Gao *et al. Complex Eng Syst* 2023;3:21
DOI: 10.20517/ces.2023.24

Complex Engineering Systems

Research Article

Open Access

Learning based multi-obstacle avoidance of unmanned aerial vehicles with a novel reward

Haochen Gao[1], Bin Kong[2], Miao Yu[3], Jinna Li[1]

[1]School of Information and Control Engineering, Liaoning Petrochemical University, Fushun 113000, Liaoning, China.
[2]School of Artificial Intelligence and Software, Liaoning Petrochemical University, Fushun 113000, Liaoning, China.
[3]Group of Physics, Shenyang Manchu Junior High School, Shenyang 110031, Liaoning, China.

Correspondence to: Prof. Jinna Li, School of Information and Control Engineering, Liaoning Petrochemical University, No. 1, West Section of Dandong Road, Wanghua District, Fushun 113000, Liaoning, China. E-mail: lijinna_721@126.com

How to cite this article: Gao H, Kong B, Yu M, Li J. Learning based multi-obstacle avoidance of unmanned aerial vehicles with a novel reward. *Complex Eng Syst* 2023;3:21. http://dx.doi.org/10.20517/ces.2023.24

Received: 20 Aug 2023 **First Decision:** 10 Oct 2023 **Revised:** 20 Nov 2023 **Accepted:** 21 Nov 2023 **Published:** 11 Dec 2023

Academic Editors: Hamid Reza Karimi, Ding Wang **Copy Editor:** Fangling Lan **Production Editor:** Fangling Lan

Abstract

In this paper, a novel reward-based learning method is proposed for unmanned aerial vehicles to achieve multi-obstacle avoidance. The Markov jump model was first formulated for the unmanned aerial vehicle obstacle avoidance problem. A distinctive reward shaping function is proposed to adaptively avoid obstacles and finally reach the target position via an optimal approach such that an adaptive Q-learning algorithm called the improved prioritized experience replay is developed. Simulation results show that the proposed algorithm can achieve autonomous obstacle avoidance in complex environments with improved performance.

Keywords: UAVs, multi-obstacle avoidance, adaptive Q-learning

1. INTRODUCTION

The rapid advancement of artificial intelligence and computer technology has significantly facilitated the extensive adoption of unmanned aerial vehicles (UAVs) in diverse domains, encompassing civil, commercial, and military sectors. Collision avoidance of UAVs continues to be a significant concern due to its direct implications on safety and the successful completion of tasks[1–3]. Therefore, there has been considerable research interest in collision avoidance techniques, which are designed to mitigate the occurrence of collisions between

UAVs and other objects[4].

The Dijkstra algorithm is adopted to address the issue of collision avoidance among UAVs and solve the shortest path problem in a power graph[5]. In three-dimensional (3D) space, the A* (A-star) algorithm is used to evaluate the generation value of scalable waypoints within the path region using heuristic functions. The value of each generation was then compared with the search operation time and the cost of distance for waypoints in order to identify the optimal path[6]. The Rapidly Exploring Random Trees (RRT) algorithm generates a search tree by randomly selecting leaf nodes and subsequently extends the search tree to cover the entire search space in order to identify the desired path[7]. One study utilized the artificial potential field method to address the issue of UAV movement in the presence of targets and obstacles. This method involves converting the influence of the target and obstacle on the UAV's movement into an artificial potential field. By utilizing the gravitational and repulsive forces between the UAV and the obstacle, the researchers were able to effectively control the UAV's movement towards the target point, following the negative gradient of the potential field[8]. The aforementioned methods pertain to the category of classical collision avoidance techniques. Although it is possible to achieve collision avoidance between UAVs and obstacles, these methods are generally limited in their applicability to addressing collision avoidance tasks in simple environments. In the presence of intricate challenges, the task of resolving them becomes increasingly arduous as the dimensions of the system expand, thereby imposing certain constraints.

With the advancement of computer and big data technologies, a variety of intelligent algorithms, including ant colony algorithms, genetic algorithms, and particle swarm optimization algorithms. In a previous study[9], the researchers employed the ant colony algorithm, which utilized pheromone as a guiding factor in the decision-making processes of ants for route selection. This approach enabled the ants to concentrate their efforts on identifying the most optimal path. Genetic algorithms can also be used to address the obstacle avoidance issue in UAVs[10]. On the basis of the particle swarm optimization algorithm, the authors[11] proposed a 3D path planning algorithm for UAV formation. The algorithm incorporates comprehensive improvements to automatically generate optimized flying paths. Researchers have extensively examined the benefits of the genetic algorithm and particle swarm optimization algorithm. Furthermore, they have integrated these two algorithms to compute the feasible and quasi-optimal trajectory for a fixed-wing UAV in a complex 3D environment[12]. The above smart algorithms can theoretically find the optimal solution of the path, which is suitable for solving UAV collision avoidance and path planning problems in stationary environments. In practical terms, the environment is typically characterized by its dynamic and unknown nature. Reinforcement learning (RL), as an emerging research trend, has demonstrated significant potential in the domains of aircraft collision avoidance and route planning through intelligent interaction with the surrounding environment.

