Proceedings of CONF-MLA 2025 Symposium: Intelligent Systems and Automation: AI Models, IoT, and Robotic Algorithms
DOI: 10.54254/2755-2721/2025.LD24727

Path Planning for Automated Guided Vehicles (AGVs) in Warehouses Based on Improved Q-learning Algorithm

Xuehao Shi

School of Naval Architecture and Ocean Engineering, University of Ulsan, Ulsan, South Korea
shixuehao525@gmail.com

Abstract. Efficient path planning for Automated Guided Vehicles (AGVs) is critical to improving logistics efficiency. Traditional Q-learning algorithms suffer from slow convergence, poor learning efficiency, and susceptibility to local optima in AGV path planning. To address these challenges, this paper proposes an improved Q-learning algorithm that integrates the attractive and repulsive force functions of the artificial potential field method, optimizing the reward mechanism and Q-value update strategy. By dynamically selecting reward values, the attractive force guides AGVs toward the target direction efficiently, while the repulsive force adjusts Q-values to enhance obstacle avoidance capabilities. Comparative simulation experiments in a 20×20 grid environment demonstrate that the improved algorithm accelerates convergence speed, enhances learning efficiency, significantly reduces path exploration steps, and improves obstacle avoidance success rates. This enables AGVs to autonomously and rapidly identify a collision-free optimal path through self-learning.

Keywords: Q-learning algorithm, path planning, artificial potential field method, AGV, dynamic reward

1. Introduction

In intelligent logistics systems, Automated Guided Vehicles (AGVs) are central to material handling tasks such as transportation, storage, and retrieval. The effectiveness of AGV operations is largely determined by the performance of their path planning algorithms. Conventional methods, including A*, genetic algorithms, and artificial potential field approaches [1–3], often face difficulties in adapting to dynamic environments and achieving globally optimal paths. Among machine learning-based techniques, Q-learning has been widely applied in mobile robotics due to its model-free nature and learning capability [4]. However, its practical deployment is hindered by issues such as slow convergence, high computational cost, and vulnerability to local optima. To overcome these limitations, prior studies have investigated improvements through neural network-based value approximation [5], heuristic-based Q-value initialization [6], and dimensionality reduction techniques [7]. Building upon these efforts, this work introduces an enhanced Q-learning algorithm that incorporates attractive and repulsive force functions into the reward mechanism, aiming to improve convergence efficiency and path safety for AGV navigation in grid-based warehouse environments. Despite significant advancements in the application of Q-learning for path planning, key research

DOI: 10.54254/2755-2721/2025.LD24727

priorities in mobile robot path planning continue to include improving convergence speed, reducing learning time, and minimizing operational costs. To enhance the performance of Automated Guided Vehicle (AGV) path planning, an advanced Q-learning algorithm has been developed by integrating attractive and repulsive force components derived from the artificial potential field method. The approach refines the reward structure and updates Q-values adaptively to improve convergence and obstacle avoidance. Simulation results demonstrate that the proposed method outperforms standard Q-learning in terms of learning efficiency, path safety, and convergence speed. The algorithm offers a practical and computationally efficient solution for autonomous navigation in warehouse grid environments.

2. The Q-learning path planning method

2.1. The core mechanism of reinforcement learning

The fundamental mechanism of reinforcement learning lies in the continuous interaction with the environment, encompassing both exploration and exploitation processes. Through the process of trial and error, the behavior strategy of the agent is gradually adjusted and optimized by means of a preset reward function. This iterative process aims to discover the optimal strategy, ensuring that the agent can obtain the maximum cumulative reward, and thus achieve the goal of maximizing the reward or fulfilling the preset objective. In reinforcement learning algorithms, the Markov Decision Process (MDP) provides a fundamental model framework . The state, action, strategy, and reward of the agent constitute four key elements. Its basic process is shown in Figure 1.

