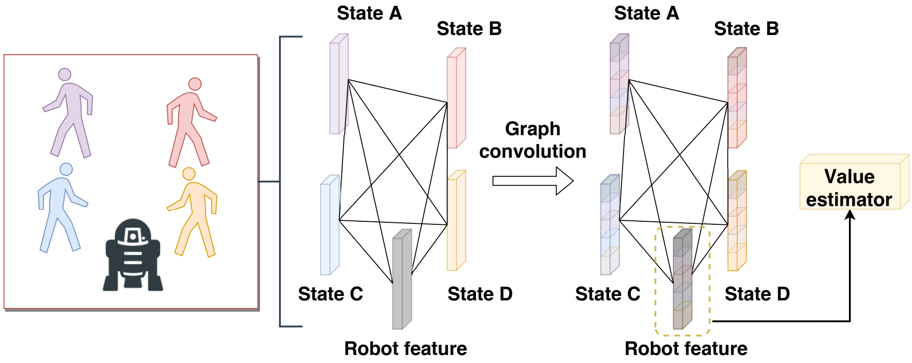


Figure 3.4: State integration using graph convolution.

3.4.5.2 Integration using occupancy map

This method uses an occupancy map for integrating the states of the PSRs corresponding to each pedestrian. The states of the pedestrians are stored in the corresponding cell depending on the position of each pedestrian. If multiple states are required to be stored in the same cell, an average value is stored. The conceptual diagram of this method is shown in Figure 3.5.

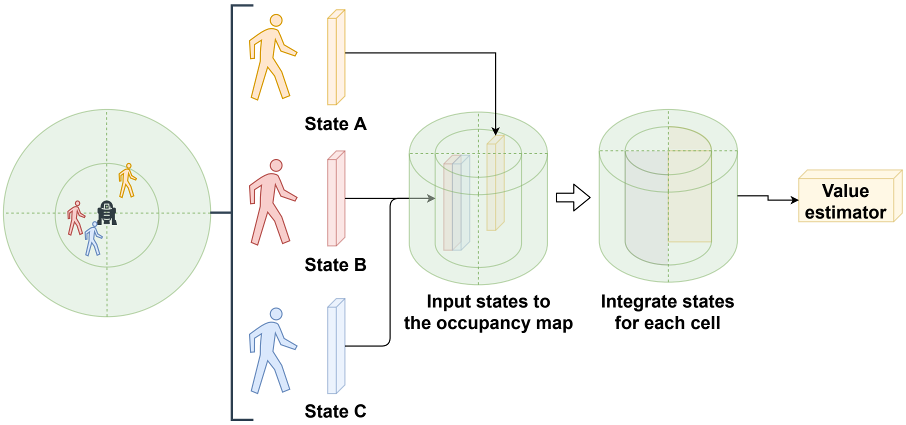

State A
State B
State C
Input states to the occupancy map
Integrate states for each cell
Value estimator

Figure 3.5: State integration using an occupancy map.

3.4.6 Action generation

The action is generated using the value estimator f_v, state updater f_p^q, and observation predictor f_p^o from the PSR model f_p by selecting an action obtaining the maximum reward from the action space $\mathcal{A}$. The formulation is shown in Eq. (3.13), where γ is the discount factor, s_t is the integrated state at time t, p_t is the robot position at time t, and $\hat{o}_t = f_p^o\left(q_t, a_t\right)$.

$$a_t \leftarrow \operatorname{argmax}_{a_t \in \mathcal{A}} R\left(\hat{o}_t\right) + \gamma^{\Delta t} f_v\left(s_t, p_t\right) \tag{3.13}$$

In addition, $R(o_t)$ is a reward function at time t. This function is expressed in Eq. (3.14), where d_t^s is the minimum separation distance between the robot and the pedestrians (distance between the centers minus each radius) and p_g is the goal position of the navigation task.

$$R(o_t) = \begin{cases} -0.25 & \text{if } d_t^s < 0 \\ -0.1 + d_t/2 & \text{else if } d_t^s < 0.2 \\ 1 & \text{else if } p_t = p_g \\ 0 & \text{otherwise} \end{cases} \tag{3.14}$$

Accordingly, I use d-step planning [48] that considers a d-step future. By rollout using the PSR and value estimator d-step, the action is selected with the maximum return along the d-step prediction using Eq. (3.15).

$$f_v^d\left(s_t, p_t\right) = \\ \begin{cases} f_v\left(s_t, p_t\right) & \text{if } d = 1 \\ \frac{1}{d} f_v^1\left(s_t, p_t\right) + \frac{d-1}{d} \max_{a_t}\big(& \\ R\left(\hat{o}_{t+1}\right) + \gamma f_v^{d-1}\left(\hat{s}_{t+1}, \hat{p}_{t+1}\right)\big) & \text{otherwise} \end{cases} \tag{3.15}$$

where $\hat{o}_{t+1} = f_p^o\left(q_{t+1}, a_{t+1}\right)$, and $\hat{q}_{t+1} = f_p^q\left(q_t, a_t, o_t\right)$.

3.4.7 Training process

Training is performed in two steps. The first step is an initialization of the PSR model and the imitation learning of the value estimator. Subsequently, I train the value estimator using RL and the PSR model using supervised learning.

3.4.7.1 Preliminary learning step

In the first step, the initial values of the PSR parameters are determined. This initialization is performed through two-stage regression [36] after collecting data using an exploration policy that follows ORCA [49] for initialization. Subsequently, the value estimator is trained in imitation learning using the collected data.

3.4.7.2 RL and supervised learning step

In the second step, I train the entire model with Algorithm 1, where E is the number of episodes for training and d is the target update frequency. The value estimator is trained with the RL procedure based on the bootstrapped DQN [50].

Algorithm 1: Optimization of the proposed method.

```
Initialize the PSR $f_p$ and the value estimator $f_v$ with two-stage regression and
  imitation learning
Initialize the target value estimator $\hat{f}_v$
for $i = 1$ to $E$ do
    Select $a_t$ following the exploration policy and obtain the reward $r_t$,
      observation $o_t$, and robot position $p_t$
    After finishing an episode, store the trajectory of $(o_t, a_t, r_t, p_t)$ to buffer $\mathcal{B}$

    Sample from the buffer $\mathcal{B}$ and obtain $M$ set of trajectories
    for $j = 1$ to $M$ do
        Obtain the trajectories below,
            observation $\mathbf{o}^j = \{o_1^j, o_2^j, \ldots, o_T^j\}$,
            robot position $\mathbf{p}^j = \{p_1^j, p_2^j, \ldots, p_T^j\}$,
            action $\mathbf{a}^j = \{a_1^j, a_2^j, \ldots, a_T^j\}$
        Calculate the trajectories below,
            state $\mathbf{q}^j = \{q_1^j, q_2^j, \ldots, q_T^j\}$,
            target value $\mathbf{y}^j = \{y_1^j, y_2^j, \ldots, y_T^j\}$,
            integrated state $\mathbf{s}^j = \{s_1^j, s_2^j, \ldots, s_T^j\}$
    end
    Update $f_p$ by minimizing $L_{\text{pred}} = \text{MSE}(f_p^o(\mathbf{q}, \mathbf{a}), \mathbf{o})$
    Update $f_v$ by minimizing $L_{\text{value}} = \text{MSE}(f_v(\mathbf{s}, \mathbf{p}), \mathbf{y})$
    if $i \bmod d$ then
        Update $\hat{f}_v$ by $\hat{f}_v \leftarrow f_v$
    end
end
```

3.5 Experiments for RPSR using NSK

In this experiment, I confirm that the NSK is beneficial for RPSR. I evaluated the performance in 1-step online prediction with benchmark on a synthetic nonlinear dynamical system used in related studies [35, 37] as follows:

$$\begin{aligned} \dot{x}_1(t) =& x_2(t) - 0.1\cos\left(x_1(t)\right)\left(5x_1(t) - 4x_1^3(t) + x_1^5(t)\right) - 0.5\cos\left(x_1(t)\right)a(t) \\ \dot{x}_2(t) =& -65x_1(t) + 50x_1^3(t) - 15x_1^5(t) - x_2(t) - 100a(t) \\ o(t) =& x_1(t) \end{aligned} \tag{3.16}$$

The input, a, is generated as white noise uniformly distributed between -0.5 and 0.5, and the target data for the prediction is output $o(t)$. I trained with 10 trajectories of 100 outputs and inputs, and tested with 5 trajectories. The trajectories for the training and testing data are shown in Figures 3.6 and 3.7, respectively. The models were trained with a future length of 10 and a past length of 20. In addition, 10000 random Fourier features were used and principal component analysis was conducted to project the features to 20 dimensions.

The prediction results of each case are shown in Figures 3.8 and 3.9. From these figures, it can be seen that the results with NSK are smoother and closer to the test data than those with the stationary case. In addition, Table 3.1 provides a numerical comparison of the prediction accuracies of all cases based on the mean square error (MSE). From this table, It can be seen that when NSK is used the accuracy is higher than when the stationary one is used. These facts can be attributed to the use of NSK, which allowed smoother estimation by learning global correlations and improved accuracy by covering non-trivial input regions. I confirmed the effectiveness of using NSK for RPSR from these results.

Table 3.1: Comparison of stationary and nonstationary spectral kernels. Each value is MSE between test data and prediction.