SARSA and Q-learning methods can enhance the obstacle avoidance ability of UAVs and optimize the calculation of the shortest path. Both methods are model-free algorithms that aim to eliminate reliance on the environmental model and make action selections based on the values associated with all available actions[13,14]. Despite possessing distinctive characteristics and benefits, classical RL methods demonstrate significant limitations when confronted with high-dimensional motion environments and multi-action inputs, particularly as the complexity of the mission environment for UAVs increases. Mnih *et al.*, (2013) were the first to propose the integration of RL with neural networks (NN) by utilizing them to approximate the Q function[15]. Afterward, there have been significant advancements in RL methods, such as the Deep Q-network (DQN) RL-based approaches. These developments have led to the creation of deep RL algorithms that effectively address decision-making problems in complex and continuous dynamic systems[16,17]. Therefore, deep RL can enable the UAV to continuously improve its obstacle avoidance strategy by interacting with the environment, thereby enabling it to have autonomous learning capabilities. The deep RL model can adapt to these complex environments through continuous trial and error learning, which means that UAVs can better adapt to different environments and tasks.

It is noteworthy to mention that the reward component in the RL method plays a pivotal role in enhancing performance. Inadequate reward configurations not only result in insufficient data and sparse rewards but also affect the intended efficacy of UAV collision avoidance. Notice that the reward is typically formulated by artificially assigning predetermined values. The adaptive rewarding or punishing of UAVs in order to optimize the prescribed performance while successfully avoiding collisions remains an unresolved matter. In this paper, a novel RL method is proposed, which incorporates an adaptive reward setting to effectively tackle the aforementioned issue. Experiments demonstrate that the proposed method has the capability to achieve autonomous obstacle avoidance for UAVs in complex and unfamiliar environments, leading to a substantial enhancement in UAV performance. The primary contributions of this paper are outlined as follows:

1. Compared to conventional optimization control methods utilized for obstacle avoidance in UAVs[9–12], the developed RL method has the capability to dynamically learn the optimal control strategy in an unfamiliar environment. This enables the UAV to effectively avoid obstacles and ultimately reach the target destination using an approximately optimal approach.

2. A new reward function has been developed, which is combined with adaptive weights to guide the agent's focus towards important aspects of the task. This approach effectively balances obstacle avoidance and reaching the target point.

The subsequent sections of this paper are structured as follows. Section 2 outlines the problem of collision avoidance for UAVs. Section 3 introduces a novel self-learning method for UAVs to avoid obstacles, incorporating an adaptive changing reward function. In Section 4, this paper presents numerical simulation results to validate the effectiveness of the developed method and analyze the advantages of the new reward function. Finally, the conclusions are presented in Section 5.

2. PROBLEM FORMULATION

Consider a UAV obstacle avoidance problem where the goal is to reach a target point in a bounded 3D region and avoid obstacles in an unknown environment. Assume that the obstacles in the search region are both static and unknown a priori.

2.1. UAV model

The kinematic model of the UAV is presented as follows:

$$\begin{aligned} \dot{x} &= V_u \cos(\theta)\cos(\psi) \\ \dot{y} &= V_u \cos(\theta)\sin(\psi) \\ \dot{z} &= V_u \sin(\theta) \end{aligned} \tag{1}$$

where (x, y, z) is the 3D position coordinate of the UAV, V_u is the speed of the UAV, and θ and ψ are the UAV vertical pitch angle and the horizontal heading angle, respectively. The above angles can be clearly seen in Figure 1.

To avoid the multiple obstacles in environments, we will monitor the nearest obstacle to the UAV. The following three detection ranges according to the distance $d_m(k)$ between the UAV and the obstacle are defined:

- $R_m < d_m(k)$, this area is known as a safe zone, where no obstacles pose a threat to the UAV.
- $R_s < d_m(k) \leq R_m$,this area is known as the danger zone, where the UAV may collide with the obstacle.
- $d_m(k) \leq R_s$, this area is known as the collisional zone, implying that there is an obstacle in this zone and the UAV will collide with the obstacle; where R_s and R_m are the radiuses of collisional spherical zones,

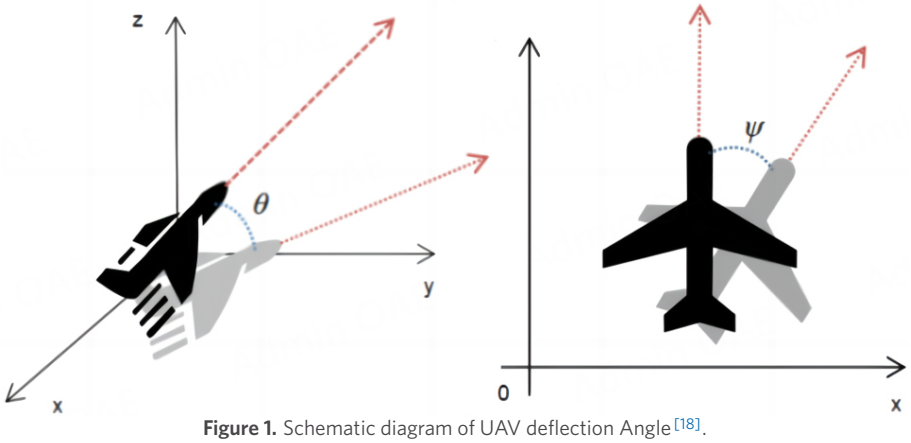

Figure 1. Schematic diagram of UAV deflection Angle[18].

