International Journal of Engineering Trends and Technology
ISSN: 2231–5381 / https://doi.org/10.14445/22315381/IJETT-V73I4P114

Volume 73 Issue 4, 139-147, April 2025
© 2025 Seventh Sense Research Group®

*Original Article*

# Control the Mobile Robot to Avoid Obstacles and Reach the Target Using Artificial Intelligence

Dang Khanh Toan[1], To Van Binh[1*]

[1]*Faculty of Electronic Engineering Technology, University of Economics-Technology for Industries, Ha Noi, Viet Nam.*

[*]*Corresponding Author : tovanbinh.uneti@moet.edu.vn*

Received: 16 August 2024 Revised: 14 November 2024 Accepted: 06 March 2025 Published: 26 April 2025

***Abstract*** *- Mobile robots can be classified based on their working environment, including air, water and land. In each place, the robot needs a different drive system. For mobile aerial robots, the moving parts are the propeller or flying wing and the motor. With underwater mobile robots, depending on the place of work on or in the water, there will be different transmission structures: working on the water surface, the moving parts are buoys or motors with control and operation parts. Moving deep underwater, the moving parts can be legs or even jet engines. Land mobile robots have quite a variety of moving parts. Depending on the operating terrain, the moving parts can be legs, wheels, crawlers or a combination of both. The most popular is the robot that moves on wheels. Mobile robots are applied to many different types of work, from construction to agriculture, mine digging to oil and gas exploration, environmental remediation, healthcare, entertainment, transportation, etc. Robots have the ability to help a lot. Many of the jobs that humans cannot do. In this article, we propose to use artificial intelligence (AI) technology based on the Deep Deterministic Policy Gradient (DDPG) algorithm to control the mobile robot to avoid obstacles and reach the target. Simulation results on Matlab-Simulink software show the feasibility of the proposed algorithm. The robot can safely accomplish its goals in environments with obstacles and become a truly intelligent system with strong self-learning and adaptation capabilities.*

***Keywords*** *- Artificial Intelligence, Deep Deterministic Policy Gradient, Mobile Robot, Machine Learning, Reinforcement Learning.*

## 1. Introduction

There are many difficult and dangerous jobs, such as dredging sewer pipes or removing mines, or jobs that must be done in difficult and dangerous terrain, such as in the deep sea or the cold Arctic region. At that time, highly customizable mobile robots will be an effective source of support. That is why, in recent decades, many articles have been published on mobile robot control. The main topics focus on research to improve the quality of trajectory tracking control, map building, and navigation in different environments for mobile robots.

In these articles, the authors have considered and taken into account surrounding environmental information and different control methods to help robots operate in real environments with and without obstacles. The main target is to focus on implementing control strategies to improve the quality of robot control to ensure safety and optimal problems, such as moving in the fastest time or consuming the least amount of energy [1-5]. Smart robots have shown their effectiveness in simple, obstacle-free environments. However, scientists are still researching to find more effective control methods in complex environments affected by external disturbances. These influencing factors include turbulence due to the unknown operating environment, random changes in the environment, and vehicle position error leading to the uncertainty of the control model, so the control method relying on a "clear" model would not be appropriate. In that context, researchers are interested in control methods based on the system's input and output data. The controller is designed from collected input/output data, which can be offline or online control or can also be knowledge from the environment to process data sent to the controller.

Fuzzy logic theory was proposed by Zadeh, L.A., and was first raised in 1965. This theory solves problems very closely with human thinking. Up to now, fuzzy logic theory has developed very strongly and applied in many life areas. Fuzzy logic control has also shown its suitability for application in mobile robot control due to its ability to calculate accurately and reason under conditions of uncertainty [6-11]. Some studies have used fuzzy logic control methods to solve the obstacle avoidance problem of mobile robots. In the document [12-15], a fuzzy logic controller is proposed to control the robot in real-time with the target of avoiding obstacles in an unknown random environment. However, the disadvantages


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

of the method are the oversampling process to handle complex problems and the less flexible structure. In that context, neural network-based control methods are proposed for research due to the advantages of using neural networks to self-learn and approximate uncertain nonlinear components in the model. Many studies have been proposed based on multilayer cognitive neural networks to classify, recognize and provide approximate sun functions. In [16-20], the authors introduce a recurrent neural network structure to track the path and stability of autonomous vehicles. The learning rate of this neural network is suitable and makes it possible for the mobile robot to generate a changing trajectory in the optimal time. Recently, a new control structure was proposed, which is a combination of a recurrent neural network and fuzzy logic, to take advantage of the advantages of each method [21-24].

Reinforcement Learning (RL) is one of the three main types of machine learning (ML) besides Supervised Learning and Unsupervised Learning. The nature of RL is trial and error, which means trying again and again and gaining experience after each try. Recently, RL has achieved remarkable achievements when DeepMind's algorithms (AlphaGo, AlphaZero, AlphaStar,...) have overwhelmingly won against world players in games that humans once thought were possible. Machines will never be able to surpass Go or StarCraft [25,26]. Therefore, the article uses a reinforcement learning algorithm to learn a mobile robot's control method (linear and angular velocity) to avoid collisions with obstacles and achieve the target.

## 2. Modeling Mobile Robots

The geometrical structure of the mobile robot is depicted in Figure 1.

