Hindawi
Journal of Sensors
Volume 2021, Article ID 4294841, 14 pages
https://doi.org/10.1155/2021/4294841

Research Article

Improved *Q*-Learning Method for Multirobot Formation and Path Planning with Concave Obstacles

Zhilin Fan,[1] **Fei Liu**,[1] **Xinshun Ning**,[1] **Yilin Han**,[1] **Jian Wang**,[2] **Hongyong Yang**,[1] **and Li Liu**[1]

[1]*School of Information and Electrical Engineering, Ludong University, Yantai 264000, China*
[2]*Yantai Municipal People's Procuratorate, Yantai 264000, China*

Correspondence should be addressed to Fei Liu; liufeildu@163.com

Received 20 June 2021; Accepted 26 October 2021; Published 3 December 2021

Academic Editor: Yunze He

Aiming at the formation and path planning of multirobot systems in an unknown environment, a path planning method for multirobot formation based on improved Q-learning is proposed. Based on the leader-following approach, the leader robot uses an improved *Q*-learning algorithm to plan the path and the follower robot achieves a tracking strategy of gravitational potential field (GPF) by designing a cost function to select actions. Specifically, to improve the Q-learning, *Q*-value is initialized by environmental guidance of the target's GPF. Then, the virtual obstacle-filling avoidance strategy is presented to fill non-obstacles which is judged to tend to concave obstacles with virtual obstacles. Besides, the simulated annealing (SA) algorithm whose controlling temperature is adjusted in real time according to the learning situation of the *Q*-learning is applied to improve the action selection strategy. The experimental results show that the improved *Q*-learning algorithm reduces the convergence time by 89.9% and the number of convergence rounds by 63.4% compared with the traditional algorithm. With the help of the method, multiple robots have a clear division of labor and quickly plan a globally optimized formation path in a completely unknown environment.

1. Introduction

As robots become more and more widely used in various industries, a single robot cannot be competent for complex tasks. Therefore, multirobot formation [1] and path planning have become research hotspots, and they have good applications [2, 3] in collaborative search, exploration, handling, rescue, and group operations. Path planning of multirobot formation requires multiple robots to form a formation and maintain this positional relationship to move to the target. It is necessary not only to avoid obstacles safely but also to find a better path. In addition, compared to the simpler path planning in the known environment, higher requirements on the ability of multiple robots to plan paths are put in the unknown environment. There have been many implementation methods for multirobot formation, including behavior-based method [4], virtual structure method [5], and leader-following method [6]. The behavior-based method is to design sub-behaviors in advance and choose the behavior to execute according to the changes in the situation, but the accuracy is not enough to integrate various behaviors in a complex environment. The virtual structure method regards the formation as a fixed rigid structure and cannot effectively avoid obstacles. The leader-following method with the advantage of simple and flexible structure mainly realizes collaboration by sharing information of leader. For robot's path planning, A∗ algorithm [7] and reinforcement learning (RL) algorithm [8] are commonly used in global path planning; the former can effectively solve the optimal path, but it needs to know all the environmental information in advance; the latter can learn autonomously in the environment, but it takes more time. The artificial potential field (APF) method [9] is widely used in local path planning, which can cope with the real-time changing environment but lacks the global planning ability.

For the problem of multirobot formation and path planning, Chen et al. [10] proposed a new leader-following

control framework by introducing the RH method, which enables fast convergence of a formation task for a group of mobile robots. Based on the path planning of a single robot, Sruthi et al. [11] designed a nonlinear controller for tracking to achieve the multirobot formation. The above two methods require rigorous modeling of the system and cumbersome theory, which are weak in the application. By mixing formation control with leader-following and priority methods, Sang et al. [12] used the MTAPF algorithm with an improved A∗ algorithm for path planning. Das and Jena [13] implemented collision-free path planning for multiple robots by using an improved particle swarm algorithm and evolutionary operators. Qu et al. [14] used a modified genetic algorithm to plan paths for multiple robots by adding a co-evolution mechanism. Lazarowska [15] used the discrete APF to find crash-free paths for robots in dynamic and static environments. Some of the above methods cannot be carried out in an unknown environment and some cannot plan an optimal path.

At present, Q-learning is a widely applied reinforcement learning algorithm. The limitation of Q-learning is that it is trial-and-error learning, which requires constant iteration and is time-consuming. Thus, it needs to be improved to quickly plan a globally optimal path. Maoudj and Hentout [16] initialize the Q-table to accelerate convergence by presenting a distance-based reward function. Soong et al. [17] integrated the prior knowledge gained from FPA into the traditional Q-learning, which provided a good exploration basis for accelerating the learning of mobile robots. Xu and Yuan [18] increased the step length of movement and the direction of the robot to plan a fast and smooth path. Oh et al. [19] specified the initial Q-value of the traditional Q-learning through the fuzzy rule-based Q-learning, which speeded up learning and stabilized convergence. Yan and Xiang [20] initialized the Q-table by using inverse Euclidean distance from the current position to the goal position, which improves the efficiency of Q-learning. The above methods all initialize the Q-value simply by some prior information to improve the algorithm, without considering the avoidance of concave obstacles and the adjustment of the action selection strategy.

In summary, there are still many difficult problems in the formation and path planning of multiple robots in unknown environments. In this paper, we adopt the leader-following approach to study the multirobot dynamic formation problem. The innovation in this paper is as follows: The improved Q-learning algorithm is presented to plan paths, in which environmental guidance and virtual obstacle-filling avoidance strategy are added to accelerate convergence and the SA algorithm is applied to improve the action selection strategy; the follower robot can achieve the tracking strategy of GPF by designing the cost function to select actions.

2. Related Methods

2.1. Q-Learning Algorithm. The Q-learning algorithm [21] is an RL algorithm based on temporal-difference, which combines the Monte Carlo sampling method and the bootstrapping idea of dynamic programming. It is described with the Markov decision process as follows: Firstly, limited state space and action space are given. When the robot needs to accomplish a certain task, it selects and performs the action in the current state, which interacts with the environment. Then, the robot enters the next state and is given an instant reward as feedback by the environment. Finally, the value function is updated according to the update rule by using the reward which is passed to it. One round is continuing the above process until the robot reaches the target, and the rounds are iterated until the cumulative reward is maximum. The update equation of the $Q-$value function is

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha\left[r_t + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)\right], \tag{1}$$

where s_t is the state and a_t is the action at current time t, s_{t+1} is the state and a is the action at next time $t+1$, r_t is the reward obtained by performing action a_t at state s_{t+1}, $\alpha \in (0, 1)$ is the learning rate, and $\gamma \in (0, 1)$ is the discount factor.

In order to ensure the exploratory nature of the algorithm, the ε-greedy strategy is usually adopted, with the probability $1-\varepsilon$ of selecting the action that maximizes the value function, and the small probability ε that is still reserved for random exploration. The mathematical equation of the strategy is

$$\pi(s_t) = \begin{cases} a_{\text{random}}, & \text{if } \delta < \varepsilon, \\ \arg\max_a Q(s, a), & \text{else}, \end{cases} \tag{2}$$

where $\pi(s_t)$ is the selected strategy, $\varepsilon \in (0, 1)$ is the greedy factor, $\delta \in (0, 1)$ is a random number, a_{random} is a randomly selected action, and $\arg\max_a Q(s, a)$ is an action that maximize Q-value at state s.

The classical Q-learning algorithm is described in Algorithm1.

