Three-stage Path Planning for Warehouse Robots Based on Q-learning Algorithm

Junjie He *

Southampton Ocean Engineering Joint Institute, Harbin Engineering University, Harbin, 150001, China

* Corresponding Author Email: jh5c23@soton.ac.uk

Abstract. With the rapid growth of e-commerce, warehouse automation has become an important area to improve efficiency and reduce operational costs. Warehouse robots have been widely used by online merchants to automate the delivery process of warehouse logistics. However, traditional research lacks clear guidance on the optimal setting of parameters for Q-learning in a warehouse environment. And it does not consider path planning for multi-stage operations. This research examines the application of the Q-learning algorithm for three-stage path planning in robotic systems inside a warehouse setting. The research optimized the convergence rate of Q-learning in a warehouse by comparing the efficiency and stability of the Q-learning algorithm under different parameters. This research provides a three-stage warehouse transportation path planning strategy. This strategy is accomplished through a three-stage "inbound-shipping-return" path planning. It enables the robot to navigate efficiently through the warehouse layout, reducing transportation time and avoiding obstacles. This research demonstrates the effectiveness of Q-learning in warehouse path planning and shows that warehouse robots have great potential to save labor costs and improve the efficiency of warehouse transportation tasks. The Q-learning-based optimization method provides a solution for the automation of warehouse transportation tasks.

Keywords: Q-learning; Warehouse; Robot; Path planning.

1. Introduction

As e-commerce and automated warehouses advance swiftly, the significance of warehouse robots in item storage and movement is becoming paramount. Mobile robots are now widely used in automated warehouses [1]. Path planning algorithms are especially critical in robot scheduling and goods transportation. How to quickly and accurately develop traveling routes for robots in warehouse environments has become a hot topic in current research. Path planning not only affects the operation efficiency of robots but also directly relates to the overall operation cost and service quality of the warehouse [2].

Currently, several research have explored the application of warehouse robots and Q-learning algorithms. Adzhar et al. discussed the challenges of path planning in dynamic warehouse environments, such as obstacle movement, real-time update requirements, etc. The article also highlights the limitations of current algorithms in dealing with these challenges [2]. Peyas et al. in their paper show that deep Q learning outperforms traditional algorithms in dynamic environments. The potential of reinforcement learning in optimizing warehouse operations is emphasized and directions for research on multi-robot collaboration are indicated [3]. Rimélé et al. show the potential of deep Q-learning DQN for dynamic optimization of e-commerce warehousing systems [4].

Although existing research has provided many theoretical foundations for warehouse robot path planning, there are still many under-researched aspects. First, the parameter selection of Q-learning algorithms has a significant impact on the final results, and the existing literature has explored fewer optimal parameter settings in warehouse environments. Second, the multistage Q-learning path planning problem is less deeply covered in existing research.

To address the above problems, this research experimentally compares the convergence speed and stability of the algorithm under different parameters, aiming to optimize the parameter settings of the Q learning algorithm. A new three-stage path planning framework is proposed in this paper, which seeks to provide a more effective solution for warehouse robot path planning. In addition, this

research will also consider the dynamic changes of multi-segmented tasks to enhance the adaptability of the algorithm in practical applications.

The research provides an in-depth discussion on parameter optimization of Q-learning algorithms in warehouse environments and proposes an adaptable multi-stage path planning method. Through this research, it is expected to provide suggestions for optimal parameter settings and ideas for multi-stage path planning for warehouse automation, to promote further development in related fields.

2. Methodology

2.1. The Q-learning Algorithm and Principle

2.1.1. Q-learning

Reinforcement learning (RL) is a machine learning method. This method optimizes the behavioral strategy of an agent through its interaction with the environment. It ultimately leads to maximizing cumulative rewards or reaching a specific goal [5]. Q-learning is a classical model-free learning algorithm based on RL, which learns the optimal policy by directly parameterizing and updating the value function [6].

The core idea of Q-learning is to learn the state-action value function (i.e., Q-value function) through interaction with the environment. This function is used to guide the agent to select the optimal action that brings the maximum cumulative reward in a given state. At each step of the decision-making process, the agent selects an action based on the Q-value, receives an immediate reward from the environment, and uses that reward to update the Q-value. The process continues to iterate until the Q-value converges, resulting in an optimal policy [7].

