IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 16, 2019, accepted February 18, 2020, date of publication March 3, 2020, date of current version March 18, 2020.

*Digital Object Identifier 10.1109/ACCESS.2020.2978077*

# The Experience-Memory Q-Learning Algorithm for Robot Path Planning in Unknown Environment

**MENG ZHAO[1], (Student Member, IEEE), HUI LU[1,2], (Senior Member, IEEE), SIYI YANG[1], AND FENGJUAN GUO[2]**

[1]School of Electronic and Information Engineering, Beihang University, Beijing 100191, China
[2]Shaanxi Key Laboratory of Integrated and Intelligent Navigation, Xi'an 710068, China

Corresponding author: Hui Lu (mluhui@vip.163.com)

This work was supported by the Shaanxi Key Laboratory of Integrated and Intelligent Navigation under Grant SKLIIN-20180204 and Grant SKLIIN-20190201.

**ABSTRACT** In order to solve the problem of slow convergence speed and long planned path when the robot plans a path in unknown environment by using Q-learning algorithm, we propose the Experience-Memory Q-Learning (EMQL) algorithm based on the continuous update of the shortest distance from the current state node to the start point. The autonomous learning ability of the robot is enhanced by the different role assignments of two tables in the proposed algorithm. EM table with $(m * 1)$ dimension is designed to record the distance information, reflecting the learning process of the robot. Q table is adopted as an auxiliary guidance for the experience transfer strategy and experience reuse strategy, and these strategies enable the robot accomplish the task even if the destination is changed or the path is blocked. Further, the learning efficiency of the robot in the EMQL algorithm is improved by the dual reward mechanism consisting of static reward and dynamic reward. The static reward is designed to prevent the robot from exploring a state node excessively. The dynamic reward is responsible for helping the robot avoid searching blindly in unknown environment. We test the effectiveness of the proposed algorithm on both grid maps and road network maps. The comparison results in planning time, iteration times and path length show that the performance of the EMQL algorithm is superior to Q-learning algorithm in convergence speed and optimization ability. Additionally, the practicability of the proposed algorithm is validated in a real-world experiment using the Turtlebot3 burger robot.

**INDEX TERMS** Path planning, Q-learning, experience memory, experience transfer, experience reuse.

## I. INTRODUCTION

In order to improve production efficiency, more and more robots have popularized its utilization in logistics warehousing, intelligent services, industrial production, rescue detection and other fields. With the expansion of robot application, the need to improve the autonomous navigation ability of robot in complex environment is becoming more and more important. Therefore, as one of the core technologies of autonomous navigation, path planning technology has become one of the hot topics in robotics.

Path planning technology can help the robot move from the start point to the target point without colliding with obstacles. According to the specified requirements, the robot needs to consider the improvement of performance indicators in the path planning process, such as planning time, path length, path smoothness, etc. The good path planning ability can ensure the robot to complete the designated work safely. If the robot lacks the adaptability to the surrounding environment in unknown environment, it will not only unable to complete the task, but also damage the working environment, resulting in unnecessary losses. Therefore, to realize the path planning in unknown environment by improving the autonomous planning ability of the robot has important realistic significance and practical value.

Numerous algorithms used for robot path planning in unknown environment have been proposed so far. Most of the literatures pay attention to the improvement of obstacle

The associate editor coordinating the review of this manuscript and approving it for publication was Vivek Kumar Sehgal.

avoidance ability for the robot, which are difficult to meet the practical application requirements when the robot needs to work in a completely unknown environment. To change this situation, recently, reinforcement learning algorithm has been applied to robot path planning. After reading the relevant articles, it can be found that to make the robot use reinforcement learning to plan a path in the real world, there are still many bottlenecks need to be broken through. First, due to the lack of prior knowledge, the robot needs a long learning time to complete the path planning task, which is difficult to meet the application requirements. Second, the robot needs to have path replanning ability to solve unexpected problems. Third, to improve the planning efficiency, the robot is expected to have an ability the transfer its experience between different tasks. Therefore, it is a challenging task to propose a robot path planning algorithm with high learning efficiency and suitable for dynamic scenarios.

In this paper, we propose the Experience-Memory Q-learning (EMQL) algorithm, which combines the Q-learning algorithm with experience memory mechanism. This algorithm can realize robot path planning in unknown environment, even when the environment or task changes during the movement of the robot. In addition, the experiment results show that the proposed algorithm is superior to Q-learning algorithm in convergence speed, optimization ability and dynamic adaptability.

Our contributions can be described in three aspects.

To begin with, we propose the dual reward, including static reward and dynamic reward after considering that the location of the target point is the only information that the robot can obtain when it plans a path in unknown environment. The static reward is related to the properties of the state nodes and the dynamic reward changes with the distance to the target point. The dual reward can help the robot avoid blind search and over exploration, which improves the learning efficiency of the robot.

Furthermore, in order to solve the problem of slow convergence speed when the robot plans by using Q-learning algorithm, we design the EM table as experience table in the EMQL algorithm. The EM table is used to record the shortest distance from different state nodes to the start point, which replaces Q table to reflect the learning progress. After the EM table converges, the robot can plan a safe and short path based on the obtained experience. If the robot explores $m$ state nodes and it has $n$ optional actions at each state node, the dimension of the EM table is $(m * 1)$, while the dimension of the Q table used in the Q-learning algorithm is $(m * n)$. Obviously, the dimension of the EM table is much smaller than that of Q table, which greatly reduces the learning time of the robot and meets the need of rapid path planning.

Finally, we propose two strategies to enable the robot to complete the assigned task successfully even when the environment or the task changes, namely experience transfer and experience reuse. In these strategies, Q table is assigned a new role, working as auxiliary guidance to assist the robot to complete the new task. If the robot detects the changes in destination or environment on its move, it will not only query the experience memorized in the EM table, but also take the action guidance in Q table into account. The design of these strategies improves the efficiency of path replanning and makes it possible for the the robot to transfer its experience between different tasks.

In the experiment part, three kinds of experiments are performed to test the applicability of the EMQL algorithm. They are the simulation experiments on grid maps, the simulation on the road network map and a practical path planning experiment of Turtlebot3 burger robot. The experiment results show that the EMQL algorithm is better than Q-learning algorithm in terms of the path optimization ability and path planning speed. Moreover, due to the use of the experience transfer strategy and experience reuse strategy, the robot can also complete the task successfully even when there are some changes in the environment.

The rest of this paper is organized as follows. Section II gives a summary of existing algorithms for robot path planning in unknown environment. Section III introduces Q-learning algorithm concisely, which is used for comparison in the experiment part. The proposed algorithm is described in detail in Section IV. Section V lays out the experiment results and a brief analysis. Section VI summarizes the entire article and discusses the implication of the findings to future research into this area.

## II. RELATED WORK

There are a wide variety of robot path planning algorithms. The algorithm proposed in this paper is based on reinforcement learning, but in order to gain a more clear understanding of the research status of robot path planning in unknown environment, we expand our investigation scope to the all widely used algorithms, instead of confining our attention to reinforcement learning. The robot path planning algorithms can be roughly classified into four types, which are respectively based on the fuzzy logic, intelligent optimization, SLAM and reinforcement learning.

The core of fuzzy logic algorithm is to construct a behavior rule database which can be used by the robot in different scenes. When the robot moves in unknown environment, it can plan a path by looking up the database according to the surrounding information obtained by sensors. Fuzzy logic makes the autonomous control of the robot possible, so it has been widely used in robot path planning since it was put forward. Yang *et al.* proposed a layered goal-oriented motion planning strategy using fuzzy logic in [1], which made the size of the rule-bases and the planning time reduced. In the explainable intelligence model proposed by Keneni *et al.*, the unmanned aerial vehicles can make decisions according to six rules when they are on the mission [2]. In [3], Nguyen *et al.* improved K-Bug algorithm by introducing the fuzzy logic for boundary following. In order to achieve 3D AUV path planning, Sun *et al.* designed a fuzzy system with accelerate/break module to enable the AUV avoid dynamic obstacles automatically [4]. In addition, some scholars

confirmed that the combination of fuzzy logic and genetic algorithm can improve the planning efficiency in terms of the distance travelled and the traveling time [5], [6].

Fuzzy logic algorithm can make the robot plan a safe path in unknown environment with partial prior knowledge, and it has been put into use in the factory. However, the design of the algorithm usually involves complex control theory. Moreover, fuzzy rules usually depend on artificial experience and are related to the application scenarios of the robot. Therefore, the generalization ability of fuzzy logic algorithm is not good.

The intelligent optimization algorithm includes genetic algorithm, immune algorithm, ant colony algorithm, particle swarm optimization and other algorithms inspired by natural phenomena or biological groups. This type of algorithm can endow the robot with powerful optimization ability through many iterations, so there are lots of research results in robot path planning. In the algorithm proposed by Garcia *et al.*, they introduced ant colony algorithm to simulate the robot navigation and selected the optimal path by using the criterion of a fuzzy rule [7]. Wang *et al.* used particle swarm algorithm to solve the problem of path planning for the UAV swarms and got the remarkable results [8]. Tong *et al.* proposed an algorithm that combined the particle swarm algorithm with genetic algorithm in [9], which can be used for the welding robot path optimization. In addition, simulated annealing algorithm [10], artificial immune network algorithm [11] and tabu algorithm [12] are also be applied in robot path planning.

However, the process of iterative computation consumes a lot of time while helping the robot improve the ability of path optimization. Besides, there are usually many parameters in the intelligent optimization algorithm. The choice of parameters directly determines the performance of the algorithm. Therefore, this type of algorithm is usually used for the theoretical research on the path planning rather than the practical application.

Simultaneous Localization and Mapping (SLAM) was proposed by Smith and Cheeseman in 1986 [13]. The core of SLAM is to deduce the observation model and motion model of the robot by the acquired external data, and then the real-time estimation of the motion state for the robot can be achieved. Since the design of this algorithm takes into account the feedback information of sensors, the SLAM algorithm is commonly used in practical applications. In order to improve the accuracy of the estimation model, many scholars improved the SLAM algorithm. Dissanayake *et al.* proved that a solution to the SLAM problem is possible and elucidated the underlying structure of the SLAM problem in [14]. In [15], Blosch *et al.* proposed an approach to enable a micro aerial vehicle to navigate in unknown environment based on the visual SLAM algorithm of Klein *et al.* [16]. To improve the positioning accuracy of the robot when exploring the environment, Maurović *et al.* put forward a path planning algorithm for active SLAM in [17]. In addition, the robot can plan efficiently in the complex 3D environments while representing the 3D environment by using the algorithm presented by Yang *et al.* [18]. In a recent paper of Lee *et al.*, they constructed the estimation models using the direction of the vanishing point, by which the robot position and the line landmark can be derived as simple linear equations [19].

When the robot plans a path by using SLAM in unknown environment, the difference matching of point clouds is the basis for the robot to achieve localization and mapping. However, the multiple matching calculations consume a lot of time while improving the positioning accuracy, causing the low planning efficiency. Therefore, when the robot do not need to build the environmental maps in real time during the planning process, the application of SLAM seems to waste its talent.

The reinforcement learning algorithm based on trial-and-error mechanism has attracted much attention in recent years because no prior data is needed to plan an optimal or sub-optimal path. The core of this algorithm is to obtain the maximum cumulative reward in the environment. If the execution of an action can increase the reward of the robot, the probability of choosing this action will be increased in the following learning process. Reinforcement learning includes Q-learning, Sarsa, deep Q network, deep deterministic policy gradient, and so forth. In the past few years, they have been used in robot path planning [20]–[24]. Q-learning is one of the most classical reinforcement learning algorithms. In recent years, many scholars have made improvements to expand its applications. In order to improve the efficiency of path planning and reduce unnecessary calculation in Q table, EQL and IQL algorithms were put forward in succession by Konar *et al.* in 2010 [25] and 2013 [26]. To address the limitation of slow convergence in the Q-learning, Soong *et al.* introduced the flower pollination algorithm to improve the initialization of the Q table [27]. The convergence speed can also be improved by the exploration region expansion strategy proposed by Gao *et al.* [28]. In addition, Q-learning algorithm can be used for path planning of multi-robot system in combination with optimization algorithms such as particle swarm optimization [29] and differential evolution [30].

Although repeated learning can make the robot plan a better path, the problem of low learning efficiency caused by which makes reinforcement learning difficult to be applied in practical applications. Thus, increasing the convergence speed of the algorithm will help to promote the development of reinforcement learning. In addition, almost all robot path planning algorithms based on the reinforcement learning aim to improve the obstacle recognition ability of the robot. Scholars ignore that the connectivity between state nodes can also be regarded as experience. Therefore, it is novel and meaningful to propose a path planning algorithm that can be applied for robots without vision module in unknown environment.

