2026 IEEE/SICE International Symposium on System Integration (SII)
January 11-14, 2026. Cancún, México

Mobile Robot Motion Planning Based on Time-Delay CNN with Open-Space Image Inputs for Multi-Obstacle Avoidance

Kenji Shibata[1] and Satoshi Hoshino[2]

Abstract— **During autonomous navigation, mobile robots often need to avoid obstacles in their path. To address this obstacle avoidance issue, we have proposed various motion planners based on deep neural networks. Focusing on obstacle avoidance, a mobile robot is typically required to move straight for a certain duration after avoiding an obstacle before reorienting itself toward the destination. However, in such cases, it is difficult for the robot to plan these different motions based on similar image inputs. To address this challenge, we propose a novel motion planner based on a Time-Delay CNN that utilizes visually distinct time-series image inputs. Through experiments, we demonstrate that the robot is able to plan appropriate avoidance motions as described above and navigate toward the destination in both simulation and real-world environments with multiple dynamic obstacles.**

I. INTRODUCTION

For autonomous navigation, a mobile robot plans a global path toward a destination from the current position localized in an environmental map. After that, the robot plans motions to move along the path. For an obstacle ahead, the robot avoids the collision by planning the path again. Therefore, path and motion planning are essential capabilities for autonomous navigation. In contrast to global path planning, motion planning can be executed locally. Moreover, the robot is able to avoid obstacles by planning the motion repeatedly. Therefore, we adopt the motion planning approach without path planning for autonomous navigation including obstacle avoidance. In this paper, motion planning is defined to determine the motion output, i.e., linear and angular velocities, for a sensor input. However, the robot is required to measure the accurate positions of obstacles in the map.

We have presented motion planners based on deep neural networks [1][2]. These motion planners were trained through imitation learning. For sensor inputs from a 2D LiDAR, the robot successfully avoided obstacles by directly determining the linear and angular velocities without environmental maps. In order for the robot to plan different avoidance motions depending on the obstacle, static or dynamic, an RGB-D camera was mounted as another sensor. Perception images generated from RGB images through obstacle detection were used as inputs to a motion planner [3] based on convolutional neural network, CNN [4], with long short-term memory, LSTM [5]. Furthermore, depth-difference images enabled the robot to plan different avoidance motions depending on the moving speed of the dynamic obstacle [6]. The planning policy was trained through imitation learning [7]. However, a single obstacle was designated as the target for motion planning. As a result, since the robot was instructed to avoid the single obstacle, it was difficult to plan the motions for multiple obstacles.

In general, there are multiple obstacles in navigation environments. Therefore, the robot is required to plan avoidance motions for these obstacles. However, the motion instruction cost increases with the number of obstacles in a framework of imitation learning. To address this issue, we proposed a motion planner that uses open-space images as inputs, which represent the areas where the robot can move, instead of obstacles [8]. As a result, the robot successfully avoided multiple obstacles, even though it was instructed with avoidance motions for only a single obstacle. However, it was difficult for the robot to determine the appropriate avoidance direction in response to the moving directions of dynamic obstacles. Therefore, we proposed a motion planner that uses open-space-difference images between two time steps as inputs. By focusing on the changed regions of the open space, the robot successfully avoided multiple dynamic obstacles responding to the moving directions.

However, the robot sometimes reoriented itself toward the destination immediately after avoiding an obstacle, which posed a risk of colliding with the obstacle again. This is caused by the high similarity between the image inputs corresponding to the two different instructed motion outputs after obstacle avoidance: straight and reorientation. For this challenge, we focus on using not only the current image but also past images as inputs to a motion planner based on a Time-Delay CNN [9].

This paper is composed of the following sections. In Section II, we describe the open-space images used as inputs to the motion planner. In Section III, we propose the motion planner based on a Time-Delay CNN that utilizes visually distinct time-series image inputs. The motion planner is trained through imitation learning as described in Section IV. In Section V, the navigation performance of the robot based on the motion planner is able to reach the destination by comparing to previous motion planners. Furthermore, we demonstrate that the robot based on the proposed motion planner is able to reach the destination by comparing to that based on a previous motion planner after obstacle avoidance in both simulation and real-world environments. Finally, in Section VI, we conclude this paper.

