2024 IEEE/SICE International Symposium on System Integration (SII)
January 8-11, 2024. Ha Long, Vietnam

Autonomous Navigation of a Mobile Robot with a Monocular Camera using Deep Reinforcement Learning and Semantic Image Segmentation

Ryuto Tsuruta[1] and Kazuyuki Morioka[1]

***Abstract*—Navigation systems for mobile robots using deep reinforcement learning have the potential to achieve autonomous movement without relying on precise environmental maps. This study focuses on autonomous navigation of mobile robots using monocular camera images as input through deep reinforcement learning. Typically, RL models are trained in simulation environments. However, there are gaps between simulation and real-world environments, making it challenging to apply trained RL models to actual robots. This challenge is particularly pronounced when using images as input. Semantic segmentation is employed to address this issue, as it can simplify complex RGB images into segmented images, thereby reducing the gaps between environments. In this study, a RL model to reach their destinations and a semantic segmentation model are acquired for mobile robot navigation. These models are applied to a ROS-based autonomous navigation system. Real-world experiments confirm the successful application of the learned models in actual environments.**

I. Introduction

In recent years, there has been a growing amount of study focused on the autonomous behavior of robots using deep reinforcement learning. In the context of mobile robot navigation, the application of deep reinforcement learning holds the promise of achieving navigation without precise environmental maps. Thus, this paper introduces a study for autonomous navigation of mobile robots to their destinations, using monocular camera images as input.

Generally, models based on deep reinforcement learning are trained in simulation environments. However, gaps between simulation and real-world environments make it difficult the trained models to be applied to actual mobile robots. Especially when utilizing images as the input of RL (Reinforcement Learning) models, more significant gaps between environments are appeared because of the diverse and rapidly changing nature of real-world images. Therefore, the paper focuses on semantic segmentation for input images of RL models. The semantic segmentation provides color-codes separating different areas such as floors and walls within the images. This enables the simplification of diverse RGB images in real-world environments and thus reducing the gaps between environments. Additionally, this approach is expected to enhance robustness against changes such as variations in lighting conditions in the real-world environments.

In this paper, the RL models are trained for the task of traveling to destinations with segmentation images as input. The general corridors such as university buildings are considered as the environments for training and navigation. Also, an actual navigation system incorporating the acquired models is developed, and the several navigation experiments demonstrate the effectiveness of the proposed system.

[1]Meiji Univ, Graduation School of Advanced Mathematical Sciences, Network Design Program, Japan (email: cs223020@meiji.ac.jp)

II. Related Works

A. Robot system using reinforcement learning

There are several studies utilizing deep reinforcement learning for navigation of mobile robots. The system configuration of the paper is based on a robot system proposed by Kato *et al*., which involves the application of deep reinforcement learning to the autonomous navigation of mobile robots [1]. They use range data obtained from a 2D-LiDAR as input states for the RL model that is trained for traveling towards the destinations while avoiding dynamic obstacles. The trained models are applied to actual robot systems and demonstration of indoor navigation in the real world was performed. The other several approaches using range data of 2D-Lidar as agent input have been proposed. R. Guldenring *et al*. achieved a local planner for human-aware navigation based on deep reinforcement learning [2]. A. Faust *et al*. presented long-range navigation over 200 m of mobile robots [3]. An application to multi-robot collision avoidance system using deep reinforcement learning was proposed by P. Long *et al* [4]. This is also based on 2D-Lidar as observation input.