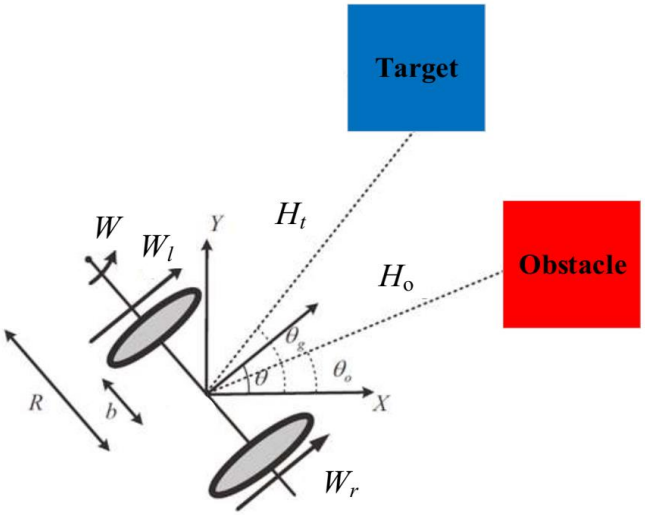


**Fig. 1 Geometric structure of mobile robot**

The model of the AVs is shown as follows [21]:

$$P(k+1) = P(k) + \Delta T \times R \times Q(k) \quad (1)$$

where $R = \begin{bmatrix} \cos\theta(k) & 0 \\ \sin\theta(k) & 0 \\ 0 & 1 \end{bmatrix}$; $P(k)$, $P(k+1)$ are the positions of the autonomous vehicle at time $k$ and $k+1$. The position of AV described by $(X, Y, \theta)$, LiDAR sensor is responsible for measuring and collecting data.

$$P = [X(k), Y(k), \theta(k)]^T \quad (2)$$

and $Q(k) = [V(k), W(k)]^T$

$$\begin{cases} X(k+1) = X + V \times cos\,\theta \times T \\ Y(k+1) = Y + V \times sin\,\theta \times T \\ \theta(k+1) = \theta + W \times T \end{cases} \quad (3)$$

Where $T$ is the sampling time, suppose the desired trajectory of the Vehicle is:

$$P_r = h\,[X_r(k), Y_r(k), \theta_r(k)]^T \quad (4)$$

We have

$$P_r(k+1) = P_r + \Delta T \times \begin{bmatrix} \cos\theta_r & 0 \\ \sin\theta_r & 0 \\ 0 & 1 \end{bmatrix} \times Q \quad (5)$$

$$P_r(k+1) = P_r + \Delta T \times \begin{bmatrix} \cos\theta_r & 0 \\ \sin\theta_r & 0 \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} V_r \\ W_r \end{bmatrix} \quad (6)$$

Where $Q_r(k) = [V_r(k), W_r(k)]^T$

$$\begin{cases} X_r(k+1) = X_r + V_r \times cos\,\theta_r \times T \\ Y_r(k+1) = Y_r + V_r \times sin\,\theta_r \times T \\ \theta_r(k+1) = \theta_r + W_r \times T \end{cases} \quad (7)$$

Tracking error

$$\begin{aligned} X_{TE}(k) &= [X(k) - X_r(k)] \times cos\,\theta_{TE}(k) \\ &\quad + [Y(k) - Y_r(k)] \times sin\,\theta_{TE}(k) \\ Y_{TE}(k) &= [X(k) - X_r(k)] \times sin\,\theta_{TE}(k) \\ &\quad + [Y(k) - Y_r(k)] \times cos\,\theta_{TE}(k) \end{aligned} \quad (8)$$

$$\theta_{TE}(k) = \theta(k) - \theta_r(k)$$

From there, we have:

$$\begin{cases} X_{TE}(k+1) = X_{TE} \times [V_1 - V_r + Y_{TE} W_r] \times T \\ Y_{TE}(k+1) = Y_{TE} \times [V_2 - X_{TE} W_r] \times T \\ \theta_{TE}(k+1) = \theta_{TE} + [W - W_r] \times T \end{cases} \quad (9)$$

Where:

$$V_1 = V_r(k)\,cos\,\theta_{TE}(k) \quad (10)$$

$$V_2 = V_l(k)\cos\theta_{TE}(k) \quad (11)$$

AV moves by two wheels, driving the right and left with the linear velocity of the two wheels, $V_r$, $V_l$ respectively.

$$V = \frac{V_r + V_l}{2} \quad (12)$$

$$W = \frac{V_r - V_l}{2b} \quad (13)$$

$$R = \frac{V_r + V_l}{b(V_r - V_l)} \quad (14)$$

Where $R$ is the turning radius, and the surroundings are not clear with obstacles and targets. To design the controller, useful information includes the distance and angle between the mobile robot and the target and the distance and angle between the mobile robot and the nearest obstacle. Then, the observed model shows the following:

$$O(k) = \left[H_{target}, \theta_g, H_{obstacle}, \theta_o\right]^T \quad (15)$$

## 3. Reinforcement Learning Techniques in Mobile Robot Control

In recent years, rapid development in the use of machine learning techniques in robot control problems has been witnessed. Reinforcement learning is a type of machine learning technique in which a system automatically learns and improves its behavior through interaction with the environment. This process is based on the principle of learning from feedback and rewards to maximize a predetermined reward function. Reinforcement learning acts as a signal for positive and negative behaviors. Its sole target is finding a suitable action pattern to increase the agent's total cumulative reward.