2.1.2. Principle of the model

State, action, reward, and policy update are the key elements of Q-learning [7]. These elements define the mechanism by which the algorithm works when interacting with the environment.

In warehouse robot path planning, the state denotes the robot's position information in the warehouse and the condition of the surrounding environment, such as the location of obstacles and the layout of shelves. The action represents the behavior taken by the robot in a specific state (four-direction movement in this experiment). In each state, the robot has to choose an action that may lead to a higher cumulative reward based on the current Q-value table. The reward is a feedback signal that the robot receives from the environment after performing a specific action. The design of rewards directly affects the learning efficiency of the agent. In this experiment, rewards are designed based on whether the robot avoids obstacles and reaches the endpoint. This reward mechanism can effectively guide the robot to choose a better path.

The essence of Q-learning resides in policy modifications: the Bellman Equation is employed to revise the Q-value. The Bellman Equation states that the Q-value of acting in the current state depends not only on the immediate reward but also on the future Q-value. In Q-learning, the Bellman equation takes the following form:

$$\mathrm{Q}(\mathrm{s}, a) = \mathrm{r}(\mathrm{s}, a) + \gamma \max_{a'} \mathrm{Q}(\mathrm{s}', a') \tag{1}$$

Where $\mathrm{Q}(\mathrm{s}, a)$ denotes the Q-value associated with the execution of action a in states, i.e., the expected total return of the warehouse robot moving in a certain direction in the current state. $\mathrm{r}(\mathrm{s}, a)$ Denotes the instant reward for performing action a in states. $\max_{a'} \mathrm{Q}(\mathrm{s}', a')$ Denotes the highest Q-value corresponding to the action a' that can be executed in the subsequent states′, indicating the payoff of the optimal future action.

In Q-learning, the Bellman equation is not computed directly but is approximated by gradual iterations. The algorithm updates the Q-value by using the following formula:

$$\mathrm{Q}(\mathrm{s}, a) \leftarrow \mathrm{Q}(\mathrm{s}, a) + \alpha[\mathrm{r} + \gamma \max_{a'} \mathrm{Q}(\mathrm{s}', a') - \mathrm{Q}(\mathrm{s}, a)] \tag{2}$$

Where α is the learning rate and γ is the discount factor, both with values between 0 and 1 [8].

2.2. Data Sources and Model Evaluation

2.2.1. Warehouse Layout Data

In warehouse automation, Q-learning-based robot path planning experiment requires environmental data to train and validate models. Warehouse layout data is the core input for path planning, which defines the physical structure of the warehouse environment. These data describe the space and constraints under which the robot may move. To accelerate the convergence of the path planning, this experiment first simplifies the modeling of the warehouse environment by rationally dividing the state space. This method effectively reduces the number of states and decreases the computational complexity. Fig. 1 shows the warehouse layout data:

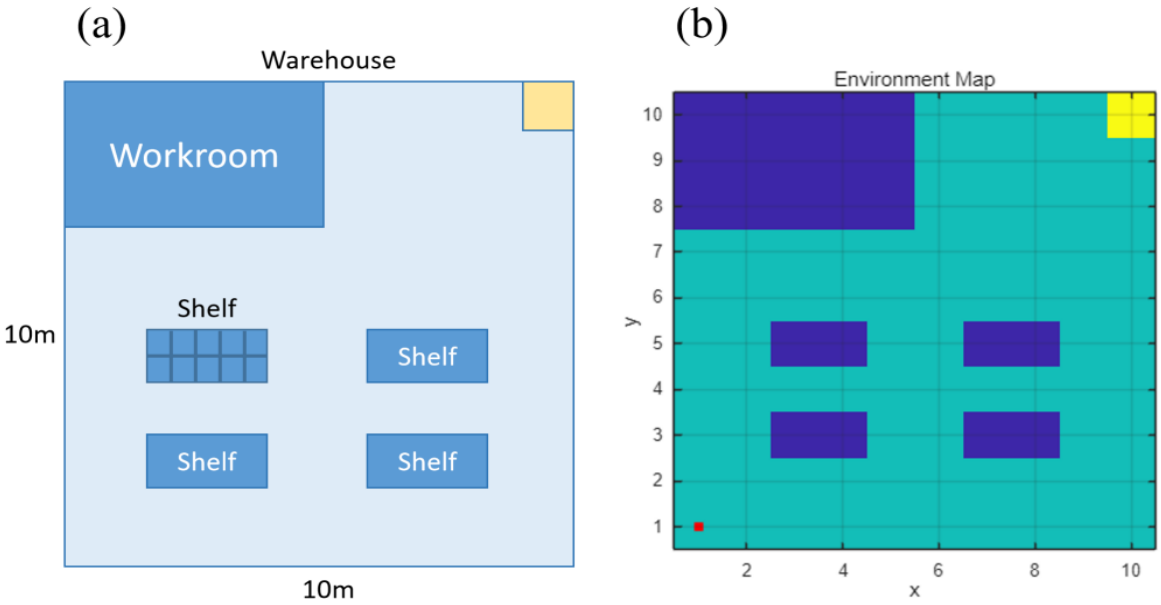


Fig 1. Warehouse layout data. (a) Simplified warehouse map.
(b) Warehouse environment grid map. (Photo/Picture credit: Original).

Fig. 1 (a) shows the map of the 10*10m simple warehouse designed for this experiment. The warehouse is planned to have a workroom, shelves, and shipping outlets. The size of the workroom is 3*5m and the size of the shelves is 2*1m. The shipping gate is located at the warehouse coordinates (10, 10), and the starting point of the robot is located at (1, 1). Fig. 1(b) shows a two-dimensional grid map modeled in MATLAB, where each cell is a state, and the robot can perform one of the four actions, left, right, up, or down, in each cell.

2.2.2. Metrics to evaluate the model

Path Length and Convergence Speed are the experiment's evaluation measures, which are used to assess how well the Q-learning-based path planning model performs. These evaluation metrics can reflect the efficiency and stability of the model in the warehouse environment application more comprehensively and provide an important reference for optimizing the Q-learning algorithm.

Path Length is a measure of the total distance traveled by a robot in completing tasks. The shorter the path, the more efficient the robot is in accomplishing the task. Therefore, the success of a single experiment is based on whether the algorithm finds the shortest path. Convergence speed is the primary measure used to evaluate the model for this experiment, and it represents the number of iterations necessary for Q-learning to identify the best strategy. Q-learning gradually approaches the optimal policy by updating the Q-value several times, and fewer iterations mean faster convergence and more efficient path planning. A model with fast convergence means the algorithm can adapt to new environments or tasks faster, improving the real-time decision-making ability of the warehouse robot.

3. Results and Discussions

3.1. Results of Q-learning Algorithm with Different Parameters (α, γ)

3.1.1. Design of experiments and prediction of results

The reward function plays a key role in Q-learning, which determines the learning rate of the algorithm. The learning rate α and discount factor γ in the function are the key parameters that affect the convergence speed and final path quality. In this experiment, the Q-learning algorithm was written in MATLAB and tested for path planning for the robot in a warehouse grid map. In the experiment, the algorithm planned the path for the warehouse robot from the starting point to the shipping gate to verify the feasibility of path planning. By adjusting the two parameters, the running results of the Q-learning algorithm under different parameter combinations are counted. The maximum number of episodes per training was set to 2000 and the maximum number of steps was 100. The results obtained were compared and summarized, and their effects on the convergence speed of the algorithm and the path-planning effect were analyzed to finally find out the optimal parameter intervals for robot path-finding planning within this warehouse environment. The pseudo-code for multi-week training in the Q-learning algorithm in the experiment is shown in Table 1.

Table 1. Principle of Q-learning.

Algorithm 1 Q-learning Multiple Training Episodes:
for each episode:
initialize starting state s' to [1, 1]
set $s = s'$, R = 0
for each timestep:
select action a based on ε-greedy policy
if the chosen action is to move left (right, up or down):
if within bounds and no obstacle:
if it is the goal:
set Reward = 1
else:
set Reward = 0
else:
set Reward = -1
update Q-table using Bellman equation:
$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$
save current state to construct path
while the goal is reached or the obstacle hit:
exit timestep loop
end
end

This experiment subdivided the two parameters into 9 levels (0.1-0.9 at intervals of 0.1). The two parameters learning rate α and discount factor γ were regulated at 9 levels for a total of 81 parameter combinations. Each parameter combination was trained 10 times, and the average value of whether or not it was related to the convergence rate was counted, for a total of 810 experimental data to be counted.

