Map Optimization of Path Planning in Q-Learning

Ye Li

Department of Ulster College, Shaanxi University of Science & Technology, Xi'an, China

202015020312@sust.edu.cn

Abstract. Path planning is significant in the field of artificial intelligence and robotics. This paper proposes a unique map optimization of path planning relying on Q-learning to overcome the shortcomings of classic Q-learning, such as delayed convergence or low efficiency. First, improvements were made to the setup environment, turning a simple environment into a more complex one. Secondly, rewards were set to ensure that each step is optimal exploration. The optimal path is the globally optimal path by setting up, down, left, and right directions simultaneously. Finally, MATLAB simulation was used for verification. As compared to the original training environment, the improved map enhances learning efficiency in a more complicated environment, increases the algorithm's convergence rate, and enables the robot to swiftly discover the collection-free path and finish the job in a complex environment. The rationality of the improvement is verified, which provides important data and a theoretical basis for the subsequent research on Q-learning.

Keywords: Reinforcement learning, Q-learning, Path planning, Artificial intelligence.

1. Introduction

Path planning has been applied in robotics, autonomous vehicles, obstacle avoidance on ships, and construction machines as science and technology have advanced and gained popularity. Nowadays, path planning is now a critical difficulty for mobile robot autonomous navigation [1]. Therefore, path planning has received a lot of attention, and more and more researchers are studying enhancements to the path planning algorithm. Path planning has long been recognized as a critical technology in the realm of mobile robots. It means that the robot determines the best path between the beginning and finishing points in a given environment. Typical traditional algorithms for path planning include tree optimization algorithm method, artificial potential field method, neural network algorithm, A* algorithm, etc [2-5]. Furthermore, there are several bionic algorithms available, including the ant colony algorithm, genetic algorithm, particle swarm optimization algorithm, etc. These algorithms have also achieved good results in the area of route planning, but the above algorithms need prior environmental information before planning, so they do not apply to unknown complex environments. In some industries, when the path planning environment info came more sophisticated, the storage environment information has significant memory needs, and path planning still relies heavily on manual sites. Subjective considerations will influence this planning process, resulting in the use of labor, material resources, and work-hours [6]. As a result, finding a better path in an unknown environment has become a research priority. Reinforcement learning, as a key machine learning approach, has received increasing interest and use in robot route planning. Reinforcement learning provides intrinsic benefits in path planning as compared to other learning approaches [7].

Reinforcement learning is a fundamental strategy in machine learning that does not need prior knowledge of the environment or the supply of any data. In contrast to both supervised and unsupervised learning, reinforcement learning requires the agent to interact with the surroundings on a continuous basis, with each engagement receiving an evaluation feedback signal in exchange for studying the optimal action by raising the total reward [8]. This strategy is highly adaptable and efficient, and it can prevent slipping into the local ideal solution. As a result, reinforcement learning is becoming increasingly popular in path planning. Robots acquire knowledge through exploration in the environment and learn in the process of trial and error. The reinforcement learning method has obvious advantages in path planning with less prior environmental information.

Q-learning is a learning algorithm that combines reinforcement learning with artificial intelligence. Reinforcement learning does not require external supervision; can interacts with the outside world

via its sensors. This continuously learns to map the condition of the external input environment to output action and maximizes the related value of this action [9]. Q-learning is used in path planning to discover the most efficient path between two places. The program works by allocating a reward to each action made by the agent and then determining the optimum sequence of actions that will result in the maximum reward using a Q-learning algorithm. The algorithm is able to learn from its mistakes and improve its performance over time. It can adjust the incentives connected with each activity according to feedback from the environment. This enables the algorithm to identify the shortest path between two places. Q-learning is a versatile path planning tool that may be utilized in a number of situations. It can, for example, be used to identify the most efficient route for a robot to follow to reach its goal or the best path for a car to travel through a metropolis. It may also be used to find the most effective path for a vehicle to go through a maze. Path planning technology may assist the robot in travelling from the starting position to the target position without encountering obstacles, and a mobile agent must determine an ideal or suboptimal collision-free path from the start point to the end position with in environment [10, 11]. Route planning in a barrier environment has historically been an issue for mobile robots. The Q-learning algorithm's utility in interacting with the environment increases its relevance [12]. Overall, path planning based on Q-learning is a powerful tool for solving complex problems. It is able to learn from its mistakes and adjust its rewards accordingly, thus allowing it to find the most efficient path between two points. It is a powerful tool for a variety of applications, such as route planning for robots and vehicles, and maze solving. Therefore, Q-learning, as a kind of reinforcement learning, began to develop gradually. Despite the fact that various Q-learning algorithms have been developed to handle path optimization issues for mobile robots, optimum pathways remain difficult to achieve; also, the convergence speed of such algorithms remains a hurdle, particularly when dealing with complicated surroundings [13].

