IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 25, 2021, accepted April 11, 2021, date of publication April 16, 2021, date of current version April 26, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3073704

UAV Path Planning Based on Multi-Layer Reinforcement Learning Technique

ZHENGYANG CUI AND YONG WANG
School of Automation Science and Electrical Engineering, Beihang University, Beijing 100191, China
Corresponding author: Zhengyang Cui (respond89@sina.cn)

ABSTRACT Unmanned aerial vehicles (UAVs) have been widely used in many applications due to its small size, swift mobility and low cost. Therefore, the study of guidance, navigation and control (GNC) system of UAV has becoming a popular research direction. Path planning plays an important role in the GNC system. In this paper, a multi-layer path planning algorithm based on reinforcement learning (RL) technique is proposed. Compared to the classic Q-learning, the proposed multi-layer algorithm has a distinct advantage that it collects both global and local information which greatly improves overall performance. The proposed RL algorithm has two layers, the higher layer deals with the local information, which could be considered as a short-term strategy. The lower layer deals with the global information, which could be considered as a long-term strategy. Both the higher layer and lower layer are working in harmony to plan a collision-free path. B-spline curve approach is applied for on-line path smoothing. Simulation results in different scenarios prove the effectiveness of multi-layer Q-learning algorithm.

INDEX TERMS Path planning, reinforcement learning, multi-layer Q-learning, B-spline curve.

I. INTRODUCTION

Nowadays, with the fast developing technology, unmanned aerial vehicle (UAV) is becoming a popular autonomous robots according to its distinct advantages such as cheaper price, smaller size, extraordinary flexibility and swift mobility [1], [2]. The UAVs are required to have higher intelligence when facing complex and complicated missions. Therefore, the self-learning and self-decision making abilities are becoming more and more important.

In the related research topics of UAV, the path planning is a fundamental topic of the guidance navigation and control (GNC) of UAV. The objective of path planning aims at finding the optimal trajectory from the start location to the terminal location. One important feature of path planning is the obstacle avoidance, which means the UAV has to avoid the obstacles during execution no matter they are static of dynamic. However, with the fast developing machine learning (ML) and reinforcement learning (RL) techniques, the UAV path planning approaches have revealed a new page.

According to the collected information, path planning approaches will be categorized into two directions [3]. One is global path planning and the other is local path planning. Global path planning means the environment information is pre-known. The objective of global path planning aims at finding a collision-free path which connects the start point and terminal point before execution. Since the environment information is known, the algorithms for global path planning problems usually use off-line policy. Voronoi graph [4]–[6], cell decomposition [7]–[10], potential field method [11]–[15] are the most commonly used methods for global path planning problem. Local path planning means the environment information is unknown or partially known. In local path planning, the moving direction of UAV is based on limited environment information acquired in real-time. As a result, the algorithms for local path planning problems often based on on-line policy. There are a number of local path planning approaches, such as particle swarm optimization (PSO) [16]–[18], fuzzy logic system (FLS) [19]–[21], and ant colony (ASO) algorithm [22]–[24]. It should be noted that the global path planning approaches will be extraordinary computationally expensive when facing a complex environment.

ML techniques have been deeply investigated and used in a wide applications in the autonomous agents and unmanned systems because of its adaptive learning ability [25], [26]. RL is one of the branches of ML, which can optimize the action policy based on the interactions

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

between agent and environment. Especially for navigation in complex environment, RL has showed its powerful learning ability [27], [28].

In recent years, RL is becoming a promising approach for solving path planning problem. Q-learning was first proposed in [29], it can be learned from delayed rewards and punishments. After that, a number of researchers focus on solving the path planning problem using RL. In [30], [31], the authors proposed a Q-learning based navigation system for obstacle avoidance problem. The successful rate is considered to the high and the overall performance is good. Some literatures improved the Q-learning algorithm for navigation system design. In [32], an extended Q-learning algorithm is investigated, only the best action of a state will be stored in the Q-table. This design greatly increases the learning speed but has poor performance when facing complicated environment. In [33], an improved Q-learning is investigated, which can be considered as the future work of [32]. The main idea of this algorithm is the definition of locked state. When a state is locked, there will be no exploration in this state. This mechanism guarantees the fast convergence speed and good performance at the same time. Some of the researchers have extended RL to a more complicated environment such as 3-D environment, Aranibar *et al.* [34] use RL to solve path planning problem in 3-D environment, both Q-learning and neural Q-learning are applied in the greedy search algorithm. It should be noted that the reward function used in Q-learning is the same one used in [30]. Simulation results showed that Q-learning is more suitable for small and medium size of environment while the neural Q-learning works better on the large size of environment. Some algorithms also try to combine RL with other intelligent methods such as FLS and neural network (NN). A hybrid method combining RL and fuzzy logic is proposed in [35], the navigation system has two tasks, one is obstacle avoidance and the other is goal seeking. Compared to other RL algorithm, the major difference of this method is that two Q-learning algorithm is constructed independently. The fuzzy rules are designed with the assistant of RL algorithms. Another hybrid approach is proposed in [36], the designed navigation system has three main functions, the first one is obstacle avoidance, the second one is goal-seeking and the last one is supervising fuzzy logic system. The fuzzy supervisor mechanism generate the best action based on the information from obstacle avoidance module and goal-seeking module. For the problem of continuous action and state space, Yang *et al.* [37] proposed a method combining deep neural network and RL. A multiple layers neural network is utilized to estimate Q-values. This algorithm can generate path from start position to goal position in a continuous environment. It can be inferred that using RL as solution of path planning problem is becoming a tendency due to the excellent performance of Q-learning algorithm.

As the literatures indicates, most of the existing RL path planning algorithms are heavily relied on the pre-known information of the map. These algorithms cannot deal with unknown environment information because it is impossible to train a new Q-table at each time step. It is a quite challenging issue for RL algorithm if there are dynamic obstacles or unknown static obstacles in the map.

In this study, a multi-layer Q-learning algorithm is proposed as the solution of path planning problem. The proposed RL approach consists of two layers of Q-learning, one is the lower layer Q-learning and the other is the higher layer Q-learning. The lower layer Q-learning aims at avoiding static obstacles and leads the UAV approaching to the terminal position. The higher layer Q-learning deals with the dynamic obstacles. Specifically, for the lower layer, the flight environment is decomposed into grid cells, each cell unit refers to a state. The actions of UAV are restricted to five directions due to the turning constraints of UAV dynamics. The lower layer Q-learning is trained by exploring the static environment using a combination of Boltzmann and ϵ-greedy strategy. The Q-value $Q_{low}(s, a)$ will be updated with the assistant of the immediate reward function $r(s, a)$. For the higher layer, the state space has four elements. The first two elements are the locations of nearest obstacle and goal. The third element represents the moving direction of dynamic obstacle. The last element is the angle between target line and obstacle line. The Q-value $Q_{high}(s, a)$ is updated using the same exploration strategy of lower layer. After the two layers of Q-learning have been well trained, the planned optimal path will be generated by selecting the best action at each state from the initial point to the target point. Since the planned path is obtained by connecting the adjacent states, there must be hard turning angle in the planned path, which is impossible for the UAVs to follow. Therefore, the B-spline curve approach is applied at each state to generate a smooth path for UAVs.

The main contributions of this paper are listed as follows. 1) The design of multi-layers Q-learning enables the proposed algorithm to consider the global information and local information at the same time. Generally, the dynamic path planning methods only considers the surrounding environment information. Which makes the algorithm easily get trapped in the local optimal path. However, the proposed multi-layer Q-learning uses a combination Q-value to select the best action each time step, which comprehensively considers the global and local optimization.

2) The design of state space in higher layer, which can be also regarded as the modeling of environment, is totally different from the traditional environment model in Q-learning path planning. The traditional model uses the specific location to represents each state in the MDP model. This is accurate but cannot deal with dynamic obstacles. Inspired by the human behavior, this paper proposes a new definition of state space which does not contain the specific locations but uses the area concept, movement information and angular relationship to model the environment.

