SSRG International Journal of Electronics and Communication Engineering
ISSN: 2348 – 8549 /doi:10.14445/23488549/IJECE-V8I1P104
Volume 8 Issue 1, 18-23, January 2021
© 2021 Seventh Sense Research Group®

Control Multi-Directional Mobile Robot Based on DDPG Intelligent Algorithm Application

Chau Thanh Phuong*

* *Faculty of Electronic Engineering Technology, University of Economics - Technology for Industries, Viet Nam*

Abstract *- The article, the implementation of the Deep Deterministic Policy Gradient algorithm on the Gazebo model and the reality of a multi-directional mobile robot, has been studied and applied. The empirical studies' goal is to make the multi-directional mobile robot learn the best possible action to travel in real-world environments when facing obstacles. When the robot moves in an environment with obstacles, the robot will automatically control to avoid these obstacles. Then, the more time it can remain within a specific limit, the more the reward is accumulated, and therefore the better results will be achieved. The author has performed various tests with many metamorphic parameters and proved that the DDPG algorithm is more efficient than algorithms like Q-learning, Machine learning, deep Q-network, etc. The research results will be the basis for designing and establishing control algorithms for present and future mobile multi-directional robots and industrial robots for application in programming engineering and home automation control industrial production machines.*

Keywords - *Multi-directional mobile robots, artificial intelligence, Obstacle robots, DDPG algorithm, autonomous navigation, reinforcement learning.*

I. INTRODUCTION

Our world is constantly evolving and changing. Underwent three industrial revolutions since 1784 was the first revolution, 1090 was the 2nd revolution, 2013 was the third revolution with computer and industrial automation robot emitted So far, the fourth industrial revolution 2020 has come into being: the strong development of digital technology, biotechnology, and physics, and also evident in the fields of artificial intelligence, Internet of Things (IoT), robot control field is receiving much attention. A lot has changed since the introduction of robots in 1917. Even Asimov's famous ideas about robots and his three famous robots' laws seem to be far behind us today. Today, machines are present in our lives, supporting us in our daily activities, [1]-[5]. Some scientists understand robotics as an applied science born of the "marriage" between computer science and machine tools, so this tool can now process and manage information. Logically and automatically without human assistance, worker replacement, tireless, non-strike, and 100% operational. However, reality shows that we are still very far from such a technology. Although trivial to humans, perceiving the environment (feel) and making decisions (to act) is a very difficult task for the computer. Therefore, Artificial Intelligence (AI) is needed for multi-directional mobile robots to solve such problems [3, 4, 5, 7, 9, 14].

The Q - learning, SARSA are all based methods, meaning that if you want to find the optimal policy, you must build that value function to estimate the value for each action in the entire workspace. Then, to find the optimal policy, the algorithm picks out the action with the largest Q value in the entire action space. Although the important approach is reinforcement learning, the goal of reinforcement learning is policy resolution, so it seems that algorithms of this type are indirectly reaching their destination. In practice, it is often combined with policy-based methods to solve complex problems with large discrete action spaces or continuous action spaces [6, 9, 21, 22]. Deep Deterministic Policy Gradient (DDPG) is a modern and typical algorithm for this problem. This algorithm combines two perfect approaches between two value-based and policy-based methods to build smart enough agents for complex problems in the continuous action space control the multi-directional mobile robot. Although the training takes thousands of episodes to be repeated repeatedly, each test episode's policy decision will take a very short time. Based on a real-world case of the mobile robot's path, the models for robot control problems are set up as a training and testing environment. To validate the proposed algorithm's energy-saving and real-time functionality, four experiments were designed and conducted. The results show that the proposed algorithm has real-time performance and significant percent energy savings under random noise during robot control [8, 11, 15, 20, 24].

In this paper, the author presents a modern and intelligent control DDPG algorithm based on controlling robots with complex environments in a continuous action space of mobile robots, which is zero. Determine when the robot is teleported. Test studies are performed on automated navigation missions for multi-directional mobile robots. The author also introduces the neural network structure to generalize and approximate all states' values based on the DDPG algorithm. This is an off-policy intensive learning algorithm, online learning, and based on Actor-Critic smart network structure. Tests were conducted on the Gazebo emulator using a high-profile computer with a multi-directional mobile robot, with its open-source extension to perform automated navigation tasks for mobile robots, in

This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/)

reality, there are complex environments that overcome obstacles, [5, 7, 8, 12, 13, 25, 27].