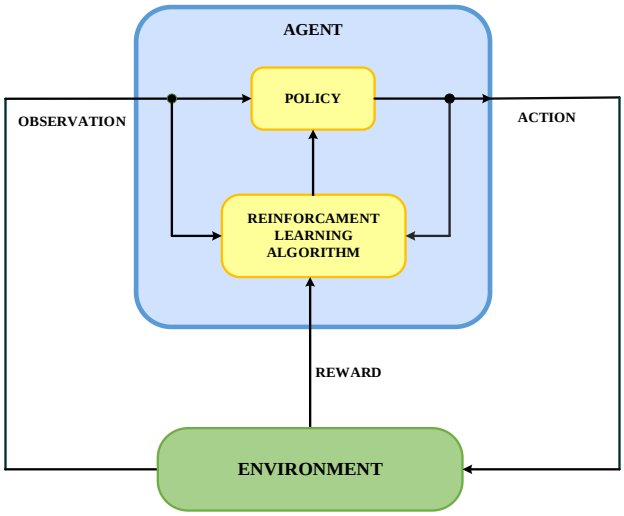


**Fig. 2 Diagram of interaction between agent and environment in reinforcement learning techniques**

This learning method allows the agent to make a series of decisions that maximize the reward index for the task without requiring human intervention or explicit programming to achieve the task. Commonly used terms in reinforcement learning, as shown in Figure 2, include:

- Environment: Environment injects a state into the network; The output is the Q-value of the corresponding actions.
- Agent: The agent selects an action using a Policy and executes that action.
- Actions: A set of methods by which an agent affects the environment.
- State: The agent's state is returned through impact on the environment.
- Reward: Reward corresponding to each state the agent receives in the environment.
- Episode: A cycle that includes interactions between an agent and the environment from start to finish.
- Policy: Policy or a rule established for an agent to accomplish a target set in the environment.
- Value Function: An agent's assessment of the goodness of a state or action. Value can be calculated by predicting or estimating the expected value of a state or action.
- Model: It defines the environment in which the system will operate. The environment can be a game, a real-world task, or any system with which the system needs to interact in order to learn.

In RL, to evaluate the agent's actions, we use a reward function. For actions such as expecting the agent to receive a reward, doing the opposite will result in a penalty. To encourage, assign a positive value when the agent performs the desired action; otherwise, a negative value will be assigned to the undesirable action. Thus, this process helps the agent seek the maximum reward when acting to achieve the optimal target. After the process of exploration, discovery and learning, the agent knows how to perform positive actions and avoid negative actions. This learning method as a way of directing unsupervised ML through rewards and punishments has been applied in AI.

### *3.1. DDPG Algorithm*

DDPG is a model-independent reinforcement learning algorithm, online learning, using off-policy methods and Actor-Critic (AC) structure. Similar to Double DQN (Deep Q-Network), the Q-value update process in DDPG is calculated according to the following equation:

$$\underbrace{max}_{a} Q_{\theta_Q}(s,a) = Q_{\theta_Q}\left(s, argmax Q_{\theta_Q}(s,a)\right) \quad (16)$$

For each state, to choose the optimal action, we build a neural network $\mu_{\theta_\mu} = arg\ \underbrace{max}_{a} Q_{\theta_Q}(s,a)$.

Optimizing $Q_{\theta_Q}$ according to the network parameter $\theta_\mu$ We get:

$$\theta_\mu \leftarrow arg \underbrace{max}_{\theta_\mu} Q_{\theta_Q}\left(s, \mu_{\theta_\mu}(s)\right) \quad (17)$$

This optimization is calculated as the following expression:

$$\frac{dQ_{\theta_Q}}{d\theta_\mu} = \frac{dQ_{\theta_Q}}{d\mu} \times \frac{d\mu}{d\theta_\mu} \quad (18)$$

Thus, in the DDPG algorithm structure, there exist two Actor-Critic components (Figure 3):

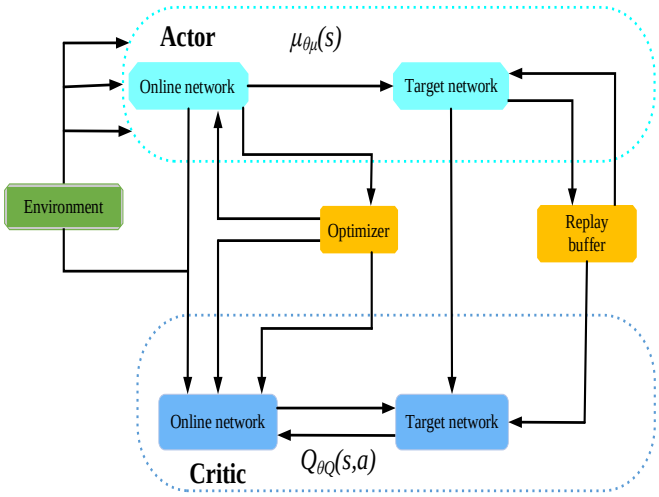


**Fig. 3 DDPG algorithm structure**

- Actor neural network represents control policy $\mu_{\theta_\mu}(s)$;
- The Critic neural network represents the cost function $Q_{\theta_Q}(s,a)$.

### *3.2. Build and Optimize Cost Function*

Equation to update the objective function value:

$$y_i = R_i + \gamma Q'\left(s_i', \mu'(s_i')\right) \quad (19)$$

The cost function for $N$ samples:

$$J = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - Q(s_i, a_i)\right)^2 \quad (20)$$

The gradient is calculated as follows:

$$\nabla_{\theta_\mu} J = \frac{1}{N}\sum_{i=1}^{N} G_{ai} G_{\mu i} \quad (21)$$

In which, $G_{ai} = \nabla_a Q(s_i, a)$ is the Critic neural network output gradient according to variable $a$; $G_{\mu i} = \nabla_{\theta_\mu}\mu(s_i)$ is the Actor neural network output gradient according to the model $\theta_\mu$.

