MM Science Journal | www.mmscience.eu
ISSN 1803-1269 (Print) | ISSN 1805-0476 (On-line)
Special Issue | TEAM 2024
Transdisciplinary and Emerging Approaches to Multidisciplinary Science
11.9.2024 – 13.9.2024, Ostrava, Czech Republic
DOI: 10.17973/MMSJ.2024_10_2024078

TEAM2024-00018

OPTIMAL OBSTACLE AVOIDANCE STRATEGY USING DEEP REINFORCEMENT LEARNING BASED ON STEREO CAMERA

CHI-HUNG NGUYEN[1], QUANG-ANH VU[1], KIM-KHOI PHUNG CONG[1], THAI-VIET DANG[1*]
[1]School of Mechanical Engineering, Hanoi University of Science and Technology, Hanoi, Vietnam
*Corresponding author; e-mail: viet.dangthai@hust.edu.vn

Abstract
Mobile robots (MRs) are exerting a significant influence in industrial as well as residential settings. Concurrently with the escalating advancement of technology, the incorporation of artificial intelligence algorithms into the obstacle evasion issue of MRs is gaining increasing attention. The paper utilizes Deep Reinforcement Learning (DRL) to a MR that is furnished with a camera. Images captured by a stereo camera will be inputted into the YOLO-v8 model to identify obstacles situated in the path of the MR. Subsequently, the distances to these obstacles will be regarded as the state of the MR. The information was utilized to train a Deep Q-Network. Throughout this training process, the system acquires the capability to determine suitable actions for the MR to advance towards the destination while circumventing obstacles. Each action executed by the MR is accompanied by a reward, with the path yielding the most desirable outcome receiving the highest reward. The outcomes of the simulations conducted on the Robot Operating System 2 (ROS2) corroborate the effectiveness of this Deep Reinforcement Learning technique for the task of obstacle avoidance.

Keywords:
Deep reinforcement learning, mobile robot, obstacle avoidance, YOLO-v8 model

1 INTRODUCTION

The evolution of robots is fundamentally transforming human existence. Concurrently, there is a growing emphasis on the challenges of path planning and obstacle avoidance for MRs [Dang 2023a]. The task of charting a course for a MR is commonly segmented into two distinct problems: in known and unknown environments. Conventional techniques like A* and Dijkstra are frequently utilized on grid-based maps derived from SLAM (Simultaneous Localization and Mapping) [Liu 2021]. Nonetheless, unfamiliar settings present notable hurdles. Employing algorithms designed for known environments typically necessitates recalculating paths upon encountering new obstacles. This can result in the MR becoming trapped in a cycle, unable to identify a feasible route. Furthermore, if the MR reaches its destination, the path may only be optimal within the existing environment [Dang 2023c].

Rapidly exploring random tree (RRT) [Dang 2023c], a well-established technique devised for MRs in unfamiliar environments, constructs a "tree" using points on the map. Branches of the tree are randomly produced and linked to the nearest point until a path is established from the starting point to the goal. An ongoing challenge for this algorithm and its variations is ensuring path smoothness. Another method for path planning in unknown environments is the artificial potential field [Li 2024], which views the MR as a point influenced by forces in a field where the target exerts a positive force, and obstacles exert repulsive forces. By responding to these forces, the MR can determine an efficient path to the goal. However, this approach may encounter the issue of local minima, hindering the MR's progress.

Recently, DRL has emerged as a prevalent machine learning technique for MRs [Shivkumar 2024]. Through data acquisition from sensors, the MR selects actions based on its observations, enabling it to make informed choices over time. Researchers often opt for LiDAR sensors to collect environmental data for DRL tasks, yet LiDAR's limitations in detecting low obstacles and susceptibility to signal interference pose challenges [Maulana 2018]. Consequently, integrating multiple sensors like GPS and IMU imposes a computational burden on the MR's hardware [Dang 2023d]. Alternatively, some researchers solely utilize a camera for the MR, although this approach burdens the hardware due to the algorithm's reliance on full-image states. In summary, this paper aims to implement DRL on the MR equipped with a stereo camera, offering the following contributions:

- Utilization of the YOLO-v8 model for obstacle detection, followed by depth estimation to extract relevant pixels from the camera image as the MR's state.