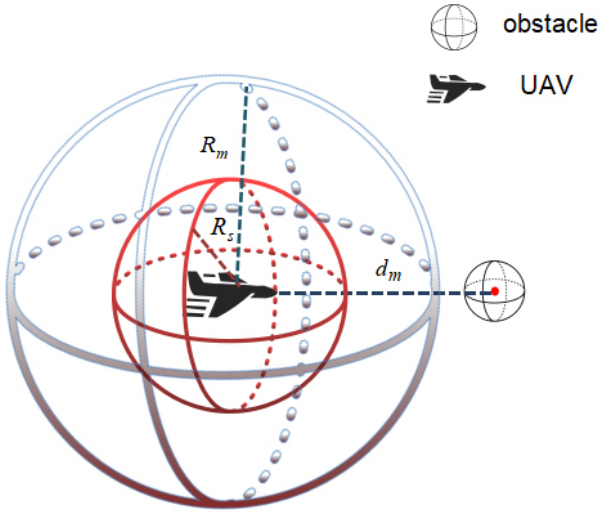

Figure 2. Schematic diagram of distance between UAVs and obstacles[18].

R_s represents the minimum safe distance radius between the drone and obstacles, and R_m represents the maximum safe distance radius between the drone and obstacles. $d_m(k)$ denotes the distance from the UAV to the obstacle. $k(k = 0, 1, 2, ...)$ is the sampling time. The aforementioned detection ranges can be clearly seen in Figure 2. The distance formula for the UAV and the obstacle is as follows:

$$d_{m(i)} = \sqrt{\left(ob_{x_i} - x_k\right)^2 + \left(ob_{y_i} - y_k\right)^2 + \left(ob_{z_i} - z_k\right)^2} \tag{2}$$

where $(ob_{x_i}, ob_{y_i}, ob_{z_i})$ represents the position coordinates of the i-th obstacle, and (x_k, y_k, z_k) represents the position coordinates of the UAV at time k .

The overall goal of this paper is to self-learn control decisions for the UAV such that it can travel on obstacle-free paths to the prescribed target destination.

3. SELF-LEARNING OBSTACLE AVOIDANCE ALGORITHM

In this section, we are going to present a self-learning framework based on the DQN algorithm for the obstacle avoidance problem of UAVs in 3D space. Firstly, the anti-collision problem of UAV is formulated as a Markov decision process (MDP), and then, a DQN algorithm with the novel reward is developed for decision-making

such that the UAV can reach the target point without collision with the obstacles.

3.1 MDP formulation of UAV obstacle avoidance

The problem of UAV obstacle avoidance can be formulated as an MDP (S, A, P, R), where S represents the state space, A is the action space, P denotes the state transition probability, and R stands for the immediate reward obtained by the UAV. In the environment of UAV obstacle avoidance, the state space, action space, and reward function are explained in detail below.

(1) State: The UAV state consists of the current coordinates of the UAV in the 3D coordinate system, that is

$$S = \{s_k\}, s_k = (x_k, y_k, z_k) \tag{3}$$

(2) Action: The horizontal and vertical heading angles of the UAV are assumed to change simultaneously. Angle variations are indicated by the angular velocity:

$$\begin{cases} \Delta\theta = \omega_{c_1} \Delta k \\ \Delta\psi = \omega_{c_2} \Delta k \end{cases} \tag{4}$$

where $\Delta\theta$ and $\Delta\psi$ represent the two angle variations of the UAV, and ω_{c_1} and ω_{c_2} are the horizontal heading angular velocity and vertical heading angular acceleration, respectively. To consider a realistic flight scenario for the UAV, the following constraints on the two angular velocities are taken into account:

$$\begin{cases} \omega_{c_1 \min} \le \omega_{c_1} \le \omega_{c_1 \max} \\ \omega_{c_2 \min} \le \omega_{c_2} \le \omega_{c_2 \max} \end{cases} \tag{5}$$

The navigational angle of the UAV in 3D space is controlled by varying its horizontal and vertical directional angular velocities. Therefore, the action a_k of the UAV at time k is defined as follows:

$$a_k = (\theta_k, \psi_k) \, (a_k \in A) \tag{6}$$

All actions compose the space of actions A.

(3) Reward: It is crucial to design the reward function for efficiently avoiding the obstacles and reaching the prescribed target point. The design of the reward function of traditional RL algorithms is usually relatively simple, requiring an artificial setting of weights to achieve a balance between different rewards, thereby encouraging the agent to complete the training task, which has certain limitations. Considering the overall UAV navigation cost and task, a novel reward function with a weight ω_1 is designed. The weight ω_1 can be adaptively adjusted based on the distance between the UAV and the obstacle. This allows the UAV to effectively balance reaching the target point and avoiding obstacles. The reward function r with weights is proposed as follows:

$$\begin{aligned} r(k) = \omega_1 r_{avoidance}(k) + (1 - \omega_1) r_{distance}(k) \\ + r_{deflection}(k) \end{aligned} \tag{7}$$

where $r_{avoidance}$ represents the reward for avoiding obstacles, $r_{distance}$ represents the reward for distance loss, and $r_{deflection}$ represents the reward for deflection angle loss. ω_1 represents the weight coefficient for the obstacle avoidance reward, and $1-\omega_1$ represents the weight coefficient for the distance reward.