3) A combination of exploration strategy is proposed. Compared to a single strategy, the proposed one combines the ϵ-greedy and Boltzmann which can guarantee the diversity

of learning data at beginning and then accelerate learning process with training episode goes on.

The reminder of this paper is organized as follows. Preliminaries and assumptions are listed in the second section. Section 3 details the multi-layer path planning algorithm. Path smoothing is achieved by B-spline curve in Section 4. Numerical simulations are carried out in Section 5. Conclusions and summaries are outlined in Section 6.

II. PRELIMINARIES

A. MARKOV DECISION PROCESS MODEL

Generally, the Markov decision process (MDP) model is the basic model for RL algorithm, especially for path planning. The MDP for path planning is defined with four basic elements (S, A, P, R) Specifically, S stands for all possible states. A refers to the total possible actions. P represents the transition probability. R is the reward which evaluates the performance of a possible action at a specific state. A typical MDP model is shown in Fig. 1.

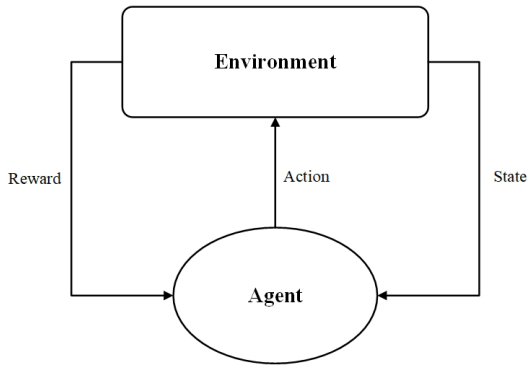

FIGURE 1. The MDP model.

During a time instant, s is the current state. Decision making mechanism may choose any action a which is available when the agent is at state s. After the execution of action a, the UAV will be at a new state s'. At the same time, it will receive the a reward $R(s, a)$. The reward R is defined as:

$$R(s, a) = \sum_{a \in A} R(s, a)p(s, a) \tag{1}$$

$Q(s, a)$ is defined as a reward value when the agent takes action a at state s. The values of $Q(s, a)$ can be generated using the Bellman equations:

$$Q(s, a) = R(s, a) + \gamma \sum_{a \in A} p(s, a) \sum_{a' \in A} Q(s', a') \tag{2}$$

B. ASSUMPTIONS

In this paper, the environment consists of two categories of obstacles, one is static obstacle and the other is dynamic obstacle. The location of static obstacles are fixed during the flight while the position of dynamic obstacles continuously changes at each time step. The goal of the proposed method is trying to plan an optimal path for the UAV to reach goal state in the environment within static and dynamic obstacles. Start point, terminal point and all non-dynamic obstacles are considered to be fixed in the path planning mission. It is also assumed that the UAV has all necessary sensors on-board to detect the dynamic obstacles within the sensor range. The following assumptions are outlined to clarify and narrow the path planning problem of this paper.

Assumption 1: The UAV position is known at each time step.

Assumption 2: The target point is known to UAV at each time step.

Assumption 3: The positions of the static obstacles are known at each time step.

Assumption 4: At each time step, the sensors can detect the position of the dynamic obstacle within the sensor range.

Assumption 5: The speed of UAV and dynamic obstacles remains constant.

Assumption 6: At each time step, the speed of UAV is greater than the speed of the dynamic obstacle.

III. MULTI-LAYER Q-LEARNING ALGORITHM

The proposed RL path planning algorithm consists of two layers. One is the lower layer, which is designed for the long-term strategy, considering the global information of the static obstacles. The other is the higher layer, which aims at the short-term strategy, considering the local information of the moving obstacles. The two layers are working in harmony to generate a collision-free path. Specific steps are as follows. At each time step, the on-board sensors are working to detect the moving obstacle. Once there is a moving obstacle detected in sensor range, the dynamic Q-learning of the higher layer will be activated. Defining the nearest moving obstacle position and the goal direction. After that, all the Q-values of higher layer at the current state are extracted. At the same time, all the Q-values of lower layer at the current state are also extracted. Comprehensively considering the Q-values of higher layer and lower layer, an action with the greatest Q-value will be generated and this is the optimal action at the current state. Repeat the process until the goal has been reached. Specific steps of the proposed multi-layer Q-learning algorithm is described in Fig. 2.

In order to use the RL algorithm in path planning problem, four fundamental parts should be defined. They are the environment model, state and action space, reward function and exploration strategy. The details will be described in the following for both static layer and dynamic layer.

A. LOWER LAYER Q-LEARNING

The lower layer of the proposed algorithm is constructed by a Q-learning algorithm which aims at learning the static environment.

1) ENVIRONMENT MODEL AND STATE SPACE

As shown in Fig. 3, a grid-based model, which is generated by cell decomposition, is proposed to illustrate the environment space in this paper. The method of cell decomposition is a common solution in the environment modeling in

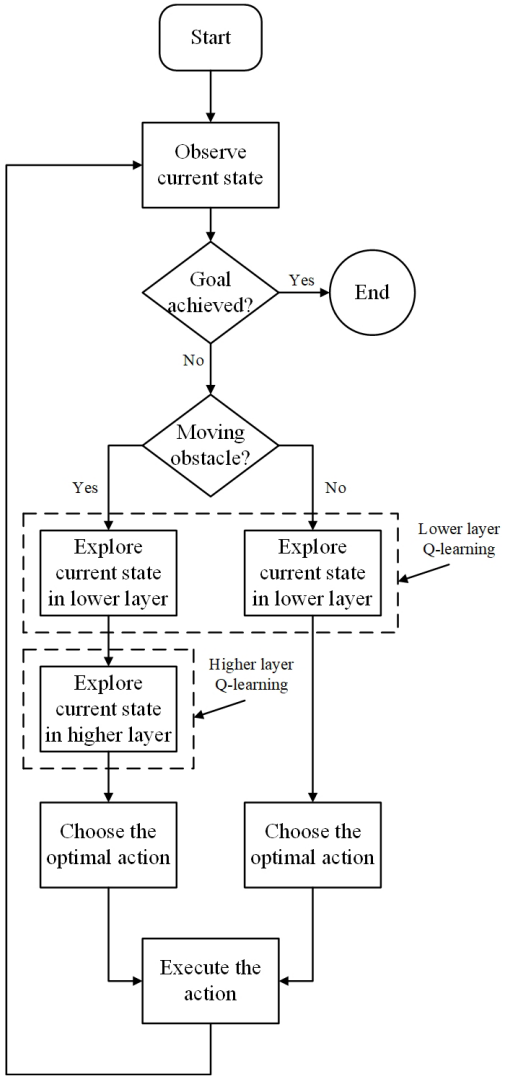

FIGURE 2. The flowchart of multi-layer Q-learning algorithm.

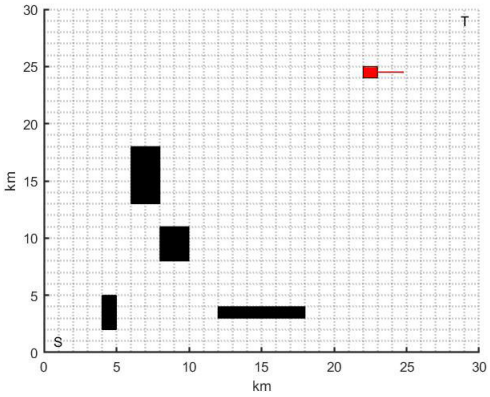

FIGURE 3. The environment model.

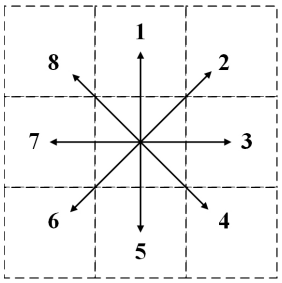

FIGURE 4. The 8 possible actions in the environment model.