Figure 1: The basic process of the reinforcement learning algorithm

2.2. The Q-learning algorithm

The Q-learning algorithm is primarily employed to address problems related to Markov Decision Processes (MDPs) and is a temporal difference (TD) method in the field of reinforcement learning. When applied to AGV (Automated Guided Vehicle) path planning scenarios, Q-learning enables the acquisition of an optimal path strategy, allowing the AGV to efficiently navigate from a starting point to a destination within a warehouse environment based on task requirements. Within the Q-learning framework, the Q-values corresponding to state-action pairs are stored in a Q-table, which records the expected cumulative rewards for executing specific actions in given states. The Q-value reflects the

DOI: 10.54254/2755-2721/2025.LD24727

anticipated reward obtainable by performing a particular action in the current state. The Q-value update formula is expressed as follows:

$$Q\left(s,a\right)=Q\left(s,a\right)+\alpha\left[R\left(s,a\right)+\gamma\max_{a'}Q\left(s',a'\right)-Q\left(s,a\right)\right] \tag{1}$$

In Equation (1), s denotes the current state of the AGV, and a represents the action selected in that state. $Q(s,a)$ is the Q-value associated with taking action a in state s; α is the learning rate ($0<\alpha\leq1$), which controls the influence of new information on existing Q-values; $R(s,a)$ denotes the immediate reward received after executing action a; γ is the discount factor ($0<\gamma\leq1$), reflecting the importance of future rewards; s' indicates the next state reached after performing action a; and $\max_{a'}Q\left(s',a'\right)$ represents the highest Q-value over all possible actions a' in state s'.

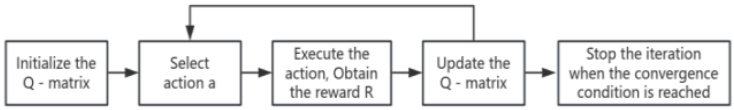

Figure 2: The Q-learning algorithm

2.3. Attractive force

Generated by the target point, this force always points toward the goal and guides the AGV's movement. Its intensity increases with the distance between the AGV and the target.

$$U_{att}=\frac{1}{2}\eta\rho^{2}(q-q_{g}) \tag{2}$$

In the equation(2), η is the gravitational gain coefficient, $\rho^{2}(q-q_{g})$ represents the distance between the current position of the AGV and the target position."

2.4. Repulsive force

Generated by obstacles, this force rapidly intensifies as the AGV approaches obstacles, effectively preventing collisions.

$$U_{rep}=\begin{cases}\frac{1}{2}\eta\left(\frac{1}{\rho(q-q_{0})}-\frac{1}{\rho_{0}}\right)^{2} & \rho(q-q_{0})\leq\rho_{0}\\ 0 & \rho(q-q_{0})>\rho_{0}\end{cases} \tag{3}$$

In the equation(3), η is the repulsive gain coefficient, while ρ_{0} represents the maximum distance within which obstacles affect the robot. The term $\rho^{2}(q-q_{g})$ indicates the distance from the AGV to the current point in the obstacle area.

3. Improved Q-learning algorithm for AGV path planning

3.1. Enhanced Q-learning algorithm

In grid-based path planning, the traditional Q-learning algorithm treats each grid as a feasible state, with an action space comprising movements in four directions (up, down, left, right) and a step size of

DOI: 10.54254/2755-2721/2025.LD24727

one grid unit. However, this algorithm exhibits limitations in grid map path planning, such as slow convergence speed, high computational costs, and prolonged runtime. To address these issues, this study enhances the traditional Q-learning framework by incorporating the attractive and repulsive force functions from the artificial potential field method. On the one hand, the reward function is optimized using the attractive force to provide heuristic guidance for the AGV, enabling it to maintain a clear directional focus toward the target. On the other hand, a correction term derived from the repulsive force is introduced to adjust the Q-table update formula, incentivizing the AGV to prioritize positions farther from obstacles during navigation.

3.2. Dynamic reward selection

First, determine whether the current state is a terminal state or an obstacle state. If the current state is neither, compute the attractive force function values U_{att} for the current state and the previous state U'_{att}, and compare their magnitudes. If U_{att} of the current state is greater than U'_{att} of the previous state, this indicates that the robot has moved farther from the target after the state transition. By dynamically determining the reward value through the comparison of U_{att} between the current and previous states, the robot is guided to move purposefully toward the target direction and select the next state strategically, thereby avoiding aimless exploration of every position. To incentivize the robot to learn obstacle avoidance and reach the target efficiently, a higher reward value is assigned to reaching the terminal state, while a lower reward is set for encountering obstacles. Based on this rationale, the reward function can be formulated as follows:

$$R\left(s,a\right)=\begin{cases}200 & \text{if the checked state is the goal state}\\ -50 & \text{if the checked state is an obstacle state}\\ -1 & \text{if } U'_{att}<U_{att}\\ 1 & \text{if } U'_{att}>U_{att}\end{cases} \tag{4}$$