II. BUILDING CONTROL MODELS FOR MULTI - DIRECTIONAL MOBILE ROBOTS

Consider a mobile robot as figure 1 with three wheels, two of which are the driving fixed standard wheels located at the back of the chassis and one is the front caster wheel, which can make the mobile robot keep balance and doesn't exert any motion constraint to the mobile robot. Two coordinate frames can be used to describe the motion of the mobile robot. One is the global coordinate frame (X, Y), which is fixed in the world, and the other is the local coordinate frame (X_l, Y_l) which is fixed on the mobile robot. The angle between the two coordinate frames is denoted as θ. The robot's motion will be defined for the navigation stack. As the global coordinate is chosen in figure 1, it is clear that the robot's velocity contains three components: the linear velocity along the OX axis and the OY axis and angular speed.

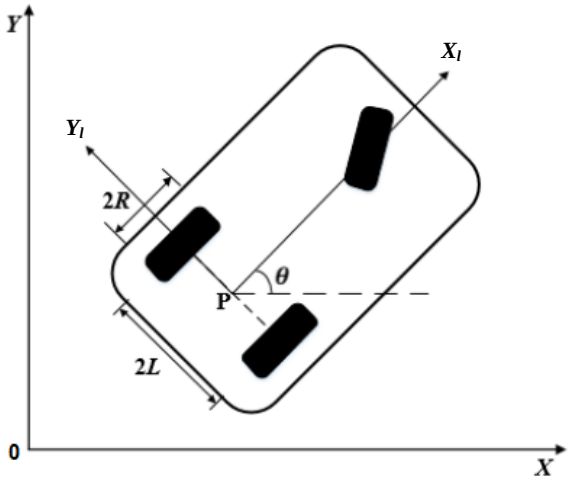

Fig 1: The model multi-directional mobile robot

To specify the multi-directional mobile robot's position, the P-coordinate is selected on the robot's frame as its control center position reference point. P is positioned by the coordinates (x, y) in the global frame of the robot's entire control environment. To describe the motion of the mobile robot as component movements, it is necessary to define the motion mapping along the axes of the spherical frame with the motion along the local frame's coordinate axes, which The entire control environment for the robot. This mapping is represented as the following expression:

$$\begin{bmatrix} \dot{x}_l \\ \dot{y}_l \\ \dot{\theta}_l \end{bmatrix} = \begin{bmatrix} cos\theta & sin\theta & 0 \\ -sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} \quad (1)$$

By considering the limit of standard wheel slip, $\dot{y}_l = 0$, meaning that the wheel cannot slide orthogonal to the wheel plane, then we can obtain:

$$-\dot{x}sin\theta + \dot{y}cos\theta = 0 \quad (2)$$

Denoting the forward velocity $\dot{x}_l$ of the multi-directional mobile robot as v and the rotation speed θ as w, the kinematics of the multi-directional mobile robot becomes:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} cos\theta & 0 \\ sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix} \quad (3)$$

The forward velocity and rotation speed of the multi-directional mobile robot have the following relationships with the linear velocities of the two wheels:

$$v = \frac{1}{2}(v_l + v_r), \qquad w = \frac{v_l - v_r}{2L} \quad (4)$$

If we consider the linear velocities at the wheels and the two qualified wheels' angular velocities, we can get: $v_l = w_l R$ and $v_r = w_r R$. Then, we can represent the angular velocities of the two standard wheels according to the forward velocity and rotation speed of the multi-directional mobile robot would be:

$$w_l = \frac{v + wL}{R}, \quad w_r = \frac{v - wL}{R} \quad (5)$$

By considering the acceleration of the multi-directional mobile robot in its local working coordinate limit, then the dynamic model can be expressed as [12]:

$$\begin{bmatrix} \dot{v} \\ \dot{w} \end{bmatrix} = \begin{bmatrix} \frac{1}{Rm} & \frac{1}{Rm} \\ \frac{L}{RI} & \frac{L}{RI} \end{bmatrix} \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \quad (6)$$

where m and I are the mass and inertia of the mobile robot, respectively; R is the radius of the two fixed standard wheels; L is the half of the distance between the two fixed standard wheels; $\tau = [\tau_1 \ \tau_2]^T$ is the input torque vector from the motors exerted to the two fixed standard wheels, [1, 4, 8, 29].