In summary, numerous algorithms can be used for robot path planning in unknown environment and each has its own merits (see Table 1). However, there are still several bottlenecks in the study of path planning. First, when the robot needs to work in a completely unknown environment, it is difficult to guarantee the safety of the robot only by establishing

**TABLE 1.** The features of typical algorithms for robot path planning in unknown environment.

| Algorithm | Examples | Features |
|---|---|---|
| Fuzzy Logic | [1],[3],[13] | Strengths: Simple and easy to understand<br>Weakness: Generalization ability is not good |
| Intelligence Optimization | [15],[18],[19] | Strengths: Searching ability is strong<br>Weakness: Calculation time is long and many parameters need to be set |
| SLAM | [22],[24],[26] | Strengths: Simultaneous localization and mapping<br>Weakness: The amount of calculation is large |
| Reinforcement Learning | [6],[27],[35] | Strengths: No prior knowledge of any environment is required<br>Weakness: Convergence speed is low |

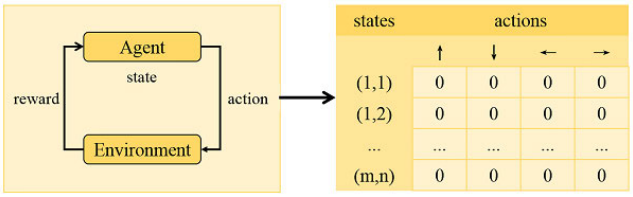


**FIGURE 1.** The model of Q-learning and the structure of Q table.

a set of general behavior rules. Second, the lack of prior knowledge results in a great deal of redundant calculation in two adjacent searches of the robot. Third, most algorithms converge slowly, which affects the efficiency of path planning. Finally, few algorithms are available for dynamic path planning, especially when the robot has no vision module. These problems limit the further advancement of the study on path planning. Consequently, it is essential to propose a path planning algorithm that can be used in unknown environment with strong generalization ability, fast convergence speed and minor calculation. In this paper, inspired by the existing algorithms, we propose the EMQL algorithm.

## III. THE Q-LEARNING ALGORITHM

Q-learning algorithm (QL) is proposed by Watkins in 1992 [31], which is one of the most classical methods of reinforcement learning. Enlightened by behaviorism psychology, no prior knowledge is required in this algorithm, so Q-learning has been widely used since it was proposed. The agent gains the environment information according to the reward obtained by performing different actions. After many iterations, the agent can get a convergent Q table which is used to guide it on how to obtain the maximum cumulative reward. The model of Q-learning and the structure of Q table are shown in Figure 1, where there are $m$ states and $n$ actions in each state and 0 represents the initial value.

There are four important elements in Q-learning algorithm: state ($s$), action ($a$), reward ($R$), Q table ($Q$). The definitions of these elements are related to the application background of Q-learning algorithm. When it is used for robot path planning, the state node of the environment is usually defined as the coordinate, and the action is the direction in which the robot can move. In addition, in many studies, the design of reward is related to the properties of the state nodes the robot reaches. If the robot moves to a dangerous place, such as the positions around the obstacles, it will receive a negative reward immediately. On the contrary, a positive reward will be gained if the robot arrives at the destination.

As the core of Q-learning algorithm, the rows in the Q table represent the state nodes of the environment, and the columns of the corresponding rows stand for the optional actions of the agent at that state. In order to calculate the Q table, Watkins used Bellman Equation to simulate the learning process of the agent in Q-learning algorithm, as shown in (1).

$$Q(s_{t+1}, a_{t+1}) = (1-\alpha) * Q(s_t, a_t) + \alpha * [R(s_t, a_t) + \gamma * \max_a(Q(s_{t+1}, a))] \quad (1)$$

Here, $\alpha$ and $\gamma$ are the learning rate and discount factor of the agent respectively. If the agent performs action $a_t$ in state $s_t$ at time $t$, it will get immediate reward $R(s_t, a_t)$ and delayed reward $\max_a(Q(s_{t+1}, a))$. Therefore, the design of reward is significant in Q-learning algorithm, which affects the action selection of the agent. According to the Bellman Equation, the agent will eventually obtain a convergent Q table and accomplish the task in the environment by searching for the maximum value of each state in the Q table. The pseudo code of Q-learning is shown as follows.

**Algorithm 1** Q-Learning Algorithm

```
Initial α, γ and Q table
if episode <= max episode then
    Initial s_t
    while s_t is not terminal do
        choose a_t using policy derived from Q table
        execute action a_t, observe R(s_t, a_t) and s_{t+1}
        update Q(s, a) by using Bellman Equation
        s_t = s_{t+1}
    end while
end if
```

## IV. THE EMQL ALGORITHM

In Q-learning algorithm, the robot updates $Q(s, a)$ by using Bellman Equation according to the reward obtained at each

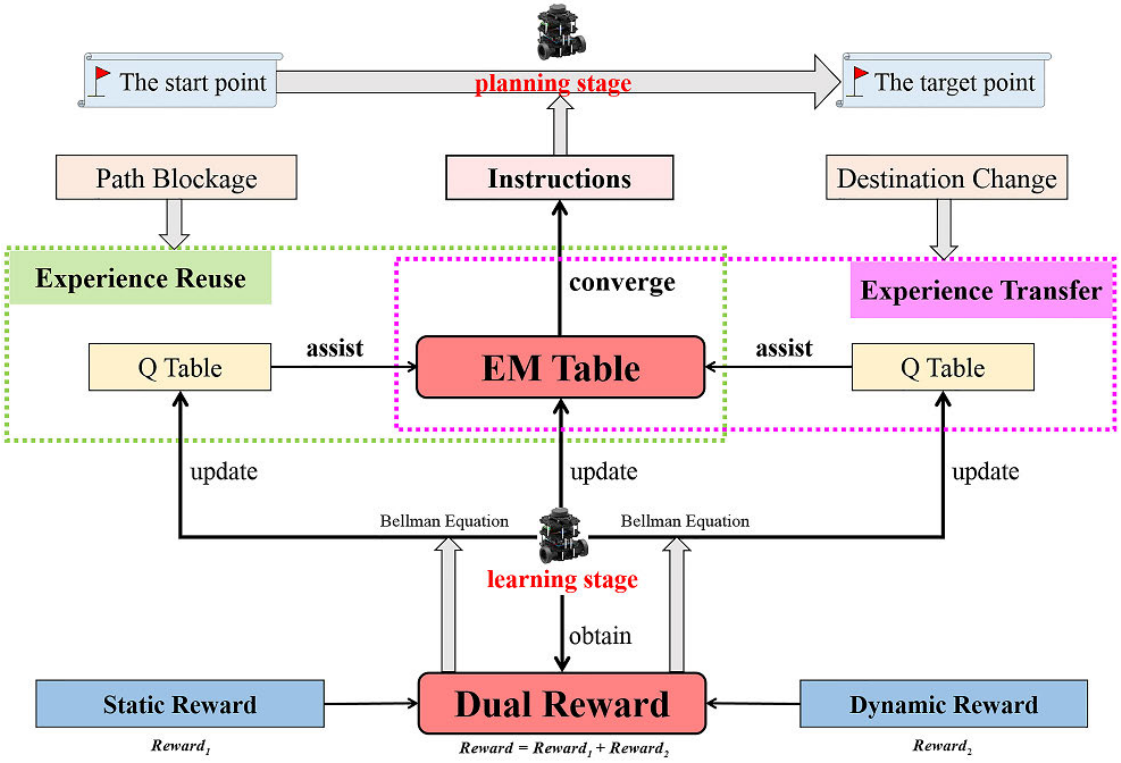


**FIGURE 2.** The model of the EMQL algorithm.

state node. The convergence of Q table means that the robot has the ability to plan the shortest path on the basis of its experience. This self-learning mode makes Q-learning algorithm widely used in the research on autonomous mobile of unmanned vehicle. However, the dimension of Q table changes with the complexity of working environment and the number of optional actions. When the working environment is complex, the convergence of Q table will be a long process. In addition, the improper reward function will also lead to inefficient path planning.

In order to make the robot efficiently complete the planning task in unknown environment,we propose the EMQL algorithm. This algorithm combines the experience memory mechanism and the dual reward method based on the Q-learning algorithm. The design of the reward function and the EM table are the shining points of this paper, which enables the EMQL algorithm performs better than Q-learning algorithm. In addition, the use of the experience transfer strategy and experience reuse strategy can improve the dynamic planning ability of the robot. Since the proposed algorithm is inspired by the pathfinding strategy of human, the robot can complete the path planning task faster and better by using the EMQL algorithm than using Q-learning algorithm.

In this section, we will introduce the proposed algorithm from five aspects. The framework of the EMQL algorithm is shown in the first part. The second part describes the dual reward method from two aspects: static reward and dynamic reward. In the third part, the update process for the EM table will be presented, which elaborates how a robot uses the accumulated experience to accomplish the task in unknown environment. Furthermore, the experience transfer strategy and the experience reuse strategy are introduced separately in the last two parts.

### A. ALGORITHM FRAMEWORK

When the robot plans a path in unknown environment by using the EMQL algorithm, the robot needs to go through two stages: learning stage and planning stage. In the learning stage, the robot has to move from the start point to the end point many times to collect the shortest path experience. During the path search process, the shortest distance will be recorded in EM table and the state nodes on the shortest path will be stored in Instructions. If the shortest path length in the EM table is invariable no matter how the robot moves, it means that the robot has got enough information of the environment and can safely move to the destination by using the experience it memorizes. In the planning stage, the robot only needs to move with the guidance of the Instructions step by step until it reaches the end point. In addition, the experience transfer strategy and the experience reuse strategy in the EMQL algorithm will solve the dynamic problems the robot encounters in the learning stage and planning stage respectively. Figure 2 shows the model of the EMQL algorithm.

In the learning stage, the robot will get static reward and dynamic reward. The static reward is related to the property of the state node the robot reaches, which is essential

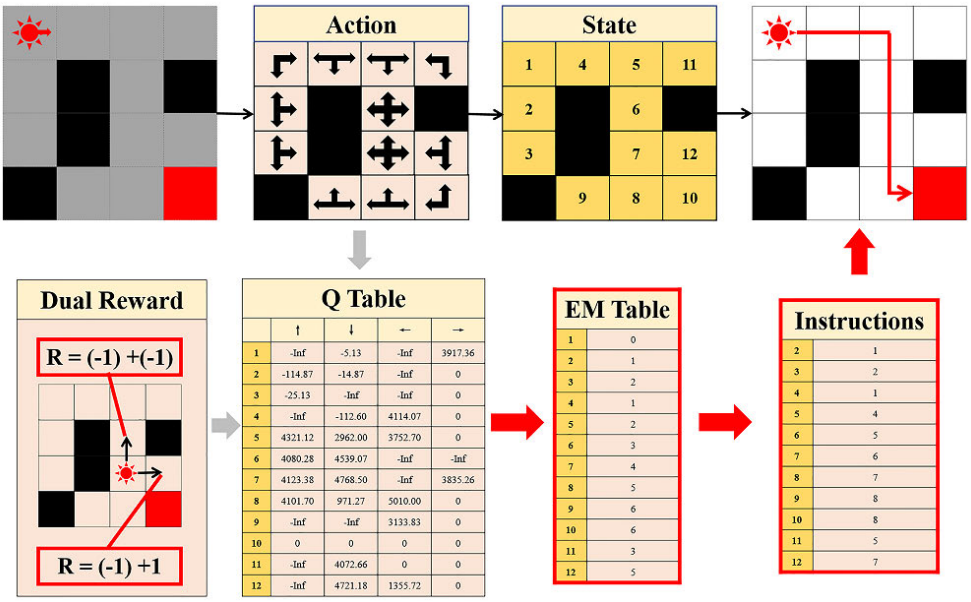


**FIGURE 3.** Flow chart for robot path planning by using the EMQL algorithm in a 4*4 grid map.

to prevent the robot from failing into local optimum. The dynamic reward associated with the distance from the current position to the end point provides convenience for the robot to reach the end point quickly. The sum of the reward is used to update $Q(s, a)$ which guides the robot to choose the action at each state node, and the experiment results prove that the combination of these two reward can help the robot learn a shorter path.

