Local Path Planning: Dynamic Window Approach With Q-Learning Considering Congestion Environments for Mobile Robot

Kobayashi, Masato
Zushi, Hiroka
Nakamura, Tomoaki
Motoi, Naoki

(Citation)
IEEE Access,11:96733-96742

(Issue Date)
2023-09-01

(Resource Type)
journal article

(Version)
Version of Record

IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 14 July 2023, accepted 28 August 2023, date of publication 1 September 2023, date of current version 11 September 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3311023

RESEARCH ARTICLE

Local Path Planning: Dynamic Window Approach With Q-Learning Considering Congestion Environments for Mobile Robot

MASATO KOBAYASHI[1], (Member, IEEE), HIROKA ZUSHI[2], TOMOAKI NAKAMURA[2], AND NAOKI MOTOI[2], (Member, IEEE)

[1]Cybermedia Center, Osaka University, Toyonaka 560-0043, Japan
[2]Graduate School of Maritime Science, Kobe University, Kobe 658-0022, Japan

Corresponding authors: Masato Kobayashi (kobayashi.masato.cmc@osaka-u.ac.jp) and Naoki Motoi (motoi@maritime.kobe-u.ac.jp)

This work was supported in part by the Hagiwara Foundation of Japan.

ABSTRACT In recent years, autonomous mobile robots have significantly increased in prevalence due to their ability to augment and diversify the workforce. One critical aspect of their operation is effective local path planning, which considers dynamic constraints. In this context, the Dynamic Window Approach (DWA) has been widely recognized as a robust local path planning. DWA produces a set of path candidates derived from velocity space subject to dynamic constraints. An optimal path is selected from path candidates through an evaluation function guided by fixed weight coefficients. However, fixed weight coefficients are typically designed for a specific environmental context. Consequently, changes in environmental conditions such as congestion levels, road width, and obstacle density could potentially lead the evaluation function to select inefficient paths or even result in collisions. To overcome this challenge, this paper proposes the dynamic weight coefficients based on Q-learning for DWA (DQDWA). The proposed method uses a pre-learned Q-table that comprises robot states, environmental conditions, and actions of weight coefficients. DQDWA can use the pre-learned Q-table to dynamically select optimal paths and weight coefficients that better adapt to varying environmental conditions. The performance of DQDWA was validated through extensive simulations and real experiments to confirm its ability to enhance the effectiveness of local path planning.

INDEX TERMS Path planning, motion planning, collision avoidance, mobile robot, dynamic window approach.

I. INTRODUCTION

As the world grapples with declining birthrates and an aging population, these demographic shifts are increasingly viewed as serious issues [1], [2]. To mitigate the resultant strain on the workforce, there has been a growing emphasis on implementing autonomous mobile robots in various contexts such as warehouses [3] and factories [4]. These robots need to navigate different environments autonomously. Therefore, the robot requires the integration of a diverse set of technologies, including localization [5], mapping [6], perception [7], and path planning [8].

The associate editor coordinating the review of this manuscript and approving it for publication was Zheng Chen.

This paper primarily explores the realm of path planning. The path planning technology is divided into global and local path planning [9]. Global path planning generates a path from the starting point to the destination based on a pre-existing map [10], [11]. However, it fails to account for unknown or unexpected obstacles in real-world environments. Therefore, in dynamic human workspaces, robots should reach their destinations and avoid obstacles autonomously and adaptively [12], [13]. Consequently, the focus has shifted towards local path planning, which factors in the dynamic obstacles not accounted for on the pre-established maps.

With this research, we delve into local path planning considering those obstacles not included on pre-built maps [14], [15], [16], [17], [18], [19]. While dynamic obstacles are certainly a consideration [20], [21], [22], [23], this paper

focuses on static environments like factories and warehouses. The Dynamic Window Approach (DWA), which accounts for dynamic constraints, has emerged as a prevalent local path planning method [24]. Despite numerous reported improvements to DWA [15], [25], [26], its limitations persist. In particular, DWA's fixed weight coefficients, which determine the optimal path based on factors such as goal position, obstacle distance, and robot velocity, fail to adapt to changes in environmental situations. This can lead to the selection of inefficient paths or even collisions, especially in confined or crowded spaces like factories and warehouses.

To address these issues, researches on dynamic weight coefficients for DWA have been carried out. Abubakr et al. and Hong et al. adjusted weight coefficients with fuzzy logic [28], [29]. These approaches dynamically adjusted weight coefficients using fuzzy logic to analyze goal positions and obstacles. Chang et al. proposed using Q-learning to dynamically adjust the weight coefficients of DWA [30]. Q-learning is a method of reinforcement learning. It doesn't require prior knowledge of the environment, making it suitable for robot path planning. Additionally, it offers a low cost for learning.

Considering these advantages, this paper focuses on the Q-learning method for adjusting weight coefficients of DWA. While the conventional method [30] adjusts weight coefficients based on goal information, velocities, and obstacles, it doesn't account for the spatial area and congestion rates of the environment. The conventional method leads to the selection of inefficient paths or even collisions, depending on the specific situation. To remedy the issue, this paper proposes a dynamic weight coefficient adjustment approach based on Q-learning for DWA that accounts for environmental situations (DQDWA). DQDWA considers environmental conditions such as goal distance, goal direction, velocity, visible area, and congestion. DQDWA can dynamically adjust the weight coefficients of the evaluation function based on these environmental conditions. Extensive simulations and experiments have been carried out to demonstrate the effectiveness and advantages of DQDWA in real-world scenarios.

The main contributions of this paper are threefold:

- This paper proposes DQDWA. DQDWA can dynamically adjust the weight coefficients of the evaluation function based on these environmental conditions.
- DQDWA incorporates the concept of context-awareness, where weight coefficients are not static but dynamically adjusted according to the area of spaces and congestion levels. This approach enhances the adaptability and performance of autonomous robots in varied situations.
- The effectiveness of DQDWA has been validated through extensive simulations and real-world experiments. The results demonstrate that the proposed method outperforms traditional DWA in terms of efficiency and safety.

This paper is organized into eight sections including the current section. Sections II, III, and IV provide a comprehensive overview of the coordinate system, the Dynamic Window Approach (DWA), and Q-learning, respectively. Section V proposes Dynamic Weight Coefficients based on Q-learning for DWA (DQDWA). Sections VI and VII show the results from our simulations and real-world experiments to highlight the effectiveness and utility of DQDWA. Finally, section VIII provides conclusions.

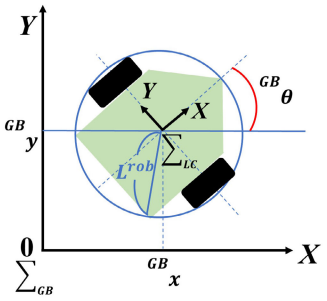


FIGURE 1. Modeling of robot.