- Development of a reward system and actions to guide the MR towards the goal.

2 PROPOSED METHOD

2.1 MR's Environment

DRL comprises six fundamental elements: agent, environment, observation, state, neural network, action, and reward. The agent, in this context, pertains to a MR, serving as the principal entity engaging with the environment and executing actions. The environment represents the framework within which the agent functions, offering observations, states, and responses to actions. Observation denotes the visual data captured by the agent through its camera. Following processing, this visual data is transformed into essential numerical values, collectively referred to as the state.

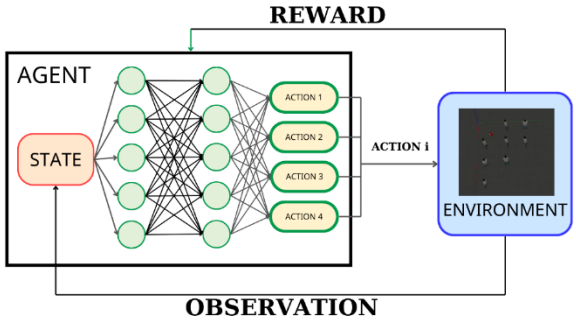


Fig. 1. Operating procedures of reinforcement learning.

A neural network is comprised of nodes within a layer, where the input corresponds to the values present in the state, and the output signifies the likelihood of different actions taking place. In this context, the actions of the MR entail navigating within the initial spatial configuration. The collaborative operation of elements, utilizing past actions to generate novel actions, is formally referred to as a Markov Decision Process (MDP) as illustrated in Fig. 1 [Adjei 2024].

2.1.1 Observation and States

The state of the MR is expressed in Eq. (1):

$$\text{state} = \left\{ s_{ob}, s_{pos} \right\}, \tag{1}$$

where s_{ob} is the distance from MR to obstacles; s_{pos} is the position relative to the destination.

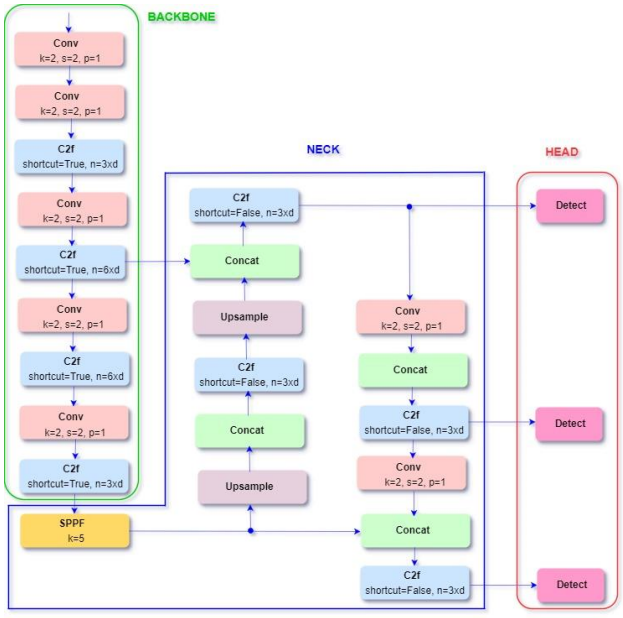


Fig. 2. YOLO-v8's architecture [Reis 2023].

a). The obstacles state

The distance between the MR and the objects situated in its frontal direction is determined through the utilization of a depth camera. Initially, the RGB images acquired by the camera will undergo processing by the YOLO-v8 model (see Fig. 2) for the purpose of recognizing obstacles and ascertaining the pixel coordinates corresponding to their centers. Therefore, the object's center coordinates helps to minimize the computational data and supports the path point search algorithm to satisfy the award functions of the proposed DRL method. Since the MR uses a depth camera, which consists of two small cameras with the same focal length. This setup helps the camera capture two slightly different images. Estimated distance to any pixel can be calculated as:

$$Z = \frac{f \times T}{d}, \tag{2}$$

where: f is the small camera's focal length; $d = x_l - x_r$ is the disparity in the horizontal coordinate of a pixel in one image relative to its corresponding pixel in the other image (see Fig. 3).

