Journal of Computers Vol. 36 No. 6, December 2025, pp. 35-50
https://doi.org/10.63367/199115992025123606003

Fire Detection UAV Path Planning Based on the Improving Q-learning Optimization Algorithm

Yu-Jing Liu, Lu-Lu Zheng, and Ling-Qing Feng*

College of Information Science and Engineering, Shanxi Agricultural University,
Jinzhong, ShanXi Province, China

fenglq@sxau.edu.cn

Received 28 November 2024; Revised 25 March 2025; Accepted 8 August 2025

Abstract. This paper presents a method based on an enhanced Q-learning algorithm to improve the efficiency and safety of UAV path planning in fire detection tasks. Addressing the shortcomings of the traditional Q-learning algorithm in complex and high-risk environments, this study incorporates a dynamic greedy strategy and a risk weighting mechanism to improve convergence speed and planning safety. Experimental results show that the improved Q-learning algorithm outperforms the classical version in convergence speed, average cumulative reward and obstacle avoidance rate. demonstrating stronger adaptability in fire detection scenarios with obstacles and high-risk regions. This method efficiently determines the optimal drone path from the starting point to the target area while minimizing the likelihood of crossing high-risk zones, thereby improving the safety and efficiency of fire detection. The research findings provide strong support for the path planning problem in drone-based fire detection and offer a theoretical basis for future related research and applications.

Keywords: Q-learning algorithm, path planning, risk factor, dynamic greedy strategy, reward function

1 Introduction

With the rapid development of UAV technology in recent years, the use of UAVs in firefighting and disaster detection has increased, particularly in the detection and monitoring of fire situations, showing their unique advantages. However, UAV path planning has many challenges when detecting complicated fire situations. Traditional path planning methods are ineffective at responding to dynamic changes in the fire scene. They tend to get stuck in local optimal solutions, resulting in resource waste and inefficiency. At the same time, uncertainties such as high-temperature obstructions and smoke are common in the fire region, exposing UAVs to significant safety risks when performing their missions. To improve the performance of UAVs in fire detection duties, it is critical to develop a path-planning method that takes into account all environmental risk factors and is robust.

This study aims to address the efficiency and safety issues of UAV path planning in high-risk fire environments by proposing a fire detection path planning method based on an improved Q-learning algorithm. By introducing a dynamic greedy strategy and a risk weight mechanism, the proposed method seeks to optimize UAV path selection in response to dynamic changes and potential hazards in fire scenarios. Fast and effective path planning is directly related to the timeliness of fire rescue and UAV safety; in emergency situations, optimized path planning can significantly improve rescue efficiency and reduce operational risk. The method incorporates an environmental risk weight factor, integrating high-risk obstacles and their dynamic characteristics in fire areas into the path planning process. It optimizes the reward function to guide UAVs in avoiding high-risk areas while effectively reaching target locations. Additionally, a risk zone penalty mechanism is designed to make UAVs prioritize safety and stability in path selection. This enhances the reliability and real-time performance of fire detection tasks.

Q-learning, a typical model-free reinforcement learning algorithm, was proposed by Watkins in 1992 [1]. Due to its independence from prior environmental knowledge and suitability for dynamic environments, it has been widely applied in fields such as UAV path planning. However, traditional Q-learning still faces from path planning instability and low learning efficiency when confronted with high-risk obstacles or dynamically changing fire environments, motivating researchers to explore improved approaches. In [2], the authors proposed an optimized Q-learning algorithm (O-QL), which improves path planning performance for mobile robots in complex

* Corresponding Author

maze environments by introducing a new Q-table initialization and the RMSprop method. The algorithm demonstrates excellent performance in reducing collisions and improving success rates. In [3], the authors introduced a Q-learning-based multi-strategy integrated artificial bee colony algorithm (QMABC), which enhances search efficiency through diversified strategies and has been validated in unmanned vehicle path planning. In [4], researchers developed an adaptive Q-learning particle swarm optimization (AQLPSO) method to improve the path planning efficiency of multiple UAVs in complex 3D flight environments. Compared with traditional PSO and its variants, AQLPSO shows significant advantages in avoiding local optima and accelerating global convergence. In [5], the authors proposed a Neural-network Smoothed Fast Q-learning (NSFQ) algorithm to address the challenges of path planning and obstacle avoidance for unmanned surface vehicles (USVs). The algorithm accelerates convergence by integrating RBF neural networks and employs Bézier curves for path smoothing. Simulation results show that the NSFQ algorithm outperforms A* and RRT in terms of path efficiency and stability. Despite these advancements, the adaptability of these algorithms in extremely dynamic environments still requires further improvement.

Although existing studies have improved the performance of Q-learning in path planning, its adaptability and real-time performance in highly dynamic and high-risk fire environments still need enhancement. Therefore, this paper first addresses the path planning problem in complex indoor disaster scenarios by modeling the environment using a grid-based method and proposing an improved Q-learning algorithm incorporating risk factors to optimize UAV path planning efficiency in fire environments. Secondly, to address the issues of low sample efficiency and slow convergence of traditional Q-learning in large-scale environments, a dynamic greedy strategy is introduced, enabling the agent to explore more unknown options in the early stages of learning and exploit optimal strategies more effectively in later stages. Finally, a reward function model that incorporates dynamic risk factors is constructed to enhance the robustness of UAV path selection.

By comparing the path safety and planning performance under different risk weights (ω_{risk} = 0.3, 0.5, 0.7), the impact of the risk perception mechanism on policy performance is validated. Meanwhile, the convergence performance under different exploration rate decay factors (λ = 0.001, 0.003, 0.005) is analyzed. This analysis demonstrates that a properly configured decay factor can enhance the algorithm's convergence efficiency and stability in complex environments. Through the application of these methods, we aim to improve the path planning performance of UAVs in fire detection tasks while ensuring operational safety, offering a practical and high-performing solution.

2 Related Work

The goal of UAV route planning is to determine the best or most feasible path from a starting point to a target location in a given environment. UAVs encounter a large number of obstacles during flight, which brings a certain amount of risk. To ensure a smooth and safe flight, UAV mature path planning algorithms are unavoidable; to reduce flight time and improve work efficiency, UAVs tend to choose a shorter path to fly. UAV path planning is significant in many complicated activities, particularly in fire detection and monitoring, where path planning optimization is critical to improving work efficiency and safety. In recent years, several studies have suggested and tested various path planning algorithms in a variety of settings and tasks.