The goal is to teach multi-directional mobile robots to track and follow certain trajectories within the right spaces and work environments $e_{x,k} \approx 0$, $e_{y,k} \approx 0$, $e_{\theta,k} \approx 0$.

III. RESEARCH AND APPLICATION OF DDPG ALGORITHM TO CONTROL MULTI-DIRECTIONAL MOBILE ROBOT

A. The DDPG algorithm learning method

DDPG algorithm is built similar to the idea of Double Deep Q-Network. It is a model with reinforcement learning, onlone learning, and an off-policy algorithm group

with an Actor - critical network structure.

$$\max_a Q_{\theta Q}(s,a) = Q_{\theta Q}\left(s, arg\max_a Q_{\theta Q}(s,a)\right) \quad (7)$$

If we build a neural network to choose the optimal action for a particular state $\mu_{\theta_\mu}(s) arg\max_a Q_{\theta Q}(s,a)$, then optimize the component $Q_{\theta Q}$ according to the network parameters θ_μ just created, then we have:

$$\theta_\mu \leftarrow arg\max_{\theta_\mu} Q_{\theta Q}\left(s, \mu\theta_\mu(s)\right) \quad (8)$$

This optimization considers the change $Q_{\theta Q}$ according to the variable θ_μ, or in other words, the evaluation of the action. This can be calculated using a string rule like the following expression:

$$\frac{dQ\theta_Q}{d\theta_\mu} = \frac{dQ\theta_Q}{d\mu} \cdot \frac{d\mu}{d\theta_\mu} \quad (9)$$

So building a function that approximates the value of an action by a $\mu_{\theta_\mu}(s)$ neural network here is the main difference of DDPG.

Thus, in each DDPG algorithm structure, there are always two components, one is Actor $\mu_{\theta_\mu}(s)$, and the other is Critic $Q_{\theta Q}(s,a)$ lattice.

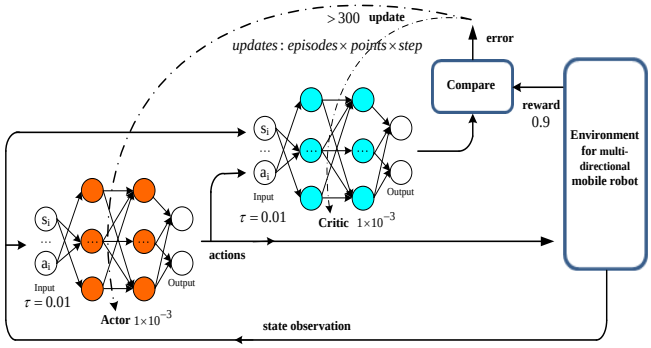

Figure 2: The DDPG algorithm structure

The relationship between the two components above and their connection to this algorithm's enhanced learning environment when controlling the multi-directional mobile robot in a random open environment with many fixed and obstacle obstacles mobile. At that time, the DDPG algorithm always meets the requirements of the control quality and the robot's working quality, as shown in figure 2.

We can understand more clearly that the DDPG algorithm is improved from other algorithms to have the ability to compute continuous action space problems for multi-directional mobile robots; when Then we go to update the target function as follows:

For an input sample set of $(s_i, a_i, R_i, s_i^{'})$, then the formula updates the target function value as follows:

$$y_i = R_i + \gamma Q'(s_i^{'}, \mu'(s_i^{'})) \quad (10)$$

Then we compute the loss function for the sample M value $(s_i, a_i, R_i, s_i^{'})$ to train the written network:

$$J = \frac{1}{M}\sum_{i=1}^{M}(y_i - Q(s_i, a_i))^2 \quad (11)$$

According to the string rule in expression (9), the gradient is calculated as follows:

$$\nabla_{\theta_\mu} J = \frac{1}{M}\sum_{i=1}^{M} G_{ai} G_{\mu i} \quad (12)$$

Which $G_{ai} = \nabla_a Q(s_i, a)$ is the output gradient of the Critic network according to variable a, estimated by the Actor-network $a = \mu(s_i)$. And $G_{\mu i} = \nabla_{\theta\mu}\mu(s_i)$ is the gradient of the Actor-network input according to the model parameter θ_μ?

