Deep Reinforcement Learning Based Mobile Robot Navigation in Unknown Indoor Environments

1st Koray Ozdemir
Dept. of Computer Engineering
Institute of Science
Yalova University
Yalova, Turkey
korayozdemir34@gmail.com

2nd Adem Tuncer
Dept. of Computer Engineering
Engineering Faculty
Yalova University
Yalova, Turkey
adem.tuncer@yalova.edu.tr

***Abstract*—The importance of autonomous robots has been increasing day by day with the development of technology. Difficulties in performing many tasks such as target recognition, navigation, and obstacle avoidance autonomously by mobile robots are problems that must be overcome. In recent years, the use of deep reinforcement learning algorithms in robot navigation has been increasing. One of the most important reasons why deep reinforcement learning is preferred over traditional algorithms is that robots can learn the environments by themselves without any prior knowledge or map in environments with obstacles. This study proposes a navigation system based on the dueling deep Q network algorithm, which is one of the deep reinforcement learning algorithms, for a mobile robot in an unknown environment to reach its target by avoiding obstacles. In the study, a 2D laser sensor and an RGB-D camera has been used so that the mobile robot can detect and recognize the static and dynamic obstacles in front of itself, and its surroundings. Robot Operating System (ROS) and Gazebo simulator have been used to model the robot and environment. The experiment results show that the mobile robot can reach its targets by avoiding static and dynamic obstacles in unknown environments.**

Keywords—deep reinforcement learning, mobile robot, navigation, dueling deep Q network, ROS, Gazebo

I. INTRODUCTION

One of the most important competencies expected from mobile robots is navigation skills. In order for a mobile robot to be considered successful in the navigation process, it has to reach the given targets as soon as possible and avoiding obstacles [1]. There are various algorithms and methods in the literature used for mobile robots to navigate more efficiently [2]. In addition to traditional navigation algorithms, deep learning-based approaches such as deep neural networks (DNNs) have been used widely in recent years. However, both the large amount of data required for the training processes and the difficulties in interpreting the data have been made it necessary to use new methods and algorithms. Among these new approaches, reinforcement learning (RL) algorithms come forward.

The use of RL algorithms has been seen in many different areas in recent years. It has been observed that these algorithms obtain successful results, especially in studies conducted on video games. Studies on the use of RL in mobile robot navigation have increased in parallel with the developments in hardware technologies.

The concept of Deep Reinforcement Learning (DRL) came up with the combined use of DNN and RL algorithms and has been successfully applied in solving mobile robot navigation problems. Mnih et al. [3] tested 49 Atari 2600 games using DQN with raw images. The game images used as states were 80×80 in size and their numbers were four. One of them was the current state and the other three consist of previous images. The different aspect of the study was the use of convolutional neural networks. In addition, experience replay is one of the important innovations of the study. Experiences during a certain iteration were recorded in the database. Then, during the training phase, learning occurred by taking uniform examples from these experiences. They stated that 75% success was achieved in the process of playing Atari 2600 games. Surmann et al. [4] used Asynchronous Advantage Actor-Critic Network (GA3C), one of the Deep Q Learning (DQN) algorithms, for the robot to reach the given targets while avoiding obstacles. They tested their models in a simulation environment using Turtlebot 2 and stated that they got successful results. In their study, Quan et al. [5] presented a model based on DQN and recursive neural networks. A grid-map-based two-dimensional environment, a three-dimensional simulation environment, and a physical environment were used to test the proposed model. They achieved successful test results in finding the target and shortening the arrival times of the robot. Xie et al. [6] developed a model that enables a mobile robot to avoid obstacles in both simulation environment and physical environment using a single RGB camera. They presented a dueling double Q network algorithm, which combines double Q network and dueling Q network algorithms.