In [6], the researchers reviewed drone path planning methods, introducing a novel classification of environmental complexity and critically analyzing existing techniques. The study highlighted the significance of path planning in dynamic environments, identified challenges in environmental modeling and path optimization, and proposed future research directions. In [7], the authors conducted a review of drone path planning methods, categorizing them into five major groups: classical methods, heuristics, metaheuristics, machine learning, and hybrid algorithms. They also critically analyzed their objectives, constraints, and environmental factors. In [8], the authors proposed a drone path planning method based on obstacle modeling, ensuring collision avoidance through mathematical proof and generating collision-free trajectories using the Dijkstra algorithm. This method has a low computational load, making it suitable for real-time path planning. Simulation results demonstrated its effectiveness in planning shorter and safer paths. In [9], the authors reviewed the application of artificial intelligence (AI) in swarm drone path planning, analyzing four major approaches: reinforcement learning, evolutionary computation, swarm intelligence, and graph neural networks. The study summarized recent advancements and revealed research trends and technological evolution through quantitative analysis, indicating continuous growth in this field and changes in the predominant application patterns. In [10], the authors conducted a classification and systematic review of autonomous UAV path planning methods, analyzing their evolution from classical techniques to advanced solutions, with a particular focus on path planning in dynamic environments. The study compared different methods in terms of time efficiency, environmental adaptability, and testing platforms, highlighting challenges in obstacle avoidance, real-time responsiveness, and energy consumption management, while also suggesting future research directions.

In fire detection missions, UAV path planning faces increasingly challenging problems. The heat, smoke, and barriers at the fire site require path planning algorithms that not only take the shortest route but also avoid high-risk regions. In [11], a drone-based forest fire patrol path planning strategy was proposed, which emphasizing the importance of path planning in fire patrols, particularly in avoiding obstacles and improving patrol efficiency in complex forest environments. This study provides new insights into the application of drone path planning for forest fire prevention and control. In [12], an autonomous flight trajectory planning method have been proposed, which can optimize drone paths using a multi-criteria algorithm to identify fire ignition sources at an early stage. This method employs a three-stage flight plan, integrating local differential operators and a nine-element matrix to assess hazardous substance concentrations, enabling precise localization and efficient navigation. In [13], the authors proposed a forest fire escape route planning method based on an improved A* algorithm, incorporating FAHP-CRITIC weighting analysis of 11 influencing factors to enhance escape safety. In [14], a modified ant colony algorithm was proposed for escape route planning, which integrated a fire spread model to enhance accuracy and adopted the A algorithm to improve convergence speed. In [15], an improved adaptive ant colony optimization (IAACO) algorithm was proposed for intelligent planning of building fire evacuation routes. This algorithm optimizes heuristic functions and pheromone updates by considering risk, energy consumption, and path length, enhancing global search capability. Experiments demonstrated that compared to traditional ACO and IACO methods, IAACO reduces path suboptimality in various environments and exhibits superior dynamic planning performance under multi-exit and multi-fire-source conditions, making it suitable for fire evacuation simulations.

3 Fire Detection UAV Path Planning Method

In this research, a UAV path planning algorithm based on improved Q-learning is proposed to address the issues of adapting to dynamic environments and ensuring path safety in fire detection scenarios. The system consists of elements for environment modeling, reward function design, and Q-learning algorithm optimization, enabling UAVs to perform path planning and fire source localization quickly and efficiently in fire situations. Each of these components is elaborated in detail below.

3.1 Grid Environment Modeling

In fire detection tasks, the fire zone often expands dynamically, characterized by high temperatures, smoke, and other hazards. Environment modeling, which simulates objects and features within this context, forms the basis for path planning. By modeling indoor barriers and humans prior to path planning, this approach ensures a clearer understanding of complex environmental details, leading to precise and thorough path planning, and supporting autonomous navigation in challenging conditions.

The UAV flight area is represented in this study by a two-dimensional grid map, which is divided into multiple grid cells to model the environment. Each grid can be represented by the index number and the right-angle coordinate method. The black grid is the obstacle grid, also known as the "obstacle area" or "high-risk area," and the white grid is the passable grid, also known as the "feasible area" (show Fig. 1(a)). And the impediments that prevent the UAV from flying in this environment are inflated (show Fig. 1(b)). The specific rules are as follows:

Feasible area (white grid): Areas where UAVs can fly safely.

Obstacle area (black grid): Areas where there are construction objects, objects or other physical obstacles that the UAV may not penetrate, such as ① is a pillar and ② is a table in Fig. 1.

High-risk area (black grid): Areas near fire sources with higher temperatures or higher smoke concentrations are potentially dangerous. The UAV needs to avoid these areas as much as possible, as shown in Fig. 1, where ③ is the area of the fire source.

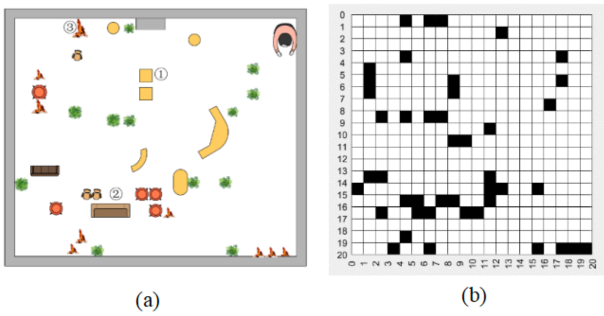
Additionally, fire parameter mapping and grid resolution selection: The grid resolution (20×20) is designed to balance the dynamic characteristics of fire and drone maneuverability. The specific criteria are as follows:

Fire Spread Rate: The fire spreads at a rate of 1 meter per second. The grid size (assumed to be 5 meters) must be less than five times the spread rate to ensure real-time updates of high-risk areas.

Drone Maneuverability: The minimum turning radius of the drone is 3 meters, and its flight speed is 5 m/s. A grid resolution that is too fine (e.g., 1 meter) would cause frequent path adjustments, reducing planning efficiency. Based on this trade-off, a 20×20 grid is chosen to balance modeling accuracy and computational real-time performance.

Dynamic Obstacle Update Mechanism: To reflect the dynamic changes in fire spread, the environmental model detects fire propagation every 50 iterations and updates the grid states accordingly.

High-Risk Area Expansion: The grid containing the fire source and its adjacent grids are updated as high-risk areas based on the heat conduction equation.



(a) Feasible and obstacle areas (b) Obstacle expansion and high-risk area representation

Fig. 1. Grid environment modeling

Obstacle Generation: When combustible materials collapse, new obstacle grids are randomly added, updating the state transition matrix and marking impassable areas, as shown in Eq. (1).

$$D = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{bmatrix}. \tag{1}$$

3.2 Q-learning Optimization Algorithm

Reinforcement learning is a distinctive method that utilizes environmental feedback as input. The core concept of reinforcement learning involves the dynamic interaction between the agent and its environment. This interaction enhances the agent's decision-making strategies, teaching it to select actions in various states to optimize long-term rewards and adapt to diverse environments and tasks. In reinforcement learning, the agent's state, action, strategy, and reward form a closed-loop feedback system. This system enables the agent to continuously learn and improve its decision-making methods. Fig. 2 illustrates the reinforcement learning mode.