Trajectory No.	Stationary	Nonstationary
1	0.058	0.035
2	0.041	0.011
3	0.081	0.012
4	0.034	0.012
5	0.072	0.068
Avg. of 1 $\sim$ 5	0.057 ± 0.018	0.028 ± 0.022

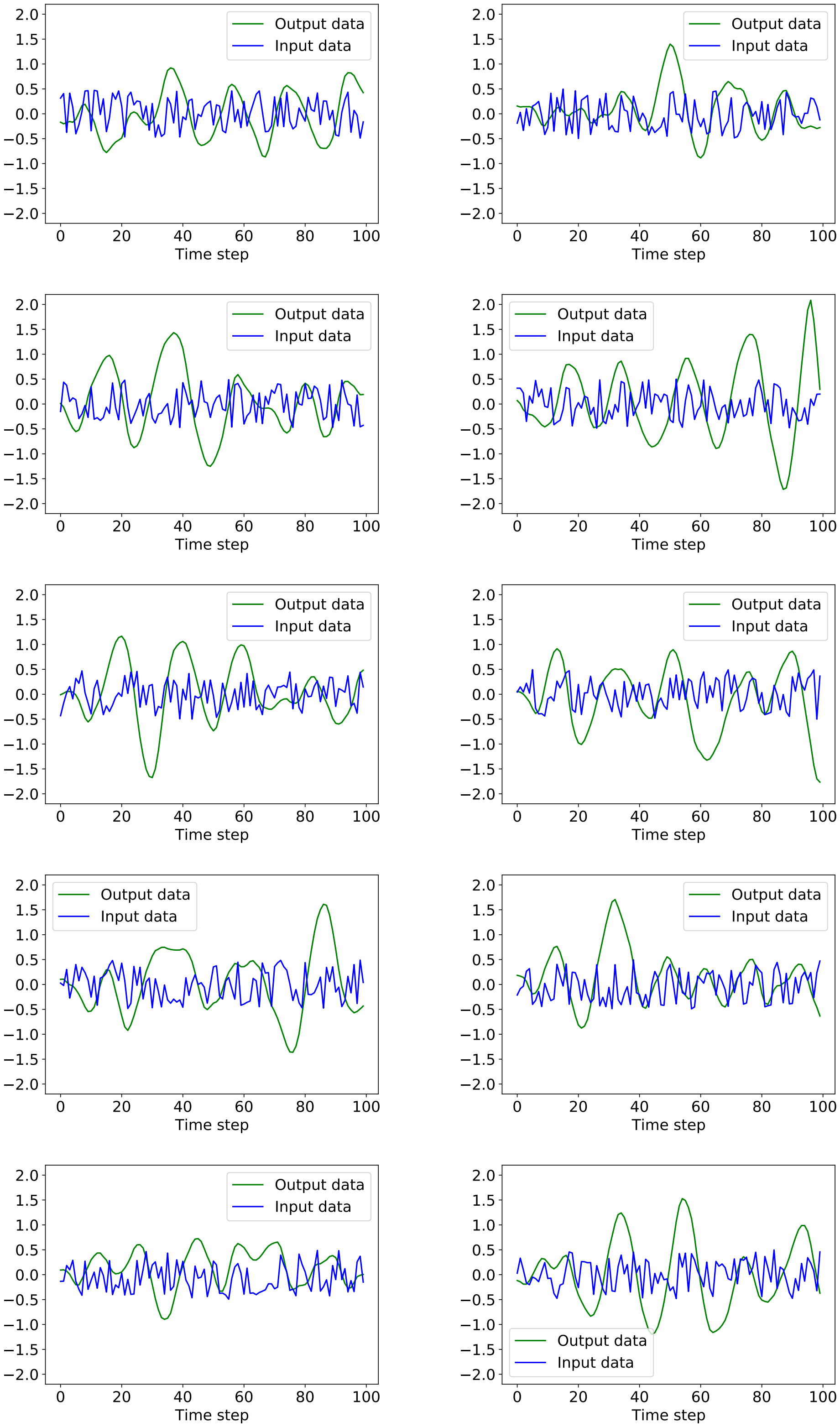


Figure 3.6: Trajectories for training.

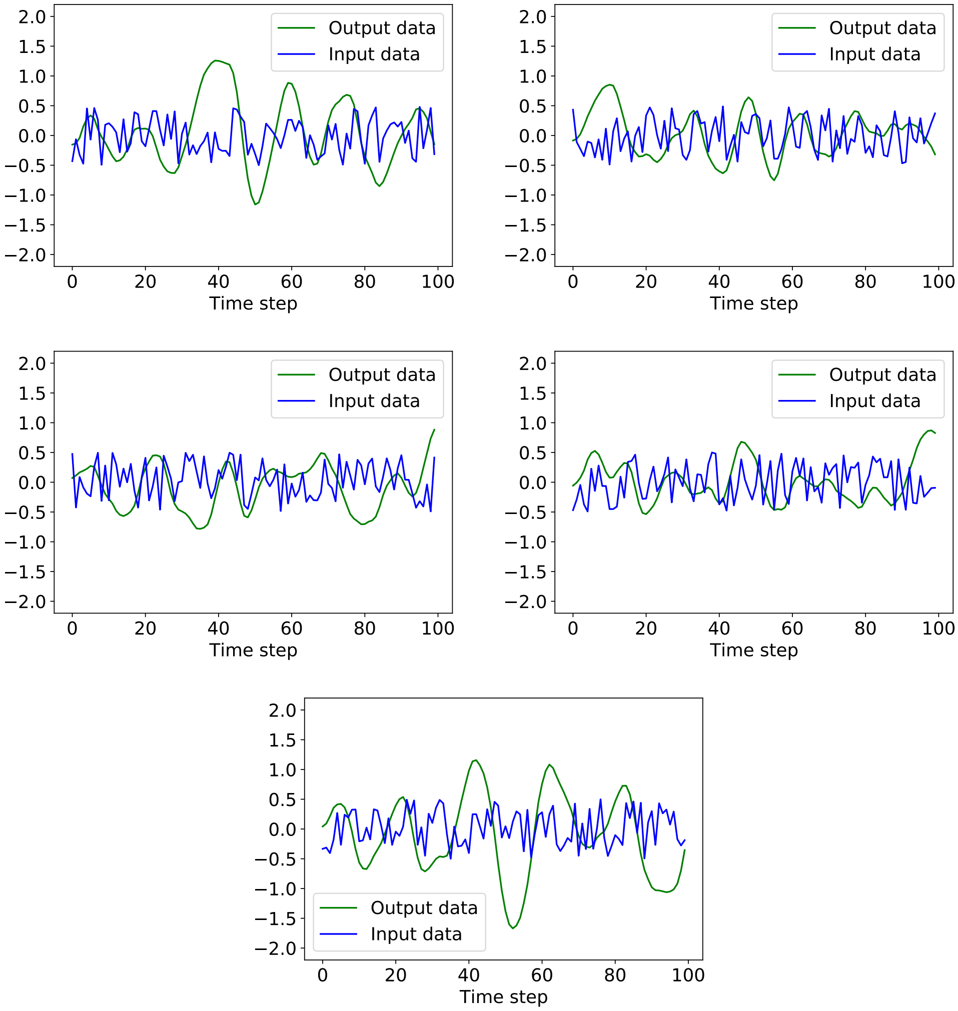


Figure 3.7: Trajectories for testing.

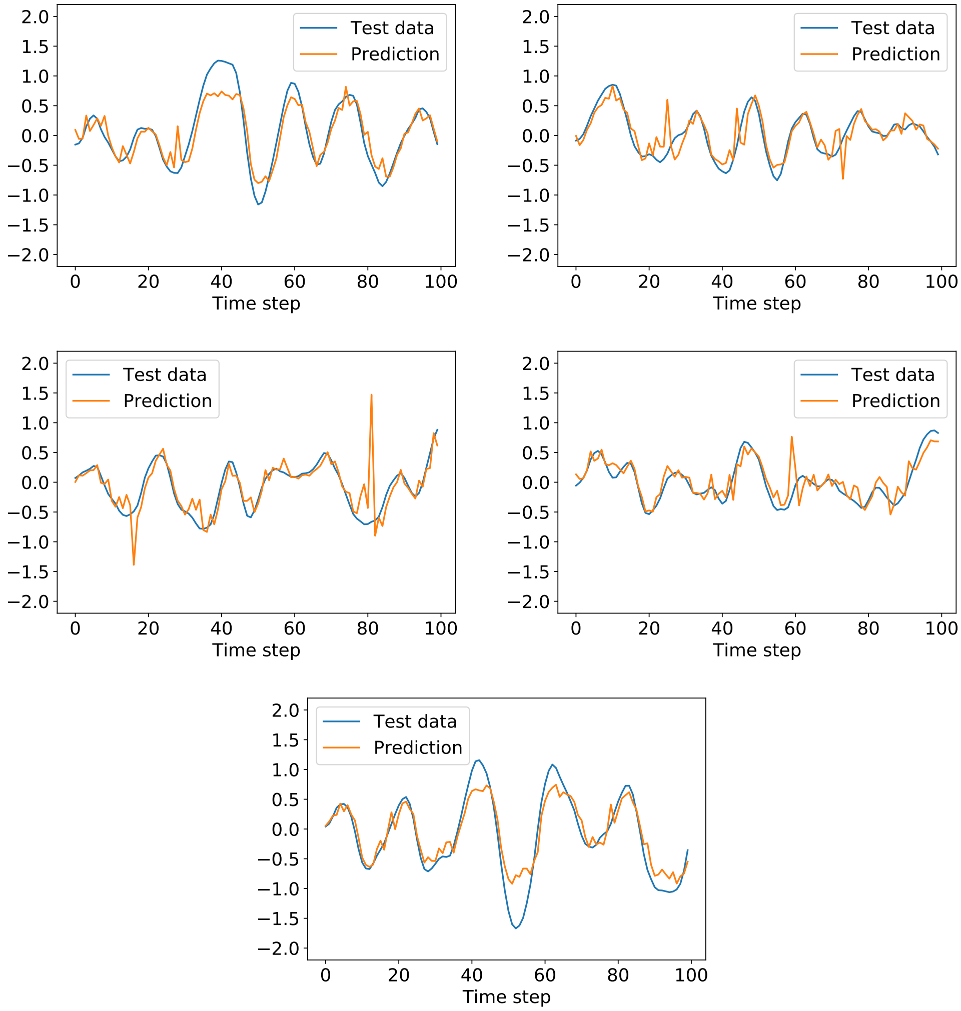


Figure 3.8: Prediction results using stationary spectral kernel.