From (7), one can find that there are three parts in the reward, and they are respectively responsible for obstacle avoidance, the target point, and energy consumption caused by actions. The parameter ω_1 takes charge of allocating the weights. The definition of ω_1 is given below:

$$\omega_1 = \begin{cases} 1 & d_m \le R_s \text{ or fly out of the map} \\ l & R_s < d_m \le R_m \\ 0 & else \end{cases} \tag{8}$$

where l is an adaptive weight parameter and is given below:

$$l = e^{-\left[\frac{\left(d_m^2 - R_S^2\right)}{\left(R_m^2 - R_S^2\right)}\right]} \tag{9}$$

The obstacle avoidance module reward function is as follows:

$$r_{avoidance}(k) = \begin{cases} -b & d_m \le R_s \text{ or fly out of the map} \\ -c & R_s < d_m \le R_m \\ 0 & d_m > R_m \end{cases} \tag{10}$$

where $b > c > 0$, this is because, in the obstacle avoidance task, when the UAV is in the collisional zone or the outside zone of the map, it indicates that the UAV has failed in the obstacle avoidance task. At this time, it needs to be given a large punishment so as to avoid this situation in the next training.

The reward function of the distance loss is as follows:

$$r_{distance}(k) = \begin{cases} -d \times dis & dis \ge 1 \\ f & dis \le 1 \end{cases} \tag{11}$$

where both d and f belong to positive real numbers greater than 0. dis represents the distance between the UAV and the target point at time k.

The reward function for deflection angle loss is given below:

$$r_{deflection}(k) = -(\sin|\Delta\psi| + \sin|\Delta\theta|) \tag{12}$$

Remark 1: The design of the reward function plays an important role in the learning and training of UAV collision avoidance problems, which also determines the effectiveness and efficiency of NN training. In existing approaches to reward setting[19], the reward function is mostly defined as a fixed positive reward value when the UAV's next state is closer to the goal point after the UVA has executed the action. Alternatively, a fixed

negative reward value is given for punishment. For collision avoidance problems, the drawback is that it is not possible to quantitatively characterize the impact of the current chosen action on its future. Moreover, that is a subjective decision and cannot guarantee the performance optimality. The improved deep Q network algorithm, based on prioritized experience replay (PER), proposed in this article, utilizes adaptive weights to dynamically adjust the reward function. Compared to the traditional DQN and PER algorithms, the improved PER (IPER) algorithm can accelerate learning convergence speed and improve sample efficiency by better handling sparse reward signals in tasks and incorporating important experiences.

Remark 2: From (8) and (9), one can notice that the weight ω_1 can balance the obstacle avoidance and the desire of reaching the target point. The weight ω_1 is time-varying according to the distance between the UAV and the obstacles. Therefore, this adaptive reward design method is the first time to be put forward, such that when the UAV is in the collisional zone, the value of ω_1 is 1; When the UAV is in the danger zone, the value of ω_1 is l, the farther away the UAV is from this area, the smaller the value of ω_1 is; When the UAV is in the safe zone, the value of ω_1 is 0.

3.2 DQN algorithm design based on prioritized experience replay

An important component of the DQN algorithm is experience replay, which stores data generated from past learning into an experience pool and randomly draws past data from the experience pool for learning in the next network training. In this way, serial correlations can be broken, and past experiences can be reused. However, the random sampling approach also leads to wasted experience, so the PER approach can effectively address this issue[20]. PER means that when extracting experience, the most valuable experience is extracted first with a certain probability to avoid overfitting. Therefore, compared to traditional experience replay, PER assigns a priority value δ_k to each transition. It then determines different sampling probabilities and learning rates based on δ_k in order to achieve non-uniform sampling. δ_k denotes the TD error represented by the following formula:

$$\delta_k = r + \gamma \max_{a'} Q\left(s', a'\right) - Q(s, a) \tag{13}$$

where $Q(s, a)$ is defined as a reward value when the agent takes action a at state s. There are two sampling methods for PER: 1. $p_k = |\delta_k| + \varepsilon$, where ε is a very small number. 2. $p_k = \frac{1}{rank(k)}$, where $rank(k)$ is the empirical sequence number of the absolute value of the chronological error (TD error) sorted from the largest to the smallest. The value of $|\delta_k|$ determines the proximity; the larger the value of $|\delta_k|$, the higher the priority, and the smaller the value of $|\delta_k|$, the lower the priority. This subsection adopts a ranking-based prioritization mechanism and defines empirical priority as follows:

$$p_k = \frac{1}{rank(k)} \tag{14}$$

The goal is to make the TD error close and possibly small. If the TD error is relatively large, it means that our current Q-function is still far from the target Q-function and should be updated more often. Therefore, the TD error is used to measure the empirical value.