In the EMQL algorithm, the robot not only needs to update the Q table as it moves, but also is required to update the EM table which records the shortest path length from each state node to the start point. If the robot finds a shorter path when it searches in the environment, the data recorded in the EM table will be updated immediately. At the same time, the state nodes on the shortest path will be recorded in Instructions. After the Q table converges, the robot completes the learning stage. If the robot searches $m$ state nodes totally in the learning stage and it has $n$ optional actions at each state node, the dimension of EM table is $(m * 1)$, while the Q table used in the Q-learning algorithm is $(m * n)$. Therefore, it can be inferred that Q table will not converge when the EM table converges. Therefore, adopting the EM table to reflect the learning progress of the robot will save a lot of learning time.

Because the robot has obtained the shortest path based on its experience in the learning stage, it only needs to move along the state nodes recorded in Instructions in the planning stage. In addition to improving the efficiency of path planning, the experience transfer strategy and experience reuse strategy for solving the dynamic problems the robot encounters are also the highlights of EMQL algorithm. In these strategies, Q table is adopted as the auxiliary experience to assist the robot to plan the path.

The robot usually encounters two kinds of dynamic scenes when it plans a path in unknown environment: the destination is changed or the path is blocked. Therefore, in order to improve the applicability of the EMQL algorithm, we propose corresponding strategies for these two dynamic scenes. When the robot is asked to change the destination, it will adopt the experience transfer strategy. Under the guidance of this strategy, the robot will try to apply the experience acquired in the original task to the new planning task. In addition, if the robot finds that the path ahead is blocked when it moves along the planned path, the experience reuse strategy will be chose. The robot will try to find a detour in Instructions, and a short-distance learning will be executed if there is no detour can be found.

In Figure 3, we take a 4*4 grid map as an example to show the process of path planning in unknown environment when the robot plans a path by using the EMQL algorithm.

In the grid map shown in Figure 3, we assume that the size of the map is 4m*4m and the robot needs to move southeast to the position which is 4.25m away from the start point. In the simulation environment, the robot can move forward, backward, left and right, but it cannot collide with obstacles or the edge of the environment. To facilitate recording, the state nodes are numbered according to the order of robot exploration. Because the action with large value in Q table will be executed by the robot, there are several values which correspond to the immovable directions are defined as $-inf$ in Q table. In addition, it can be seen that the shortest moving distance from each state node to the state node 1 is recorded in the EM table. The table named Instructions on the right side of the EM table stores the waypoints of the shortest path, which is used to guide the robot to finish the planning task.

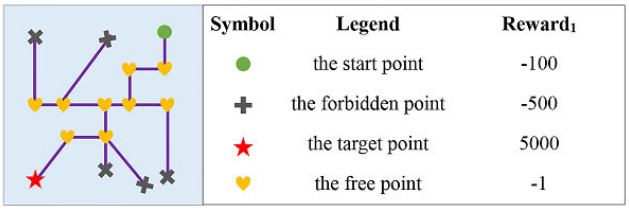


**FIGURE 4.** The static reward function designed in the EMQL algorithm.

### B. DUAL REWARD

The reward function is a timely evaluation of the action performed by the robot. A well-designed reward function will help improve the learning efficiency of the robot in unknown environment. Previous studies in reinforcement learning suggested that we can set the default reward to a small negative value when the robot needs to get a short path. In this way, the average reward in free space will be small, and there is a certain numerical distance between the reward the robot obtains at the target point and near the obstacles. Based on this theoretical basis, we design a static reward which is related to the property of the state node. Apart from this common reward function, we devise a dynamic reward in the EMQL algorithm. The dynamic reward changes with the Euclidean distance between the state node the robot reaches and the end point. The combination of these two kinds of reward prevents the robot from blind search and over exploration in the environment. The total reward that the robot gets as it moves is shown in (2), where $Reward_1$ and $Reward_2$ represent static reward and dynamic reward respectively.

$$Reward = Reward_1 + Reward_2 \tag{2}$$

#### 1) THE STATIC REWARD FUNCTION

The state nodes in the environment can be divided into four categories: the start point, the forbidden point, the target point and the free point. These types of nodes are shown with different logos in Figure 4, the purple line represents the free space, and the blue area is the obstacle space. Based on this classification, the robot will get different reward at different state nodes, as shown in (3).

$$Reward_1 = \begin{cases} -100, & \text{if } s_{t+1} \text{ is the start point} \\ -500, & \text{if } s_{t+1} \text{ is the forbidden point} \\ 5000, & \text{if } s_{t+1} \text{ is the target point} \\ -1, & \text{if } s_{t+1} \text{ is the free point} \end{cases} \tag{3}$$

As shown in Figure 4, the forbidden point is defined as the state node with only one reachable direction. The robot is bound to turn back if it finds that the state node it arrives at is not the target point and there is no direction to move on, which will consume plenty of learning time. Therefore, in order to improve the planning efficiency, it is necessary to give the robot punishment when it arrives at such state nodes.

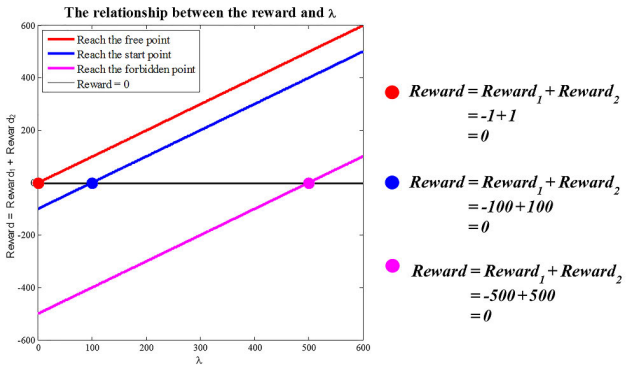


**FIGURE 5.** The relationship between the total reward obtained by the robot and λ.

#### 2) THE DYNAMIC REWARD FUNCTION

When the robot works in unknown environment, it needs to collect as much environment information as possible in a shorter time to plan the path quickly. Therefore, in the learning stage, avoiding blind search is essential for the robot to improve the planning efficiency. In the EMQL algorithm, we design the dynamic reward referring to human path planning strategy. Based on this reward, the robot can obtain well-directed environment information and be familiar with the working environment quickly. The dynamic reward is related to the change of the distance between the state nodes the robot reaches and the target point at two moments, as shown in (4) to (6):

$$d_t = \sqrt{(x_{goal} - x_t)^2 + (y_{goal} - y_t)^2} \tag{4}$$

$$d_{t+1} = \sqrt{(x_{goal} - x_{t+1})^2 + (y_{goal} - y_{t+1})^2} \tag{5}$$

$$Reward_2 = \lambda * \frac{d_t - d_{t+1}}{|d_t - d_{t+1}|} \tag{6}$$

Here, $(x_t, y_t)$ and $(x_{t+1}, y_{t+1})$ are the coordinates of the state nodes the robot reaches at $t$ and $t+1$ respectively, and $(x_{goal}, y_{goal})$ is the target point. λ is a parameter related to the static reward function, which is used to adjust the degree of the robot moving towards the end point. The value of λ will affect the proportion of static reward in the total reward the robot obtains.

According to the definition of the dynamic reward, the robot will get a negative reward if it moves away from the end point. The total reward it obtains in this case is still a negative value, which will not change the result that the robot is punished. However, when the robot moves closer to the destination, it will receive a positive reward. At this time, the sum of the static reward and the dynamic reward depends on the value of λ. If λ is too large, the impact of the static reward that the robot gets at some state nodes will be covered. If λ is too small, the willingness to move to the end point of the robot is not strong, which will make the design of dynamic reward meaningless. Therefore, according to the definition of the static reward, the value of λ can be divided into three intervals: [1, 100); [100, 500); [500, ∞). As shown in Figure 5, the red, blue and magenta points on the curve

**EM table**

| states | The shortest distance to the start point |
|---|---|
| (1,1) | 1 |
| (3,2) | 5 |
| ... | ... |
| (m,n) | 10 |

**Instructions**

| states | The best way to move |
|---|---|
| (1,1) | (2,1) |
| (3,2) | (2,2) |
| ... | ... |
| (m,n) | (m-1,n) |

**FIGURE 6.** The structures of the EM table and Instructions.

represent that when the $\lambda$ is 1, 100 and 500, the reward the robot gains will be 0 when it reaches the free point, the start point and the forbidden point.

### C. EM TABLE AND INTRODUCTIONS

The design of the EM table is the core of the EMQL algorithm and the information in Instructions guides the robot move safely to the end point. Therefore, in this part, we will give the definitions of the EM table and Instructions, and the update process for the EM table will be shown in the end.

There are two tables designed in the EMQL algorithm. One is the experience table for storing the shortest distance between each state node to the start point, which is named as the EM table. Another table is Instructions, which stores the waypoints corresponding to the shortest distances recorded in the EM table. The waypoints stored in Instructions are updated according to the content of the EM table. Therefore, the convergence of the EM table means the end of the learning stage for the robot, and the robot can plan a path according to the connection relationships of state nodes recorded in Instructions. The structures of these tables are shown in Figure 6.

In the EMQL algorithm, the shortest distance recorded in the EM table will be continuously updated as the robot searches the path in unknown environment. Ideally, if the data in the EM table remains the same no matter how the robot changes its action, then a short and safe path can be planned by the robot. Hence the convergence of the EM table is the sign of completing the learning stage in the proposed algorithm. However, in terms of the simulation, due to the different criteria for the convergence of the EM table, the robot will not necessarily find the global shortest path. Therefore, the shortest path in this paper is defined on the basis of the experience the robot has.

The update process of the EM table in the EMQL algorithm is shown in Algorithm 2.

According to the update process of the EM table, the design of the EM table has two advantages. First, if the robot explores $m$ state nodes in the environment, no matter how many feasible actions at each state node, the dimension of the EM table is $m * 1$. However, the dimension of the Q table used to judge the completion of learning stage in Q-learning algorithm is $m * n$. In terms of the dimension of the experience list, the convergence time of EM table is much shorter than that of Q table. Therefore, the design of the EM table ameliorates the problem of slow convergence speed

**Algorithm 2** Update of the EM Table and Instructions

```
Initial Q table=[];EM table=[];Instructions=[]
if episode <= max episode then
    Initial s_t =the start point
    while s_t is not terminal do
        if s_t is not in the EM table then
            add s_t and d(s_t, S) into the EM table
        else
            if d(s_t, S) is smaller than before then
                update the EM table and Instructions
            end if
        end if
        add reachable state nodes into Q table
        choose a_t using policy derived from Q table
        execute action a_t, observe R(s_t, a_t) and s_{t+1}
        update Q(s, a) by using Bellman Equation
        s_t = s_{t+1}
        if s_{t+1} is not in Instructions then
            add s_{t+1} and s_t into Instructions
        end if
    end while
end if
```

caused by the large dimension of the Q table in Q-learning algorithm. Second, if the robot finds a path which can help it reach another state node faster, the data recorded in the EM table and Instructions will be updated immediately. However, $Q(s, a)$ needs to be calculated by Bellman Equation many times before the best action at a state node is changed, resulting in the delay of updating the optimal action. Therefore, the EM table makes the path optimization ability of the robot stronger in unknown environment, which is verified in the experiment part of this paper.

### D. EXPERIENCE TRANSFER STRATEGY

In our daily life, we often need to change the destination in the middle of driving because of the change in the itinerary. Timely adjustment of route can not only save time, but also save energy consumption of vehicles. However, in unknown environment, the lack of map information makes it difficult for us to achieve rapid path replanning. We can only grope our way forward in the experiment based on existing experience. According to this life experience, we design the experience transfer strategy in the EMQL algorithm. This strategy can help the robot complete the task as usual even when the destination is changed in the learning stage, which improves the autonomous learning ability of robot.

Although the data recorded in Q table is not used to guide the robot to plan the final path, it is adopted as a reference for the action selection in the learning process. Hence to make full use of the storage space, the Q table is used as the auxiliary experience to assist the robot to complete the new planning task quickly when the destination is changed. The process of the experience transfer strategy used in the EMQL algorithm is shown in Figure 7.

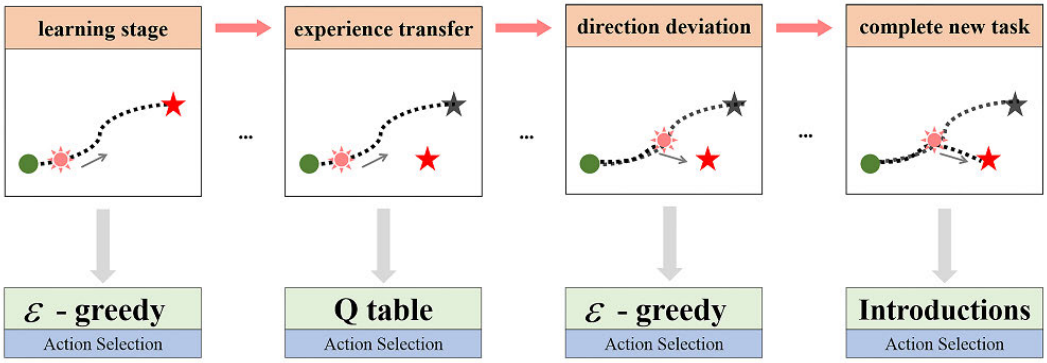


**FIGURE 7.** The experience transfer strategy used by the robot in the learning stage when the destination of the planning task is changed.