In order to reduce the complexity associated with testing 81 parameter combinations, this experiment used a staged statistics strategy. By fixing one parameter first and then gradually optimizing another parameter, the number of times the model was trained in the experiment is reduced. First, the optimal learning rate α interval was searched under several fixed values of the discount factor γ. By comparing the performance of the algorithm under each α value, the best α value interval is selected. After finding the optimal α value, compared the algorithm performance under different γ values to determine the optimal γ value interval. Through the above optimization process, the scope

of the experiment was narrowed down so that only a portion of the parameter combinations were tested to find the range of the optimal parameter combinations.

The discount factor γ directly affects how much the agent values long-term rewards and can control the trade-off between short-term and long-term rewards. In this experiment, the warehouse robot not only needed to find the optimal solution during the process but also needed to make sure that it can complete the whole task and avoid local optimization. Therefore, the long-term payoff of the task was more important than focusing only on the immediate benefits of each step.

In general, a discount factor γ as close to 1 as possible was considered optimal for tasks with significant long-term benefits [8]. Therefore, experiments expect superior results for higher values of γ. However, in recent years, it has been shown that in some cases better performance than before has been achieved with lower discount factors γ, and also that results with lower discount factors are superior to those with higher discount factors [9]. Therefore, in this experiment, the parameter discount factor γ was divided into three groups: low discount factor group {0.1, 0.2, 0.3}, medium discount factor group {0.4, 0.5, 0.6}, and high discount factor group {0.7, 0.8, 0.9}, and the median of the three groups (γ=0.2, γ=0.5, and γ=0.8) was chosen as the initial γ value. Using these three values allowed for a more complete exploration of the effects of learning rate with a reduced number of model training.

The learning rate α controls the proportion of past experience that is replaced by new information in each Q-value update. A larger value of α can make the replacement of old information by new information faster and may accelerate convergence, but too high a learning rate may cause the algorithm to be unstable during the learning process. Due to the large magnitude of each update, the agent may ignore the previous learning, which makes it difficult to converge to the optimal solution. While a smaller value of α will make the algorithm more stable, it may prolong the convergence time, especially in dynamic environments that cannot adapt to changes quickly [8]. In this experiment, the warehouse environment was relatively simple and without dynamic changes, so the optimal α value was expected to be between (0.1, 0.4).

3.1.2. Analysis of Efficiency and Reliability of Algorithms

After the statistics of the results of several experiments under different learning rates α and discount factors γ, the data on the average number of episodes for accomplishing optimal path planning were derived. These data reflected the convergence speed of the algorithm under different parameters. The number of times the same parameter combination failed to find the shortest path in 10 training sessions was also derived. The data was plotted as line graphs and histograms, tables, and graphs are presented in Table 2.

Table 2. Average number of episodes to complete the optimal path planning.

	γ=0.2	γ=0.5	γ=0.8	average
α=0.1	1042.00	1258.83	934.71	1078.52
α=0.2	874.29	928.00	984.00	928.76
α=0.3	773.44	997.75	1022.33	931.18
α=0.4	871.86	1122.40	1045.71	1013.32
α=0.5	856.43	955.22	1036.38	949.34
α=0.6	1094.71	1235.50	977.86	1102.69
α=0.7	1140.44	1007.43	1043.40	1063.76
α=0.8	997.88	1063.33	1014.71	1025.31
α=0.9	1452.00	1154.75	1072.63	1226.46