In recent studies, camera and sensor data are used for robots to perceive their environment and to perform their actions according to the observations obtained. Some studies are only sensor-based [7], while others are performed using camera images [6]. Hybrid approaches that use both camera and sensor data together are also preferred, as we have used in this study [4]. In this study, we propose a Dueling DQN using a 2D laser sensor and RGB-D camera for mobile robot navigation. The reason for such a preference is that the laser sensor gives more accurate results than deep cameras in calculating the distance of objects to the robot. The mobile robot is expected to reach the targets as soon as possible and avoid obstacles in different unknown simulation environments. In recent studies, static obstacles are widely used. However, in the real world, there are also dynamic obstacles in the environments, such as humans or animals. Both static and dynamic obstacles were used in simulation studies, and thus, it has been shown that the proposed model could produce a solution to the real-world problem.

The contributions of this paper are the following:

- A Dueling DQN model has been applied using the 2D laser sensor and RGB-D camera. The proposed model has been trained with only the data from the laser sensor without the need for image processing.

- Since the 2D laser sensor only gives 2-dimensional data, the RGB-D camera data has been converted into laser

data and used in the training of the network so that the robot has a 3-dimensional view.

• Simple and complex simulation environments with static and dynamic obstacles have been used to create an environment similar to the real world and to show the performance of the model.

• Under the specified conditions, the mobile robot is able to reach the specified targets using the shortest path.

II. Background

A. Reinforcement Learning

RL is a field of machine learning affected by behaviorism. Learning occurs as a result of successful and unsuccessful attempts by the structure called the agent using the previous experiences. The agent is expected to find the most appropriate action sequence on its own without being told what actions it should take [8]. RL differs from both supervised and unsupervised learning in terms of features such as not providing a certain data set beforehand, being based on a reward and punishment system, and aiming to maximize reward. It has a similar approach to human learning. Algorithms learn a policy that determines how they should behave in a given environment. Each action also affects the environment. The environment guides the agent with the help of the rewards.

The basic components of RL can be listed as agent, environment, policy, reward, and value function [9]. The agent is defined as the algorithm that performs actions. The environment is where the agent acts. The reward indicates how good or bad the actions by the agent are in the short term. The reward, R obtained depending on the action, A performed in a certain situation, S is shown in (1).

$$R_s^a = E[R_{t+1}|S_t = s, A_t = a] \quad (1)$$

The value function, V indicates the largest reward amount that the agent expects to collect from start to finish. The values are calculated using (2).

$$V_\pi(s) = E_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots | S_t = s] \quad (2)$$

where γ is a discount factor which is a range of [0, 1]. It is used to determine the importance of future rewards. If set to 0, only the first reward will be used, and others will have no effect. If it is set to 1, all rewards will have equal importance.

Policy, π are reference charts or functions that guide agents in the performs of their actions. It determines the behavior of the agent at a specified time. The main purpose of RL is to find the most appropriate policy to solve the problem.

B. Q Learning

Q learning is an off-policy RL algorithm that does not need any environmental model [10]. It is based on the Bellman Equation. It is classified as an off-policy because the policy used to carry out the actions and the updated policy are different. The main aim is to learn the policy that maximizes the total reward. When the algorithm is run, a matrix in the form of [action, state] called Q-table is created and initial values are set to 0. At each iteration, Q values are updated and at the end of the algorithm, the best policy that provides the solution is obtained. The updating of the Q-table is shown in (3).

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r(s,a) + \gamma \max_{a'} Q(s',a') - Q(s,a)] \quad (3)$$

where s is the state or observation, a is the action the agent takes, r is the reward, t is the time step, α is the learning rate, γ is the discount factor. The discount factor causes rewards to lose their value over time, so more immediate rewards have higher values.

C. Deep Q Network

When the number of state-action pairs is large, it becomes difficult to show the policy parameters in a table. Therefore, it is not appropriate to use Q-tables in solving complex and large scale problems.