In this paper, based on the existing Q-learning environment, an optimized map of the Q-learning algorithm is presented to tackle the problem. Improving the more complex environment and training in the environment several times to maximize learning efficiency. Simultaneously, the data in the Q table is shown and examined. It is envisaged that this study would give solid data to demonstrate the use of Q-learning, with the ultimate objective of applying it to future vehicle path planning.

2. Problem Description

The path planning problem addressed in this research is as follows: the robot starts from the starting point, finds a safe path through path planning, acts according to the planned path and avoids obstacles, and finally reaches the target point safely.

3. Q-Learning Improvements

3.1. Principles of Q-Learning Algorithm

WATKINS (Watkins, 1989) *et al.* proposed the Q-learning algorithm, which is a model-free value function reinforcement learning algorithm. Due to the benefits of being model-free and being able to collect and update knowledge through ongoing contact with the environment, Q-learning has a broad application possibility for solving complicated optimization issues in path planning [14]. The Markov decision process may be used to codify the procedure (MDP). MDP is represented as a quadruple $(S, A, P(s,s',a), R(s,s',a))$, where S is the state space, A is the action space, and $P(s,s',a)$ is the state transition probability function (model). This model expresses the likelihood that the environment state $s \in S$ will change to the new state $s' \in S$ after the agent performs the action $a \in A$.

Once the agent picks and performs action $a \in A$, reward $R(s,s',a)$ is artificially set to convert the environment state $s \in S$ to the new state $s' \in S$. When the reinforcement learning agent follows the strategy π from the state at the time t_1 to the end of the state at time t_2, the sum of state cumulative rewards and rewards is defined as:

$$R_t = \sum_{t'=t1}^{t2} \gamma^{t'-t1} r_{t'} \tag{1}$$

Formula (1), r_i is the reward for the environmental feedback after every choice and action made by the agent; $\gamma \in (0,1)$ is the factor of discount. Based on the agent's state space and action space, the Q-learning algorithm will create a Q table that will be used to record the Q values of all state-activity pairings. Simultaneously, a reward function is meant to evaluate the advantages and downsides of the agent's behavior in the environment. If the state-action pair receives a positive reward from the environment, the corresponding Q value will gradually increase. If the environment provides a negative reward to the state-action pair, the associated Q value will decrease. Q (st, a) is progressively revised with each training round and approaches Q* (st, a). The updated formula of the Q table based on the Behrman formula is:

$$Q(s_t, a) = Q(s_t, a) + \alpha[r_t + \gamma \max Q(s_{t+1}, a) - Q(s_t, a)] \tag{2}$$

In the Formula (2), $\alpha \in (0, 1)$ represents the learning rate; r_t represents the reward that the agent gets every time it chooses an action; $\gamma \in (0,1)$ represents the discount factor. Ultimately, the agent chooses the best action in each state based on the Q table to perform the objective job.

The traditional Q-learning algorithm is commonly used for ε-greedy strategy. The strategy's objective is to discover an exploration factor ε. When choosing an action, the agent chooses the optimal action with a plausibility of ε and an irregular action with a probability of 1-ε. The algorithm expression is:

$$\pi(s_t) = \begin{cases} \arg\max Q(s_t, a) \\ a \in A \end{cases} \tag{3}$$

In Equation (3), $\varepsilon \in [0,1]$. When ε tends to zero, the agent merely investigates its surroundings. When ε approaches one, the agent just uses the surroundings. σ is a chance variable, $\sigma \in [0,1]$. If σ is smaller or equal to ε, the agent does the action that achieves the greatest Q. If σ exceeds ε, the agent selects an action at random.

3.2. Parameter Setting

3.2.1 Reward Value Setting

Since the ultimate purpose of path planning is to choose the path with the highest long-term cumulative reward value, the reward function is critical. The Q-learning algorithm may evaluate the efficacy of an activity in a particular state based on the specified reward function. The reward function's feedback is classified as either positive or negative. The feedback acquired to choose the next action is used to compute the predicted reward of the next state. Therefore, the agent will get a positive reward $(r = +1)$ if the goal is reached; a negative reward $(r = -1)$ if an obstacle or world boundary is hit; and no reward $(r = 0)$ if goal is not reached.