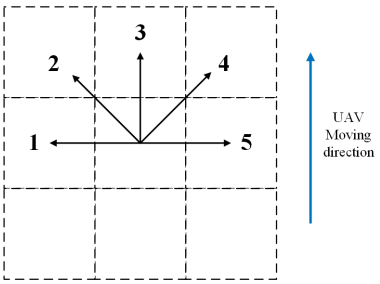

FIGURE 5. The 5 possible actions for UAV path planning.

path planning approaches and it is also the best model for RL algorithms. The environment contains both static and dynamic obstacles. It can be seen form Fig. 3, the black region represents the static obstacle while the area with red color stands for the dynamic obstacle, the red arrow is pointing the moving direction of dynamic obstacle.

The state space is defined as the each cell grid in the environment model. The state is illustrated in two dimensions, one is the coordinate of x-axis and the other is the coordinate of y-axis.

2) ACTION SPACE

In the environment model as shown in Fig. 3, there are eight adjacent states of one state. Therefore, the possible actions contains eight directions, as shown in Fig. 4.

However, for the fixed-wing UAV, it is impossible to move backwards due to the turning constraints. As a result, the eight possible actions will be restricted to 5 of them. They are left, forward-left, forward, forward-right and right, as shown in Fig. 5.

3) REWARD FUNCTION

The reward function of a RL algorithm is used to give reward or punishment while UAV takes an action. Reward function is considered to be the most important factor of a RL algorithm because it directly determines the effectiveness and efficiency of the algorithm. In the proposed lower layer Q-learning, the UAV takes an action and finally arrive at next state, then the UAV will receive a reward. The goal of the UAV is to find an optimal path which has the greatest reward. The reward function of the proposed lower layer Q-learning is

$$r_{low}(s, a) = \alpha_{low}(d_{s-1} - d_s) - \beta(\sum_{i=1}^{n} \frac{1}{ob_i}) \tag{3}$$

where d_s and d_{s-1} are the distance to the terminal point of current position and previous position. ob_i refers to the distance between the ith obstacle and current position. n stands for the number of obstacles. α_{low} and β are the parameters to balance the importance of the two terms.

4) EXPLORATION STRATEGY

The exploration strategy is an important mechanism in the learning efficiency of the proposed lower layer Q-learning algorithm. The most widely used approaches are ϵ-greedy and Boltzmann strategy. In this study, the lower layer Q-learning uses a combination of Boltzmann and ϵ-greedy exploration strategy. Compared to using a single strategy, the advantage of the proposed strategy is that it can avoid to be trapped in local optimal value and accelerate the convergence speed. In addition, it can also guarantee the diversity of learning data at beginning and then accelerate learning process with training episode goes on.

The specific steps of the exploration strategy is shown as follows. At first step, generate a random value p_{flag}, $p_{flag} \in (0, 1)$. Then, comparing this value p_{flag} to the value of ϵ. If p_{flag} is smaller than ϵ, the action is randomly selected in A. If p_{flag} is greater than or equal to ϵ, the action is selected by the Boltzmann law. In Boltzmann exploration strategy, each action is assigned a probability. The probability of selecting the possible action a_i at state s_t is defined as $p(s_t, a_i)$,

$$p(s_t, a_i) = \frac{\frac{Q(s_t, a_i)}{T_k}}{\sum_{a_j \in A} \frac{Q(s_t, a_j)}{T_k}} \tag{4}$$

where T_k is the temporary parameter, this parameter will decrease with the number of iteration increases.

$$T_k = \lambda^k T_0 \tag{5}$$

where λ is the discount factor and T_0 refers to the initial value of temporary parameter.

5) UPDATING Q-TABLE

During the execution of the path planning problem, the UAV is searching for the optimal action starting from initial position to the target position. During this process, the UAV is trying to find an action at each time step which has the maximum cumulative reward. The Q-value is updated by state-action pair, which is shown as follows,

$$\begin{aligned} Q_{low}(s, a) = {} & Q_{low}(s, a) + \alpha_{learning} r_{low}(s, a) \\ & + \alpha_{learning}(\gamma \max_{a'} Q_{low}(s', a') - Q_{low}(s, a)) \end{aligned} \tag{6}$$

where $\alpha_{learning}$ is the learning efficiency ranged from 0 to 1. $\gamma (0 \leq \gamma \leq 1)$ refers to the discount rate. $r_{low}(s, a)$ stands for the immediate reward which is generated by taking the action a at state s.

B. HIGHER LAYER Q-LEARNING

The higher layer of the proposed algorithm is designed for local path planning, which aims at avoiding the dynamic obstacle. Let us consider how a human tries to avoid moving obstacles. Human will focus on the surrounding environment and ignore the environment far away until it comes near. And human will never go too close to the obstacle, instead, he or she will move away from the obstacle in distance. In other words, human cares more about the direction and approximated distance of the obstacle and care less about the specific location or specific distance of the obstacle. The proposed algorithm is motivated by this idea.

1) ENVIRONMENT MODEL AND STATE SPACE

The environment model of higher layer Q-learning is quite different from the one of lower layer Q-learning. The reason is that the higher layer Q-learning deals with the local environment which contains dynamic obstacles. Compared to the lower layer, the major difference of the environment model used in higher layer Q-learning is that the environment is a local environment, it only considers the states within the sensor range. At each time step, the UAV is located in the center of the sensor area. In this paper, the radius of the sensor range is considered as the length of 3 cell units. As shown in Fig. 6, the sensor range consists of 29 cell units.

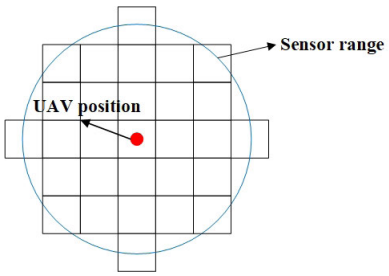

FIGURE 6. The sensor range of UAV.

However, since the position of dynamic obstacle is changing rapidly, the fixed location could not be used as the system state. Aiming at presenting the full status of moving obstacle, the state space of higher layer Q-learning is defined as follows.

The first element in the state space is the obstacle location. The sensor area will be divided into 8 small regions due to the eight possible directions of action. The 8 small regions are labeled as $R1, R2, \ldots, R8$ shown in Fig. 7.

The second element in the state space is the moving direction of dynamic obstacle. There are only four moving directions of the dynamic obstacle. They are upward, rightward, downward and leftward, which are labeled as $D1, D2, D3$ and $D4$.

The third element in the state space is the location of terminal point. Extend the lines in Fig, 7, the entire environment will be divided into eight regions: they are $R1, R2, \ldots, R8$, as illustrated in Fig. 7. The terminal point will be located in one of the 8 regions.

However, the above state definition is not accurate enough to illustrate the surrounding environment. For example, as shown in Fig. 8, the obstacle and the terminal are located in the same region. Under this circumstance, it is impossible for the UAV to recognize the surrounding environment and then make the best action. As a result, the angle α, which is the included angle of the two lines: obstacle line and target line, should be added in the state space. The obstacle line refers to the line connect the center of the closest obstacle and UAV

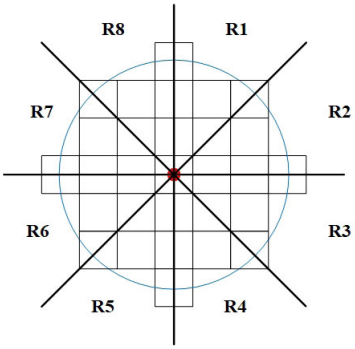

FIGURE 7. The 8 sub-regions of the sensor range.

