

Received 7 August 2022, accepted 25 August 2022, date of publication 31 August 2022, date of current version 9 September 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3203072

RESEARCH ARTICLE

Improved Q-Learning Applied to Dynamic Obstacle Avoidance and Path Planning

CHUNLEI WANG¹, XIAO YANG², AND HE LI³

¹School of Public Administration, China University of Mining and Technology, Xuzhou, Jiangsu 221116, China

²School of Computer Science and Technology, Harbin Engineering University, Harbin, Heilongjiang 150001, China

³Department of Languages and Communication Studies, Beijing Jiaotong University, Beijing, Haidian 100044, China

Corresponding author: Xiao Yang (yangxiaoxiaovip@126.com)


This work was supported by the Jiangsu Social Science Foundation of China under Grant 21YYB014.

ABSTRACT Due to the complexity of interactive environments, dynamic obstacle avoidance path planning poses a significant challenge to agent mobility. Dynamic path planning is a complex multi-constraint combinatorial optimization problem. Some existing algorithms easily fall into local optimization when solving such problems, leading to defects in convergence speed and accuracy. Reinforcement learning has certain advantages in solving decision sequence problems in complex environments. A Q-learning algorithm is a reinforcement learning method. In order to improve the value evaluation of the algorithm in solving practical problems, this paper introduces the priority weight into the Q-learning algorithm. The improved algorithm is compared with existing algorithms and applied to dynamic obstacle avoidance path planning. Experiments show that the improved algorithm dramatically improves the convergence speed and accuracy and increases the value evaluation. The improved algorithm finds the shortest path of 16 units in 27 seconds.

INDEX TERMS Dynamic obstacle avoidance, sequence problems, reinforcement learning, Q-learning algorithm.

I. INTRODUCTION

In recent years, with the development of the IT industry and the proposal of a large number of intelligent algorithms, these algorithms have been applied to path planning technology and are relatively mature. However, path planning still depends on manual sites to a great extent. This planning method will be affected by subjective factors, resulting in the consumption of the workforce, material resources, and working time. Therefore, in the era of industry-wide intelligence, it is of great significance to apply an intelligent path planning algorithm to the field of robot path planning. Compared with the path planning of static obstacles, the path planning of dynamic obstacles brings new challenges. Because there is no prior knowledge of dynamic obstacles, it is inevitable to encounter various difficulties in path planning. Therefore, the path planning of dynamic obstacles has vital practical significance.

The associate editor coordinating the review of this manuscript and approving it for publication was Moussa Aayash .

Some scholars have performed much research on route planning. Dong *et al.* [1] proposed a new dual ant colony algorithm (NDACA) based on dynamic feedback to realize accurate and efficient route planning of ships in complex marine environments. First, the energy consumption model is established by analyzing the ship's motion, which is used for the pheromone update strategy. Second, according to the energy consumption information of the route, the ant colony is divided into an exploratory ant colony and an optimization ant colony. The closed-loop feedback strategy is used to continuously adjust the number of ants in each colony to ensure the algorithm's solution quality and convergence speed. Meng *et al.* [2] presented a smooth DGWW global planning method based on the beetle antennae search (BAS) algorithm to optimize the grey wolf optimization (GWO) algorithm and solve the problems of local path difference and low convergence speed in complex path planning environments. This method integrates the method of using the two antennae of the beetle head to optimize the search in the BAS algorithm into the GWO algorithm to avoid local

optimization. Liu *et al.* [3] proposed a multistep computing framework to extract economically safe routes. First, track simplification and a density clustering algorithm are used to generate a maritime traffic network. Then, the kernel density estimation (KDE) method is introduced to generate routes related to the traffic count frequency in specific areas. Finally, the best route is extracted by the sliding window algorithm executed on the transportation road map. Dong *et al.* [4] used the A* algorithm and genetic algorithm (GA) to solve path planning. First, the path is obtained through the A* algorithm. The improved GA is used to reformulate the connection point strategy for the obtained paths to obtain secure and diversified paths. In addition, some scholars [5], [6] have combined motion characteristics with a swarm intelligence algorithm to solve the problem of path planning. Zhang *et al.* [7] designed a path planning method based on the heuristic algorithm. The distance and the changing angle as genetic material are used to construct the path to avoid the collision. In the experiment, a random pattern developer and genetic algorithm (GA) are used as optimization methods to verify the feasibility of the path planning method. These methods are of great significance to the research of path planning. However, in the case of many obstacles, they have considerable time and space complexity, and these algorithms are not intelligent in the case of new obstacles.

Under environmental situation awareness and planning algorithm selection, the path planning algorithm is a combinatorial optimization problem under complex constraints. The acquisition of obstacle information is affected by the environment modeling method. The appropriate environment modeling method is more conducive to planning the correct path. Most methods do not put forward the judgment of environmental modeling methods. Some existing path planning algorithms adopt many static strategies, and each strategy interacts with an independent environment instance for a long time. However, strategy is a function of the state to action, a dynamic process. The state and action information of independent individuals can make the search more efficient, and the static strategy does not take advantage of these contents. A Q-learning algorithm is a method of learning in interaction with the environment. It is a model-free reinforcement learning method. However, the original Q-learning algorithm affects the value evaluation because of the choice of actions, so this paper improves the value evaluation.

Q-learning aims to learn strategies that tell agents what actions to take under what circumstances. It does not need a model of the environment, and it can deal with the problems of random transformation and reward. Q-learning finds the optimal strategy for any finite Markov decision process (FMDP). In this sense, it starts from the current state and maximizes the expectation of total return in all consecutive steps. Q-learning can determine the optimal action selection strategy for any given FMDP, given infinite exploration time and partial random strategy. The main advantage of Q-learning is that the temporal difference method (TD) can

be used for offline learning, and the Bellman equation can be used to solve the optimal strategy of the Markov process.

The remaining structure of this paper is as follows: the second section describes the related work of the path planning problem. The third section introduces the improved Q-learning algorithm, and the fourth section displays the experiment. The last section gives the conclusion of this paper.

II. RELATED WORK

Path planning [8] is necessary for objects that need to move independently. In the process of moving, we need to know the surrounding environmental information and the moving method to reach the destination accurately. Therefore, in path planning, whether a ship or a mobile robot, it is necessary to simulate the accurate environmental information into a plane or a three-dimensional model that the path algorithm can process.