On the other hand, camera images can be applied to RL models as input states. Yokoyama *et al*. [5] achieved mobile robot navigation tasks using depth information extracted from monocular camera images. J. Ding *et al*. also proposed monocular camera-based obstacle avoidance system [6]. In this system, pseudo-laser measurement is generated from depth-semantic images processed from the monocular camera images. The model configuration of these approaches is almost same with 2D-Lidar-based model, although monocular camera images are utilized as observation. Also, the RL model design based on camera image inputs has been proposed. Zhu *et al*. [7] performed indoor navigation using observation images and target images as inputs. Hong *et al*. [8] utilized semantic segmentation processing in reinforcement learning to adapt RL models trained in simulation environments to real-world applications. They conducted several experiments involving obstacle avoidance and target following tasks, demonstrating that the model using segmentation outperforms existing models. However, navigation tasks such as traveling long distance were not described in [8] at all. Compared with them, our study acquires a RL model capable of navigation even in situations where the destinations are not present within the images by providing distance and angle to the destination as additional input alongside the segmented image. Consequently,

Explain why they choose the semantic segmentation as the input

The results:

2024 IEEE/SICE International Symposium on System Integration (SII) | 979-8-3503-1207-2/24/$31.00 ©2024 IEEE | DOI: 10.1109/SII58957.2024.10417188

the model can take appropriate actions even in scenarios where the destinations are located in unseen areas beyond corners. Several real-world experiments in this paper demonstrate the feasibility of long-distance navigation by sequentially following waypoints.

B. Simulation Environment

In this study, training environments using the Unity game engine are performed with the Unity ML-Agents Toolkit as a tool for conducting reinforcement learning within the Unity platform. Juliani *et al.* [9] discussed the use of the Unity as a simulation environment for reinforcement learning. They conducted simulation experiments using the Unity game engine and the Unity ML-Agents Toolkit. Through their work, they demonstrated the effectiveness of utilizing Unity as a platform for reinforcement learning.

III. Proposed System

The overview of the robot system proposed in the paper is presented in Figure 1. The system assumes that the mobile robot has the capability of self-localization and acquiring monocular camera images of the front view. The inputs to the RL model include segmentation images separating the floor region and other regions obtained through semantic segmentation. Also, relative distances and angles between the self-localization and the target location are added as inputs for RL models. The outputs of the RL models consist of continuous values of linear velocity and angular velocity. The system requires proper training of both the RL model for determining robot behavior and the semantic segmentation model.

Training of the RL models is performed by employing Unity as simulation environments. In the environments in Unity, segmented images are obtained and utilized for training the action through deep reinforcement learning based on PPO[10]. This ensures that the mobile robot reaches the destinations without colliding with obstacles. Generally, to obtain the semantic segmentation model, the labor-intensive task of image annotation is required at first. In this study, the mask images were automatically generated by employing stereo cameras for floor detection. This is enabled us to collect a sufficient amount of training data and successfully obtain a highly accurate segmentation model.

To integrate the acquired models into an actual robot system, ROS (Robot Operating System) platform is utilized. Especially, a node publishing segmented images through the semantic segmentation model, and a node subscribing self-localization and segmented images and publishing output linear velocity and angular velocity through the RL model are developed for utilizing trained models.

Figure 2 illustrates the appearance of the robot system used in this study. The robot is equipped with a monocular camera and a 2D-LiDAR. In the current system, although velocity outputs from RL models are obtained by the monocular camera, the 2D-LiDAR is also utilized for localization.

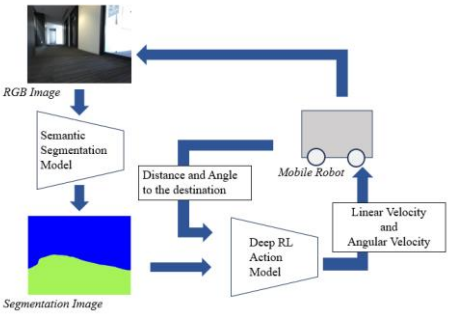


Figure 1. Proposed robot system.

Figure 2. Appearance of the robot used for experiment.