[1]Kenji Shibata is with Department of Mechanical and Intelligent Engineering Graduate School of Engineering Utsunomiya University, Japan mc246772@s.utsunomiya-u.ac.jp

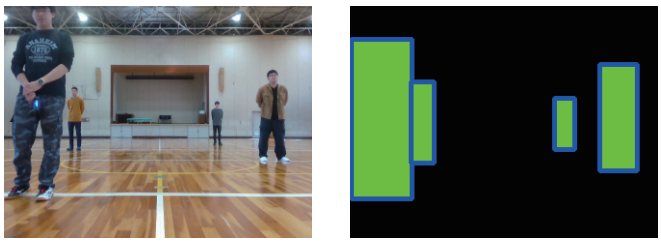
[2]Satoshi Hoshino is with Department of Mechanical and Intelligent Engineering Graduate School of Engineering Utsunomiya University, Japan hosino@cc.utsunomiya-u.ac.jp

978-1-6654-5784-2/26/$31.00 ©2026 IEEE

II. OPEN-SPACE IMAGES

We previously proposed the use of perception images for motion planning in our prior work [3]. These perception images were generated based on obstacle detection results. For each detected obstacle in an RGB image, a bounding box was drawn, and the pixels within the box were colored green, while all other pixels were set to black. This representation enabled the robot to plan avoidance motions in consideration of the obstacle. **Fig. 1** illustrates two examples of such perception images.

(a) Single obstacle



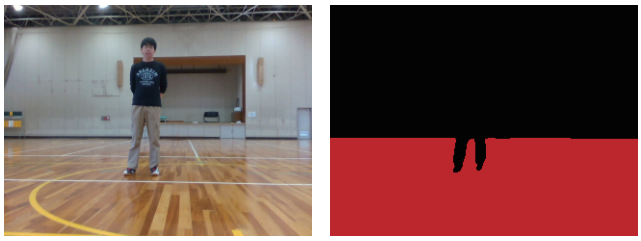
(b) Multiple obstacles

Fig. 1. Perception images for single obstacle and multiple obstacles

For an obstacle, the perception image as shown in Fig. 1(a) is generated. The single obstacle was designated as the target for motion planning. Since the robot was instructed to avoid the single obstacle, it was difficult to plan the motions for multiple obstacles as shown in Fig. 1(b). This is because the perception images were completely different, as shown in Fig. 1(a) and Fig. 1(b), depending on the number of obstacles. As a result, the perception image shown in Fig. 1(b) is unknown to the robot for motion planning.

For the problem described above, it is possible to instruct the robot to avoid multiple obstacles in a framework of imitation learning. However, the instruction cost increases with the number of obstacles and positions. For this challenge, we propose to use open-space images as inputs to the motion planner. The open-space image is generated using image segmentation and object detection techniques. For image segmentation from RGB images, we employ U-Net [10], a widely used method for semantic segmentation [11]. All the pixels of an RGB image are classified into a road/floor surface, where the robot can move, or the others. **Fig. 2** shows an example image generated through semantic segmentation.

In Fig. 2(a), a person is standing on the floor and is regarded as an obstacle. The RGB image is fed as the input to U-Net. Fig. 2(b) is the resulted segmentation image. In

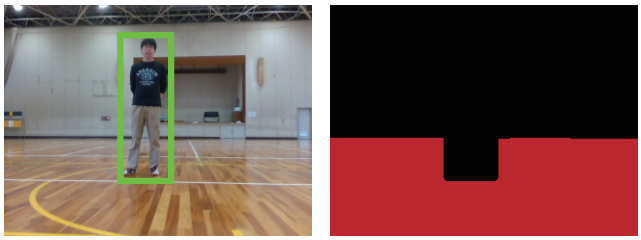


(a) RGB image (b) Segmentation image I_{Seg}

Fig. 2. Image processing through semantic segmentation based on U-Net

the image, the floor surface is colored by red and the others are colored by black. We can see that the floor area occupied by the person (around legs) is also colored by black.

In the segmentation image as shown in Fig. 2(b), the robot is allowed to move on the floor colored by red. In this regard, however, the classification accuracy for the person depends on the performance of the U-Net and the lighting conditions. Furthermore, the area between the legs of the person is also recognized as the floor. Since the robot might move toward the floor between the legs, we additionally detect the person as the obstacle from the RGB image shown in Fig. 2(a). In this paper, YOLOv5 [12] is used as the obstacle detector. **Fig. 3** shows the open-space image I_{OS} generated through the image segmentation and object detection techniques.