The path planning method is an essential branch of object motion control. For moving objects, according to their own sensors' perception of the external environment, they can independently plan a collision-free smooth route from the starting point to the target point. This description is called the path planning of a mobile robot. Path planning is a route without collision and meeting the constraints in a particular environment, such as a global static or dynamic local environment. The steps of path planning are generally divided into two parts: obstacle information description and searching the path according to the algorithm.

The robot arm's path planning to reach the target and avoid obstacles plays an essential role in manufacturing automation. Although many path planning algorithms have been proposed, including rapid exploring random trees (RRT), artificial potential field (APF), probabilistic roadmap (PRM), and reinforcement learning (RL), they have many problems: time-consuming and high computational cost. This article uses Q-learning [9] to find the robot's action to take optimization measures. Bonny *et al.* [10] presented a new method to find the best path for mobile robots in a two-dimensional environment. Use the bees algorithm (BA) and Q-learning algorithm to find the best path. Path planning based on computer vision can be crucial in many technology-driven intelligent applications. Although various path planning methods have been proposed, there are still limitations. To overcome these limitations, Abdi *et al.* [11] developed a new path planning method combining computer vision, Q-learning, and neural networks. Sahu *et al.* [12] proposed an innovative method to solve robot pairs' path planning and synchronization in known static and complex environments. This problem solves the robot's transporting the rod from the predefined starting position to the preset target position. The design of a hybrid algorithm combines the unique advantages of improved Q-learning (IQ) and democratic robot particle swarm optimization (DRPSO). AlphaGo has confirmed the practicability of reinforcement learning, and the research on the application of reinforcement learning in autonomous

driving is being actively promoted. Kim *et al.* [13] used Q-learning, one of the reinforcement learning algorithms, to realize path planning. Product assembly is a crucial stage of complex product manufacturing. How to intelligently plan the assembly process according to the dynamic product and environmental information has become an urgent problem to be solved. Guo *et al.* [14] proposed an improved Q-learning algorithm to solve the path planning problem in product assembly in virtual space.

However, the above method does not mention how to solve the problem of dynamic obstacles in path planning. The combination of evolutionary algorithm and Q-learning algorithm has advantages in solving other path planning problems. However, introducing the evolutionary algorithm will increase the complexity of the Q-table, and the original Q-learning algorithm and evolutionary algorithm are deficient in convergence speed and accuracy. This paper overcomes the algorithm's shortcomings from the path planning sequence solved by the algorithm.

A. THE ENVIRONMENT OF DYNAMIC OBSTACLE

Environmental information is the premise and foundation of path planning and a compelling description of the environment where the robot is located. The area where the robot is located is divided into a feasible area and an infeasible area. The treatment of many different forms of obstacles or impassable areas is the key. Replacing obstacles with geometric polygons has been widely used.

The visibility graph [15] method expresses the obstacles in the environment in the form of a geometric polygon, set the starting point, target point, and vertices of the robot, and connects them. It is stipulated that the connection between the vertices and starting points of the polygon, the target points, and the vertices of the deformed obstacle itself cannot pass through the obstacles. This connection is called "visible"; otherwise, it is "invisible", and the whole connected graph is called the visual graph. The disadvantage of the visibility graph is that there is no more selectivity in path planning. Agents crossing the best path must pass through obstacles, significantly increasing the collision risk.

The link graph method [16] is a modeling method based on the free generation of environmental space. First, various polygons must be preset to represent the situation of obstacles. According to the shape of polygons, the whole environment space is divided into two states: obstacle space and free space. The robot can search for the optimal path using the path planning algorithm in free space. Due to the need to preset obstacles, the link graph may not be able to consider additional obstacles in solving dynamic obstacle path planning.

The grid method [17] is one of the most widely used modeling methods in path planning research. This method decomposes the robot's working environment into multiple simple grids, simplifies the complex environmental information into grid information with a unit nature, and changes the complex motion relationship in the environmental space. Based on

the traveling area and environmental obstacles, the grids can be divided into traffic grids and blocking grids according to their different attributes. In the simulation of the grid method, the different grid sizes and the number of grids will directly affect the algorithm's performance. The smaller the grid is, the more accurate the environmental information is and the closer it is to the natural navigation environment. However, it will also lead to too much miscellaneous information and slow down the efficiency of the path planning algorithm. In contrast, the larger the grid is, the lower the resolution of environmental information, the greater the error with the proper environment, the faster the algorithm's efficiency, and the lower the natural effect.

Fig 1 shows the path planning environment of this paper. The width and height set by the test simulation are 9×9 , and the cell is set to 40 pixels. The robot's position represents the starting point, the green square represents the obstacle, and the flag represents the target point. The agent needs to find a path from the starting point to the target point with the shortest cost, which should avoid obstacles ideally. In this paper, dynamic means that in addition to the starting point, obstacles and the target point are randomly generated in the process of agent travel.

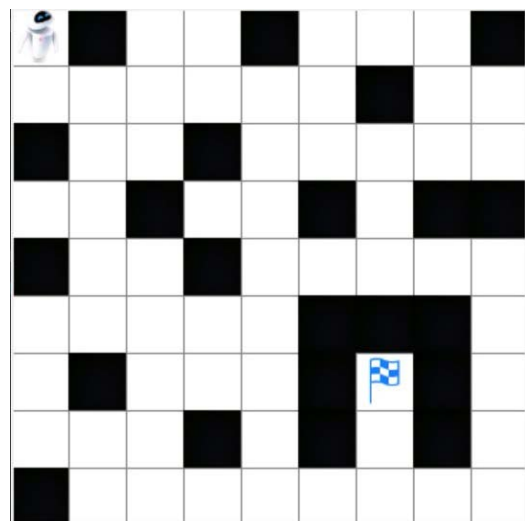


FIGURE 1. Environment modeling of path planning.

B. ALGORITHM OF PATH PLANNING

Plan planning finds a possible path from the starting point to the endpoint according to the corresponding algorithm based on the known environmental model and meets the constraints to make the predetermined performance function reach the optimal value. In the search process, according to the advantages and disadvantages of the algorithm and its adaptability, choosing and using the appropriate planning algorithm has become the research's key.