2.2. APF Method. The APF method is a virtual potential field established artificially, including the GPF and the repulsive potential field. The target generates a gravitational force on the robot to make the robot move towards it. The GPF function is

$$U_{\text{gra}}(q) = \frac{1}{2}\xi d^2\left(q, q_{\text{goal}}\right), \tag{3}$$

where ξ is the GPF factor, q is the state position of the robot, $U_{\text{gra}}(q)$ is the GPF at q, q_{goal} is the state position of the target to be reached by the robot, and $d(q, q_{\text{goal}})$ is the distance between the robot and the target, which can be measured by specific sensors in practice and is one-dimensional. Gravitation is the negative gradient of the GPF, and the gravitational function is defined as

$$F_{\text{gra}} = -\nabla U_{\text{gra}} = -\xi d\left(q, q_{\text{goal}}\right). \tag{4}$$

begin
Initialization:
$Q(s, a) = \{0\}$, $\forall s \in S$, $a \in A(s) = \{\text{up, down, left, right}\}$ %Initialize Q value with 0, determine the state set and the action set containing four actions
for(episode < m) %The episode cannot exceed m which is the maximum number of episodes
Given initial state s_0;
while ($s_t \neq$ target state)
(1) Select an action a_t at state s_t according to ε-greedy; % ε-greedy is the action selection strategy;
(2) Execute the action a_t, then enter state s_{t+1} and get a reward r_t; %Get immediate rewards by performing actions to interact with environment
(3) Update $Q(s_t, a_t)$ using $Q(s_t, a_t) = Q(s_t, a_t) + \alpha[r_t + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)]$;
% Update the value function according to the update equation by using the reward
(4) $s_t \longleftarrow s_{t+1}$; %Update state
end-while
Episode = episode + 1; % Update episode
end-for
end

Algorithm 1: Classical Q-learning algorithm.

Obstacles generate a repulsive force on the robot to make the robot move away from it. The repulsive potential field function is

$$U_{\text{rep}}(q) = \begin{cases} \frac{1}{2}\eta\left(\frac{1}{d(q, q_{\text{obst}})} - \frac{1}{d_0}\right), & \text{if } d(q, q_{\text{obst}}) \le d_0, \\ 0, & \text{if } d(q, q_{\text{obst}}) > d_0, \end{cases} \tag{5}$$

where η is the repulsive potential field factor, q is the state position of the robot, $U_{\text{rep}}(q)$ is the repulsive potential field at q, q_{obst} is the obstacle state position, $d(q, q_{\text{obst}})$ is the distance between the robot and the obstacle, which can be measured by specific sensors in practice and is one-dimensional. d_0 is the influence radius of the obstacle, which is artificially set according to the experimental requirements in practice. Repulsion is the negative gradient of the repulsion potential field, and the repulsion function is defined as

$$F_{\text{rep}} = -\nabla U_{\text{rep}} = \begin{cases} \eta\left(\frac{1}{d(q, q_{\text{obst}})} - \frac{1}{d_0}\right)\frac{1}{d^2(q, q_{\text{obst}})}\nabla d(q, q_{\text{obst}}), & \text{if } d(q, q_{\text{obst}}) \le d_0, \\ 0, & \text{if } d(q, q_{\text{obst}}) > d_0. \end{cases} \tag{6}$$

Therefore, the traditional APF method guides the robot's direction of movement based on the combined force of gravitation and repulsion, but its shortcomings are as follows:

(1) When the robot is far away from the target, the gravitational force is much greater than the repulsive force, and it may hit an obstacle

(2) When the distance between them is relatively close, the obstacles will repel the robot too much to reach the target

(3) When the two reaction forces just cancel out, the phenomenon of local optimum or oscillation may appear

Because of the above shortcomings, the APF method generally cannot be used directly and needs to be improved to use.

2.3. SA Algorithm. The idea of the SA algorithm comes from the solid annealing process, which is an algorithm that jumps out of the local optimum to get the global optimum. The algorithm uses temperature parameters T to control convergence in a finite time. Firstly, the initial temperature and the end temperature are set. The algorithm starts from the initial state and takes it as the current state. Then, it generates a new state in its neighborhood and determines whether to accept the new state based on the Metropolis criterion. The generation process of the new state iterates while the T decays until T is the end temperature. Finally, the algorithm ends with the global approximate optimal solution.

The Metropolis criterion is that when a system enters a state s_{new} due to a certain change in state s_{old}, the energy of the system correspondingly changes from $E(s_{\text{old}})$ to $E(s_{\text{new}})$ and then the accepted probability equation of the system from s_{old} to s_{new} is

$$P(s_{\text{old}} \longrightarrow s_{\text{new}}) = \begin{cases} 1, & E(s_{\text{new}}) \le E(s_{\text{old}}), \\ e^{E(s_{\text{new}})-E(s_{\text{old}})/T}, & E(s_{\text{new}}) > E(s_{\text{old}}). \end{cases} \tag{7}$$

When $E(s_{\text{new}}) \le E(s_{\text{old}})$, the new state is accepted as the current state. When $E(s_{\text{new}}) > E(s_{\text{old}})$, if $e^{E(s_{\text{new}})-E(s_{\text{old}})/T} > \delta$, the new state is accepted as the current state; otherwise, the new state is not accepted and the system remains in the current state. $\delta \in (0, 1)$ is the randomly generated number.

3. Improved Q-Learning Proposed for Path Planning of Leader Robot

3.1. Environmental Guidance Based on GPF of Target. The traditional Q-learning algorithm has no prior knowledge. In the early learning process, the robot's aimless exploration causes many invalid iterations and slow convergence. So, the idea of the APF method is introduced to guide moving. In this paper, the robot plans a path in an unknown environment where only the start and the target of the task are known. Due to the less environmental information and the shortcomings of the traditional APF method, only the GPF of the target is introduced to initialize the Q value without considering the effect of the repulsive potential field. In order to make the target direction consistent with the increasing direction of the Q-value, the GPF function is constructed as

$$U'_{\text{gra}}(s) = \xi * \frac{d_{\text{aim}}(s)}{d_{\text{aim}}(s) + \eta}, \tag{8}$$

where ξ is the GPF factor which is negative and controls the value inversely proportional to the distance, $d_{\text{aim}}(s)$ is the distance from the current position to the target, and η is a positive constant that prevents the denominator from being 0.

When the robot moves, the instant reward is detected by sensors and the Q-table is initialized at the same time. Therefore, the instant reward of environmental information is added to the Q-value initialization. The purpose of RL is to maximize the cumulative reward by maximizing the Q-value. The robot always tends to choose the action with the maximum Q-value, which will guide the robot to move toward the target while avoiding obstacles. The mathematical equation of Q-value initialization with environmental guidance based on GPF of the target is

$$Q(s, a) = k_q * \left(r_q + \gamma * U'_{\text{gra}}(s)\right), \tag{9}$$

where $r_q = \begin{cases} 1, & \text{at target} \\ 0, & \text{else} \\ -1, & \text{at obstacle} \end{cases}$, k_q is the scale coefficient adjusted according to the actual algorithm, γ is the discount factor, and $U'_{\text{gra}}(s)$ is the GPF at state s.

3.2. Virtual Obstacle-Filling Avoidance Strategy. There will be concave obstacles in a more complex environment. The traditional Q-learning algorithm can escape from such obstacles through continuous exploration, which greatly extends the learning time. In addition, the robot is more likely to fall into concave obstacles and cannot escape after adding GPF guidance. In the grid map environment, the obstacle grid is the infeasible area and the rest are feasible areas. Setting certain key position grids which is feasible in the path of possibly tending to concave obstacles as infeasible areas can effectively fill and avoid concave obstacles. Therefore, a virtual obstacle-filling avoidance strategy is established for concave obstacles. The strategy is to judge whether the current grid possibly tends to concave obstacles by adding real-time detection information based on the target tendency before the robot takes the next step. Then, it fills non-obstacles on the path of possibly tending to the concave obstacle with virtual obstacles until the concave shape is filled. The filled concave obstacle as a whole becomes an infeasible area, so the robot will not fall into the concave obstacle in subsequent iterations. This strategy makes full use of sensors and the environmental information which have been learned. It not only prevents the robot from falling into concave obstacles but also reduces invalid exploration of some infeasible positions, which improves the efficiency of path planning.

The specific implementation of the virtual obstacle-filling avoidance strategy is as follows.