IV. Behavior Model

A. Learning Environment

In this study, we use an indoor environment consisting of corridor in the university building as the robot navigation environment. A simulation environment, that is similar with the shape of the actual environment, was developed for model training of the mobile robot's RL model. Unity ML-Agents is utilized as the tool for conducting reinforcement learning within the Unity environment. The developed simulation environment, as depicted in Figure 3, features a yellow-green color for the floor and blue for other elements, because the proposed system assumes that the mobile robot acquires color-segmented camera images from the environment. In this study, the agent learns the task of navigating to destinations located approximately 5 to 8 meters away without colliding with walls.

This is important, the agent is only rely on the camera and lidar successing the movement without colliding the walls. They didin'y consider the condition if the ground is not clean

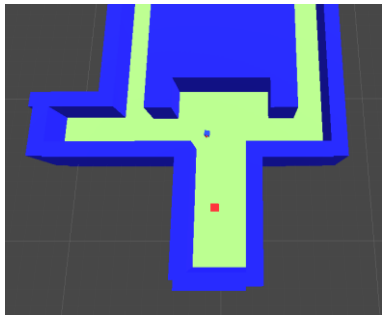


Figure 3. Driving environment. The mobile robot learns the behavior to reach the destination by running in this environment.

B. *Learning Settings*

In the context of reinforcement learning, essential components include the state, action, and reward. At first, we describe "state". "State" refers to the observable information by the agent, serving as inputs to the RL model. In this context, three distinct states have been defined:

1. Monocular camera image (with dimensions of 84x84 pixels)
2. Distance to the destination
3. Angle to the destination

The reason for using low-quality input images is to improve the convergence of the learning process. In this learning scenario, camera images are only needed to distinguish road shapes and identify wall locations, so high resolution images are not necessary.

Next, we describe "action". "Action" refers to the outputs of the RL model. In this context, the possible actions that the agent can take have been defined as two continuous values:

1. Linear velocity, denoted as v [m/s] with a range of $0 \leq v \leq 2$.
2. Angular velocity, denoted as ω [rad/s] with a range of $-1 \leq \omega \leq 1$.

Finally, we describe "reward". "Reward" refers to what the environment provides to the agent when it takes a specific action in a particular state. The agent learns model to maximize the obtained rewards. The rewards configured for training are presented in Table 1. The desired behavior of the agent is to reach the destination as quickly as possible without colliding with walls. Therefore, upon successfully reaching the destination, the agent obtains a substantial positive reward, whereas collisions with walls result in significant negative rewards. Also, s slight negative reward is given for each action taken to reduce unnecessary actions. Additionally, the other rewards are added to encourage the agent to move towards the destination more effectively by providing rewards based on the changes in distance to the destination and the angular difference between the agent's orientation and the direction of the destination.

TABLE I. REWARD SETTINGS

Reward condition	Reward
Robot reaches the destination	+100.0
Robot collides with the wall	-50.0
Robot selects an action	-0.01
Closes the distance between the agent and the destination	+0.01
The angle to the destination is less than -90 or more degrees	-0.1
The angle to the destination is less than -150 or more than 150 degrees	-5.0

C. *Training Flow and Learning Results*

In training of models, an episode in reinforcement learning is iterated. When an episode starts, the agent's initial position and orientation, a target destination are decided. The potential initial points for the agent are distributed throughout the entire corridor, and one of these points is chosen as the initial position. The agent's initial orientation is set to a random direction. The target destination of the agent is selected from candidate destination points. Candidate destination points for the agent are pre-set at locations 5 to 8 meters away from initial candidate points respectively. The target destination is randomly selected from the candidate destination points corresponding to the decided initial position of the agent. Once the agent's initial position and destination are set, the agent proceeds to navigate until it either reaches the destination or collides with any walls.

This sequence constitutes the content of a single episode. By repeating episodes, the RL model is trained based on the specified rewards, allowing the agent to learn and improve its behavior over time.

The training process was carried out for 1,500,000 steps. In this context, the term "step" refers to the number of times the agent has taken an action. As the evaluation, 1,000 episode trials deploying the trained model in the simulation environment were performed. The agent managed to successfully reach the destination in 994 trials. This indicates the effective RL model, guiding the agent to nearby destinations, was generated.