3.2.2 Action and State Setting

For the action at each cell, 4 actions can be taken, left, right, up, or down, and for the state each cell in the grid world is a state.

4. Simulation Analysis

4.1. Complex Environment Model Establishment

In this paper's path planning problem, the grid technique is employed to generate the environment model. The grid map is composed of 10*10 grids. The map contains several static obstacles in

different locations. Fig. 1 depicts the initial path-planning environment, and Fig. 2 depicts the enhanced path-planning environment in this work.

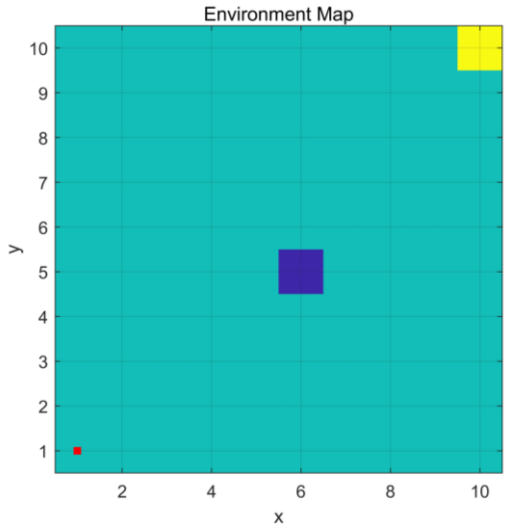


Figure 1. Original MATLAB environment map

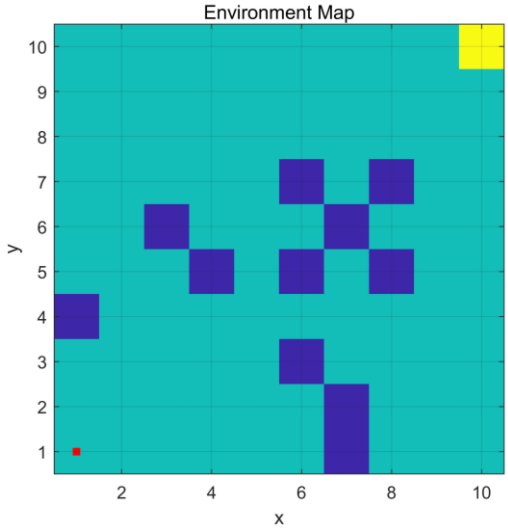


Figure 2. Original MATLAB environment map

In Fig. 1 and Fig. 2, the red dot shows the vehicle's beginning place, while the regions in light green, dark blue, and yellow indicate the moveable area, the location of static obstacles on the map, and the robot's goal point, respectively. The information for the beginning and target points is known, but the information for the barrier is unknown.

4.2. Path Planning Simulation in MATLAB

In this study, the Robotics Playground Toolbox in MATLAB is utilized to undertake designed to manipulate utilizing the toolbox's Q-learning. This package provides a thorough introduction to Mobile Robotics using MATLAB and Simulink. Users may learn about State flow for autonomous robots by participating in a series of task activities using digital robots. The toolkit includes exercises that walk users through the integration of traditional robot sensors and the creation of remotely operated and driverless mobile robot strategies. The contents include a digital environment library in both Simulink and MATLAB meant to help with the introduction of essential ideas in robotics such as vehicle control, autonomous software, object tracking, and device interfaces [15].

Robotics Playground can be used in MATLAB toolbox to simulate the whole process. When the vehicle starts from the initial position, which is the red point in the map, an optimal collision-free and pickup path can be formed by designing the state, action, and reward function.

In this path, the vehicle will avoid all the static obstacles, and then get to the yellow part which is the target point. When the vehicle sensors detect that obstacles have changed, the obstacles information detected will be converted into a digital signal by sensors and transmitted to the vehicle controller. Therefore, the vehicle will make the next decision based on the current state and the values of the Q-table.

Finally, the optimal path is obtained by the reward function. Over and over again, through constant training, the vehicle will smoothly proceed along the path to the goal place.