In the process of robot path planning, we should first establish the global robot navigation environment model and then select the correct path planning algorithm. Different

path planning algorithms have different effects in different environments. Choosing the appropriate algorithm is of great significance. The Dijkstra algorithm [18] proposed by Dijkstra started scholars' pursuit of path planning algorithms. Currently, with the advent of the intelligent era, a large number of scholars have conducted a variety of research investigations on path planning algorithms and have developed many common intelligent algorithms with strong computing power, such as genetic algorithm [19], particle swarm optimization [20], artificial neural networks [21], reinforcement learning [22], [23], [24], [25] and the ant colony algorithm [26]. Currently, these algorithms have been applied in different fields, and different improvement methods have been produced according to the use environment and have achieved great success. To achieve the goal of the optimal path, the reinforcement learning control agent interacts with the environment and fully considers the different conditions of the natural environment. Dijkstra and genetic algorithms adopt static strategies to interact with the environment and cannot solve the dynamic programming problem.

III. THE IMPROVED Q-LEARNING ALGORITHM

Robot technology is increasingly widely used in production and life. One of the main tasks of robot control systems is to complete path planning. When facing high-dimensional systems and systems with complex constraints, the traditional path planning algorithm has shortcomings, such as high time complexity and ease of falling into local optimization. The Q-learning algorithm [27] is a typical reinforcement learning method that does not require complete environmental knowledge. These algorithms enable the robot to learn appropriate behavior from the environment and have achieved good results in robot path planning. With the advent of the era of artificial intelligence, more complex environments put forward higher requirements for the rapidity and flexibility of path planning algorithms. Research on effectively improving the speed of path planning has essential application value and practical significance. The history of reinforcement learning can be traced back to the 1950s. Its basic idea is to transform the sequential decision-making problem into a Markov model [28]. It establishes the mapping between the environmental state and the state action value function through the interaction between the agent (robot) and the environment. It then finds the optimal state-action value function iteratively to obtain the optimal action sequence. The reinforcement learning method is widely used in robot path planning and has achieved good results. In 1995, Beom *et al.* [29] combined a fuzzy logic algorithm and reinforcement learning algorithm to realize the navigation of a ground mobile robot. Ye [30] proposed an incremental mobile robot navigation method based on reinforcement learning theory. At the European Symposium on Artificial Neural Networks in 2013, Bischoff *et al.* [31] proposed a hierarchical reinforcement learning method for robot navigation.

The purpose of Q-learning studies how agents act in the environment to maximize cumulative rewards. As shown in

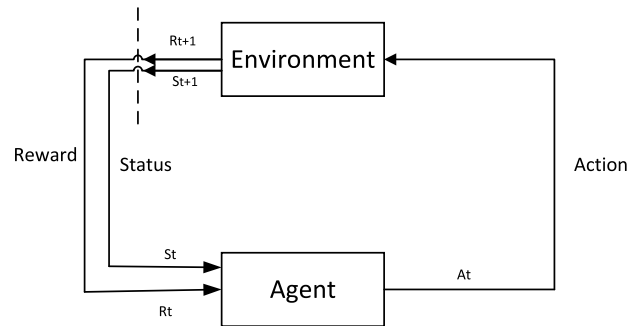


FIGURE 2. The structure of Q-learning.

Fig 2, the Q-learning framework mainly comprises the agent, environment, state, action, and reward. After the agent acts, the environment will transition to a new state and give a reward signal. Then, the agent executes new actions according to specific strategies according to the reward of the new state and environmental feedback. Through this learning method, agents can know what actions they should take in a particular state to maximize their reward. Because the interaction between agents and the environment is similar to that between humans and the environment, Q-learning is regarded as a general learning framework to solve general AI problems.

A. THE ORIGINAL Q-LEARNING ALGORITHM

The Q-learning algorithm was proposed in 1989 and is one of the methods for solving reinforcement learning tasks. The Q-learning algorithm is based on the theory of the time difference method. The time difference method is a typical model-free reinforcement learning method and the primary method of the reinforcement learning solution. It does not need complete environmental knowledge, so it is widely used. It can be used for reinforcement learning tasks with long experience tracks and for continuous reinforcement learning tasks. In path planning, the Q-learning algorithm can be used to solve the path planning problem based on a grid graph. In 2010, Goswami *et al.* [32] proposed an extended Q-learning algorithm, which accelerated the Q function's convergence speed and improved the algorithm's efficiency by introducing flag variables. Konar *et al.* [33] improved the traditional Q-learning algorithm and proposed a new deterministic algorithm with theoretical knowledge. The algorithm updates the entry Q-table at one time and significantly reduces the time complexity. The Q-learning algorithm can also be combined with the sampling-based path planning algorithm to search for an optimal path in the road map quickly.

Q-learning aims to learn the optimal action-value function Q_* . The algorithm makes the agent learn a series of trajectories according to the optimal Behrman equation. The optimal Behrman equation is shown in (1).

$$Q_*(s_t, a_t) = \mathbb{E}_{s_{t+1} \sim p(\cdot | s_t, a_t)} [R_t + \gamma \max_{A \in \mathcal{A}} Q_*(s_{t+1}, A) | s_t = s_t, A_t = a_t] \quad (1)$$

Both sides of (1) are approximated. $Q_*(s_t, a_t)$ on the left of the equation can be approximated as $\tilde{Q}(s_t, a_t)$. The estimation of $\tilde{Q}(s_t, a_t)$ is made at time t . The expectation on the right side of the equation is about state S_{t+1} at the next time. Given the current state s_t , the agent executes the action a_t , and the environment will give the reward r_t and the new state s_{t+1} . Using the observed r_t and s_{t+1} , the expectation is obtained by Monte Carlo approximation, as shown in (2).