Firstly, the sensor is used to detect the position status and distance in real time. And a current position-action array is established to store the feasible adjacent positions from the current position. Before the robot moves, the Euclidean distance from the $3 * 3$ grid positions adjacent to the robot's current position to the target position is calculated in turn. Next are the specific judgement steps to judge whether the current grid possibly tends to concave obstacles according to the calculated distances.

If the distance is less than the distance from the current position to the target position, it is further judged whether this adjacent position is an obstacle or not. If the adjacent position is not an obstacle, it is feasible and will be added to the corresponding position of the current position-action array.

If the adjacent position is further away from the target or it is further judged an obstacle, it is an infeasible position and will not be added to the corresponding position.

If the final current position-action array is empty, it indicates that the current position completely tends to infeasible areas which may be in a concave obstacle. The current position will be filled with a virtual obstacle.

Finally, each step of the robot is judged until concave obstacles are filled.

One-step filling of the virtual obstacle-filling avoidance strategy is shown in Figure 1. In the figure, the red grid is the robot, and the yellow grid is the target. As Figure 1(a) shows, the robot enters the grey concave obstacle during the path planning process. According to the distance calculation, the adjacent positions which are down, right, and lower right of the robot's current position are determined

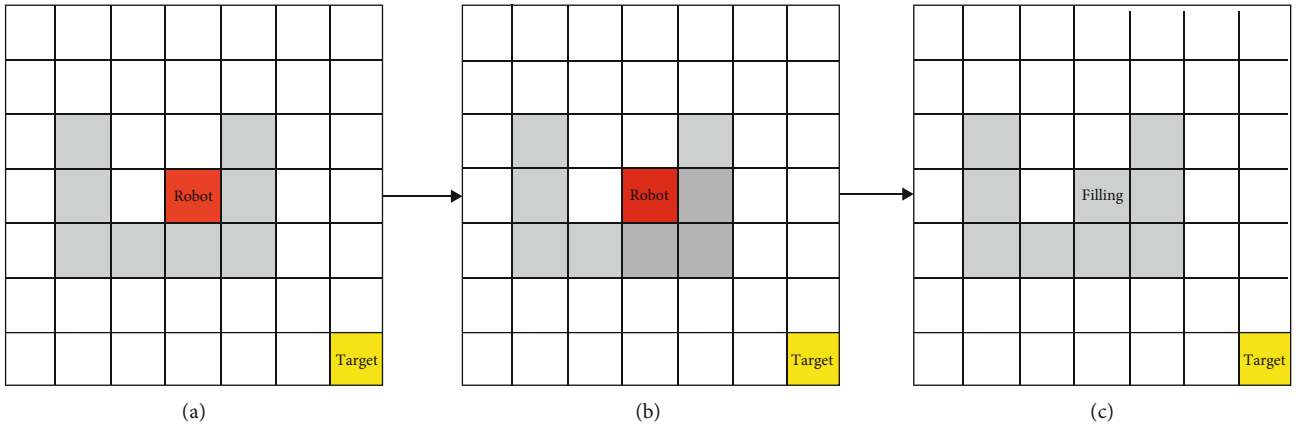


Figure 1: One-step filling of the virtual obstacle filling avoidance strategy [22].

begin
Initialization:
$Q(s, a) = \varnothing$, current$(s, a) = \varnothing$, n, T, $\forall s \in S$,
$a \in A(s) = \{$up, down, left, right, upleft, upright, downleft, downright$\}$
%Establish Q-table and current position-action array, define n as times of consecutive iterations, define T as the initial temperature, determine the state set and the action set containing eight actions
for (episode < m) %The episode cannot exceed m which is the maximum number of episodes
Given initial state s_0;
If episode%n == 0 Then use $\varepsilon = e^{(Q(s,a_{\text{random}})-Q(s,a_{\max})-q)/T}$ to calculate and update ε; %Adjust ε dynamically using T of the SA algorithm.
while ($s_t \neq$ target state)
(1) If s_t exists in the Q-table then continue to next step;
Else use $Q(\text{s}, \text{a}) = k_q * (r_q + \gamma * \xi * (d_{\text{aim}}(s')/(d_{\text{aim}}(s') + \eta)))$ to initialize $Q(s_t, a)$;
%Initialize the Q-table
(2) If (s_t, a) is a feasible area towards the target then add it to the corresponding position of current(s_t, a);
Else the corresponding position of current(s_t, a) is kept empty;
% Add the feasible adjacent positions from the current position to the current position-action array
(3) If current(s_t, a) is empty then (s_t, a) is completely toward the infeasible area which possibly tends to concave obstacles and fill it with virtual obstacle;
Select action a_t in state s_t according to ε-greedy which is improved by SA;
%Fill concave obstacles using the virtual obstacle-filling avoidance strategy while selecting actions
(4) Execute a_t in s_t, enter s_{t+1} and get r_t;
(5) Update $Q(s_t, a_t)$ using $Q(s_t, a_t) = Q(s_t, a_t) + \alpha[r_t + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)]$;
(6) $s_t \longleftarrow s_{t+1}$;
end-while
Episode = episode + 1;
end-for
end

Algorithm 2: Improved Q-learning algorithm.

as the positions which are near the target more and are the dark grey grid in Figure 1(b). The three adjacent positions are further judged obstacles which are infeasible positions, indicating that the current position is completely toward the infeasible area which may be in a concave obstacle. Thus, the current position is filled with light grey virtual obstacles, which is seen in Figure 1(c).

3.3. Action Selection Strategy Improved by SA. In the process of path planning with Q-learning, the robot expands the range of movement by exploring the environment and accumulates knowledge of environmental rewards and punishments. Finally, it uses the value function to select the optimal action. In the robot's iterative learning process, more exploration is required in the early stage, but too much or too long exploration will greatly extend the learning time and reduce the learning efficiency. On the contrary, too little exploration will lead to insufficient experience and the action selected finally may be sub-optimal. Thus, it is necessary to balance exploration and utilization. The traditional ε-greedy strategy

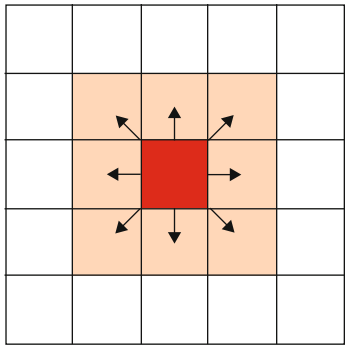


Figure 2: The multi-robot action, step length, and detection range of sensor[22].

often used in the Q-learning algorithm balances exploration and utilization to a certain extent by setting ε. However, the fixed greedy factor in the learning process makes random actions selected with the same probability each time, which causes slow convergence and fluctuations after convergence. Therefore, the greedy factor needs to be adjusted dynamically with the learning process. One method commonly used in experiments is to set ε to decrease at a fixed rate, but it is not universal to set a fixed rate of decrease based on experience.