Fig. 3, Fig. 4, Fig. 5, and Fig. 6 depict the previous map's training progress. When the vehicle is trained on the original map, the convergence is very fast because the map is very simple, but the convergence speed is not significantly improved in the process of four repeated experiments, and the overall learning efficiency is not high. However, the optimized map has some different results in Fig. 7, Fig. 8, Fig. 9, and Fig. 10. First, due to the complexity of the map, it is obvious that the number of episodes is significantly increased, indicating that its environmental complexity is realized in this environment. Secondly, with four training sessions, the number of its episode decreases, indicating that its independent convergence rate is increasing during each training session, enabling the vehicle to reach the destination faster and more accurately. Therefore, by comparing the two kind of training progress, it can be found that the improved map is significantly improved from the original map in terms of convergence speed and learning efficiency. Finally, in Fig. 11, Fig. 12, Fig. 13, and Fig. 14, by conducting the function, the optimal path can be displayed, which shows that the vehicle can accomplish the task.

The data of the Q table is shown in Table 1, Table 2, Table 3 and Table 4.

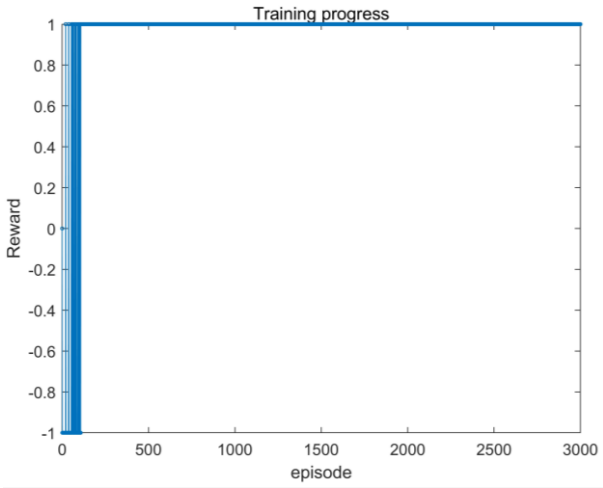


Figure 3. Training progress of the original environment

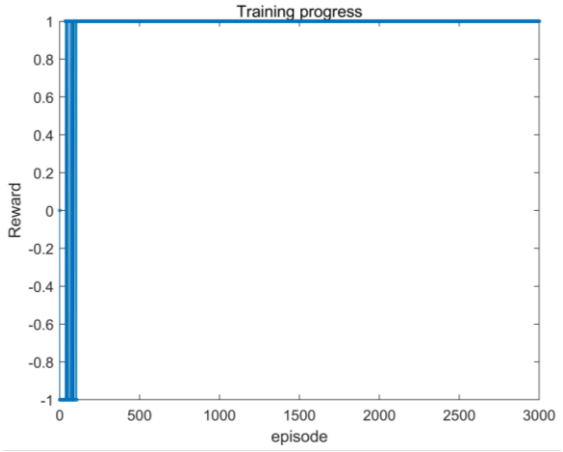


Figure 4. Training progress of the original environment

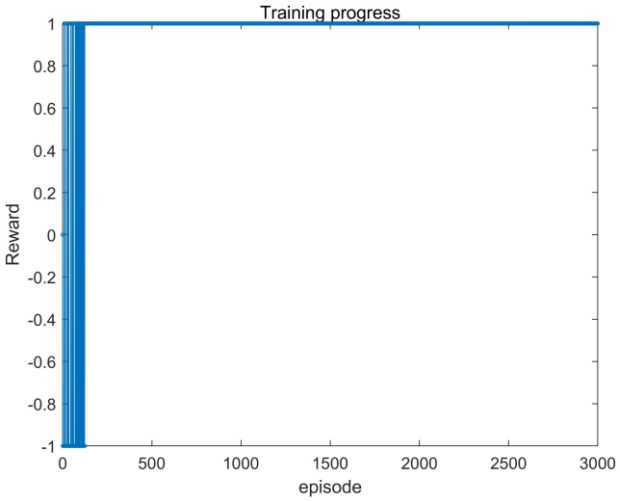


Figure 5. Training progress of the original environment

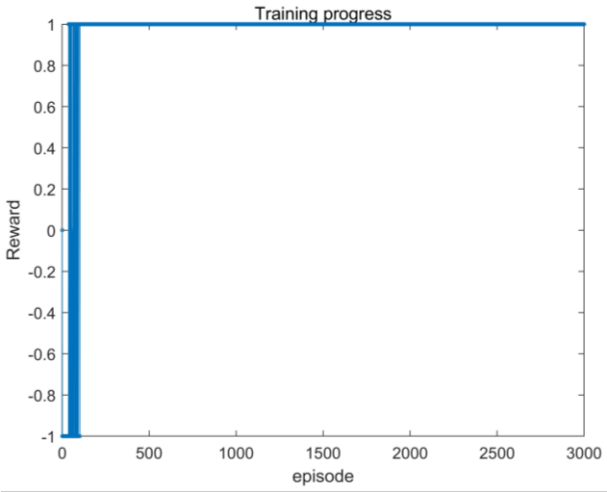


Figure 6. Training progress of the original environment

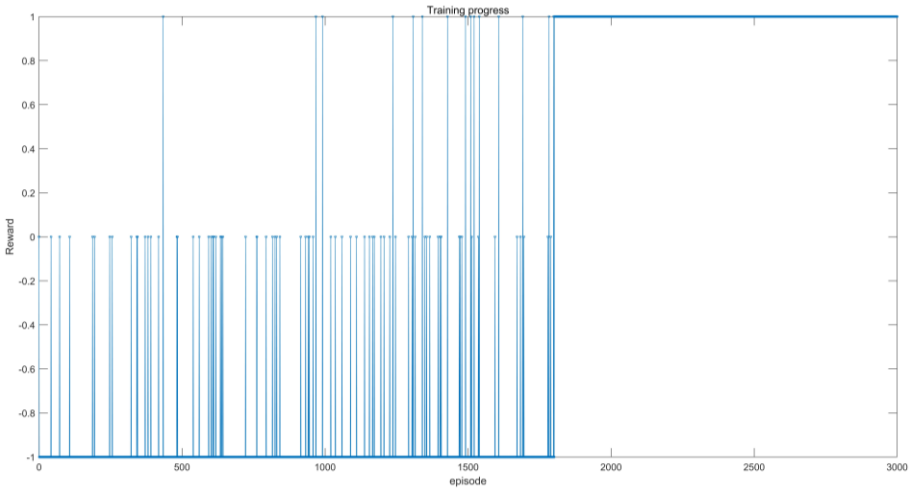