DQN has been developed with the combined use of DNN and RL algorithms. DNN is used to obtain appropriate Q values in DQN. In addition, the agent's experiences are stored using experience replay memory. The agent is expected to learn from experiences by taking uniform samples from the stored experiences using (4). It has been stated that the use of experience replay is successful in providing stability [11].

$$e_t = (s_t, a_t, r_t, s_{t+1}) \quad (4)$$

where e_t is the experience stored at time t, s_t is the state at time t, a_t is the action at time t, r_t is the reward at time t, s_{t+1} is the state at time $t+1$.

D. Dueling Q Network

Dueling DQN, introduced by Wang et al. [12], uses the same network structure as DQN. The fully-connected layer is divided into two parts, unlike DQN. In the first part, the value V of the state s is estimated and shown as $V(s)$. In the second part, the advantage A of the action a performed in a certain state s is estimated and shown as $A(s,a)$. Thus, some unnecessary actions are avoided and the performance increases. The Q values obtained are shown in (5).

$$Q^\Pi(s,a) = V^\Pi(s) + A^\Pi(s,a) \quad (5)$$

III. Dueling Q Network Based Mobile Robot Navigation

In this section, firstly the current problem situation is explained and then the observations, action space, and reward and punishment system used in the learning phase are introduced. Finally, the proposed Dueling DQN model is presented in detail.

A. Problem Statement

The aim of the developed model is that the mobile robot can reach the given targets as soon as possible by avoiding the obstacles in simple and complex unknown environments with static and dynamic obstacles.

In our study, we used Turtlebot3 mobile robot and two different environments designed in Gazebo simulator integrated with Robot Operating System (ROS). In the training process of the model, the data obtained from the robot's 2D laser sensor and RGB-D camera were used, and thus, the distance between the mobile robot and the objects was determined. Then, actions that enable the mobile robot to move in different directions were obtained. When the robot hits obstacles, it receives penalties, and when it reaches the targets, it is rewarded. The aim of the learning process is to

maximize reward on a policy Π(s). Dueling DQN is used to obtain the Q values of each action.

B. Learning Setup

• Observations: We used a 2D laser sensor that can obtain 360 degrees of data and an RGB-D camera to obtain a 3-dimensional view. 3D point cloud data from the camera was converted into 2D laser data. Point cloud to laser scan package that is one of the packages in ROS was used for this purpose. The obtained data then were filtered according to the height (Z-axis) of the robot. In addition, the distance and angle of the robot to the target were used in the training of the network.

• Action space: Linear and angular velocities can be determined for Turtlebot3. Two different linear velocities (0.2, 0.4) m/s and five different angular velocities $\left(\frac{\Pi}{4}, \frac{\Pi}{8}, 0, -\frac{\Pi}{8}, -\frac{\Pi}{4}\right)$ rad/s were used in our study. Our agent can choose seven different actions. Since the camera is only located on the front of the robot, it is limited to forward movement only.

• Reward design: The aim of the agent is to reach the target as soon as possible avoiding obstacles. For this purpose, a reward and punishment system (6) is used. The robot receives a negative reward when it hits obstacles and a positive reward when it reaches the target. In addition, the agent receives rewards in each time step depending on its speed and heading to the target. Thus, the mobile robot is aimed to reach the target as soon as possible.

$$r = \begin{cases} +2 & Target \\ -1.5 & Collision \\ v\cos(\theta) - 0.01 & Other \end{cases} \quad (6)$$

where r is the reward, v is the linear velocity and θ is the heading to the target.

C. Model and Network Architecture

In the study, obtaining the Q values and choosing the actions to perform were done using the Dueling DQN algorithm. As input layer, we created a 39 dimensional vector which contains 37 laser range data, the distance and angle to the target. After the input layer, a fully connected layer consisting of 128 neurons was used. Then, two fully connected layers of 64 neurons were used to calculate the value and advantage separately. As output layer, 1 output for value and 7 outputs for advantage values depending on the number of actions were produced. In the last stage, Q values were obtained by using these values. Details of the proposed Dueling DQN model are shown in Table 1 and the architecture of the model is shown in Fig. 1 (b). The standard DQN was also used in the same simulation environments to demonstrate the performance of the proposed model. Details of the standard DQN model are shown in Table 2, and the architecture of the model is shown in Fig. 1 (a).

