Research on Intelligent Obstacle Avoidance Trajectory Planning Method for Unmanned Aerial Vehicles in Urban Complex Scenarios

Chong Cheng[1,a,*], Jiayuan Wang[2, b]

[1]School of Low Altitude Economy, Sichuan Vocational College of Science and Technology, Chengdu, Sichuan Province, 620500 ,China

[2]School of Culture and Tourism, Gingko College of Hospitality Management, Chengdu, Sichuan Province, 611743, China

[a] 898240802@qq.com, [b]jiayuan.wang@gingkoc.edu.cn

*Corresponding author

Keywords: Unmanned Aerial Vehicle, Urban Complex Environment, Intelligent Obstacle Avoidance and Trajectory Planning

Abstract: In response to the problems such as the failure of traditional two-dimensional obstacle avoidance planning in the complex urban building environment, based on theories such as reinforcement learning and deep reinforcement learning, the influencing factors of path planning and obstacle avoidance were analyzed, an energy consumption model for unmanned aerial vehicles was constructed, and an improved reinforcement learning algorithm was proposed, taking into account factors such as environmental wind and air resistance, a dynamic and energy consumption model for unmanned aerial vehicles was constructed, and an improved deep reinforcement learning algorithm was proposed. Through simulation and flight practice, it has been proved that the above algorithms can improve the intelligent obstacle avoidance trajectory planning level of unmanned aerial vehicles in complex urban scenarios.

1. Introduction

The cost of using drones is low and their maneuverability is good, making them widely used in various fields of cities. However, the urban building complexes divide the available airspace, and traditional algorithms often get stuck in an unsolvable state, unable to accurately identify spatial positional relationships, which affects the expansion of urban drone applications. It is necessary to conduct in-depth research on the intelligent obstacle avoidance and trajectory planning methods for drones in complex urban scenarios.

In urban environments with dense obstacles, obstacle avoidance and trajectory optimization for drones are very complex and critical tasks. They have evolved from rule-based algorithms to graphics processing-based algorithms, to bionic intelligent algorithms and various optimization algorithms. Traditional methods are effective in two-dimensional environments and are suitable for situations with fewer obstacles, but they consume a lot of time and have poor real-time performance [1]. Intelligent methods have great potential in path planning in complex environments [2]. The combined method integrates the advantages of multiple methods to compensate for the limitations of a single method, thereby improving the adaptability and efficiency of path planning [3]. Theile et al. proposed an autonomous drone path planning method based on deep reinforcement learning DRL, which integrates the task goal and navigation constraints into the reward function [4]. Tian Guangjian combined deep learning technology with the reinforcement learning framework for autonomous drones, implemented adaptive strategies by extracting features from raw data in complex environments, and ensured safe and efficient flight [5]. When planning three-dimensional paths for drones in urban environments with dense obstacles, optimizing energy consumption is the main goal for improving flight efficiency and task execution capability. Tian Xiaohong proposed an energy consumption model of Q = KxS [6]. Ji Xiang studied the relationships between turns, distances, arc radii, arc lengths, and energy consumption, and based on this, established an energy

  DOI: 10.25236/ehmit.2025.035

consumption estimation model for drone arc paths using the fusion of arc radius, arc length, turn angle, and distance [7].

Domestic and international research on drone obstacle avoidance and planning has made significant progress, but research on urban environments with dense obstacles has been short, and further research is needed in the optimization of algorithms, the speed and effectiveness of control, and other aspects.

This paper constructs a three-dimensional quadrotor drone energy consumption model and plans the drone's trajectory. It adopts the improved reinforcement learning Q-learning algorithm to solve the three-dimensional path optimization problem for drones in complex urban environments.

2. Conceptual Definition and Theoretical Foundation

2.1 Analysis of Unmanned Aerial Vehicle Energy Consumption

The energy consumption in the problem of unmanned aerial vehicle path optimization mainly comes from two aspects: Firstly, it is the flight energy consumption E_{fly} from the current path node to the next mission node; Secondly, it is the hovering energy consumption E_{hover} when conducting data collection at the current node.

$$E_1 = E_{fly} + E_{hover} = \sum_{i=1}^{N}\sum_{j=1}^{N} E_{ij}K_{ij} + \sum_{i=1}^{N} E_i$$

The energy consumption issue of unmanned aerial vehicles (UAVs) in obstacle avoidance and trajectory optimization is one of the core challenges in current UAV technology research. To achieve efficient and safe flight, a balance must be struck between obstacle avoidance capabilities and energy consumption control. The energy consumption of UAVs mainly consists of three components: basic energy consumption E_{base}, wind influence energy consumption E_{wind}, and lift/drop energy consumption E_{alt}.

$$E_2 = E_{base} + E_{wind} + E_{alt} = P_{base} \cdot t + K_{wind}\left|V_{fly} - V_{wind}\right| + mgV_{alt} \cdot t_{alt}$$

P_{base} represents the power required by the unmanned aerial vehicle under normal flight conditions (with no wind and no changes in altitude), t is the total flight time.