Figure 7. Training progress of the optimized environment

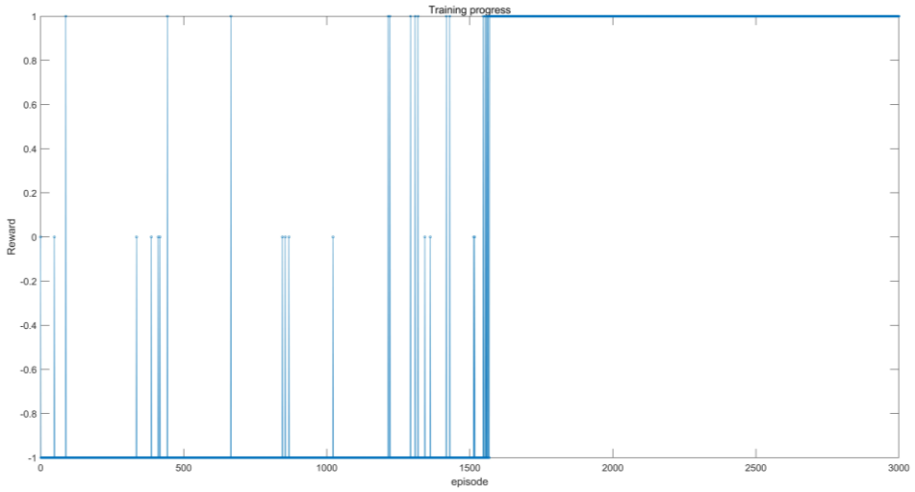


Figure 8. Training progress of the optimized environment

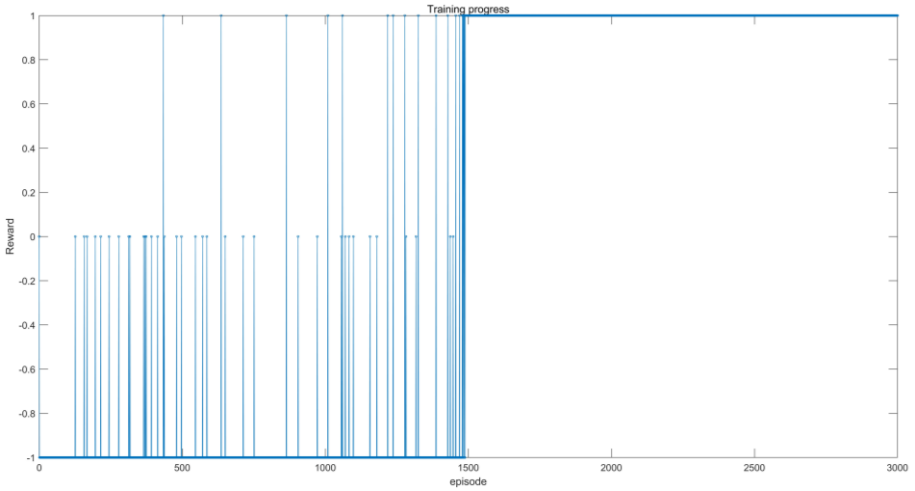


Figure 9. Training progress of the optimized environment

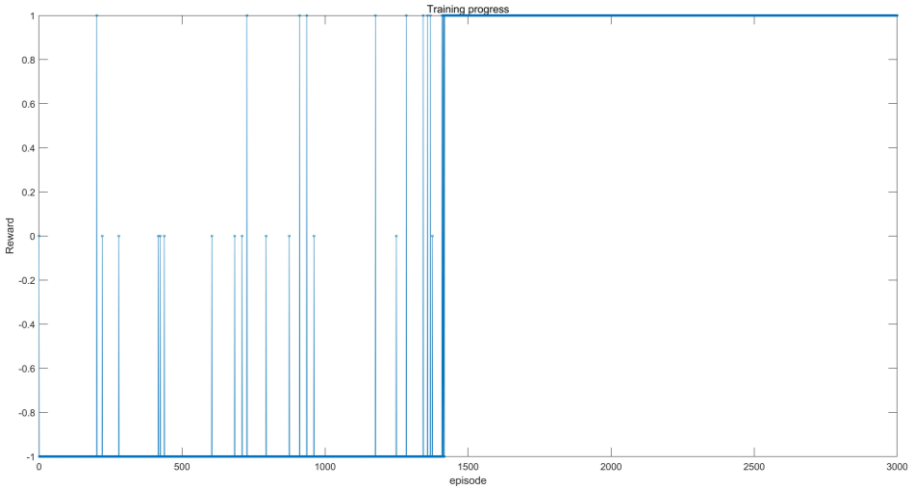


Figure 10. Training progress of the optimized environment

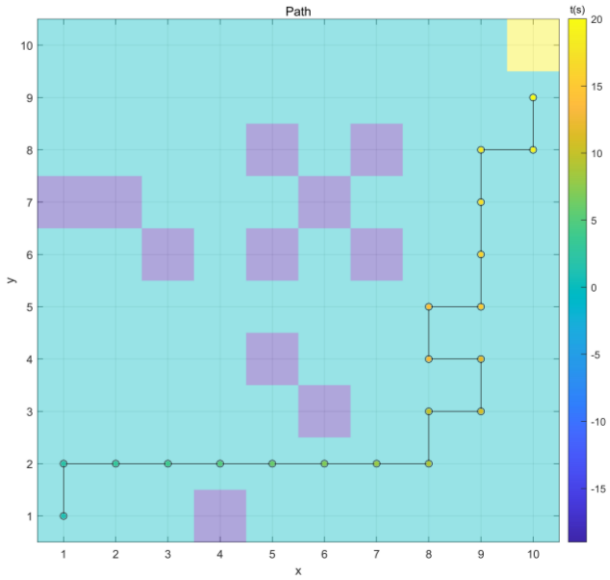


Figure 11. Optimal path in the new environment

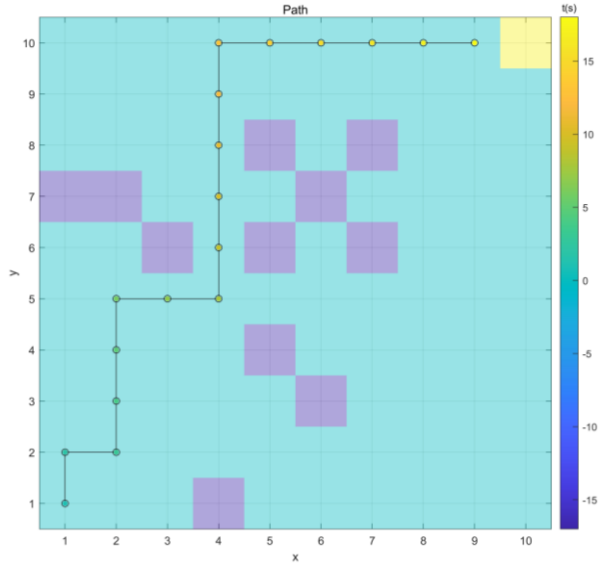


Figure 12. Optimal path in the new environment

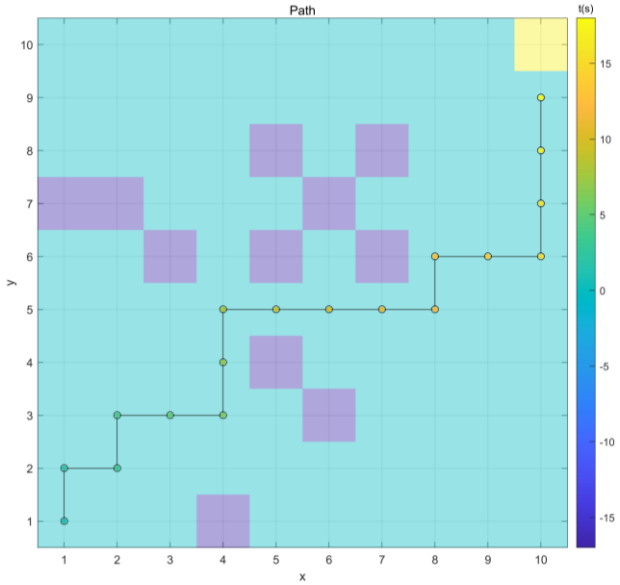


Figure 13. Optimal path in the new environment

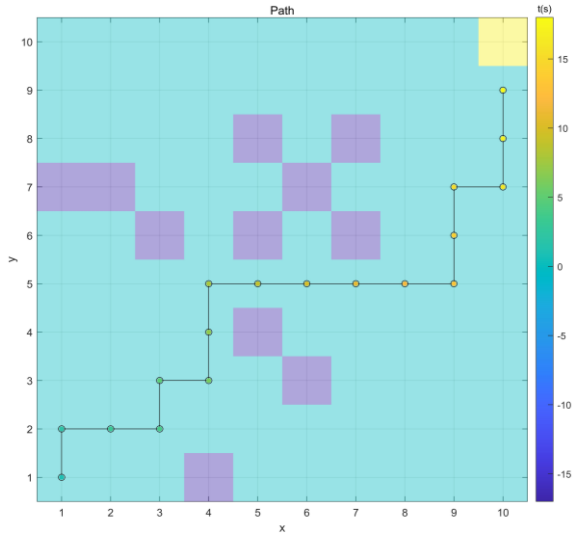


Figure 14. Optimal path in the new environment

Table 1. Q table

-1.0000	0	0	0	0	0	0	0	0	0
0	-0.9000	0.0000	0	0	0	0	0	0	0
0	-1.0000	0	0	0	0	0	0	0	0
0	0	-0.9000	0	0	0	0	0	0	0
0	-1.0000	0	0	0	0	0	0	0	0