The DDPG algorithm with neural network training and training always ensures accurate and reliable control; DDPG Agent updates the network parameter θ of the Q rating (*S, a*) at each step of the process. Network trainer, to do action a, receive new algorithm R, then it will significantly improve the performance of the control model for multi-directional mobile robot when using DDPG algorithm control programming. Moreover, DDPG always constantly explores the space for action, which is also a great challenge for scientists [5, 6].

B. The robot navigation using the DDPG algorithm

DDPG is a very recently developed algorithm for deep reinforcement learning to solve complex sensory input tasks, not multidimensional processing [23]. This algorithm is comparable to the human level in many control problems for industrial robots, mobile robots, and civil robots helping humans. However, it has not yet been used in intelligent control in industrial factories or workshops. This algorithm inherits its advantages from previous algorithms such as Q-learning and deep Q-network [3, 5, 7]. Furthermore, compared to Q-learning, it has continuous action space. Compared with a deep Q - network, it has a policy network that provides definite action. Here, the principle of DDPG is introduced.

Both DDPG and Q-learning have a deterministic policy gradient. The deterministic policy gradient is the expected gradient of the action-value function gradient, which can be estimated much efficiently than the usual stochastic policy gradient [7]. Considering the target policy μ: $S \rightarrow A$ directly maps state to deterministic action, The action-value function

is built based on continuous action space thanks to the robot's intelligent automatic navigation process.

The DDPG is a member of the actor-critic algorithm, which contains four neural networks: Current critic network $Q(s,a|\theta^Q)$, current actor-network $\mu(s|\theta^\mu)$), target critic network $Q'(s,a|\theta^{Q'})$, and target actor-network $\mu'(s|\theta^{\mu'})$, where $\theta^Q, \theta^\mu, \theta^{Q'}, \theta^{\mu'}$ are the weights of each network. The ingredient Q' and μ' are a copy of Q and μ respectively in the structures. Both $\theta^{Q'}$ and $\theta^{\mu'}$ are partially updated from the current networks at each timestep. The current critic network is updated by minimizing the loss function. Then, the gradient function is continuous, ensuring that the robot's agent action when controlled in an obstacle environment and now the algorithm is updated in a continuous space. The specific process of the DDPG algorithm for navigating multi-directional mobile robots is described in detail as follows:

Algorithm: Deep Deterministic Policy Gradient algorithm for multi-directional mobile robot

Randomly initialize critic network $Q(s,a|\theta^Q)$, actor network $\mu(s|\theta^\mu)$ with weights θ^Q and θ^μ.
Initialize target network Q' and μ' with weights $\theta^{Q'} \leftarrow \theta^Q, \theta^{\mu'} \leftarrow \theta^\mu$
Initialize replay buffer R
for *episode* $= 1$ to M **do**
  Initialize a random process N
  Receive initial state s_1 from environment E
  for $t = 1$ to T **do**
    Select action $a_t = \mu(s_t|\theta^\mu) + N_t$ according to the current actor network
    Execute action a_t in the environment E, and receive reward r_t and new state s_{t+1}
    Store transition (s_t, a_t, r_t, s_{t+1}) in buffer R
    Sample a random minibatch of N transitions (s_i, a_i, r_i, s_{i+1}) from R
    Set $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$
    Update the critic by minimizing the loss: $L = \frac{1}{N}\sum_i [y_i - Q(s_i, a_i|\theta^Q)]^2$
    Update the actor policy using the sampled gradient:
$$\nabla_{\theta^\mu}\mu|_{s_i} \approx \frac{1}{N}\sum_i \nabla_a Q(s,a|\theta^Q)|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^\mu}\mu(s|\theta^\mu)|_{s=s_i}$$
    Update the target networks:
$$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}, \theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$$
  end for
end for

The performance of the DDPG algorithm deployed is very positive on the multi-directional mobile robot control model. One of the reasons the author chose to study this algorithm for the primary control of multi-directional robots was to develop something industrially controllable. Comparing the DDPG algorithms with other algorithms has also been successful for mobile navigation for robots. Some tests for each form have been given, and it is clear that the DDPG algorithm works better than other algorithms like Q-learning, RL, etc. Therefore, to build a complete DDPG algorithm, it is always necessary to meet the needs: from selecting a control action to a robot, executing the action; receive rewards, store and sample to train the algorithm, then go to the calculation of the target function, thereby updating the model parameters by minimizing the loss function on all selected samples, followed by selecting the method to update the target neural network parameters, and finally updating the environmental discovery coefficient during the control process, [1, 3, 10, 15, 21].