II. COORDINATE SYSTEM

Fig. 1 illustrates the coordinate system for the robot utilized in this study. This paper defines two coordinate systems: the local coordinate system Σ_{LC}, and the global coordinate system Σ_{GB}. The quantities measured in the global coordinate system are expressed with the superscript ${}^{GB}\bigcirc$. Variables belonging to the local coordinate system do not carry a superscript. The origin in the global coordinate system is situated at the initial position of the robot. The origin in local coordinate system is positioned at the midpoint between the robot's wheels. As shown in Fig. 1, $({}^{GB}x, {}^{GB}y)$ and ${}^{GB}\theta$ represent the position and angle of the robot in the global coordinate system, respectively. L^{rob} denotes the radius of the robot.

III. DYNAMIC WINDOW APPROACH (DWA)

A. OVERVIEW OF DWA

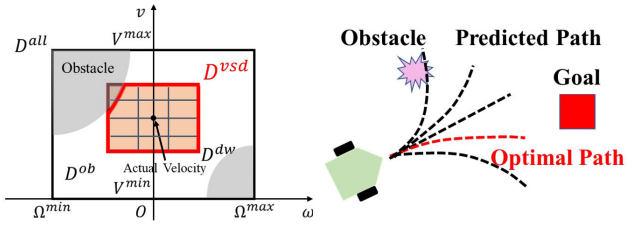
The Dynamic Window Approach (DWA) is a commonly used method in local path planning [24]. Initially, the velocity space with dynamic constraints (VSD) is determined based on the robot's current velocities. Subsequently, at each time step, an optimal path is selected from the VSD using an evaluation function. This optimal path selection is dependent on the weight coefficients of the evaluation function. Details about the velocity space and the optimal path selection are further elaborated in Sections III-B and III-C, respectively.

B. VELOCITY SPACE

DWA generates a velocity space with dynamic constraint, denoted as D^{vsd}, using translational and angular velocities as illustrated in Fig. 2 (a). The velocity space D^{vsd} is defined as follows.

$$D^{vsd} = D^{all} \cap D^{dw} \cap D^{obs} \tag{1}$$

where D^{all} represents the range of maximum and minimum velocities determined by the robot's specifications. D^{dw} known as the dynamic window, defines the range of



(a) Velocity Space (b) Optimal Path

FIGURE 2. Dynamic window aprroach.

velocities that the robot can achieve at the next time step. D^{obs} consists of velocities that enable the robot to stop before colliding with an obstacle.

C. OPTICAL PATH

The velocity space D^{vsd} is discretized by equally dividing the range of the translational and angular velocities. This results in pairs of translational and angular velocities within the velocity space D^{vsd}, which serve as the velocity candidates. As shown in Fig. 2 (b), DWA generates predicted paths for each velocity candidate under the assumption of constant velocity motion.

Path candidates are evaluated using the following evaluation function J.

$$J = W^{gol} \cdot c^{gol} + W^{vel} \cdot c^{vel} + W^{obs} \cdot c^{obs} \tag{2}$$

where W^{gol}, W^{vel}, and W^{obs} represent the weight coefficients associated with the goal, velocity, and obstacles, respectively. c^{gol} indicates the distance between the predicted robot position and the goal position. c^{vel} corresponds to the current translational velocity. c^{obs} represents the shortest distance from the predicted robot position on the path to the obstacle. The optimal path is then determined by maximizing the evaluation function J. More details on DWA can be found in [27].

IV. Q-LEARNING

To dynamically adjust the weight coefficients of the evaluation function in DWA, this study incorporates Q-learning [31], a type of reinforcement learning method. Fig. 3 outlines the concept of Q-learning, which updates a Q-table that stores the Q-values for each action in each state. The Q-table is a $m \times n$ matrix, where m and n correspond to the numbers of states and actions, respectively. The formula to update the Q-value is defined as follows.

$$Q(s, a) = (1 - \alpha)Q(s, a) + \alpha[R(s, a) + \gamma Q(s', a)] \tag{3}$$

where α and γ represent the learning rate and discount rate, respectively. $R(s, a)$ and $Q(s', a)$ denote the reward for the agent and the maximum Q-value in the next state, respectively. The training process for Q-learning involves four steps, as outlined in Fig. 3:

Train1: In the current state s, the agent chooses action a using the ϵ-greedy method with the Q-table.

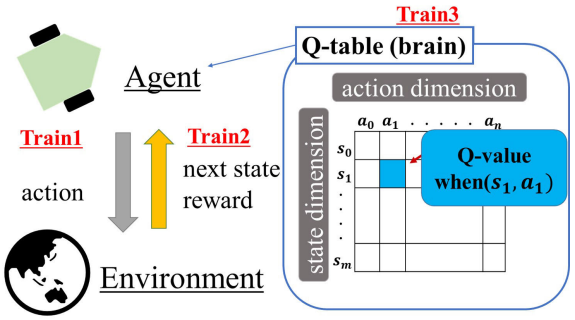


FIGURE 3. Q-learning.

Train2: The agent receives the next state and reward R from the environment.

Train3: The Q-value in the Q-table is updated using (3).

Train1-Train3 are repeated until the Q-table converges to a threshold value.

The ϵ-greedy method [32] is utilized for action selection. With probability ϵ, the action is chosen randomly, while with probability $1 - \epsilon$, the action with the highest expected reward is chosen. More details about Q-learning can be found in [31].

V. PROPOSED METHOD (DQDWA)

A. OVERVIEW OF DQDWA

This section proposes Dynamic Weight Coefficients based on Q-learning for DWA approach considering environmental situations (DQDWA). While the conventional method [30] adjusts weight coefficients based on certain parameters, it does not consider visible area and congestion rate as environmental factors. Therefore, the conventional method may lead to inefficient path selection or even collisions depending on the circumstances. To address this limitation, the proposed method includes visible area and congestion as key factors in defining environmental situations. Fig. 4 provides a overview of DQDWA which consists of four steps:

Step1: Determine the robot state s_1-s_5 based on environmental information measured by the distance sensor. The definitions of s_1-s_5 are detailed in Section V-B.

Step2: The trained Q-table chooses the appropriate combination of weight coefficients for the given state. The definition of the action dimension and the Q-learning process are detailed in Sections IV, V-C, and V-D.

Step3: The chosen weight coefficients are applied to the evaluation function of DWA. DWA is elaborated on in Section III.

Step4: The robot moves according to the translational and angular velocity that maximizes the evaluation function.