(a) Obstacle detection in RGB image (b) Open-space image I_{OS}

Fig. 3. Generated open-space image

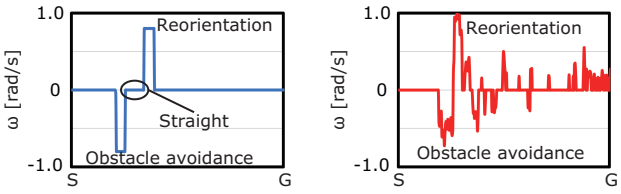
For the person in Fig. 2(a) detected by YOLOv5 as the obstacle, the bounding box is drawn by green as shown in Fig. 3(a). In Fig. 2(b), the pixels inside the bounding box are filled in black. Through the image segmentation and object detection as shown in Fig. 2(b) and Fig. 3(a), another image is generated as shown in Fig. 3(b). This is the open-space image, I_{OS}, used as inputs to the motion planner. In contrast to Fig. 2(b), we can see that the area between the legs of the person is also recognized as the obstacle in Fig. 3(b). Thus, the open-space image enables the robot to plan avoidance motions toward the right or left side of the obstacle.

III. MOTION PLANNER BASED ON TIME-DELAY CNN

A. Problem of Previous Motion Planners

A mobile robot based on a motion planner proposed in our previous works has successfully avoided multiple dynamic obstacles [8]. However, immediately after avoiding

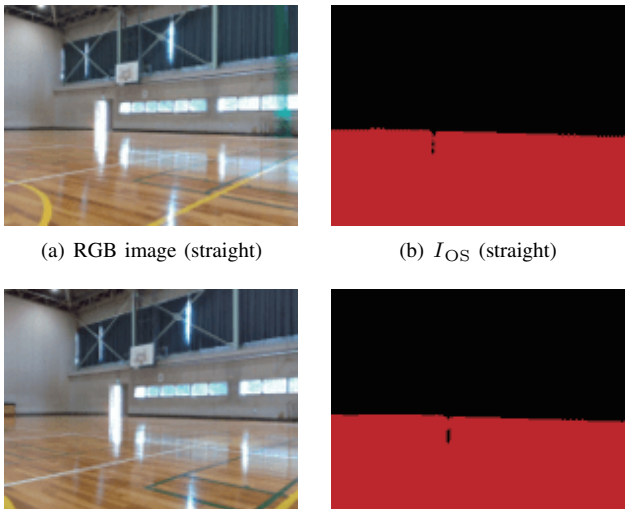
an obstacle, the robot sometimes planned a motion to reorient itself toward the destination. This motion was different from the one provided during instruction and may lead to a collision with an obstacle. **Fig. 4** shows the angular velocity outputs during both instruction and autonomous navigation.



(a) Instructed ω (b) Planned ω

Fig. 4. Comparison of angular velocity ω

From Fig. 4(a), it can be seen that the robot was instructed to move straight for a few seconds after avoiding an obstacle, and then to adjust its orientation toward the destination, i.e., reorientation. On the other hand, Fig. 4(b) shows that during autonomous navigation, the robot planned a motion to reorient itself immediately after avoiding an obstacle. As a result, the robot may fail to completely avoid the obstacle and be at risk of a collision. Fig. 5 shows the RGB and open-space images I_{OS} taken just before the straight and reorientation motions during the instruction phase.



(a) RGB image (straight) (b) I_{OS} (straight)

(c) RGB image (reorientation) (d) I_{OS} (reorientation)

Fig. 5. Comparison of image inputs for different motions

In the RGB image just before the straight motion shown in Fig. 5(a), the generated I_{OS} in Fig. 5(b) indicates that open space is observed across the entire image. Similarly, in the RGB image just before the reorientation motion shown in Fig. 5(c), the generated I_{OS} in Fig. 5(d) also shows open space throughout the image. Although the instructed motion outputs are different, the input images in Fig. 5(b) and Fig. 5(d) are highly similar. As a result, the motion planner failed to correctly associate the image inputs with the instructed motion outputs in the learning phase. Consequently, the robot reoriented toward the goal immediately after avoiding the obstacle.