$$r_t + \gamma \max_{a \in \mathcal{A}} Q_*(s_{t+1}, a) \quad (2)$$

Furthermore, Q_* in (2) is approximated as \tilde{Q} . Equation (3) is called the temporary difference. It is the estimation of $Q_*(s_t, a_t)$ made by the agent at time $t + 1$. $\tilde{Q}(s_t, a_t)$ and \hat{y}_t are the estimation of the optimal action value $Q_*(s_t, a_t)$. Since \hat{y}_t contains reward r_t partially based on real observations, we believe that \hat{y}_t is the more reliable estimate, so it is encouraged for $\tilde{Q}(s_t, a_t)$ to be closer to \hat{y}_t . Therefore, (4) is used to update \tilde{Q} .

$$\hat{y}_t \triangleq r_t + \gamma \max_{a \in \mathcal{A}} \tilde{Q}(s_{t+1}, a) \quad (3)$$

$$\tilde{Q}(s_t, a_t) \leftarrow (1 - \alpha)\tilde{Q}(s_t, a_t) + \alpha\hat{y}_t \quad (4)$$

The process of the Q-learning algorithm is divided into two parts: collecting training data and experience replay. Q-learning updating \tilde{Q} does not depend on a specific strategy in collecting training data. We can use any strategy to control the agent, interact with the environment, divide the obtained trajectory into (s_t, a_t, r_t, s_{t+1}) and store it in the empirical playback array. This strategy for controlling agents is called behavior policy. The most commonly used behavior strategy is $\epsilon - greedy$, which is shown as (5). The update logic of the value function $\tilde{Q}(s_t, a)$ is shown in Fig 3. In state s , the $\epsilon - greedy$ algorithm selects action a , executes action a , obtains the immediate reward r , and then transfers it to state s' . The Q-learning algorithm uses $\epsilon -greedy$ algorithm to selects action a' in-state s' , that is, selects a that maximizes $Q(s', a)$ to update the value function.

$$\begin{cases} \operatorname{argmax}_a \tilde{Q}(s_t, a) & 1 - \epsilon \\ \text{Uniform} & \epsilon \end{cases} \quad (5)$$

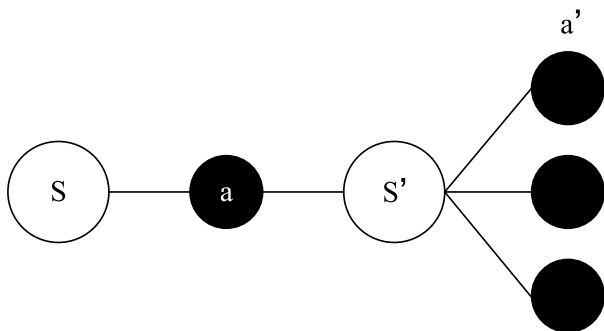


FIGURE 3. $\epsilon -greedy$ algorithm update action a .

The process of experience replay is as follows: First, a quad is randomly extracted from the experience replay array and

record it as (s_j, a_j, r_j, s_{j+1}) , \tilde{Q}_{now} , \tilde{Q}_{new} . Second, the element of \tilde{Q}_{now} at position (s_j, a_j) is recorded as (6). Third, the maximum value of line s_{j+1} of \tilde{Q}_{now} is calculated as (7). Fourth, (8) is used to calculate the TD target and TD error; Finally, (9) is adopted to update the element at (s_j, a_j) .

$$\hat{q}_j = \tilde{Q}_{now}(s_j, a_j) \quad (6)$$

$$q_{j+1} = \max_a \tilde{Q}_{now}(s_{j+1}, a) \quad (7)$$

$$\hat{y}_j = r_j + \gamma q_{j+1}, \delta_j = \hat{q}_j - \hat{y}_j \quad (8)$$

$$\tilde{Q}_{new}(s_j, a_j) \leftarrow (1 - \alpha)\tilde{Q}_{now}(s_j, a_j) + \alpha\delta_j \quad (9)$$

B. Q-LEARNING BASED ON PRIORITIZED WEIGHT

One advantage of prioritized weight is to break the correlation of sequences. Another advantage is to reuse the collected experience instead of discarding it once. It can achieve the same performance with fewer samples. It can not only make the convergence faster but also make the average return higher. In general experience, one sample is obtained by uniform sampling each time. The prioritized weight gives each sample weight and then makes nonuniform random sampling according to the weight.

Due to the different importance of samples, many samples normally travel in path planning, but few encounter obstacles. These few samples will have a great impact on path planning. Therefore, the samples that encounter obstacles should have a higher weight and receive more attention. Normal samples should not be treated equally. In practice, we cannot know which samples are more important. Therefore, this can be judged according to the absolute value of the TD error. If the absolute value of the TD error δ_j is large, it indicates that the current evaluation of the real value of (s_j, a_j) is inaccurate; then, a higher weight should be set for (s_j, a_j, r_j, s_{j+1}) .

In this paper, the sampling method arranges the error values δ_j in descending order and then calculates the sampling probability, as shown in (10). If we perform nonuniform sampling, we should adjust the learning rate according to the sampling probability α . If a sample has a high probability of being sampled, its learning rate should be relatively low. The learning rate α can be set according to (11).

$$p_j \propto \frac{1}{\operatorname{rank}(j)} \quad (10)$$

$$\alpha_j = \frac{\alpha}{(b * p_j)^\beta} \quad (11)$$

where b is the total number of samples in the experience replay and $\beta \in (0, 1)$ is a super parameter that needs to be adjusted.

The Q-learning algorithm has the problem of overestimating the following action. This paper adopts prioritized weight to select the following action to solve this problem. At any position, the agent will choose the action with the most significant reward with probability $1 - \phi$ and randomly choose an action with probability ϕ , called the random occurrence probability. After each selection, a probability delivery is obtained, which describes the probability of each action

selected. This probability delivery is used as input for the following selection, and the process is iterated repeatedly.

Algorithm 1 Q-Learning Based on Prioritized Weight

```

Initialize Q table,  $b, \beta, \gamma$ 
for e=1 to episodes
    Initialization status  $S$ 
    repeat
        Using (4), select action  $a$  under state  $s$ ;
        Using (9), obtain reward  $r$  and the next status  $s'$ ;

        Using (10), random sampling;
        prioritized weight update Q;
         $s \leftarrow s'$ ;
        until  $s$  is terminated;
    end for
end for
    
```

IV. EXPERIMENT

A. TEST ENVIRONMENT AND PARAMETERS SETTINGS

The setting of the test running environment of the paper is shown in Table 1. The experiments in this paper are based on this equipment.