3.3. Q-value update after calculating the γ value

First, determine whether the current state is the terminal state or the obstacle state. If the current state is neither the terminal state nor the obstacle state, calculate the repulsive force function U_{rep} of the current state and the repulsive force function U_{rep} of the previous state. Then, compare the values of U_{rep} and U'_{rep}, and check whether the repulsive force function U_{rep} of the current state is greater than the set threshold of 1. If the repulsive force function U_{rep} of the current state is greater than the repulsive force function U'_{rep} of the previous state, it indicates that the robot has moved farther away from the obstacle after the state transition. On the other hand, if the repulsive force function U_{rep} of the current state is less than 1, it means that the robot is currently far from the obstacle, and the repulsive force value is small enough to be negligible. Therefore, only when the repulsive force function U_{rep} of the current state is greater than the repulsive force function U'_{rep} of the previous state and the repulsive force function U_{rep} of the current state is greater than the set value of 1, the correction value is calculated dynamically. Then, substitute this correction value into the Q - table update formula, so that after the Q - table is updated, the robot is more likely to choose a state position farther away from the obstacle. This significantly enhances the robot's ability to avoid obstacles. The calculation formula for the correction value is as follows.

$$\lambda\left(s,a\right)=\begin{cases}\frac{\log_{10}U'_{rep}}{\log_{10}U_{rep\_max}}, & \text{if } U_{rep}<U'_{rep} \text{ and } U'_{rep}>1\\ 0, & \text{otherwise}\end{cases} \tag{5}$$

DOI: 10.54254/2755-2721/2025.LD24727

Improved Q-value Update Equation:

$$Q\left(s,a\right) \leftarrow Q\left(s,a\right) + \alpha \left[R\left(s,a\right) - \lambda\left(s,a\right) + \gamma \max_{a'} Q\left(s',a'\right) - Q\left(s,a\right)\right] \tag{6}$$

3.4. Steps of algorithm implementation

1) Step 1: Initialize Environment and Parameters Determine the size of the grid environment, define the positions of the start point, end point, and obstacles. Select appropriate values for the learning rate α, discount factor γ, set the ϵ value in the ϵ-greedy strategy, and specify the maximum number of iterations episodes

2) Step 2: Initialize the Q-table by setting all Q(s, a) values to 0.

3) Step 3: Initialize the state s by placing the robot at the start point.

4) Step 4: Check if it is the terminal state. If the current state s is the terminal state, obtain the reward and update the Q-value using the formula:

$$Q\left(s,a\right) \leftarrow Q\left(s,a\right) + \alpha \left[R\left(s,a\right) + \gamma \max_{a'} Q\left(s',a'\right) - Q\left(s,a\right)\right] \tag{7}$$

Then go to Step 10; else go to Step 5

5) Step 5: At the current state s, the robot selects and carries out action a according to the ϵ - greedy strategy, and then updates the state from s to s'

6) Step 6: Check whether the next state s' is an obstacle state. In case s' is an obstacle state, update the Q - value with the formula:

$$Q\left(s,a\right) \leftarrow Q\left(s,a\right) + \alpha \left[R\left(s,a\right) + \gamma \max_{a'} Q\left(s',a'\right) - Q\left(s,a\right)\right] \tag{8}$$

Return to the previous state s and re - execute Step 5; otherwise, execute Step 7。

7) Step7: Calculate the value of the gravitational function and dynamically select the reward value. Check whether the next state s' is an obstacle state. If s' is not an obstacle state, calculate the gravitational function U_{att} of the previous state and the gravitational function U'_{att} of the updated state. Dynamically select the reward value based on the comparison of the values of the two gravitational functions.

8) Step 8: Compute the values of the repulsive function and the correction value. Calculate the repulsive function U_{rep} of the previous state and the repulsive function U'_{rep} of the updated state. By comparing the values of the two repulsive functions, dynamically compute the correction value as follows:

$$\lambda\left(s,a\right) = \begin{cases} \frac{\log_{10} U'_{rep}}{\log_{10} U_{rep\_max}}, & U_{rep} < U'_{rep} \quad U'_{rep} > 1 \\ 0, & otherwise \end{cases} \tag{9}$$

9) Step 9: Calculate and update the Q-value. Based on the current state s, action a, reward value $R(s,a)$, discount factor γ, and the maximum Q-value of the next state s', update the value $Q(s,a)$ in the Q-table. The update formula is:

$$Q\left(s,a\right) \leftarrow Q\left(s,a\right) + \alpha \left[R\left(s,a\right) - \lambda\left(s,a\right) + \gamma \max_{a'} Q\left(s',a'\right) - Q\left(s,a\right)\right] \tag{10}$$

DOI: 10.54254/2755-2721/2025.LD24727

After the update, return to Step 4.