B. DEFINITION OF STATE DIMENSION

Fig. 5 provides a visual representation of state dimensions s_1-s_5. s_1-s_5 indicate states related to the goal distance, goal direc-

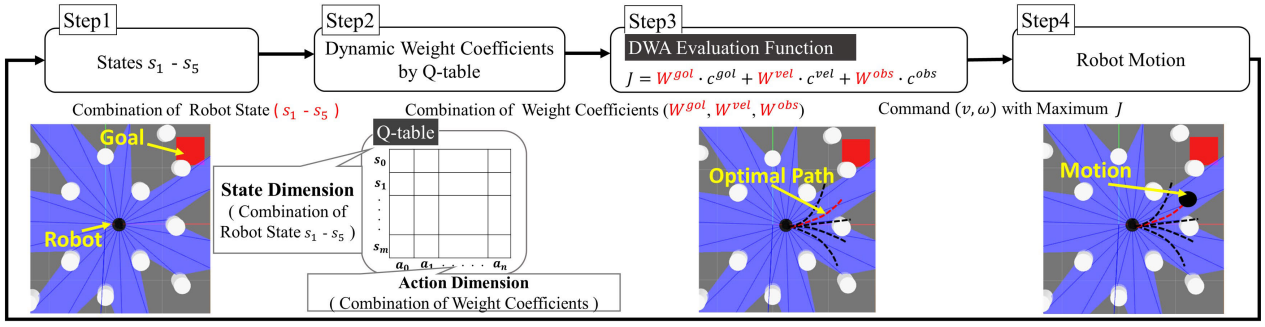
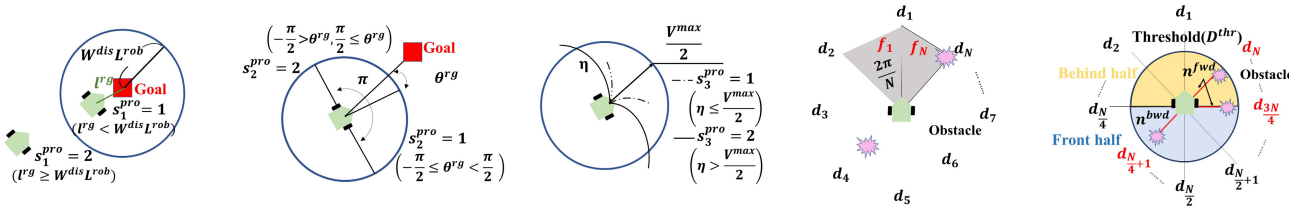


FIGURE 4. Overview of DQDWA.



(a) s_1 (b) s_2 (c) s_3 (d) s_4 (e) s_5

FIGURE 5. Image of state in proposed method.

tion, traveled distance, visible area, and congestion. The state vector $\boldsymbol{s}$ is defined as follows.

$$\boldsymbol{s} = \begin{bmatrix} s_1 & s_2 & s_3 & s_4 & s_5 \end{bmatrix}^{\mathsf{T}} \tag{4}$$

where s_1, s_2, s_3, and s_4 has two possible patterns, while s_5 has four. Thus, the total size of the state dimension is 64. The detailed descriptions of s_1-s_5 are presented below.

1) DEFINITION OF STATE DIMENSION s_1 (GOAL DISTANCE)

s_1 represents the state associated with the distance between the robot's position and the goal position. s_1 is defined as follows.

$$s_1 = \begin{cases} 1 & \text{if} \quad l^{rg} < W^{dis} L^{rob} \\ 2 & \text{otherwise} \end{cases} \tag{5}$$

where W^{dis} is the weight coefficient for the goal distance and l^{rg} is the distance between the robot and the goal.

2) DEFINITION OF STATE DIMENSION s_2 (GOAL DIRECTION)

s_2 represents the state indicating the angular difference between the robot's direction and the goal's direction. s_2 is defined as follows.

$$s_2 = \begin{cases} 1 & \text{if} \quad \theta^{rg} \in [-\frac{\pi}{2}, \frac{\pi}{2}) \\ 2 & \text{otherwise} \end{cases} \tag{6}$$

where θ^{rg} is the angle between the robot and the goal.

3) DEFINITION OF STATE DIMENSION s_3 (TRAVELLED DISTANCE)

s_3 is the state related to the distance that the robot will travel from its current position after one second. The travelled distance η is calculated as follows.

$$\eta = \begin{cases} \dfrac{2v}{\omega} & \text{if} \quad |\omega| > \pi \\ v & \text{else if} \quad \omega = 0 \\ \dfrac{2v}{\omega} \sin(\dfrac{\omega}{2}) & \text{otherwise} \end{cases} \tag{7}$$

s_3 is defined as follows.

$$s_3 = \begin{cases} 1 & \text{if} \quad |\eta| \leq \dfrac{V^{max}}{2} \\ 2 & \text{otherwise} \end{cases} \tag{8}$$

where V^{max} is the maximum translational velocity of the robot.

4) DEFINITIONOF STATE DIMENSION s_4 (VISIBLE AREA)

s_4 is the state that quantifies the visible area around the robot. The divided area f_i for the state s_4 is defined as follows.

$$f_i = \frac{1}{2} d_i d_{i+1} \sin(\frac{2\pi}{N}) \tag{9}$$

where d_i is the i-th distance data measured by the distance sensor, and N is the total number of distance data points. Distance data is obtained in $\dfrac{2\pi}{N}$ radian increments in a counter-clockwise direction. The total divided area f^{all} is calculated as follows.

$$f^{all} = \sum_{i=1}^{N} f_i \tag{10}$$

s_4 is defined as follows.

$$s_4 = \begin{cases} 1 & \text{if} \quad f^{all} \leq W^{are} F^{max} \\ 2 & \text{otherwise} \end{cases} \tag{11}$$

where W^{are} is the weight coefficient of the area, and F^{max} is the maximum possible sum of the divided areas.

5) DEFINITION OF STATE DIMENSION s_5 (CONGESTION)

s_5 is the state associated with congestion. s_5 is defined based on the number of obstacles surrounding the robot.

$$s_5 = \begin{cases} 1 & \text{if} \quad n^{fwd} > \dfrac{N}{4} \\ 2 & \text{else if} \quad n^{bwd} > \dfrac{N}{4} \\ 3 & \text{else if} \quad n^{all} > \dfrac{N}{4} \\ 4 & \text{otherwise} \end{cases} \tag{12}$$

where n^{fwd} and n^{bwd} are the number of sensor data points within a threshold distance D^{thr} in the front and rear halves of the robot, respectively. n^{all} is defined as follows.

$$n^{all} = n^{fwd} + n^{bwd} \tag{13}$$

C. DEFINITION OF REWARD

To adjust the weight coefficients of the evaluation function with Q-learning, the reward R is defined as follows.

$$R = R_1 + R_2 + R_3 \tag{14}$$

Note that the initial value of R is set to 0. R_1 is a reward related to the result; goal or collision.