According to the above formula, the probability formula for the extracted sample k is as follows:

$$p(k) = \frac{p_k^\beta}{\sum_n p_n^\beta} \tag{15}$$

where n is the size of the replay experience pool. The value range of β is [0,1]. When $\beta = 0$, it means uniform sampling.

Recall the DQN algorithm [21]; the Q-function update formula is as follows:

$$Q^*(s,a) \leftarrow Q(s,a) + \alpha\left(r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right) \tag{16}$$

where α is the learning rate, and γ is the discount factor. It can be generated using the Bellman equations:

$$Q(s,a) = r + \gamma \sum_{a' \in A} Q(s',a') \tag{17}$$

The objective function y of a DQN is defined as follows:

$$y = r + \gamma \max_{a'} Q(s',a';\omega^-) \tag{18}$$

The DQN algorithm uses two NNs for error backpropagation and weight update. One of them is called evaluate_ net, which is used to generate an estimated value. Another is called target_ net, which generates a target Q value. The two networks have the same structure. The difference is that the weight ω of the evaluate net is updated continuously, while the weight ω^- of the target_ net is updated regularly, which records the historical weight of the evaluate_ net. The specific update procedure can be seen in Algorithm 1. Based on the above formulation, the loss function of the DQN algorithm can be defined as follows:

$$loss(\omega) = E[(y - Q(s,a;\omega))^2] \tag{19}$$

A dynamically decaying ε-greedy action selection policy is employed, which selects actions as follows:

$$a_k = \begin{cases} \underset{a}{argmax} Q(s,a) & 1-\varepsilon \\ a_{random} & \varepsilon \end{cases} \tag{20}$$

where ε decays naturally exponentially according to the following formula:

$$\varepsilon = \varepsilon_0 \times e^{(-g \times episode/h)} \tag{21}$$

where g and h are adjustable parameters; the parameter ε_0 is the initial exploration probability of the system. Algorithm 1 is presented below to show the procedure of the developed DQN algorithm in detail for the obstacle avoidance of the UAV.

Remark 3: In Algorithm 1, the highlighted contribution is that the adaptive reward works for the reward or punishment for the behavior of a UAV. Moreover, it plays a key role in performance evaluation and policy

Algorithm 1 Improved PER

Initialize the replay memory D with the capacity N
Initialize evaluate_ net with weights ω randomly
Initialize target_ net with $\omega^{-} = \omega$
for episode = 1, M **do**
  Initialize the reward and set the UAV to the starting point
  for k =1, T **do**
    Choose action a_k at random with probability ε
    Otherwise, select action: $\underset{a}{argmax} Q(s, a)$
    Perform action a_k to get the next time reward $r_{(k+1)}$ and the new state $s_{(k+1)}$
    Calculate: $\delta_k = r + \gamma \underset{a'}{max} Q(s', a') - Q(s, a)$
    Order δ_k from the largest to the smallest to get $rank(k)$
    Calculate the precedence of state action transition experience: $p_k = \frac{1}{rank(k)}$
    Store the transformed experience $(s_k, a_k, r_{(k+1)}, s_{(k+1)})$ to D according to the priority sequence number p_k
    Calculate the sampling probability $p(k)$ using (15)
    Draw samples according to the sampling probability $p(k)$
    Calculate loss function: $loss(\omega) = E\left[(y - Q(s, a; \omega))^2\right]$
    Update the weights ω of through a gradient descent procedure on loss: $\omega_{k+1} = \omega_k + \alpha \left[r + \gamma \underset{a'}{max} Q(s', a'; \omega^{-}) - Q(s, a; \omega)\right] \nabla Q(s, a; \omega)$
    Every C steps assign the weights of the evaluate_ net to targe_ net $\omega^{-} \longleftarrow \omega$
  end for
end for

updates, together with DQN and PER. In the sense of the novel reward function, the developed Algorithm 1 is called IPER.

Remark 4: Notice that the natural exponential decay algorithm[22] to the time-varying greedy policy is employed for fully exploring the environment at the start of learning.

4. SIMULATION RESULTS

This section aims to evaluate the effectiveness of the proposed algorithm by implementing the developed DQN algorithm with adaptive changing rewards.

This section assumes a 3D map area of 100 m × 100 m × 100 m, in which two environments are generated. The first approach involves the random generation of multiple obstacles within the designated area. The second objective is to generate multiple mountain terrains in this area using a randomization process. It is specified that the UAV is operating at a constant velocity of 10 m/s in both environments. The initial coordinates of the UAV are (1,1,1), and the desired destination is located at (80,80,80). Table 1 and Table 2 present the parameters that will be employed in the experiment. The actions θ and φ can be selected from a set of (-15 deg, -10 deg, -5 deg, 0 deg, 5 deg, 10 deg). Consequently, there are a total of 36 possible actions that can be chosen.

By implementing Algorithm 1, Figure 3 and Figure 4 show the UAV trajectories in three obstacle environments with different numbers and positions. By adopting the proposed algorithm, the UAV can successfully accomplish obstacle avoidance in some complex environments following a certain amount of training time.