Figure 4 depicts the cumulative reward progression throughout the training process. The graph demonstrates that the agent accumulates higher rewards as training advances. This indicates effective learning progress. Figure 5 illustrates episode length during the training process. The graph indicates that the number of actions per episode decreases as training progresses. These results suggests that the agent has obtained an effective RL model that enables it to reach the destination with fewer actions.

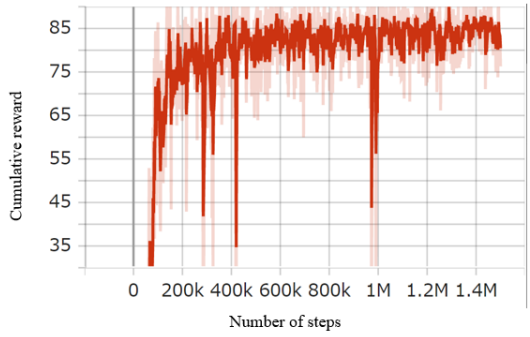


Figure 4. Cumulative reward.

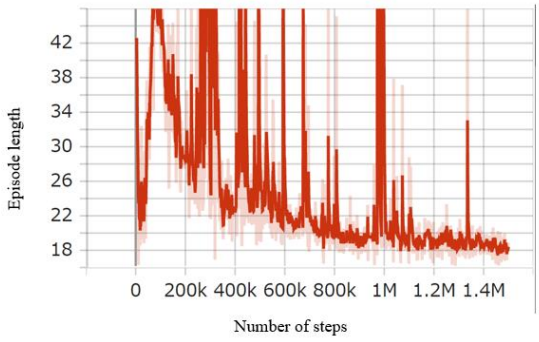


Figure 5. Episode length.

D. Preliminary Experiment

We conducted preliminary real-world experiments to validate whether the model obtained through reinforcement learning can be applied in an actual environment. A ROS-based system constructed for real-world robot navigation is depicted in Figure 6. "1) /zed_plane" node generates segmentation images from raw camera images. In this experiment, segmentation image generation is achieved using floor plane detection through the ZED2 stereo camera. "2)/onnx_node" loads the obtained RL model, subscribes segmentation images, distance and angle difference to the destination as inputs and publishes velocity command values as outputs.

The obtained RL model is trained to accomplish the task of reaching destinations at close distances. Long range navigation can be achieved by following a sequence of waypoints set in advance. The actual robot path in the experiment is presented in Figure 7. This shows that the long range navigation can be achieved using the trained RL model. However, several problems were found in this experiment. As an example, the transition of velocity outputs from the RL model while moving in the segment 1 are illustrated in Figures 8. Although this segment is a simple straight corridor, Figure 8 reveals that the robot was iterating acceleration and deceleration. That indicates instability of velocity outputs. This instable behavior is attributed to inaccurate floor detection using the stereo camera as shown in Figure 9. When floor detection fails, the input image to the RL model consists of large blue color areas. During training process, this blue color pattern signifies the presence of walls in front of the mobile robot. In this case, the velocity outputs become almost 0 for avoiding walls.

Segment 1 Problem: the failure of floor detection make the velocity and angular velocity rapid and frequent changes

On the other hand, for the segment 2 where the robot turns left at the corner, the velocity outputs of are shown in Figure 10. This figure shows that the continuous angular velocity of 0.2 (rad/s) was output consistently during the left turn. The successful navigation in segment 2 indicates that the acquired RL model can output efficient velocity commands even when turning around corners, provided that accurate segmentation images are obtained.

The 0.2 rad/s is the highest turn rate that we set.

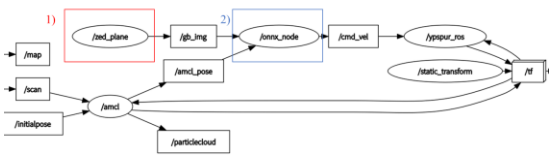


Figure 6. ROS system configuration in preliminary experiment.