TABLE I. DETAILS OF THE DUELING DQN MODEL

Layer Name	No of Neurons	Activation Type
Input	39	-
Shared FC	128	ReLU
FC1 for value	64	ReLU
FC1 for advantage	64	ReLU
FC2 for value	1	Linear
FC2 for advantage	7	Linear
Output	7	-

TABLE II. DETAILS OF THE STANDARD DQN MODEL

Layer Name	Number of Neurons	Activation Type
Input	39	-
FC1	128	ReLU
FC2	128	ReLU
Output	7	Linear

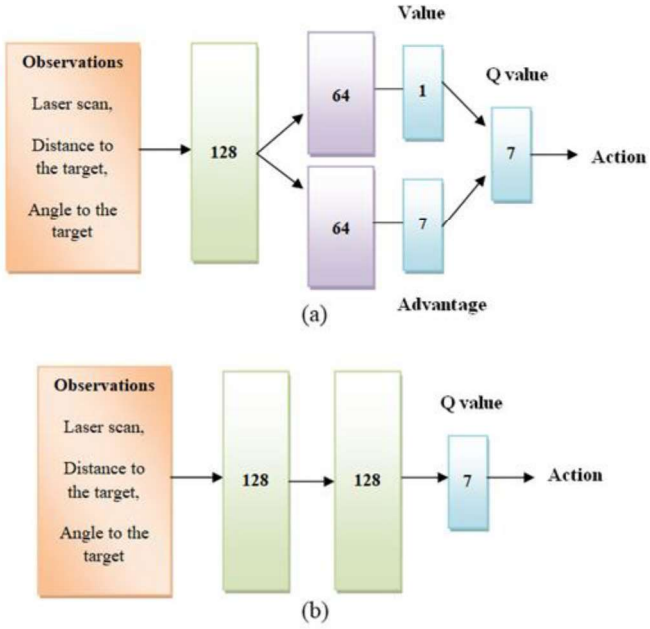


Fig. 1. Network structures (a) standard DQN (b) Dueling DQN

The training process begins with the initialization of the environment and experience replay memory used for data storage. Then, the robot is supposed to reach the surrounding targets by avoiding the obstacles. While there are static obstacles in the simple environment, in addition to static obstacles, dynamic obstacles have also been used in the complex environment. Maximum episode steps have been defined for each episode. When the maximum step is reached, it is passed to the other episode. The locations of the targets are changed in each new episode. If the robot hits any obstacle or reaches the target, the episode ends and the next episode starts. An epsilon value is used to determine the probability of choosing a random action, and an epsilon decay value is used to reduce the probability. The data stored in the experience replay memory is taken as mini batches. Table 3 shows the hyper-parameters and their values used in the training process of the model.

IV. EXPERIMENTAL RESULTS

Our model has been trained on two different environments designed in Gazebo simulator. In the simple environment, there are 8 box-shaped obstacles placed regularly inside the

room. The robot, which starts its movement in the middle of the room, is supposed to reach the target that is changed its location in each episode as soon as possible by avoiding the obstacles.

TABLE III. HYPER-PARAMETERS AND VALUES FOR THE MODEL

Hyper-parameter	Value
Maximum episode step	5000
Epsilon	1
Epsilon decay	0.99
Epsilon minimum	0.01
Batch size	64
Memory	1000000

The complex environment is aimed to resemble a real room. For this purpose, household items have been also placed in the room and the robot is supposed to reach the specified targets in the same way. Besides, a box-shaped dynamic obstacle has been added to the complex environment. As mentioned in the reward design section, the robot receives a negative reward when it hits the obstacles, and a positive reward when it reaches the target positions. Fig. 2 and Fig. 3 show the view of the simple environment from different angles, while Fig. 4 and Fig. 5 show the complex environment.