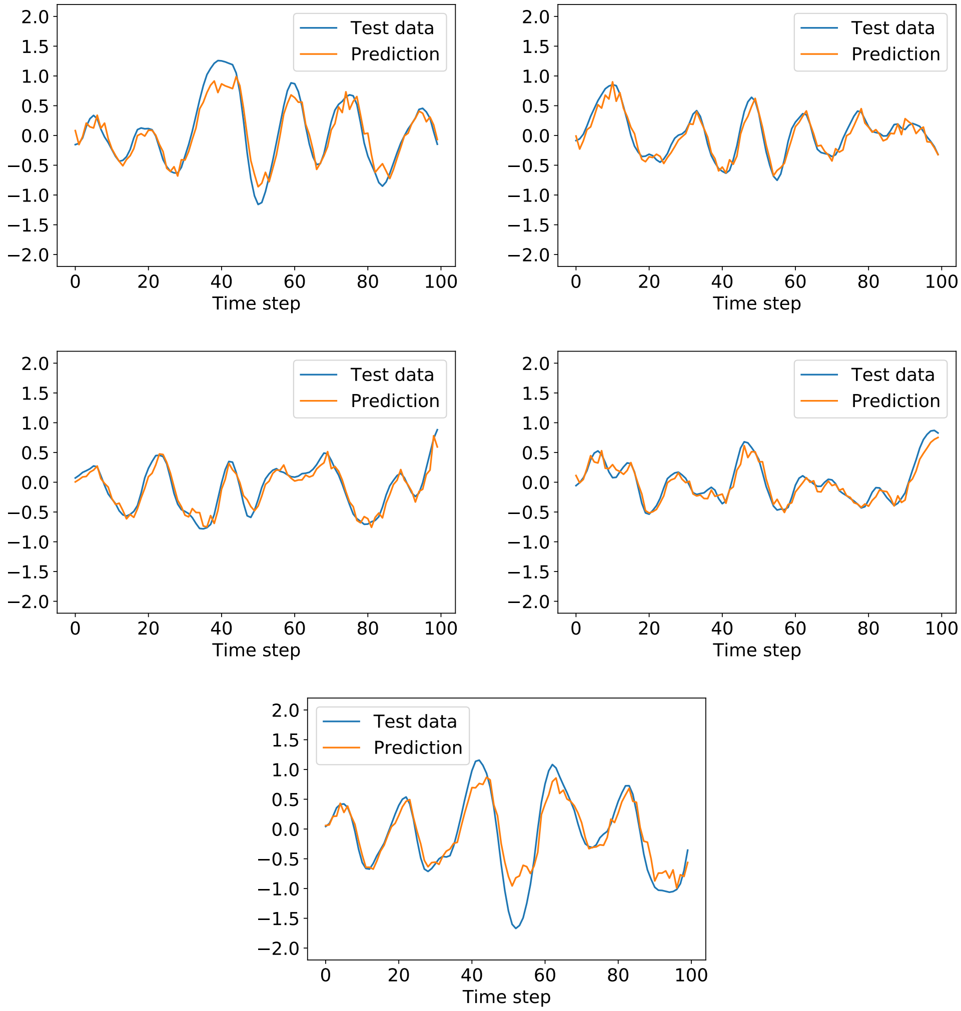


Figure 3.9: Prediction results using NSK.

3.6 Experiments for GC-RPSR

3.6.1 Implementation details

The structure of the value estimator is a multilayer perceptron with a hidden size of $\{500, 300\}$. In the RPSR structure, the feature extractor uses 500 random Fourier features and sets the PCA dimensionality reduction to 40 dimensions. The parameters are trained using AdaBelief [51] in the imitation, supervised, and reinforcement learning phases, and the learning rate is 10^{-4}. I trained my models in 10k episodes by setting the batch size of the trajectory to 5 and the target update frequency to 1k. During training, models were saved for every 100 episodes, and I used the best-performing model for comparison. The discount factor γ is set to be 0.9. The ε-greedy policy is used for the exploration with decays from 0.5 to 0.1 linearly in the first 4k episodes. The action space $\mathcal{A}$ consists of 80 discrete actions, consisting of 5 velocities exponentially spaced between (0, 1] and 16 headings evenly spaced between [0, 2π). Regarding action generation, I set the depth and width of d-step planning to 2. For the state integration of the second model, I use an occupancy map with a radius of 3 m, a distance resolution of 0.3 m, and an angular resolution of 30°.

3.6.2 Experimental environment

I use the circle crossing scenarios in the CrowdNav environment used in some related works [30, 32]. In this environment, pedestrians are controlled by ORCA [49]. The pedestrians are randomly positioned on a circle of radius 4 m with random perturbation added to their position (x, y). The robot and pedestrians are influenced by each other, and their behavior changes with position and velocity. The evaluation is conducted with 500 random test cases.

In the training phases of all the experiments and the testing phases of the first and second experiments, I set the number of pedestrians N to 5.

3.6.3 Comparison of GC-RPSR and RPSR

Here, I confirm the effectiveness of GC-RPSR by comparing it with the basic RPSR. I compare the success rate, collision rate, execution time, and average return. I use a simple concatenate for the state integrator. The result is shown in Table 3.2.

Table 3.2: Numerical comparison of RPSR and GC-RPSR.

Method	Success [%]	Collision [%]	Exec. time [s]	Avg. return
RPSR	57.8	33.8	8.84 ± 1.32	0.258
GC-RPSR	64.0	11.0	8.73 ± 1.31	0.356

From this table, it can be seen that GC-RPSR outperforms RPSR in terms of all the evaluation items. Accordingly, the proposed PSR model enhances the performance of the mobile robot navigation task compared with basic RPSR.

3.6.4 Comparison models with baseline

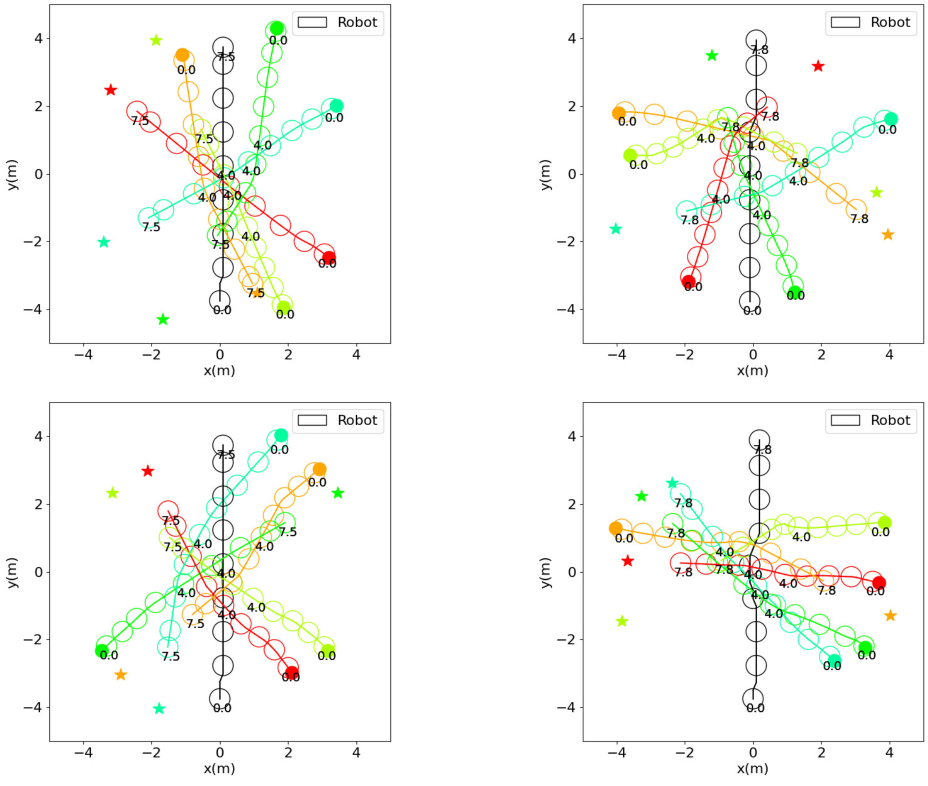


Figure 3.10: Sample trajectories of the results of the GGC-RPSR. The black trajectories describe the robot, and the colored trajectories describe pedestrians.