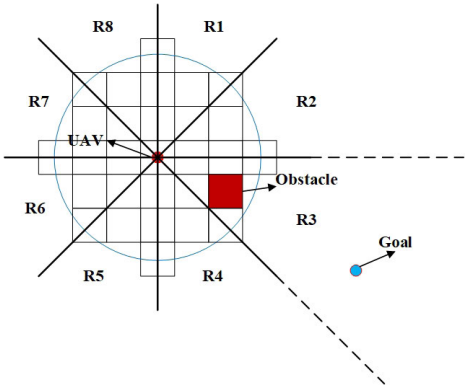

FIGURE 8. The obstacle and goal are located in the same sub-region.

while the target line stands for the line connecting the terminal point and UAV. The definition of angle α and two lines are shown in Fig. 9. The value of angle α is calculated as follows. At each time step, the UAV location is $P_{uav} = [x_u, y_u]$, the terminal point is located at $P_{terminal} = [x_t, y_t]$, and the position of the obstacle is $P_{obstacle} = [x_o, y_o]$. Therefore, the distance from the UAV current location to the terminal point can be generated by

$$d_{u\to t} = \sqrt{(x_u - x_t)^2 + (y_u - y_t)^2} \tag{7}$$

The distance between obstacle and UAV can be given by

$$d_{u\to o} = \sqrt{(x_u - x_o)^2 + (y_u - y_o)^2} \tag{8}$$

The equation of the target line can be defined as

$$y = kx + b \tag{9}$$

where k and b are the parameters calculated by

$$\begin{aligned} k &= \frac{y_t - y_u}{x_t - x_u} \\ b &= y_u - kx_u \end{aligned} \tag{10}$$

The distance between the target line and obstacle position can be calculated by

$$d_{o\to line} = \frac{|x_o - ky_u - b|}{\sqrt{k^2 + b^2}} \tag{11}$$

Based on (8) and (11), the angle α is obtained by

$$\alpha = sin^{-1}(\frac{d_{o\to line}}{d_{u\to o}}) \tag{12}$$

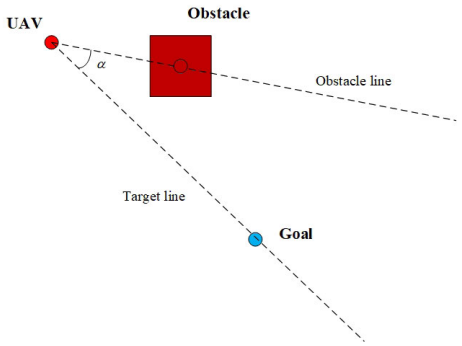

FIGURE 9. The definition of obstacle line and target line.

The angle α is ranged from 0 to π, $\alpha \in [0, \pi]$. Since the two distances $d_{o\to line}$ and $d_{u\to o}$ is positive, so the ratio of two distances $\frac{d_{o\to line}}{d_{u\to o}}$ is also positive. As a result, the value of α is from 0 to π. Directly adding the value of angle α to the state space will lead to an infinite number of states, which is impossible for the UAV to learn the optimal policy. Aiming at decreasing the total number of states, the value range $[0, \pi]$ is divided into 8 sub-regions.

$$A_{t\to o} = \begin{cases} a_1, & 0 \le \alpha < \frac{1}{8}\pi \\ a_2, & \frac{1}{8}\pi \le \alpha < \frac{1}{4}\pi \\ a_3, & \frac{1}{4}\pi \le \alpha < \frac{3}{8}\pi \\ a_4, & \frac{3}{8} \le \alpha < \frac{1}{2}\pi \\ a_5, & \frac{1}{2}\pi \le \alpha < \frac{5}{8} \\ a_6, & \frac{5}{8}\pi \le \alpha < \frac{3}{4}\pi \\ a_7, & \frac{3}{4}\pi \le \alpha < \frac{7}{8}\pi \\ a_8, & \frac{7}{8}\pi \le \alpha < \pi \end{cases} \tag{13}$$

Therefore, the fourth element in the state space is $A_{t\to o}$. Using this state definition will clearly illustrate the surrounding environment of UAV. For example, the surrounding environment is shown in Fig. 14, the state space at time step t will be

$$S = [R_o, D_o, R_t, A_{t\to o}] \tag{14}$$

where R_t is the region where the goal located and R_o stands for the region of obstacle. $A_{t\to o}$ refers to the angle value between terminal line and obstacle line.

2) ACTION SPACE

After the state space has been defined, the possible actions at each state should be addressed. Unlike other autonomous moving robot, UAV has turning constraints and cannot move backward. As a result, the UAV can only execute 5 possible actions in the cell decomposition environment (as shown in Fig. 5), they are left, forward-left, forward, forward-right and right, which are labeled as A_1, A_2, A_3, A_4 and A_5.

3) REWARD FUNCTION

Reward function can be regarded as an immediate feedback from the environment while UAV takes an action. It is an evaluation indicates the performance of the action at current state.

$$r_{high}(s,a) = \beta_1(d_{u\to t}(s-1) - d_{u\to t}(s)) + e^{\beta_2(d_{u\to o}(s-1)-d_{u\to o}(s))} + \beta_3 \sin\alpha \quad (15)$$

It should be noted that the exploration strategy used in the higher layer Q-learning is the same exploration strategy used in lower layer Q-learning.

4) UPDATING Q-TABLE

In the high layer Q-learning algorithm, the Q-value is quite different from the lower layer Q-learning because it has two dimensions of states. One state is the nearest obstacle position and the other state is the goal direction. The initial value of Q-table is set to be zero. With the training phase, Q-table will be full-filled with Q-values updated by the following equation.

$$Q_{high}(s,a) = r_{high}(s,a) + \gamma \mathrm{Max}(Q_{high}(s_{t+1}, a_{t+1})) \quad (16)$$

C. TRAINING THE ALGORITHM

Since the lower layer and higher layer Q-learning are working separately. The training process of the proposed algorithm is divided into two independent sections. One is training the lower layer Q-learning algorithm and the other is training the higher layer Q-learning algorithm. Both training processes will be realized by exploring the map and then reach the goal position. Every success trial will be considered as an episode.

In the proposed multi-layer Q-learning algorithm, the lower layer Q-learning focuses on the avoidance of static obstacles. In each training episode, the goal position and the static obstacles remain the same. Generally, the more training episode has been executed, the better performance the algorithm will have. However, considering the training time, there will be a balance between the algorithm performance and total training time. In this paper, there are two end conditions for the training process. One is the maximum limit of total episode, when the training process reach a specific number of episode, the training process is end. The second end condition is the changes between total accumulated discount reward. If the accumulated reward does not change within 10 episodes, the training process is end. The structure of the lower layer Q-learning training process is shown in Fig. 10 and the training process of higher layer is illustrated in Fig. 11. From the two flowcharts, we can see the training will repeat until end condition has been achieved.

D. PLANNING THE PATH

After both the lower layer and higher layer Q-learning algorithm have been trained, the collision-free path can be generated by the updated Q-table. Specific steps are shown in Algorithm 1.

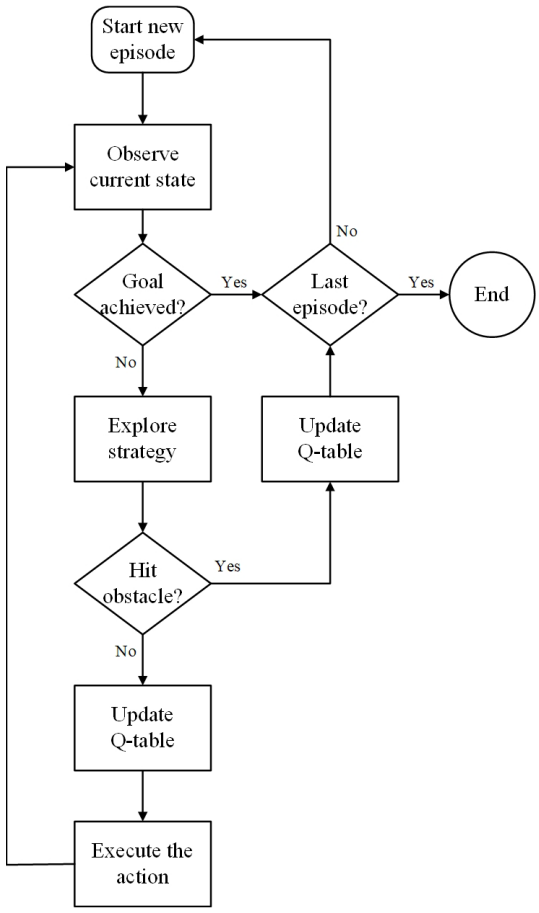

FIGURE 10. The Flowchart of training lower layer Q-learning algorithm.