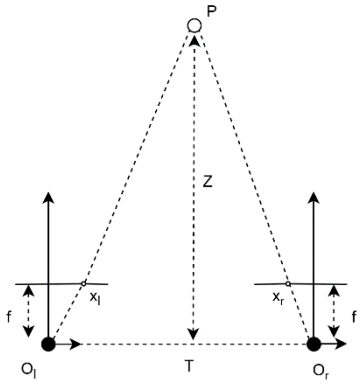


Fig. 3. Depth map estimation.

Then, a depth map is then created in Fig. 4:

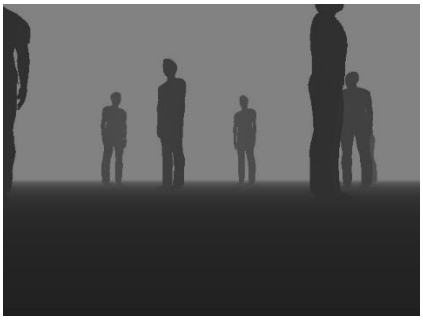


Fig. 4. Depth image.

The distance data between the MR and the obstacles is obtained by calculating the distance to the centroids of these pixels in Fig. 5. To streamline the procedure, the frame of the MR is segmented into five parts, allowing for the detection of the closest object within each segment, in Eq. (3).

$$s_{ob} = \left(d_1, d_2, d_3, d_4, d_4 \right), \tag{3}$$

where $d_1, d_2, d_3, d_4, \text{and } d_4$ are defined according to the object's position in Fig. 5.

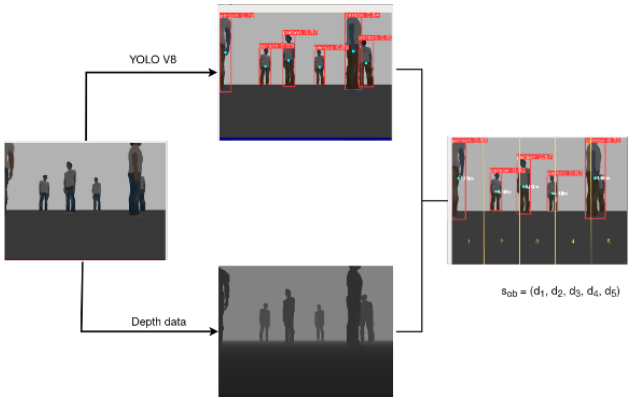


Fig. 5. Images from the camera are used as MR's state.

b). The position state

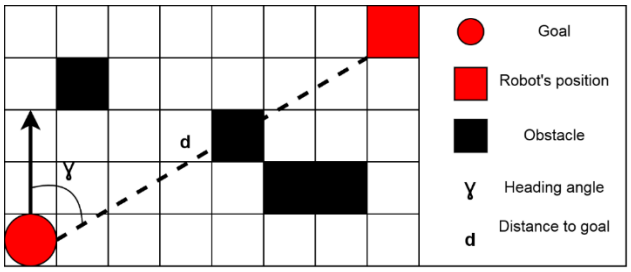


Fig. 6. Distance to goal and heading of the MR's state

The state of the MR's position is determined in Eq. 4.

$$s_{pos} = (d, \gamma), \tag{4}$$

where: d is distance to the destination and γ is its direction relative to the destination (heading angle).

2.1.2 Rewards

The MR's objective revolves around navigating towards the designated destination while circumventing obstacles, thus prompting an assessment of the appropriateness of each MR's action in accordance with these delineated criteria. Any movement by the MR that contributes to its proximity to the destination will warrant a reward. The attainment of the destination will culminate in the allocation of a substantial reward. Conversely, an encounter with an obstacle will precipitate a noteworthy penalty. Hence, the function award is constructed by pseudo code as follows:

FUNCTION REWARD

Input: $\{s_{ob}, s_{dis}\}$

Output: reward, done

done: False

reward: 0

If $s_{dis}[1] \le \text{HEADING\_THRESHOLD}$:

reward = $2^{\frac{s_{dis}[0]}{\text{ABSOLUTE\_DISTANCE}}}$

For i in s_{ob}:

If $i \le \text{ERROR\_DISTANCE}$:

reward = 100

done = True

If $s_{dis}[0] \le \text{GOAL\_THRESHOLD}$:

reward = 300

done = True

2.1.3 Actions

The mobility of a mobile MR is determined by its linear velocity v_l and angular velocity ω, in Eq. (5).

$$V = (v_1, \omega). \tag{5}$$

Next, the MR will be equipped with five predefined angular velocities in two directions to facilitate fast or slow rotations as needed, in Fig. 7. Meanwhile, the linear velocity will directly relate to the reward it receives, in Eq. (6).

$$\omega = (\omega_0, \omega_1, -\omega_1, \omega_2, -\omega_2). \tag{6}$$

Then, (5) and (6) yields to Eq. (7)

$$v_l = v \times \text{reward}. \tag{7}$$

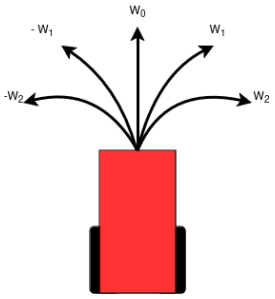


Fig. 7. MR consists of five fixed angular velocities.

Hence, this approach ensures that the MR begins its search at a high speed, gradually reduces it, and eventually halts upon reaching the destination.

2.2 Models

The model employed in this investigation is the Deep Neural Network. Comprising two layers, with each layer containing 32 nodes, this network takes the values of state as input and predicts the action of the MR as output. The optimization technique utilized is the Adam function [], accompanied by the Rectified Linear Unit (ReLU) activation function $y = \max(0, x)$, and the Mean Squared Error (MSE) is employed as the loss function.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2 \tag{8}$$

2.2.1 Traning Model

Once there are models to use, MRs will be trained T times, each time there will be a limit of steps ST according to the following algorithm "Train Robot" of Deep Q-Network :

TRAIN ROBOT

Input: The state

Output: Q-value (from which action will be selected)

Initialize **memory M**

Initialize neural network **policy_dqn**

Initialize neural network **target_dqn**

Initialize **best_reward, mini_batchsize**

Initialize **array rewards**

for episode i=1 to T:

Initialize **current_state**, **terminal**=False, **rewards**[i] =0, **current_step**=0

if **current_step** < ST and **terminal** = = False:

If mobile robot is moving

action random with function ε - greedy else

action argmax **policy_dqn** (with **input** = current_state)

new_state, **reward**, **terminal**=step(action)

#update information when MR action

rewards[i]+ =**reward** #update total reward in each step

current_step+=1

save (current_state, action, new_state, reward, terminal) to **memory M**

current_state = new_state

else: #if mobile robot crash obstacle or went to goal

if **rewards**[i] > **best_reward**:

update **best_reward** and **save policy_dqn**

if **length of memory** > **mini_batchsize**:

pick random **mini_batchsize** from **memory M** to **Function optimize Q-values**

after N step then copy from **policy_dqn** to **target_dqn**

reset current_state, terminal

The policy_dqn and the target_dqn are similar in structure but have different roles. The policy_dqn network is the main network used to select actions during training and practice in the environment. Therefore, policy_dqn is continuously updated to predict Q-values for actions based on the state and select the next action. The target_dqn network is used to stabilize the training process and provide target Q-values to calculate the loss function when training policy_dqn.

2.2.2 Optimizer Q-values

The optimizer function will calculate the current Q-values and target Q-values for each state and action in the minibatch. Then it will calculate the loss function between the two values according to the function MSE. Finally, it will update the parameters of the DQN model using Adam optimizer to minimize the value of the Loss function [Liu 2023]. The optimizer function is repeated in the following model in function optimize Q-values:

FUNCTION OPTIMIZE Q- VALUES

Input: Mini_batchsize

Output: Optimized model

Initilize current q_list, target_q_list

for **current_state, action, newstate, reward, terminal in mini_batchsize**:

if **terminal** = = True:

target = reward

else:

target=reward + discount_factor*max_target_dqn (with input=new_state)

current_q=policy_dqn(input=state)

current_q_list append **current_q**

target_q = target_dqn (input = state)