In response to the above problems, the ε-greedy strategy improved by SA which is used to adjust ε dynamically is proposed. The controlled temperature of SA is adjusted in real time according to the learning situation of the Q-learning algorithm. The algorithm explores as much as possible in the early stage of path planning to increase more prior knowledge and prevent local optimum and cancels unnecessary exploration when approaching convergence later. The steps of the action selection strategy improved by the SA algorithm are as follows:

(1) Define the temperature parameter T and set the initial value T_0. Then, use the sample standard deviation of step numbers for n consecutive iterations to control the cooling temperature. The mathematical equation of T is

$$T = i + k * \sqrt{\frac{\left(\text{step}_{m+1} - \text{step}_{\text{avg}}\right)^2 + \cdots + \left(\text{step}_{m+n} - \text{step}_{\text{avg}}\right)^2}{n-1}}, \tag{10}$$

where $\text{step}_{m+1}, \cdots, \text{step}_{m+n}$, respectively, are the number of steps for n consecutive iterations, step_{avg} is the average number of n consecutive iterations, and k is the control factor, which is obtained by repeatedly adjusting according to the experimental effect and controls T in a suitable range. i is a smaller non-zero constant to prevent T from being 0 after convergence

(2) Calculate the accepted probability of randomly selected actions according to the Metropolis criterion. And use it to redefine the greedy factor ε at T. The mathematical equation of ε is

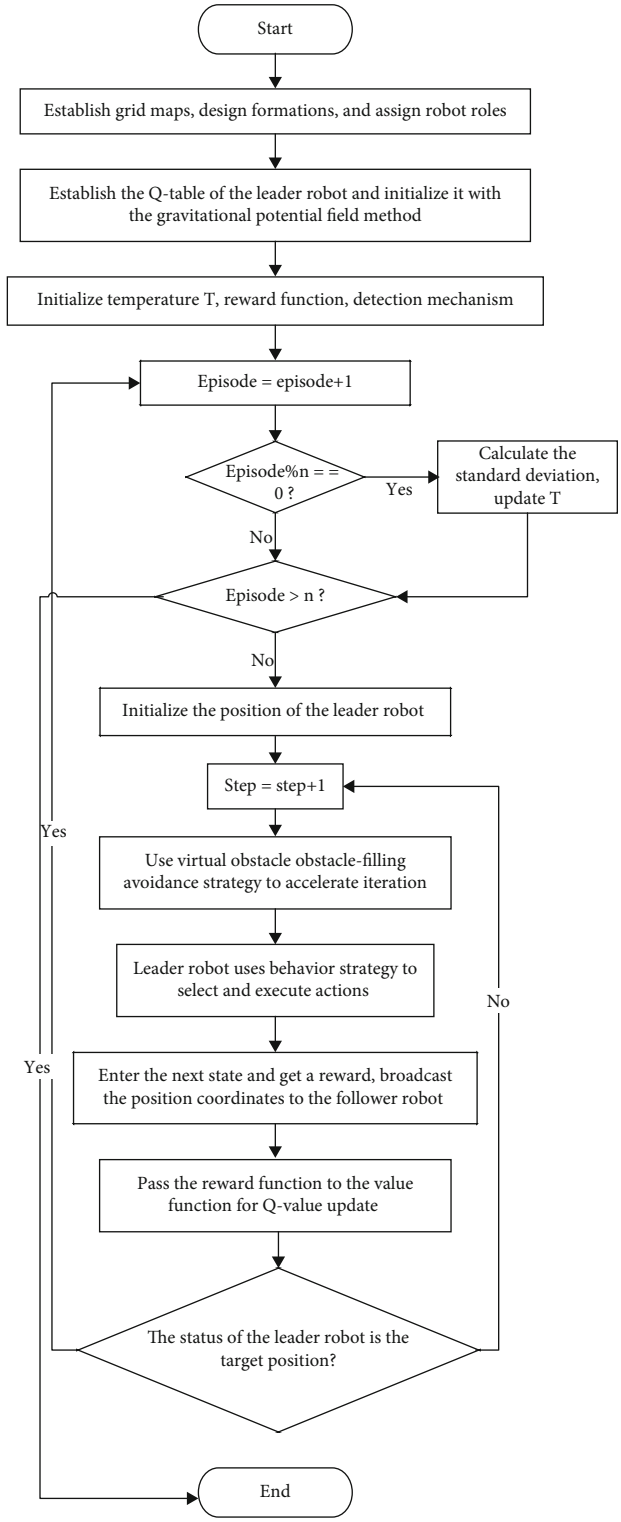


Figure 3: The flow chart of the leader robot's path planning.

$$\varepsilon = e^{(Q(s,a_{\text{random}}) - Q(s,a_{\max}) - q)/T}, \tag{11}$$

where $Q(s, a_{\text{random}})$ is the Q-value of the random action selected at state s, $Q(s, a_{\max})$ is the Q-value of the optimal action at state s, q is a non-zero constant

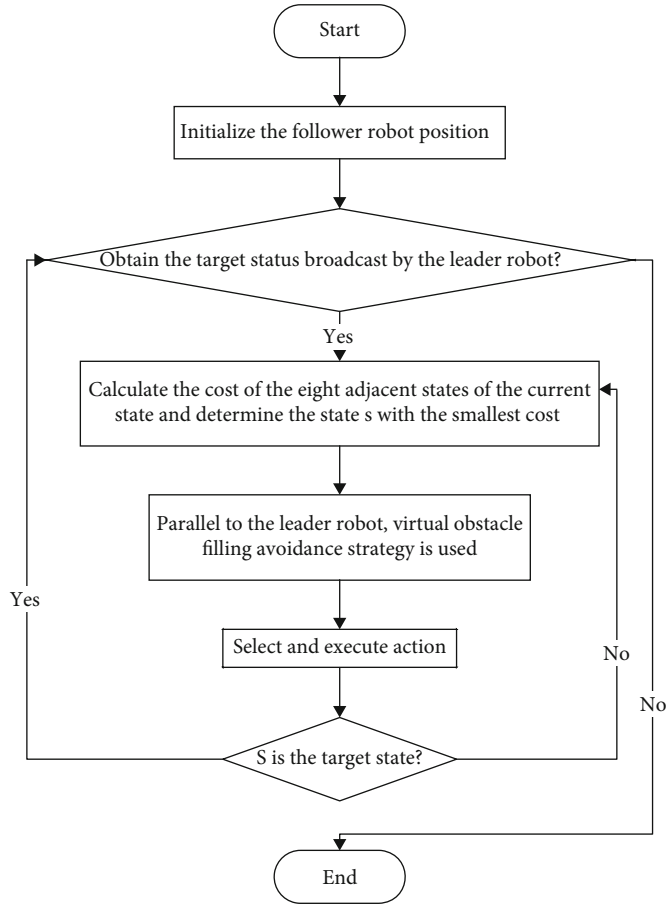


Figure 4: The flow chart of the follower robot's path planning.

Table 1: Implementation details of Q-L1 to Q-L5 algorithms.

Algorithm number	Implementation details
Q-L1	Algorithm 1
Q-L2	Q-L1 with the dynamic greedy factor of SA
Q-L3	Q-L2 with environmental guidance
Q-L4	Algorithm 2
Q-L5	Q-L4 with a modified reward function

to prevent the numerator from being 0, and T is the temperature parameter

(3) If $\delta < \varepsilon$, choose the action randomly, otherwise choose $\arg\max_a Q(s, a)$, where δ is the randomly generated number

3.4. Improved Q-Learning Algorithm. Compared with the original algorithm, there are three innovations in the improved Q-learning algorithm proposed in this paper.

Firstly, the Q-table of the original Q-learning algorithm is initially a zero-value table without any prior knowledge. The improved Q-learning algorithm uses the GPF of the known target in the task to initialize the Q-table, which adds environmental guidance and reduces invalid exploration.

Secondly, the robot moves immediately after selecting an action in the original algorithm. This algorithm designs a virtual obstacle-filling avoidance strategy for judgment before each step. It fills non-obstacles which is judged to tend to concave obstacles with virtual obstacles.

Finally, the original algorithm uses the traditional ε-greedy strategy to select actions. The strategy improved by the SA algorithm is proposed in the new algorithm. It adjusts ε dynamically by adjusting the temperature in real time according to the learning situation of Q-learning.