IV. PATH SMOOTHING

As shown in Fig. 12, the planned path is constructed by connecting the cell units sequentially. However, this path is not feasible for the UAVs to follow because it has sharp turning angles. Therefore, the B-spline path smoother is introduced in this paper to produce a smooth path for UAV to follow as shown in Fig. 13. The cubic B-spline method is the best solution for path smoothing in this paper because it can comprehensively consider the smoothness of path and the computational complexity [38]. In addition, the B-spline modeling aims at providing local control of the curves [39]. Adding or changing a waypoint only changes the local curve and will not change the previous curve, which is perfect match for the on-line path planning algorithm. In other words, you can do path smoothing at anytime without knowing the all the control points.

The B-spline curve is defined as

$$P(t) = \sum_{i=1}^{n+1} B_i N_{i,k}(t), 0 \le t \le t_{max} \quad (17)$$

where B_i is the ith control point, $P(t)$ refers to the $(k-2)$ times continuously differentiable, $N_{i,k}(t)$ stands for the normalized B-spline basis function, which is defined by the follows

$$N_{i,k}(t) = \frac{(t-x_i)N_{i,k}(t)}{x_{i+k-1}-x_i} + \frac{(x_{i+k}-t)N_{i+1,k-1}(t)}{x_{i+k}-x_{i+1}} \quad (18)$$

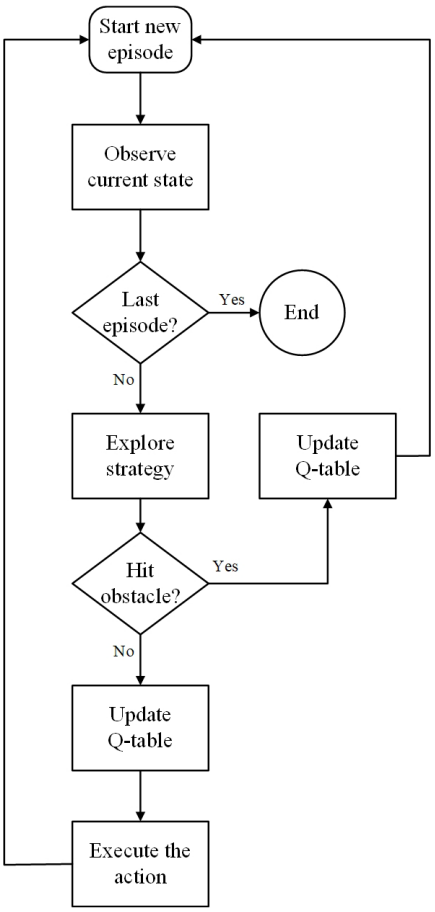

FIGURE 11. The Flowchart of training higher layer Q-learning algorithm.

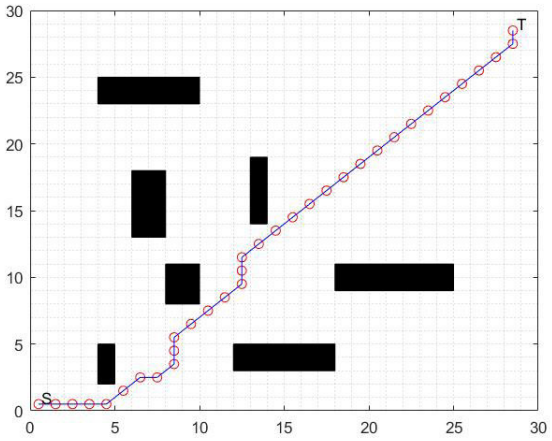

FIGURE 12. The planned path contains hard turning angles.

where $t \leq n-k+2$ and x_i is knot value. These knot values are monotonically increasing, which means the next knot value must be greater than the previous knot values.

$$x_i = \begin{cases} 0, & 0 \leq i < k \\ i-k+1, & k \leq i \leq n \\ n-k+2, & n < i \leq n+k \end{cases} \tag{19}$$

Algorithm 1 Path Planning Algorithm

Input:Initial state S and goal state T
Q-table of lower layer Q_{low} and higher layer Q_{high}
Initialize path sequence $P = S$
Environment Information
Planning process
Set the UAV current state as the initial state: $S_{current} = S$
While $(S_{current} \neq T)$
If(No moving obstacles in sensor range)
Find the best action at current state using Q_{low}
$Q(S_{current}, a_{max}) = \text{Max}[Q_{low}(S_{current}, a)]$
Execute the best action and obtain the next state
$S_{move} = Execute(S_{current}, a_{max})$
End If
If(There is moving obstacle in sensor range)
Find the best action at current state using Q_{low} and Q_{high}
$Q(S_{current}, a_{max}) = \text{Max}[Q_{low}(S_{current}, a) + Q_{high}(S_{current}, a)]$
Execute the best action and obtain the next state
$S_{move} = Execute(S_{current}, a_{max})$
End If
Set the current state as next state
$S_{current} = S_{move}$
Store the movement to the path sequence
$P = [P, S_{move}]$
End While
Output:The path sequence P

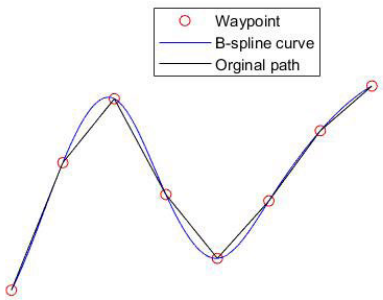

FIGURE 13. The B-spline curve with 8 control points.

It should be noted that in special cases when $k = 1$, the value of $N_{i,1}(t)$ should be defined as

$$N_{i,1}(t) = \begin{cases} 1, & x_i \leq t \leq x_{i+1} \\ 0, & \text{otherwise} \end{cases} \tag{20}$$

V. SIMULATION RESULTS

The numerical simulation will be carried out in the Matlab software to validate the proposed method. An environment of 30×30 cell grids is considered. S and T denote start point and the terminal point and they are fixed during the following simulation. The UAV remains a constant speed while the speed of dynamic obstacles is half of the UAV speed.

As it can be seen from Fig. 14, the coordinate of the start point is [1, 1], the coordinate of the terminal position

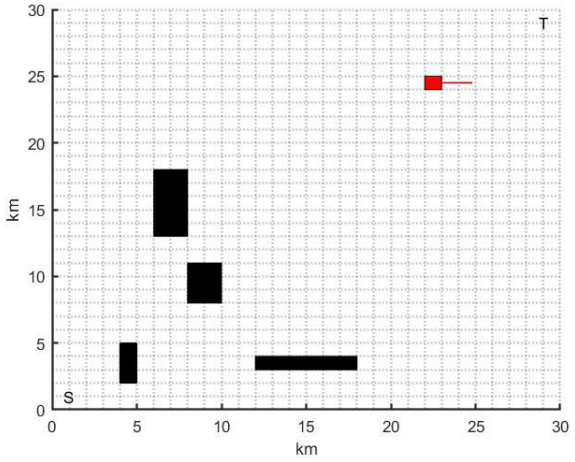

FIGURE 14. The simulation environment.

is [29, 29]. Seven static obstacles and one moving obstacle block the way from start to terminal. The static obstacles can be presented by a black square-shaped area. The obstacle location can be determined by the location of four vertices of the rectangular area. The specific coordinates of the four vertices are listed as follows.

$$\begin{aligned} Obstacle_1 &= [(8, 8), (10, 8), (10, 11), (8, 11)] \\ Obstacle_2 &= [(4, 2), (5, 2), (5, 5), (4, 5)] \\ Obstacle_3 &= [(6, 13), (8, 13), (8, 18), (6, 18)] \\ Obstacle_4 &= [(12, 3), (12, 5), (18, 5), (18, 3)] \\ Obstacle_5 &= [(13, 14), (14, 14), (14, 19), (13, 19)] \\ Obstacle_6 &= [(4, 23), (4, 23), (10, 25), (10, 25)] \\ Obstacle_7 &= [(18, 9), (18, 11), (25, 11), (25, 9)] \end{aligned} \quad (21)$$

The red square-shaped area refers to the moving obstacle, the red arrow shows its moving direction.