TABLE 1. Experimental environment.

| Components | Attribute |
|---------------------|--|
| Operating system | CentOS Linux release 7.6.1810 (Core) |
| Memory | 754G |
| CPU | Intel(R) Xeon(R) Platinum 8260 CPU @ 2.40GHZ |
| Basic frequency | 2.40GHZ |
| Programing language | Python 3.8.3 |
| Graphics card | NVIDIA Corporation TU102GL (rev a1) |

In the experiment section, we compare Q-learning with the A* algorithm [34], PRM [35], RRT [36], and BRRT(bidirectional rapid exploring random trees) [37]. The parameters of the comparison algorithm are shown in Table 2.

TABLE 2. Experimental parameters.

| Algorithms | parameters |
|---------------------|--|
| Improved Q-learning | $b=5000, \beta = 0.3, \gamma \in [0.1, 0.9], \epsilon=0.9$ |
| Q-learning | $\gamma \in [0.1, 0.9], \epsilon=0.9$ |
| A* | $thelengthofcomeFrom=1000$ |
| PRM | $k = 50$ |
| RRT | $step=20, disTh=20, maxAttempts=10000$ |
| BRRT | $step=20, disTh=20, maxAttempts=10000$ |

$\gamma = 0$ means that the agent only cares about the recent rewards, while $\gamma = 1$ makes it committed to higher rewards in the long term. If the discount factor exceeds 1, the action value Q may diverge. In this case, the learning efficiency can be improved by gradually increasing the discount factor from a lower value to the final value. As the number of iterations increases, the value of γ increases from 0.1 to 0.9. The total number of samples in the experience replay $b=5000$ and $\beta = 0.3$. In the experimental simulation, the actions taken by the agent are only up, down, left, and right.

B. THE COMPARATIVE EXPERIMENT OF ALGORITHMS

In the same experimental environment, we verify the performance of various comparison algorithms from the starting point (0, 0) to the endpoint (240, 240). In the path planning from the beginning to the end, the comparison algorithm needs to find an optimal path with the shortest length and is time-consuming under the price adjustment of obstacles. The experimental results of the comparison algorithm are shown in Fig 4-Fig 9.

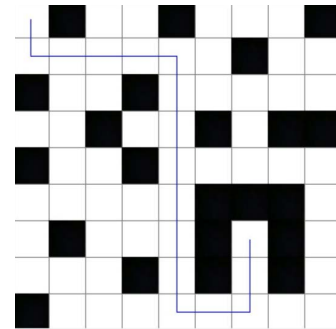


FIGURE 4. The improved Q-learning algorithm for path planning.

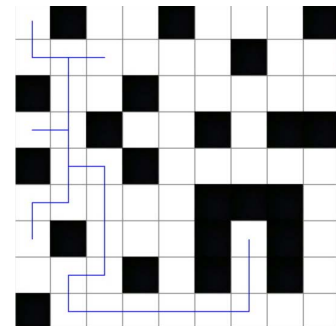


FIGURE 5. The Q-learning algorithm for path planning.

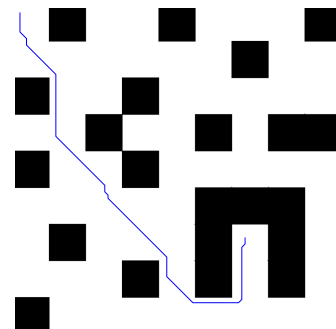


FIGURE 6. The A* algorithm for path planning.

Fig 4 and Fig 5 use the grid method to divide the path graph, which intuitively shows the path planning. Due to the defects of the algorithm, although Fig 5 finds the absolute path, additional costs are incurred in the planning process. Fig 6-9 do not use the grid method to segment the path map. They

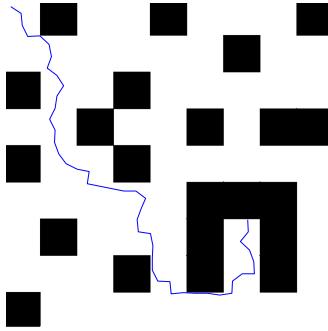


FIGURE 7. The BRRT algorithm for path planning.

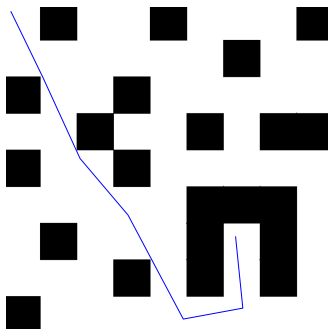


FIGURE 8. The PRM algorithm for path planning.

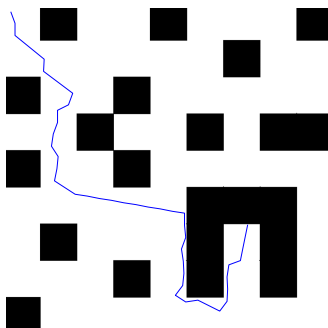


FIGURE 9. The RRT algorithm for path planning.

TABLE 3. The shortest path distance and time.

| Algorithms | Distance(cell) | Time(s) |
|---------------------|----------------|---------|
| Improved Q-learning | 16 | 27 |
| Q-learning | 18 | 36 |
| A* [34] | 78 | 130 |
| PRM [35] | 90 | 31 |
| RRT [36] | 102 | 9.3 |
| BRRT [37] | 92 | 4.7 |

plan by constantly exploring the surrounding environment in path planning so that the formed path will be irregular. Some path planning will be close to obstacles, indicating no safety distance constraint. The path cost in Fig 6-9 is also relatively high. The comparison algorithm’s shortest path distance and time cost are shown in Tables 3.

As seen from Table 3, the path found by the improved Q-learning algorithm is the shortest among all the comparison algorithms, followed by the original Q-learning algorithm. The A* algorithm takes a long time because it needs to spread continuously to find the endpoint in the iterative process of the algorithm. BRRT algorithm takes the shortest time because the algorithm starts from the starting point and the endpoint to find the path until a certain point coincides. PRM algorithm will randomly generate points according to the set value in the iteration process. The algorithm can find the path if the generated points can be connected into a line from the starting point to the endpoint. Therefore, if the environment is complex, the PRM algorithm may not be able to find the path unless more points are generated. Although the time consumption in the path planning process of the improved Q-learning algorithm is not the least, the path is the shortest. Therefore, from this perspective, the improved Q-learning algorithm performs better in path planning.