K_{wind} is the coefficient that represents the impact of wind on the flight power.

$|V_{fly} - V_{wind}|$ represents the relative difference between wind speed and flight speed, which affects the power demand of the drone.

The energy consumption for ascending and descending is related to the altitude changes during the drone's flight, especially during the ascending and descending operations. Due to the effect of gravity, the drone needs to consume more energy to change the flight altitude. t is the total flight time, and t_{alt} is the time for ascending and descending during the flight.

2.2 The reinforcement learning theory for obstacle avoidance trajectory planning

The reinforcement learning theory applied to intelligent obstacle avoidance trajectory planning for unmanned aerial vehicles in complex urban scenarios is an adaptive decision-making paradigm without a model. The unmanned aerial vehicle (agent) autonomously learns the optimal trajectory strategy through continuous interaction with the dynamic urban environment (the environment). In the low-altitude urban environment with dense obstacles, complex spatial constraints, and dynamic targets, the agent gradually acquires the closed-loop decision-making mechanism of "perception - decision - execution", ultimately achieving the dual goals of safe obstacle avoidance and efficient path planning.

This theory does not require the pre-construction of an accurate environmental model. Instead, it guides the agent to iteratively learn through the design of reasonable reward signals, enabling the unmanned aerial vehicle to gradually master the "perception - decision - execution" closed-loop decision-making mechanism and ultimately achieve the dual goals of safe obstacle avoidance and

efficient path planning. Compared with traditional graph search algorithms (such as A*) and random sampling algorithms (such as RRT*), its prominent advantages lie in dynamic environment adaptability and high-order intelligent decision-making ability, which can effectively handle the complex constraints brought by static and dynamic obstacles such as buildings, signal towers, low-altitude aircraft, pedestrians, and vehicles in the city.

Reinforcement learning is a machine learning method that allows intelligent agents to learn the optimal behavioral strategies through interaction with the environment through trial and error. Its core objective is to maximize long-term cumulative rewards. This process does not rely on labeled data but adjusts the strategy based on environmental feedback (rewards or penalties) after each action. The reward (Reward) and policy (Policy) in the reinforcement learning system are two key components. The reward is the feedback on the effect of the agent's behavior, determining the direction and goal of learning. The policy is the rule that guides the intelligent agent on how to choose actions in different states, and it can be deterministic or stochastic. The process of reinforcement learning is actually continuously optimizing this strategy to achieve higher reward returns.

2.3 Markov Decision Process

The Markov Decision Process (MDP) is the mathematical framework of reinforcement learning theory. It fully describes the decision-making and interaction process of unmanned aerial vehicles through the quadruple (S, A, P, R). The elements are defined as follows:

State set S: It is a continuous or discrete state space, covering the three-dimensional position (x, y, z) of the unmanned aerial vehicle, flight speed, attitude angle, the distance and direction of obstacles sensed by the sensor, remaining energy, and the relative position of the target point, etc., constituting a complete state description of the unmanned aerial vehicle in the urban environment;

Action set A: It is the set of controllable actions that the unmanned aerial vehicle can perform, including the movement directions in the three-dimensional space (front-back, left-right, up-down, and diagonal), step length adjustment, speed and attitude correction, etc., and the trajectory is optimized in real time through continuous or discrete actions;

State transition probability P: It represents the probability of transitioning to a new state $s' \in S$ after executing action $a \in A$ in the state $s \in S$. Mathematically, it is defined as:

$$P(s'|s,a)=P(S_{t+1}=s'|S_t=s,A_t=a)$$

It is used to depict the dynamic uncertainties brought about by the movement of obstacles and environmental noise in urban scenarios;

Reward function R: The core guiding mechanism is defined as R(s,a,s’), including collision penalty (-100), target reward arrival (+100), path distance reduction reward (+5 to +20), energy consumption penalty (-1 to -5). These are balanced through weight adjustment to achieve a balance between flight safety and path efficiency.

The core characteristic of MDP is non-reversibility, that is, the probability distribution of the next state only depends on the current state and the executed action, and is independent of the historical state trajectory. This characteristic significantly simplifies the decision complexity in complex urban scenarios, enabling the agent to make optimal decision reasoning without needing to remember the complete historical trajectory, only based on the current state.