A. TRAINING SETTINGS

In this paper, the two layers of Q-learning are trained independently.

1) LOWER LAYER Q-LEARNING

The training environment for lower layer Q-learning remains the same during the training process. As shown in Fig. 14, there are 7 static obstacles in the environment which block the way from starting position to the terminal position. The goal of the training process is to explore this environment and updated Q-table based on (3). In the training process, the specific steps of an episode are illustrated as follows First, the UAV starts from the initial position. Second, the UAV observes the environment, collects information. Third, the exploration strategy will generate an action. Fourth, the UAV follows this action will receive an immediate reward. Fifth, the Q-table will be updated. Repeat step 1 to step 5 until the UAV has reached the goal position or the UAV fails the mission. Two conditions will be considered as a failure in the training phase. One is the UAV hits the obstacle. The other is the total number of iterations exceeds the maximum value 1000. Due to the random exploration strategy and uncompleted Q-table, it is easy for the UAV to be trapped in the obstacles. Therefore, a maximum number of iterations within an episode is necessary to prevent the endless loop.

The maximum episode for training is 1000.

TABLE 1. Training effect on successful rate.

Number of training episodes	Number of successful episodes	Successful rate
100	65	65%
200	142	71%
300	231	77%
400	325	81.3%
500	422	84.4%
600	518	86.3%
700	616	88%
800	715	89.4%

2) HIGHER LAYER Q-LEARNING

The training environment for higher layer Q-learning is quite different compared to the one of lower layer Q-learning. Only the moving obstacles are used in the dynamic environment. Training in the higher layer Q-learning is completed by exploring a series of dynamic environments. Each dynamic environment is considered as an episode. In each dynamic environment, the number of moving obstacles are different. Also, the locations and directions of the moving obstacles are chosen randomly. It should be noted that the directions of moving obstacles are restricted into 8 directions, which are the same directions shown in Fig. 4. The direction of a moving obstacle will not change within the episode.

B. TRAINING EFFECT

Success rate or miss rate is considered to be an important evaluation factor of the training effect in learning algorithm design. During training phase, if the UAV hits obstacle or the UAV cannot reach the terminal point within the maximum iterations, this episode is considered to be unsuccessful. Generally, for all the learning algorithm, it is expected that with more training episodes, the performance will be better and the successful rate is greater. The successful rate of the training process is sampled every 100 episodes. During the training phase, the number of successful episode is counted and then successful rate can be calculated. The training effect on successful rate is presented in Table 1 and Fig. 15. In Table 1, the successful rate increases as the episodes goes up. When the training just started, there is only 100 training episodes, the successful rate is 65%. After that, the algorithm has been trained hundreds of times until 800 training episodes was reached, where the successful rate is increased to almost 90%, which is considered to be a high successful rate.

Fig. 15 shows the successful rate of every 100 episodes. The successful rate increases fast at the beginning and maintains in a high level after 500 episodes.

Another important evaluation factor of the training effect is the algorithm performance. In this paper, the objective of the path planning algorithm is to find a shortest path which

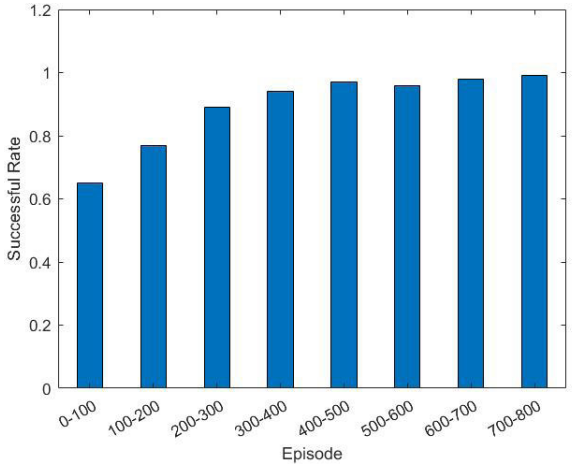

FIGURE 15. The successful rate of every 100 episodes.

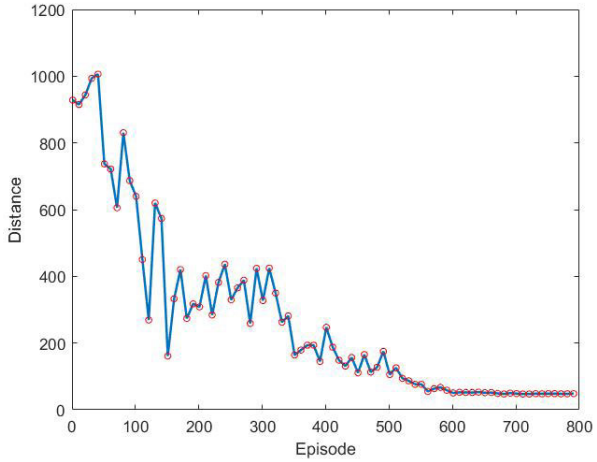

FIGURE 16. The distance of the planned path at different training episodes (with moving window of every 10 episodes).

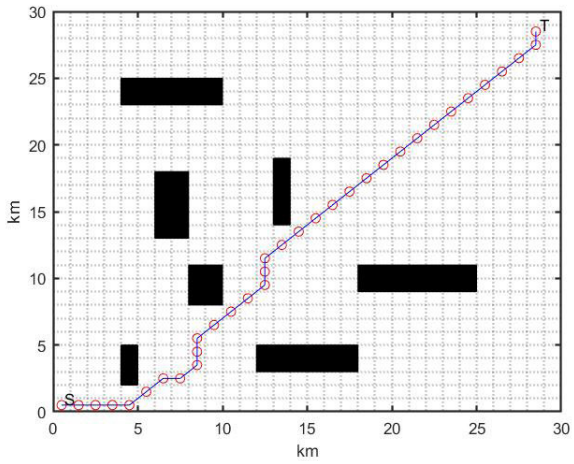

FIGURE 17. The planned path facing static obstacles.

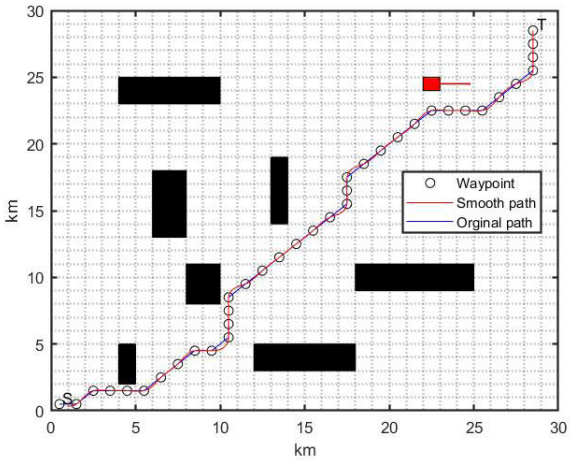

FIGURE 18. The planned path when facing seven static obstacles and one moving obstacle.

can avoid both the static and dynamic obstacles. Therefore, the performance evaluation can be simply considered as the total length of the planned path. Fig. 16 shows the length of the planned path in different stages of the training process. At the beginning, the total length is extremely high because the environment is unknown to the UAV. The algorithm needs to explore a wide range of states to find the best path. As the training episodes increase, the total length of planned path is decreasing quickly and finally converges.

C. TESTING PERFORMANCES

After the proposed algorithm has been well trained, the testing phase will be carried out under more complicated environment in several testing scenarios.

1) SCENARIO 1

The first testing scenario is to validate the algorithm performance of avoiding static obstacles. Since there is no dynamic obstacles, the result of testing scenario 1 shows the effectiveness of lower layer Q-learning.

In Fig. 17, the UAV successfully reaches the terminal position with a collision-free path. The black circle stands for the waypoints, the blue lines is the path planned by multi-layer Q-learning algorithm, the red line refers to the smooth path generated by the B-spline curve approach.

2) SCENARIO 2

The second testing scenario is shown in Fig. 18. The static obstacles remain the same. Besides the seven static obstacles, there is a dynamic obstacle with shuttle movement in the left and right direction. Starting from (23, 25) to (25, 25).