Here, I compare the proposed models with a method proposed in a related study [32], which is considered as the baseline. The evaluation items are the same as earlier. Figure 3.10 shows samples of the result trajectory of the proposed model with state integration using graph convolution (GGC-RPSR) and Figure 3.11 shows samples of the result trajectory of the proposed model with state integration using an occupancy map (OGC-RPSR). In each figure, the black trajectory represents the robot, the colored tra-

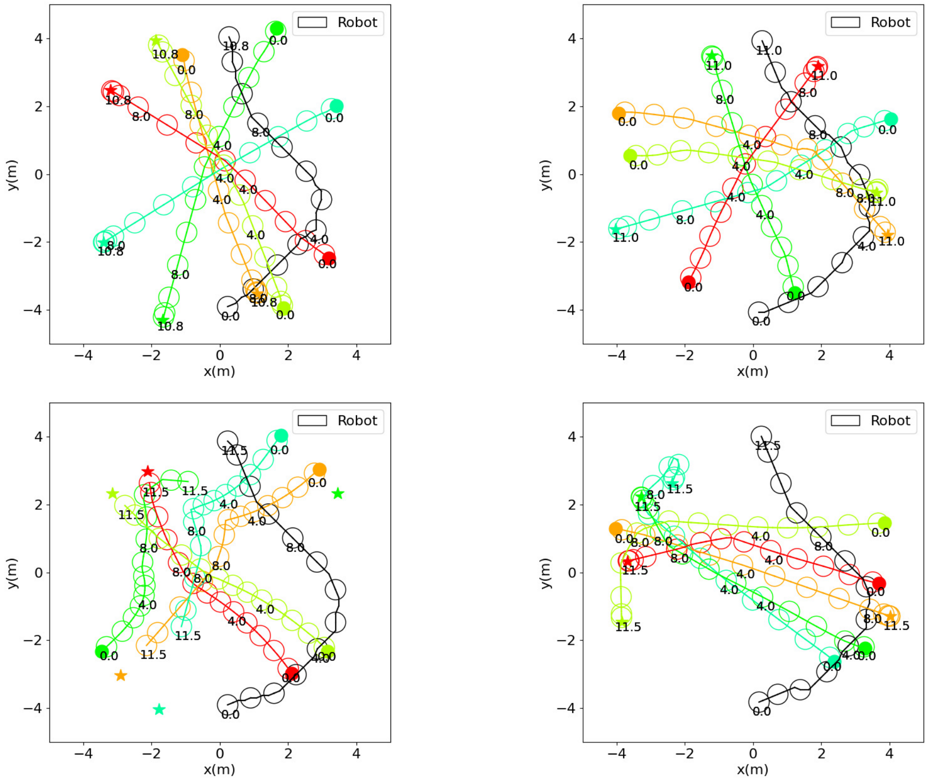


Figure 3.11: Sample trajectories of the results of OGC-RPSR. The black trajectories describe the robot, and the colored trajectories describe pedestrians.

jectories represent pedestrians, and the numbers indicate time steps. Both methods can navigate the robot to the goal while avoiding pedestrians. However, GGC-RPSR has a higher efficiency because it can generate shorter paths. The numerical comparison of the two proposed methods and the baseline method is shown in Table 3.3.

The GGC-RPSR outperforms the other methods in terms of all the evaluation items. This result demonstrates the effectiveness of the proposed method using GC-RPSR and state integration using graph convolution. I speculate that the reason why the OGC-RPSR outperformed the baseline is that the structure of the PSR was effective in achieving a more optimal path planning. In addition, since the optimization performance of the state integration method using graph convolution is higher, I be-

Table 3.3: Numerical comparison of the proposed methods and the baseline method.

Method	Success [%]	Collision [%]	Exec. time [s]	Avg. return
RGL [32]	92.0	7.0	9.09 ± 1.31	0.560
GGC-RPSR	94.6	5.4	7.79 ± 0.10	0.587
OGC-RPSR	91.4	6.8	11.54 ± 1.03	0.512

lieve that the GGC-RPSR performed better than the OGC-RPSR in this experiment where the number of pedestrians during training and testing is the same.

3.6.5 Comparison of the models in the situation where the number of pedestrians differs between training and testing

Here, I compared the two proposed methods in the situation where the number of pedestrians differs between training and testing. Both the models were trained using an environment in which the number of pedestrians was 5 and were tested in environments in which the number of pedestrians was 1 to 10.

Figure 3.12 and Figure 3.13 show the success case of result trajectories of GGC-RPSR. In addition, Figure 3.14 and Figure 3.15 show the success case of result result trajectories of OGC-RPSR. From these trajectories, it can be seen that GGC-RPSR can generate efficient trajectories to reach to goal quickly.

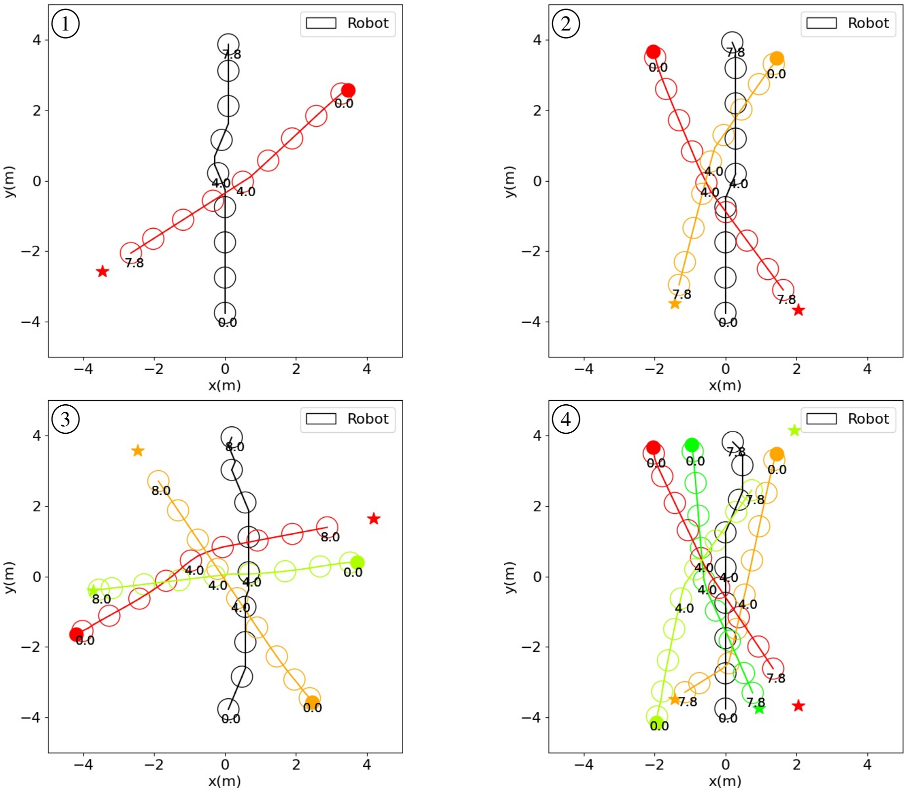


Figure 3.12: Sample result trajectories of GGC-RPSR in environments in which number of pedestrians is 1–4. Black trajectories represent robot, colored trajectories denote pedestrians, and circled numbers in upper left corner represent pedestrians in environment.

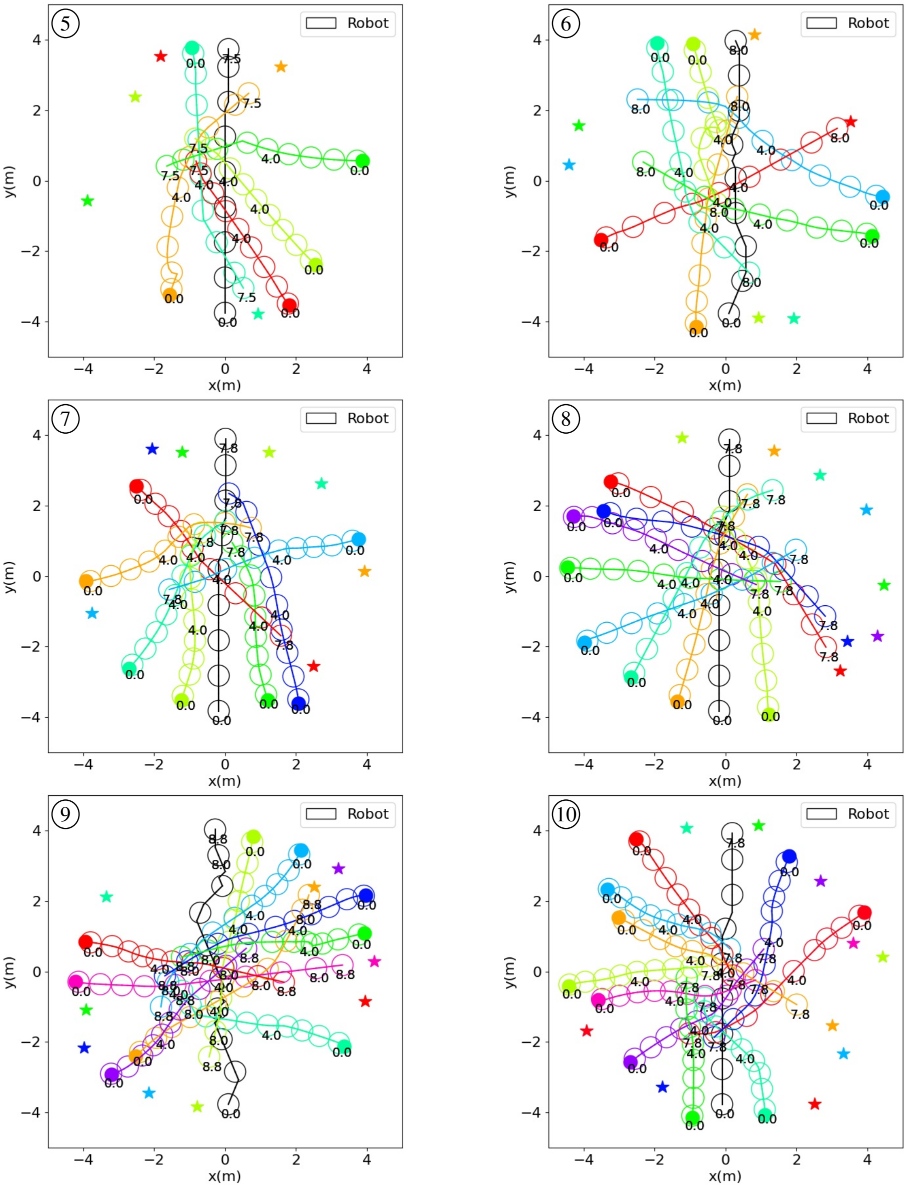


Figure 3.13: Sample result trajectories of GGC-RPSR in environments in which number of pedestrians is 5–10. Black trajectories represent robot, colored trajectories denote pedestrians, and circled numbers in upper left corner represent pedestrians in environment.