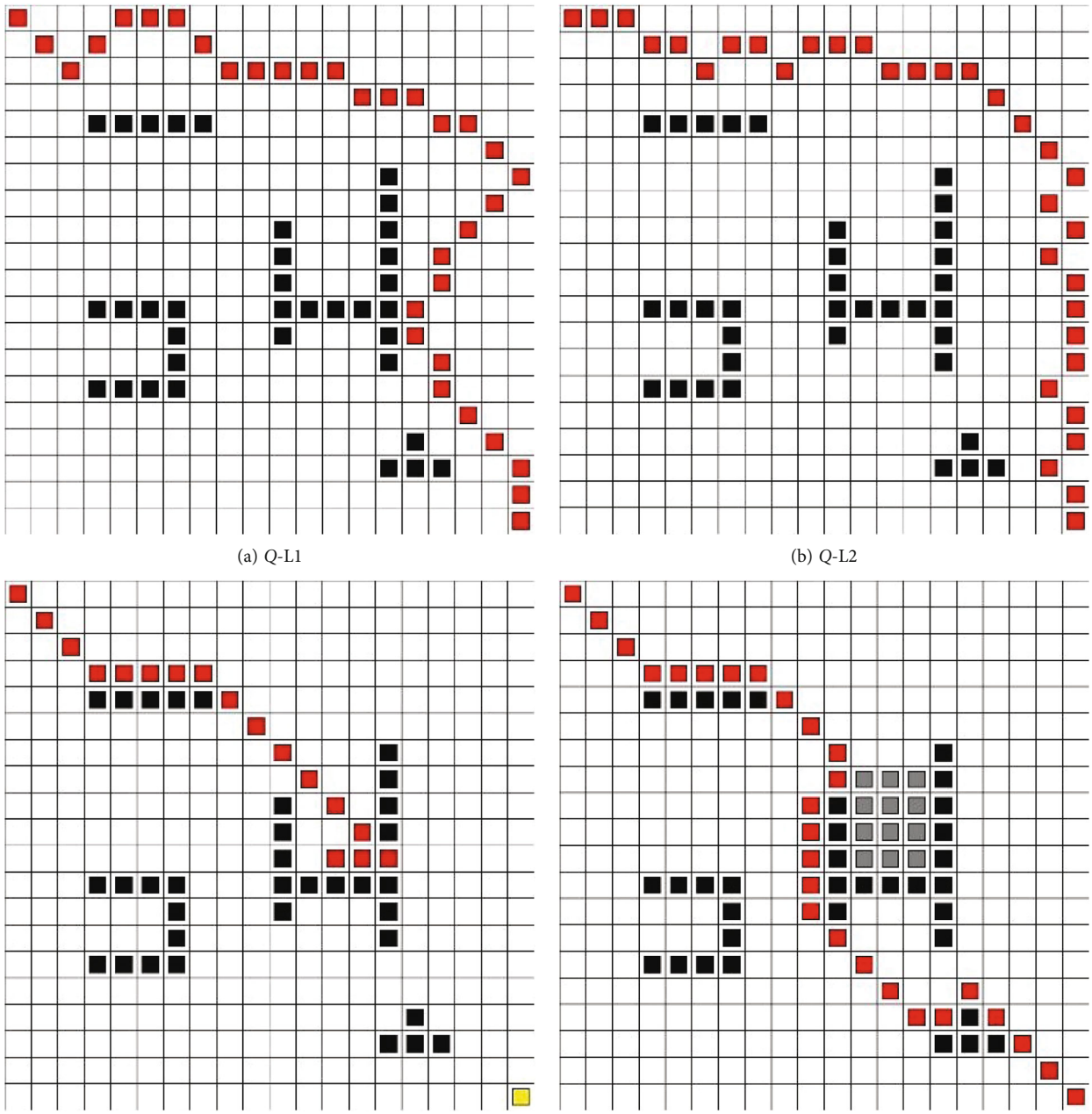
The steps of the improved Q-learning algorithm are shown in Algorithm 2.

4. A Path Planning Method for multirobot Formation

4.1. Tracking Strategy Based on GPF for Follower Robot. The steps of the tracking strategy based on GPF for the follower robot are as follows:

Step 1: if the follower robot obtains the coordinates broadcast by the leader robot, it will determine the next target state according to the formation, i.e., the desired target position at this time. Otherwise, it means that the formation has reached the target position and the path planning ends.

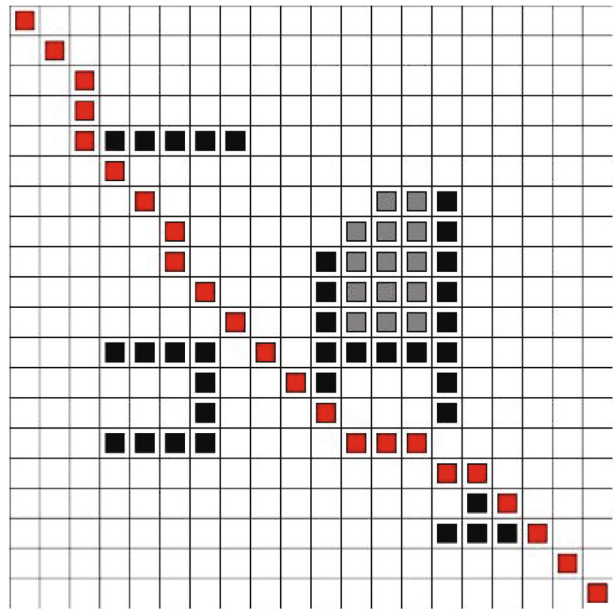
Step 2: the follower robot moves to the target position. Firstly, the robot uses the cost function to calculate the cost



(a) Q-L1 (b) Q-L2

(c) Q-L3 (d) Q-L4

Figure 5: Continued.



(e) Q-L5

Figure 5: Path planning maps of 5 Q-learning algorithms.

for the eight neighboring states of the current state, which determines the state s with the smallest cost. Then, it selects the corresponding action and executes it. At the same time, it adopts the virtual obstacle-filling avoidance strategy in parallel with the leader robot to share information. Specifically, the cost function is designed by using the idea of GPF. The GPF of the target to the current position is measured by the Euclidean distance from the current position to the target position, which is proportional to the distance. When the checked state is an obstacle, the penalty function R_{static} is given a positive value; otherwise, the value is 0. The equation for measuring the GPF is

$$d_{\text{attr}} = \sqrt{\left(x_s - x_{\text{goal}}\right)^2 + \left(y_s - y_{\text{goal}}\right)^2}. \tag{12}$$

The cost function equation is

$$C(s_t, a_t) = c * d_{\text{attr}} + R_{\text{static}}(s_t, a_t), \tag{13}$$

where d_{attr} is the GPF which is measured, x_s is the horizontal coordinate of the current state, y_s is the vertical coordinate of the current state, x_{goal} is the horizontal coordinate of the target at this moment, y_{goal} is the vertical coordinate of the target at this moment, $C(s_t, a_t)$ is the cost function at t, s_t is the state at t, a_t is the action at t, c is the adjustment coefficient, and $R_{\text{static}}(s_t, a_t)$ is the penalty function.

Step 3: if the state with the minimum cost entered is the target state at this time, return to step 1 and continue. If the state is not the target state at this time, go to step 2 and continue.

4.2. Design Scheme of Path Planning for Leader-following Formation. Adopting the leader-following method, the design scheme of path planning for leader-following formation proposed in this paper includes three parts:

(1) Initialization: the grid environment is adopted, and the starting position and target position of the multiple robots are determined. A leader-following formation is designed and the robots are divided into two types: leader and follower. Then, one robot is selected as the leader or a virtual robot is supposed to act as the leader, and the rest are follower robots. Multiple robots have eight actions including up, down, left, right, upper left, upper right, lower left, and lower right. Each robot is equipped with a sensor, which can detect the environmental information of the $3 * 3$ grids centering on its position. The multirobot action, step length, and detection range of the sensor are shown in Figure 2

(2) Path planning of leader robot: the leader robot is responsible for planning the path. It uses the improved Q-learning algorithm to plan a globally optimal path with avoiding simple obstacles and concave obstacles after trial-and-error training. At the same time, it broadcasts the position of each step and some environmental information to the follower robot. The process of the leader robot's path planning is shown in Figure 3

(3) Local following of follower robot: the follower robot is responsible for following the leader robot to maintain the formation according to the requirements. When the follower robot receives the position