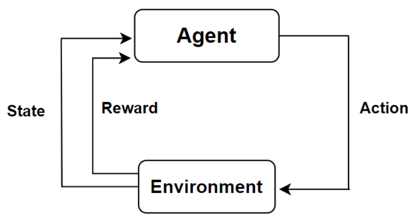


Fig. 2. Reinforcement learning basic model

Q-learning is the most commonly used reinforcement learning algorithm based on the value function approach to solution. To achieve the optimal policy, Q-learning iterates on state-action pairs Q (s, a). In every iteration, the algorithm checks the value of each state-action pair Q (s, a). Q (s, a) represents the expected return of each action performed by the robot in each state. During iteration, Q-learning creates a sequential a table Q {s, a} with values representing the cumulative expected return of taking action a in state s. The UAV's or robot's) decision-making process consists of determining the best action a. The function update rule for the Q learning method is depicted in Eq. (2).

$$Q(s_t, a_t) \leftarrow (1-\alpha) * Q(s_t, a_t) + \alpha * \left[r_{t+1} + \gamma \max_a Q(s_{t+1}, a_{t+1}) \right]. \tag{2}$$

The above equation can also be replaced by Eq. (3) and (4)

$$V(s_t) \leftarrow V(s_t) + \alpha * \left[r_{t+1} + \gamma V(s_{t+1}) - V(s_t) \right]. \tag{3}$$

$$Q(s_t, a_t) \leftarrow (1-\alpha) * Q(s_t, a_t) + \alpha * \left[r_{t+1} + \gamma V(s_{t+1}) \right]. \tag{4}$$

Where Q (s_t, a_t) means the Q value of doing action an in the current states s_t, α denotes the learning rate, r denotes the immediate reward earned following action a, and γ denotes the discount factor, which is used to balance the relevance of future rewards.

After Q-learning has converged, a converged Q-table can be obtained to guide the UAV (or relevant agent) in maximizing the cumulative reward and eventually learning the optimal policy to accomplish the path planning task. When the states and actions are finite, the Q-values can be represented using Q-tables. When each state and action has unlimited access to the Q-table, and the action is chosen after convergence, Q-learning converges to the optimal policy, yielding the largest Q-value for each state, as illustrated in Eq. (5) below.

$$V^*(s_t) = \max_a Q(s_t, a_t). \tag{5}$$

Pseudo-code for the classical Q-learning algorithm is summarized in Algorithm 1.

Algorithm 1. Classical Q-learning pseudocode

(1) Set the learning rate α, discount factor to 0, and initialize all state-action pairs Q values to zero

(2) For each trial, select an initial state s_t

(3) Repeat for multiple trials, each trial as follows:

(4) Check if the current state s_t is terminal:

(5) If not, select and execute an action a_t

(6) Obtain the immediate reward r_t after executing action a_t

(7) Observe the new state s_{t+1}

(8) Update the Q value using the formula:

(9) $Q(s_t, a_t) \leftarrow (1-\alpha) * Q(s_t, a_t) + \alpha * \left[r_{t+1} + \gamma \max_a Q(s_{t+1}, a_{t+1}) \right]$

(10) Update s_t to s_{t+1} for the next iteration

(11) End the trial when s_t is a terminal state

(12) Loop ends

(13) Derive the optimal policy V*(s_t) from the Q values, which is the action a that maximizes Q for each state s_t

3.3 Reward Function Design

The reward function is a necessary component of the Q-learning algorithm, and it has a direct impact on the efficacy of path planning. In the fire detection task, to ensure that the UAV can quickly approach the fire source area while avoiding the potentially dangerous areas, the UAV's action space is integrated into the state's position. There are four actions: up, down, left, and right, which are indicated by the blue arrows in Fig. 3(a). The obstacle area is inaccessible, as indicated by the red arrows in Fig. 3(a). Also, the diagonal area is considered due to the irregular shape of the environment.

When an agent performs a certain action in a given state, it receives an appropriate reward. When the agent is at the edge of the scene, it can only perform two possible actions, as shown in Fig. 3(b). The feedback from the environment is obtained in three cases: when approaching the target fire region, the agent gets a reward of 1; when approaching the obstacle region, the reward is -1; and otherwise, the reward is 0, as shown in Fig. 3 in (c), (d), and (e).

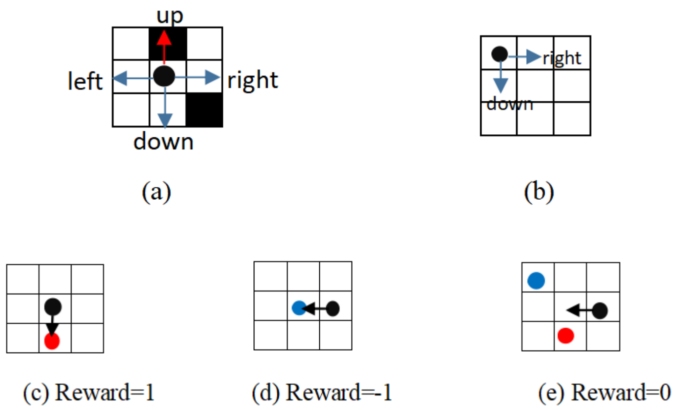


Fig. 3. Reward function design

When the intelligent body acts in the ith state, it receives the reward can be stated using the formula Equation (6).

$$\Delta(s_i,a)=\begin{cases}1, & \textit{reach the target fire area}\\ -1, & \textit{reach the obstacle area}\\ 0, & \textit{reach the other area}\end{cases}. \tag{6}$$

In this study, we propose a reward function that fits task efficiency with security. The function contains the following important parts:

1) Target Reward (R_{goal}): When the UAV successfully approaches the fire location, it receives a high positive reward, R_{goal}, to motivate it to approach the fire as soon as possible and accomplish the task. Specifically, the target reward might be set to a huge constant value that matches the mission's overall purpose.

2) Step Penalty (R_{step}): To regulate the length of the path and prevent the UAV from taking a long diversion, this study proposes the step penalty R_{step}. A little negative reward is supplied for each step of the process, motivating the UAV to take a shorter path, reducing path length, and increasing task efficiency.

3) Obstacle Penalty ($R_{obstacle}$): To prevent UAVs from striking obstacles or entering risky areas throughout the operation, a significant negative reward $R_{obstacle}$ is given whenever a UAV enters an obstacle area. This penalty term is intended to drive the algorithm to avoid obstacles as much as possible in order to ensure the safety of the drone.

4) Risk penalty (R_{risk}): To make the UAV avoid high-risk areas near the fire source, this research incorporates risk weights into the reward function. The risk weights are changed constantly based on the distance D between

the UAV's current location and the fire source. When the UAV is closer to the fire source, the risk penalty rises. The risk penalty term is Eq. (7).

$$R_{risk} = \omega_{risk} \cdot D. \tag{7}$$

Here, D represents the straight-line distance between the drone's current position and the fire source. ω_{risk} is the risk weight based on distance from the fire source, dynamically calculated using fire sensor data, including temperature (T) and smoke concentration (C):

$$\omega_{risk} = 0.6 \cdot \frac{T}{800°C} + 0.4 \cdot \frac{C}{500\,ppm}. \tag{8}$$

Combining the aforementioned reward and punishment terms, the overall reward function proposed in this study is Eq. (9):

$$R = R_{step} + \alpha \cdot R_{goal} - \beta \cdot R_{obstacle} - \omega_{risk} \cdot D. \tag{9}$$

Here, α and β are weighting parameters used to balance the relationship between the goal reward R_{goal} and the obstacle penalty $R_{obstacle}$, ensuring both path safety and task effectiveness. Based on empirical analysis, the key parameter selection in this study is as follows: $\alpha = 1.2$, $\beta = 1.5$, $\omega_{risk} = 0.5$.