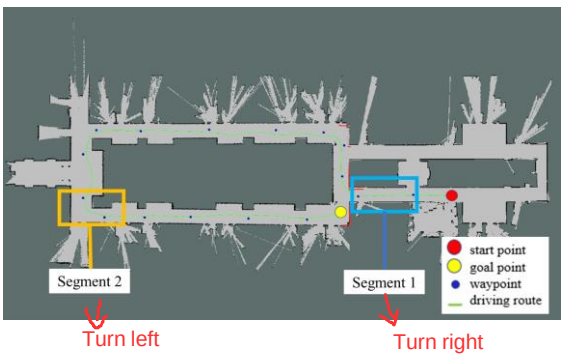


Figure 7. Driving route in preliminary experiment.

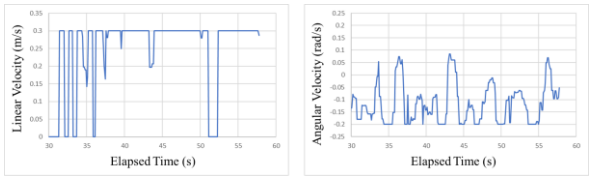


Figure 8. Actions in segment 1.

ZED2 is a binocular camera

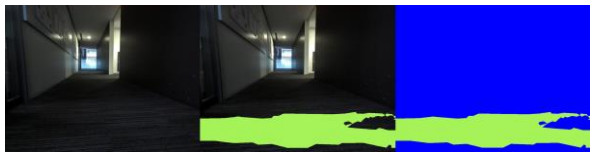


Figure 9. Failed segmentation image.

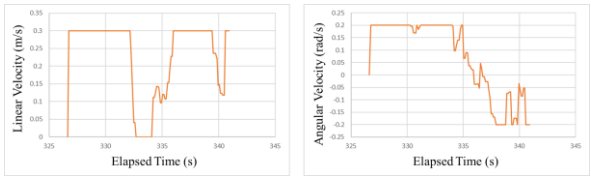


Figure 10. Action in segment 2.

V. Experiment

In this section, we acquire a semantic segmentation model to segment monocular camera images. Furthermore, we integrate the obtained model into a real-world system and conduct driving experiments.

A. Semantic Segmentation

To train the semantic segmentation model, a dataset consisting of raw images and corresponding segmented mask images is required. We utilized the floor detection images achieved through preliminary experiments with the ZED2 for mask images, because manually making mask images is a laborious task. Specifically, we collected raw RGB images and their corresponding mask images while operating a mobile robot equipped with ZED2. During this process, low-accuracy floor detection images, as shown in Figure 9, were included in the dataset. Therefore, after collecting the images, we manually removed the low-accuracy mask images to create a high-quality dataset.

Examples of the saved dataset are presented in Figure 11. In this study, a total of 191 image pairs as the dataset were obtained. Among these images, 150 sets were allocated for training, 16 sets for model validation during the learning process, and the remaining 25 sets for testing purposes.

For training the semantic segmentation model, the "Segmentation Models" library was utilized. The "Segmentation Models" library is a Python library that facilitates the easy implementation and training of neural networks for segmentation tasks. In this study, the Unet architecture was employed as the network model, with ResNet50 as the encoder. Pretrained weights from ImageNet

were utilized for training the model. The training was conducted over 40 epochs.

The validation of the trained model is conducted for model evaluation. The evaluation of the model employs the Dice Loss as the loss function, and the IoU value, a commonly used metric for measuring the accuracy of semantic segmentation.

Figure 12 illustrates the progression of the Dice Loss during the training process. The figure shows that as the training progresses, the Dice Loss consistently decreases. Decreasing the Dice Loss for both the training data and the validation data over time means that a generalized model for floor segmentation was obtained.