target_q[action] = target

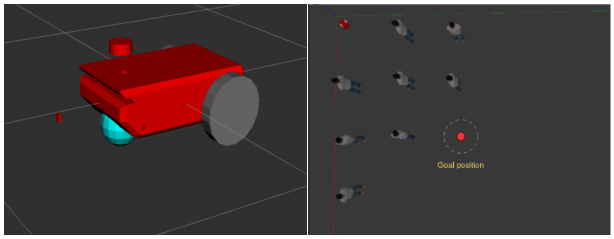
target_q_list append target_q

loss = MSE (current_q_list, target_q_list)

optimize the model by Adam

3 EXPERIMENTAL RESULTS AND DISCUSSION

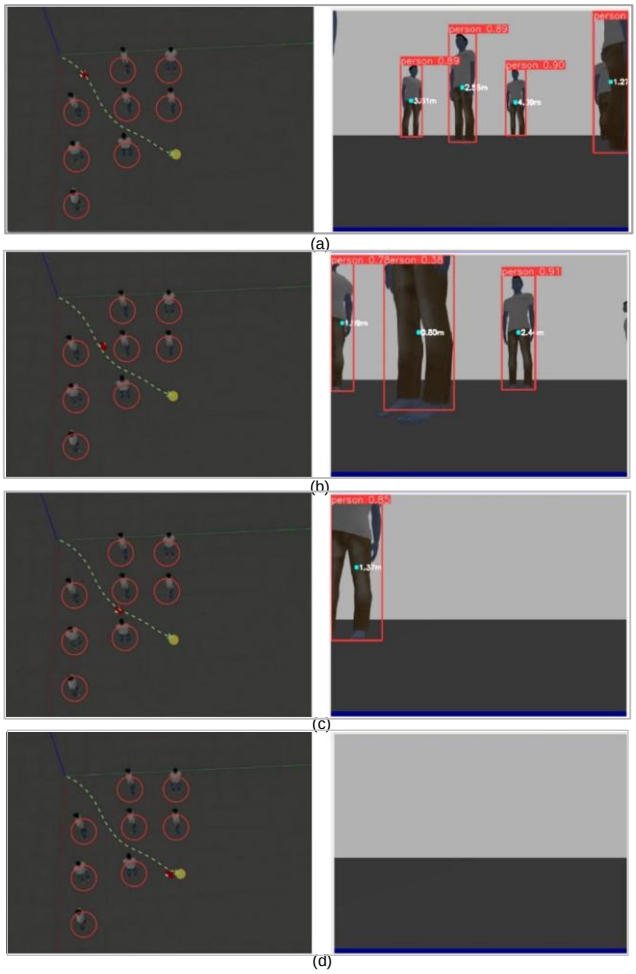
The following experiment is conducted to verify the correctness of the approach. Based on the experimental system configuration as follows: CPU: Intel Core™ i5-10300H 2.5Ghz, RAM: 16GB DDR4, GPU: NVIDIA GTX 1650Ti 4GB GDDR6, ROS2 and GAZEBO simulator are installed. A differ-entail drive MR equipped with a depth camera is spawned in a gazebo world with 6 simulated persons for training. For each of 1500 episodes, the MR will take 300 actions. The reward for reaching the goal at (5, 5) is 300, while colliding results in a penalty of -100.



(a) (b)

Fig. 8. Simulation environment with (a): three wheeled MR and (b): obstacles and goal position.

The MR's goal is to move to achieve the highest reward, which means finding the optimal path to the destination while avoiding obstacles, in Fig. 8. In Scenario 1, the MR will explore the map by moving randomly. In the following episodes, the MR will move closer to the destination with increasingly prioritized paths thanks to DRL.



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

Fig. 9. MR's feasible path planning in a simple environment.

After 1500 episodes of 8-hours-training, the simulation shows that the MR was able to create a feasible path to the goal while avoiding the obstacles. In Fig. 9, four snap shots of Fig. 9a to 9d illustrate that the MR has found the optimal way to move in a narrow simple environment with many obstacles. Continuously updating the state to determine the distance to the center of gravity of the object helps the object to move between obstacles more accurately. Fig. 10 shows that calculation process of reward through time-varying. Hence, the value of the reward is stable, meeting the optimal requirement at 1500 episodes.