2.4 Bellman's Expectation Equation

The Bellman equation serves as the mathematical foundation for solving the optimal policy of MDP. Its core concept is to decompose the long-term cumulative reward into the immediate reward and the future discounted reward in a recursive relationship. Through iterative updates, it achieves the gradual approximation of the optimal policy. It is mainly divided into two types: value function equation and Q-value function equation:

Value function equation: The state value function $v\pi(s)$ represents the expected cumulative reward when following the policy $\pi(a|s)$ (the probability distribution of actions given the state) in

state s. The Bellman expectation equation for this is:

$$v_\pi = E_\pi[R_{t+1} + \gamma v_\pi(S_{t+1})|S_t = s]$$

Here: $\gamma \in (0, 1)$ is the discount factor, which is used to balance the importance of immediate rewards and future long-term rewards (the closer γ is to 1, the more the agent focuses on long-term gains);

$E_\pi[\cdot]$represents the mathematical expectation under the policy π.

The Q-value function equation: The action value function $Q_\pi(s,a)$ directly characterizes the value of the state-action pair, that is, the expected cumulative reward after executing action a in state s under the policy π. Its Bellman expectation equation is:.

$$Q_\pi(s,a) = E_\pi\left[R_{t+1} + \gamma \max_{a' \in A} Q_\pi(S_{t+1}, a') \middle| S_t = s, A_t = a\right]$$

In the equation, $\max_{a' \in A} Q_\pi(S_{t+1}, a')$ represents the value of choosing the optimal action in the new state s', providing a direct quantitative basis for the action selection of the unmanned aerial vehicle. The intelligent agent gradually approaches the optimal strategy by selecting the action that maximizes $Q_\pi(s,a)$, through continuous iterative updates of the value function or Q-value function. The essence of the Bellman equation is to decompose the long-term optimization problem of "global optimal trajectory planning" into a recursive problem of "single-step optimal action selection". By continuously updating the value function or Q-value function, the unmanned aerial vehicle gradually approaches the optimal strategy π*, ultimately solving the collaborative problem of "local obstacle avoidance" and "global path optimization" in urban scenarios.

3. Research on Obstacle Avoidance Path Planning for Unmanned Aerial Vehicles Based on Q-learning

3.1 Application of Reinforcement Learning in Philosophy

The core philosophy of applying reinforcement learning to unmanned aerial vehicle (UAV) obstacle avoidance is "learning from interaction". Trial-and-error learning and environment adaptation are adopted, completely discarding the traditional algorithm's reliance on an accurate environmental model.

In complex urban scenarios, the UAV uses the ε-greedy greedy strategy to balance exploration of the unknown environment and utilization of known experiences. At the beginning of training, a higher ε probability (e.g., ε = 0.8) is set, prioritizing exploration of unknown areas and accumulating environmental experiences such as obstacle distribution, spatial constraints, and dynamic disturbances; as the training progresses, the ε probability is gradually reduced (e.g., final ε = 0.1), focusing on leveraging the learned optimal strategies to achieve efficient obstacle avoidance.

This trial-and-error learning mechanism enables the UAV to possess extremely strong environmental adaptability, enabling it to quickly adapt to dynamic obstacles in the city, such as pedestrians, vehicles, and sudden low-altitude targets, and also to quickly adapt to static obstacles in the city, such as temporary no-fly zones and construction sites. It does not require manual re-modeling or parameter adjustment.

Goal-oriented reward synergy: The design of the reward function embodies the core philosophy of "prioritizing safety, complementing efficiency, and multi-objective synergy". Through the collaborative construction of R(s,a,s'), the reward function guides the UAV to make rational decisions under complex constraints: collision penalty component (substantial negative reward), fundamentally ensuring flight safety; target arrival reward component (substantial positive reward), driving the task to be completed efficiently; path distance reward and energy consumption penalty component, forming constraints to avoid excessive detours that lead to efficiency loss and energy waste.

As demonstrated by relevant academic research by the Chinese software developer network CSDN, a well-designed reward mechanism can enable the UAV to simultaneously meet the three

core goals in a city with dense obstacles: safety requirements with a collision rate lower than 0.5%; efficiency requirements with the ratio of the planned path length to the theoretical optimal path length not exceeding 1.2; energy constraints with energy consumption reduced by more than 15% compared to traditional algorithms.

3.2 The core framework of the algorithm and the design of random initialization

The random initialization Q-learning path planning algorithm for unmanned aerial vehicles (UAVs) is an extension of the traditional Q-learning method. By introducing a randomization mechanism and adaptive optimization strategies, it addresses issues such as slow convergence speed and local optimal traps in obstacle avoidance trajectory planning in complex urban scenarios. The algorithm uses a discretized urban environment as the environmental model, with the UAV as the agent. Through the iterative update of Q-values for states and actions, it learns the optimal trajectory strategy. The innovation lies in the deep integration of "random initialization" and "dynamic adaptive optimization".