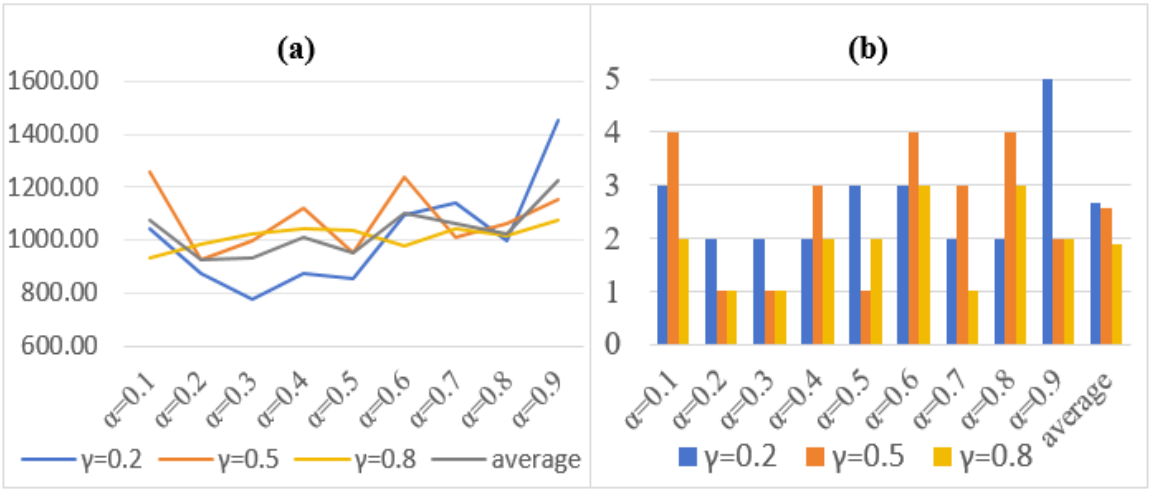


Fig 2. Comparison of algorithm training result data.
(a) Line graph of an average number of episodes. (b) Histogram of the number of failures to find the optimal path. (Photo/Picture credit: Original).

As can be observed from Fig. 2(a), the number of episodes required fluctuated between 1000 and 1200 for most learning rates. At too low a learning rate ($\alpha = 0.1$), the algorithm requires more episodes to find the optimal path and convergence slows down. At lower learning rates ($0.2 \leq \alpha \leq 0.5$), the algorithm required fewer episodes, with the average number of episodes falling below 1000. When the learning rate is too high ($\alpha \geq 0.6$), the number of episodes required by the algorithm increases significantly and the fold fluctuations become more chaotic. When $\alpha = 0.2$, the average number of episodes required by the Q-learning algorithm is the lowest. The low discount factor γ fluctuates sharply in the line graph. The higher the discount factor γ, the smoother the folds of the algorithm.

From Fig. 2(b), it can be observed that when α is too low ($\alpha = 0.1$), the algorithm fails to find the optimal path more often. When α is low ($0.2 \leq \alpha \leq 0.3$), the algorithm path planning fails less often. When α is too high ($\alpha \geq 0.6$), the number of algorithm failures rises significantly. The number of failures was usually higher under the low discount factor group (blue). While higher γ values (orange and yellow) perform better with fewer failures at $0.2 \leq \alpha \leq 0.5$. Failures generally increase when $\alpha \geq 0.6$ and the low discount factor group shows the greatest volatility.

Overall, lower learning rates ($0.2 \leq \alpha \leq 0.3$) had faster convergence and fewer failures. Higher learning rates ($\alpha \geq 0.6$) led to unstable training and more failures. Higher discount factors (e.g., $\gamma = 0.8$) typically led to fewer failures. It suggested that a greater emphasis on long-term payoffs improves the performance of the algorithm for long-term goal-oriented tasks. The data analysis revealed that the experimental and projected findings are in agreement. In this environment and task, the combination of a learning rate α between 0.2 and 0.3 and a discount factor γ around 0.8 seemed to be the optimal setting for efficient and successful path planning.

3.2. Three-stage Path Planning

3.2.1. Design of experiment

To ensure that warehouse robots have clear task planning and operate efficiently, a Three-stage path planning method was designed in this experiment. This method can help optimize the warehouse robot task management and cargo delivery. Based on the Q-learning path planning algorithm of the previous experiment, this experiment added a three-stage task flow and tested Three-stage path planning for the robot in a warehouse grid map.