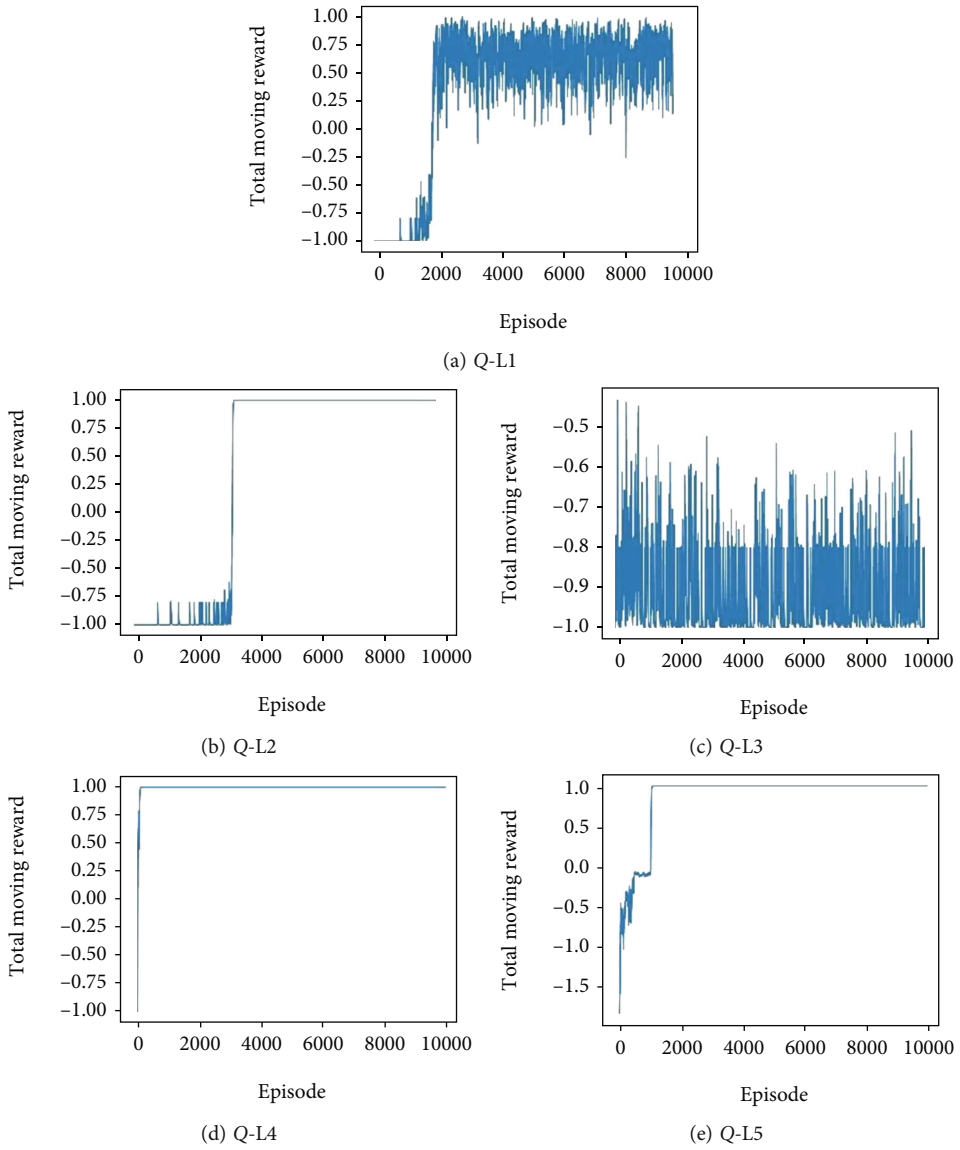


Figure 6: The cumulative reward change graphs with rounds of 5 Q-learning algorithms.

information broadcast by the leader robot, it determines the desired target depending on the formation. Then, it follows locally using the tracking strategy based on the GPF and avoids obstacles autonomously. The process of the follower robot's path planning is shown in Figure 4

5. Experiments Analysis

According to the design scheme of path planning for multiple robots, the method is tested experimentally. The experiment uses Python standard GUI toolkit Tkinter to establish simulation environments.

5.1. Comparison Experiments of Improved Q-Learning Algorithm. For the improved Q-learning algorithm, a grid map with three elements: starting point, target point, and obstacles, is first established. The map size is set to 20×20 grids, and the resolution of each grid is 26×26 pixels. The starting position of the robot represented by a red grid is set at (0, 0), and the target position represented by a yellow grid is at (19, 19). Obstacles which are black grids are randomly placed on the map, including concave and simple obstacles. To distinguish actual obstacles from virtual obstacles filled during the algorithm operation, virtual obstacles are gray grids.

The experiment is carried out in a comparative way, and five algorithms are implemented: Q-L1 is the traditional Q -learning algorithm, Q-L2 is the Q-learning algorithm with the dynamic greedy factor of SA, Q-L3 adds environmental guidance of GPF on the basis of Q-L2, Q-L4 is the Q -learning algorithm proposed in this paper with the improvements 3.1, 3.2, and 3.3, Q-L5 is the Q-learning algorithm with a modified reward function based on Q-L4. The implementation details of Q-L1 to Q-L5 algorithms are shown in Table 1.

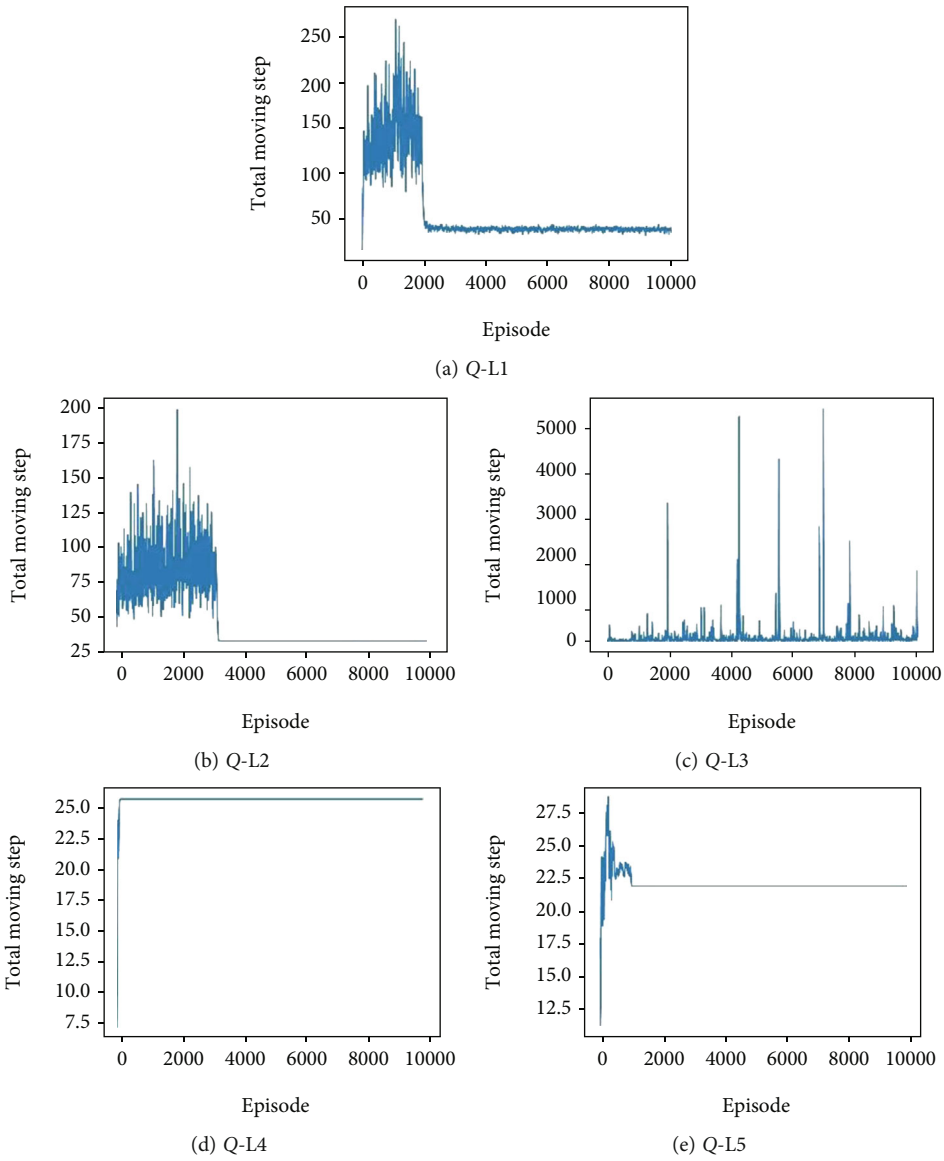
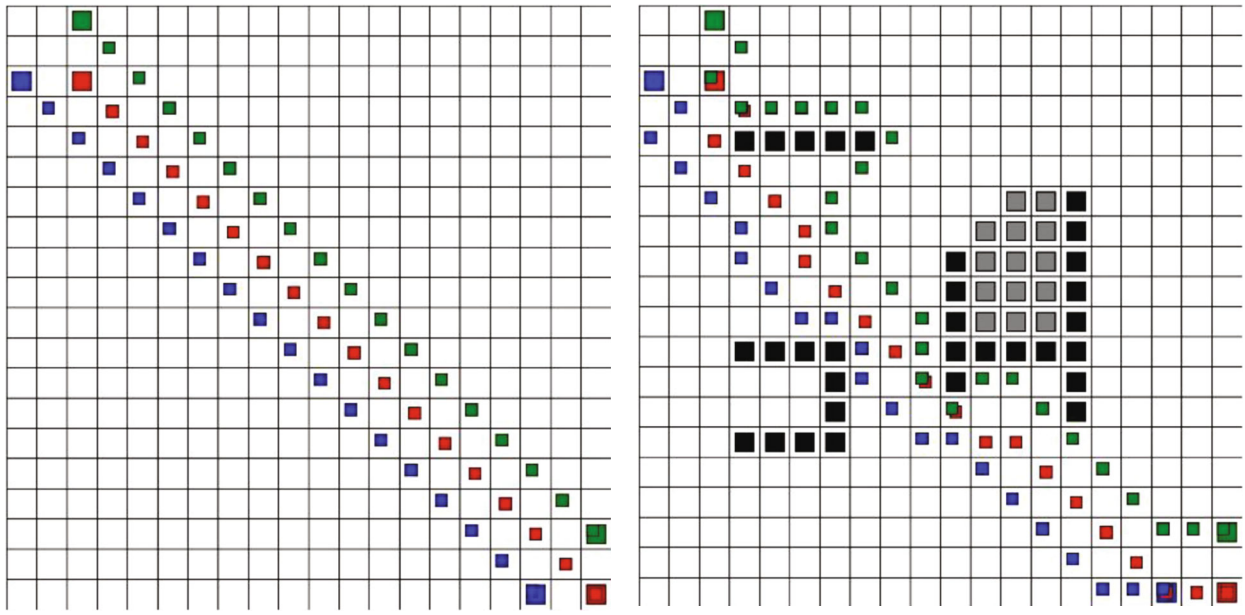