**Algorithm 3** Experience Transfer Strategy in EMQL

```
Initial count = 0
if the target point is changed then
    if count < 5 then
        choose a_t according to the Q table
        execute action a_t, observe s_{t+1} and d(s_{t+1}, T)
        if d(s_{t+1}, T)>d(s_t, T) then
            count = count + 1
        else
            count = 0
        end if
    else
        choose a_t using policy derived from Q table
        execute action a_t, observe R(s_t, a_t) and s_{t+1}
    end if
end if
```

Since the experience in the EM table is collected in the original task, the robot cannot use it all for the task with new destination. However, the Q table can be used in the new task because there is reference information of each action in the Q table. Therefore, we regard it as an assisted experience list of the robot. As shown in Figure 7, when the robot detects the change of the target point in the learning stage, it will abandon the $\epsilon$-greedy strategy and choose the action corresponding to the maximum value of the state node in the Q table. If the deviation of the moving direction is detected five times in a row, the $\epsilon$-greedy strategy will be used again to select the action until the completion of the new task. Because part of the experience gained in the old task is transferred to the new task, the relearning time is decreased and the planning efficiency in the new task is improved.

The pseudo code of the experience transfer strategy in the EMQL algorithm is shown in Algorithm 3.

### E. EXPERIENCE REUSE STRATEGY

In addition to planning a safe path, the robot also needs to have the ability of relearning to ensure that it can adjust its path according to the changes of the environment. Therefore, in order to help the robot search a detour as soon as possible while having the ability to replanning, we propose the experience reuse strategy in the EMQL algorithm. The whole process of the experience reuse is shown in Figure 8.

If the robot detects that part of the path is blocked when it moves along the planned path, it has two methods to bypass the blocked path. First, to maximize the use of existing experience and avoid spending a lot of time relearning, the robot will try to find a detour in Instructions. The start point and the end point of the detour are the current state node, and the first feasible state node in the remaining planned path respectively. If there is a detour in the Instructions, the robot will first follow the detour and then continue moving along the remaining planned path. On the contrary, if no detour can be found, the robot will perform a short-distance relearning and plan a new path to bypass the terrible path. In addition, there is a post processing procedure after relearning, which aims to avoid the extra energy consumption caused by the overlap between the original planned path and the new path.

The pseudo code of the experience reuse strategy in the EMQL algorithm is shown in Algorithm 4.

**Algorithm 4** Experience Reuse Strategy in EMQL

```
if s_{t+1} is blocked then
    find the detour in the Instructions
    if no detour can be found in the Instructions then
        relearn
        post processing
    end if
    move along the detour
else
    move along the (remaining) planned path
end if
```

## V. EXPERIMENT AND ANALYSIS

We design simulation experiments and application experiments of robot path planning to test the performance of the EMQL algorithm. In simulation experiments, we validate the algorithm on MATLAB based on the grid maps and road network maps respectively. Since grid maps are the most

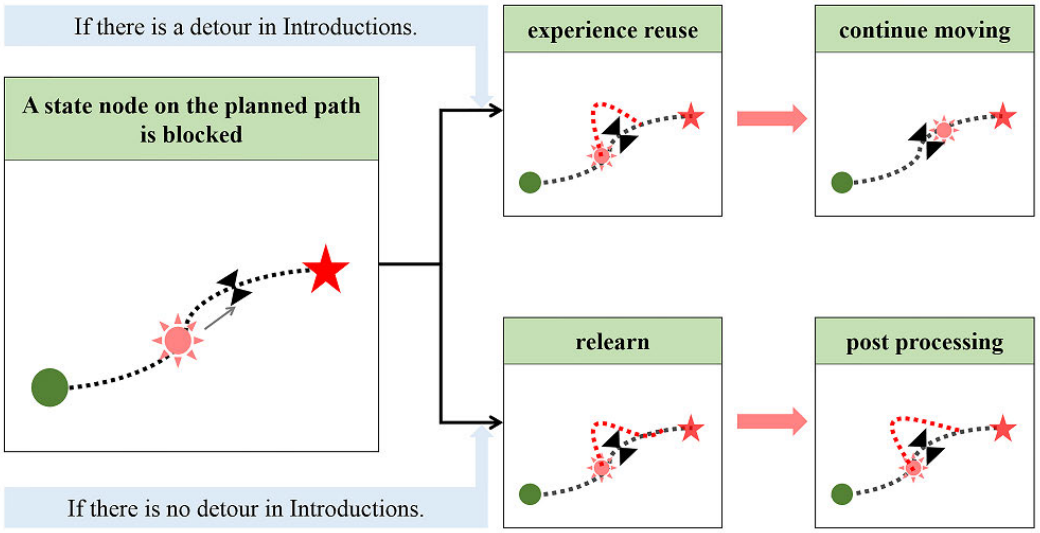


**FIGURE 8.** The experience reuse strategy used by the robot in the planning stage when the planned path is blocked.

commonly used model in robot path planning, we conduct experiments on the grid maps obtained by sampling the city maps to ensure the feasibility of EMQL. Here, we only show the planning results and compare them with Dijkstra algorithm in terms of the path length without analyzing them in depth. However, considering that the road network maps have more practicality than the grid maps, we analyze the experiment results on the road network maps from various aspects, and compare them with Q-learning algorithm to clarify the superiority of the EMQL algorithm. In addition, we design two dynamic path planning scenes and perform some experiments to further verify the adaptability of the algorithm. Finally, Turtlebot3 burger robot completes the path planning task in the indoor environment by using the proposed algorithm and some snapshots are shown in the text.

### A. SIMULATION BASED ON THE GRID MAP

In order to verify the effectiveness of the EMQL algorithm, we conduct the experiments on the grid maps obtained from the Real-World Benchmarks proposed by N. Sturtevant [32]. Figure 9 shows the paths planned by the robot in the simulation environment, where the size of the robot is assumed as one pixel. In addition, we compare the experiment results with Dijkstra algorithm in terms of path length, and the comparison results are shown in Table 2.

It can be seen obviously that the robot can accomplish the planning task in unknown environment by using the EMQL algorithm, and the length of the planned path is the shortest in most cases. As the data shown in Table 2, when the obstacles on the line between the start point and the end point are relatively dense, the path planned by the robot is not the shortest. The reason for this result is that the robot is asked to move towards the end point under the guidance of dual reward mechanism in the EMQL algorithm. If there are many obstacles in the direction of the end point, the robot will temporarily fall into the local optimum around the obstacles, which affects the length of the final planned path.

### B. SIMULATION BASED ON THE ROAD NETWORK MAP

In our daily life, some coordinates of special positions can represent a map of a region. Therefore, compared with grid maps, road network maps with intersections as state nodes have more reference value in the study of path planning. In this part, to verify the practicability of the EMQL algorithm, we regard the robot as a particle to perform the experiments on the road network map.

The experiment data is derived from Beijing 2008 road network GIS data set [33]. The map is shown in Figure 10, the scale of which is 1:100000. It is remarkable that the two lines intersected in the map are not necessarily connected in real life, because the road network map is 2D and these roads may be projections of overpasses on the ground. In addition, because the robot has sensor feedback information in actual operation, it can see where it can reach at each intersection. Considering that the robot cannot get any feedback information of sensors in the simulation environment, we assume that the robot can get the corresponding road connectivity information at any intersection in unknown environment.

To test the performance of the proposed algorithm, we design experiments based on the road network map from two aspects: static path planning and dynamic path planning. In the experiments of static path planning, we compare the performance of the EMQL algorithm with Q-learning algorithm in terms of the convergence speed and optimization ability. In the dynamic path planning experiment, two dynamic scenes including destination change and road blockage are designed for the robot to test the dynamic adaptability of the EMQL algorithm.

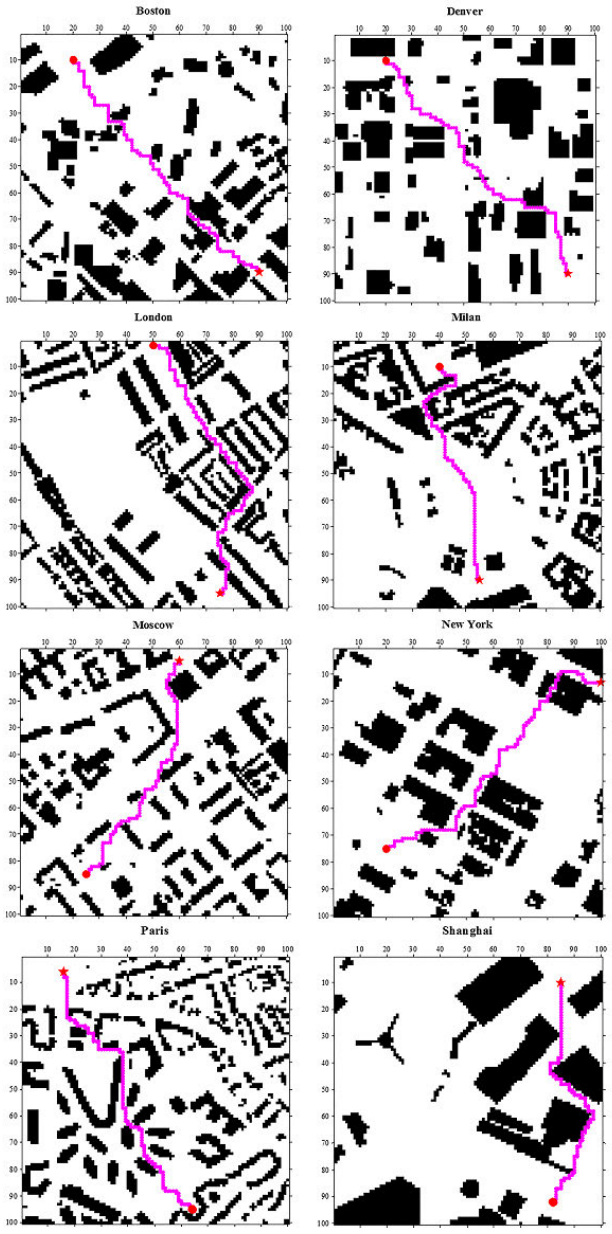


**FIGURE 9.** The planned paths on the grid maps when the robot plans by using the EMQL algorithm. The red dot, the red pentagram and the magenta line in the figure represent the start point, the target point and the planned path respectively.

### 1) STATIC PATH PLANNING

We set the coordinate of the start point to (948210.71, 4332801.27) and randomly select 10 points distributed at different places of the map as the target point. Because of the randomness of the action selection in the learning stage, the paths planned by the robot may be different in each experiment. In order to ensure the reliability of the experiment results as far as possible, we repeat the experiment of the same path planning task 50 times. Then we calculate the average values of planning time, iteration times and path length,

**TABLE 2.** The results of the planned path based on the grid maps.

| City | Path Length ($L_{EMQL}$) | Path Length ($L_{Dijkstra}$) | $\frac{L_{Dijkstra}}{L_{EMQL}}$ |
|---|---|---|---|
| Boston | 150 | 150 | 100% |
| Denver | 150 | 150 | 100% |
| London | 150 | 138 | 92% |
| Milan | 119 | 119 | 100% |
| Moscow | 123 | 123 | 100% |
| NewYork | 150 | 150 | 100% |
| Paris | 137 | 137 | 100% |
| Shanghai | 117 | 117 | 100% |

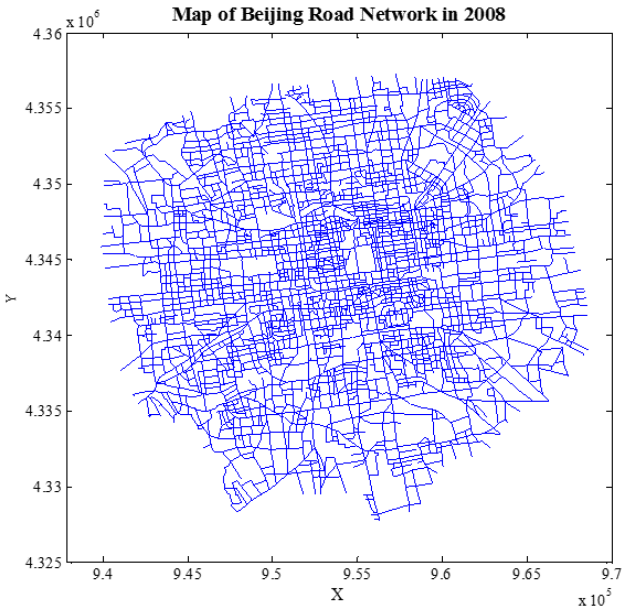


**FIGURE 10.** The map of Beijing road network in 2008, where the coordinate system used is GCS Krasovsky.

and the comparison results are shown in Table 3, Table 4 and Table 5, respectively.