Table 1. Parameters of the environment and UAV model

Parameter	Value
Map size	100 m× 100 m × 100 m × 100 m
Flight speed V_u	10 m/s
Safety distance R_{safe}	5 m
Warning distance R_m	10 m

Table 2. Parameters of network training

Reply memory size	1,000
Mini-batch size	64
Learning rate α	0.01
Discount factor γ	0.8

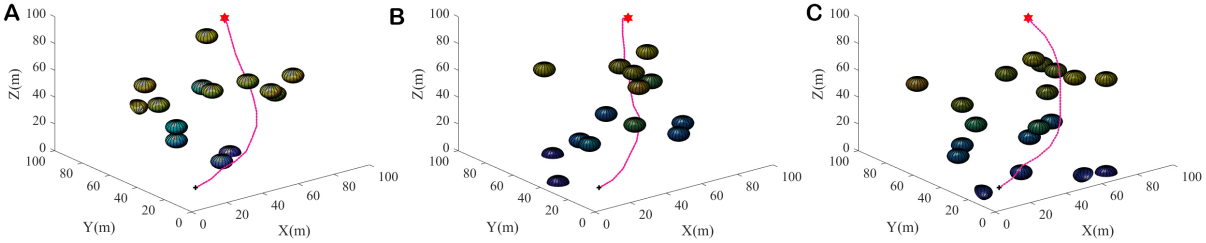

Figure 3. UAV trajectories in 3D multi-obstacle environments. (A) Environment 1; (B) Environment 2; (C) Environment 3.

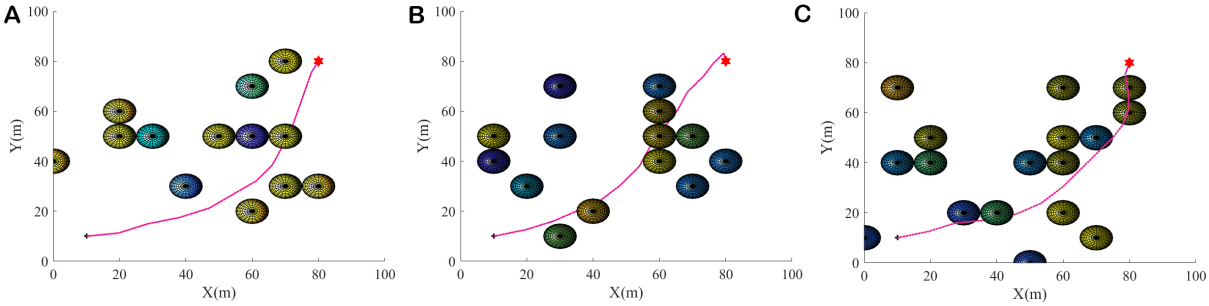

Figure 4. UAV trajectories in 3D multi-obstacle environments (aerial view). (A) Environment 1; (B) Environment 2; (C) Environment 3.

The obstacle avoidance capability of the algorithm under consideration is evaluated in 3D mountainous terrain. Figure 5 illustrates the trajectories of UAVs as they navigate through obstacle avoidance paths in three distinct environments, each characterized by varying numbers and heights of peaks. It is evident that the UAV is capable of effectively avoiding obstacles in challenging environments through the utilization of the proposed algorithm.

Figure 6 depicts the diagram of the loss function for the algorithm proposed in this paper. It is evident that the loss function tends to converge after 120 rounds. In particular, the validation of the loss function performs satisfactorily as 0.83733 at round 121.

For comparisons, the PER and DQN methods are implemented in the multiple mountain environments as well. The IPER algorithm proposed in this article combines PER and adaptive weight dynamic adjustment of the reward function. Compared to the traditional DQN algorithm and PER algorithm, it has the advantages of improving sample utilization and accelerating the convergence speed of the algorithm. Figure 7 and Figure 8 plot the average reward curves and the average step of reaching the target point using PER, DQN, and IPER in this paper. From Figure 7 and Figure 8, one can see that the IPER algorithm developed in this paper

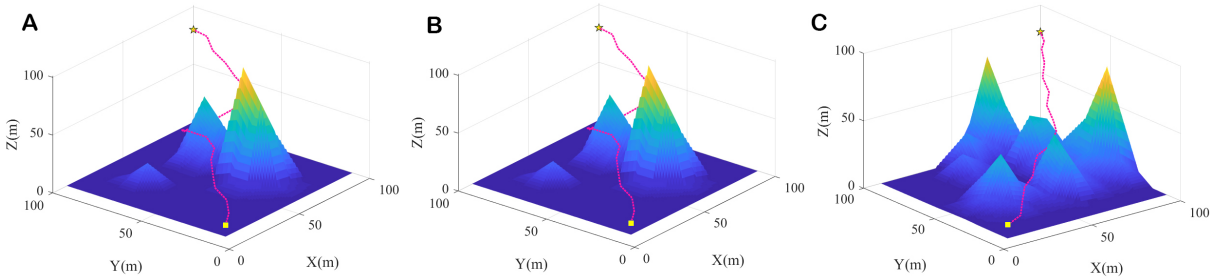

Figure 5. UAV trajectories in 3D mountain environments. (A) Environment 1; (B) Environment 2; (C) Environment 3.