## 4. Simulation Results

To demonstrate the effectiveness of the proposed control method, validation simulations are performed in this section using Matlab-Simulink software. The control structure diagram is shown in Figure 4. We use Lidar sensors to scan the operating environment and detect obstacles and targets for mobile robots. The accuracy of environmental perception is limited. Therefore, the simulation area is limited to $10m \times 10m$ space. In an environment where obstacles and targets exist, the mobile robot must reach the target while avoiding collisions. Let $v$ be the linear velocity $\omega$ be the angular velocity of the mobile robot. Thus, the action of the agent is represented by vector $a = [v \quad \omega]$, and $[x \quad y \quad \theta]$ is the initial position of the mobile robot. The agent's action is a scalar between -1 and 1. The reason we used normalized input for linear and angular velocity and multiplied by maxLinSpeed parameters and maxAngSpeed.

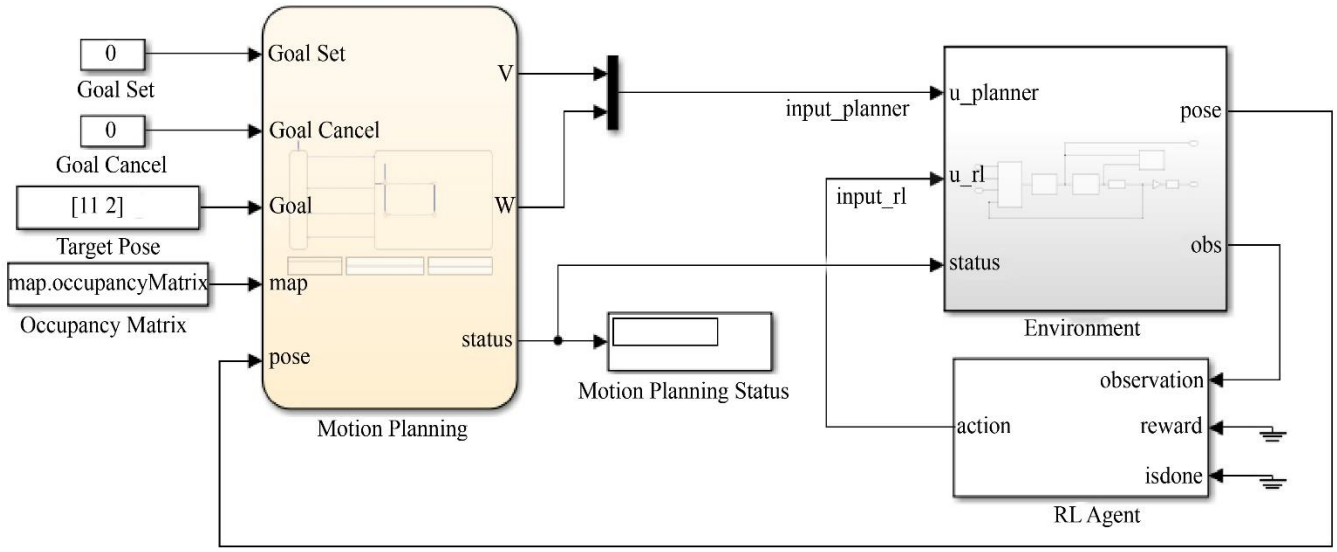


**Fig. 4 Structure diagram of autonomous robot control in Matlab Simulink**

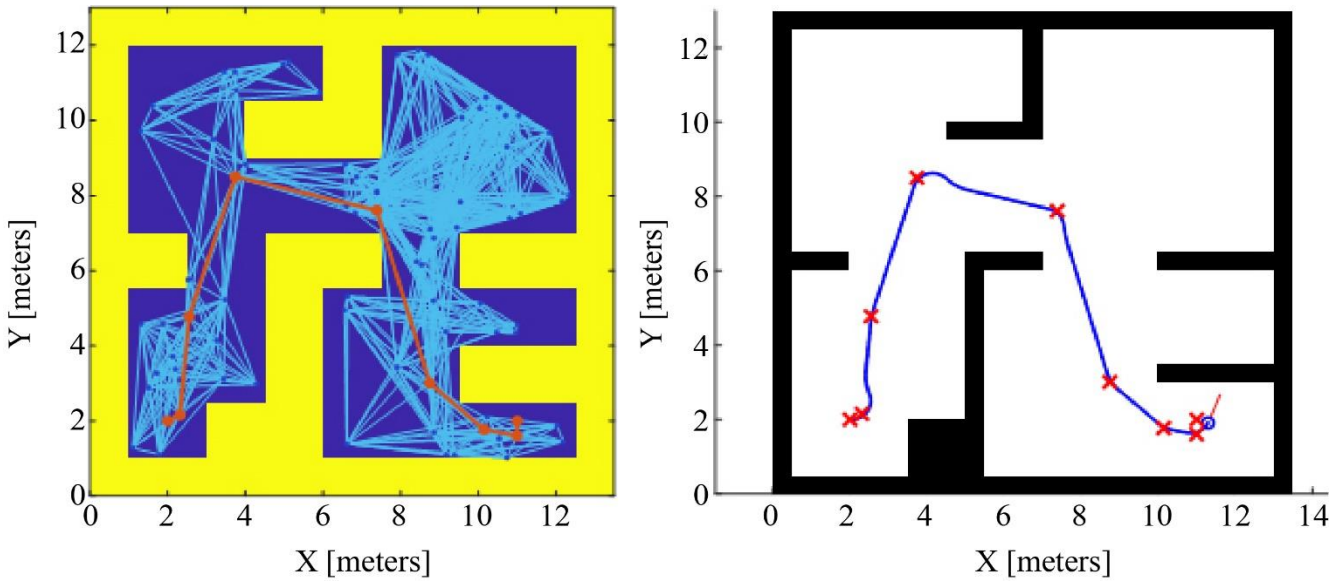