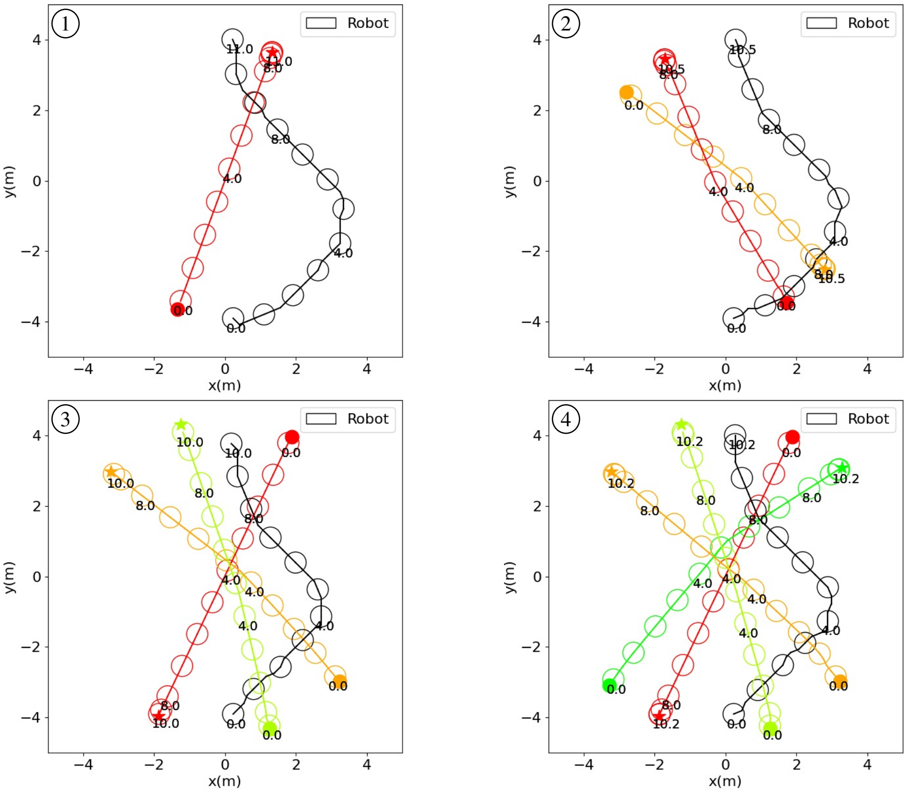


Figure 3.14: Sample result trajectories of OGC-RPSR in environments in which number of pedestrians is 1–4. Black trajectories represent robot, colored trajectories denote pedestrians, and circled numbers in upper left corner represent pedestrians in environment.

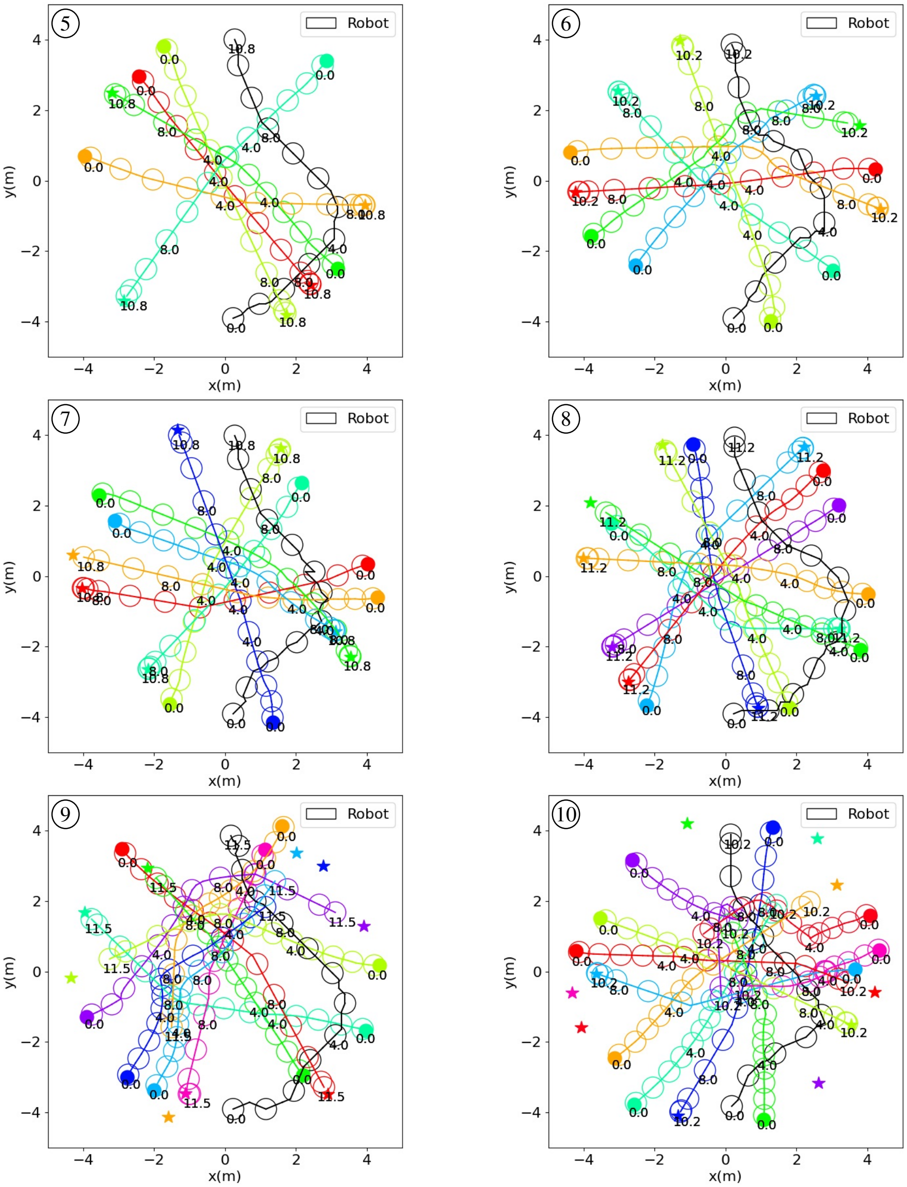


Figure 3.15: Sample result trajectories of OGC-RPSR in environments in which number of pedestrians is 5–10. Black trajectories represent robot, colored trajectories denote pedestrians, and circled numbers in upper left corner represent pedestrians in environment.

Table 3.4: Success rate of GGC-RPSR and OGC-RPSR while changing the number of pedestrians N (in percentage).

Method	$N = 1$	$N = 2$	$N = 3$	$N = 4$	$N = 5$	$N = 6$	$N = 7$	$N = 8$	$N = 9$	$N = 10$
GGC-RPSR	63.6	68.2	65.8	91.0	94.6	76.2	93.6	89.0	51.2	89.0
OGC-RPSR	98.2	95.0	94.4	92.0	91.4	88.0	91.2	91.2	84.0	85.8

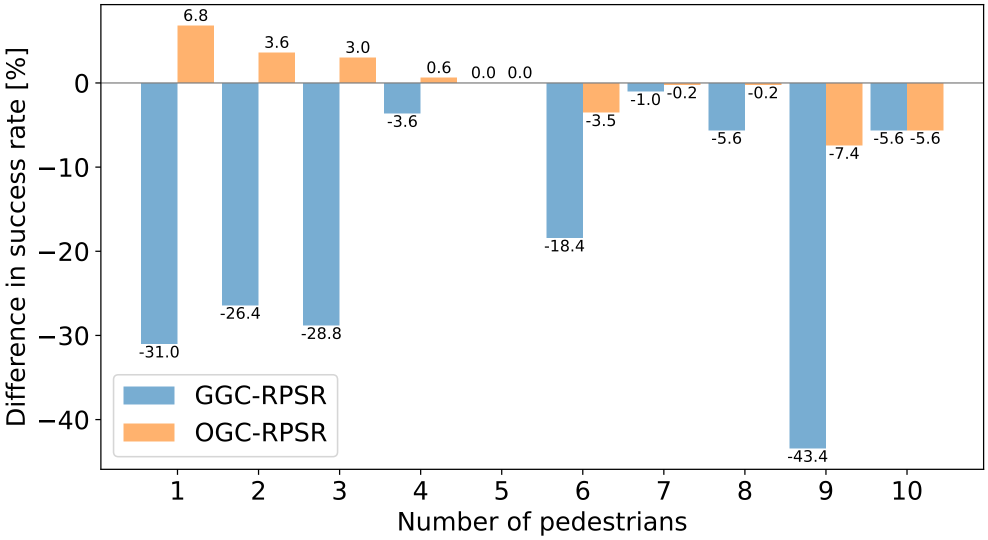


Figure 3.16: Difference in success rate from the result in the situation with 5 pedestrians.