Since the Three-stage path planning of this experiment is essentially a task extension of the previous experiment in the same environment, the optimal parameter combination should also be close to the previous experiment. Therefore, in this experiment, the discount factor γ was set to 0.8 and the learning rate α to 0.2. The maximum number of episodes and the maximum number of steps for each training were set to 2000 and 200, respectively. The average episode number and Failure

frequency of the experimental results were counted to verify the feasibility of the Q-learning algorithm in Three-stage path planning for warehouse robots. The performance of the algorithm and the effect of path planning were analyzed to finally conclude the advantages of Q-learning algorithm path planning.

3.2.2. Three-stage path planning

The Three-stage path planning approach divided path planning into three stages: from Start to Target, from Target to Exit, and from Exit back to Start. Each stage of path planning was adjusted by the Q-learning algorithm to ensure that the robot achieves the optimal path in the warehouse environment.

In stage I, the robot moves from an initial location (e.g., a warehouse inlet or a designated point) to a shelf location, where the robot can either deliver goods from the inlet to the shelf (stocking mode) or go to the shelf to pick up goods (picking mode). In this stage, the algorithm randomly selects a location on the shelf, and the algorithm needs to avoid obstacles and choose the shortest path for the robot to reach the goal point. In the case of stocking mode, the task will jump directly to stage III.

In stage II, after the robot successfully reaches the target point and completes the stocking or picking task, the next step is to plan the path to deliver the goods to the shipping gate. The goal of this stage is to ensure that the robot safely and quickly reaches the warehouse shipping port to improve shipping efficiency.

In stage III, after completing the cargo transportation task, the robot needs to return to the initial position to prepare for the next task. Similar to the previous two stages, the algorithm needs to adjust the path in the environment and avoid other obstacles in the warehouse as much as possible, and find the shortest path back to the starting point.

After many experiments to verify that the Q-learning algorithm can effectively complete Three-stage path planning, Fig. 3 shows the Three-stage optimal path planned by the Q-learning algorithm in a certain training:

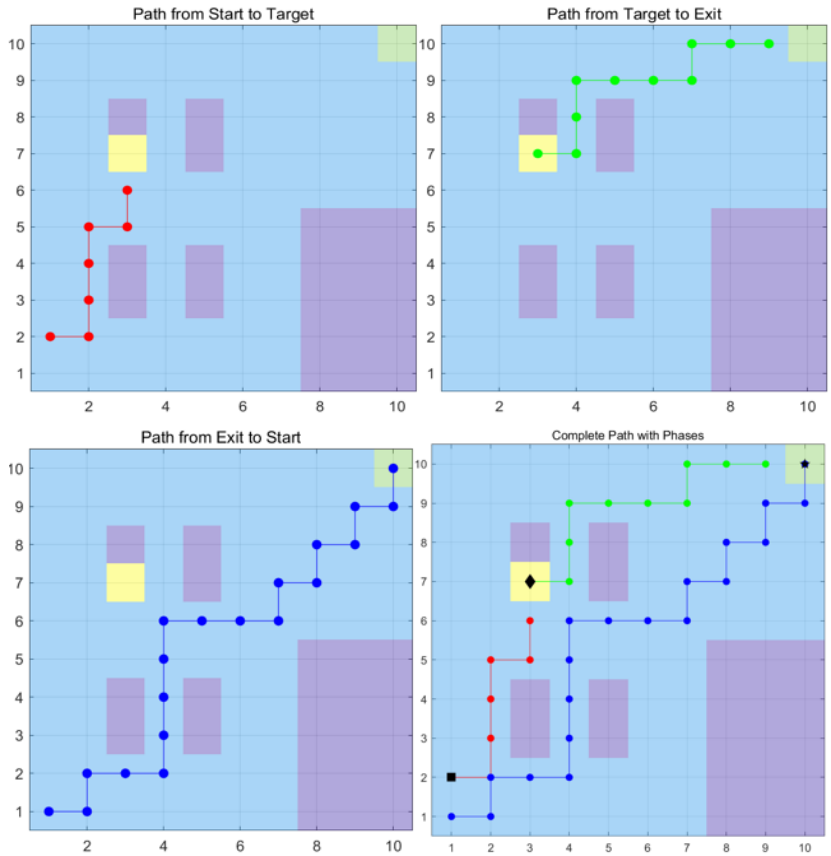
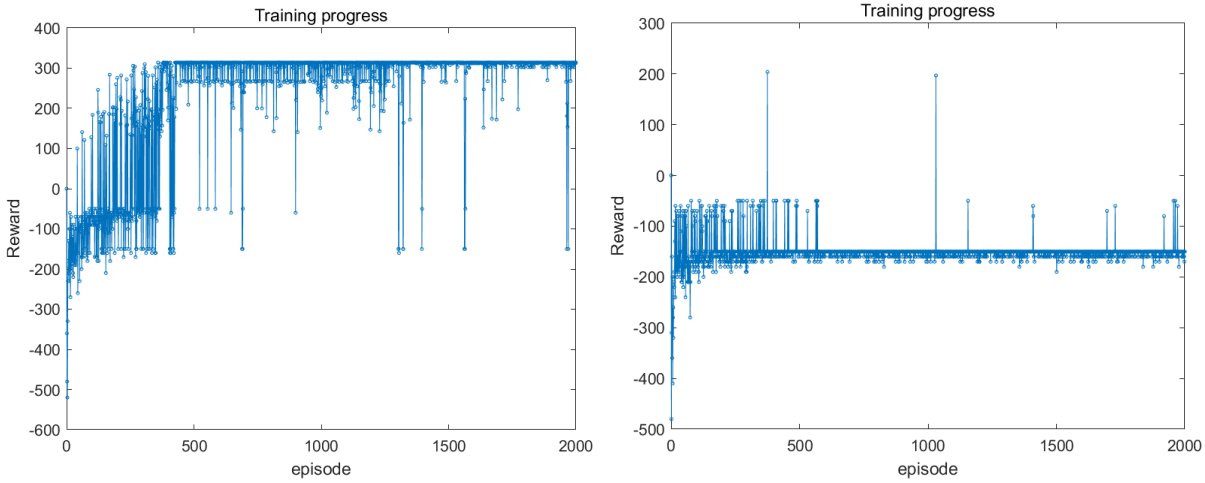