Experimental results indicate that an excessively high α value (>1.5) may cause the drone to overlook obstacle risks. When $\omega_{risk} = 0.5$, the path length and safety are relatively balanced, whereas $\omega_{risk} = 0.7$ allows the drone to completely avoid high-risk areas but increases the path length by approximately 8%. Considering these factors, this study selects appropriate parameters to ensure efficient and stable drone navigation in fire environments.

3.4 Improving Q-learning Optimization Algorithm

To improve the convergence efficiency and dynamic adaptability of path planning, this work refines the classic Q-learning algorithm, which performs poorly in dynamic environments and hazardous areas. The two components of the improvement approach are as follows.

Dynamic Greedy Strategy. This research proposes a dynamic strategy for improving UAV adaptability in unexpected situations. The algorithm begins with a high exploration rate (ϵ) and gradually decreases it as the training process progresses, leading to the more frequent selection of the optimal path. The formula is shown in Eq. (10).

$$\epsilon_t = \epsilon_{\min} + \left(\epsilon_{\max} - \epsilon_{\min}\right) e - \lambda t. \tag{10}$$

$$\epsilon_t = 0.1 + 0.8e^{-0.003t}. \tag{11}$$

Where $\epsilon_{\max}$ and $\epsilon_{\min}$ are the maximum and minimum exploration rates, respectively, λ is the exploration rate decay parameter, and t is the current number of cycles. As a result, the algorithm can explore a sufficient number of options at an early stage before stabilizing to select the optimal path later.

To adapt to different fire environments, this study optimizes the exploration rate decay factor λ. When the fire spreads rapidly (e.g., advancing two grid cells per second), increasing λ to 0.005 accelerates convergence and reduces exploration in high-risk areas. In relatively stable fire conditions, setting $\lambda = 0.001$ ensures sufficient exploration. When $\lambda = 0.003$, it helps balance exploration and exploitation, enabling the algorithm to adapt more efficiently to dynamic fire environments.

Risk-weighted Update Mechanism. The typical Q-learning update formula is based on the current state's reward value and the maximum Q value of the future state. However, in the fire detection task, if the future state is in a high-risk region, the algorithm may still take a dangerous path. As a result, this study proposes a risk-weighted update method to modify the Q-value update formula Eq. (12) based on the risk factor of the current state:

$$\varrho(s,a) \leftarrow \varrho(s,a) + a\left[R(s,a) + \gamma(1-\omega_{risk})\max_{\acute{a}} \varrho(s',a') - \varrho(s,a)\right] \quad \textbf{(12)}$$

Among them:

- ω_{risk} are risk weight, updated in real-time based on fire sensor data, with parameter adjustments every 10 iterations.
- ω_{risk} higher when the future state is in the high-risk region,
- Reduces the magnitude of the Q update, prompting the algorithm to choose a safer path.

3.5 Algorithm Flow

The flow of the improved Q-learning algorithm in this paper is shown in Fig. 4.

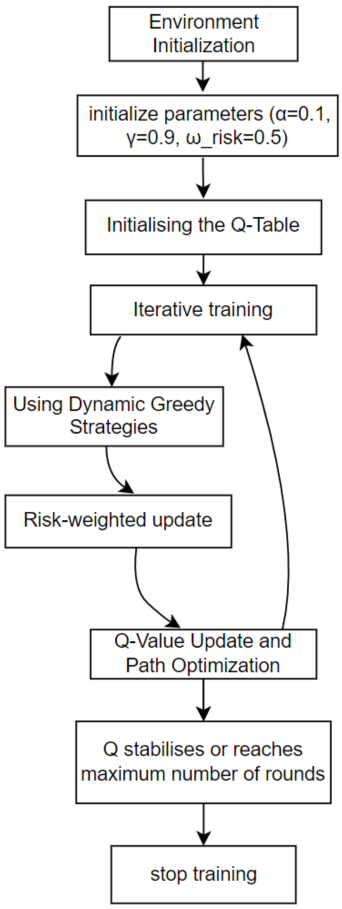


Fig. 4. Improvement of Q-learning algorithm flow

Table 1. Algorithm flow analysis

Step	Explanation
Q-table initialization	Set the initial position of the UAV, environment map, and Q-value table
Iterative training	In each episode, select the optimal or random action based on the current state and policy, then calculate the new position and state
Reward calculation	Update Q-values based on the reward function, including target reward, step penalty, obstacle penalty, and risk penalty
Risk weight update	Use the improved Q-learning update formula to update the Q-table
Convergence condition check	Stop training when Q-values stabilize or the maximum number of episodes is reached

3.6 Algorithm Complexity Analysis

The algorithm involves accessing and updating the Q-table in each state update, with a time complexity of O(N·T), where N is the number of states and T is the maximum number of steps. The incorporation of the risk-weighting mechanism and dynamic greedy strategy does not significantly increase computational overhead, thus balancing computational efficiency while ensuring path optimization effectiveness.

Additionally, in terms of real-time performance, dynamic parameter updates (e.g., adjusting λ and obstacle detection) introduce only an O (1) overhead. Experimental results show that the average training time per iteration is 2.1 seconds (Intel i7-12700K), meeting the real-time requirements of fire scenarios, where fire spread updates occur at intervals of at least 5 seconds.

4 Experimental Settings

The purpose of this experiment is to verify the effectiveness and superiority of the improved Q-learning algorithm in fire detection path planning. The experiment compares the path planning performance of traditional and improved Q-learning algorithms to determine differences in convergence speed, average cumulative reward, and obstacle avoidance rate. The experimental setup is divided into the following sections:

4.1 Experimental Environment

The studies were carried out using a PC equipped with an Intel Core i7-12700K processor and 16GB of RAM. To ensure data objectivity and repeatability, the experiments were carried out under identical environmental and hardware conditions. The UAV's flight environment was simulated using a 20x20 raster map, with the initial position (1,1) in the upper left corner and the fire source fixed in the lower right corner (20,20). To simulate barriers like buildings and trees, 10% of the obstacles were randomly placed on the map. Furthermore, high-risk regions are established near the fire source, and different risk factors are assigned to different areas in order to simulate the risk of heat and smoke around the fire.

4.2 Experimental Steps

The experimental steps are shown in Table 2.