(1) Definition of the environment and state-action space:

Environment modeling: The three-dimensional urban space is discretized into a grid map, with grid states divided into three categories: free space, obstacles (buildings, dynamic targets, etc.), and target points. The grid states are updated in real time by sensors to depict dynamic environmental changes;

State space: Defined as the combination of grid coordinates (x, y, z) and environmental constraint information (obstacle distance, remaining energy), i.e., s = (x, y, z, de, Er), where de is the distance to the nearest obstacle and Er is the remaining energy;

Action space: Includes eight directions of movement (front-back, left-right, up-down, diagonal) and stay actions of the UAV in the three-dimensional space, i.e., $a=\{a_1,a_2,\cdots,a_9\}$, with each action corresponding to a fixed step size transfer between grids.

(2) Random initialization strategy for the Q-table

The initialization of the Q-table directly affects the convergence speed and the quality of the optimal solution of the algorithm. Traditional fixed-value initialization (such as all zeros, random small values) is prone to causing initial exploration blindness or falling into local optima. A hierarchical random initialization strategy is designed:

Base layer initialization: For state-action pairs in non-obstacle areas, uniformly distributed values U(-α, α) are randomly assigned ($\alpha \in [0.5, 1.5]$), avoiding homogeneity of initial Q-values that leads to exploration bias;

Guidance layer initialization: For state-action pairs in the neighborhood of the target point (distance to the target point ≤ 3 grids), on the basis of random values, a positive offset β ($\beta \in [2, 5]$) is added to guide the UAV to explore in the initial direction towards the target;

Constraint layer initialization: For dangerous actions (moving towards obstacles) in the neighborhood of obstacles (distance to obstacles ≤ 2 grids), a fixed negative initial value -10 is assigned to reduce the risk of collision, reducing ineffective trial and error.

This strategy through "randomization + directional guidance" retains the diversity of initial exploration while reducing ineffective trial and error, laying the foundation for subsequent iterative optimization.

3.3 Reward function optimization mechanism based on randomization method

The traditional reward functions mostly adopt fixed weight designs, which have poor adaptability. The random perturbation weighted mechanism dynamically optimizes the weight of reward components through randomization methods, enhancing the algorithm's adaptability to complex urban scenarios. The basic form of the reward function is:

$$R(s,a,s')=\omega_1 R_{\text{coll}}+\omega_2 R_{\text{goal}}+\omega_3 R_{\text{dist}}+\omega_4 R_{\text{energy}}$$

Where: Rcoll: collision penalty (-100); Rgoal: goal achievement reward (+200); Rdist: path distance reward (positively correlated with the reduction in distance to the target point, Rdist takes

values [0-30]); Renergy: energy consumption penalty (negatively correlated with action energy consumption, Renergy takes values [-20, 0]).

Randomization optimization strategy: weight random perturbation. In each iteration, the weights ω_1-ω_4 are randomly perturbed based on the baseline values (ω_1=0.4,ω_2=0.3,ω_3=0.2,ω_4=0.1) according to the normal distribution N(0, 0.05), ensuring the dynamic adaptability of the reward function;

Exploration period reinforcement randomization: within the first 30% of training iteration steps, increase the standard deviation of perturbation to 0.1, encouraging the agent to explore more potential optimal paths; Convergence period reduce the standard deviation of perturbation to 0.02, stabilizing the optimal strategy.

Through random weight adjustment, the reward function can dynamically adapt to environmental changes (such as the appearance of dynamic obstacles, changes in energy constraints), avoiding the rigidity of fixed weights in strategies.

3.4 Elite strategy enhances algorithm performance

To further accelerate convergence and improve the quality of the optimal path, the elite strategy is introduced. By retaining the high-quality experiences during the iterative process, the algorithm can quickly approach the optimal solution: Elite experience selection: Every 10 iterations, the cumulative rewards of all executed state-action pairs (s,a) are counted.

$$R_{total}(s,a) = \sum_{t=0}^{T} \gamma^t R(s_t,a_t,s_{t+1})$$

Select the top 20% state-action pairs with the highest cumulative rewards as "elite experiences";

Select the top 20% state-action pairs with the highest cumulative rewards as "elite experiences"; Experiments show that this strategy can reduce the number of algorithm convergence iterations by more than 35%, while reducing the distance and energy cost of the optimal path.