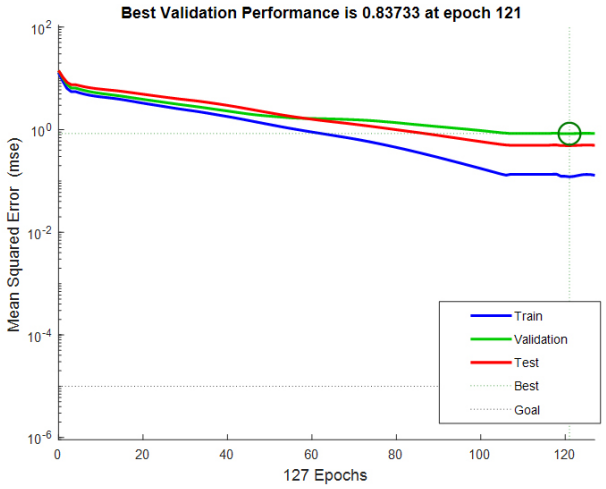

Figure 6. Loss function of the IPER algorithm.

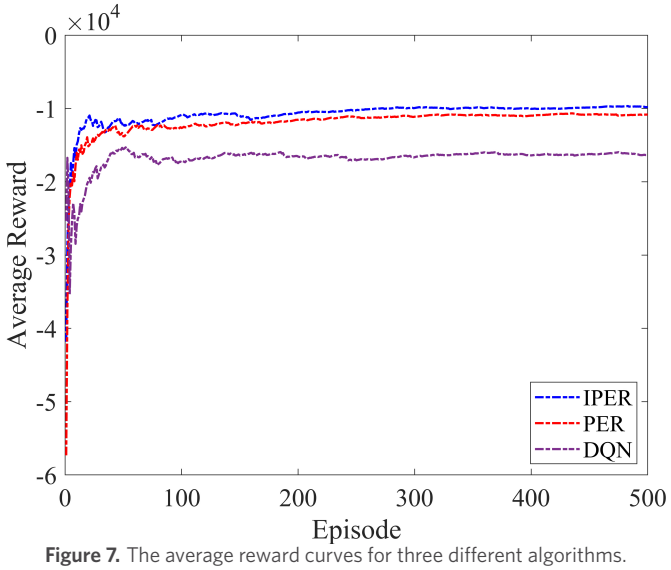

Figure 7. The average reward curves for three different algorithms.

outperforms the PER and DQN methods.

The above figures show the flight trajectories of the UAV in multiple obstacle environments and mountain simulation environments from different angles, showing that the UAV can reach the target point in a shorter path while successfully avoiding obstacles. In addition, by comparing the two indicators of average reward and average step size, it is evident that the IPER algorithm proposed in this paper outperforms both the traditional

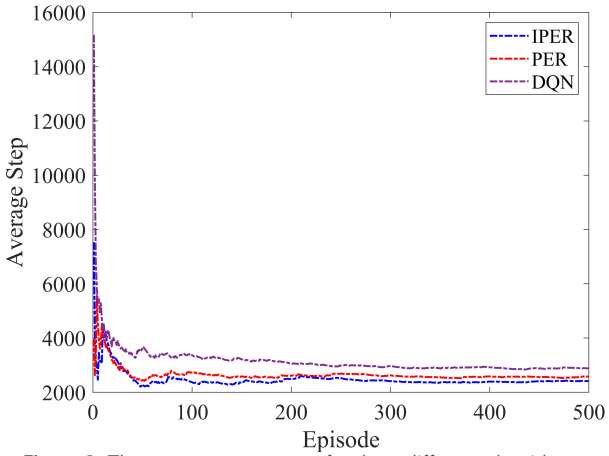

Figure 8. The average step curves for three different algorithms.

DQN algorithm and the PER algorithm.

5. CONCLUSIONS

This paper proposes a deep Q network algorithm that integrates an adaptive reward function and a PER method to address the challenge of intelligent obstacle avoidance for UAVs. Compared to the traditional DQN algorithm, rewards are dynamically adjusted based on the spatial distance between the drone and various obstacles along its path. This adjustment aims to strike a balance between obstacle avoidance and performance indicators. According to our experimental results, this adaptive reward method can effectively enable the UAV to navigate to the target point successfully without colliding with obstacles. In addition, deep RL also suffers from the issue of requiring a substantial amount of data samples, which is both unavoidable and expensive in real-world scenarios. In future research, we will address this problem and explore ways to effectively utilize existing samples for training in order to enhance efficiency.

The DQN algorithm learns the optimal strategy through a large number of training samples in a simulated environment. However, this strategy may not be very suitable for UAV navigation tasks in real-world environments because there may be significant differences between the real environment and the simulation environment. Therefore, studying how to adapt the strategies learned in the simulation environment to the real environment is a question worth exploring. Therefore, investigation transfer RL for practical obstacle avoidance of UAVs will be our future research direction.