$$R_1 = \begin{cases} 5000 & \text{if} \quad \text{reach goal} \\ -200 & \text{else if} \quad \text{collide obstacle} \\ -2 & \text{otherwise} \end{cases} \tag{15}$$

R_2 is a reward related to distance from the goal position.

$$R_2 = \begin{cases} 0 & \text{if} \quad \text{reach goal or collide obstacle} \\ 10 & \text{if} \quad \text{get close to goal position} \\ -10 & \text{else if} \quad \text{farther from goal position} \end{cases} \tag{16}$$

R_3 is a reward related to distance from the obstacle.

$$R_3 = \begin{cases} 0 & \text{if} \quad \text{reach goal or collide obstacle} \\ -5 & \text{if} \quad \text{approach obstacle} \\ 5 & \text{else if} \quad \text{go away from obstacle} \end{cases} \tag{17}$$

D. DEFINITION OF ACTION DIMENSION

The weight coefficients for position, velocity, and obstacles are each selected from the set {1,2,3}. We omit the sets {2,2,2} and {3,3,3} as they are equivalent to the set {1,1,1}. As a result, 25 unique combinations are obtained, which form the action dimension.

The Q-table comprises the actions and the states of the robot in various environmental situations. By utilizing the learned Q-table, DQDWA selects the optimal path using dynamic weight coefficients considering these environmental situations.

TABLE 1. Control parameters.

V^{max}	Maximum Translational Velocity	0.22 $[m/s]$
V^{min}	Minimize Translational Velocity	-0.1 $[m/s]$
Ω^{max}	Maximum Angular Velocity	1.5 $[rad/s]$
Ω^{min}	Minimize Angular Velocity	-1.5 $[rad/s]$
$\dot{V}^{max}$	Maximum Translational Acceleration	2.5 $[m/s^2]$
$\dot{\Omega}^{max}$	Maximum Angular Acceleration	3.2 $[rad/s^2]$
T^{max}	Maximum Predicted Time	4.0 $[s]$
ΔT	Time Step	0.2 $[s]$
D^{thr}	Distance Threshold on s_5	1.5$[m]$
L^{rob}	Robot Radius	0.13$[m]$
W^{are}	Weight Coefficient of Visible Area	0.3
W^{dis}	Weight coefficient of Goal Distance	3
F^{max}	Max Visible Area in s_4	38$[m^2]$
N	Number of Sensor Data	24

TABLE 2. Goal positions in each environment of learning phase.

Environment	$({}^{GB}x^{gol}, {}^{GB}y^{gol})$
$Env.\ 1$	($[-1.2, 1.2]$, $[-1.2, 1.2]$) (resolution: 0.1)
$Env.\ 2$	($[-1.2, 1.2]$, $[-1.2, 1.2]$) (resolution: 0.1)
$Env.\ 3$	(1.3,1.6), (-1.3,1.5), (0.5,-1.5), (-0.5,-1.5)
$Env.\ 4$	(0.0,4.0), (-0.5,3.0), (0.0,3.0), (-1.5,2.5), (-2.0,0.5), (0.5,-3.0), (1.0,-3.0), (1.5,-2.5)
$Env.\ 5$	(0.0,8.0)

VI. SIMULATION

A. SIMULATION SETUP

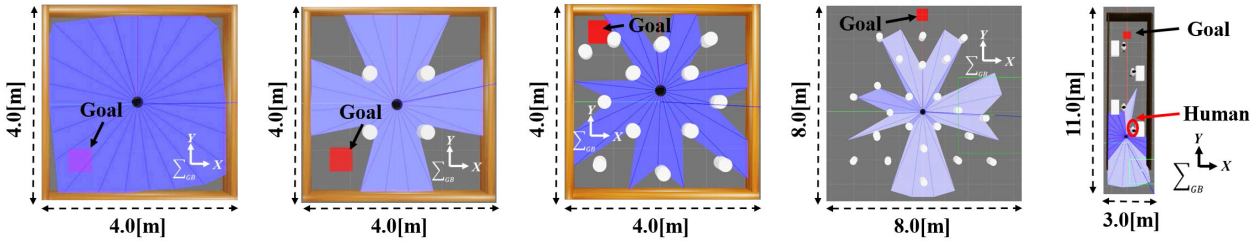
The simulation system was implemented using the Robot Operating System (ROS) and Gazebo. In this simulation, we evaluated five patterns of DWA weight coefficients: DWA I, DWA II, DWA III, Conventional DWA with Q-learning (CDQ) [30], and DQDWA. The constant weight coefficients $W^{gol}, W^{vel}, W^{obs}$ for DWA I, DWA II, and DWA III were set as {1,1,2}, {1,2,1}, and {2,1,1}, respectively. Table 1 displays the simulation parameters.

B. PRE-TRAIN OF Q-TABLE

Fig. 6 (a)-(e) depict the environments utilized during the learning process. Env. and goal positions were randomly selected at the start of each trial as outlined in Table2. ${}^{GB}x^{gol}$ and ${}^{GB}y^{gol}$ denote the X and Y coordinates of the goal positions, respectively. The notation ($[-1.2, 1.2]$, $[-1.2, 1.2]$) means that ${}^{GB}x^{gol}$ and ${}^{GB}y^{gol}$ are randomly selected from the range of $[-1.2, 1.2]$. As illustrated in Fig. 6 (a)-(c), Env. 1-3 were designed to examine the differences in robot behavior due to crowding within restricted spaces. Fig. 6 (d) shows Env. 4, which was established to investigate robot behavior in a spacious area filled with numerous obstacles. In Fig. 6 (e), Env. 5 was designed to evaluate robot behavior amidst obstacles and humans. All of these environments were thoughtfully designed with real-world scenarios in mind, specifically warehouse and factory settings. The learning process was continued until the Q-table had been updated 30,000 times.

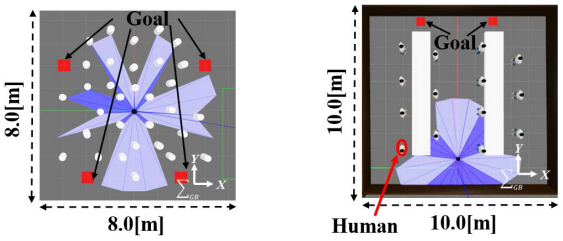
C. SIMULATION ENVIRONMENT

In this simulation, the following two types of simulations were conducted.



(a) Env. 1 (b) Env. 2 (c) Env. 3 (d) Env. 4 (e) Env. 5

FIGURE 6. Image of Learning Environment. Blue area means the visible sensor area.



(a) Env. 6 (b) Env. 7

FIGURE 7. Image of Non-Learning Environment. Blue area means the visible sensor area.

- Case S1: Conducted a single simulation in each environment (Env. 1-5).
- Case S2: Performed 30 simulations in unfamiliar environments (Env. 6,7).