It can be seen from the date in the tables that the path planning efficiency of the robot in the EMQL algorithm is significantly higher than that in Q-learning algorithm. Besides, the length of the path searched by the robot is shorter when using the proposed algorithm. These results indicate that the optimization ability of the EMQL algorithm is better than that of Q-learning algorithm, and the learning efficiency can be guaranteed when the robot plans the path by using the proposed algorithm. In the rest of this part, we will take the experiment with the target point of (958508.87, 4355976.17) as an example, and prove that the use of dual reward and experience memory makes the EMQL algorithm perform better than Q-learning algorithm.

Figure 11 shows the paths planned by using Q-learning algorithm and the EMQL algorithm. The red dot, the red

**TABLE 3.** The comparison of planning time when planning by using Q-learning algorithm and the EMQL algorithm.

| No. | The Target Point | Planning Time (QL) | Planning Time (EMQL) | Comparison |
|---|---|---|---|---|
| 1 | (958508.87,4355976.17) | 706.23s | 516.49s | ↓26.87% |
| 2 | (949806.10,4354281.25) | 610.51s | 397.65s | ↓34.87% |
| 3 | (956821.20,4350914.58) | 629.32s | 358.64s | ↓43.01% |
| 4 | (953423.30,4355219.39) | 638.68s | 438.27s | ↓31.38% |
| 5 | (966349.16,4347350.72) | 552.96s | 495.20s | ↓10.45% |
| 6 | (943744.00,4353509.70) | 614.90s | 244.56s | ↓60.23% |
| 7 | (942330.23,4341038.01) | 331.22s | 83.14s | ↓74.90% |
| 8 | (941827.77,4350828.07) | 535.94s | 190.53s | ↓64.45% |
| 9 | (941146.23,4346533.47) | 519.88s | 185.13s | ↓64.39% |
| 10 | (958677.74,4328891.17) | 207.41s | 25.01s | ↓87.94% |

**TABLE 4.** The comparison of iteration times when planning by using Q-learning algorithm and the EMQL algorithm.

| No. | The Target Point | Iteration Times (QL) | Iteration Times (EMQL) | Comparison |
|---|---|---|---|---|
| 1 | (958508.87,4355976.17) | 18423 | 4409 | ↓76.07% |
| 2 | (949806.10,4354281.25) | 16967 | 3143 | ↓81.48% |
| 3 | (956821.20,4350914.58) | 17817 | 3866 | ↓78.30% |
| 4 | (953423.30,4355219.39) | 17082 | 4488 | ↓73.73% |
| 5 | (966349.16,4347350.72) | 14056 | 3614 | ↓74.29% |
| 6 | (943744.00,4353509.70) | 14702 | 2469 | ↓83.21% |
| 7 | (942330.23,4341038.01) | 12593 | 826 | ↓93.44% |
| 8 | (941827.77,4350828.07) | 14647 | 1764 | ↓87.96% |
| 9 | (941146.23,4346533.47) | 14330 | 1421 | ↓90.08% |
| 10 | (958677.74,4328891.17) | 8320 | 722 | ↓91.32% |

**TABLE 5.** The comparison of path length when planning by using Q-learning algorithm and the EMQL algorithm.

| No. | The Target Point | Path Length (QL) | Path Length (EMQL) | Comparison |
|---|---|---|---|---|
| 1 | (958508.87,4355976.17) | 36981.10m | 33363.94m | ↓9.78% |
| 2 | (949806.10,4354281.25) | 29257.95m | 27391.58m | ↓6.38% |
| 3 | (956821.20,4350914.58) | 30353.76m | 26868.77m | ↓11.48% |
| 4 | (953423.30,4355219.39) | 31429.21m | 29591.71m | ↓5.85% |
| 5 | (966349.16,4347350.72) | 33877.98m | 30997.88m | ↓8.50% |
| 6 | (943744.00,4353509.70) | 29295.86m | 25194.90m | ↓14.00% |
| 7 | (942330.23,4341038.01) | 15359.28m | 13958.47m | ↓9.12% |
| 8 | (941827.77,4350828.07) | 25791.77m | 23234.12m | ↓9.92% |
| 9 | (941146.23,4346533.47) | 23043.07m | 22430.52m | ↓2.66% |
| 10 | (958677.74,4328891.17) | 16766.33m | 15921.22m | ↓5.04% |

pentagram and the magenta line in the figure represent the start point, the target point and the planned path respectively. In addition, the lines marked blue in the picture are the roads the robot searches in the learning stage, and the magenta curve is the path the robot ultimately plans. In the picture shown on the left, the roads searched by the robot almost

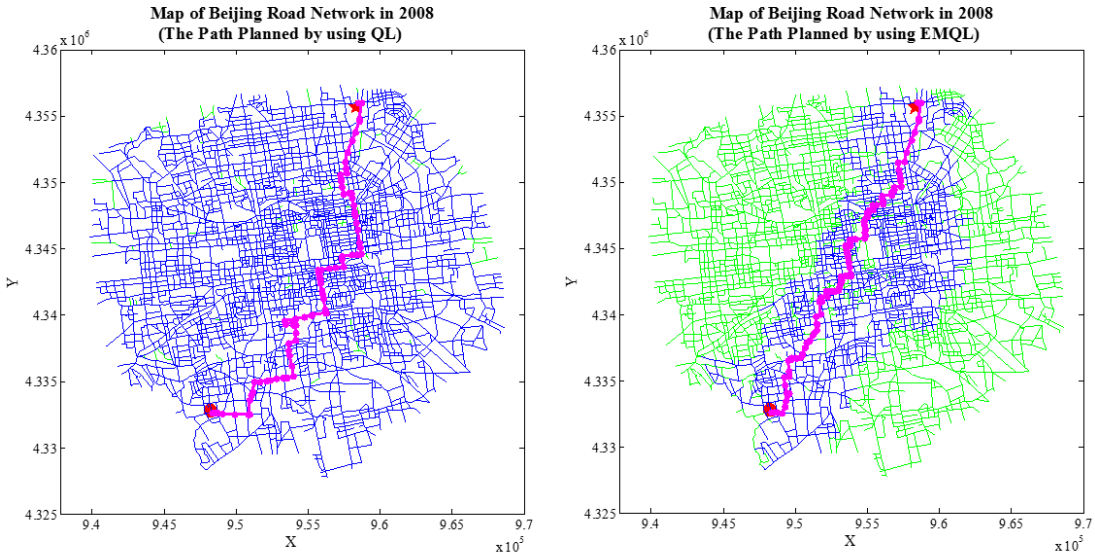


**FIGURE 11.** The results of path planning of the Q-learning algorithm (left) and the EMQL algorithm (right).

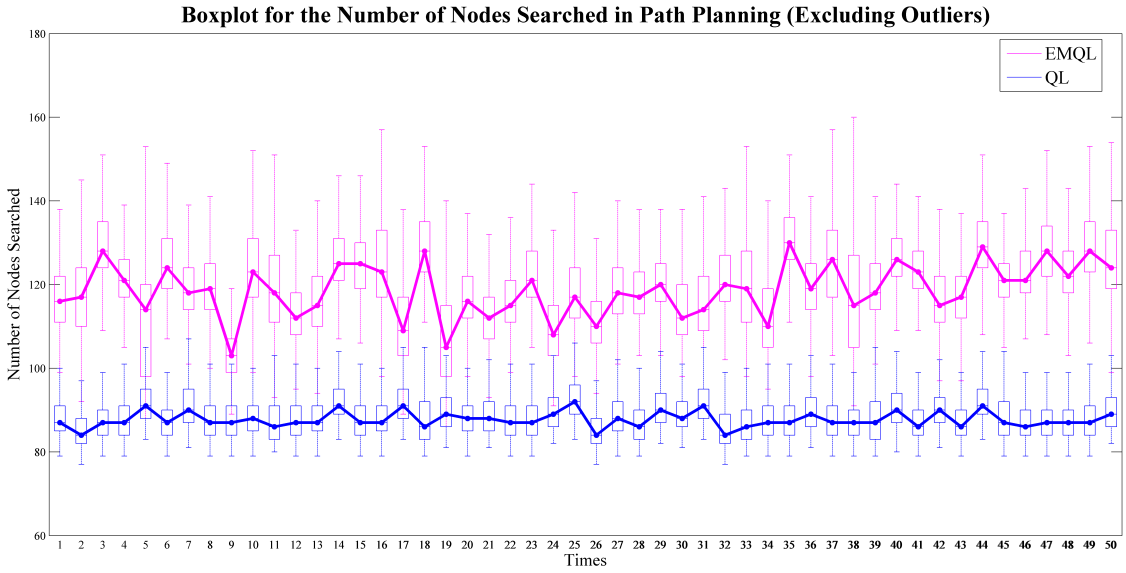


**FIGURE 12.** The boxplot for comparing the number of nodes searched by the robot in path planning without considering the outliers.

cover the whole map, which illustrates that the robot searches blindly when it plans by using Q-learning algorithm. However, in the EMQL algorithm, most of the searched nodes are located on both sides of the line between the start point and the target point. Obviously, such a search costs less time than the blind search in Q-learning algorithm. Therefore, we can draw a preliminary conclusion that if a robot needs to plan a path in unknown environment, this dual reward mechanism would improve the learning efficiency of the robot, which meets the practical application requirements.

Despite that the number of nodes searched by the robot in Q-learning algorithm is much more than that searched in the EMQL algorithm, the performance of the EMQL algorithm is better than Q-learning algorithm in terms of the length of the planned path. In order to explore the reasons for this result, we record the number of nodes searched by the robot of each iteration in one experiment.

The completion of an iteration represents that the robot completes a path search from the start point to the target point, but the number of nodes searched is different due to the different experience memorized in the process. Figure 12 shows the boxplot for the number of state nodes the robot searches during the learning stage in each iterations. Here, the outliers refer to the number of nodes that the robot searches less

**TABLE 6.** Number of nodes searched by the robot in the first experiment obtained from the boxplot.

| Algorithm | QL | EMQL |
|---|---|---|
| Iteration Times | 16839 | 5244 |
| Lower Adjacent | 79 | 99 |
| Upper Adjacent | 100 | 138 |
| Median | 87 | 116 |

frequently in many iterations. For example, the robot has experienced 3000 iterations to complete the path planning task. It only searches 400 state nodes of the environment in two iterations, and 400 is the abnormal value of the experiment data. In addition, we take the first group of experiments as an example, and list the relevant data obtained from the boxplot in Table 6.

As the data shown in Table 6, in the EMQL algorithm, the robot searches 99 to 138 nodes in most cases during each search, and the information of these nodes helps the robot plan the final path. As for Q-learning algorithm, the upper adjacent of the nodes the robot searches in the environment is 100, which is almost equal to the lower adjacent of that the robot searches in the EMQL algorithm. This result insinuates that most nodes searched by the robot are repetitive when the robot plans by using Q-learning algorithm. This conclusion can also be verified from Table 7 which lays out the proportion of outliers in the total number of node the robot searches.

The median of the number of searched nodes are marked in Figure 12, which represents the general level of all data. Obviously, the median number of searched nodes in the EMQL algorithm is higher than that of the Q-learning algorithm in each experiment. Over exploring the same node will limit the useful information of the environment obtained by the robot, which leads to the loss of the opportunity to explore a better path. Therefore, the robot is more likely to get the shortest path when it plans by using the EMQL algorithm in unknown environment because the diversity of nodes the robot searches in the proposed algorithm is higher than that of Q-learning algorithm.