From this figure, it is obviously that the UAV successfully avoid crashing on the moving obstacle and finally reaches the terminal point.

3) SCENARIO 3

The environment of the third scenario 3 is shown in Fig. 19. The static obstacles remain the same. It has two dynamic obstacles, one is moving in the left and right direction and the other is moving in the up and down direction. In addition, the second dynamic obstacle is hiding in the static obstacles, which makes the environment more complicated to recognize and plan collision-free path.

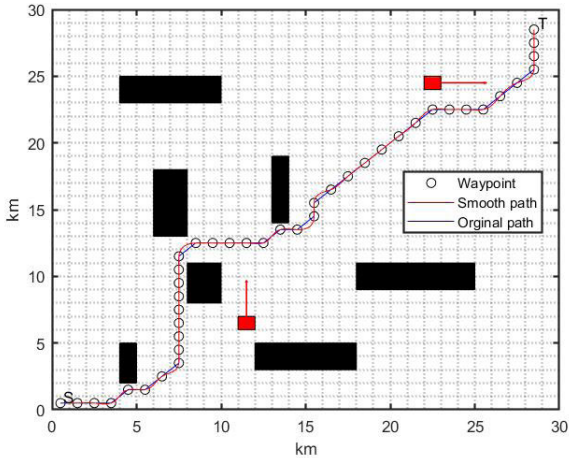

FIGURE 19. The planned path when facing seven static obstacles and two moving obstacles.

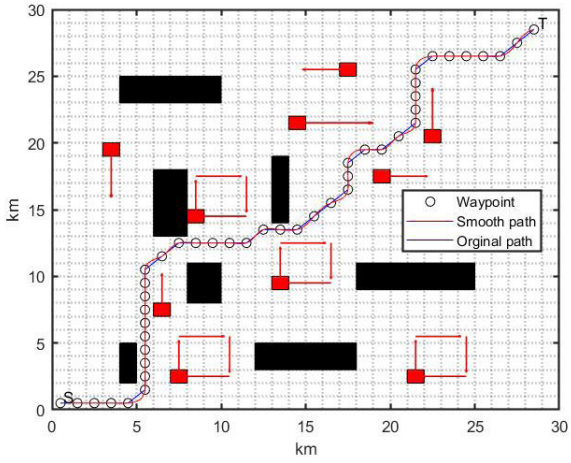

FIGURE 20. The planned path when facing seven static obstacles and ten moving obstacles.

D. SCENARIO 4

The above three testing scenarios have validated the effectiveness of the proposed algorithm, however, the environment is comparative basic since it has only one or two moving obstacles. The environment of the third scenario 4 is shown in Fig. 20. The static obstacles remain the same. It consists of 10 moving obstacles. Among them, four obstacles are moving in a rectangular-shaped path. The red arrow shows the moving direction of them. This testing environment is complicated and the UAV has to fly carefully to find a collision-free path.

E. SCENARIO 5

The environment of scenario 5 only consists of moving obstacles. However, they are extremely densely distributed as shown in Fig. 21. This is quite challenging for the real-time planning ability of the proposed algorithm.

The total distances of the planned path in different scenarios are shown in Fig. 22. For the testing environment with $30km \times 30km$, the ideal optimal path length will be $39.59km$, which is directly connecting the start point and the target point. Among the five testing scenarios, the shortest path length is 42.53 of scenario 1. The longest path length is 51.21 of scenario 5. It can be concluded from the figure that the optimal path will be longer when facing more moving obstacles.

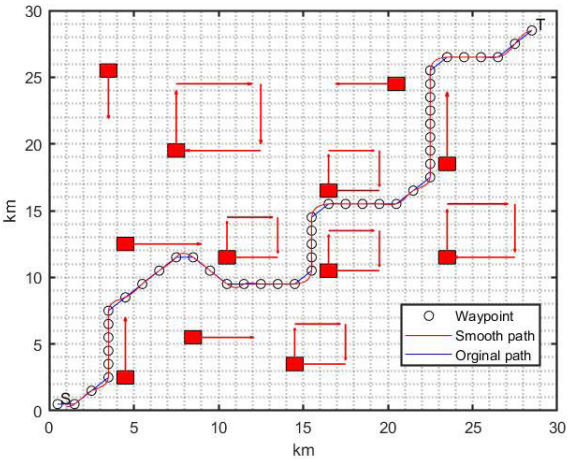

FIGURE 21. The planned path when facing twelve moving obstacles.

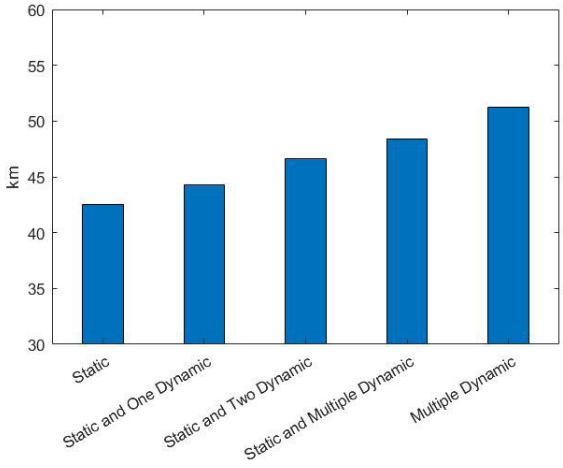

FIGURE 22. The path length of different scenarios.

VI. CONCLUSION

In this paper, a new multi-layer Q-learning approach for UAV path planning problem is proposed. This new algorithm has two layers of Q-learning. One is lower layer Q-learning, which collects the information globally. It is used for global planning. The other one is higher layer Q-learning, which focuses on the local information. This one is considered as local planning. The performance of the planned path is improved by B-spline method for path smoothing. Different testing scenarios have been carried out to validate the effectiveness of the proposed method when facing different types of environment.

REFERENCES

[1] S. Li, X. Wang, L. Hu, and Y. Liu, "Mobile robot path planning based on Q-learning algorithm," in *Proc. WRC Symp. Adv. Robot. Automat. (WRC SARA)*, Aug. 2019, pp. 160–165.