B. Motion Planner with Past Time-Series Image Inputs

To address the challenge described in Section III-A, we focus not only on the current image but also on past images. **Fig. 6** shows the open-space images I_{OS} obtained two seconds before the images shown in Fig. 5.

(a) Straight (b) Reorientation

Fig. 6. I_{OS} taken two seconds before Fig. 5

As shown in Fig. 6(a), two seconds before the straight motion, open space is observed on both sides of the obstacle. In contrast, as shown in Fig. 6(b), two seconds before the reorientation motion, open space is observed across the entire image. Therefore, although the two current image inputs in Fig. 5 are highly similar, the corresponding images taken two seconds earlier are clearly different.

Therefore, past time-series images are also used as inputs in addition to the current image. This enables the motion planner to associate the image inputs with the different motion outputs instructed for each case. For these image inputs, we propose a motion planner based on the Time-Delay CNN [9]. **Fig. 7** shows the architecture of the proposed motion planner. Each filter has 32 channels. The convolutional layers use a kernel size of 5×5 with a stride of 1. The first pooling layer uses a 2×2 kernel with a stride of 2, and the second pooling layer uses a 3×3 kernel with a stride of 3.

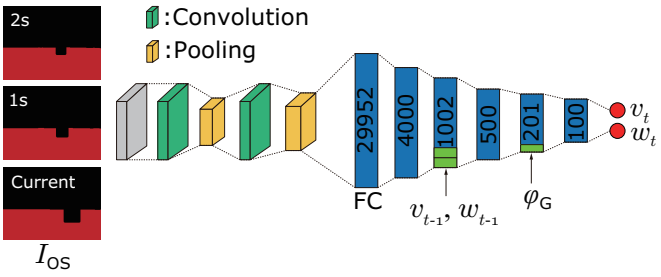


Fig. 7. Architecture of motion planner based on Time-Delay CNN

The three open-space images I_{OS} captured at different time, current, one second before, and two seconds before, are concatenated along the channel dimension and used as input. In the convolutional layers, temporal changes in the open space caused by obstacles and the robot's movement are extracted as features. The extracted features are flattened and fed into the fully connected layers. In the third layer, the velocity v_{t-1} and angular velocity ω_{t-1} in the previous

step are fed as inputs. In the fifth layer, the goal direction ϕ_{G} is used as an input. In the out layer, the velocity v_t and angular velocity ω_t are resulted as the outputs from the motion planner. The ReLU function is used as the activation function for each node in the fully connected layers.

IV. IMITATION LEARNING FOR MOTION PLANNER

A. Motion Instruction and Imitation Learning

In a framework of imitation learning, a policy is determined to plan appropriate motion outputs in response to the sensor inputs acquired by the robot at time t. The policy is represented by a deep neural network. To prepare the training dataset, a supervisor manually controls the robot to provide motion instructions, and the robot records the corresponding instruction data. The dataset D is mathematically expressed as follows:

$$D = \{\langle s_i, a_i^* \rangle\}_{i=1}^{N}, \tag{1}$$

where s_i represents the sensor input at the time of instruction, and a_i^* represents the instructed motion corresponding to the sensor input s_i. N represents the total number of instruction data samples recorded.

In the motion instruction phase, it is assumed that the expert follows an optimal policy π^*. Under this assumption, the robot is provided with optimal motion outputs a_i^* for inputs s_i. Similar to the expert policy, the robot's policy is represented by π, which determines motion outputs a_i based on the inputs s_i, as follows:

$$a_i = \pi(s_i; \theta), \tag{2}$$

where θ represents the convolutional filters and connection weights in the Time-Delay CNN.

In the learning phase, the parameter θ in Eq. (2) is updated to associate the inputs and outputs contained in the dataset defined in Eq. (1). The loss function L defined in Eq. (3) is used for updating the parameters θ as follows:

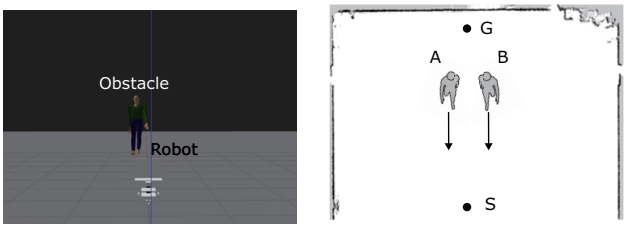
$$L = \frac{1}{N}\sum_{i=1}^{N}(a_i - a_i^*)^2. \tag{3}$$