If the agent drives to the end, the reward value is 1. If the agent drives to an obstacle, the reward value is - 1. In other cases, the reward value is 0. The whole episode set in the paper is 1000. With the increase in episodes, the steps and cost are shown in Fig 10 to Fig 13. With the increase in episodes, the steps of Fig 10 based on the improved Q-learning change significantly during the continuous exploration in the early stage. However, after the 400th episode, the steps are stable, and most of them are disturbed at steps 0 to 25. However, Fig 11 stabilizes only in the 920th episode. In the early stage of the test simulation, the reward value of the agent is negative. Similarly, after 400 episodes, the robot using the improved algorithm, finds the path of the target position. It is shown in Fig 12. However, the cumulative return of Fig 13 is lower than that of improved Q-learning due to the unreasonable use of different samples.

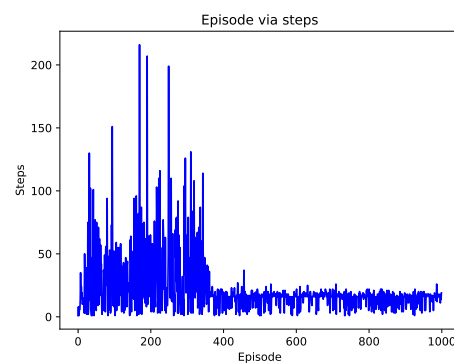


FIGURE 10. Episode via steps of improved Q-learning.

To further verify the performance of the improved algorithm, we dynamically generate obstacles and let the robot plan a reasonable route. The results are shown in Fig 14 and Fig 15. The results show that the improved Q-learning algorithm plans a good path. From the steps and costs of each episode, since the obstacles in Fig 14 and Fig 15 are randomly generated, the exploration in the process of traveling is

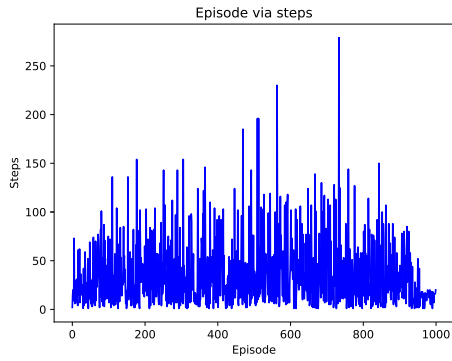


FIGURE 11. Episode via steps of original Q-learning.

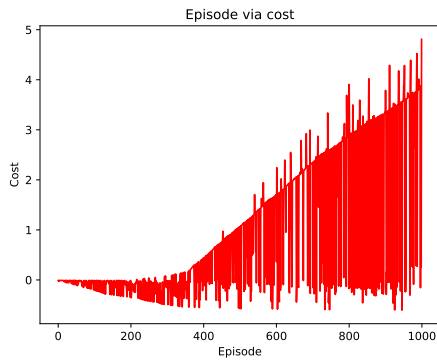


FIGURE 12. Episode via cost of improved Q-learning.

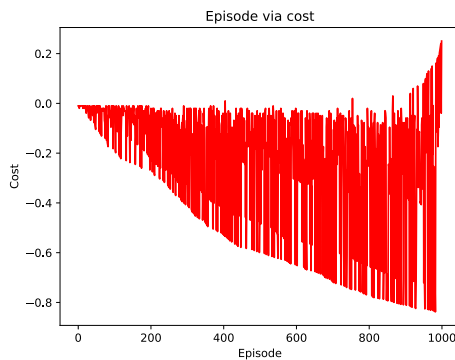


FIGURE 13. Episode via cost of original Q-learning.

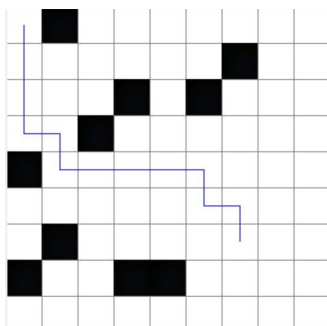


FIGURE 14. Robot path 1 planning simulation test.

different, and the steps and costs of each episode are different, as shown in Fig 16 to Fig 19. Fig 16 and Fig 18 tend to be

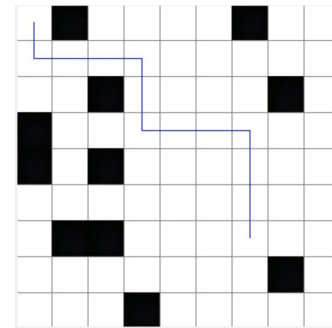


FIGURE 15. Robot path 2 planning simulation test.

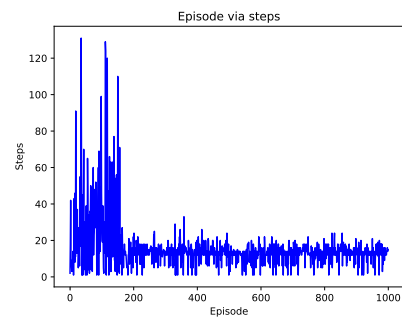


FIGURE 16. Episode via step of path 1.

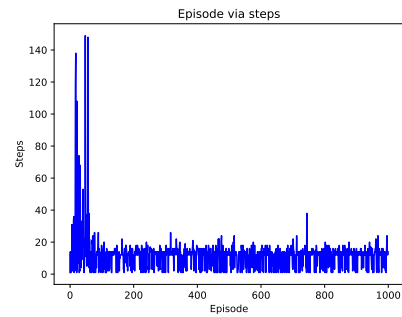


FIGURE 17. Episode via step of path 2.