The starting positions in Cases S1 and S2 were set to $({}^{GB}x^{sta}, {}^{GB}y^{sta}) = (0.0, 0.0)$. In Case S1, the goal positions of Env. 1-5, denoted as $({}^{GB}x^{gol}, {}^{GB}y^{gol})$, were established as $(-1.2, -1.2)$, $(-1.2, -1.2)$, $(-1.3, 1.5)$, $(0.0, 4.0)$, and $(0.0, 8.0)$, respectively. For Case S2, goal positions were randomly selected from the red-filled areas as shown in Fig. 7.

Env. 6 was designed with a higher density of obstacles compared to Env. 4. This was intended to test the robot's ability to deal with unforeseen obstacles that were not encountered during the learning phase. Env. 7 was designed to simulate a manufacturing plant. In this environment, the robot had to recognize and avoid not only static obstacles but also dynamic obstacles, such as humans, while navigating toward its goal.

D. SIMULATION RESULTS

1) CASE S1

Tables 3 - 4 present the results for Case S1. The abbreviations TL and PD denote the trajectory length and the movement posture displacement, respectively. Table 4 indicates the number of collisions, along with the average time, trajectory length (TL), and posture displacement (PD). Figs. 8-12 depict the trajectories in each environment.

For DWA I-III, while they delivered satisfactory results in some environments, there were instances where the robot collided with obstacles. Moreover, the robot often required a long duration to reach the goal position. The simulation results for DWA I-III varied depending on the environmental situation, as these methods utilize fixed weight coefficients.

In the case of CDQ, the simulation results for Env. 1-3 were better than those for DWA I-III, since CDQ selects weight coefficients considering the environmental situation. However, the results for Env. 4 and 5 were nearly identical to those for DWA I-III. This is because CDQ does not define environmental situations based on visible space size or obstacle count. Therefore, optimal weight coefficients were not chosen in narrow or crowded spaces.

In DQDWA, the robot successfully reached the goal position in the shortest time and with the smallest TL and PD. This is because DQDWA takes into account both space size and congestion, enabling the selection of optimal weight coefficients tailored to each environment. DQDWA allows for

TABLE 3. Simulation results in case S1 (1time in each environment).

Environment	Method	Success Rate [%]	Time [sec]	TL [m]	PD [rad]
Env. 1 (1 Time)	DWA I	100	19.9	1.75	3.88
	DWA II	100	14.9	1.98	2.95
	DWA III	100	16.4	2.06	10.4
	CDQ	100	14.5	1.91	3.26
	DQDWA	100	14.3	1.98	3.35
Env. 2 (1 Time)	DWA I	100	15.4	2.02	9.47
	DWA II	100	15.3	2.19	9.38
	DWA III	100	18.2	2.25	4.19
	CDQ	100	16.0	2.41	17.1
	DQDWA	100	14.0	2.04	3.12
Env. 3 (1 Time)	DWA I	100	21.5	2.32	4.33
	DWA II	100	15.1	2.14	1.60
	DWA III	0	-	-	-
	CDQ	100	17.5	2.23	2.64
	DQDWA	100	15.5	2.22	2.72
Env. 4 (1 Time)	DWA I	100	26.6	4.14	3.35
	DWA II	100	24.3	3.96	2.32
	DWA III	0	-	-	-
	CDQ	100	33.8	5.55	14.6
	DQDWA	100	23.8	3.96	2.04
Env. 5 (1 Time)	DWA I	100	46.7	7.84	5.33
	DWA II	100	44.0	7.65	4.34
	DWA III	100	43.6	7.55	3.72
	CDQ	100	44.8	7.83	5.18
	DQDWA	100	42.8	7.55	3.83

TABLE 4. Simulation results in case S1 (average in each Env.).

Method	Collision [-]	Time Average [sec]	TL Average [m]	PD Average [rad]
DWA I	0	26.0	3.61	5.27
DWA II	0	22.7	3.58	4.12
DWA III	2	26.1	3.95	6.10
CDQ	0	25.3	3.99	8.56
DQDWA	0	22.1	3.55	3.01

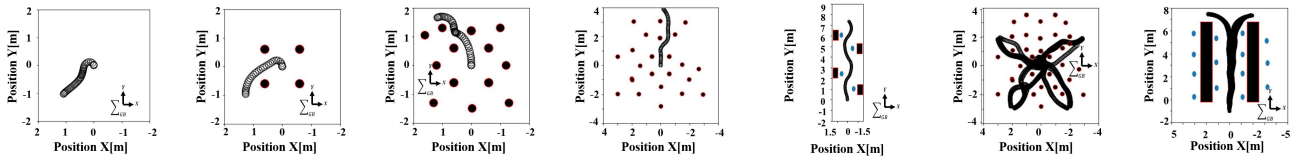


FIGURE 8. Trajectories of DWA I ($\{W^{gol}, W^{vel}, W^{obs}\} = \{1,1,2\}$).

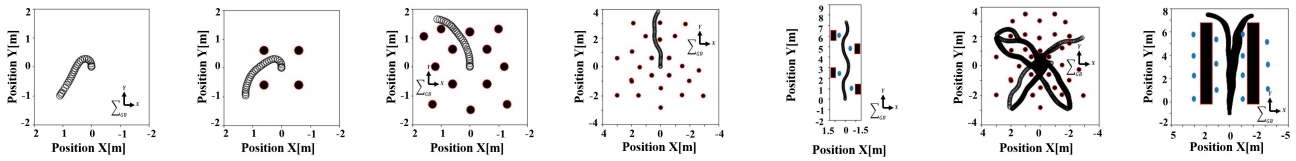


FIGURE 9. Trajectories of DWA II ($\{W^{gol}, W^{vel}, W^{obs}\} = \{1,2,1\}$).

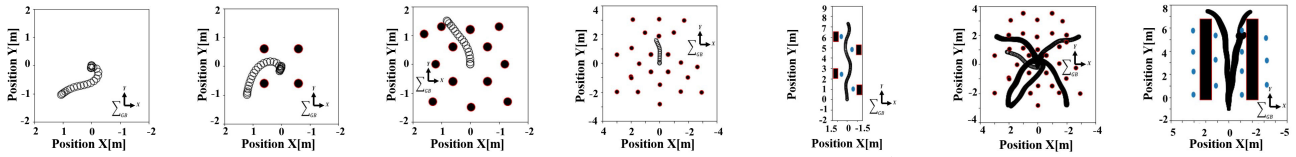


FIGURE 10. Trajectories of DWA III ($\{W^{gol}, W^{vel}, W^{obs}\} = \{2,1,1\}$).