Fig 3. Three-stage optimal path. (Photo/Picture credit: Original).

3.2.3. Average number of episodes completing the optimal path & failure frequency

In the experiments of three-stage path planning, the average number of episodes to complete the optimal path and the failure frequency are the key evaluation indexes of the algorithm performance. After several model training, a series of experimental result data were statistically obtained.

Fig. 4(a) shows the convergence of the Q-learning algorithm three-stage path planning in a certain training. The algorithm's speed at finding the best course of action is shown by the number of episodes needed for convergence; the more episodes the algorithm requires to converge, the faster it does so in that environment. The number of times the robot failed to finish the assignment in the allotted maximum number of episodes (2000) is the failure frequency. The graphic illustrates a significant reward departure from 300, which indicates the stability and resilience of the path design. Fig. 4(b) shows the failure of the algorithm path planning under non-optimal parameter combinations.



(a) (b)

Fig. 4 Training process of three-stage path planning for Q-learning.
(a) Algorithmic convergence of three-stage path planning. (b) Algorithmic convergence failure cases for three-stage path planning ($\alpha = 1$, $\gamma = 1$). (Photo/Picture credit: Original)

By organizing and analyzing the experimental data, the failure frequency at the beginning of training is relatively high, which is mainly due to the more random action selection of the Q-learning algorithm at the initial stage of the learning process [10]. As the training progressed, the Q-value table was constantly updated, and the robot gradually learned how to avoid obstacles, choose the shortest path, and optimize its actions according to the reward mechanism. At around 250 episodes, the algorithm began to converge gradually, the episodes required to successfully find the optimal path decreased significantly, and the failure frequency also decreased significantly. The experimental data was plotted as a histogram, and the episodes of three-stage path planning were displayed in Fig. 5.