The success rate of the two models are shown in Table 3.4. In addition, Figure 3.16 shows the difference in success rate from the result in the situation with 5 pedestrians of each model. The maximum decline of the success rate of GGC-RPSR is 43.4%; in contrast, the corresponding result for OGC-RPSR is 7.4%. From the results, it can be seen that although GGC-RPSR can generate efficient paths and appears to have high performance, OGC-RPSR is more stable when the number of pedestrians is varied between training and testing.

Comparing the two methods, GGC-RPSR integrates the states corresponding to each pedestrian into a single feature, regardless of the number of pedestrians. Therefore, it can be inferred that it may not work well with the number of untrained pedestrians. On the other hand, OGC-RPSR integrates the states corresponding to the pedestrians in each cell. That fact may often result in only a certain number of states being integrated into each cell, regardless of the number of pedestrians. Therefore, it is speculated that OGC-RPSR performed more stable. However, since I have not been able to verify the results sufficiently, I need to conduct more detailed discussions and experiments in the future to determine the cause of the results.

3.7 Conclusion

This study used a deep reinforcement learning method based on PSR in mobile robot navigation tasks in a dynamic environment with pedestrians. I proposed a new PSR architecture for mobile robot navigation in a dynamic environment with pedestrians, considering the interactions among pedestrians. In addition, I proposed methods for integrating the state of the PSR corresponding to each pedestrian for dealing with the change in the number of pedestrians between training and testing.

The conducted experiments showed the effectiveness of the proposed model. In addition, I confirmed that OGC-RPSR, which uses an occupancy map-based state integration, is more stable in the scenario in which the number of pedestrians differs between training and testing.

The two types of state integration methods proposed in this study have not yet been fully discussed; in future studies, further experiments are needed to clarify in detail the causes of the differences between the two methods. For the occupancy map-based method, more experiments are needed to clarify the relationships among parameters such as the number of grids and the performance change. In addition, in the future, I will consider the more effective model and the learning process in a real-world scenario. The learning costs and the exploration processes are critical problems in real-world deep reinforcement learning tasks. If a robot can learn in the real world directly, the trained system can perform better in actual environments than in simulated environments. However, the robot would be required to repeat its interactions with the environment and learn from numerous success and failure patterns, which is difficult to achieve practically. Thus, a more efficient learning process is required to overcome these problems.

4 Conclusion and Outlook

In this chapter, I summarize the contributions of the studies presented in this dissertation and further discuss the remaining challenges for the two targets. Finally, I discuss the future directions in terms of autonomous mobile service robot systems that consider more real-world scenarios and learning-based mobile robot navigation that is more flexible and fully usable in real-world environments.

4.1 Summary

The contributions can be summarized under two topics: development of tour guide and co-experience robots and learning-based navigation method based on PSR.

Development of tour guide and co-experience robots: In Chapter 2, I compared the performance of the QZSS, a Japanese GNSS that can perform positional positioning with a single module, with that of an existing high-precision GNSS. A simple module implies an advantage for the operation of an autonomous mobile robot; therefore, I believe that the QZSS has a major benefit. In addition, my main contributions are the development of a tour guide robot system using the QZSS and the development and field experiments of a co-experience robot system using the latest high-capacity communication system, 5G. The experiments confirmed that the desired tasks can be performed by the mobile robots. The challenges include developing a method to understand whether a user is following the robot when providing a tour guide and transferring more information about the experience of the robot to the user side than simply videos in the co-experience system. In addition, it is necessary to devise evaluation criteria for each system as a service and conduct experiments to clarify them.

Learning-based navigation method based on PSR: In Chapter 3, I considered the navigation of a mobile robot in a dynamic environment by a pedestrian and proposed a learning-based navigation method that considers the fact that the behavior of a pedestrian changes with the behavior of the robot using PSR. In addition, I proposed a more effective PSR model to model the pedestrian behavior and two state integration methods to work even when the number of pedestrians is different from the learning time. By experiments I demonstrated the effectiveness of the proposed methods based on the new PSR model and compared the state integration methods. The remaining task is to implement the proposed method on a real robot and conduct experiments in the real world. In addition, a method that enables learning in a real environment is required to fully utilize the fact that the method is learning-based.

4.2 Outlook

Expectations for service robot systems such as the ones proposed in this study will continue to rise to address various social problems. For robots to operate in a real environment, it is important to find a suitable method to implement them, and this is the foundation for any robot to perform services. The development of such robots requires various technologies, and I believe that it is important to actively incorporate evolving technologies such as the QZSS and 5G, as discussed in this study. In addition, from the perspective of providing services, it is important for a robot to be user-friendly. The tour guide robot proposed in this study can receive voice requests, travel to a destination, and explain sightseeing points. However, it does not have a function to determine whether the user is following it closely during the guidance. To achieve this, a system that can accurately identify the user from among the various pedestrians that pass through the theme park environment is needed. In addition, the system needs to be able to pause or adjust its speed if a pedestrian is not following. Accordingly, I believe that future progress in this study will be achieved by considering mechanisms for improving the quality of services and policies for evaluating robots, instead of simply focusing on constructing robots. In addition, using 5G, it is possible to send and receive large amounts of data with low latency. Although the implementation of the tour guide robot in this study did not utilize these features, I believe that it is possible to improve the system using the data acquired from the robot for obstacle detection and location estimation using methods such as deep learning.

In contrast, for the co-experience system, the most important point is that the user can feel the experience of the robot as if it was a real experience. In this study, I developed a system to share the vision experienced by the robot by transmitting 360-degree 4k images using 5G high-capacity communication. However, because humans have other important senses besides vision, I believe that it will be possible to obtain and share more information to achieve a more immersive experience. The senses of hearing, smell, taste, and touch can be considered. In the case of hearing, it is relatively easy to share sounds using a microphone and earphones or speakers. For the senses of smell and taste, by installing sensors on the robot, it is possible to transfer the data of the experience of the robot; however, finding a method to feed it back to the user is a major challenge. Regarding the sense of touch, both methods for sensing and feed back to the user are significant challenges. In my future work, I believe that I can raise the level of immersion of the shared experience by exploring these points.

For an autonomous mobile service robot to perform its task smoothly, it must be able to move autonomously even in an environment crowded with pedestrians. However, the behavior of pedestrians may be affected by environmental factors, such as the surrounding pedestrians and the shape of the road, and may also change depending on their intentions at the time. It is difficult to model this behavior explicitly; therefore, it is desirable to develop a learning-based method based on collected data.

The method proposed in this study considers the fact that the behavior of a pedestrian changes depending on the behavior of the robot. In a real environment, more factors associated with each environment need to be considered. One of the approaches to solve this problem is to learn by collecting data in a real operating environment. Considering the fact that the behavior of a robot affects people, it may be impossible to deal with the problem by simply acquiring data in advance. Moreover, because a robot may be unable to sense all information, it will be necessary to learn directly in each environment to consider that the intentions of pedestrians may change depending on the road shape, stores, and other structures that attract their attention. For this purpose, an online learning method is needed while collecting data in real environments.

The reinforcement learning method used in this study can also be viewed as a method that learns sequentially while collecting data; however, the fact that data are collected randomly during data collection is a problem in a real environment. When an autonomous mobile robot behaves randomly in a real environment, there is a risk of collisions with objects and pedestrians. In addition, if a robot is designed to perform a service, it will be necessary to collect data while performing the service; therefore, it will be problematic if the robot behaves in a way that it cannot achieve the service. Considering the above, I believe that I need a system in which a robot that initially operates with a simple model-based approach gradually transforms into a smart system as it collects data and learns. Studying such intelligent robot systems that can learn in the field is a future prospect.

Appendices

A

Supplementary Explanation

A.1 Differential wheeled mobile robot

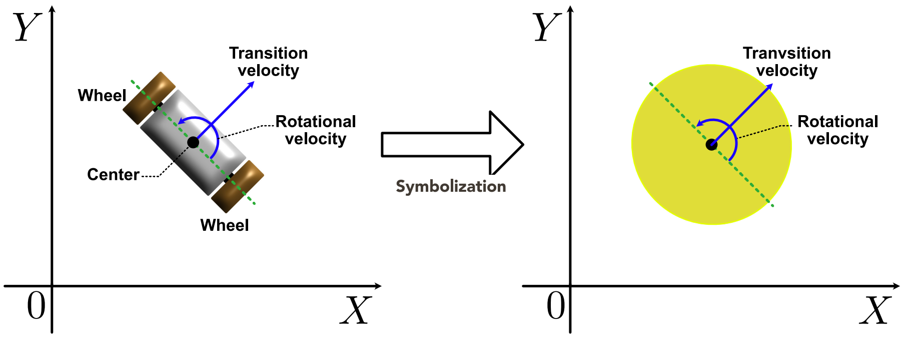


Figure A.1: Fundamental differential wheeled mobile robot in 2D space and its simplified symbol.

In the research in this dissertation, I deal with differential wheeled-type mobile robots. A differential wheeled mobile robot is equipped with two wheels, and by controlling the rotation of these wheels, the robot can move straight or turn. A robot with this structure can easily perform a spin turn; therefore, it can move flexibly. Figure A.1 shows a differential wheeled mobile robot in 2D space and its simplified symbol. In 2D space, the factors to consider for controlling differential wheeled mobile robots are position and orientation (x, y, θ) and velocity (v, ω) as the state of the robots. In addition, velocity is considered as the input command for controlling such robots. Figure A.2 shows the relations among the input command and position, orientation, and velocity information in the global coordinate system.