Table 2. Experimental steps

Step	Explanation
Initialization	To initialize the raster map, barriers, and high-risk zones, parameterize each algorithm independently.
Algorithm training	The algorithm determines the next state and the reward, selects an action based on the greedy approach and current state, and updates the Q value until it reaches the end point or the maximum number of steps in each round.

Data recording	To observe the algorithm's convergence properties, the cumulative reward, path length, and avoidance of obstacles or high-risk zones were recorded for every round throughout the training phase.
Dynamic environment update	Every 50 iterations, the fire spread is assessed, updating the distribution of obstacles and high-risk areas, and reinitializing the drone's position to a safe area.
Performance evaluation	Following algorithm training, the two methods were compared in terms of path safety (the percentage of obstacles and high-risk areas avoided), path optimization efficacy (final path length), and speed of convergence (cumulative rewards over the number of rounds).

4.3 Performance Evaluation Indicators

In order to comprehensively assess the effectiveness of the improved Q-learning algorithm, the experiment uses the evaluation metrics shown in Table 3.

Table 3. Evaluation indicators

Indicator	Instructions
Convergence rate	To determine whether the algorithm can rapidly converge and accrue more prizes, compute the curve of cumulative awards as a function of the number of rounds.
Path length	Record the final path length for each round and evaluate whether the path generated by the algorithm is optimized, i.e., the goal is reached using the shortest way.
Path security	Statistical algorithms to avoid obstacles and high-risk locations on the path and calculate the fraction of obstacles and high-risk areas avoided, which is used to quantify the algorithm's safety.
Average cumulative award	Calculate the average reward per round as a metric of the algorithm's learning efficiency to assess the algorithm's adaptability in various situations.
Parameter sensitivity	Comparison of the effect of different ω_{risk} and λ values on performance metrics (e.g., path length ±8%)

4.4 Experimental Comparison of Algorithms

The experiment selects the classic Q-learning algorithm as a comparison method and verifies the superiority of the improved Q-learning algorithm in UAV path planning for fire detection through comparative analysis. First, in the static parameter group, the path safety and length were compared under ω_{risk} values of 0.3, 0.5, and 0.7. Second, in the dynamic parameter group, the differences in path planning between dynamically calculated ω_{risk} and a fixed ω_{risk} of 0.5 were analyzed. Finally, in the decay coefficient group, the impact of different decay coefficients (λ = 0.001, 0.003, 0.005) on convergence speed was compared. The algorithm parameter settings are shown in Table 4.

Table 4. Algorithm parameter setting

Parameters	Value	Remarks
Learning rate α	0.1	Controls the update rate of Q-values
Discount factor γ	0.9	Balances the weight between current rewards and future rewards
Maximum training episodes	5000	The maximum number of episodes the algorithm runs in each experiment
Maximum steps per episode	100	The maximum number of steps the UAV executes in each episode
Maximum exploration rate ϵ_{max}	0.9	Initial exploration probability
Minimum exploration rate ϵ_{min}	0.1	Minimum exploration probability
Exploration decay factor λ	0.001-0.005	Dynamically adjusted based on fire spread rate (default 0.003)
Risk weight ω_{risk}	0.3/0.5/0.7	Static values for comparative experiments; dynamically calculated based on sensor data
Dynamic obstacle update frequency	Update every 50 episodes	Simulated fire spread-generated obstacles

5 Experiment and Analysis

This section compares and analyzes the performance of the conventional and improved Q-learning algorithms in fire detection UAV path planning. The experimental results include evaluations of four performance metrics: convergence speed, path length, path safety, and average accumulated reward. Because the experimental results can reach the optimal result after many iterations, the experiment selected a variety of cases in which the obstacles in the path accounted for 10%, 30%, 50%, and 70% of the feasible area for comparison. By comparing the experimental results, we can analyze in depth the advantages and effects of the improved algorithm.

In the experiment, black regions represent obstacles (impassable areas), pink regions indicate high-risk areas (near fire sources, with color depth representing risk levels), and the red polyline represents the optimal path planned by the UAV. As shown in Fig. 5(a), the fire source area is relatively small, and the UAV successfully avoids obstacles while moving toward the target area. Fig. 5(b) illustrates the spread of the fire, where the pink high-risk area significantly expands. However, the UAV is still able to dynamically adjust its path, avoid high-risk areas, and approach the fire source. These results intuitively demonstrate the differences between various algorithms in path optimization, obstacle avoidance, and adaptability to dynamic environments.

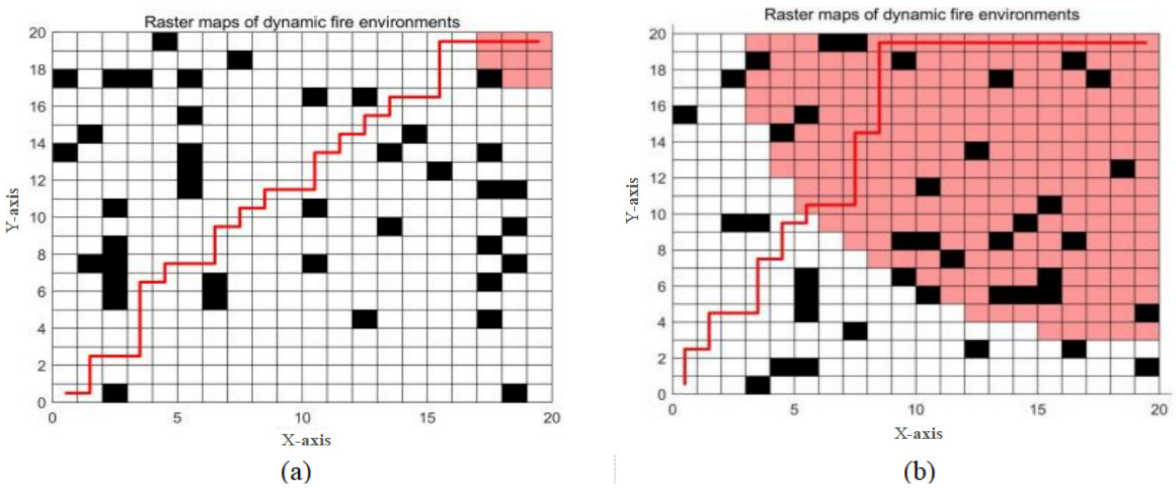
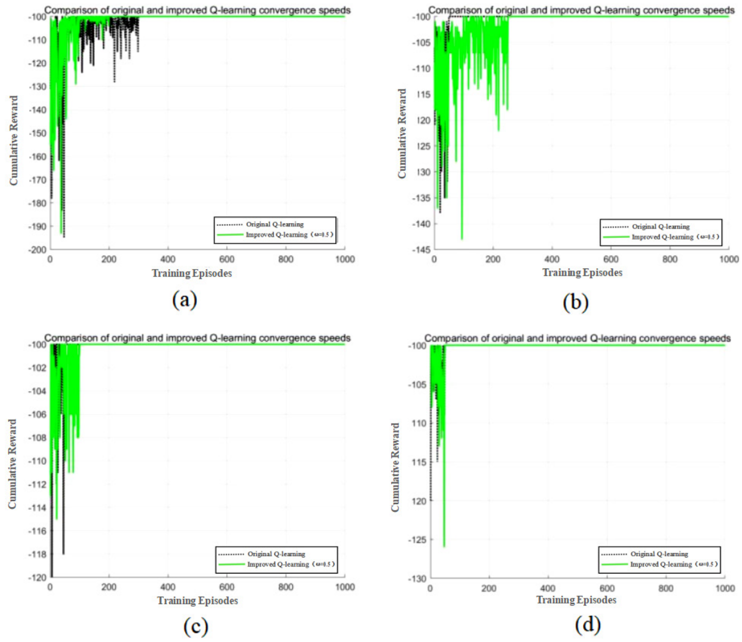