-1.0000	0	0	0	0	0	0	0	0	0
0	-0.9000	0	-0.9900	0	0	0	0	0	0
-0.9000	0	0	0	0	0	0	0	0	0
-0.9000	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Table 2. Q table

0.0000	0	0	0	0	-0.9000	0	0	0	0
-0.9000	0.0000	0	0	-0.9000	0	0	0	0	0
0	0.0000	0	-0.9000	0	0	0	0	0	0
0	0.0000	-1.0000	0	-1.0000	0	-0.9000	0	0	0
0	0.0000	0	0	0	0	0	0	0	0
0	0.0000	0	0	0	0	0	0	0	0
0	0.0000	0	0	0	0	0	0	0	0
-0.9000	0.0000	0.0000	0	0.0002	0	0	0	0	0
0	-0.9000	-0.9000	-0.9000	-0.9000	0	-0.9000	0	0.2500	0
0	0	0	0	0	-0.9000	0	-0.9000	0	0

Table 3. Q table

0	0	0	0	-0.9000	0	0	0	0	0
0.0000	0	0	0	-0.9000	0	0	0	0	0
0	0	0	-0.9000	0	0	0	0	0	0
0	0	0	0	-1.0000	0	0	0	0	0
0	-1.0000	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
-0.9000	-1.0000	0	-0.9000	0	0	0	0	0	0
0.0000	0	0	0.0000	-0.9900	0	0	0	0	0
0	0.0000	0.0001	0.0002	0.0010	0.0039	0.0156	0.0625	-0.9000	0
0	0	0	0	0	0	0	0	1.0000	0