IV. RESULTS OF SIMULATION AND EXPERIMENTAL

A. Research multi-directional mobile robot

Here, the author studies the multi-directional mobile robot model: with the actual hardware architecture, which is shown in figure 3. Each hardware module will perform a number of tasks. In the sequence of activities of this multi-directional mobile robot: such as finding a path, crossing obstacles, etc.

Figure: 3. The components of the Turtlebot robot mobile multi-directional

The authors implement some tasks in the multi-directional mobile robot operation on a Raspberry Pi 3 Model B+, which supports Ubuntu in the experiments. The embedded computer Raspberry Pi 3 Model B+ directly processes information from a series of sensors, including Astra camera intelligence, and sensor intelligence, then transmits the command to a microcontroller intelligence. For capturing images from the environment as well as measuring the distance between the multi-directional mobile robot an unknown obstacle, the mobile robot is equipped with a camera and sensor intelligence, in which the camera intelligent can perform 360-degree and laser scanning range within 15m that produces map data used for the mapping process. The intelligent microprocessor control circuit will be the part that receives the control signal from Raspberry Pi 3 Model B+ and then directs the signal to the MOSFET bridge circuit to operate two motors.

B. The simulation results

In this section, some simulations are conducted based on the powerful simulation engine and the environment in Gazebo. Figure 4 shows the map built on Gazebo is a map created with strict walls, and a mobile robot can be manipulated to move around fixed obstacles creating

a context for robot movements used to build mobile robot action mapping. The blue lines represent the robot's path when avoiding obstacles created by smart tree sensors, smart cameras, and updated robot current position (the blue tree denotes mobile robot) using the geometrical measuring dimensions. This visual tool can provide live updates of maps generated from the SLAM algorithm to control the robot. Furthermore, a multi-directional mobile robot's motion trajectory in the map can also be visualized and generated in an obstacle environment, as shown in figure 4 for the robot to move.

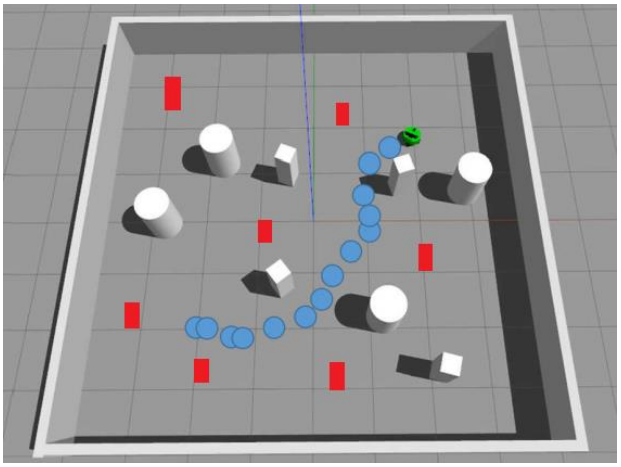

Figure 4: The Constructing visual maps and robot simulation models in Gazebo

As shown in figure 4, a navigation system for the mobile robot is fully automated; the path planning will determine the route for the robot that needs to be programmed with a smart control algorithm to reach the desired final destination without hitting any obstacles.

C. The experimental results

This section shows Turtlebot, the actual multi-directional mobile robot into a real environment, which is the environment used for real testing.

In figure 5, to create an operating environment to control, navigate, and navigate the robot, the author has set up the mapping to create obstacle environments and then program the controller for multi-directional mobile robots. Move around and avoid obstacles with devices like cameras and smart sensors. The environment here includes obstacles created by different flower pots, the robot's path will be taught in advance through a computer, a Wifi network, and the actual Turtlebot Robot. Based on the DDPG algorithm and SLAM algorithm. This is primarily a visualization tool that can provide live updates of the maps generated from the SLAM and DDPG algorithms. Moreover, the vehicle's trajectory in the map can also be displayed in the real environment that the training and teaching process, identification so that the robot knows during obstacle avoidance smartly and completely good.