In Eq. (3), a consists of the robot's velocity v and angular velocity ω. For clarity, we denote the component-wise losses in text by $L_v := \frac{1}{N}\sum_{i=1}^{N}(v_i - v_i^*)^2$ and $L_\omega := \frac{1}{N}\sum_{i=1}^{N}(\omega_i - \omega_i^*)^2$, so that the total loss satisfies $L = L_v + L_\omega$. The parameters θ are updated using Adam [13] to minimize this total loss L. As a result, the robot's policy π approximates the expert policy π^*, and thus π in Eq. (2) is used as the robot's motion planner.

B. Motion Instruction via Simulator

In imitation learning, the robot is manually controlled to provide motion instructions for obstacle avoidance and navigation toward the goal. For the motion instruction, simulation environment as illustrated in **Fig. 8** is used.

For the environment shown in Fig. 8(a), the occupancy grid map built using SLAM is shown in Fig. 8(b). By the supervisor, the robot is controlled to move from the start point S toward the goal point G, which is located 15 [m] ahead. During the instruction phase, the dataset D defined in Eq. (1) is recorded. The obstacle in front of the robot moves straight as indicated by the allowed lines at 1.0 [m/s]. The robot is instructed to avoid obstacle A by turning right and obstacle B by turning left, and then continue moving toward the goal point G. The robot begins obstacle avoidance at a distance of 5.0 [m]. The average duration of the straight motion after avoidance, based on 20 instructions, was 1.2 [s].



(a) Robot in environment for motion instruction (b) Top view of occupancy gridmap with obstacle

Fig. 8. Settings for motion instruction

The robot moves with a maximum velocity of 0.5 [m/s] and an angular velocity ranging from -0.8 to 0.8 [rad/s]. In addition, RGB images are taken by the onboard camera, and open-space images I_{OS} at the current time, one second before, and two seconds before are generated. Furthermore, for the goal direction input ϕ_{G} into the motion planner shown in Fig. 7, the current position is localized based on AMCL [14] using a 2D LiDAR.

V. NAVIGATION EXPERIMENTS

A. Settings for Experiments

In this experiment, the motion planner π_{OS}^{TD} shown in Fig. 7 is applied to the robot. For comparison, we further apply the motion planner $\pi_{\mathrm{OS+OSD}}^{TD}$, which is also based on Time-Delay CNN. In addition to the time-series open space images I_{OS}, the time-series open-space difference images I_{OSD} are used as inputs to the motion planner. Moreover, we apply the conventional motion planner $\pi_{\mathrm{OS+OSD}}$ proposed in our previous work [8], which also uses I_{OS} and I_{OSD} as inputs. In this regard, the motion planner is based on CNN with LSTM. In other words, only the current images of I_{OS} and I_{OSD} are fed to the motion planner. I_{OSD} is generated from the temporal difference between two open-space images I_{OS} obtained at the current time and 0.1 seconds before.

The robot based on each motion planner conducts five navigation trials toward the goal include obstacle avoidance for either A or B in the environment shown in Fig. 8(b). Through the navigation experiments, the motion planner with the highest avoidance performance is applied to the robot for both simulation and real-world environments with multiple obstacles. If the distance between the robot and an obstacle becomes less than or equal to 0.5 [m], the robot is stopped emergently, and the trial has failed.

B. Navigation Results

In the environment shown in Fig. 8(b), the robot successfully avoided obstacles and reached the goal in all five trials, regardless of the motion planners. In addition, as instructed by the supervisor, the robot avoided obstacle A by turning right and obstacle B by turning left. **Table I** shows the average distance at which the robot began planning to avoid the obstacle and the average time of straight motion after the avoidance, for each motion planner.

TABLE I

COMPARISON OF NAVIGATION RESULTS (5 TRIALS)

Motion planner	Relative distance [m]	Straight time [s]
$\pi_{\mathrm{OS+OSD}}$	4.9	0.20
π_{OS}^{TD}	4.9	1.1
$\pi_{\mathrm{OS+OSD}}^{TD}$	5.1	0.60

The relative distance at which the robot planned the obstacle avoidance motion was 4.9 [m] for both $\pi_{\mathrm{OS+OSD}}$ and π_{OS}^{TD}, and 5.1 [m] for $\pi_{\mathrm{OS+OSD}}^{TD}$, all of which were generally consistent with the instructed motions. In contrast, the robot based on π_{OS}^{TD} moved straight for 1.1 [s] after avoiding the obstacle. This result more closely matched the instructed motion than the results from $\pi_{\mathrm{OS+OSD}}$ and $\pi_{\mathrm{OS+OSD}}^{TD}$. **Fig. 9** shows the angular velocity of π_{OS}^{TD} during autonomous navigation in the environment with obstacle A.