**Fig. 5 Schematic diagram of mobile robot path planning and tracking**

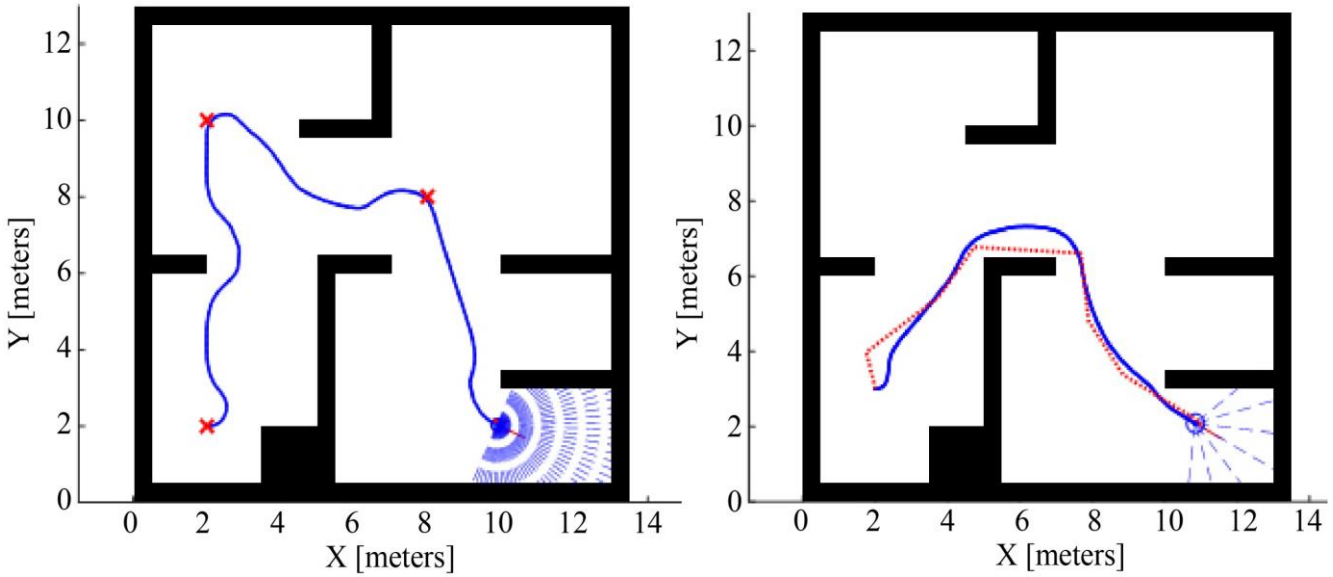


**Fig. 6 The mobile robot's motion trajectory reaches the target**

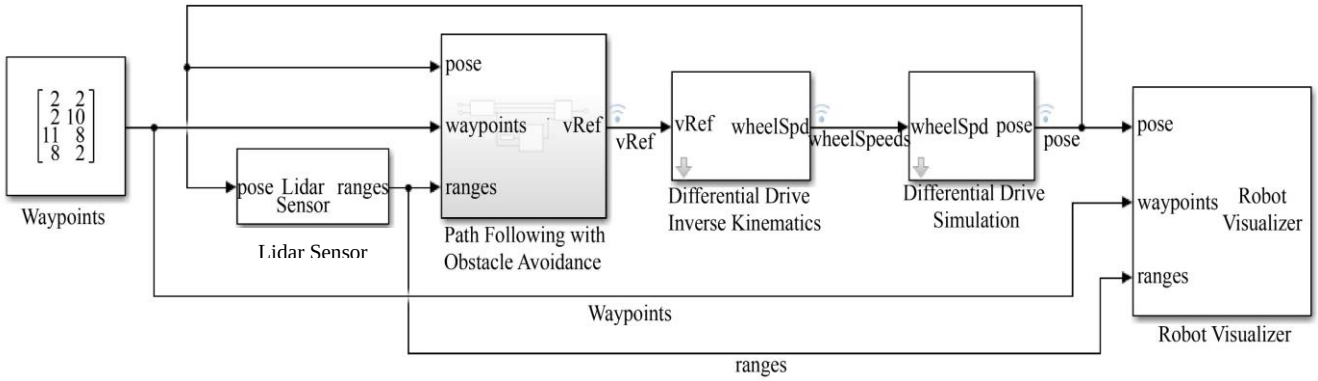


**Fig. 7 Tracking and obstacle avoidance of mobile robots**

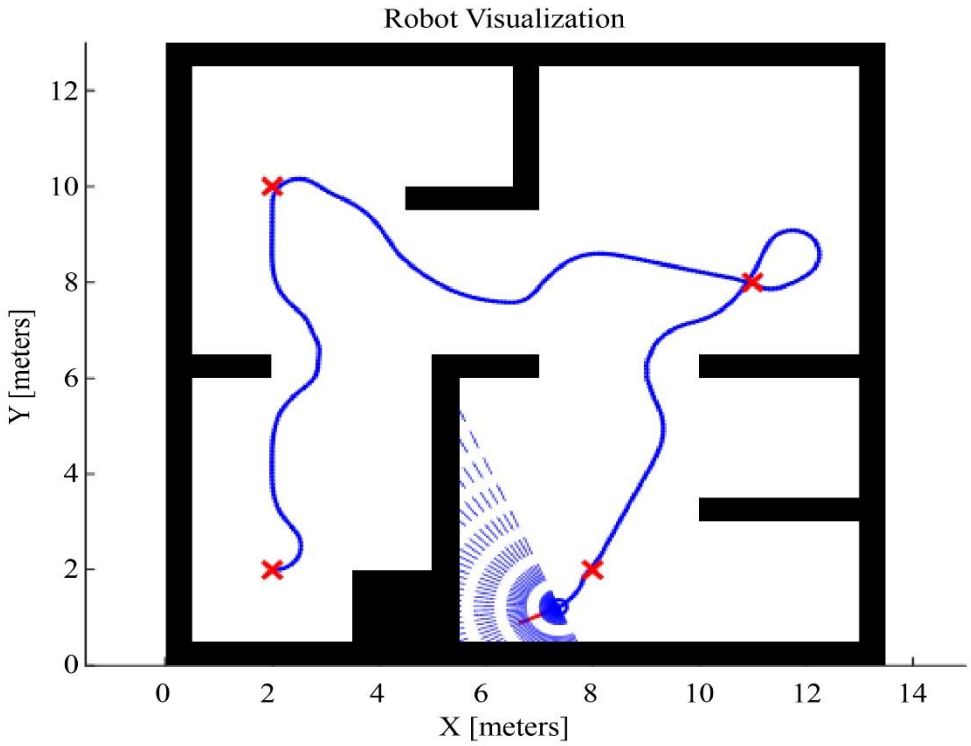


**Fig. 8 Mobile robot trajectory to targets**

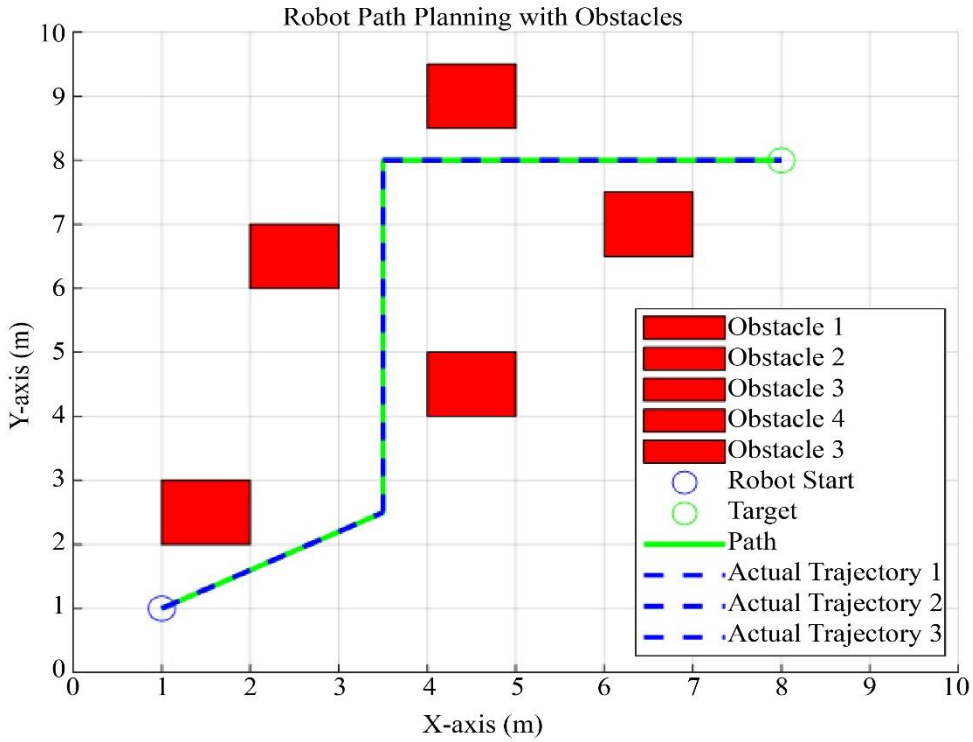


**Fig. 9 Robot path planning with obstacles**

Figure 9 illustrates the robot's motion plan, showing that it starts from a specified location, needs to pass through an area with many obstacles, and finally reaches a specific target. The actual trajectories show how the robot adjusts its direction to avoid obstacles, showing the complexity of path planning. Robot's starting position: the blue point (circle) located at position (1, 1) represents the robot's starting point. Target position: the green point (circle) located at position (8, 8)

represents the target the robot needs to reach. Red squares represent five obstacles. They are scattered in space and can block the robot's path. Each obstacle is numbered from 1 to 5. The robot's path, represented by a continuous green line, shows the direction in which the robot will move to reach the target. The blue dashed lines and other colors show the actual trajectory of the robot as it passes through the points on the path. This allows you to see how the robot adjusts direction to avoid obstacles. Figure 10 illustrates the blue line that represents the trajectory that the robot will travel from the starting point to the target. However, this trajectory may not be feasible because of obstacles in the way. The green dashed line shows the trajectory adjusted to avoid the obstacles. This represents the path-planning action of the robot to choose a safer route. This figure synthesizes information about the robot's location, obstacles, and path, showing how the robot plans to move safely from the starting point to the target while avoiding obstacles. It clearly illustrates the process of planning and adjusting the trajectory to achieve the target in a complex environment.