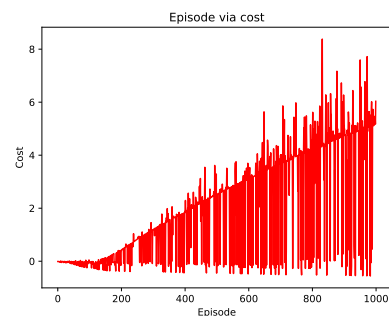


FIGURE 18. Episode via cost of path 1.

stable when exploring no more than 185 episodes. Fig 17 and Fig 19 are stable around the 100th episode due to the complex environment. The maximum path length explored in Fig 16 is 128, which takes more time in the early stage. After planning the shortest path, the improved Q-learning algorithm sets the

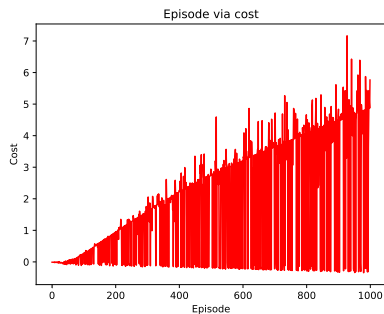


FIGURE 19. Episode via cost of path 2.

highest weight in the q-table. The path is planned according to the shortest distance in the subsequent iteration process. In the reward value shown in Fig 18, the maximum reward value obtained by the improved Q-learning algorithm after reaching the stable state is 9, which shows that the improved algorithm improves the reward value of the algorithm. Therefore, it can be concluded that when the environment is not complex, the Q-learning algorithm reaches a steady state in a short time. If the environment becomes complex, the algorithm needs more exploration of the unknown environment.

V. CONCLUSION

This paper applies the improved Q-learning algorithm to dynamic obstacle avoidance and path planning and compares it with the A* algorithm, PRM, RRT, and BRRT. The results show that the priority weight improves the value evaluation of Q-learning and the algorithm's performance. The improved Q-learning algorithm has dramatically improved the convergence speed and accuracy and can find a better path in the path planning of dynamic obstacles.

As the scale of the problem increases, the Q-table in the Q-learning algorithm will also expand, increasing the algorithm's complexity. Q-learning has the problem of overestimation, which makes it impossible to choose the optimal action. The Q-learning algorithm may produce a locally optimal solution rather than a globally optimal one, resulting in the agent not obtaining a higher reward. The solutions to these problems are our future research work.

REFERENCES

- [1] L. Dong, J. Li, W. Xia, and Q. Yuan, "Double ant colony algorithm based on dynamic feedback for energy-saving route planning for ships," *Soft Comput.*, vol. 25, no. 7, pp. 5021–5035, Apr. 2021.
- [2] H. Meng, P. Zhi, W. Zhu, H. Qiu, H. Wang, and Y. Wu, "Research on unmanned ship route planning based on the smoothed DGWW algorithm," in *Proc. 4th IEEE Int. Conf. Ind. Cyber-Phys. Syst. (ICPS)*, May 2021, pp. 816–819.
- [3] R. W. Liu, M. Liang, J. Nie, S. Garg, Y. Zhang, and Z. Xiong, "Extraction of hottest shipping routes: From positioning data to intelligent surveillance," in *Proc. IEEE 22nd Int. Conf. Inf. Reuse Integr. Data Sci. (IRI)*, Aug. 2021, pp. 255–262.
- [4] Z. Dong and X. Bian, "Ship pipe route design using improved A* algorithm and genetic algorithm," *IEEE Access*, vol. 8, pp. 153273–153296, 2020.
- [5] L. Wang, Z. Zhang, Q. Zhu, and S. Ma, "Ship route planning based on double-cycling genetic algorithm considering ship maneuverability constraint," *IEEE Access*, vol. 8, pp. 190746–190759, 2020.
- [6] W. Zhang, C. Yan, H. Lyu, P. Wang, Z. Xue, Z. Li, and B. Xiao, "Colregs-based path planning for ships at sea using velocity obstacles," *IEEE Access*, vol. 9, pp. 32613–32626, 2021.
- [7] X. Zhang, Y. Zuo, Y. Li, T. Li, and C. L. P. Chen, "Path planning of ship collision avoidance based on stochastic schemata exploiter," in *Proc. Int. Conf. Secur., Pattern Anal., Cybern. (SPAC)*, Jun. 2021, pp. 47–53.
- [8] S. Xu, Y. Gu, X. Li, C. Chen, Y. Hu, Y. Sang, and W. Jiang, "Indoor emergency path planning based on the Q-learning optimization algorithm," *ISPRS Int. J. Geo-Inf.*, vol. 11, no. 1, p. 66, Jan. 2022.
- [9] A. Abdi, D. Adhikari, and J. H. Park, "A novel hybrid path planning method based on Q-learning and neural network for robot arm," *Appl. Sci.*, vol. 11, no. 15, p. 6770, Jul. 2021.
- [10] T. Bonny and M. Kashkash, "Highly optimized Q-learning-based bees approach for mobile robot path planning in static and dynamic environments," *J. Field Robot.*, vol. 39, no. 4, pp. 317–334, Jun. 2022.
- [11] A. Abdi, M. H. Ranjbar, and J. H. Park, "Computer vision-based path planning for robot arms in three-dimensional workspaces using Q-learning and neural networks," *Sensors*, vol. 22, no. 5, p. 1697, Feb. 2022.
- [12] B. Sahu, P. K. Das, and M. R. Kabat, "Multi-robot cooperation and path planning for stick transporting using improved Q-learning and democratic robotics PSO," *J. Comput. Sci.*, vol. 60, Apr. 2022, Art. no. 101637.
- [13] H. Kim and W. Lee, "Real-time path planning through Q-learning's exploration strategy adjustment," in *Proc. Int. Conf. Electron., Inf., Commun. (ICEIC)*, Jan. 2021, pp. 1–3.
- [14] X. Guo, G. Peng, and Y. Meng, "A modified Q-learning algorithm for robot path planning in a digital twin assembly system," *Int. J. Adv. Manuf. Technol.*, vol. 119, nos. 5–6, pp. 3951–3961, Mar. 2022.
- [15] S. Oh and H. W. Leong, "Edge N-level sparse visibility graphs: Fast optimal any-angle pathfinding using hierarchical taut paths," in *Proc. 10th Annu. Symp. Combinat. Search*, 2017, pp. 1–9.
- [16] E. Maliaroudakis, K. Boland, S. Dietze, K. Todorov, Y. Tzitzikas, and P. Fafalios, "ClaimLinker: Linking text to a knowledge graph of fact-checked claims," in *Proc. Companion Proc. Web Conf.*, Apr. 2021, pp. 669–672.
- [17] B. Yang, J. Yan, Z. Cai, Z. Ding, D. Li, Y. Cao, and L. Guo, "A novel heuristic emergency path planning method based on vector grid map," *ISPRS Int. J. Geo-Inf.*, vol. 10, no. 6, p. 370, May 2021.
- [18] M. Parimala, S. Jafari, M. Riaz, and M. Aslam, "Applying the Dijkstra algorithm to solve a linear diophantine fuzzy environment," *Symmetry*, vol. 13, no. 9, p. 1616, Sep. 2021.
- [19] Y. V. Pehlivanoglu and P. Pehlivanoglu, "An enhanced genetic algorithm for path planning of autonomous UAV in target coverage problems," *Appl. Soft Comput.*, vol. 112, Nov. 2021, Art. no. 107796.
- [20] M. D. Phung and Q. P. Ha, "Safety-enhanced UAV path planning with spherical vector-based particle swarm optimization," *Appl. Soft Comput.*, vol. 107, Aug. 2021, Art. no. 107376.
- [21] H.-S. Jo, C. Park, E. Lee, H. K. Choi, and J. Park, "Path loss prediction based on machine learning techniques: Principal component analysis, artificial neural network, and Gaussian process," *Sensors*, vol. 20, no. 7, p. 1927, Mar. 2020.
- [22] F. Gismondi, C. Possieri, and A. Tornambe, "A solution to the path planning problem via algebraic geometry and reinforcement learning," *J. Franklin Inst.*, vol. 359, no. 2, pp. 1732–1754, Jan. 2022.
- [23] J. Zheng, S. Mao, Z. Wu, P. Kong, and H. Qiang, "Improved path planning for indoor patrol robot based on deep reinforcement learning," *Symmetry*, vol. 14, no. 1, p. 132, Jan. 2022.
- [24] Z. Cui and Y. Wang, "UAV path planning based on multi-layer reinforcement learning technique," *IEEE Access*, vol. 9, pp. 59486–59497, 2021.
- [25] X. Liu, D. Zhang, T. Zhang, Y. Cui, L. Chen, and S. Liu, "Novel best path selection approach based on hybrid improved A* algorithm and reinforcement learning," *Int. J. Speech Technol.*, vol. 51, no. 12, pp. 9015–9029, Dec. 2021.
- [26] Y. Gan, Y. He, L. Gao, and W. He, "Propagation path optimization of product attribute design changes based on Petri net fusion ant colony algorithm," *Expert Syst. Appl.*, vol. 173, Jul. 2021, Art. no. 114664.
- [27] C. Qu, W. Gai, M. Zhong, and J. Zhang, "A novel reinforcement learning based grey wolf optimizer algorithm for unmanned aerial vehicles (UAVs) path planning," *Appl. Soft Comput.*, vol. 89, Apr. 2020, Art. no. 106099.
- [28] P. Kumar and C. Dudeja, "Shadowed type 2 fuzzy-based Markov model to predict shortest path with optimized waiting time," *Soft Comput.*, vol. 25, no. 2, pp. 995–1005, Jan. 2021.
- [29] H. R. Beom and K. S. Cho, "A sensor-based navigation for a mobile robot using fuzzy logic and reinforcement learning," *IEEE Trans. Syst., Man, Cybern.*, vol. 25, no. 3, pp. 464–477, Mar. 1995.