Figure 7: The step numbers change graphs with rounds of 5 Q-learning algorithms.

Table 2: Comparison table of Q-L1 to Q-L5 algorithm performance.

Algorithm	Potential field	Filling	r	ε	Convergence time	Convergence round	Steps	Length
Q-L1	No	No	$r1$	0.2	479.0037	2760	30	37.4558
Q-L2	No	No	$r1$	Dynamic	379.4773	3520	30	36.0416
Q-L3	Yes	No	$r1$	Dynamic	No	No	No	No
Q-L4	Yes	Yes	$r1$	Dynamic	5.7183	120	26	32.6274
Q-L5	Yes	Yes	$r2$	Dynamic	48.3259	1010	22	28.6274

The same parameter settings of the algorithm are as follows: the maximum number of iteration rounds is 10000, the learning rate α is 0.01, and the discount factor γ is 0.9. For using the traditional ε-greedy strategy in the algorithm, the greedy factor ε is 0.2, and the convergence is determined that the standard deviation of step numbers for 10 consecutive iterations is less than 5. Parameter settings for using SA in the algorithm are as follows: the initial temperature T_0 is set to 10, the number of consecutive iterations n is set to 10, the constant i is set to 0.1, the control factor k is set to 0.03, and the non-zero constant q is set to 1. In the algorithm using the GPF method to improve, the GPF factor ξ



(a) In an obstacle-free environment (b) In a static obstacle environment

Figure 8: Map of multi-robot formation's path planning experiments.

is set to -10, the constant η is set to 735, and the scale coefficient k_q is set to 0.1. The reward function is set to

$$r1 = \begin{cases} 1, & \text{reach the target,} \\ 0, & \text{else,} \\ -1, & \text{hit an obstacle,} \end{cases} \qquad r2 = \begin{cases} 1, & \text{reach the target,} \\ -0.05, & \text{else,} \\ -1, & \text{hit an obstacle.} \end{cases} \tag{14}$$

After setting the parameters, simulation experiments are conducted. The path planning map, the cumulative reward change graph with the round, and the path planning step numbers change graph with the round are obtained. From the figure, the path planning and convergence of each algorithm can be seen. Figures 5(a)–5(e), respectively, show the path planning maps of the robot under algorithms Q-L1 to Q-L5. Figures 6(a)–6(e), respectively, show the change of cumulative reward with rounds for the robot under algorithms Q-L1 to Q-L5. Figures 7(a)–7(e), respectively, show the change of step numbers with rounds for the robot under algorithms Q-L1 to Q-L5.

Figure 5(c) shows that the robot is trapped in a concave obstacle and cannot escape. Figure 6(c) shows that the cumulative reward curve of path planning changes irregularly during the iterative process. Figure 7(c) shows that the step number curve of path planning changes irregularly during the iterative process. The above three results of Q-L3 ndicate the algorithm does not converge in the iterative process, and the robot cannot reach the target when only adding the GPF of the target to improve when encountering concave obstacles.

Figures 5(a)–5(e) show that the robot uses the Q-L1, Q-L2, Q-L4, and Q-L5 algorithms to effectively avoid black obstacles and plan a red path from the starting to the target, but the path planned by the Q-L1 and Q-L2 algorithms is more tortuous, the Q-L4 algorithm plans a smoother feasible path, and the Q-L5 algorithm plans the optimal path. The cumulative reward curve showed by Figures 6(a)–6(e) and the step number curve showed by Figures 7(a), 7(b), 7(d), and 7(e) are both stable after iterating to a certain round, indicating that the algorithms gradually converge as the iterations proceed.

However, curves of Figures 6(a) and 7(a) converges with small fluctuations, curves of Figures 6(b) and 7(b) converges with smoothness, curves of Figures 6(d), 7(d), 6(e), and 7(e) reach smoothness in fewer rounds of iteration. The above shows that by adding the improvements proposed in this paper to Q-L4 algorithm and Q-L5 algorithm, the experiments achieve better results. The robot moves while initializing the Q-value by the environmental guidance based on the GPF of the target, which makes the robot guided by the target direction all the time. It removes invalid movement and speeds up the convergence time. When the robot encounters a concave obstacle, it identifies effectively the infeasible area and fills it with light gray virtual obstacles to prevent the robot from falling into the concave obstacle. The SA method is used to dynamically adjust the greedy factor to accelerate the algorithm convergence and make it stable after convergence. In addition, the Q-L5 algorithm adjusts the reward function on the basis of the Q-L4 algorithm by giving each step a smaller penalty, and the robot learns the optimal path with the maximum cumulative reward.