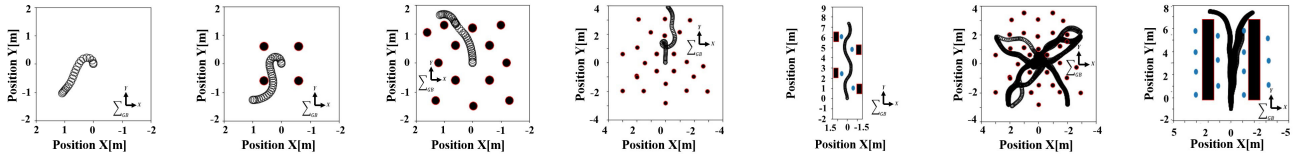


FIGURE 11. Trajectories of the conventional method (CDQ).

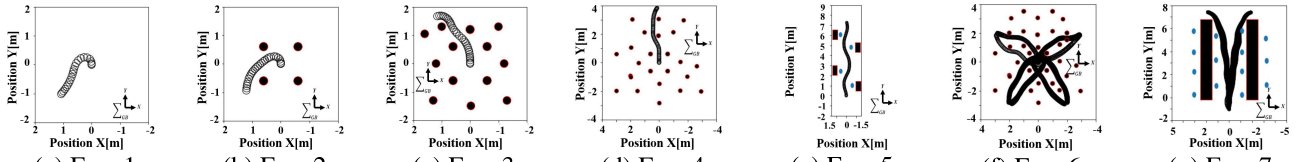


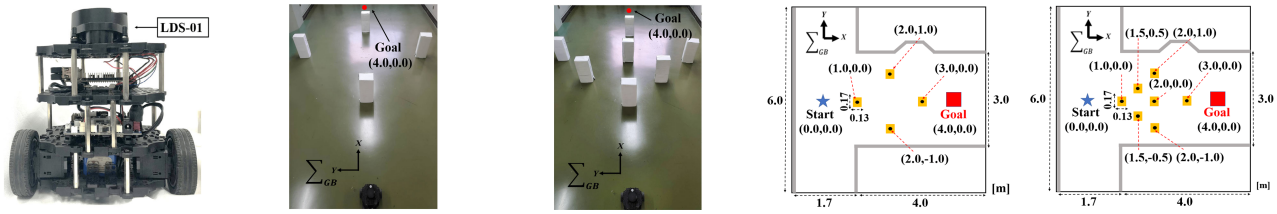
FIGURE 12. Trajectories of the proposed method (DQDWA).

more efficient routing while ensuring safety and preventing the robot from circling in one place.

2) CASE S2

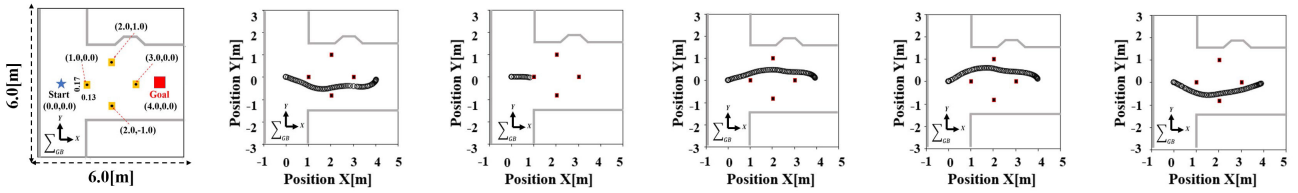
Table 5 presents the results for Case S2, while Figs. 8-12 (f)-(g) illustrate the corresponding trajectories. In this simulation, the goal position was randomly selected at the start of each trial. For Env. 6, the goal position was selected from four points with $({}^{GB}x^{gol}, {}^{GB}y^{gol})$ being (2.0, −3.0), (3.0, 2.0), (−2.0, −3.0), and (−3.0, 2.0). In Env. 7, the goal position was selected from two points, with $({}^{GB}x^{gol}, {}^{GB}y^{gol})$ being (8.5, 2.0) and (8.5, −2.0).

In the case of DWA I, while the success rate was high, the robot took a significantly longer time to reach the goal position. DWA II and DWA III achieved smaller values for time, trajectory length (TL), and posture displacement (PD), but their success rates were comparatively lower. This is because these approaches prioritized high translational velocity and goal distance over obstacle avoidance, leading to more collisions.



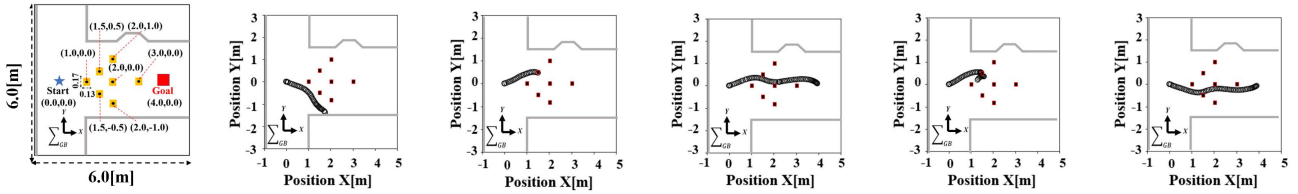
(a)Turtlebot3 (b) Case E1 (c) Case E2 (d) Image of Case E1 (e) Image of Case E2

FIGURE 13. Experimental Set-up.



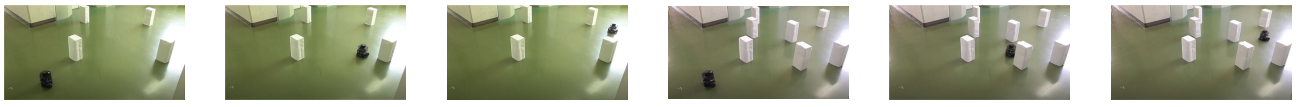
(a) Case E1 (b) DWA I (c) DWA II (d) DWA III (e) CDQ (f) DQDWA

FIGURE 14. Trajectories in Case E1.



(a) Case E2 (b) DWA I (c) DWA II (d) DWA III (e) CDQ (f) DQDWA

FIGURE 15. Trajectories in Case E2.



(a) E1 (2 [sec]) (b) E1 (10 [sec]) (c) E1 (18 [sec]) (d) E2 (2 [sec]) (e) E2 (9 [sec]) (f) E2 (14 [sec])

FIGURE 16. Snapshots in DQDWA.

TABLE 5. Case S2 results (30 times in each environment).

Environment	Method	Success Rate [%]	Time [sec]	TL [m]	PD [rad]
Env. 6 (30 Times)	DWA I	93	29.1	4.27	11.76
	DWA II	80	26.8	4.25	8.76
	DWA III	70	27.5	4.03	7.48
	CDQ	83	31.7	4.89	17.46
	DQDWA	93	27.3	4.10	7.83
Env. 7 (30 Times)	DWA I	100	62.9	9.29	4.68
	DWA II	97	58.4	9.09	4.07
	DWA III	67	59.1	9.01	3.66
	CDQ	97	61.2	9.19	4.33
	DQDWA	100	58.9	8.83	3.90

CDQ yielded a lower success rate than DQDWA. Additionally, the averages of time, TL, and PD were the largest in Env. 6 and the second largest in Env. 7. This indicates that CDQ didn't select efficient paths in environments that were not encountered during the learning phase.