- [30] B. Bischoff, D. Nguyen-Tuong, I. H. Lee, F. Streichert, and A. Knoll, "Hierarchical reinforcement learning for robot navigation," in *Proc. Eur. Symp. Artif. Neural Netw., Comput. Intell. Mach. Learn. (ESANN)*, 2013, pp. 1–6.
- [31] P. K. Das, A. Konar, and R. Janarthanan, "Extended Q-learning algorithm for path-planning of a mobile robot," in *Proc. Asia-Pacific Conf. Simulated Evol. Learn.* Springer, 2010, pp. 379–383.
- [32] C. Ye and J. Borenstein, "A method for mobile robot navigation on rough terrain," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, vol. 4, Apr. 2004, pp. 3863–3869.
- [33] A. Konar, I. G. Chakraborty, S. J. Singh, L. C. Jain, and A. K. Nagar, "A deterministic improved Q-learning for path planning of a mobile robot," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 43, no. 5, pp. 1141–1153, Sep. 2013.
- [34] X. Xiong, H. Min, Y. Yu, and P. Wang, "Application improvement of A* algorithm in intelligent vehicle trajectory planning," *Math. Biosci. Eng.*, vol. 18, no. 1, pp. 1–21, 2021.
- [35] G. Chen, N. Luo, D. Liu, Z. Zhao, and C. Liang, "Path planning for manipulators based on an improved probabilistic roadmap method," *Robot. Comput.-Integr. Manuf.*, vol. 72, Dec. 2021, Art. no. 102196.
- [36] Z. Zhang, B. Qiao, W. Zhao, and X. Chen, "A predictive path planning algorithm for mobile robot in dynamic environments based on rapidly exploring random tree," *Arabian J. Sci. Eng.*, vol. 46, no. 9, pp. 8223–8232, Sep. 2021.
- [37] J. Wang, B. Li, and M. Q.-H. Meng, "Kinematic constrained bi-directional RRT with efficient branch pruning for robot path planning," *Expert Syst. Appl.*, vol. 170, May 2021, Art. no. 114541.



XIAO YANG received the B.Eng. degree from Guangxi University For Nationalities, China, in 2018. He is currently pursuing the Ph.D. degree with Harbin Engineering University, China. His research interests include data mining, swarm intelligence algorithm, and machine learning.



CHUNLEI WANG is currently pursuing the Ph.D. degree with the China University of Mining and Technology. He is a Professor of foreign studies with Suqian University. His research interests include wisdom education, intelligent algorithm, and artificial intelligence.



HE LI is a Professor with the Department of Languages and Communication Studies, Beijing Jiaotong University. Her research interests include intelligent algorithm, machine learning, and wisdom education.

• • •