(1) Complete algorithm flow

1) Initialization: Discretize the urban environment, define the state-action space, initialize the Q-table using a hierarchical random strategy, set the initial learning rate α0, discount factor γ = 0.9, and maximum number of iterations Kmax;

2) State observation: The unmanned aerial vehicle perceives the current state sk, including position, distance to obstacles, and remaining energy;

3) Action selection: Based on the ε-greedy strategy, select the action a_k. Combining the elite strategy, we preferentially select elite actions;

4) Reward calculation: After executing action ak, transition to the new state sk+1, and calculate the immediate reward Rk using the randomized reward function;

5) Q-value update: Use a gradually decreasing learning rate α (k), and update the Q-table according to the following Q Bellman expectation equation:

$$Q(s_k,a_k) = Q(s_k,a_k) + \alpha(k)\left[R_k + \gamma \max_{a'} Q(s_{k+1},a') - Q(s_k,a_k)\right]$$

6) Termination criterion: If the target point is reached or the maximum number of iterations is exceeded, the iteration ends; otherwise, return to step 2 until convergence is achieved.

(2) Performance advantages

Convergence speed is faster: Hierarchical random initialization reduces initial blind exploration, gradually reduces the learning rate to avoid oscillation in the later stage, and the elite strategy reuses high-quality experience. The synergy of these three elements increases the algorithm's convergence speed by 30%-40%.

Path quality is better: Randomized reward function adapts to dynamic environments, the elite strategy strengthens the guidance of the optimal path, and the collision rate of the planned path is lower than 0.3%. The ratio of the path length to the theoretical optimal value is $\leqslant$ 1.15, and energy consumption is reduced by more than 20%.

Environmental adaptability is stronger: The randomized mechanism and dynamic learning rate

enable the algorithm to quickly respond to changes in dynamic obstacles, temporary no-fly zones, etc. in the city. No re-modeling is required, and its robustness is significantly superior to traditional Q-learning.

4. Research on Urban Dense Obstacle-Avoidance for Unmanned Aerial Vehicles and Trajectory Optimization Based on Deep Reinforcement Learning DQN Algorithm

4.1 Further improvements of the Q-learning algorithm

The Q-learning algorithm RIRQL performs well in static and simple dynamic scenarios, but it struggles to handle continuous states and actions in complex urban environments, and has poor adaptability; moreover, the complex distribution of obstacles can easily lead to computational crashes and low search efficiency, requiring further improvement.

The Deep Q-Network (DQN) algorithm is an unsupervised adaptive algorithm that combines the feature extraction ability of deep learning with the reinforcement learning decision-making mechanism of traditional Q-learning. It is mainly used to solve the core pain points of "high state space dimension and complex dynamic environment" in urban dense obstacle scenarios, where drones avoid obstacles and optimize trajectories - breaking the limitations of traditional Q-learning algorithms that rely on Q-tables to store state-action values, and using deep neural networks (DNN) to fit the Q-value function to achieve optimal action decisions in high-dimensional states.

The core architecture of the DQN algorithm consists of two key modules: one is the experience replay buffer (Replay Buffer), which is used to store the state-action-reward-next state (s,a,r,s') experience data generated by the interaction between the drone and the environment. Through random sampling, it breaks the correlation between experiences and avoids overfitting and training oscillations during neural network training; the other is the target Q-network (Target Q-Network), which is structurally identical to the current Q-network and updates parameters after a certain period to reduce the correlation between current Q-value updates and target Q-values, improving training stability.

The core update formula of the DQN algorithm is as follows:

$$Q_\theta(s,a) \leftarrow Q_\theta(s,a) + \alpha\left[r + \gamma \max_{a'} Q_{\theta'}(s',a') - Q_\theta(s,a)\right]$$

Among them, θ represents the parameters of the current Q network, θ' represents the parameters of the target Q network, α is the learning rate, γ is the discount factor, and $\max_{a'} Q_{\theta'}(s',a')$ represents the optimal action value corresponding to the next state s'. In the urban dense obstacle scenario, the unmanned aerial vehicle acts as an agent. It perceives the environmental state (obstacle distribution, its own position, wind speed, etc.) through sensors, inputs it into the neural network to obtain the Q values of each action, selects the optimal action to execute, and then adjusts the network parameters through the reward function to gradually learn the optimal strategy for obstacle avoidance and trajectory optimization.

4.2 The application of random obstacle training methods

The core feature of the urban dense obstacle scenario is the random and dynamic distribution of obstacles (such as pedestrians, vehicles, temporary construction areas, etc.). Traditional fixed obstacle training methods will lead to insufficient generalization ability of the algorithm and make it unable to adapt to the uncertainty of real urban scenarios. Therefore, this paper integrates the random obstacle training method into the training process of the DQN algorithm, with the core objective of improving the robustness of the drone's obstacle avoidance in a randomly dense obstacle environment. The specific application methods are divided into three levels:

We employ a grid-based approach to construct a three-dimensional urban environment model, classifying obstacles into static random obstacles and dynamic random obstacles: static random obstacles (such as buildings, signal towers) are randomly generated according to the actual layout probability of the city to ensure that the obstacle density and distribution location conform to the characteristics of the real dense urban scenario; dynamic random obstacles (such as moving vehicles,

pedestrians) move within the grid according to random speed and random trajectories, with the speed range set at 0.5-5m/s, and the trajectory is generated using a random walk model to simulate the uncertainty of dynamic obstacles in the real scenario.