10) Step 10: Check whether the maximum number of episodes has been reached by verifying if the current episode has hit the predefined maximum iteration limit. If the limit is reached, terminate the learning process; otherwise, return to Step 3 to begin a new iteration.

4. Simulation environment construction

4.1. Grid map environment

In the simulation of the warehouse AGV environment, the two-dimensional grid method is adopted for modeling, as shown in the figure 2. All subsequent path planning will be carried out on this grid map. The grid method can simplify complex environmental problems into relatively simple small problems and is suitable for path planning in static environments. This method involves a relatively small amount of calculation and is easy to implement.

5. Experimental results and analysis

A. To evaluate the performance of the improved Q-learning algorithm, a simulation experiment was conducted using MATLAB (R2024a) was used as the development tool, and the grid map method was employed to simulate and model the intelligent warehouse AGV environment. The simulation environment is a 20×20 grid map, with the lower-left corner as the origin, a two-dimensional coordinate system was constructed, where the horizontal direction is the X-axis and the vertical direction is the Y-axis. As shown in Figure 3, the "Start" mark in the figure indicates the starting position of the mobile robot, and the "Goal" mark indicates the target position. The white area is the region where the robot can move freely, while the solid black squares represent the non-traversable obstacle areas. The action space of the robot consists of four directions, namely up, down, left, and right. The entire grid map has a total of 400 grids, corresponding to 400 states in the Q-learning algorithm. The starting point is located at coordinates (1,1), and the ending point is at (20,20).

DOI: 10.54254/2755-2721/2025.LD24727

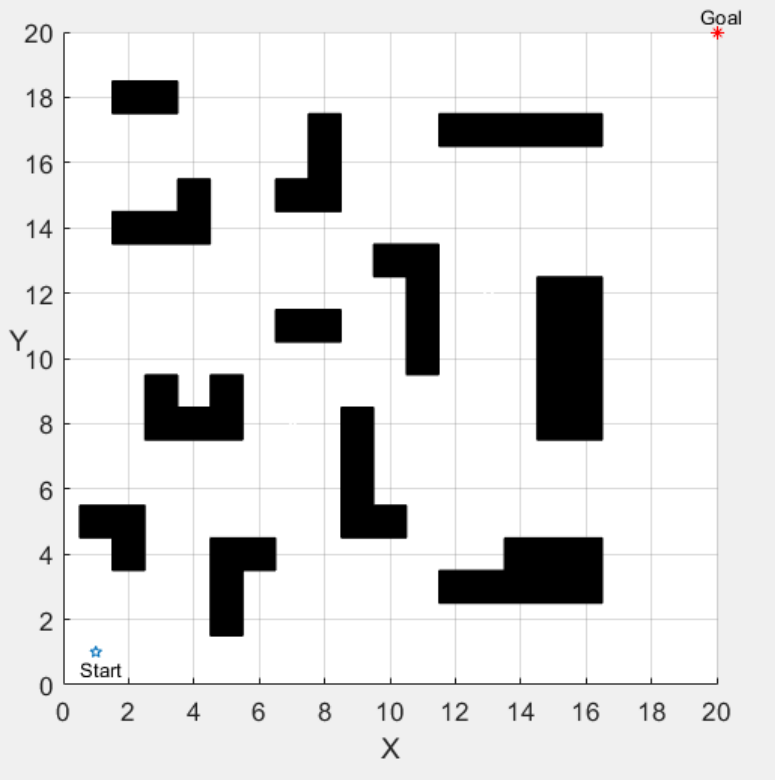

Figure 3: Simulated warehouse environment grid map

Based on the above experimental environment and parameters, experimental simulations were successively carried out for the traditional Q - learning algorithm and the algorithm proposed in this study on a specific grid map. When the Q - learning algorithm is integrated with the gravitational potential field function, the directionality of the AGV's movement is significantly enhanced. The probability of the AGV moving towards the target point increases substantially, and the number of blind searches is effectively reduced. The relevant experimental results are shown in the figures on the next page. As can be seen from Figure 4, the traditional Q - learning algorithm tends to converge when the number of iterations reaches 100. Figure 5 shows that the algorithm proposed in this study converges when the number of iterations is 60. By comparing the figures, it can be found that before convergence, the traditional Q - learning algorithm consumes more steps for exploration, while the improved Q - learning algorithm in this study uses far fewer steps. This fully demonstrates that in the same simulation environment, the improved Q - learning algorithm in this study is more purposeful during the AGV exploration process. It can quickly determine the next position, thereby significantly reducing the exploration steps of the AGV and achieving rapid convergence. Figures 6 and 7 show the path - planning route maps obtained by the two algorithms on the grid map. "Start" represents the starting position, "Goal" is the ending position, and the black circles indicate the obstacle - avoidance paths of the mobile robot. From Figures 4 to 7, it can be seen that the algorithm adopted in this paper performs excellently in multiple aspects. Compared to other common algorithms, it not only has a significant improvement in operating efficiency but also consumes fewer resources and has a lower cost when completing the learning process. In addition, this algorithm also exhibits strong obstacle - avoidance performance. In complex environments, it can effectively avoid collisions with obstacles, thus ensuring that the mobile AGV can quickly and safely find a collision - free path to reach the target position smoothly.