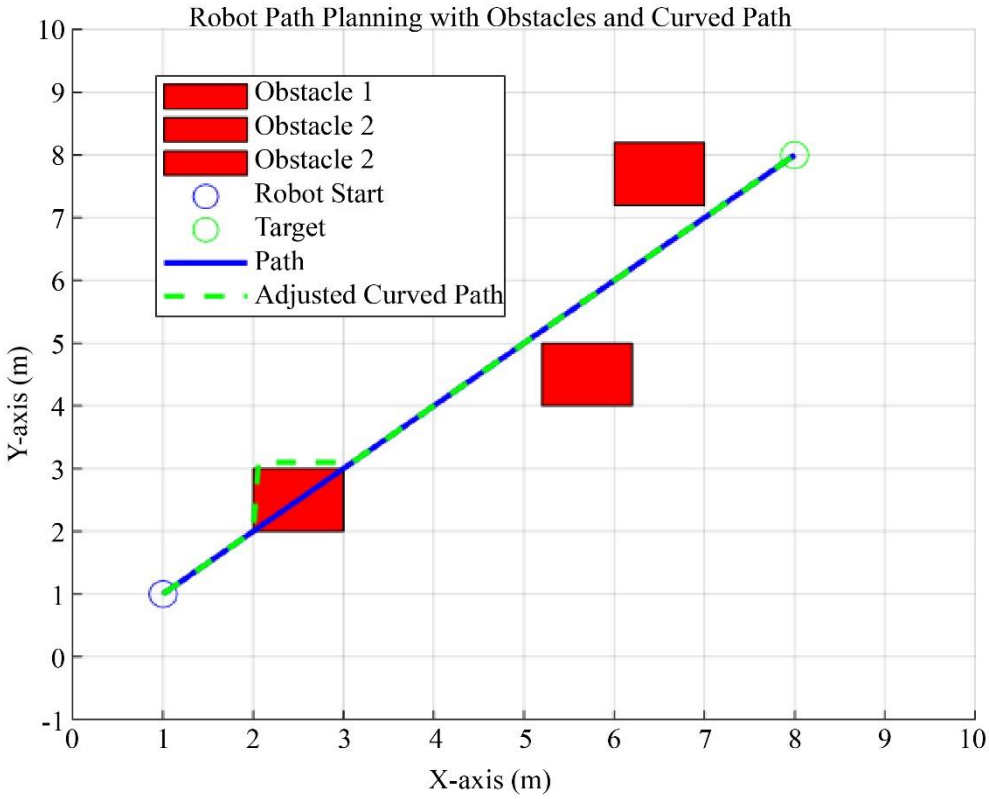


**Fig. 10 Robot path planning with obstacles and curved path**

From the simulation results, we can see that the mobile robot has performed intelligent navigation in an environment with fixed obstacles, such as walls, creating awareness for the robot to move to its destination safely without encountering any problems. Finally, the robot can intelligently navigate to the target quickly without any collision with obstacles in both simple and complex environments. This is also the basis for evaluating the ability to implement the proposed algorithm and control method, which can be fully applied in practice. The simulation results shown in Figures 3 to 6 were performed using Matlab software. It can be seen that at first, the two neural networks, Critic NN and Actor NN, are in the learning process, so the results of tracking the reference trajectory of WMR are not good.However, after this period, the weights of the two neural networks converge. The controller designed for WMR approximates and converges to optimal tracking quality. This results in the WMR's trajectory tracking quality increasing, and the WMR is tracking the reference trajectory with a tracking error of almost zero for all variables.

## 5. Conclusion

Mobile robots have been widely researched in recent years and have popular applications in factories, healthcare, defense, etc. Controlling mobile robots in unknown environments to reach locations, position desired targets and avoid obstacles accurately and safely in various complex environments. In this paper, the controller comprises a model structure designed to realize the purpose of automatic navigation for mobile robots.

The research results are verified by simulation using Matlab-Simulink software. We can see that the control model with the proposed structure gives accurate and safe results when the robot operates. The controller provides sustainable,

optimal results when the robot operates in environments affected by external disturbances, model uncertainty, and obstacles.

## Author Contributions

To Van Binh conducted the research; Dang Khanh Toan analyzed the data; To Van Binh and Dang Khanh Toan wrote, commented, and revised the paper; all authors approved the final version.

## Acknowledgements

This study was supported by the University of Economics-Technology for Industries, Ha Noi - Vietnam; http://www.uneti.edu.vn/.

## References

[1] Jiajia Chen et al., "Path Planning for Autonomous Vehicle Based on a Two-Layered Planning Model in Complex Environment," *Journal of Advanced Transportation*, vol. 2020, pp. 1-14, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[2] Bai Li et al., "Tractor-Trailer Vehicle Trajectory Planning in Narrow Environments with a Progressively Constrained Optimal Control Approach," *IEEE Transactions on Intelligent Vehicles*, vol. 5, no. 3, pp. 414-425, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[3] Xiaoyun Lei, Zhian Zhang, and Peifang Dong, "Dynamic Path Planning of Unknown Environment Based on Deep Reinforcement Learning," *Journal of Robotics*, vol. 2018, pp. 1-10, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[4] Jeevan Raajan et al., "Real Time Path Planning of Robot using Deep Reinforcement Learning," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 15602-15607, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[5] Dae Jung Kim, Yong Woo Jeong, and Chung Choo Chung, "Lateral Vehicle Trajectory Planning Using a Model Predictive Control Scheme for an Automated Perpendicular Parking System," *IEEE Transactions on Industrial Electronics*, vol. 70, no. 2, pp. 1820-1829, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[6] Berk Korkmaz et al., "Fuzzy Logic Based Self-Driving Racing Car Control System," *2018 6th International Conference on Control Engineering & Information Technology*, Istanbul, Turkey, pp. 1-6, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[7] Ugur Bugra Etli et al., "A Fuzzy Logic-Based Autonomous Car Control System for the JavaScript Racer Game," *Transactions of the Institute of Measurement and Control*, vol. 43, no. 5, pp. 1028-1038, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[8] Khizir Mahmud, and Lei Tao, "Vehicle Speed Control through Fuzzy Logic," *IEEE Global High Tech Congress on Electronics*, Shenzhen, China, pp. 30-35, 2013. [CrossRef] [Google Scholar] [Publisher Link]