Hierarchical random training strategy, in the initial training (1-30% of the iteration steps): set a low obstacle density (grid obstacle proportion 20%-30%), mainly static random obstacles, with the proportion of dynamic obstacles not exceeding 10%, focusing on allowing the drone to learn basic obstacle avoidance actions, quickly mastering the core logic of "perceiving obstacles - avoiding obstacles", and accumulating basic experience data;

Mid-term training (31%-70% of the iteration steps): gradually increase the obstacle density (grid obstacle proportion 40%-60%), increase the proportion of dynamic random obstacles to 30%-50%, randomly adjust the obstacle movement speed and trajectory, guiding the drone to learn dynamic obstacle avoidance strategies, and improving its adaptability to environmental changes;

Late-term training (71%-100% of the iteration steps): maintain a high obstacle density (grid obstacle proportion 60%-70%), with the proportion of dynamic obstacles remaining stable at 50%, and randomly add extreme scenarios such as sudden obstacle appearance and trajectory mutation to strengthen the algorithm's robustness, ensuring that the drone can still stabilize obstacle avoidance in complex random environments.

Training optimization and supplementation, in the sampling process of the experience replay pool, introduce a random obstacle experience weighting mechanism, giving higher sampling weights to the experience samples with high random obstacle density and intense dynamic changes (weight coefficient 1.2-1.5), to enhance the contribution of such difficult samples to the neural network training, and further enhance the algorithm's adaptability to the urban dense obstacle scenarios with randomness.

4.3 Construction of Energy Consumption Model for Unmanned Aerial Vehicles with Wind Disturbance

In urban low-altitude environments, wind disturbances (such as gusts and turbulence) are key factors affecting the energy consumption and trajectory stability of unmanned aerial vehicles (UAVs). Traditional studies often ignore the influence of wind disturbances, resulting in higher energy consumption and larger trajectory deviations of the algorithms in real scenarios. This paper incorporates wind disturbance factors into the UAV energy consumption model and simultaneously optimizes the design of the reward function to achieve the triple goals of "obstacle avoidance safety, optimal trajectory, and lowest energy consumption".

The energy consumption of UAVs mainly comes from the energy consumed by the propulsion system to overcome air resistance, its own gravity, and wind disturbances. Considering the random characteristics of wind disturbances in urban low-altitude environments (wind speed range of 0-8 m/s, random changes in wind direction), the energy consumption model including wind disturbances is constructed as follows:

$$E = \int_0^T \left(F_{lift} v + F_{drag} v + F_{wind} v\right) dt$$

Among them, E represents the total energy consumption of the drone, T represents the flight time of the trajectory, v represents the flight speed of the drone, F_{lift} represents the lift force, F_{drag} represents the air resistance, and F_{wind} represents the resistance caused by wind disturbances (which is positively correlated with wind speed and wind direction. When the wind direction is opposite to the flight direction, F_{wind} takes a positive value, increasing the energy consumption; when the wind direction is the same as the flight direction, F_{wind} takes a negative value, reducing the energy consumption).

To simplify the calculation and adapt to the training of the DQN algorithm, the energy consumption model is discretized into the energy consumption values for each action step. That is, each time an action is executed, the energy consumption increment for this step is calculated based on the current wind speed and wind direction, and the cumulative total energy consumption is obtained, which serves as one of the core bases for the subsequent reward function design.

The reward function adopts a multi-dimensional weighted form, taking into account the safety of obstacle avoidance, trajectory optimization, and energy economy, and integrating the influence of wind disturbances. It guides the drone to avoid obstacles while choosing the path with the lowest energy consumption and the smoothest trajectory. The specific form is as follows:

$$R(s, a, s', w) = w_1 R_{coll} + w_2 R_{goal} + w_3 R_{traj} + w_4 R_{energy}(w)$$

The definitions of each component and the weight settings (optimized in combination with the urban dense obstacle scenario) are as follows:

Collision penalty term R_{coll}: If the drone collides with an obstacle after performing an action, a fixed negative reward of -200 is given; if no collision occurs, the value is 0. The weight $w_1 = 0.4$, prioritizing flight safety;