Another highlight of the EMQL algorithm is the experience memory mechanism, using the EM table to judge the completion of the planning task. In order to verify that the use of EM table enables the robot to complete the planning task more quickly, we take one of the 50 experiments as

**TABLE 7.** The proportion of outlier in the total number of nodes searched by the robot when planning by using Q-learning algorithm and the EMQL algorithm.

| No. | Proportion of Outliers (QL) | Proportion of Outliers (EMQL) | No. | Proportion of Outliers (QL) | Proportion of Outliers (EMQL) |
|---|---|---|---|---|---|
| 1 | 8.63% | **4.25%** | 26 | 8.79% | **4.01%** |
| 2 | 8.06% | **1.67%** | 27 | 9.49% | **4.57%** |
| 3 | 7.49% | **4.96%** | 28 | 6.72% | **5.09%** |
| 4 | 7.22% | **2.63%** | 29 | 8.25% | **6.35%** |
| 5 | 8.23% | **0.77%** | 30 | 7.99% | **6.56%** |
| 6 | 7.65% | **3.47%** | 31 | 7.72% | **2.80%** |
| 7 | 6.97% | 6.98% | 32 | 8.66% | **2.99%** |
| 8 | 7.71% | **4.02%** | 33 | 6.65% | **1.31%** |
| 9 | 7.34% | **4.92%** | 34 | 7.89% | **2.80%** |
| 10 | 8.22% | **1.51%** | 35 | 8.35% | **3.11%** |
| 11 | 5.86% | **1.51%** | 36 | 7.90% | **4.55%** |
| 12 | 8.43% | **5.05%** | 37 | 7.74% | **1.05%** |
| 13 | 8.67% | 9.11% | 38 | 8.16% | **0.72%** |
| 14 | 8.41% | **2.11%** | 39 | 5.22% | 6.43% |
| 15 | 7.61% | **2.14%** | 40 | 7.49% | **5.47%** |
| 16 | 8.32% | **1.62%** | 41 | 7.79% | **5.51%** |
| 17 | 7.83% | **4.62%** | 42 | 8.77% | **3.68%** |
| 18 | 6.31% | **2.88%** | 43 | 7.19% | **4.02%** |
| 19 | 7.40% | **1.26%** | 44 | 8.34% | **4.84%** |
| 20 | 9.25% | **5.74%** | 45 | 7.73% | **4.91%** |
| 21 | 6.36% | **1.80%** | 46 | 7.75% | **6.36%** |
| 22 | 7.93% | **4.38%** | 47 | 7.39% | **2.86%** |
| 23 | 8.10% | **4.12%** | 48 | 7.09% | **4.46%** |
| 24 | 7.17% | **4.69%** | 49 | 7.49% | **2.33%** |
| 25 | 8.93% | **1.88%** | 50 | 6.29% | **4.21%** |

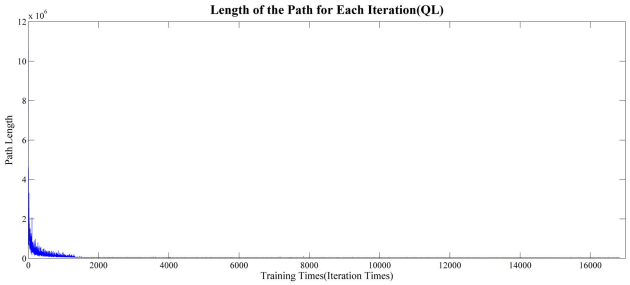


**FIGURE 13.** Length of the path for each training when planning by using Q-learning algorithm.

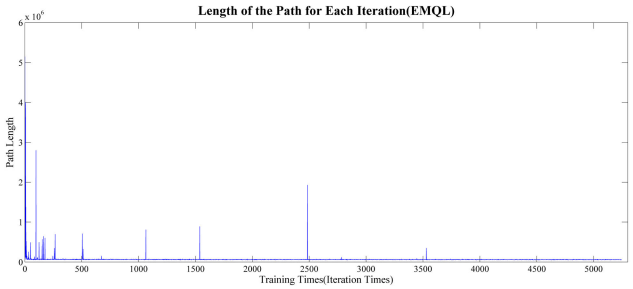


**FIGURE 14.** Length of the path for each training when planning by using the EMQL algorithm.

an example, and record the length of the paths that the robot moves from the start point to the end point each time in the learning stage. Figure 13 and Figure 14 show the path length in each training when the robot plans by using the EMQL algorithm and Q-learning algorithm respectively.

As the curve shown in Figure 13, the robot searches the ''shortest'' path quickly based on the search strategy of Q-learning algorithm. However, the path searched by the robot is almost unchanged after 1300 iterations, which can be inferred that it falls into local optimum too early. The reason for this result is that the optimal action choice at a node only be changed after many iterations due to the large dimension of the Q table. The delayed update makes the robot lack the ability to adjust actions in time. In order to make the experiment result more intuitive, we randomly select five moving paths of the robot in its learning stage and draw them in Figure 15. Obviously, when the robot plans the path by using Q-learning algorithm, most of the paths it searches are similar.

However, according to the experience memory mechanism in the EMQL algorithm, the distance between a state node and the start point is constantly compared with the existing experience. If a shorter path to a state node is found, the information recorded in the EM table and Instructions will be updated immediately. Therefore, even the nodes explored few times can help the robot get the experience of the optimal path in time. As illustrated in Figure 16, the paths searched by the robot in the learning stage are more diverse than that of Q-learning algorithm.

Therefore, the shortcoming of Q-learning algorithm which is easy to fall into local optimum in the process of path

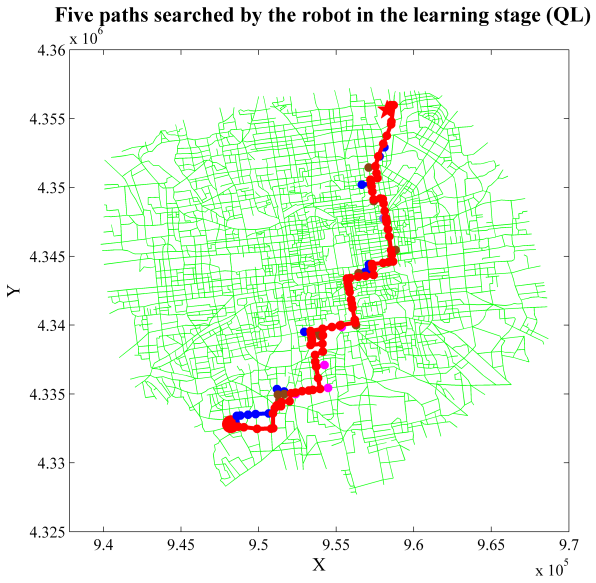


**FIGURE 15.** The five paths searched by the robot in the learning stage when it plans a path by using Q-learning algorithm, which are labeled in five different colors. The red dot and the red pentagram in the figure represent the start point and the target point respectively.

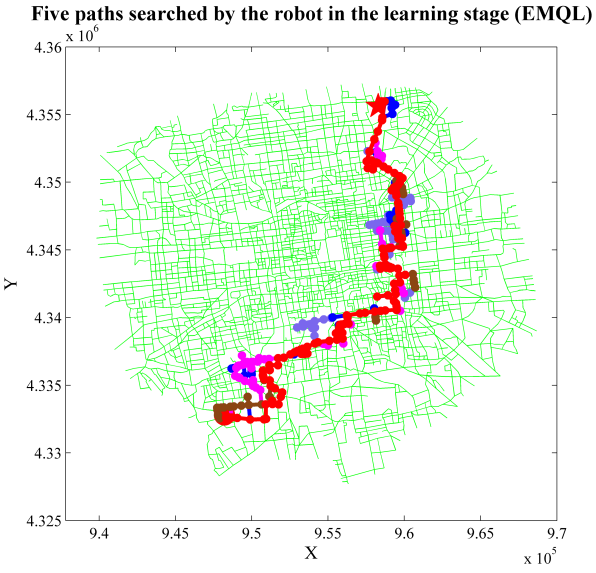


**FIGURE 16.** The five paths searched by the robot in the learning stage when it plans a path by using the EMQL algorithm, which are labeled in five different colors. The red dot and the red pentagram in the figure represent the start point and the target point respectively.

planning can be made up by the use of the EM table. Besides, the EMQL algorithm has better optimization ability than the Q-learning algorithm in robot path planning.

The EMQL algorithm is not only better than the Q-learning algorithm in the path optimization, but also in the convergence speed. In the experiments of this paper, the criterion of convergence is that the difference of experience table is less than 0.0001 for 100 consecutive times. The experience lists in Q-learning algorithm and the EMQL algorithm are Q table

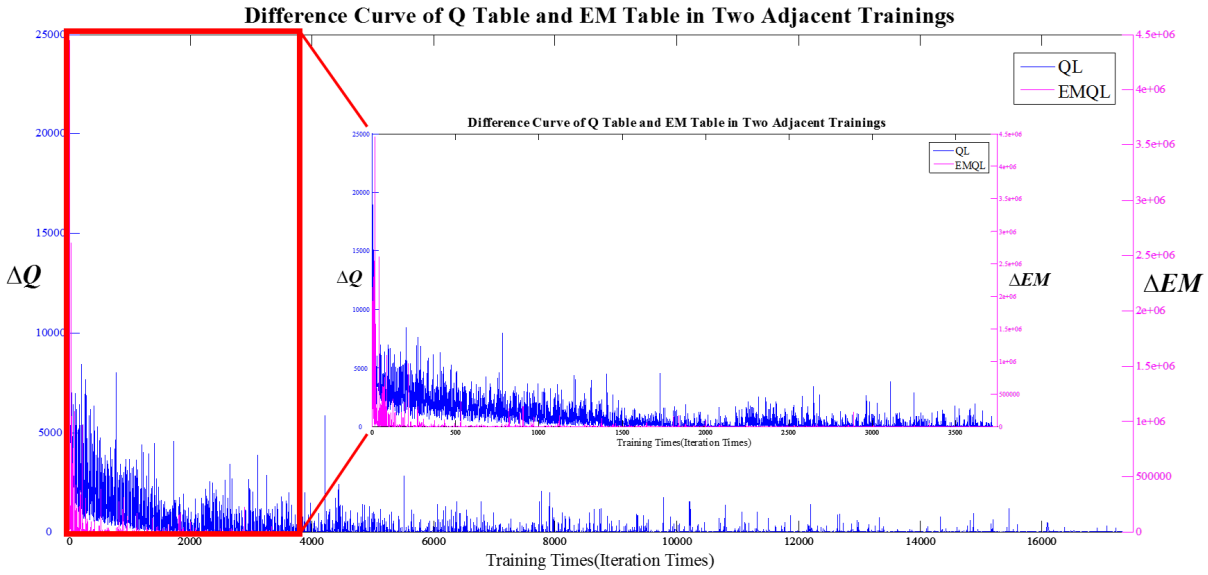


**FIGURE 17.** Difference curve of Q table and EM table in two adjacent trainings.

and EM table respectively. Figure 17 shows the difference curve of Q table and EM table in two adjacent trainings, where the difference of the Q table and EM table are defined as (7) and (8) respectively.

$$\triangle Q = |\sum_{i=1}^{m}\sum_{j=1}^{n} Q_{t+1} - \sum_{i=1}^{m}\sum_{j=1}^{n} Q_t| \tag{7}$$

$$\triangle EM = |\sum_{k=1}^{m} EM_{t+1} - \sum_{k=1}^{m} EM_t| \tag{8}$$

As shown in Figure 17, the EM table converges after 3720 iterations in the EMQL algorithm, while the robot needs 16894 iterations to make the Q table converge. Therefore, the convergence speed of the EMQL algorithm is obviously faster than Q-learning algorithm.

From the perspective of dimension of experience list, if the robot finds $m$ nodes in unknown environment and the maximum number of the optional actions for each node is $n$. The experience list of Q-learning algorithm is Q table whose dimension is $m*n$, while the dimension of the EM table used in the EMQL algorithm is $m*1$. The difference between the two table dimensions indicates that there would be a huge gap in the convergence time between them if the robot works in a large unknown environment. Moreover, in the problem of path planning, achieving convergence means that the robot completes the learning stage and has the ability to accomplish the task. Therefore, the learning efficiency of the robot will be improved in the proposed algorithm due to the use of the EM table.

To sum up, due to the experience memory mechanism and dual reward in the EMQL algorithm, the robot can find a shorter path on the premise of fast convergence, which makes it possible for the robot to plan a short path quickly in unknown environment.