Table 2 compares the performance of the above five algorithms after path planning. The data in the table are the average results from conducting several experiments. The analysis is as follows: based on the traditional Q-learning

algorithm, Q-L2 uses the SA algorithm to improve ε-greedy strategy. Although the convergence round of the algorithm increases, the convergence time is shortened and the overall stability of path planning is improved. Comparing the algorithms Q-L3 and Q-L4, if the environmental guidance of GPF is added to the algorithm without the virtual obstacle-filling avoidance strategy, the algorithm is difficult to converge when encountering concave obstacles. Comparing the algorithms Q-L2 and Q-L4, by adding environmental guidance and the virtual obstacle-filling avoidance strategy, the convergence time is reduced by 98.5%, and the convergence rounds is reduced by 96.6%; the total step numbers and the length of the path are stabilized at 26 and 32.6274, respectively. Comparing the algorithms Q-L4 and Q-L5, adjusting the reward function reasonably on the improved Q-learning algorithm proposed in this paper will make the robot plan the optimal path quickly, which is 89.9% shorter than the traditional Q-learning algorithm. And the number of convergence rounds is reduced by 63.4%, the step numbers are reduced to 22 and the length of the path is reduced to 28.6274.

5.2. Experiments of Path Planning for multirobot Formation. After experimenting with the improved algorithm of the leader robot's path planning, the follower robot is added to verify the effectiveness of the path planning method for multirobot formation in this paper. The experiment uses a triangular formation and three robots. The leader robot is represented by a red grid, and its initial position is $(2, 2)$. The follower robots are represented by a blue grid and a green grid, respectively, and their initial positions are $(0, 2)$ and $(2, 0)$, respectively. The target position of the leader robot is $(19, 19)$, which also determines the target position of the follower robots. The leader robot uses the improved Q-learning algorithm Q-L5 to plan the optimal path with the same parameter settings as in 5.1 simulation experiments. The two follower robots, respectively, use the tracking strategy based on the GPF to follow: the penalty function $R_{\text{static}} = 1$ and the adjustment coefficient $c = 0.09$.

The experimental effect is shown in Figure 8(a) that the three robots quickly reach the target with a fixed triangle formation in an obstacle-free environment. In an environment with static obstacles, the leader robot moves first, and the two follower robots immediately move to the corresponding positions in the formation. The green follower robot firstly encounters a black lateral obstacle. It moves along the obstacle to the target direction and smoothly avoids the obstacle. Then, it continues to accelerate to move to the corresponding position of the formation at the current time to maintain the formation. Finally, the leader robot plans a red path, the two follower robots avoid obstacles by themselves during the following process and plan a green path and a blue path, respectively. The three robots reach the target at the same time and complete the formation task. The experimental effect is shown in Figure 8(b).

6. Conclusion

In this paper, by combining the improved Q-learning algorithm and the idea of the GPF method, a method for multirobot formation and path planning is proposed. The division of labor among multiple robots is clear. The leader robot uses the improved Q-learning algorithm to plan the path. It is found that adding environment guidance of the target's GPF and virtual obstacle-filling avoidance strategy effectively accelerates iterative convergence and avoids concave obstacles. It is stable and efficient for the action selection strategy to be improved by the SA method. At the same time, the follower robot uses a tracking strategy based on the improved GPF to follow in real time, which is simple and efficient. This formation method effectively solves the formation and path planning problems of multiple robots in an unknown environment with concave obstacles. In the future, the multirobot formation will be further studied in the context of dynamic environments and privacy protection.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

The work is supported by the National Natural Science Foundation of China (61673200), the Major Basic Research Project of Natural Science Foundation of Shandong Province of China (ZR2018ZC0438), and the Key Research and Development Program of Yantai City of China (2019XDHZ085).

References

[1] K. K. Oh, M. C. Park, and H. S. Ahn, "A survey of multi-agent formation control," *Automatica*, vol. 53, pp. 424–440, 2015.

[2] A. Muxfeldt, D. Kubus, and F. M. Wahl, "Developing new application fields for industrial robots - four examples for academia-industry collaboration," in *2015 IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA)*, pp. 1–7, Luxembourg, Luxembourg, 2015.

[3] G. Dissanayake, J. Paxman, J. V. Miro, O. Thane, and H. Thi, "Robotics for urban search and rescue," in *First International Conference on Industrial and Information Systems*, pp. 294–298, Tirtayasa, Indonesia, 2006.

[4] M. Z. Rashid, F. Yakub, S. A. Zaki et al., "Comprehensive review on controller for leader-follower robotic system," *Indian Journal of Geo-Marine Sciences*, vol. 48, no. 7, pp. 985–1007, 2019.

[5] M. A. Lewis and K. H. Tan, "High precision formation control of mobile robots using virtual structures," *Autonomous Robots*, vol. 4, no. 4, pp. 387–403, 1997.

[6] P. Wang and Z. Geng, "Leader-follower formation control of multirobot systems using the dynamic surface approach," in *The 35th China Control Conference (CCC)*, pp. 7757–7762, Chengdu, China, 2016.

[7] W. Yin and X. Yang, "A totally Astar-based multi-path algorithm for the recognition of reasonable route sets in vehicle

navigation systems," *Procedia-Social and Behavioral Sciences*, vol. 96, pp. 1069–1078, 2013.

[8] C. Chen, X. Q. Chen, F. Ma, X. J. Zeng, and J. Wang, "A knowledge-free path planning approach for smart ships based on reinforcement learning," *Ocean Engineering*, vol. 189, article 106299, 2019.

[9] P. Sudhakara, V. Ganapathy, B. Priyadharshini, and K. Sundaran, "Obstacle avoidance and navigation planning of a wheeled mobile robot using amended artificial potential field method," *Procedia Computer Science*, vol. 133, pp. 998–1004, 2018.

[10] Jian Chen, Dong Sun, Jie Yang, and Haoyao Chen, "Leader-follower formation control of multiple non-holonomic mobile robots incorporating a receding-horizon scheme," *International Journal of Robotics Research*, vol. 29, no. 6, pp. 727–747, 2010.

[11] M. Sruthi, K. Rao, and V. Jisha, "Vector field based formation control of multirobot system," *IFAC-PapersOnLine*, vol. 49, no. 1, pp. 189–194, 2016.

[12] H. Sang, Y. You, X. Sun, Y. Zhou, and F. Liu, "The hybrid path planning algorithm based on improved A∗ and artificial potential field for unmanned surface vehicle formations," *Ocean Engineering*, vol. 223, no. 3–4, article 108709, 2021.

[13] P. Das and P. Jena, "Multirobot path planning using improved particle swarm optimization algorithm through novel evolutionary operators," *Applied Soft Computing*, vol. 92, article 106312, 2020.

[14] H. Qu, K. Xing, and T. Alexander, "An improved genetic algorithm with co-evolutionary strategy for global path planning of multiple mobile robots," *Neurocomputing*, vol. 120, no. 23, pp. 509–517, 2013.

[15] A. Lazarowska, "Discrete artificial potential field approach to mobile robot path planning," *IFAC-PapersOnLine*, vol. 52, no. 8, pp. 277–282, 2019.

[16] A. Maoudj and A. Hentout, "Optimal path planning approach based on Q-learning algorithm for mobile robots," *Applied Soft Computing*, vol. 97, no. 2020, article 106796, 2020.

[17] L. E. Soong, O. Pauline, and C. K. Chun, "Solving the optimal path planning of a mobile robot using improved Q-learning," *Robotics and Autonomous Systems*, vol. 115, pp. 143–161, 2019.

[18] X. Xu and J. Yuan, "Path planning for mobile robot based on improved reinforcement learning algorithm," *Journal of Chinese Inertial Technology*, vol. 27, no. 3, pp. 314–320, 2019.

[19] C. H. Oh, T. Nakashima, and H. Ishibuchi, "Initialization of Q-values by fuzzy rules for accelerating Q-learning," in *IEEE World Congress on IEEE International Joint Conference on Neural Networks*, pp. 2051–2056, Anchorage, AK, USA, 1998.

[20] C. Yan and X. Xiang, "A path planning algorithm for UAV based on improved Q-learning," in *2018 2nd International Conference on Robotics and Automation Sciences (ICRAS)*, pp. 1–5, Wuhan, China, 2018.

[21] C. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[22] https://link.springer.com/book/10.1007%2F978-981-16-6320-8.