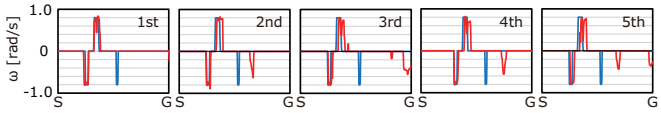
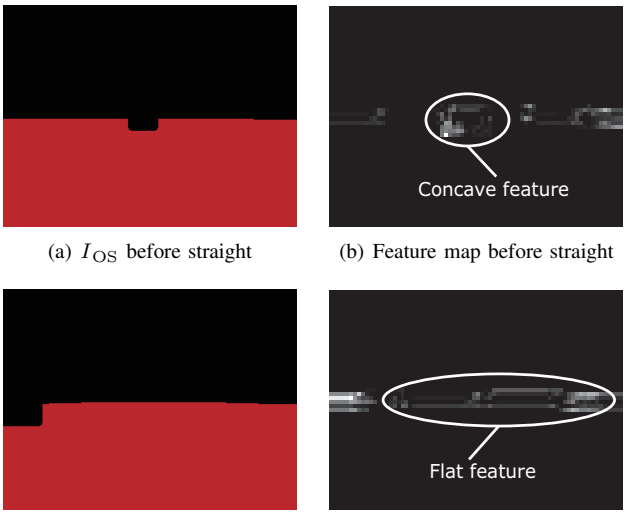


Fig. 9. Instructed (blue) and planned (red) angular velocities ω

Compared to the angular velocities shown in Fig. 4, the red angular velocity closely matches the instructed angular velocity shown in blue. In particular, it can be observed that in all five trials, the robot planned a straight motion for a few seconds immediately after avoiding the obstacle, followed by a reorientation motion. Therefore, based on the motion planner π_{OS}^{TD}, the robot successfully avoided dynamic obstacles and reached the goal, while accurately imitating the instructed motions using only the open-space images I_{OS} as input. This result indicates that by using the three time-series I_{OS} images as the inputs to the motion planner based on the Time-Delay CNN, the different instructed motion outputs were correctly associated with the inputs through imitation learning. Furthermore, the Time-Delay CNN enabled the robot to plan the motions in consideration of the temporal changes in open space. As a result, the robot successfully avoided the dynamic obstacles using only the I_{OS} as the inputs. **Fig. 10** shows the open-space images I_{OS} obtained two seconds before the straight motion after avoiding obstacle A and before the reorientation motion, as well as the corresponding feature maps.

In Fig. 10(a), we can see that open space is present on both sides of the obstacle. As a result of using the time-series images, in Fig. 10(b), a concave pattern near the center



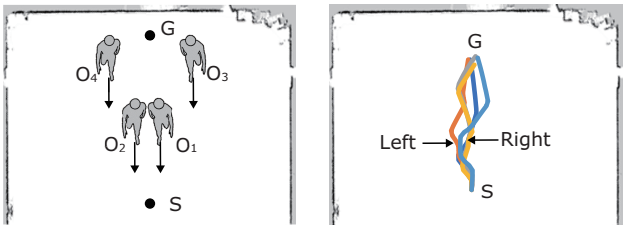
(a) I_{OS} before straight (b) Feature map before straight

(c) I_{OS} before reorientation (d) Feature map before reorientation

Fig. 10. Comparison of image features extracted from I_{OS} taken 2 seconds before motions

was extracted as the image feature. On the other hand, as shown in Fig. 10(c), we can see that open space is present on the right side of the obstacle. Therefore, in the feature map shown in Fig. 10(d), we can see that flat features are extracted along the boundary between the open space and other areas. Comparing Fig. 10(b) and Fig. 10(d), we can see that different features were extracted. This suggests that the robot was able to plan both the straight motion and the reorientation motion according to the different images.