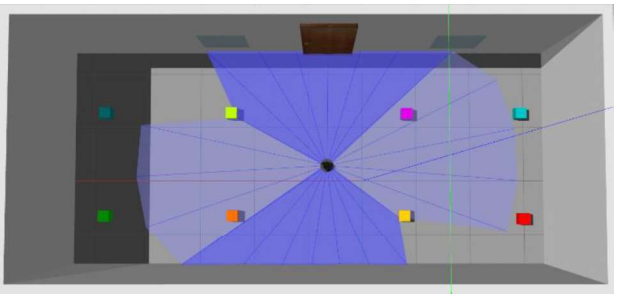


Fig. 2. Top view of the simple environment

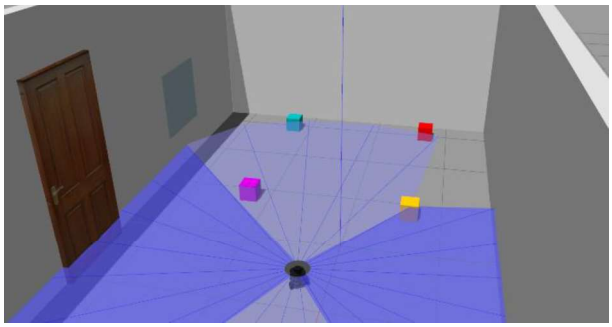


Fig. 3. Side view of the simple environment

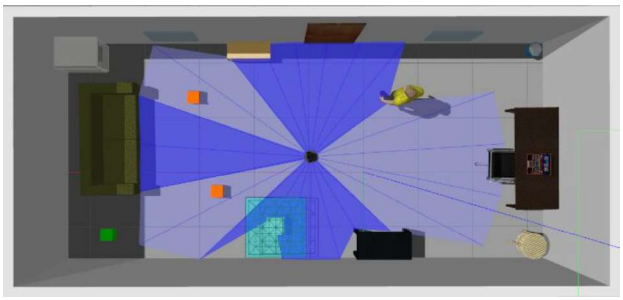


Fig. 4. Top view of the complex environment

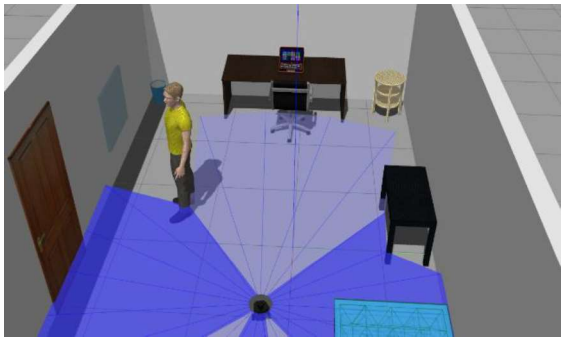


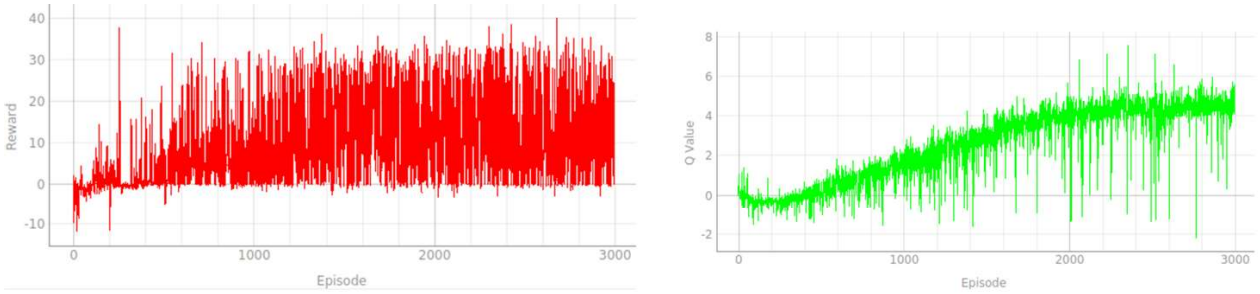
Fig. 5. Side view of the complex environment

The proposed Dueling DQN model has been trained on both environments. As a result of the training process, the total reward and average Q values have been obtained. Fig. 6 and Fig. 7 show the results of the Dueling DQN for the simple environment and the complex environment, respectively. Fig. 8 and Fig. 9 show the results of the standard DQN model for the simple environment and the complex environment, respectively. When the total amount of rewards and average Q values have been evaluated, it is seen that the proposed model gives more successful results than the standard DQN algorithm.