DECLARATIONS

Authors' contributions

Made significant contributions to the conception and experiments: Gao H
Made significant contributions to the writing: Kong B
Made substantial contributions to the revision: Yu M, Li J

Availability of data and materials

Not applicable.

Financial support and sponsorship

This work was supported in part by the National Natural Science Foundation of China under Grant 62073158 and the Basic research project of the Education Department of Liaoning Province (LJKZ0401).

Conflicts of interest
All authors declared that there are no conflicts of interest.

Ethical approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

REFERENCES
1. Zhou Y, Baras JS. Reachable set approach to collision avoidance for UAVs. In 2015 54th IEEE Conference on Decision and Control (CDC). 2015; pp. 5947-52. DOI
2. Iacono M, Sgorbissa A. Path following and obstacle avoidance for an autonomous UAV using a depth camera. *Robot Auton Syst* 2018;106:38-46. DOI
3. Radmanesh M, Kumar M, Guentert PH, Sarim M. Overview of path-planning and obstacle avoidance algorithms for UAVs: a comparative study. *Unmanned Syst* 2018;6:95-118. DOI
4. Ait Saadi A, Soukane A, Meraihi Y, Benmessaoud Gabis A, Mirjalili S, Ramdane-Cherif A. UAV path planning using optimization approaches: a survey. *Arch Comput Methods Eng* 2022;29:4233-84. DOI
5. Maini P, Sujit PB. Path planning for a UAV with kinematic constraints in the presence of polygonal obstacles. In 2016 International Conference on Unmanned Aircraft Systems (ICUAS); 2016. pp. 62-7. DOI
6. Mandloi D, Arya R, Verma AK. Unmanned aerial vehicle path planning based on A* algorithm and its variants in 3D environment. *Int J Syst Assur Eng Manag* 2021;12:990-1000. DOI
7. Wu X, Xu L, Zhen R, Wu X. Biased sampling potentially guided intelligent bidirectional RRT* algorithm for UAV path planning in 3D environment. *Math Probl Eng* 2019;2019:1-12. DOI
8. Liu H, Liu HH, Chi C, Zhai Y, Zhan X. Navigation information augmented artificial potential field algorithm for collision avoidance in UAV formation flight. *Aerosp Syst* 2020;3:229-41. DOI
9. Perez-Carabaza S, Besada-Portas E, Lopez-Orozco JA, de la Cruz JM. Ant colony optimization for multi-UAV minimum time search in uncertain domains. *Appl Soft Comput* 2018;62:789-806. DOI
10. Li J, Deng G, Luo C, Lin Q, Yan Q, Ming Z. A hybrid path planning method in unmanned air/ground vehicle (UAV/UGV) cooperative systems. *IEEE Trans Veh Technol* 2016;65:9585-96. DOI
11. Shao S, Peng Y, He C, Du Y. Efficient path planning for UAV formation via comprehensively improved particle swarm optimization. *ISA Trans* 2020;97:415-30. DOI
12. Roberge V, Tarbouchi M, Labonte G. Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning. *IEEE Trans Industr Inform* 2013;9:132-41. DOI
13. Jembre YZ, Nugroho YW, Khan MT, et al. Evaluation of reinforcement and deep learning algorithms in controlling unmanned aerial vehicles. *Appl Sci* 2021;11:7240. DOI
14. Wu J, Sun Y, Li D, et al. An adaptive conversion speed Q-learning algorithm for search and rescue UAV path planning in unknown environments. *IEEE Trans Veh Technol* 2023;1-14. DOI
15. Mnih V, Kavukcuoglu K, Silver D, et al. Playing Atari with deep reinforcement learning. *arXiv* 2013. Available from: https://doi.org/10.48550/arXiv.1312.5602 [Last accessed on 16 Aug 2023].
16. Ye Z, Wang K, Chen Y, Jiang X, Song G. Multi-UAV navigation for partially observable communication coverage by graph reinforcement learning. *IEEE Trans Mob Comput* 2023;22:4056-69. DOI
17. Wang L, Wang K, Pan C, Xu W, Aslam N, Nallanathan A. Deep reinforcement learning based dynamic trajectory control for UAV-assisted mobile edge computing. *IEEE Trans Mob Comput* 2022;21:3536-50. DOI
18. Lin Z, Castano L, Mortimer E, Xu H. Fast 3D collision avoidance algorithm for Fixed Wing UAS. *J Intell Robot Syst* 2019;97:577-604. DOI
19. Jiang L, Huang H, Ding Z. Path planning for intelligent robots based on deep Q-learning with experience replay and heuristic knowledge. *IEEE/CAA J Automatica Sinica* 2020;7:1179-89. DOI
20. Schaul T, Quan J, Antonoglou I, Silver D. Prioritized experience replay. *arXiv* 2016. Available from: https://arxiv.org/abs/1511.05952 [Last accessed on 16 Aug 2023]
21. Mnih V, Kavukcuoglu K, Silver D, et al. Human-level control through deep reinforcement learning. *Nature* 2015;518:529-33. DOI
22. She D, Jia M. Wear indicator construction of rolling bearings based on multi-channel deep convolutional neural network with exponentially decaying learning rate. *Measurement* 2019;135:368-75. DOI