Based on the above results, the robot subsequently conducted navigation trials in an environment with multiple dynamic obstacles using the motion planner π_{OS}^{TD}. It should be noted that this environment was not used during motion instruction and is therefore unknown to the robot. **Fig. 11** shows the environment and the trajectory of the robot during autonomous navigation.

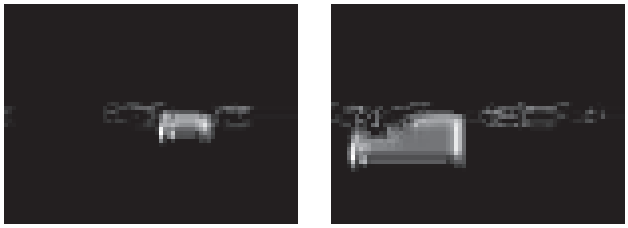


(a) Top view with multiple obstacles (b) Navigation trajectories

Fig. 11. Navigation trials in unknown environment

In the environment shown in Fig. 11(a), the robot moved from the start point S to the goal point G over a distance of 15 [m]. Through the experiments, the robot successfully avoided all obstacles, O_1 to O_4, and reached the goal in all five trials. In this regard, as shown in Fig. 11(b), the robot avoided obstacle O_4 either to the left or to the right. To discuss these different avoidance motions, Fig. 12 shows the feature maps extracted from the final convolutional layer

when the robot began planning to avoid obstacle O_4 to the left and to the right.

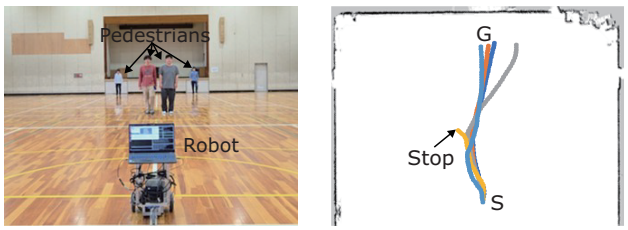


(a) Left (b) Right

Fig. 12. Comparison of feature maps right before obstacle avoidance toward right and left

In Fig. 12(a), features are extracted on the right side of the image where the open space has temporally changed. In Fig. 12(b), the features are extracted on the left side. The open space temporally changes due to the movement of obstacles. Therefore, the robot focused on these regions in the images and planned avoidance motions to the left and right, respectively.

These results demonstrate that the robot, based on the proposed motion planner, can autonomously navigate toward the goal while avoiding multiple dynamic obstacles, even in an unknown environment. Therefore, the motion planner, π_{OS}^{TD}, was further applied to the actual robot for autonomous navigation in a real-world environment that replicated the settings shown in Fig. 11. **Fig. 13** shows the environment and the navigation trajectories of the robot for the five trials. In these trials, we used U-Net trained on a real-world dataset.

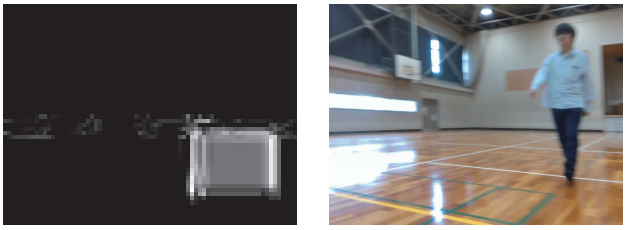


(a) Actual robot and multiple obstacles (pedestrians) (b) Navigation trajectories

Fig. 13. Navigation trials of actual robot in real-world environment

In Fig. 13(a), four pedestrians are present in front of the robot as dynamic obstacles. The robot is equipped with a single camera, RealSense D435. As shown in Fig. 13(b), the robot avoided these obstacles and reached the goal in four out of five trials. In one trial, however, the robot failed to avoid an obstacle and was forced to make an emergency stop. **Fig. 14** shows the feature map and RGB image taken just before the failed obstacle avoidance.

In the map shown in Fig. 14(a), features are extracted on the right side, where the open space has temporally changed. As a result, the robot planned an avoidance motion to the left. The motion was correct, as instructed. However, as shown in Fig. 14(b), the pedestrian was walking straight toward



(a) Feature map (b) RGB image

Fig. 14. Feature map extracted from RGB image during failure

the robot from the front right, and the robot, which avoided to the left, eventually moved closer to the pedestrian. This is because the robot was instructed to avoid an obstacle approaching directly from the front, and no instructions were provided for obstacles approaching from diagonal directions. In our previous work, we showed that providing instructions for avoidance motions against obstacles approaching diagonally is effective [15]. With the additional instructions, the motion planner enables the robot to avoid obstacles moving in different directions by utilizing the open space.