In contrast, DQDWA achieved the highest success rate. Furthermore, it reached the goal position in a time span comparable to DWA II, which prioritizes translational velocity, and with TL and PD as small as DWA III, which prioritizes the goal distance. Therefore, DQDWA selected efficient paths while maintaining safety in unlearned environments.

The effectiveness of the proposed method, DQDWA, was thus confirmed through the simulation results for both Case S1 and Case S2.

VII. EXPERIMENT

A. EXPERIMENT SETUP

The experiment was carried out with ROS and Turtlebot3. Fig. 13 (a) shows an overview of Turtlebot3. Turtlebot3 is equipped with a distance sensor (LDS-01). The distance sensor measured environmental information. Fig. 13 (b)-(c)

TABLE 6. Experiment results.

Case	Method	Success Rate	Time [sec]	TL [m]	PD [rad]
$Case1$ (1 time)	DWA I	100	25.8	4.27	3.00
	DWA II	0	-	-	-
	DWA III	100	24.9	4.09	2.44
	CDQ	100	25.3	4.23	2.71
	DQDWA	100	24.2	4.09	1.76
$Case2$ (1 time)	DWA I	0	-	-	-
	DWA II	0	-	-	-
	DWA III	100	25.3	4.08	2.87
	CDQ	0	-	-	-
	DQDWA	100	23.6	4.03	2.55

show the experiment environments. Fig. 13 (d)-(e) show their images. For the experiments, we have two scenarios defined as follows.

- Case E1: This represents a simple environment with four obstacles.
- Case E2: This represents a crowded environment with seven obstacles.

The start position was $({}^{GB}x^{sta}, {}^{GB}y^{sta}) = (0.0, 0.0)$, and the goal position was $({}^{GB}x^{gol}, {}^{GB}y^{gol}) = (4.0, 0.0)$. The models and parameters used in the experiment were the same as the simulation; DWA I, DWA II, DWA III, CDQ, and DQDWA.

B. EXPERIMENT RESULTS

Table 6 presents the experimental results. Figs.14-15 illustrate the trajectories for each case, and Fig. 16 shows snapshots from DQDWA run.

In DWA I-III, collisions sometimes occurred. Even in cases where the goal was reached, these methods resulted in longer times, larger trajectory lengths (TL), and greater posture displacements (PD) compared to DQDWA. Their inability to adjust weight coefficients dynamically led to collisions and the selection of inefficient paths.

CDQ also resulted in a collision in Case E2. Moreover, its time, TL, and PD in Case E1 were larger than those of DWA II and DQDWA. These results suggest that CDQ was not able to select appropriate weight coefficients based on the environmental situations.

Conversely, DQDWA successfully reached the goal position and registered the shortest time, smallest TL, and least PD in both cases. DQDWA was capable of adjusting weight coefficients effectively in real-time. The effectiveness of the proposed method, therefore, was confirmed by the experimental results.

VIII. CONCLUSION

This paper introduced DQDWA, the dynamic weight coefficients based on Q-learning for DWA considering environmental situations. We focused on defining the state for Q-learning and included definitions for the area of space, taking into account congested areas. With DQDWA, the robot could select optimal paths by dynamic adjustments of weight coefficients. The effectiveness of the proposed method was validated through simulations and real-world experiments.

In the future, we aim to refine and improve DQDWA as follows.

- *Incorporating Moving Obstacles:* The current evaluations of DQDWA have been conducted in static environments. Future work will look into accommodating moving obstacles in the learning and experiment environments.
- *Experiments in Diverse Environments:* Our experiments have been performed with a single type of robot and sensor. We plan to evaluate DQDWA's performance across various environments and using different types of robots and sensors.
- *Exploring Alternative Learning Methods:* Presently, we utilize Q-learning as the sole learning method to adjust weight coefficients. Future efforts will investigate other learning methods for dynamic adjustment of these coefficients.

REFERENCES

[1] N. Yamamoto, "The strategic plan of the Japan pharmaceutical association, the aspect of a super aged society," *YAKUGAKU ZASSHI*, vol. 142, no. 9, pp. 951–963, 2022.

[2] C. Nam, S. Lee, J. Lee, S. H. Cheong, D. H. Kim, C. Kim, I. Kim, and S.-K. Park, "A software architecture for service robots manipulating objects in human environments," *IEEE Access*, vol. 8, pp. 117900–117920, 2020.

[3] H. A. M. Tran, H. Q. T. Ngo, T. P. Nguyen, and H. Nguyen, "Develop of AGV platform to support the arrangement of cargo in storehouse," in *Proc. 24th Int. Conf. Autom. Comput. (ICAC)*, Sep. 2018, pp. 1–5.

[4] R. Kaneko, Y. Nakamura, R. Morita, and S. Ito, "Point cloud data map creation from factory design drawing for LiDAR localization of an autonomous mobile robot," *Artif Life Robot.*, vol. 28, pp. 1–9, Jan. 2022.

[5] S. Zhang, J. Shan, and Y. Liu, "Variational Bayesian estimator for mobile robot localization with unknown noise covariance," *IEEE/ASME Trans. Mechatronics*, vol. 27, no. 4, pp. 2185–2193, Aug. 2022.

[6] R. Liu, Y. He, C. Yuen, B. P. L. Lau, R. Ali, W. Fu, and Z. Cao, "Cost-effective mapping of mobile robot based on the fusion of UWB and short-range 2-D LiDAR," *IEEE/ASME Trans. Mechatronics*, vol. 27, no. 3, pp. 1321–1331, Jun. 2022.

[7] A. Bonci, P. D. C. Cheng, M. Indri, G. Nabissi, and F. Sibona, "Human–robot perception in industrial environments: A survey," *Sensors*, vol. 21, no. 521, pp. 1571–1579, 2021.

[8] J. Wang and M. Q. H. Meng, "Socially compliant path planning for robotic autonomous luggage trolley collection at airports," *Sensors*, vol. 19, no. 12, pp. 2759–2773, 2019.

[9] P. Marin-Plaza, A. Hussein, D. Martin, and A. D. L. Escalera, "Global and local path planning study in a ROS-based research platform for autonomous vehicles," *J. Adv. Transp.*, vol. 2018, pp. 1–10, Jan. 2018.

[10] F. Li, X. Fan, and Z. Hou, "A firefly algorithm with self-adaptive population size for global path planning of mobile robot," *IEEE Access*, vol. 8, pp. 168951–168964, 2020.