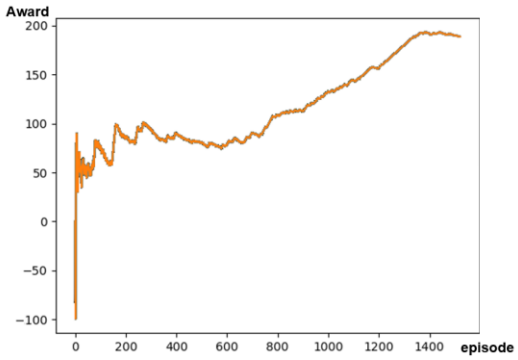


Fig. 10. Average reward through times.

Scenario 2 is created more complex with 20 simulated objects. For each episode, it will perform 300 actions, get a score of 300 after reaching the target position (7,10), or be -100 if colliding with the object. Four snap shots of Fig. 11a to 11d show that MR obtained the feasible path after 2000+ episodes. In Fig. 11, the MR can find its way to the destination after avoiding obstacles. Avoiding obstacles in a large environment makes the MR have a more complex path and takes more training time. However, the proposed method has still been achievable in moving to the destination in an unknown environment with many different obstacles.

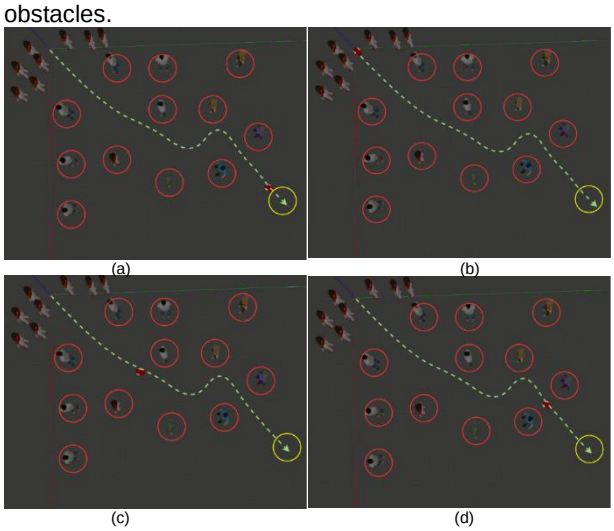


Fig. 11. The obtained feasible path after 2000+ episodes.

Although the environment becomes more complex and larger, the value of the reward has still been stable, meeting the optimal requirement at 2000 episodes.

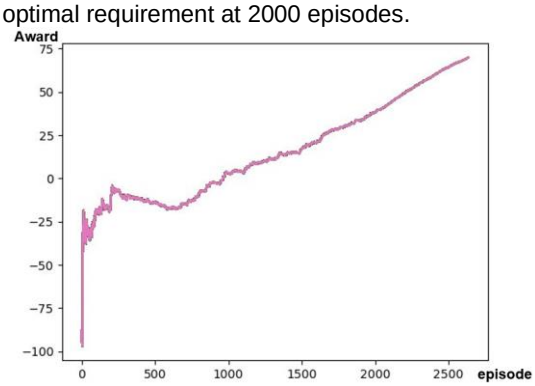


Fig. 12. Average reward through times in the larger environment.

Figs. 10 and 12 show the average rewards the MR earns after each episode. It's clear that these rewards increase over time, suggesting the MR is steadily advancing toward the goal, thereby improving its success rate in reaching it.

In the 6 x 6 (m) spatial environment, MR finds the way to the destination in Episode 189, taking about 1 hour. In the remaining environment, MR finds the way to the destination in Episode 382, taking about 2.2 hours. Hence, MR tries to find the optimal path between the obstacles in complex in 7 x 10 (m). Table 1 illustrates the feasible path after completing training.

Tab. 1: Detailed metrics in two environments

Environment	First success episode	Training time
6x6(m) having 8 obstacles	Episode 189	8 hours
7x10(m) having 12 obstacles	Episode 382	**3.5 hours**

In the comparison with [Prasuna 2024], although the DQN method is simple, easy to apply and combines well with YOLO-V8 using depth camera, the model has poor performance, leading to unstable results. Because, our proposed method is designed base on DRL combining with the object's center of gravity technique to reduce the traning parameters and data. After obtaining 2D image segmentation, optimal MR's path planning will be successfully.