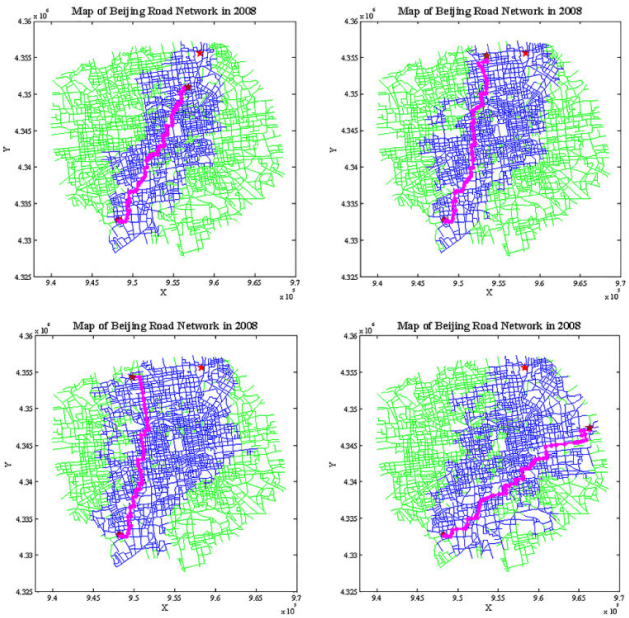


**FIGURE 18.** The planned path in the new task when the robot is let to arrive another destination in the stage of learning. The red dot, the red pentagram, the brown pentagram and the magenta line in the figure represent the start point, the original target point, the new target point and the final planned path respectively.

### 2) DYNAMIC PATH PLANNING

In order to verify the effectiveness of experience transfer strategy and experience reuse strategy, we design two

**TABLE 8.** The comparison of the iteration times when planning the path using the experience reuse strategy or not.

| No. | The Target Point | Iteration Times (No Experience) | Iteration Times (Experience) | Comparison |
|---|---|---|---|---|
| 1 | (956821.20,4350914.58) | 3866 | 3380 | ↓12.57% |
| 2 | (953423.30,4355219.39) | 4488 | 2951 | ↓34.25% |
| 3 | (949806.10,4354281.25) | 3143 | 2932 | ↓6.71% |
| 4 | (966349.16,4347350.72) | 3614 | 2424 | ↓32.93% |

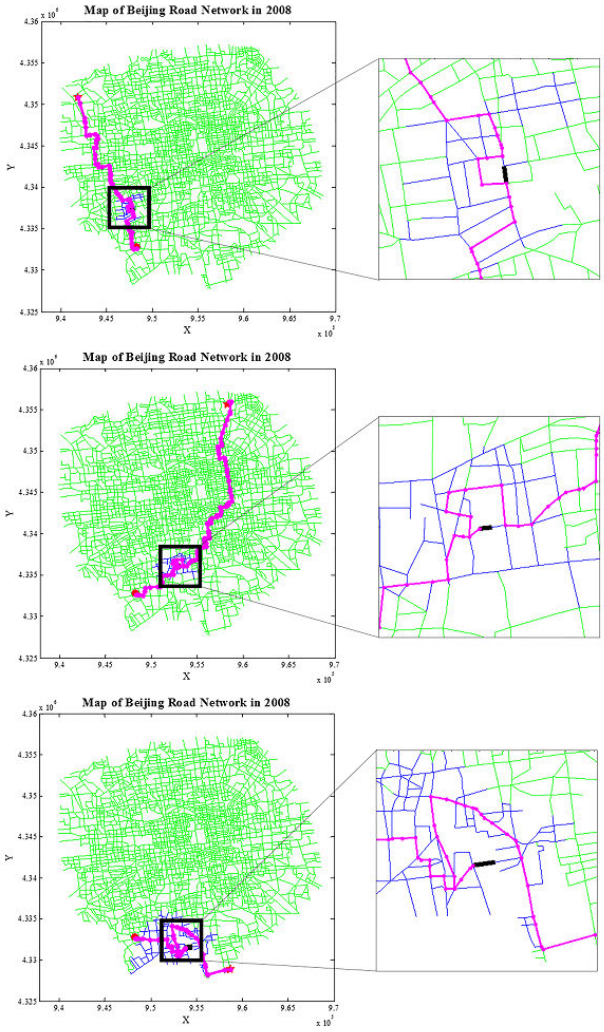


**FIGURE 19.** The new planned path when the original planned path is blocked. The red dot, the red pentagram, the bold black line and the magenta line in the figure represent the start point, the target point, the blocked road and the final planned path respectively.

experiments of robot dynamic path planning in the road network map: destination change and path blockage.

### *a: DESTINATION CHANGE*

In this dynamic scene, the node (958508.87, 4355976.17) is set as the original target point, which will change after 1000 iterations. In order to ensure the reliability of the experiment results, we select four nodes located in different positions in the environment as the new target points, and show one of the replanning paths in fifty experiments in Figure 18. The red pentagram in the picture is the original target point, while the new destination is marked as brown pentagram. In this experiment, the process of the robot directly reaching the new target point is defined as inexperienced path planning, while the process of robot reaching the new target by using the experience transfer strategy is regarded as the experienced path planning. In order to verify the effectiveness of the experience transfer strategy, we count the average iterations times needed by the robot to complete the planning task under the experienced and inexperienced conditions in 50 experiments, and compare them in Table 8.

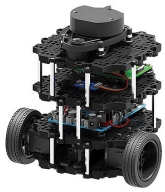

**FIGURE 20.** Turtlebot3 burger robot.

Obviously, when the planning task is identical, the robot with experience can complete the path work faster than the robot with no experience. This conclusion has nothing to do with the relative position of the new and original target points in the environment. Therefore, it is proved that the experience transfer strategy can help the robot adjust the path in time and improve the efficiency of path replanning in unknown environment.

### *b: PATH BLOCKAGE*

In addition to the dynamic scene of destination change, the robot also needs to adjust its moving path when it finds that the planned path is blocked. Therefore, to prove that the experience reuse strategy is effective, we randomly block the road when the robot moves along the planned path. Some of the replanning results are shown in Figure 19. The blue curves in the picture are the roads the robot searches in the process of short-distance relearning and the thick black lines represents the blocked path. Obviously, the robot can adjust the route in time and accomplish the task safely.

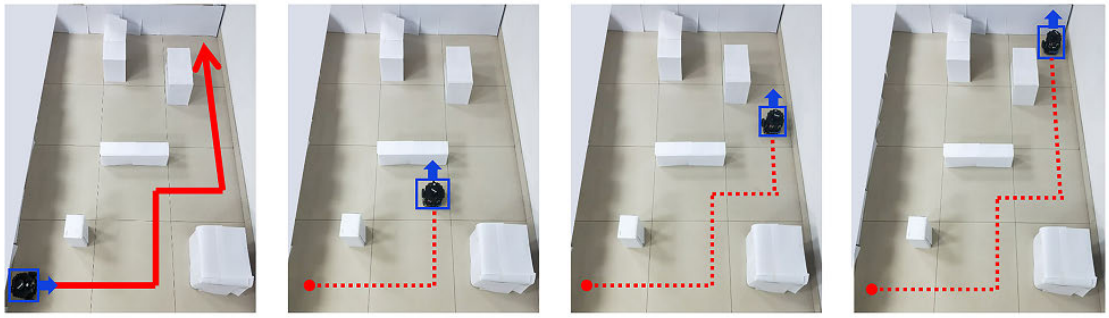

**FIGURE 21.** Some snapshots of the robot as it moves from the start point to the target point when the task and environment is constant.

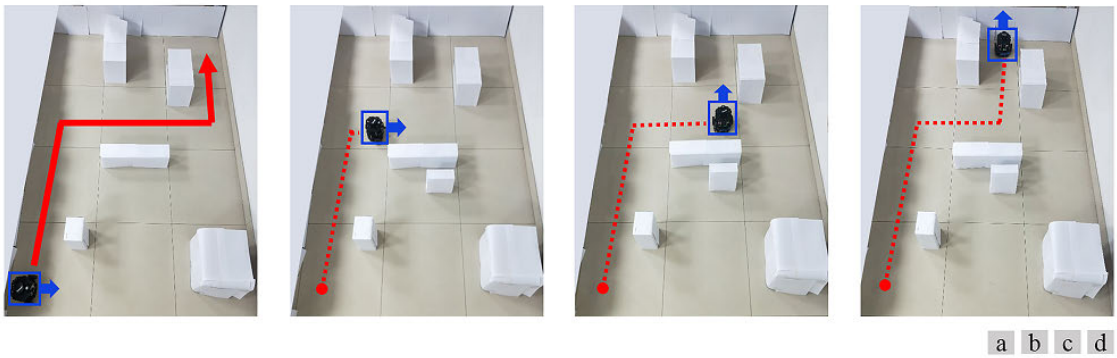


**FIGURE 22.** Some snapshots of the robot as it moves from the start point to the target point when the task is changed.

The experiment results show that the short-distance relearning can improve the ability of robot path replanning by reusing the original experience as much as possible. Therefore, the use of experience reuse strategy enables the robot to adapt to dynamic working scenes, which further illustrates that the EMQL algorithm is applicable to the path planning of the robot in unknown environment.

### C. RUNNING ON Turtlebot3 BURGER ROBOT

Turtlebot3 burger (see Figure 20) is a miniature robot equipped with a 360° laser radar, the size of which is 138 * 178 * 192 (L * w * h, mm). Because it is low-cost and open-source while ensuring function and quality, it has been widely used in the field of robot path planning.

In this paper, we do three experiments on Turtlebot3 burger to test the practicability of the EMQL algorithm, and some of the snapshots are shown in each experiment. Because the maximum speed of the robot is 0.22m/s, it will take a lot of time for the robot to repeatedly move from the start point to the end point hundreds of times. In order to better present the experiment results, the learning process of robot in unknown environment is executed in the simulation environment. After the learning stage is over, the robot will complete the planning stage in the actual environment according to the information memorized in the EM table, Q table and Instructions.

In the first experiment, the robot needs to plan a path in unknown environment. Both of the planning task and the environment is constant in this experiment. Therefore, the robot only needs to learn the shortest distance from each state node to the start point in the simulation environment, and then move along the state nodes recorded in Instructions in the actual environment. Figure 21 shows the planned path of the robot and some snapshots of the moving process.

In the second experiment, the effectiveness of the experience transfer strategy in the EMQL algorithm will be verified. Figure 22(a) shows the original planned path. The robot knows that the target point is changed in the position shown in Figure 22(b), and then it continues moving by referring to the $Q(s, a)$. According to the experience transfer strategy, the robot will adopt the $\epsilon$-greedy strategy again to select the action if it detects that there is a deviation in the moving direction until the task is completed. Figure 22(d) shows the actual movement path of the robot.

The third experiment is designed to verify that the experience reuse strategy in EMQL algorithm can help the robot arrive the end point safely even when the path is blocked suddenly. In Figure 23, the robot finds that it cannot reach the next waypoint (see Figure 23(c)) when it moves along the planned path. Then, with the guidance of the experience reuse strategy, it learns a detour shown in Figure 23(d). After the post-processing of the detour, the robot continues moving

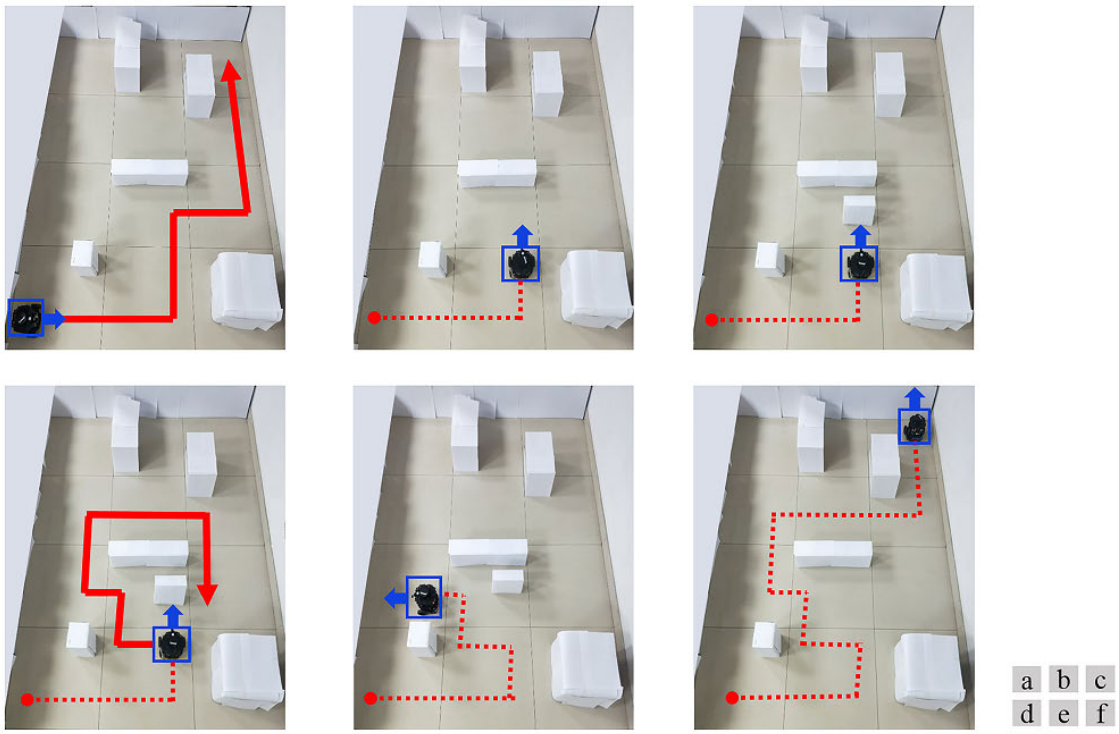


**FIGURE 23.** Some snapshots of the robot as it moves from the start point to the target point when the environment is changed.

to the destination. The actual moving path of the robot is shown in Figure 23(f), which is different from the original planned path shown in Figure 23(a) because of the change of the environment.