Figure 13 illustrates the progression of the IoU values during the training process. The figure shows that the IoU values increase as the training advances and converge around high value to both training and validation data. These results indicate the successful learning of the model and capable of accurate segmentation. This also suggests that the trained model possesses generalization capabilities, which is inferred from the consistent convergence of IoU values.

They choose a epoch in test data process for evaluating the model performance

Segmentation performances to test data should be also evaluated. The trained model in the 38th epoch, which exhibited the highest IoU value on the validation data, is utilized for validation of the test data. The Dice Loss for the test data is measured at 0.05842, and the IoU value is 0.9147. These results indicate consistency with the Dice Loss and the IoU value observed in the training process.

Figure 14 show several examples of applying the trained model to the raw test data through semantic segmentation. The figure displays the raw image, the mask image created using ZED2, and the predicted mask image by the acquired model from left to right respectively. This illustration demonstrates the successful acquisition of a model capable of effectively segmenting the floor regions.

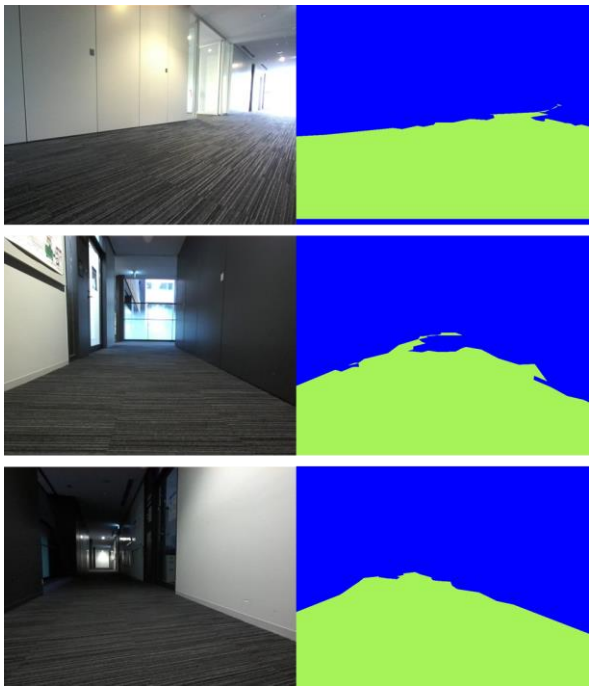
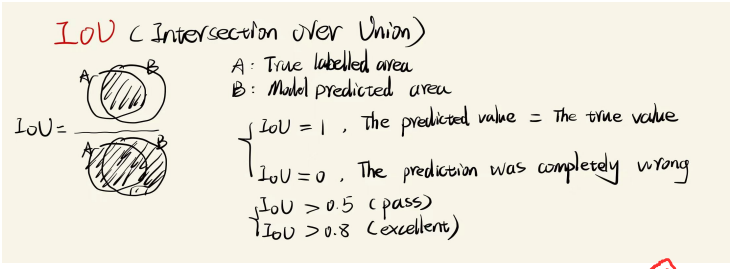


Figure 11. Examples of the dataset.

IoU close to 0.9, the performance is good.



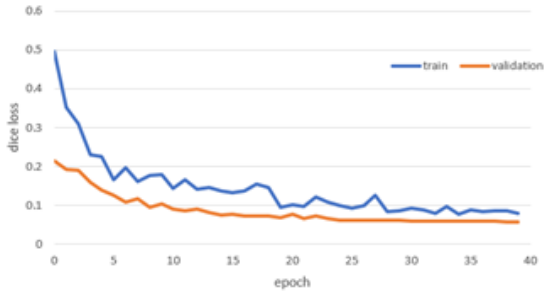


Figure 12. Dice loss.