The results above demonstrate that the motion planner developed in the simulation environment was fully applicable to the actual robot in the real-world setting. In fact, both the virtual and actual robots successfully reached their destination while avoiding multiple dynamic obstacles. Therefore, it was finally shown that the proposed motion planner is effective for autonomous navigation of a mobile robot, including obstacle avoidance.

VI. CONCLUSION

In this paper, we focused on the autonomous navigation of mobile robots, particularly with respect to motion planning in environments with multiple dynamic obstacles. In the context of motion planning, we proposed a novel motion planner based on a Time-Delay CNN, which uses past time-series images as inputs. These visually distinct inputs allowed the motion planner to associate them with corresponding motion outputs through imitation learning. As a result, the robot based on the motion planner was able to imitate the instructed motions. Finally, even though the avoidance motions were instructed using only a single obstacle in a simulation environment, both the virtual and actual robots successfully reached the destination while avoiding multiple obstacles in unknown simulation and real-world environments. From these results, it was demonstrated that the proposed motion planner is effective for autonomous navigation including obstacle avoidance.

REFERENCES

[1] S. Hoshino and J. Sumiyoshi, "End-to-End Discrete Motion Planner based on Deep Neural Network for Autonomous Mobile Robots," *IEEE/SICE International Symposium on System Integration*, pp. 12–17, 2020.

[2] S. Hoshino and J. Sumiyoshi, "Discrete Motion Planner based on Deep Recurrent Neural Network for Mobile Robot Obstacle Avoidance in Dead-End Environments," *IEEE/SICE International Symposium on System Integration*, pp. 979–984, 2022.

[3] Satoshi Hoshino and Yu Kubota, “Mobile Robot Motion Planning through Obstacle State Classifier,” *Proceedings of the SICE Annual Conference 2023*, pp. 120-126, 2023.
[4] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-Based Learning Applied to Document Recognition,” *Proceedings of the IEEE*, Vol. 86, No. 11, pp. 2278–2324, 1998.
[5] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Computation*, Vol. 9, No. 8, pp. 1735–1780, 1997.
[6] S. Hoshino, Y. Kubota, and Y. Yoshida, “Motion Planner Based on CNN with LSTM through Mediated Perception for Obstacle Avoidance,” *SICE Journal of Control, Measurement, and System Integration*, Vol, 17, No. 1, pp. 19–30, 2024.
[7] Y. Liu, A. Gupta, P. Abbeel, and S. Levine, “Imitation from Observation: Learning to Imitate Behaviors from Raw Video via Context Translation,” *IEEE International Conference on Robotics and Automation*, pp. 1118–1125, 2018.
[8] K. Shibata and S. Hoshino, “Open-Space-Based Motion Planner of Mobile Robot for Multiple Obstacle Avoidance,” *SICE Festival with Annual Conference*, pp. 770–775, 2024.
[9] J. Lang *et al*., “A Time-Delay Neural Network Architecture for Isolated Word Recognition,” *Neural Networks*, Vol. 3, pp. 23–43, 1990.
[10] O. Ronneberger, P. Fischer, and T. Brox, “U-Net: Convolutional Networks for Biomedical Image Segmentation,” *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241, 2015.
[11] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz and D. Terzopoulos, “Image Segmentation Using Deep Learning: A Survey,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–22, 2021.
[12] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” *Proceedings of the IEEE Computer Vision and Pattern Recognition*, pp. 779–788, 2016.
[13] D. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” *Proceedings International Conference for Learning Representations*, pp. 1–15, 2015.
[14] D. Fox, W. Burgardy, F. Dellaert and S. Thrun, “Monte Carlo Localization: Efficient Position Estimation for Mobile Robots,” *Innovative Applications of Artificial Intelligence Conference*, pp. 343–349, 1999.
[15] S. Hoshino and R. Suaga, “Mobile Robot Obstacle Avoidance Based on Optical Flow Images in Motion Planning,” *SICE Festival with Annual Conference*, 2025, (accepted).