[9] Xi L, and Byung-Jae Choi, "Design of Obstacle Avoidance System for Mobile Robot using Fuzzy Logic Systems," *International Journal of Smart Home*, vol. 7, no. 3, pp. 321-328, 2013. [Google Scholar] [Publisher Link]

[10] Tae-Seok Jin, "Obstacle Avoidance of Mobile Robot Based on Behavior Hierarchy by Fuzzy Logic," *International Journal of Fuzzy Logic and Intelligent Systems*, vol. 12, no. 3, pp. 245-249, 2012. [CrossRef] [Google Scholar] [Publisher Link]

[11] Liew Chia Woon, "*Obstacle Avoidance Robot Applying Fuzzy Control System*," Master's Thesis, Tun Hussein Onn University Malaysia, pp. 1-39, 2014. [Google Scholar] [Publisher Link]

[12] Anish Pandey, Shalini Pandey, and Parhi DR, "Mobile Robot Navigation and Obstacle Avoidance Techniques: A Review," *International Robotics & Automation Journal*, vol. 2, no. 3, pp. 96-105, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[13] A. Ramirez-Serrano, and M. Boumedine, "Real-time Navigation in Unknown Environments Using Fuzzy Logic and Ultrasonic Sensing," *Proceedings of the IEEE International Symposium on Intelligent Control*, Dearborn, MI, USA, pp. 26-30, 1996. [CrossRef] [Google Scholar] [Publisher Link]

[14] Pooya Mobadersany, Sohrab Khanmohammadi, and Sehraneh Ghaemi, "A Fuzzy Multi-Stage Path-Planning Method for a Robot in a Dynamic Environment with Unknown Moving Obstacles," *Robotica*, vol. 33, no. 9, pp. 1869-1885, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[15] Weria Khaksar, Md Zia Uddin, and Jim Torresen, "Fuzzy Motion Planning for Nonholonomic Mobile Robot Navigation in Unknown Indoor Environments," *International Journal of Mechanical Engineering and Robotics Research*, vol. 8, no. 1, pp. 6-11, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[16] Nathan A. Spielberg et al., "Neural Network Vehicle Models for High-Performance Automated Driving," *Science Robotics*, vol. 4, no. 28, pp. 1-13, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[17] Jelena Kocić, Nenad Jovičić, and Vujo Drndarević, "An End-to-End Deep Neural Network for Autonomous Driving Designed for Embedded Automotive Platforms," *Sensors*, vol. 19, no. 9, pp. 1-26, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[18] Sung Jin Yoo, Sung Jin Yoo, and Jin Bae Park, "Generalized Predictive Control Based on Self-Recurrent Wavelet Neural Network for Stable Path Tracking of Mobile Robots: Adaptive Learning Rates Approach," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 53, no. 6, pp. 1381-1394, 2006. [CrossRef] [Google Scholar] [Publisher Link]

[19] Liu Xiaohua, Wang Xiuhong, and Wane Yunge, "Generalized Predictive Control Based on Error Correction Using the Dynamic Neural Network," *Proceedings of the 3rd World Congress on Intelligent Control and Automation*, Hefei, China, vol. 3, pp. 1863-1865, 2000. [CrossRef] [Google Scholar] [Publisher Link]

[20] Gowtham Garimella et al., "Neural Network Modeling for Steering Control of An Autonomous Vehicle," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vancouver, BC, Canada, pp. 2609-2615, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[21] Yu Han, Qidan Zhu, and Yao Xiao "Data-Driven Control of Autonomous Vehicle Using Recurrent Fuzzy Neural Network Combined with PID Method," *2018 37th Chinese Control Conference*, Wuhan, China, pp. 5239-5244, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[22] Chong Wui Lim et al., "Obstacle Avoidance for Autonomous Driving Using Neuro-Fuzzy Architecture in an Urban Landscape," *2018 International Conference on Intelligent Rail Transportation*, Singapore, pp. 1-5, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[23] Raulcézar M.F. Alves, and Carlos R. Lopes, "Obstacle Avoidance for Mobile Robots: A Hybrid Intelligent System based on Fuzzy Logic and Artificial Neural Network," *IEEE International Conference on Fuzzy Systems*, Vancouver, BC, Canada, pp. 1038-1043, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[24] Tommaso Novi, Renzo Capitani, and Claudio Annicchiarico, "An Integrated Artificial Neural Network-Unscented Kalman Filter Vehicle Sideslip Angle Estimation Based on Inertial Measurement Unit Measurements," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 233, no. 7, pp. 1864-1878, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[25] Matthew Hausknecht et al., "A Neuroevolution Approach to General Atari Game Playing," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 6, no. 4, pp. 355-366, 2014. [CrossRef] [Google Scholar] [Publisher Link]

[26] David Silver et al., "Mastering the Game of Go without Human Knowledge," *Nature*, vol. 550, no. 7676, pp. 354-359, 2017. [CrossRef] [Google Scholar] [Publisher Link]