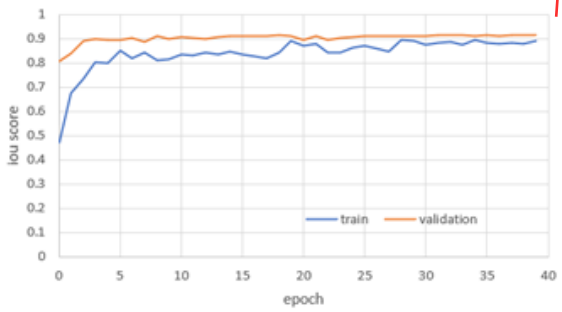


Figure 13. IoU score.

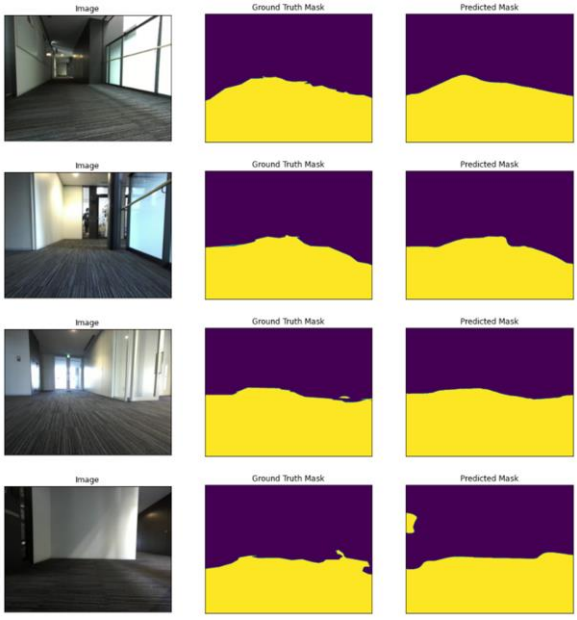


Figure 14. Results of semantic segmentation.

B. *Navigation Experiment*

An actual mobile robot system incorporating the acquired RL model and semantic segmentation model was developed and tested through navigation experiments in a real world.

/onnx_node : Run RL model to generate robot motion commands based on input environment information
/amcl (Adaptive Monte Carlo Localization) : is used for localization
/ypspur_ros : receive the '/cmd_vel' control signal to directly drive the robot movement.
/tf: Robot position information

Figure 15 shows the ROS system configuration implemented in the mobile robot. The only difference from the preliminary experiment system was the image segmentation node. The node performing segmentation using ZED2 in the preliminary system was replaced with a new node utilizing the semantic segmentation model "/img_segmentation". This node performs semantic segmentation with the trained segmentation model and publishes floor segmentation images.

The navigation experiment using the segmentation model was conducted on a similar path as the preliminary experiment. As a result, the mobile robot followed almost the same trajectory as in the preliminary experiment. However, we confirmed that the robot's driving performance significantly decreases when the camera height deviates from the set value used in training, which is 25cm from the floor. Therefore, additional experiments at camera heights of 33 cm, 44 cm, and 71 cm were conducted for evaluation to differences of image appearances. As a result, achieving long-distance driving, as shown in Figure 16, was only possible when the camera height was approximately the same as during training, at 33cm. At the height of 44 cm, the velocity outputs were unstable, and the robot frequently failed to make turns. At the height of 71 cm, the velocity often dropped to zero, preventing effective driving. Figure 17 illustrates the changes in camera images due to variations in camera height. This observation indicates that when the visual perception of the environment differs from images in training, driving performance is significantly compromised.