Target arrival reward term R_{goal}: If the drone reaches the target point, a fixed positive reward of +300 is given; if not reached, the value is 0. The weight $w_2 = 0.3$, driving the drone to complete the trajectory planning task;

Trajectory smoothness reward term R_{traj}: It is negatively correlated with the change in the drone's action angle. The smaller the angle change (the smoother the trajectory), the higher the reward value, with a range of 0-50, used to avoid the drone's sharp turns and improve trajectory stability; the weight $w_3 = 0.1$;

Energy consumption reward term $R_{energyw}$: Combined with the influence of wind disturbances, it is negatively correlated with the incremental energy consumption per step. The lower the energy consumption and the more reasonable the utilization of wind disturbances (such as flying with the wind), the higher the reward value, with a range of -80 to 40; the weight $w_4 = 0.2$, guiding the drone to reduce energy consumption.

At the same time, to adapt to the randomness of wind disturbances, the weights of the reward function will be dynamically adjusted according to the real-time wind speed: the greater the wind speed, w_4 is appropriately increased (up to 0.3), w_3 is also increased simultaneously, prioritizing energy economy and trajectory stability; when the wind speed is relatively small, the baseline weights are maintained, and the focus is on optimizing obstacle avoidance and target arrival efficiency.

5. Simulation analysis and comparative experiments

5.1 Simulation analysis

To verify the effectiveness of the DQN algorithm proposed in this paper (including random obstacle training, wind disturbance energy consumption model, and optimized reward function) in urban dense obstacle drone obstacle avoidance and trajectory optimization, simulation experiments and comparative experiments were designed to clarify the performance advantages of the algorithm. The experimental environment and details are as follows.

(1) Simulation environment setup

1) Simulation platform: The DQN algorithm training and simulation platform was built using the PyTorch framework, combined with the powerful open-source 3D robot simulator Gazebo. A three-dimensional urban dense obstacle scene was constructed with a grid size of 1m×1m×1m, and the simulation area was 50m×50m×20m (consistent with the urban low-altitude flight range);

2) Drone parameters: A multi-rotor drone was selected, with a flight speed of 2-5m/s, a maximum endurance time of 600s, a sensor perception range of 10m, and the ability to collect information such as obstacle distance, wind speed, and wind direction in real time;

3) Environmental parameters: The random obstacle density was set at 60%-70% (simulating the urban dense obstacle scenario), and the dynamic obstacle movement speed was 0.5-5m/s; the wind disturbance was set as gusts, with wind speed ranging from 0 to 8m/s and random wind direction changes;

4) Algorithm parameters: The initial learning rate α was set at 0.8, and it was gradually reduced to 0.1 using an exponential decay strategy; the discount factor γ was set at 0.9; the capacity of the

experience replay pool was 100,000; the update cycle of the target Q network was 100 steps; the maximum number of iterations was 50,000 steps.

(2) Simulation analysis

The simulation analysis focused on the convergence, obstacle avoidance success rate, trajectory quality, and energy consumption performance of the algorithm. The core results are as follows:

1) Convergence analysis: The algorithm reached stable convergence after approximately 35,000 iterations. After convergence, the fluctuation range of Q values was controlled within a very small range, indicating that the random obstacle training method and dynamic weight reward function effectively improved the training stability and avoided training oscillations;

2) Obstacle avoidance success rate: In the dense random obstacles + wind disturbance environment, the obstacle avoidance success rate reached 98.5%. Only in extreme scenarios (wind speed ≥ 7m/s, sudden dense appearance of obstacles) did 2 collisions occur, indicating that the algorithm has a strong adaptability to urban dense obstacle scenarios;

3) Trajectory quality: The smoothness of the planned trajectory was improved by 32% compared to the traditional DQN algorithm. The trajectory deviation (from the optimal path) was controlled within 0.5m, and the trajectory offset caused by wind disturbance was ≤ 0.3m, meeting the requirements for precise urban drone flight;

4) Energy consumption performance: The algorithm with the wind disturbance energy consumption model had a total energy consumption 28% lower than the traditional algorithm. Especially in the tailwind scenario, the energy consumption reduction was more obvious, achieving a coordinated improvement in energy consumption and trajectory optimization.