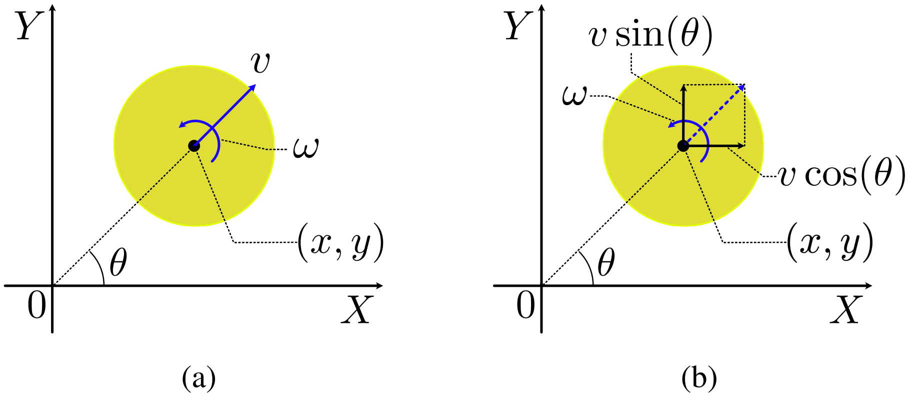


Figure A.2: (a) Command for controlling robot (velocity), (b) position, orientation, and velocity information in global coordinate system.

Velocity in the 2D global coordinate system is described by decomposing v into x and y components, as expressed in Eq. A.1.

$$\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \begin{pmatrix} v\cos(\theta) \\ v\sin(\theta) \end{pmatrix} \tag{A.1}$$

In addition, the angular velocity is expressed below.

$$\dot{\theta} = \omega \tag{A.2}$$

The robot's position can be estimated in a discrete system by summing the incremental distance traveled. The transition of the robot while time t and $t + 1$ is described below [52].

$$x_{t+1} = x_t + v\Delta t \cos\left(\theta_t + \omega\Delta t/2\right) \tag{A.3}$$

$$y_{t+1} = y_t + v\Delta t \sin\left(\theta_t + \omega\Delta t/2\right) \tag{A.4}$$

$$\theta_{t+1} = \theta_t + \omega\Delta t \tag{A.5}$$

where Δt is time difference between t and $t+1$.

A.2 Reinforcement Learning for Autonomous Mobile Robot Navigation

In recent years, machine learning methods have been rapidly developing, and deep learning, in particular, has produced good results in various fields and has also been applied in robotics. Various methods using machine learning have been proposed for the abovementioned elements: object detection and tracking, localization, and path planning [53, 54, 55, 56, 32, 33]. Supervised learning methods are used for object detection, tracking, and location estimation, whereas reinforcement learning and imitation learning are frequently used for path planning. In the study in this dissertation, I specifically focus on path planning using reinforcement learning. Reinforcement learning is a subfield of machine learning, which trains an agent how to act in environments. The procedure of reinforcement learning is interacting with an environment, to learn the good actions to be taken to maximize rewards. Figure A.3 shows a conceptual diagram of reinforcement learning. Reinforcement learning methods are classified

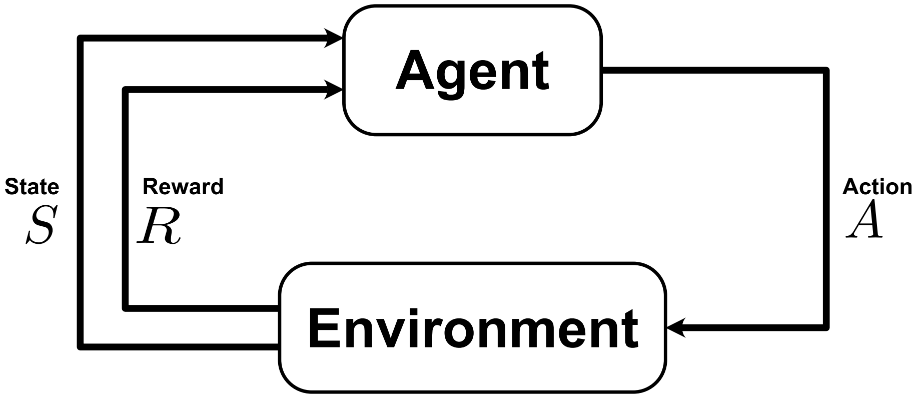


Figure A.3: Conceptual diagram of reinforcement learning.

into two types: off-policy and on-policy methods. In on-policy reinforcement learning, the policy, π_k, is updated with the data collected by itself. It optimizes the current policy and uses it to collect data. Specifically, it improves the same policy that the agent is already using for exploration. In off-policy learning, old samples can be used for computation. For updating a policy, experiences are sampled from a buffer collected from its former policy. This improves the efficiency of the sample as it does not need to be re-collected each time the policy is changed. Sample efficiency is an important element when reinforcement learning methods are used in difficult data collection problems, such as mobile robot navigation in the real world. Another major focus in reinforcement learning is on whether to learn a strategy. The actor–critic method is an off-policy method that learns a policy. It is a combination of policy learning and value learning, where the critic is learned to estimate the value of the action and the actor is learned to generate the action with the maximum value obtained by the critic.

With the development of deep learning, deep neural networks (DNNs) have begun to be used in reinforcement learning in recent years. A DNN can approximate arbitrary functions and is commonly used, particularly when dealing with values and behaviors with continuous values. The simplest one-layer neural network is represented by the equation, $y = ax + b$. This is called the fully connected layer in the DNN model structure and is used as the basic structure. Adjusting the parameters, a and b, in this equation implies training this network. This parameter adjustment is determined by calculating the gradient by partial differentiation concerning a and b. By combining this simple structure with an activation function to represent the nonlinearity, multilayered DNN models can be constructed.

The action space of mobile robot navigation consists of a combination of translational velocity v and rotational velocity ω or a combination of x-direction velocity v_x and y-direction velocity v_y in 2D space. This action space is not complex, and owing to hardware and other constraints, it is not typically possible for a real robot to perform strictly continuous-valued actions. Therefore, deep reinforcement learning-based mobile robot navigation can be realized in many cases using a value estimator based on DNN models to evaluate the values sampled from the action space and select the optimal action. Figure A.4 shows a process for training of value learning using a DNN. A simple explanation of this training is as follows:

1. The agent acts and obtains the state and reward after the action.

2. The action, state, and reward are stored as a single combination in a buffer.

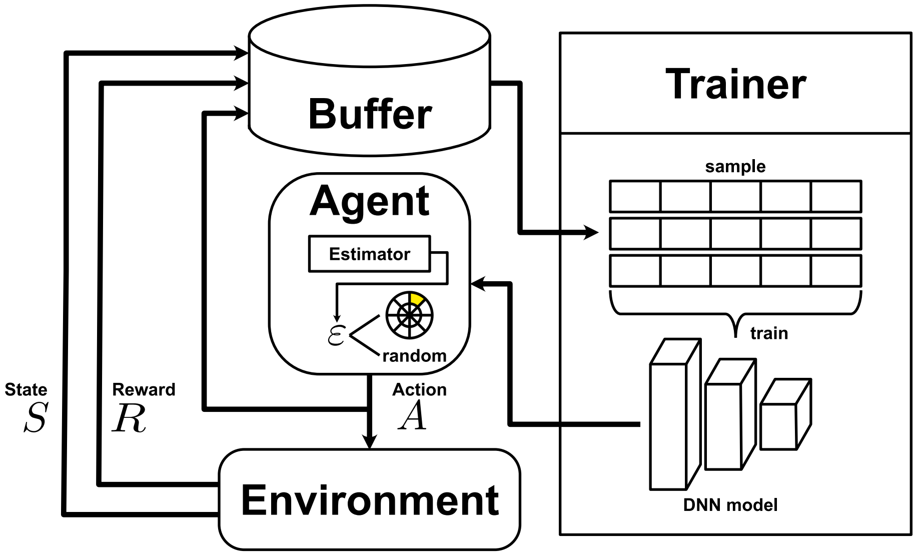


Figure A.4: Process for training of value learning using DNN.

3. The trainer reads data randomly from the buffer and trains the DNN model to perform value estimation.
4. The course of action of the agent is updated to the learned value estimation model.

It is crucial to explore action generation to obtain samples that have not been experienced before when collecting data. In the case of value learning, the ε-greedy method is generally used, which acts randomly with a certain probability based on a threshold value ε.

Bibliography

[1] "Quasi-Zenith Satellite Orbit (QZO) | Technical Information | QZSS (Quasi-Zenith Satellite System) - Cabinet Office (Japan)." `http://qzss.go.jp/en/technical/technology/orbit.html`, Feb. 2019.

[2] R. Triebel, K. Arras, R. Alami, L. Beyer, S. Breuers, R. Chatila, M. Chetouani, D. Cremers, V. Evers, M. Fiore, H. Hung, O. A. Ramírez, M. Joosse, H. Khambhaita, T. Kucner, B. Leibe, A. J. Lilienthal, T. Linder, M. Lohse, M. Magnusson, B. Okal, L. Palmieri, U. Rafi, M. Van Rooij, and L. Zhang, "SPENCER: A socially aware service robot for passenger guidance and help in busy airports," *Field and Service Robotics*, vol. 113, pp. 607–622, 2016.

[3] H. M. Gross, A. Scheidig, S. Müller, B. Schütz, C. Fricke, and S. Meyer, "Living with a mobile companion robot in your own apartment - Final implementation and results of a 20-weeks field study with 20 seniors," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2253–2259, 2019.

[4] H. Andersen, Y. H. Eng, W. K. Leong, C. Zhang, H. X. Kong, S. Pendleton, M. H. Ang, and D. Rus, "Autonomous personal mobility scooter for multi-class mobility-on-demand service," in *Proceedings of the IEEE Conference on Intelligent Transportation Systems (ITSC)*, pp. 1753–1760, 2016.