V. CONCLUSION

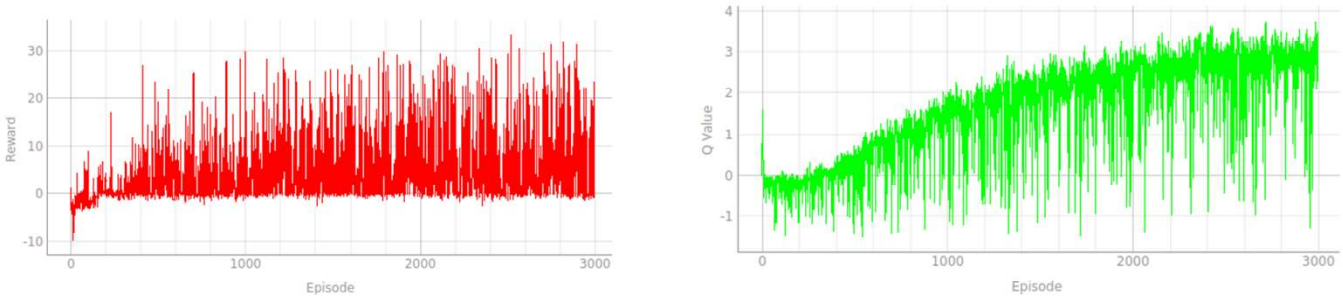
Autonomous mobile robots are expected to perform tasks such as target recognition, navigation, and obstacle avoidance on themselves, without human assistance. There are various algorithms developed to fulfill these tasks, and new algorithms and methods continue to be developed. In this study, a model based on the Dueling DQN algorithm has been proposed for the autonomous navigation of a mobile robot in an unknown environment. In addition, the standard DQN has also been used in the study and a performance comparison has been made with proposed model.

Robot Operating System (ROS) and Gazebo simulator have been used to model the robot and its environment. Experimental results show that the mobile robot can reach its targets in environments with previously unknown in which static and dynamic obstacles using both Dueling DQN and standard DQN. On the other hand, it is seen Dueling DQN achieves better results than standard DNQ.



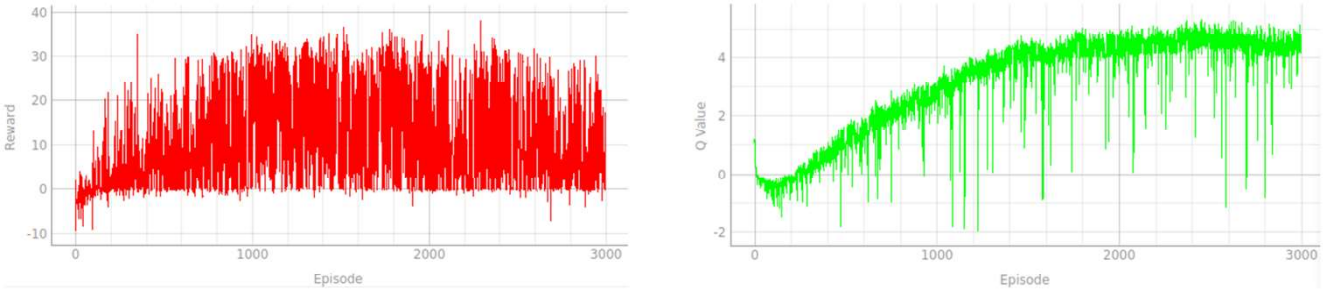
(a) (b)