Fig. 5. Raster maps of dynamic fire environments

5.1 Convergence Speed and Stability

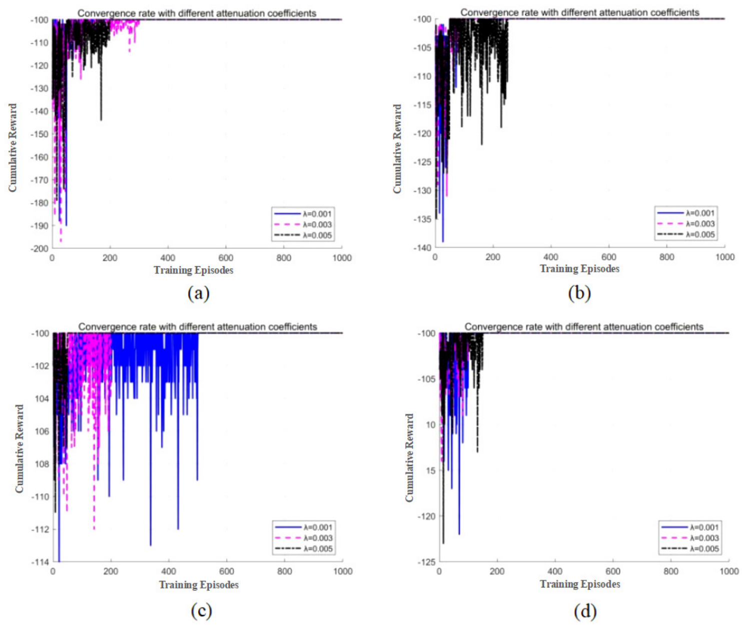
Experimental results show that the improved Q-learning (ω=0.5) achieves faster convergence and higher stability in different obstacle environments (Fig. 6). The improved Q-learning algorithm effectively reduces early - stage fluctuations and enhances exploration efficiency. In high-obstacle environments (50%-70%), as shown in Fig. 6(c) and Fig. 6(d), its convergence time is significantly shorter than that of the original Q-learning, which exhibits greater fluctuations and weaker adaptability in complex environments. By optimizing risk weights, the improved Q-learning enhances policy stability and environmental adaptability. Compared to fixed risk weight (ω=0.5) setting and the original Q-learning, it converges faster and demonstrates more stable performance, as shown in Fig. 10.

Furthermore, different exploration rate decay factors (λ=0.001, 0.003, 0.005) also affect convergence speed and stability (Fig. 7). In low-obstacle environments (10%-30%), as shown in Fig. 7(a) and Fig. 7(b), all decay factors achieve relatively fast convergence with minimal impact. However, in high-obstacle environments (50%-70%), as shown in Fig. 7(c) and Fig. 7(d), a smaller decay factor (λ=0.001) prolongs the exploration phase, resulting in greater policy fluctuations. In contrast, a larger decay factor (λ=0.005) reduces ineffective exploration more quickly, accelerates convergence, and enhances stability. Therefore, in simple environments, the decay factor has a minimal impact, whereas in complex environments, a larger decay factor is more beneficial for improving convergence efficiency and policy stability.



(a) The obstacles occupy 10% of the feasible area. (b) The obstacles occupy 30% of the feasible area. (c) The obstacles occupy 50% of the feasible area. (d) The obstacles occupy 70% of the feasible area

Fig. 6. Comparison of original and improved Q-learning convergence speeds

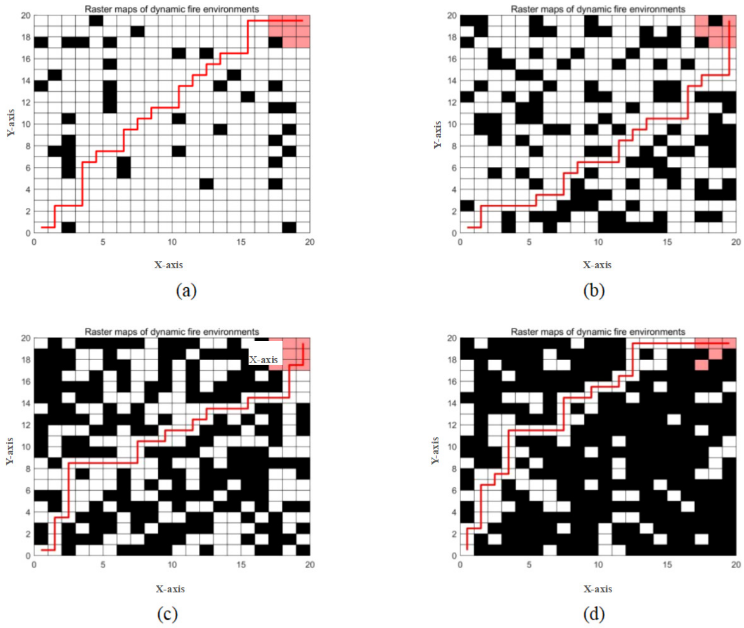


(a) The obstacles occupy 10% of the feasible area. (b) The obstacles occupy 30% of the feasible area. (c) The obstacles occupy 50% of the feasible area. (d) The obstacles occupy 70% of the feasible area