5.2 Controlled experiment

Three mainstream algorithms were selected as the comparison objects: the traditional DQN algorithm (without random obstacle training and without wind disturbance energy consumption model), the Q-learning algorithm, and the RRT* algorithm. Comparative experiments were conducted in the same simulation environment. The core performance indicators are compared as follows (units: mean ± standard deviation):

1) Convergence iteration steps: The algorithm in this paper (35200 ± 800) < Traditional DQN algorithm (42500 ± 1200) < Q-learning algorithm (58000 ± 1500) < RRT* algorithm (no definite convergence steps, requires real-time planning);

2) Obstacle avoidance success rate: The algorithm in this paper (98.5% ± 1.2%) > Traditional DQN algorithm (89.3% ± 2.5%) > RRT* algorithm (85.7% ± 3.1%) > Q-learning algorithm (78.9% ± 4.2%);

3) Average energy consumption: The algorithm in this paper (125 ± 15J) < Traditional DQN algorithm (174 ± 22J) < RRT* algorithm (198 ± 25J) < Q-learning algorithm (212 ± 28J);

4) Trajectory smoothness: The algorithm in this paper (mean angle change 8.2° ± 2.1°) < Traditional DQN algorithm (12.5° ± 3.3°) < RRT* algorithm (15.7° ± 4.1°) < Q-learning algorithm (18.9° ± 4.5°).

The results of the comparative experiments show that the unmanned aerial vehicle obstacle avoidance and trajectory optimization method based on the DQN algorithm proposed in this paper has improved the generalization ability and obstacle avoidance success rate of the algorithm through the random obstacle training method. By incorporating wind disturbance into the energy consumption model and reward function, it effectively reduces the energy consumption of the unmanned aerial vehicle and improves the trajectory stability. Compared with the traditional DQN, Q-learning, and RRT* algorithms, this algorithm has significant advantages in convergence speed, obstacle avoidance performance, energy consumption economy, and trajectory quality, and can effectively adapt to low-altitude flight scenarios with dense obstacles in urban areas.

6. Conclusion

Due to the complex distribution and height differences of buildings in the city, the characteristics of the drone's flight operation are that there are many path nodes, and it faces two major difficulties:

path optimization and obstacle avoidance. It needs to plan the path in a three-dimensional space.

Based on the Q-learning reinforcement learning theory and the energy consumption model characteristics of drones in urban dense obstacles, an improved reinforcement learning algorithm named RIQL (Random Initialization Q-Learning) was proposed.

A simulation comparison experiment was conducted to compare the performance of the improved RIQL algorithm with that of the traditional Q-learning algorithm. The RIQL algorithm demonstrated good system stability and performance, effectively reducing the path length and energy consumption of the drone.

Further improving the RIQL algorithm, factors such as environmental wind and air resistance were included in the calculation scope, and the drone dynamics and energy consumption model were constructed. An improved DDQN algorithm was innovatively proposed.

Through simulation experiments and actual flight tests, the execution efficiency of the RIQL algorithm and the DDQN algorithm in the path planning task in the 3D urban environment was verified. The experimental results fully proved that these algorithms could achieve path optimization under multi-dimensional constraints, providing reliable technical and theoretical support for the autonomous flight planning and route planning of drones.

Acknowledgment

The author expresses heartfelt gratitude to the following sponsors for their support in this research: The soft science project of the Sichuan Provincial Science and Technology Department, titled "Research on the Path and Influencing Factors of Promoting Major Technological Transformations through Cheng-Mei Synergy", 24RKX0065. The high-end talent introduction project of the Sichuan Provincial Science and Technology Department, titled "Research on the Intelligent Networked Physical Entity Numerical Control System Based on Physical Motion Laws", 25RCYJ0011.

References

[1] Wang J, Li Y, Li R, et al. Trajectory planning for UAV navigation in dynamic environ-ments with matrix alignment Dijkstra[J]. Soft Computing, 2022, 26(22): 12599-12610.

[2] Kladis G P, Doitsidis L, Tsourveloudis N C. Energy-Efficient Path-Planning for UAV Swarm Based Missions: A Genetic Algorithm Approach[C]//2024 International Conference on Unmanned Aircraft Systems (ICUAS). IEEE, 2024: 458-463.

[3] Yue C, Zhongliang Z, Jiancheng J, et al. Path planning for support jammers formation in penetration operation based on improved PSO-GA[C]//2017 2nd International Con-ference on Image, Vision and Computing (ICIVC). IEEE, 2017: 1090-1096.

[4] Theile M, Bayerlein H, Nai R, et al. UAV path planning using global and local map information with deep reinforcement learning[C]//2021 20th International Conference on Advanced Robotics (ICAR). IEEE, 2021: 539-546.

[5] Tian Guangjian, Dai Jiyang, Ying Jin, et al. Multi-UAV Trajectory Planning Based on Adaptive Segmented Potential Field Method [J]. Journal of System Simulation, 2022, 34(11): 2368-2376.

[6] Tian Xiaohong. Construction and Application Research of Energy Consumption Model for Transmission Tower Inspection by Quadrotor UAV [D]. Guangxi University, 2021.

[7] Ji Xiang. Research on Energy Optimization Path Planning Method for Multi-feature Fusion of Helicopter UAV [D]. Northwest University, 2021.