DOI: 10.54254/2755-2721/2025.LD24727

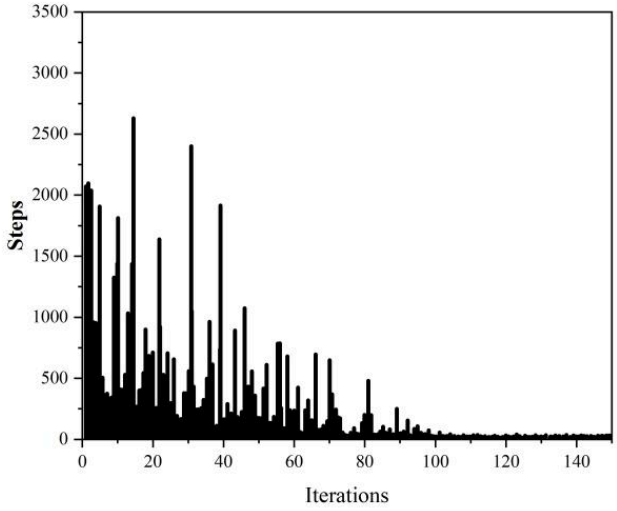

Figure 4: Traditional Q-learning algorithm iteration convergence plot

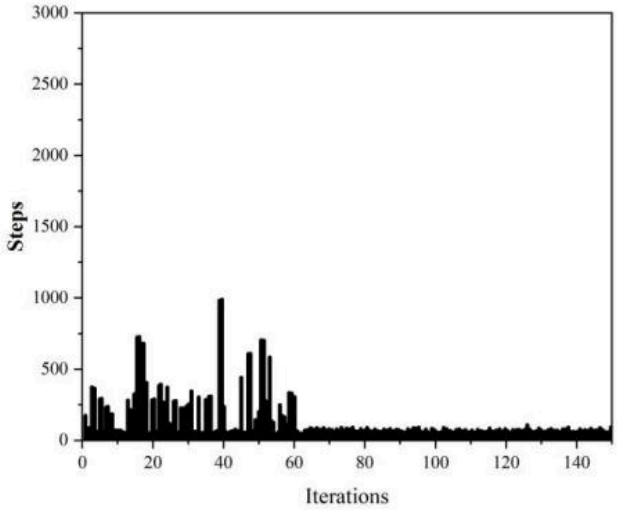

Figure 5: Improved Q-learning algorithm iteration convergence plot

DOI: 10.54254/2755-2721/2025.LD24727

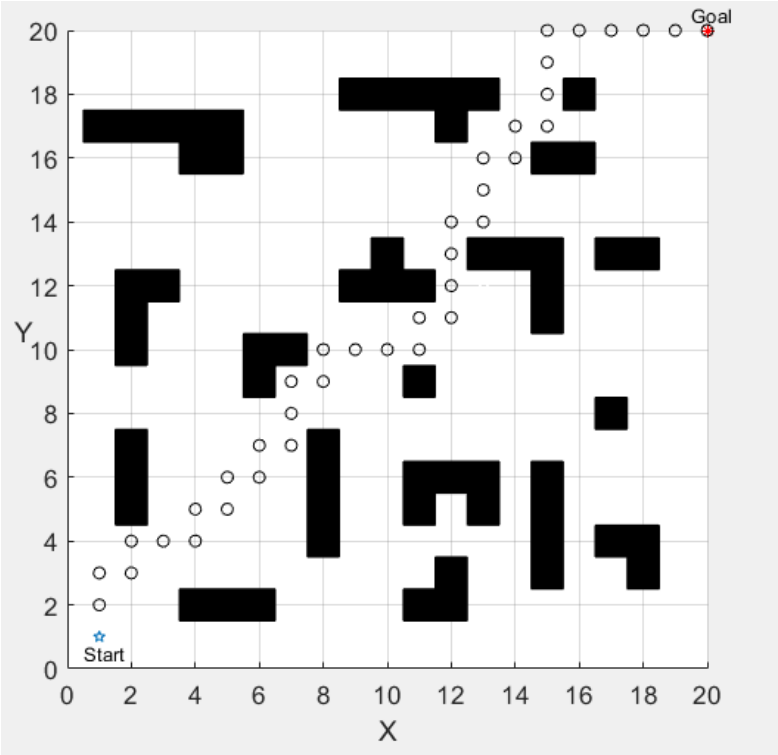

Figure 6: Traditional Q-learning algorithm path planning route map

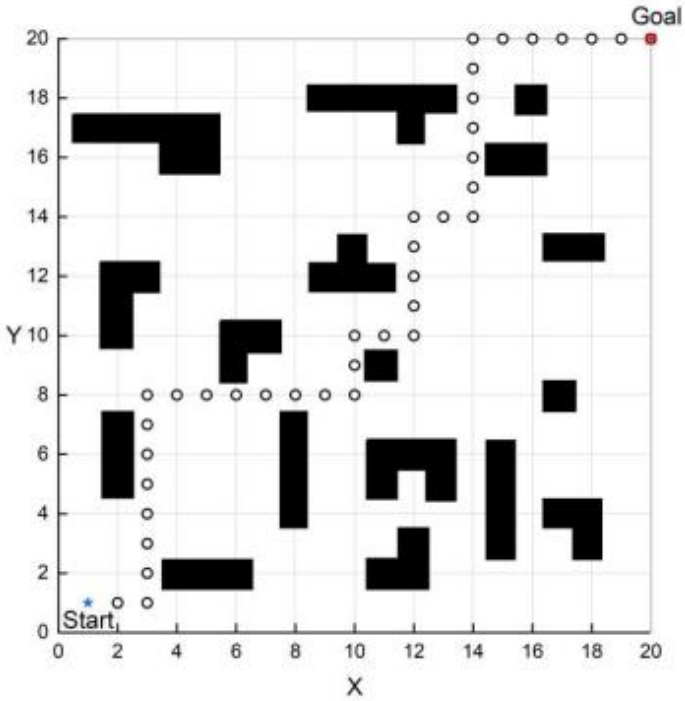

Figure 7: Improved Q-learning algorithm path planning route map

DOI: 10.54254/2755-2721/2025.LD24727

6. Conclusion

This paper proposes a path planning algorithm tailored for warehouse Automated Guided Vehicle (AGV) motion environments, which is based on grid-based simulation and improves the Q-learning algorithm. The enhanced Q-learning algorithm introduces attractive and repulsive force functions to achieve dynamic reward value selection and Q-value updates. The attractive force function enables the AGV to maintain a clear directional focus during exploration, avoiding aimless searches, while the repulsive force function dynamically adjusts Q-values to guide the AGV away from obstacles, thereby accelerating and improving navigation accuracy to the target. Simulation results demonstrate that the Q-learning algorithm integrated with potential field functions exhibits