### D. SUMMARY

Based on the experiment results and the comparison with Q-learning algorithm, it can be found that the EMQL algorithm has good applicability for robot path planning in unknown environment. Also, the dual reward mechanism combining static reward and dynamic reward can help the robot avoid search blindly in unknown environment and make the learning efficiency of the robot improved. Besides, the dimension of the EM table used in the proposed algorithm is far smaller than that of the Q table, and the robot will spend less time completing the learning stage when using the EMQL algorithm. In addition, it has been verified from the repeatability of the paths and the diversity of the state nodes searched by the robot each time that the path optimization ability of the EMQL algorithm is stronger than that of Q-learning algorithm. Moreover, Q table is employed as the auxiliary experience list in the experience transfer strategy and the experience reuse strategy. With the guidance of these strategies, the robot can accomplish the path planning task safely even when the target is changed or the path is blocked. Overall, the EMQL algorithm performs better than Q-learning algorithm in terms of the applicability, convergence, optimization ability and dynamic adaptability.

## VI. CONCLUSION

Path planning in unknown environment is a significant research area in robotics. In order to improve the path planning ability of the robot, we propose the EMQL algorithm. There are three main highlights in the proposed algorithm. First, the dual reward mechanism makes the robot avoid blind search in unknown environment, increasing the likelihood to find a shorter path. Second, based on the experience memory mechanism, we design the EM table instead of Q table to reflect the learning process of the robot. The dimension of the EM table is much smaller than that of Q table, improving the learning efficiency of the robot. Third, two strategies are put forward in the EMQL algorithm to ensure that the robot can successfully complete the planning task in dynamic scenes. When the target point changes or the road feasibility changes, the robot will adopt the experience transfer strategy and the experience reuse strategy respectively. In addition, Q table works as an auxiliary experience in these strategies, which helps the robot accomplish the new task efficiently. The statistical results of different experiments illustrate that the EMQL algorithm is superior to Q-learning algorithm in terms of convergence, optimization ability and dynamic adaptability, and it meets the actual application requirements.

The successful completion of the planning task when the task changes and the environment changes illustrates the effectiveness of experience transfer strategy and experience reuse strategy in the EMQL algorithm. Therefore, in order

to improve the applicability of path planning algorithm, our future work is to improve the transfer strategy so that the experience of the robot can be transferred in different working environments. Most of the existing path planning algorithm based on reinforcement learning can only be learned and applied in the same environment. The experience gained before becomes useless as the working environment changes and the robot has to plan the path by relearning, which costs plenty of time. Hence improving the experience transfer ability of the robot is one of the great solutions to realize robot intelligent path planning.

## REFERENCES


[1] X. Yang, M. Moallem, and R. V. Patel, "A layered goal-oriented fuzzy motion planning strategy for mobile robot navigation," *IEEE Trans. Syst., Man Cybern. B, Cybern.*, vol. 35, no. 6, pp. 1214–1224, Dec. 2005.

[2] B. M. Keneni, D. Kaur, A. Al Bataineh, V. K. Devabhaktuni, A. Y. Javaid, J. D. Zaientz, and R. P. Marinier, "Evolving rule-based explainable artificial intelligence for unmanned aerial vehicles," *IEEE Access*, vol. 7, pp. 17001–17016, 2019.

[3] Q. M. Nguyen, L. N. M. Tran, and T. C. Phung, "A study on building optimal path planning algorithms for mobile robot," in *Proc. 4th Int. Conf. Green Technol. Sustain. Develop. (GTSD)*, Nov. 2018, pp. 341–346.

[4] B. Sun, D. Zhu, and S. X. Yang, "An optimized fuzzy control algorithm for three-dimensional AUV path planning," *Int. J. Fuzzy Syst.*, vol. 20, no. 2, pp. 597–610, Feb. 2018.

[5] W.-J. Chen, B.-G. Jhong, and M.-Y. Chen, "Design of path planning and obstacle avoidance for a wheeled mobile robot," *Int. J. Fuzzy Syst.*, vol. 18, no. 6, pp. 1080–1091, Dec. 2016.

[6] S. Sahloul, D. Benhalima, and C. Rekik, "Comparative study of hybrid fuzzy logic methods for mobile robot navigation in unknown environments," in *Proc. 19th Int. Conf. Sci. Techn. Autom. Control Comput. Eng. (STA)*, Mar. 2019, pp. 164–169.

[7] M. A. P. Garcia, O. Montiel, O. Castillo, R. Sepĺšlveda, and P. Melin, "Path planning for autonomous mobile robot navigation with ant colony optimization and fuzzy cost function evaluation," *Appl. Soft Comput.*, vol. 9, no. 3, pp. 1102–1110, Jun. 2009.

[8] Y. Wang, P. Bai, X. Liang, W. Wang, J. Zhang, and Q. Fu, "Reconnaissance mission conducted by UAV swarms based on distributed PSO path planning algorithms," *IEEE Access*, vol. 7, pp. 105086–105099, 2019.

[9] T. Yifei, Z. Meng, L. Jingwei, L. Dongbo, and W. Yulin, "Research on intelligent welding robot path optimization based on GA and PSO algorithms," *IEEE Access*, vol. 6, pp. 65397–65404, 2018.

[10] P. Caricato and A. Grieco, "Using simulated annealing to design a material-handling system," *IEEE Intell. Syst.*, vol. 20, no. 4, pp. 26–30, Jul. 2005.

[11] A. Bhaduri, "A mobile robot path planning using genetic artificial immune network algorithm," in *Proc. World Congr. Nature Biologically Inspired Comput.*, Dec. 2009, pp. 1536–1539.

[12] K. Balan and C. Luo, "Optimal trajectory planning for multiple waypoint path planning using tabu search," in *Proc. 9th IEEE Annu. Ubiquitous Comput., Electron. Mobile Commun. Conf. (UEMCON)*, Nov. 2018, pp. 497–501.

[13] R. C. Smith and P. Cheeseman, "On the representation and estimation of spatial uncertainty," *Int. J. Robot. Res.*, vol. 5, no. 4, pp. 56–68, Dec. 1986.

[14] M. W. M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba, "A solution to the simultaneous localization and map building (SLAM) problem," *IEEE Trans. Robot. Autom.*, vol. 17, no. 3, pp. 229–241, Jun. 2001.

[15] M. Blosch, S. Weiss, D. Scaramuzza, and R. Siegwart, "Vision based MAV navigation in unknown and unstructured environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2010, pp. 21–28.

[16] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. 6th IEEE ACM Int. Symp. Mixed Augmented Reality*, Nov. 2007, pp. 225–234.

[17] I. Maurovic, M. Seder, K. Lenac, and I. Petrovic, "Path planning for active SLAM based on the D* algorithm with negative edge weights," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 8, pp. 1321–1331, Aug. 2018.

[18] S. Yang, S. Yang, and X. Yi, "An efficient spatial representation for path planning of ground robots in 3D environments," *IEEE Access*, vol. 6, pp. 41539–41550, 2018.

[19] T.-J. Lee, C.-H. Kim, and D.-I.-D. Cho, "A monocular vision sensor-based efficient SLAM method for indoor service robots," *IEEE Trans. Ind. Electron.*, vol. 66, no. 1, pp. 318–328, Jan. 2019.

[20] 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 Eng.*, vol. 189, Oct. 2019, Art. no. 106299.

[21] S. Zhou, X. Liu, Y. Xu, and J. Guo, "A deep Q-network (DQN) based path planning method for mobile robots," in *Proc. IEEE Int. Conf. Inf. Autom. (ICIA)*, Aug. 2018, pp. 366–371.

[22] W. Zhang, J. Gai, Z. Zhang, L. Tang, Q. Liao, and Y. Ding, "Double-DQN based path smoothing and tracking control method for robotic vehicle navigation," *Comput. Electron. Agricult.*, vol. 166, Nov. 2019, Art. no. 104985.

[23] D. Xu, Y. Fang, Z. Zhang, and Y. Meng, "Path planning method combining depth learning and Sarsa algorithm," in *Proc. 10th Int. Symp. Comput. Intell. Design (ISCID)*, Dec. 2017, pp. 77–82.

[24] Y. Liu, W. Zhang, F. Chen, and J. Li, "Path planning based on improved deep deterministic policy gradient algorithm," in *Proc. IEEE 3rd Inf. Technol., Netw., Electron. Autom. Control Conf. (ITNEC)*, Mar. 2019, pp. 295–299.

[25] I. Goswami, P. K. Das, A. Konar, and R. Janarthanan, "Extended Q-learning algorithm for path-planning of a mobile robot," in *Proc. 8th Int. Conf. Simulated Evol. Learn.* Kanpur, India: Springer-Verlag, Dec. 2010, pp. 379–383.

[26] 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.

[27] E. S. Low, P. Ong, and K. C. Cheah, "Solving the optimal path planning of a mobile robot using improved Q-learning," *Robot. Auto. Syst.*, vol. 115, pp. 143–161, May 2019.

[28] Q. Gao, B. Hong, Z. He, L. Jie, and G. Niu, "An improved Q-learning algorithm based on exploration region expansion strategy," in *Proc. World Congr. Intell. Control Automat.*, vol. 1, Jun. 2006, pp. 4167–4170.

[29] P. K. Das, H. S. Behera, and B. K. Panigrahi, "Intelligent-based multi-robot path planning inspired by improved classical Q-learning and improved particle swarm optimization with perturbed velocity," *Eng. Sci. Technol., Int. J.*, vol. 19, no. 1, pp. 651–669, Mar. 2016.

[30] P. Rakshit, A. Konar, P. Bhowmik, I. Goswami, S. Das, L. C. Jain, and A. K. Nagar, "Realization of an adaptive memetic algorithm using differential evolution and Q-learning: A case study in multirobot path planning," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 43, no. 4, pp. 814–831, Jul. 2013.

[31] C. J. Watkins and P. Dayan, "Technical note: Q-learning," *Mach. Learn.*, vol. 8, no. 3–4, pp. 279–292, May 1992.

[32] N. R. Sturtevant, "Benchmarks for grid-based pathfinding," *IEEE Trans. Comput. Intell. AI Games*, vol. 4, no. 2, pp. 144–148, Jun. 2012.

[33] J. Song, L. Gao, and X. Shan, "Historical street network GIS datasets of Beijing within 5th ring-road," *Chin. Sci. Data*, vol. 2, pp. 1–6, Dec. 2016.


**MENG ZHAO** (Student Member, IEEE) received the B.Sc. degree from the School of Electronic and Information Engineering, Communication University of China, Beijing, China, in 2018. She is currently pursuing the Ph.D. degree with Beihang University. Her main research areas include robot path planning and group intelligent navigation.

**HUI LU** (Senior Member, IEEE) received the Ph.D. degree in navigation, guidance and control from Harbin Engineering University, Harbin, China, in 2004. She is currently a Professor with Beihang University and a member of the Shaanxi Key Laboratory of Integrated and Intelligent Navigation. Her research interests include information and communication systems, intelligent optimization, and fault diagnosis and prediction.

**SIYI YANG** received the B.Sc. degree from the School of Information Science and Technology, Xiamen University, Xiamen, China, in 2019. She is currently pursuing the master's degree with Beihang University. Her main research areas include communication and path planning in multirobot systems.

**FENGJUAN GUO** received the Ph.D. degree in systems engineering from Northwestern Polytechnical University, Xi'an, China, in 2012. She is currently a Researcher with the Shaanxi Key Laboratory of Integrated and Intelligent Navigation. Her research interests in integrated navigation and intelligent navigation.