[5] S. Witwicki, J. C. Castillo, J. Messias, J. Capitan, F. S. Melo, P. U. Lima, and M. Veloso, "Autonomous Surveillance Robots: A Decision-Making Framework for Networked Muiltiagent Systems," *IEEE Robotics and Automation Magazine*, vol. 24, no. 3, pp. 52–64, 2017.

[6] D. Fox, W. Burgard, and S. Thrun, “The dynamic window approach to collision avoidance,” *IEEE Robotics Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.

[7] M. Saito, A. Yamagishi, J. Takiguchi, and K. Asari, “Introduction to High-Accuracy Satellite-based Positioning System utilizing QZSS,” Tech. Rep. 26, Mitsubishi Space Software Co.,Ltd., 2016.

[8] G. Wan, X. Yang, R. Cai, H. Li, Y. Zhou, H. Wang, and S. Song, “Robust and Precise Vehicle Localization Based on Multi-Sensor Fusion in Diverse City Scenes,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4670–4677, 2018.

[9] M. Imperoli, C. Potena, D. Nardi, G. Grisetti, and A. Pretto, “An Effective Multi-Cue Positioning System for Agricultural Robotics,” *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3685–3692, 2018.

[10] P. Geneva, K. Eckenhoff, and G. Huang, “Asynchronous Multi-Sensor Fusion for 3D Mapping and Localization,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[11] G. Kouros, I. Kotavelis, E. Skartados, D. Giakoumis, D. Tzovaras, A. Simi, and G. Manacorda, “3D Underground Mapping with a Mobile Robot and a GPR Antenna,” in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3218–3224, 2018.

[12] J. Almeida, A. Ferreira, B. Matias, C. Lomba, A. Martins, and E. Silva, “¡VAMOS! Underwater Mining Machine Navigation System,” in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1520–1526, 2018.

[13] M.-A. Raúl and T. J. D., “ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras,” *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[14] J. Zhang and S. Singh, “Loam: Lidar odometry and mapping in real-time,” *Robotics: Science and Systems*, vol. 2, no. 9, 2014.

[15] T. Takasu and A. Yasuda, “Development of the low-cost RTK-GPS receiver with an open source program package RTKLIB,” *International Symposium on GPS/GNSS*, 2009.

[16] R. Kurazume, Y. Pyo, K. Nakashima, T. Tsuji, and A. Kawamura, “Feasibility study of iort platform “big sensor box”,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3664–3671, 2017.

[17] J. Sakamoto, K. Kiyoyama, K. Matsumoto, Y. Pyo, A. Kawamura, and R. Kurazume, “Development of ROS-TMS 5.0 for Informationally Structured Environment,” *ROBOMECH Journal*, vol. 5, no. 24, 2018.

[18] G. Bresson, Z. Alsayed, L. Yu, and S. Glaser, “Simultaneous Localization and Mapping: A Survey of Current Trends in Autonomous Driving,” *IEEE Transactions on Intelligent Vehicles*, vol. 2, no. 3, pp. 194–220, 2017.

[19] J. K. Suhr, J. Jang, D. Min, and H. G. Jung, “Sensor Fusion-Based Low-Cost Vehicle Localization System for Complex Urban Environments,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 5, pp. 1078–1086, 2017.

[20] S. Fujuta, M. Miya, and J. Takiguchi, “Establishment of Centimeter-class Augmentation System utilizing Quasi-Zenith Satellite System (<Special Issue>Recent Topics on GNSS and Its Applications),” *System/Control/Information*, vol. 59, no. 4, pp. 126–131, 2015.

[21] Cabinet Office (Japan), “Quasi-Zenith Satellite System Interface Specification Centimeter Level Augmentation Service (IS-QZSS-L6-001),” tech. rep., 2018.

[22] G. Wübbena, M. Schmitz, and A. Bagge, “PPP-RTK : Precise Point Positioning Using State-Space Representation in RTK Networks,” in *Proceedings of the International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS)*, 2005.

[23] “SoftEther VPN Project.” `https://www.softether.org/`, Apr. 2020.

[24] “SimpleRT.” `https://github.com/vvviperrr/SimpleRT`, Apr. 2020.

[25] T. Moore and D. Stouch, “A Generalized Extended Kalman Filter Implementation for the Robot Operating System,” in *Proceedings of the International Conference on Intelligent Autonomous Systems (IAS)*, pp. 335–348, 2016.

[26] "DOCOMO AI Agent API." `https://docs.sebastien.ai/`, Apr. 2020.

[27] "WebRTC." `https://webrtc.org/`, Apr. 2020.

[28] "A-Frame: Hello WebVR." `https://aframe.io/`, Apr. 2020.

[29] M. Everett, Y. F. Chen, and J. P. How, "Motion Planning among Dynamic, Decision-Making Agents with Deep Reinforcement Learning," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3052–3059, 2018.

[30] C. Chen, Y. Liu, S. Kreiss, and A. Alahi, "Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6015–6022, 2019.

[31] Y. Chen, C. Liu, B. E. Shi, and M. Liu, "Robot Navigation in Crowds by Graph Convolutional Networks with Attention Learned from Human Gaze," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2754–2761, 2020.

[32] C. Chen, S. Hu, P. Nikdel, G. Mori, and M. Savva, "Relational graph learning for crowd navigation," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10007–10013, 2020.

[33] L. Lucia, D. Daniel, C. Gianluca, S. Roland, and D. Renaud, "Robot Navigation in Crowded Environments Using Deep Reinforcement Learning," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5671–5677, 2020.

[34] S. Singh, M. James, and M. Rudary, "Predictive State Representations: A New Theory for Modeling Dynamical Systems," in *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, pp. 512–519, 2004.

[35] B. Boots, A. Gretton, and G. J. Gordon, "Hilbert space embeddings of predictive state representations," in *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, pp. 92–101, 2013.

[36] A. Hefny, C. Downey, and G. J. Gordon, "Supervised learning for dynamical system learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 1963–1971, 2015.

[37] A. Hefny, C. Downey, and G. Gordon, “An efficient, expressive and local minima-free method for learning controlled dynamical systems,” in *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 3191–3198, 2018.

[38] A. Venkatraman, N. Rhinehart, W. Sun, L. Pinto, M. Hebert, B. Boots, K. M. Kitani, and J. A. Bagnell, “Predictive-state decoders: Encoding the future into recurrent networks,” in *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 1173–1184, 2017.

[39] A. Hefny, Z. Marinho, W. Sun, S. S. Srinivasa, and G. Gordon, “Recurrent predictive state policy networks,” in *Proceedings of the International Conference on Machine Learning (ICML)*, vol. 5, pp. 3104–3119, 2018.

[40] “Predictive state representations for grounding human-robot communication,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 178–185, 2010.

[41] J. A. Stork, C. H. Ek, Y. Bekiroglu, and D. Kragic, “Learning Predictive State Representation for in-hand manipulation,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3207–3214, 2015.

[42] K. Fukumizu, L. Song, and A. Gretton, “Kernel Bayes’ rule: Bayesian inference with positive definite kernels,” *Journal of Machine Learning Research*, vol. 14, pp. 3753–3783, 2013.

[43] Y.-L. K. Samo and S. Roberts, “Generalized Spectral Kernels,” tech. rep., University of Oxford, 2015. arXiv:1506.02236.

[44] S. Remes, M. Heinonen, and S. Kaski, “Non-stationary spectral kernels,” in *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 4643–4652, 2017.

[45] J. F. Ton, S. Flaxman, D. Sejdinovic, and S. Bhatt, “Spatial mapping with Gaussian processes and nonstationary Fourier features,” *Spatial Statistics*, vol. 28, pp. 59–78, 2018.

[46] H. Xue, Z. F. Wu, and W. X. Sun, “Deep spectral kernel learning,” in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 4019–4025, 2019.

[47] A. Mohamed, K. Qian, M. Elhoseiny, and C. Claudel, "Social-STGCNN: A Social Spatio-Temporal Graph Convolutional Neural Network for Human Trajectory Prediction," in *Proceedings of the IEEE/CVF Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14412–14420, 2020.

[48] J. Oh, S. Singh, and H. Lee, "Value prediction network," in *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 6119–6129, 2017.

[49] J. Van Den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Proceedings of the International Symposium of Robotic Research*, pp. 3–19, 2011.

[50] I. Osband, C. Blundell, A. Pritzel, and B. Van Roy, "Deep exploration via bootstrapped DQN," in *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 4033–4041, 2016.

[51] J. Zhuang, T. Tang, Y. Ding, S. Tatikonda, N. Dvornek, X. Papademetris, and J. S. Duncan, "AdaBelief optimizer: Adapting stepsizes by the belief in observed gradients," in *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 18795–18806, 2020.

[52] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to Autonomous Mobile Robots*, pp. 270–271. The MIT Press, 2nd ed., 2011.

[53] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Scaled-YOLOv4: Scaling cross stage partial network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13029–13038, 2021.

[54] D. Jia, A. Hermans, and B. Leibe, "DR-SPAAM: A Spatial-Attention and autoregressive model for person detection in 2D range data," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10270–10277, 2020.

[55] D.-K. Kim and M. R. Walter, "Satellite image-based localization via learned embeddings," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2073–2080, 2017.

[56] W. Lu, Y. Zhou, G. Wan, S. Hou, and S. Song, "L3-Net: Towards learning based LiDAR localization for autonomous driving," in *Proceedings of the IEEE/CVF*

Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6382–6391, 2019.