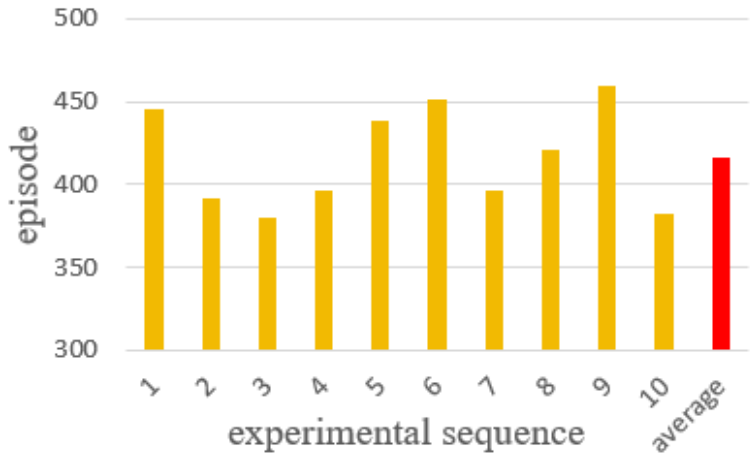


Fig 5. Episodes of three-stage path planning. (Photo/Picture credit: Original).

Fig. 5 shows that the Q-learning algorithm performed better in the three-stage path-planning task as evidenced by the fact that the number of episodes in the three-stage path-planning task ranges from 350 to 450. The algorithm's path planning did not fail in any of the ten training sessions, which also demonstrates the reliability of the Q-learning algorithm in accomplishing similar tasks in similar environments. This experimental data proved that Q-learning multi-stage path planning has very high potential.

4. Conclusion

In this research, a simplified warehouse environment was first designed to use the Q-learning algorithm for robot path planning. By adjusting the parameter learning rate α and discount factor γ of the Q-learning algorithm, the convergence speed of the algorithm and the path optimization effect of different parameters are compared and analyzed. Then, this research proposed a three-stage path planning strategy for the warehouse environment to optimize the path selection of warehouse robots in multi-stage tasks. The research idea focused on dividing the warehouse robot path planning into three stages. By means of reinforcement learning, the robot autonomously learned the optimal path planning so as to improve transportation efficiency.

The research's results demonstrated that the stability and effectiveness of the Q-learning algorithm can be raised with a suitable parameter configuration. The three-stage path planning strategy can effectively complete the experiment, specifically, the optimized algorithm not only accelerated the convergence speed but also reduced the robot's path deviation. It demonstrated the good applicability and effectiveness of the method in warehouse automation tasks. This research provided case support and practical value for improving warehouse logistics efficiency and intelligent robot path planning.

Despite the encouraging results, this research still has some limitations. First, this research used a simplified warehouse environment, but the actual warehouse is more complex. Therefore, future research should consider more complex dynamic warehouse environments to verify the robustness and applicability of the Q-learning algorithm. Second, the research focused on the path planning of individual warehouse robots and ignored the collaboration problem in multi-robot systems. To enhance the algorithm's performance in more intricate application settings, future research can investigate the problems of multi-robot collaborative optimization, dynamic path planning, and path adjustment under real-time environment changes.

References

[1] Zhang H, Lin W, Chen A. Path planning for the mobile robot: A review. Symmetry, 2018, 10(10): 450.

[2] Adzhar N, Yusof Y, Ahmad M A. A review on autonomous mobile robot path planning algorithms. Advances in Science, Technology and Engineering Systems Journal, 2020, 5(3): 236-240.

[3] Peyas I S, Hasan Z, Tushar M R R, et al. Autonomous warehouse robot using deep Q-learning. TENCON 2021-2021 IEEE Region 10 Conference (TENCON). IEEE, 2021: 857-862.

[4] Rimélé A, Grangier P, and Gamache M, et al. E-commerce warehousing: learning a storage policy. arXiv preprint arXiv:2101.08828, 2021.

[5] Sutton R S. Reinforcement learning: An introduction. A Bradford Book, 2018.

[6] Phan M Q, Azad S M B. Input-decoupled Q-learning for optimal control. The Journal of the Astronautical Sciences, 2020, 67(2): 630-656.

[7] Watkins C J C H, Dayan P. Q-learning. Machine learning, 1992, 8: 279-292.

[8] Wagenbach J, Sabatelli M. Factors of influence of the overestimation bias of Q-learning. arXiv preprint arXiv:2210.05262, 2022.

[9] Van Seijen H, Fatemi M, Tavakoli A. Using a logarithmic mapping to enable lower discount factors in reinforcement learning. Advances in Neural Information Processing Systems, 2019, 32.

[10] Haarnoja T, Zhou A, Abbeel P, et al. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. International conference on machine learning. PMLR, 2018: 1861-1870.