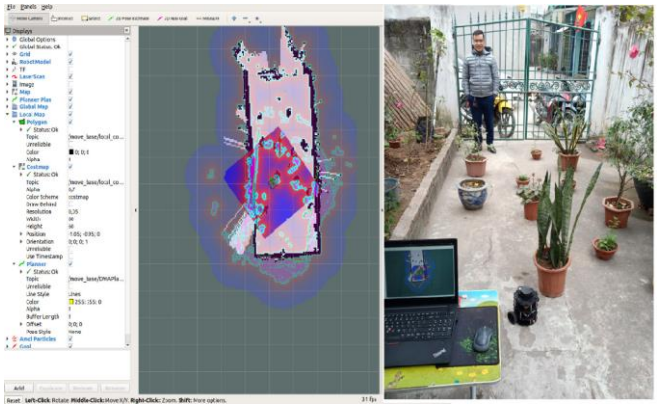

Figure 5: The navigation for Turtlebot mobile multi-directional robot in realistic map with the obstacle

Compared with some other algorithms: deep learning Q-learning, DDPG algorithm is better than DQN, Q-learning in terms of value accuracy and control strategy [13, 17, 21, 28], this is also consistent with the DDPG algorithm that the author has suggested in this article. Accelerated learning technology and rapid action processing in large environments can be used to achieve the state-to-action mapping and meet mobile robots' moving needs. The data also demonstrated that the robot path planning method based on the DDPG method, the end-to-end mobile robot path planning method, was also studied in a study of [17]. The above results illustrate the algorithm's strength, the optimal problem of the proposed method in planning the path of the multi-directional mobile robot.

V. CONCLUSIONS

In this article, the author has presented the robot self-training strategy without prior experience under clear feedback. The author explores the navigation problem of the multi-directional mobile robot using the reinforcement learning method and the neural network. The DDPG algorithm is applied to improve multi-directional mobile robots' self-learning through trial-and-error interactions with an unknown environment. The author has designed the expression using and introduced the neural network architecture to store and train network training on a large scale and generalize performance and learning capabilities for action spaces and extended-scale states. Experimental studies performed on automated navigation tasks for mobile multi-directional robots have yielded better results than previous studies. The simulation results show that the stability and applicability of the method using the DDPG algorithm are very effective; these new studies can completely apply to the control and navigation of mobile robots in Industrial factories in Vietnam as well as in the world, more than before [13, 17, 21, 28], the research results of the paper are better. The DDPG algorithm is then simpler, making the navigation of the mobile robot possible to be monitored through intuitive tools, such as cameras and smart sensors for precise control and avoiding obstacles.

ACKNOWLEDGMENT

This study was supported by the Faculty of Electronic Engineering Technology, University of Economics - Technology for Industries, Viet Nam; No 353 Tran Hung Dao Road, Ba Trieu District, Nam Dinh City, Viet Nam International. http://www.uneti.edu.vn/.

REFERENCES

[1] Tran Hoai Linh, Neural network and its application in signal processing, Hanoi Polytechnic Publishing House, (2015).

[2] M.N. Cirstea, A. Dinu, J.G. Khor, M. McCormick, Neural and Fuzzy Logic Control of Drives and Power Systems,Linacre House, Jordan Hill, Oxford OX2 8DP, First published (2002).

[3] Nguyen Thanh Tuan, Base Deep learning, The Legrand Orange Book. Version 2, last update, (2020).

[4] Charu C. Aggarwal,Neural Networks and Deep Learning, Springer International Publishing AG, part of Springer Nature, (2018).

[5] Nils J. Nilsson, The quest for artificial intelligence a history of ideas and achievements, Web Version Print version published by Cambridge University Press, Publishing 13(2010), http://www.cambridge.org/us/0521122937.

[6] Vu Thi Thuy Nga, Ong Xuan Loc, Trinh Hai Nam, Enhanced learning in automatic control with Matlab Simulink, Hanoi Polytechnic Publishing House, (2020).

[7] Mohit Sewak, Deep Reinforcement Learning, Springer Nature Singapore Pte Ltd.(2019).

[8] Latombe, J.C.,Robot Motion Planning, Kluwer Academic Publishers: Norwell, MA, USA, (1992).

[9] Han, J., and Seo, Y.,Mobile robot path planning with surrounding point set and path improvement, Appl. Soft Comp. 57(2018) 35–47.

[10] V. Matt and N. Aran.,Deep reinforcement learning approach to autonomous driving, ed: arXiv, (2017).