[2] J. Tang, "Conflict detection and resolution for civil aviation: A literature survey," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 34, no. 10, pp. 20–35, Oct. 2019.
[3] S. Li, X. Xu, and L. Zuo, "Dynamic path planning of a mobile robot with improved Q-learning algorithm," in *Proc. IEEE Int. Conf. Inf. Automat.*, Aug. 2015, pp. 409–414.
[4] O. Takahashi and R. J. Schilling, "Motion planning in a plane using generalized Voronoi diagrams," *IEEE Trans. Robot. Autom.*, vol. 5, no. 2, pp. 143–150, Apr. 1989.
[5] P. Bhattacharya and M. L. Gavrilova, "Voronoi diagram in optimal path planning," in *Proc. 4th Int. Symp. Voronoi Diagrams Sci. Eng. (ISVD)*, Jul. 2007, pp. 38–47.
[6] F. Benavides, G. Tejera, M. Pedemonte, and S. Casella, "Real path planning based on genetic algorithm and Voronoi diagrams," in *Proc. IEEE 9th Latin Amer. Robot. Symp. IEEE Colombian Conf. Autom. Control*, Oct. 2011, pp. 1–6.
[7] C. Cai and S. Ferrari, "Information-driven sensor path planning by approximate cell decomposition," *IEEE Trans. Syst., Man, Cybern., B, Cybern.*, vol. 39, no. 3, pp. 672–689, Jun. 2009.
[8] M. Kloetzer, C. Mahulea, and R. Gonzalez, "Optimizing cell decomposition path planning for mobile robots using different metrics," in *Proc. 19th Int. Conf. Syst. Theory, Control Comput. (ICSTCC)*, Oct. 2015, pp. 565–570.
[9] F. Lingelbach, "Path planning using probabilistic cell decomposition," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, vol. 1, Apr./May 2004, pp. 467–472.
[10] F. Zhu and J. Tang, "Graphical composite modeling and simulation for multi-aircraft collision avoidance," *Softw. Syst. Model.*, pp. 1–15, Nov. 2020.
[11] A. K. Pamosoaji and K.-S. Hong, "A path-planning algorithm using vector potential functions in triangular regions," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 43, no. 4, pp. 832–842, Jul. 2013.
[12] P. Vadakkepat, K. C. Tan, and W. Ming-Liang, "Evolutionary artificial potential fields and their application in real time robot path planning," in *Proc. Congr. Evol. Comput.*, vol. 1, Jul. 2000, pp. 256–263.
[13] Y.-B. Chen, G.-C. Luo, Y.-S. Mei, J.-Q. Yu, and X.-L. Su, "UAV path planning using artificial potential field method updated by optimal control theory," *Int. J. Syst. Sci.*, vol. 47, no. 6, pp. 1407–1420, Apr. 2016.
[14] J. Sun, J. Tang, and S. Lao, "Collision avoidance for cooperative UAVs with optimized artificial potential field algorithm," *IEEE Access*, vol. 5, pp. 18382–18390, 2017.
[15] J. Tang, J. Sun, C. Lu, and S. Lao, "Optimized artificial potential field algorithm to multi-unmanned aerial vehicle coordinated trajectory planning and collision avoidance in three-dimensional environment," *Proc. Inst. Mech. Eng., G, J. Aerosp. Eng.*, vol. 233, no. 16, pp. 6032–6043, Dec. 2019.
[16] Y. Zhang, D.-W. Gong, and J.-H. Zhang, "Robot path planning in uncertain environment using multi-objective particle swarm optimization," *Neurocomputing*, vol. 103, pp. 172–185, Mar. 2013.
[17] V. Roberge, M. Tarbouchi, and G. Labonte, "Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning," *IEEE Trans. Ind. Informat.*, vol. 9, no. 1, pp. 132–141, Feb. 2013.
[18] L. Guangshun and S. Hongbo, "Path planning for mobile robot based on particle swarm optimization," in *Proc. Chin. Control Decis. Conf.*, Jul. 2008, pp. 3290–3294.
[19] B. Sun, D. Zhu, L. Jiang, and S. X. Yang, "A novel fuzzy control algorithm for three-dimensional AUV path planning based on sonar model," *J. Intell. Fuzzy Syst.*, vol. 26, no. 6, pp. 2913–2926, 2014.
[20] A. Pandey, R. K. Sonkar, K. K. Pandey, and D. R. Parhi, "Path planning navigation of mobile robot with obstacles avoidance using fuzzy logic controller," in *Proc. IEEE 8th Int. Conf. Intell. Syst. Control (ISCO)*, Jan. 2014, pp. 39–41.
[21] Q. Song, Q. Zhao, S. Wang, Q. Liu, and X. Chen, "Dynamic path planning for unmanned vehicles based on fuzzy logic and improved ant colony optimization," *IEEE Access*, vol. 8, pp. 62107–62115, 2020.
[22] I. Châari, A. Koubâa, S. Trigui, H. Bennaceur, A. Ammar, and K. Al-Shalfan, "SmartPATH: An efficient hybrid ACO-GA algorithm for solving the global path planning problem of mobile robots," *Int. J. Adv. Robot. Syst.*, vol. 11, no. 7, p. 94, Jul. 2014.
[23] M. Brand, M. Masuda, N. Wehner, and X.-H. Yu, "Ant colony optimization algorithm for robot path planning," in *Proc. Int. Conf. Comput. Design Appl.*, vol. 3, Jun. 2010, pp. V3-436–V3-440.
[24] J. Liu, J. Yang, H. Liu, X. Tian, and M. Gao, "An improved ant colony algorithm for robot path planning," *Soft Comput.*, vol. 21, no. 19, pp. 5829–5839, 2017.
[25] S. Y. Choi and D. Cha, "Unmanned aerial vehicles using machine learning for autonomous flight; state-of-the-art," *Adv. Robot.*, vol. 33, no. 6, pp. 265–277, Mar. 2019.
[26] A. I. Khan and Y. Al-Mulla, "Unmanned aerial vehicle in the machine learning environment," *Procedia Comput. Sci.*, vol. 160, pp. 46–53, Jan. 2019.
[27] B. Zhang, Z. Mao, W. Liu, and J. Liu, "Geometric reinforcement learning for path planning of UAVs," *J. Intell. Robot. Syst.*, vol. 77, no. 2, pp. 391–409, Feb. 2015.
[28] B. Kim and J. Pineau, "Socially adaptive path planning in human environments using inverse reinforcement learning," *Int. J. Social Robot.*, vol. 8, no. 1, pp. 51–66, Jan. 2016.
[29] C. J. C. H. Watkins, "Learning from delayed rewards," Ph.D. dissertation, King's College, London, U.K., 1989.
[30] W. D. Smart and L. P. Kaelbling, "Practical reinforcement learning in continuous spaces," in *Proc. ICML*, Jun. 2000, pp. 903–910.
[31] W. D. Smart and L. P. Kaelbling, "Effective reinforcement learning for mobile robots," in *Proc. IEEE Int. Conf. Robot. Automat.*, vol. 4, May 2002, pp. 3404–3410.
[32] I. Goswami, P. K. Das, A. Konar, and R. Janarthanan, "Extended Q-learning algorithm for path-planning of a mobile robot," in *Proc. Asia–Pacific Conf. Simulated Evol. Learn.* Berlin, Germany: Springer, Dec. 2010, pp. 379–383.
[33] A. Konar, I. G. Chakraborty, S. J. Singh, L. C. Jain, and A. K. Nagar, "A deterministic improved Q-learning for path planning of a mobile robot," *IEEE Trans. Syst., Man, Cybern., Syst.* vol. 43, no. 5, pp. 1141–1153, Sep. 2013.
[34] D. B. Aranibar and P. J. Alsina, "Reinforcement learning-based path planning for autonomous robots," in *Proc. EnRI-24th Congresso da Sociedade Brasileira de Computaç ao*, vol. 10, 2004, pp. 1–10.
[35] H. R. Beom and H. S. Cho, "A sensor-based navigation for a mobile robot using fuzzy logic and reinforcement learning," *IEEE Trans. Syst., Man, Cybern.*, vol. 25, no. 3, pp. 464–477, Mar. 1995.
[36] N. H. C. Yung and C. Ye, "Self-learning fuzzy navigation of mobile vehicle," in *Proc. 3rd Int. Conf. Signal Process. (ICSP)*, Oct. 1996, pp. 1465–1468.
[37] G.-S. Yang, E.-K. Chen, and C.-W. An, "Mobile robot navigation using neural Q-learning," in *Proc. Int. Conf. Mach. Learn. Cybern.*, vol. 1, Aug. 2004, pp. 48–52.
[38] C.-C. Tsai, H.-C. Huang, and C.-K. Chan, "Parallel elite genetic algorithm and its application to global path planning for autonomous robot navigation," *IEEE Trans. Ind. Electron.*, vol. 58, no. 10, pp. 4813–4821, Oct. 2011.
[39] T. Berglund, A. Brodnik, H. Jonsson, M. Staffanson, and I. Soderkvist, "Planning smooth and obstacle-avoiding B-spline paths for autonomous mining vehicles," *IEEE Trans. Autom. Sci. Eng.*, vol. 7, no. 1, pp. 167–172, Jan. 2010.

ZHENGYANG CUI received the B.S. and M.S. degrees in control theory and control engineering from Xidian University, Xian, China, in 2011 and 2014, respectively. He is currently pursuing the Ph.D. degree in navigation, guidance and control with Beihang University (BUAA), China. His current research interests include path planning, path following control, and formation control of UAVs.

YONG WANG received the B.S., M.S., and Ph.D. degrees in navigation, guidance and control from Beihang University (BUAA), China, in 1988, 1990, and 2001, respectively. Since 2009, he has been a Professor in navigation, guidance and control with BUAA. He is currently an Assistant Chief Engineer and a Senior Technical Fellow with the Institute of Unmanned System, BUAA. His main research interests include advance control theory and its application in flight control engineering, flight control of multi-model vehicles, and autonomous flight management and control system of intelligent UAVs.