faster convergence speed, lower computational cost, higher operational efficiency, and significantly reduces the probability of collisions with obstacles. These advantages make the proposed algorithm superior in practical AGV path planning applications.

References

[1] Huang, X., & Li, C. G. (2022). Global path planning based on a bidirectional alternating search A* algorithm for mobile robots. Computers & Industrial Engineering, 168, 108123.
[2] Suresh, K. S., Venkatesan, R., & Venugopal, S. (2022). Mobile robot path planning using multi-objective genetic algorithm in industrial automation. Soft Computing, 26(15), 7387-7400.
[3] Xin, J., Li, Z., Zhang, Y., et al. (2024). Efficient real-time path planning with self-evolving particle swarm optimization in dynamic scenarios. Unmanned Systems, 12(2), 215-226.
[4] Watkins, C. J. C. H., & Dayan, P. (1992). Q-learning. Machine Learning, 8, 279-292.
[5] Gaskett, C. (2002). Q-learning for robot control (Doctoral dissertation). The Australian National University, Canberra.
[6] Dung, L. T., Komeda, T., & Takagi, M. (2008). Reinforcement learning for POMDP using state classification. Applied Artificial Intelligence, 22(7-8), 761-779.
[7] Diao, A. (2017). Fast realization of ECG signal correlation dimension based on Labview. Chinese Medical Equipment, 14(12), 23-26.