[11] Andrea Bacciotti**,**Stability and Control of Linear Systems,Publishing Ltd; Springer Nature Switzerland AG,(2019).

[12] L. Xin, Q. Wang, J. She, and Y. Li, Robust adaptive tracking control of wheeled mobile robot, Robotics and Autonomous Systems, 78(2016) 36–48.

[13] Bakdi, A., Hentout, A., Boutami, H., Maoudj, A., Hachour, O., and Bouzouia, B.,Optimal path planning and execution for mobile robots using genetic algorithm and adaptive fuzzy-logic control, Robot. Autonomous Syst. 89(2017) 95–109.

[14] I. Zamora, N. G. Lopez, V. M. Vilches, and A. H. Cordero, "Extending the open-air gym for robotics: A toolkit for reinforcement learning using ros and gazebo, arXiv preprint arXiv:1608.05742, (2016).

[15] Gu, S.; Holly, E.; Lillicrap, T.; Levine, S.,Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates.In Proceedings of the 2017 IEEE International Conference on Robotics and Automation, Marina Bay Sands, Singapore,(2017) 3389–3396, 29 May–2.

[16] A. D. Pambudi, T. Agustinah, and R. Effendi.,Reinforcement Point and Fuzzy Input Design of Fuzzy Q-Learning for Mobile Robot Navigation System,International Conference of Artificial Intelligence and Information Technology, (2019).

[17] Do Quang Hiep, Ngo Manh Tien, Nguyen Manh Cuong, Pham Tien Dung, Tran Van Manh, Nguyen Tien Kiem, Nguyen Duc Duy,An Approach to Design Navigation System for Omnidirectional Mobile Robot Based on ROS, (IJMERR);11(9)(2020) 1502-1508.

[18] X. Ruan, D. Ren, X. Zhu, and J. Huang,Mobile Robot Navigation based on Deep Reinforcement Learning, 2019 Chinese Control And Decision Conference (CCDC),(2019).

[19] N. Navarro-Guerrero, C. Weber, P. Schroeter, and S. Wermter,Real-world reinforcement learning for an autonomous humanoid robot, Robotics and Autonomous Systems,(2012).

[20] Saleem, Y.; Yau, K.L.A.; Mohamad, H.; Ramli, N.; Rehmani, M.H.; Ni, Q.,Clustering and Reinforcement Learning-Based Routing for Cognitive Radio Networks.,IEEE Wirel. Commun.(2017).

[21] Z. Miljković, M. Mitić, M. Lazarević, and B. Babić.,Neural network reinforcement learning for visual control of robot manipulators, Expert Systems with Applications,40(2013) 1721–1736.

[22] Pham Ngoc Sam, Tran Duc Chuyen,Research and Designing a Positioning System, Timeline Chemical Mapping for Multi-Direction Mobile Robot,7(11)(2020) 7-12, Publishing by SSRG - IJECE Journal,.

[23] Fu X, Du J, Guo Y, Liu M, Dong T, et al.,A machine learning framework for stock selection., arXiv, cited (2018).

[24] Shota Ohnishi, Eiji Uchibe, Yotaro Yamaguchi, Kosuke Nakanishi, Yuji Yasui, and Shin Ishii,constrained Deep Q-Learning Gradually Approaching Ordinary Q-Learning,13(2019) 7-12, Publishing by Frontiers in Neurorobotics Journal,.

[25] https://www.mathworks.com/help/reinfocerment-learning/ug/ddpg-Agent.html, (2020).

[26] Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G., Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al.,Human-level control through deep reinforcement learning., Nature(2015).

[27] Ganggang Guo, Yulei Rao, Feida Zhu, Fang Xu.,Innovative deep matching algorithm for stock portfolio selection using deep stock profiles,PLoS One. 15(11)(2020) Published online 2020 November.

[28] Wang, C.; Zhang, Q.; Tian, Q.; Li, S.; Wang, X.; Lane, D.; Petillot, Y.; Wang, S.,Learning Mobile Manipulation through Deep Reinforcement Learning., Sensors (2020).

[29] Evan Prianto, MyeongSeop Kim, Jae-Han Park, Ji-Hun Bae, and Jung-Su Kim,Path Planning for Multi-Arm Manipulators Using Deep Reinforcement Learning: Soft Actor-Critic with Hindsight Experience Replay,Sensors, Published: 19(2020).