Fig. 6. Simple environment training results for the Dueling DQN model



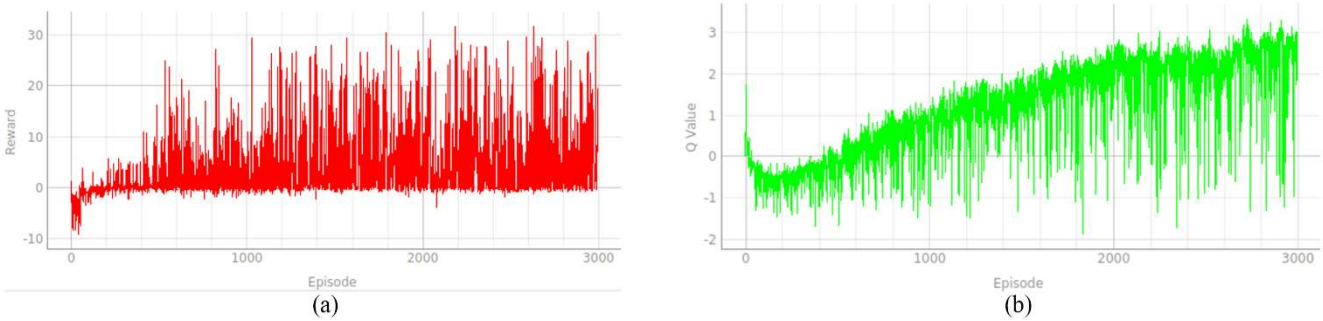
(a) (b)

Fig. 7. Complex environment training results for the Dueling DQN model



(a) (b)

Fig. 8. Simple environment training results for the standard DQN model



(a) (b)

Fig. 9. Complex environment training results for the standard DQN model

ACKNOWLEDGMENT

This study was supported by the Research Fund of Yalova University (Project Number: 2020/YL/0014).

REFERENCES

[1] A. Tuncer, M. Yildirim, and K. Erkan, "A hybrid implementation of genetic algorithm for path planning of mobile robots on FPGA," In Computer and Information Sciences III, Springer, London, pp. 459–465, 2013.

[2] B.K. Patle, A. Pandey, D.R.K. Parhi, and A. Jagadeesh, "A review: On path planning strategies for navigation of mobile robot," Defence Technology, vol. 15(4), 582–606, 2019

[3] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," arXiv preprint arXiv:1312.5602, 2013.

[4] H. Surmann, C. Jestel, R. Marchel, F. Musberg, H. Elhadj, and M. Ardani, "Deep Reinforcement learning for real autonomous mobile robot navigation in indoor environments," arXiv preprint arXiv:2005.13857, 2020.

[5] H. Quan, Y. Li, and Y. Zhang, "A novel mobile robot navigation method based on deep reinforcement learning," International Journal of Advanced Robotic Systems, vol. 17(3), pp. 1–11, 2020.

[6] L. Xie, S. Wang, A. Markham, and N. Trigoni, "Towards monocular vision based obstacle avoidance through deep reinforcement learning," arXiv preprint arXiv:1706.09829, 2017.

[7] S.H. Han, H.J. Choi, P. Benz, and J. Loaiciga, "Sensor-based mobile robot navigation via deep reinforcement learning," In 2018 IEEE International Conference on Big Data and Smart Computing (BigComp), pp. 147–154, IEEE, 2018.

[8] S. Russell and P. Norvig, "Artificial intelligence: a modern approach," Prentice Hall, 2009

[9] R.S. Sutton and A.G. Barto, "Reinforcement learning: An introduction," MIT press, 2018.

[10] C.J.C.H Watkins, "Learning from delayed rewards,", Ph.D. thesis, Cambridge University, 1989

[11] L.J. Lin, "Self-improving reactive agents based on reinforcement learning, planning and teaching," Machine learning, vol. 8(3-4), pp. 293–321, 1992.

[12] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," In International conference on machine learning, PMLR, pp. 1995–2003, 2016.