Table 4. Q table

-1.0000	0	-1.0000	-1.0000	0	0	0	0	0	0
0	-0.9000	-0.9000	-0.9000	-0.9000	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
-0.9000	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	-0.9000	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0.0103	0	0

5. Conclusion

By studying the Q-learning in path planning and optimizing its environment. MATLAB software is used to simulate the vehicle path planning under the location environment. The robotics playground toolbox in MATLAB software is used to establish and analyze various Q-learning parameters using the toolbox's calculation and update functions, and the training process diagram and final path planning diagram are generated. Through data analysis, the rationality of the environment is proved. The results reveal that the upgraded environment has a noticeable improvement in both convergence velocity and learning efficiency after continuous training, and that it can match theoretical predictions in general, proving its rationale and applicability. The collected data has significant practical implications for future study and Q-learning development. Future research should concentrate on structural optimization of Q-learning algorithms and changes in reflection speed.

References

[1] X. Cui, Z. Liu, et al. "Research on Mobile Robot path Planning Based on improved QLearning," ICMLCA 2021; 2nd International Conference on Machine Learning and Computer Application, Shenyang, China, 2021, pp. 1-5.

[2] Gao X, Wu H, Zhai L, et al. A rapidly exploring random tree optimization algorithm for space robotic manipulators guided by obstacle avoidance independent potential field. International Journal of Advanced Robotic Systems. 2018; 15(3).

[3] Lu Cao, Dong Qiao, Jingwen Xu, Suboptimal artificial potential function sliding mode control for spacecraft rendezvous with obstacle avoidance, Acta Astronautica, Volume 143, 2018, Pages 133-146, ISSN 0094-5765.

[4] C. -F. Juang and Y. -T. Yeh, "Multiobjective Evolution of Biped Robot Gaits Using Advanced Continuous Ant-Colony Optimized Recurrent Neural Networks," in IEEE Transactions on Cybernetics, vol. 48, no. 6, pp. 1910-1922, June 2018.

[5] Zhang, H.-M.; Li, M.-L.; Yang, L. Safe Path Planning of Mobile Robot Based on Improved A* Algorithm in Complex Terrains. Algorithms 2018, 11, 44.

[6] C. Wang, X. Yang and H. Li, "Improved Q-Learning Applied to Dynamic Obstacle Avoidance and Path Planning," in IEEE Access, vol. 10, pp. 92879-92888, 2022.

[7] Gu, S., Mao, G. (2021). An Improved Q-Learning Algorithm for Path Planning in Maze Environments. In: Arai, K., Kapoor, S., Bhatia, R. (eds) Intelligent Systems and Applications. IntelliSys 2020. Advances in Intelligent Systems and Computing, vol 1251. Springer, Cham.

[8] Xie, R., Meng, Z., Zhou, Y., Ma, Y., & Wu, Z. (2020). Heuristic Q-learning based on experience replay for three-dimensional path planning of the unmanned aerial vehicle. Science Progress, 103(1), 0036850419879024.

[9] B. Tan, Y. Peng and J. Lin, "A Local Path Planning Method Based on Q-Learning," 2021 International Conference on Signal Processing and Machine Learning (CONF-SPML), Stanford, CA, USA, 2021, pp. 80-84.

[10] M. Zhao, H. Lu, S. Yang and F. Guo, "The Experience-Memory Q-Learning Algorithm for Robot Path Planning in Unknown Environment," in IEEE Access, vol. 8, pp. 47824-47844, 2020.

[11] Q.Yao et al. "Path Planning Method With Improved Artificial Potential Field—A Reinforcement Learning Perspective," in IEEE Access, vol. 8, pp. 135513-135523, 2020.

[12] Bulut, V. Optimal path planning method based on epsilon-greedy Q-learning algorithm. J Braz. Soc. Mech. Sci. Eng. 44, 106 (2022).

[13] Abderraouf Maoudj, Abdelfetah Hentout, Optimal path planning approach based on Q-learning algorithm for mobile robots, Applied Soft Computing, Volume 97, Part A, 2020, 106796, ISSN 1568-4946.

[14] Ee Soong Low et al. Modified Q-learning with distance metric and virtual target on path planning of mobile robot, Expert Systems with Applications, Volume 199, 2022, 117191, ISSN 0957-4174.

[15] Robotics Playground an Introduction to Mobile Robotics with MATLAB and Simulink Courseware-MATLAB&Simulink(mathworks.cn).https://ww2.mathworks.cn/academia/courseware/robotics-playground.html.