[11] Z. Liu, H. Liu, Z. Lu, and Q. Zeng, "A dynamic fusion pathfinding algorithm using Delaunay triangulation and improved A-star for mobile robots," *IEEE Access*, vol. 9, pp. 20602–20621, 2021.

[12] J. Fang, S. Qin, and H. Zhou, "Research on the obstacle avoidance system of restaurant delivery robot," in *Proc. Int. Conf. Artif. Intell. Comput. Inf. Technol. (AICIT)*, Sep. 2022, pp. 1–5.

[13] K.-S. Kim, D.-E. Kim, and J.-M. Lee, "Deep learning based on smooth driving for autonomous navigation," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatronics (AIM)*, Jul. 2018, pp. 616–621.

[14] Q. Jin, C. Tang, and W. Cai, "Research on dynamic path planning based on the fusion algorithm of improved ant colony optimization and rolling window method," *IEEE Access*, vol. 10, pp. 28322–28332, 2022.

[15] U. Patel, N. K. S. Kumar, A. J. Sathyamoorthy, and D. Manocha, "DWA-RL: Dynamically feasible deep reinforcement learning policy for robot navigation among mobile obstacles," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 6057–6063.

[16] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 2, Mar. 1985, pp. 500–505.

[17] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using the relative velocity paradigm," in *Proc. IEEE Int. Conf. Robot. Autom.*, Mar. 1998, pp. 760–772.
[18] T. M. Howard and A. Kelly, "Optimal rough terrain trajectory generation for wheeled mobile robots," *Int. J. Robot. Res.*, vol. 26, no. 2, pp. 141–166, Feb. 2007.
[19] U. Orozco-Rosas, K. Picos, J. J. Pantrigo, A. S. Montemayor, and A. Cuesta-Infante, "Mobile robot path planning using a QAPF learning algorithm for known and unknown environments," *IEEE Access*, vol. 10, pp. 84648–84663, 2022.
[20] M. Desai and A. Ghaffari, "CLF-CBF based quadratic programs for safe motion control of nonholonomic mobile robots in presence of moving obstacles," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatronics (AIM)*, Jul. 2022, pp. 16–21.
[21] Z. Meng, H. Sun, K. B. H. Teo, and M. H. Ang, "Active path clearing navigation through environment reconfiguration in presence of movable obstacles," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatronics (AIM)*, Jul. 2018, pp. 156–163.
[22] A. H. Qureshi, Z. Tahir, G. Tariq, and Y. Ayaz, "Re-planning using Delaunay triangulation for real time motion planning in complex dynamic environments," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatronics (AIM)*, Jul. 2018, pp. 905–911.
[23] X. Tao, N. Lang, H. Li, and D. Xu, "Path planning in uncertain environment with moving obstacles using warm start cross entropy," *IEEE/ASME Trans. Mechatronics*, vol. 27, no. 2, pp. 800–810, Apr. 2022.
[24] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robot. Autom. Mag.*, vol. 4, no. 1, pp. 23–33, Mar. 1997.
[25] X. Yan, R. Ding, Q. Luo, C. Ju, and D. Wu, "A dynamic path planning algorithm based on the improved DWA algorithm," in *Proc. Global Rel. Prognostics Health Manage.*, Oct. 2022, pp. 1–7.
[26] Z. Huo, S. Dai, M. Yuan, X. Chen, and X. Zhang, "A reinforcement learning based multiple strategy framework for tracking a moving target," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatronics (AIM)*, Jul. 2020, pp. 1292–1297.
[27] M. Kobayashi and N. Motoi, "Local path planning: Dynamic window approach with virtual manipulators considering dynamic obstacles," *IEEE Access*, vol. 10, pp. 17018–17029, 2022.
[28] O. A. Abubakr, M. A. Jaradat, and M. F. Abdel-Hafez, "Intelligent optimization of adaptive dynamic window approach for mobile robot motion control using fuzzy logic," *IEEE Access*, vol. 10, pp. 119368–119378, 2022.
[29] Z. Hong, S. Chun-Long, Z. Zi-Jun, A. Wei, Z. De-Qiang, and W. Jing-Jing, "A modified dynamic window approach to obstacle avoidance combined with fuzzy logic," in *Proc. 14th Int. Symp. Distrib. Comput. Appl. Bus. Eng. Sci. (DCABES)*, Aug. 2015, pp. 523–526.
[30] L. Chang, L. Shan, C. Jiang, and Y. Dai, "Reinforcement based mobile robot path planning with improved dynamic window approach in unknown environment," *Auto. Robots*, vol. 45, no. 1, pp. 51–76, Jan. 2021.
[31] C. J. C. H. Watkins, *Learning From Delayed Rewards*. London, U.K.: King's College, 1989.
[32] A. dos Santos Mignon and R. L. de Azevedo da Rocha, "An adaptive implementation of ε-greedy in reinforcement learning," *Proc. Comput. Sci.*, vol. 109, pp. 1146–1151, Jan. 2017.

MASATO KOBAYASHI (Member, IEEE) received the Bachelor of Maritime Sciences, Master of Maritime Sciences, and Doctor of Engineering degrees from Kobe University, Japan, in 2017, 2019, and 2022, respectively. From 2019 to 2021, he was an Engineer and a Researcher with the Technology Development Division, Seiko Epson Corporation, Japan. From 2021 to 2022, he was a Research Intern with OMRON SINIC X Corporation, Japan. Since 2022, he has been an Academic Researcher with Kobe University. He was with the Yutaka Matsuo Laboratory, The University of Tokyo, to research and develop robotics and AI. Since 2023, he has been with Osaka University, Japan, where he is currently an Assistant Professor. His current research interests include robotics, motion control, haptic, XR, AI, and mechatronics.

HIROKA ZUSHI received the Bachelor of Maritime Sciences degree from the Faculty of Maritime Sciences, Kobe University, Japan, in 2023. Her current research interests include machine learning, robotics, and motion control.

TOMOAKI NAKAMURA received the Bachelor of Maritime Sciences and Master of Maritime Sciences degrees from Kobe University, Japan, in 2021 and 2023, respectively. His current research interests include machine learning, robotics, and motion control.

NAOKI MOTOI (Member, IEEE) received the B.E. degree in system design engineering and the M.E. and Ph.D. degrees in integrated design engineering from Keio University, Japan, in 2005, 2007, and 2010, respectively. In 2007, he joined the Partner Robot Division, Toyota Motor Corporation, Japan. From 2011 to 2013, he was a Research Associate with Yokohama National University, Japan. Since 2014, he has been with Kobe University, Japan, where he is currently an Associate Professor. From 2019 to 2020, he was a Visiting Professor with the Automation and Control Institute (ACIN), TU Wien, Austria. His current research interests include robotics, motion control, and haptic.