4 CONCLUSIONS

The paper proposed a path planning and obstacle avoidance strategy for a camera-equipped MR. First, the image from the camera will be processed by YOLO-v8 to detect the positions of obstacles. This information will be combined with the depth map, selecting specific pixels as the state representation for the MR. Subsequently, a velocity function proportional to the reward will be applied to help the MR adjust its speed appropriately. Simulation in Gazebo and the graph of average rewards have demonstrated the validity of this approach. The MR's perception system is based on Deep Reinforcement Learning combining with YOLO v8. Therefore, the ability to find the optimal path in an unknown environment becomes efficient. In addition, the study also shows the ability to avoid obstacles with complex height obstacles by determining the distance to the center of the obstacle. However, from the research results, the training is still time-consuming, and the number of training sessions is still large. Therefore, improving the environment and model to help MRs find their way more efficiently is essential in developing MRs. Finally, the DRL method can be applied in changing environments, where there are many people and obstacles are not too large. Similar environments such as indoors or industrial can be suitable for applying this method.

5 ACKNOWLEDGMENTS

Lab-506: Computer Vision and Autonomous Mobile Robot (CVMR), VJIIST, HUST is gratefully acknowledged for providing work location, guidance, and expertise.

6 REFERENCES

[Adjei 2024] Adjei, P. et al. Safe Reinforcement Learning for Arm Manipulation with Constrained Markov Decision Process. Robotics, 2024, Vol.13, No.4, pp 63.

[Alshammrei 2022] Alshammrei, S. et al. Improved Dijkstra Algorithm for Mobile Robot Path Planning and Obstacle Avoidance. Computers, Materials & Continua, 2022, Vol.72, No.3, pp 5939-5954.

[Dang 2023a] Dang, T.V. and Bui, N.T. Multi-Scale Fully Convolutional Network-Based Semantic Segmentation for Mobile Robot Navigation. Electronics, 2023, Vol.12, No.3, pp 533.

[Dang 2023b] Dang, T.V. and Bui, N.T. Obstacle Avoidance Strategy for Mobile Robot Based on Monocular Camera. Electronics, 2023, Vol.12, No.8., pp 1932.

[Dang 2023c] Dang, T.V. Research and design of a path planning using an improved RRT * algorithm for an autonomous mobile robot. MM Science Journal, 2023, Vol.10, pp 6712-6716.

[Dang 2023d] Dang, T.V., Tran, D.M.C., and Tan, P.X. IRDC-Net: Lightweight Semantic Segmentation Network Based on Monocular Camera for Mobile Robot Navigation. Sensors, 2023, Vol.23, No.15, pp 6907.

[Li 2024] Li, X., Li, G., Bian, Z. Research on Autonomous Vehicle Path Planning Algorithm Based on Improved RRT* Algorithm and Artificial Potential Field Method. Sensors, 2024, Vol.24, No.12, pp 3899.

[Liu 2021] Liu, L.S. et al. Path Planning for Smart Car Based on Dijkstra Algorithm and Dynamic Window Approach. Wireless Communications and Mobile Computing, 2021, Vol.2021, No.4, pp 1-12.

[Liu 2023] Liu, M, Yao, D, Liu, Z, Guo J, and Chen, J. An Improved Adam Optimization Algorithm Combining Adaptive Coefficients and Composite Gradients Based on Randomized Block Coordinate Descent. Computational Intelligence and Neuroscience. 2023, Vol.5, pp 4765891.

[Maulana 2018] Maulana, I., Rasdina, A., and Priramadhi, R.A. Lidar Application for Mapping and Robot Navigation on Closed Environment. Journal of Measurements Electronics Communications and Systems, 2018, Vol.4, No.1, pp 767-782.

[Prasuna 2024] Prasuna, R. G., Potturu, S. R. Deep reinforcement learning in mobile robotics–a concise review. Multimedia Tools and Applications, 1-22.

[Reis 2023] Reis, D., Kupec, J., Hong J., and Daoudi, A. Real-Time Flying Object Detection with YOLOv8. Preprint, 2023.

[Shivkumar 2024] Shivkumar, S., Amudha, J., and Nippun-Kumaar, A.A. Federated deep reinforcement learning for mobile robot navigation. Journal of Intelligent & Fuzzy Systems, 2024, Vo. 47, pp 1-16.