Fig. 7. Convergence rate with different attenuation coefficients

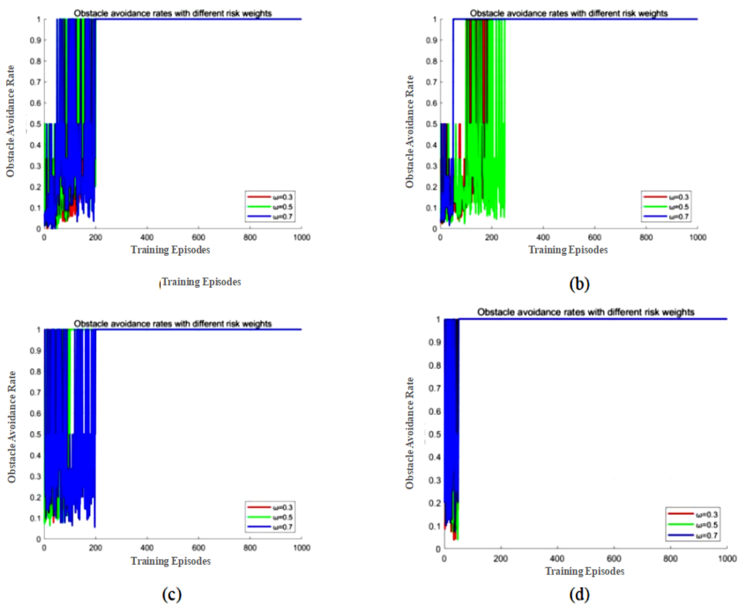
5.2 Path Optimization and Security

In terms of path length and path safety, Fig. 8 presents the final paths generated under obstacle ratios of 10%, 30%, 50%, and 70% of the feasible area. When obstacles are sparse (Fig. 8(a) and Fig. 8(b)), the generated paths are nearly straight, allowing the UAV to reach the target point quickly. As the number of obstacles increases (Fig. 8(c) and Fig. 8(d)), the paths become more winding, indicating that path planning is constrained by obstacles. However, the proposed method can still effectively avoid obstacles. Ultimately, the paths converge to the optimal solution, demonstrating that the improved Q-learning algorithm can discover more efficient paths during the exploration process.



(a) The obstacles occupy 10% of the feasible area. (b) The obstacles occupy 30% of the feasible area. (c) The obstacles occupy 50% of the feasible area. (d) The obstacles occupy 70% of the feasible area

Fig. 8. Path comparison under four scenarios



(a) The obstacles occupy 10% of the feasible area. (b) The obstacles occupy 30% of the feasible area. (c) The obstacles occupy 50% of the feasible area. (d) The obstacles occupy 70% of the feasible area

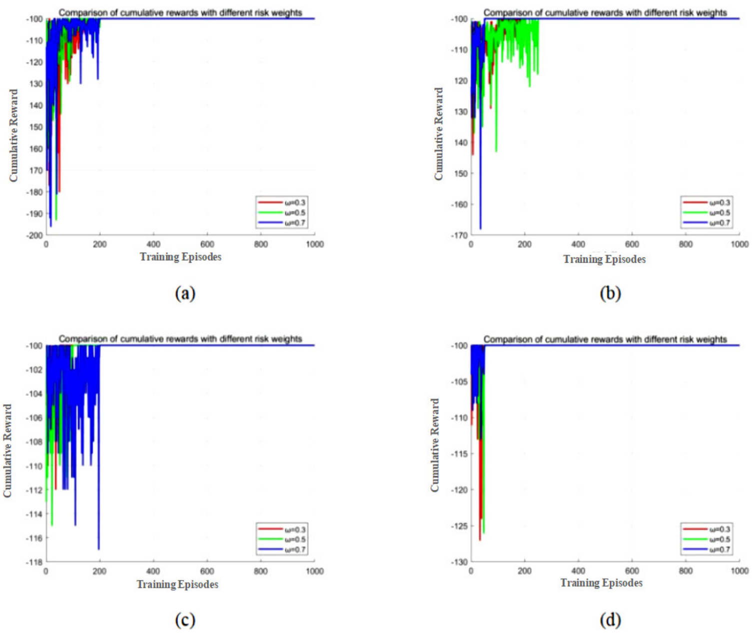
Fig. 9. Obstacle avoidance rates with different risk weights

Fig. 9 further verifies the impact of different risk weights on path safety. As the number of episodes increases, all three risk weights eventually converge, with the avoidance rate dropping to zero, indicating that the algorithm has good adaptability. As the obstacle ratio increases, the convergence time becomes longer. The high-risk weight (ω=0.7) exhibits greater fluctuations and slower convergence in complex environments, whereas the low-risk weight (ω=0.3) demonstrates better stability. In low-obstacle environments (10%-30%), as shown in Fig. 9(a) and Fig. 9(b), all risk weight strategies converge quickly, and the high-risk strategy can still efficiently find a path. In moderate-obstacle environments (50%), as shown in Fig. 9(c), a moderate risk weight (ω=0.5) achieves a better balance. In high-obstacle environments (70%), as shown in Fig. 9(d), the low-risk strategy (ω=0.3) converges the fastest, ensuring more stable path planning. Overall, the algorithm can ultimately find a feasible path, and moderately reducing the risk weight helps accelerate convergence and improve stability.

5.3 Cumulative Rewards and Risk Weight Optimization

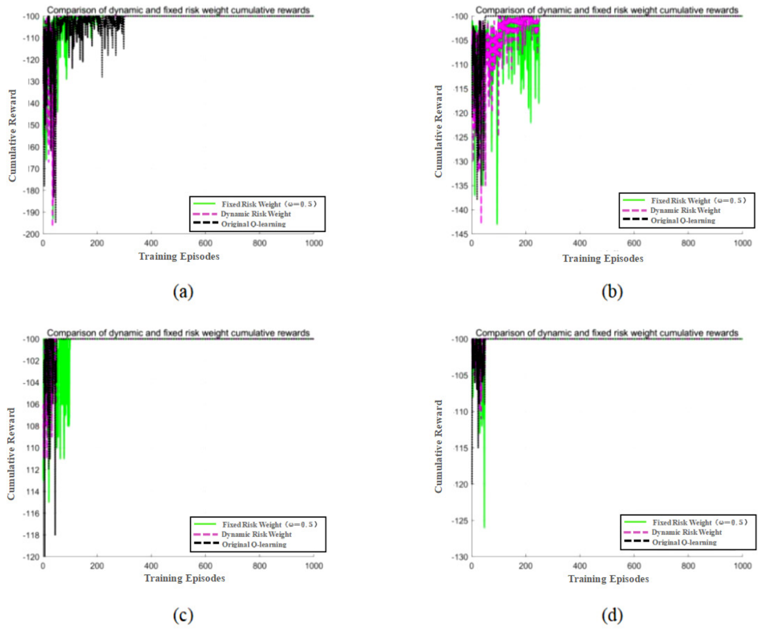
Fig. 10 and Fig. 11 illustrate the cumulative reward variations under different risk weights and the comparison between fixed and dynamic risk weight strategies. When obstacles are sparse (Fig. 10(a) and Fig. 10(b)), the cumulative rewards converge quickly and stabilize after 100–150 episodes, indicating that risk weight has little impact on path optimization. As the number of obstacles increases (Fig. 10(c) and Fig. 10(d)), moderate risk weights (ω=0.3, 0.5) demonstrate more stable performance. In contrast, a higher risk weight (ω=0.7) leads to greater fluctuations in cumulative rewards due to increased exploration.

In comparison, the dynamic risk weight strategy in Fig. 11 better adapts to different environments and enhances cumulative rewards. Compared to the fixed risk weight (ω=0.5) and the original Q-learning, the dynamic risk weight achieves faster convergence and more stable performance. As the number of obstacles increases, the fixed risk weight strategy can still learn paths but exhibits weaker adaptability. The original Q-learning, on the other hand, shows greater fluctuations and the slowest convergence, making it particularly difficult to find optimal paths in high-obstacle environments. Overall, the dynamic risk weight strategy demonstrates stronger adaptability and performs best in complex environments.