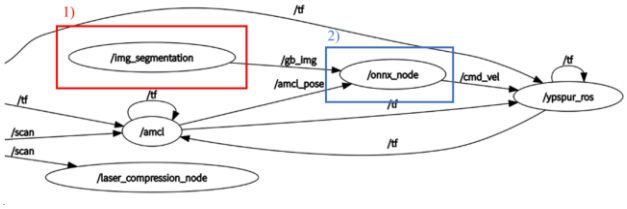
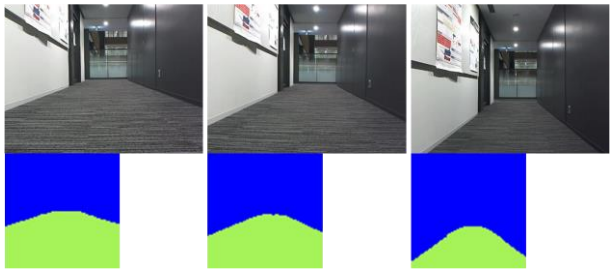


Figure 15. ROS system configuration. Node 1) contains semantic segmentation model and publish segmentation image. Node 2) contains action model and publish linear velocity and angular velocity.

Figure 16. Driving route.



(a)33cm (b)44cm (c)71cm

Figure 17. Differences in images due to changes in camera height.

VI. Conclusion

In this paper, we introduced an autonomous mobile robot system with a trained model based on monocular camera images using deep reinforcement learning, along with utilizing a semantic segmentation model. We implemented a ROS-based driving system by applying these models and conducted driving experiments. The mobile robot achieved long-distance navigation following waypoints utilizing trained models in the real-world experiments. Although the results describe the effectiveness of the proposed system, changes in camera height, that mean differences of floor detection appearances, significantly affect the driving performance. This implies that the autonomous driving model obtained in this study might lack versatility and may not adequately adapt to variations in factors such as road width. As future works, we will focus on acquiring a robust model that can handle changes in camera height and road width, as well as achieving navigation in complex environments with obstacles.

References

[1] Y. Kato, K. Kamiyama and K. Morioka, "Autonomous robot navigation system with learning based on deep Q-network and topological maps," 2017 IEEE/SICE International Symposium on System Integration (SII), Taipei, Taiwan, 2017, pp. 1040-1046, doi: 10.1109/SII.2017.8279360.

[2] R. Guldenring, M. Görner, N. Hendrich, N. J. Jacobsen and J. Zhang, "Learning Local Planners for Human-aware Navigation in Indoor Environments," 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 2020, pp. 6053-6060, doi: 10.1109/IROS45743.2020.9341783.

[3] A. Faust et al., "PRM-RL: Long-range Robotic Navigation Tasks by Combining Reinforcement Learning and Sampling-Based Planning," 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia, 2018, pp. 5113-5120, doi: 10.1109/ICRA.2018.8461096.

[4] P. Long, T. Fan, X. Liao, W. Liu, H. Zhang and J. Pan, "Towards Optimally Decentralized Multi-Robot Collision Avoidance via Deep Reinforcement Learning," 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia, 2018, pp. 6252-6259, doi: 10.1109/ICRA.2018.8461113.

[5] K. Yokoyama and K. Morioka, "Autonomous Mobile Robot with Simple Navigation System Based on Deep Reinforcement Learning and a Monocular Camera," 2020 IEEE/SICE International Symposium on System Integration (SII), Honolulu, HI, USA, 2020, pp. 525-530, doi: 10.1109/SII46433.2020.9025987.

[6] J. Ding et al., "Monocular Camera-Based Complex Obstacle Avoidance via Efficient Deep Reinforcement Learning," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 33, no. 2, pp. 756-770, Feb. 2023, doi: 10.1109/TCSVT.2022.3203974.

[7] Y. Zhu et al., "Target-driven visual navigation in indoor scenes using deep reinforcement learning," 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 2017, pp. 3357-3364, doi: 10.1109/ICRA.2017.7989381.

[8] Z. Hong et al., "Virtual-to-Real: Learning to control in visual semantic segmentation, " arXiv preprint arXiv:1802.00285, 2018.

[9] A. Juliani et al., "Unity: A general platform for intelligent agents," arXiv preprint arXiv:1809.02627, 2018.

[10] J. Schulman, F. Wolski, P. Dhariwal, A. Radford and O. Klimov, "Proximal policy optimization algorithms", arXiv preprint arXiv:1707.06347, 2017