(a) The obstacles occupy 10% of the feasible area. (b) The obstacles occupy 30% of the feasible area. (c) The obstacles occupy 50% of the feasible area. (d) The obstacles occupy 70% of the feasible area

Fig. 10. Comparison of cumulative rewards with different risk weights



(a) The obstacles occupy 10% of the feasible area. (b) The obstacles occupy 30% of the feasible area. (c) The obstacles occupy 50% of the feasible area. (d) The obstacles occupy 70% of the feasible area

Fig. 11. Comparision of dynamic and fixed risk weight cumulative rewards

6 Conclusions

In this research, we develop and apply a UAV path planning algorithm based on improved Q-learning to the fire detection task. The revised method improves the efficiency and safety of classic Q-learning-based path planning by incorporating a dynamic greedy strategy and risk weights. The experimental results reveal that the modified Q-learning algorithm performs better than the conventional Q-learning algorithm in terms of convergence speed, path length, path safety, and average cumulative reward. During training, the revised algorithm can converge faster and find better paths when environmental complexity and obstacle density increase, demonstrating flexibility in dynamic and high-risk contexts.

Experimental results indicate that appropriately setting the risk weight (ω_{risk}) and exploration rate decay factor (λ) has a significant impact on algorithm performance. A moderate risk weight (e.g., ω_{risk} =0.5) effectively balances path safety and planning efficiency across different obstacle environments. In high-obstacle environments (obstacle ratio≥50%), a lower risk weight (e.g., ω_{risk}=0.3) demonstrates greater stability and faster convergence. Increasing the exploration rate decay factor (e.g., λ=0.005) helps reduce ineffective exploration, accelerate convergence in complex environments, and improve training efficiency and policy stability. In terms of path planning quality, risk weight has a minimal impact in low-obstacle environments. In high-obstacle environments, dynamically adjusting the risk weight can enhance path safety and cumulative rewards.

Overall, this study proposes an efficient, stable, and safe algorithmic solution for the application of improved Q-learning in fire detection path planning, laying a foundation for future research in related fields. The proposed approach not only enhances the real-time performance of UAV-based fire detection tasks, ensures path safety, and improves detection efficiency, but also provides theoretical support for the application of reinforcement learning in high-risk dynamic environments, offering new research directions for UAV autonomous navigation

7 Acknowledgement

This work was financed by Science and Technology Innovation Project of Colleges and Universities in Shanxi Province, China (No. 2022L085).

References

[1] C.J.C.H. Watkins, P. Dayan, Q-learning, Machine learning 8(3-4)(1992) 279-292.
https://doi.org/10.1007/bf00992698

[2] Q. Zhou, Y. Lian, J. Wu, M. Zhu, H. Wang, J. Cao, An optimized Q-learning algorithm for mobile robot local path planning, Knowledge-Based Systems 286(2024) 111400.
https://doi.org/10.1016/j.knosys.2024.111400

[3] X. Ni, W. Hu, Q. Fan, Y. Cui, C. Qi, A Q-learning based multi-strategy integrated artificial bee colony algorithm with application in unmanned vehicle path planning, Expert Systems with Applications 236(2024) 121303.
https://doi.org/10.1016/j.eswa.2023.121303

[4] L. Tan, H. Zhang, Y. Liu, T. Yuan, X. Jiang, Z. Shang, An adaptive Q-learning based particle swarm optimization for multi-UAV path planning, Soft Computing 28(13-14)(2024) 7931-7946.
https://doi.org/10.1007/s00500-024-09691-2

[5] Y. Wang, C. Lu, P. Wu, X. Zhang, Path planning for unmanned surface vehicle based on improved Q-learning algorithm, Ocean Engineering 292(2024) 116510.
https://doi.org/10.1016/j.oceaneng.2023.116510

[6] M. Jones, S. Djahel, K. Welsh, Path-planning for unmanned aerial vehicles with environment complexity considerations: a survey, ACM Computing Surveys 55(11)(2023) 1-39.
https://doi.org/10.1145/3570723

[7] A.A. Saadi, A. Soukane, Y. Meraihi, A.B. Gabis, S. Mirjalili, A. Ramdane-Cherif, UAV path planning using optimization approaches: a survey, Archives of Computational Methods in Engineering 29(6)(2022) 4233-4284.
https://doi.org/10.1007/s11831-022-09742-7

[8] N. Bashir, S. Boudjit, G. Dauphin, S. Zeadally, An obstacle avoidance approach for UAV path planning, Simulation modelling practice and theory 129(2023) 102815.
https://doi.org/10.1016/j.simpat.2023.102815

[9] A. Puente-Castro, D. Rivero, A. Pazos, E. Fernandez-Blanco, A review of artificial intelligence applied to path planning in UAV swarms, Neural Computing and Applications 34(1)(2022) 153-170.
https://doi.org/10.1007/s00521-021-06569-4

[10] G. Sharma, S. Jain, R.S. Sharma, Path Planning for Fully Autonomous UAVs-A Taxonomic Review and Future Perspectives, IEEE Access 13(2025) 13356-13379.
https://doi.org/10.1109/ACCESS.2025.3529754

[11] Y. Xu, J. Li, F. Zhang, A UAV-based forest fire patrol path planning strategy, Forests 13(11)(2022) 1952.
https://doi.org/10.3390/f13111952

[12] S. Filist, R.T. Al-Kasasbeh, R.A. Tomakova, A.K. Al-Fugara, O.M. Al-Habahbeh, O. Shatolova, N.A. Korenevskiy, I.N. Gorbachev, A. Shaqadan, I. Maksim, An unmanned aerial vehicle autonomous flight trajectory planning method and algorithm for the early detection of the ignition source during fire monitoring, International Journal of Remote Sensing 45(12)(2024) 4178-4197.
https://doi.org/10.1080/01431161.2024.2358451

[13] Y. Zhu, G. Zhang, R. Chu, H. Xiao, Y. Yang, X. Wu, Research on escape route planning analysis in forest fire scenes based on the improved A* algorithm, Ecological Indicators 166(2024) 112355.
https://doi.org/10.1016/j.ecolind.2024.112355

[14] N. Li, Z. Shi, J. Jin, J. Feng, A. Zhang, M. Xie, L. Min, Y. Zhao, Y. Lei, Design of intelligent firefighting and smart escape route planning system based on improved ant colony algorithm, Sensors 24(19)(2024) 6438.
https://doi.org/10.3390/s24196438

[15] Y.Q. Zhang, J.H. Wang, Y. Wang, Z.C. Jia, Q. Sun, Q.Y. Pei, D. Wu, Intelligent planning of fire evacuation routes in buildings based on improved adaptive ant colony algorithm, Computers & Industrial Engineering 194(2024) 110335.
https://doi.org/10.1016/j.cie.2024.110335