Electronic Research Archive

https://www.aimspress.com/journal/era

ERA, 33(4): 1946–1967.
DOI: 10.3934/era.2025087
Received: 24 January 2025
Revised: 25 March 2025
Accepted: 28 March 2025
Published: 03 April 2025

Research article

Stratospheric airship fixed-time trajectory planning based on reinforcement learning

Qinchuan Luo[1], Kangwen Sun[1], Tian Chen[2,*], Ming Zhu[2] and Zewei Zheng[3]

[1] School of Aeronautic Science and Engineering, Beihang University, Beijing 100191, China

[2] Institute of Unmanned System, Beihang University, Beijing 100191, China

[3] School of Automation Science and Electrical Engineering, Beihang University, Beijing 100191, China

* **Correspondence:** Email: chentian@buaa.edu.cn; Tel: +8601082339250; Fax: +8601082339250.

Abstract: Large-scale movement over a fixed time is one of the unique tasks of stratospheric airships. In practical applications, stratospheric airships often need to arrive at the designated location on time when performing tasks such as monitoring and detection. Due to the large wind resistance and low ship speed, stratospheric airships are easily affected by wind during long-distance movement. Therefore, determining how to ensure that the airship arrives at the designated location within the target time under the influence of dynamic wind fields is an urgent problem to be solved. This paper proposes an innovative solution. Based on the dueling double deep Q-network (D3QN) architecture, a trajectory planning algorithm (named FTD3) for fixed-time large-scale maneuvers was constructed. By preprocessing the wind field data and reducing the amount of input data, all information about the future wind field can be retained without introducing the instantaneous wind field. A new reward function was designed to incorporate time and distance constraints into the same dimension through time–distance mapping. Comparative experiments with other architectures showed that in the test set verification, the success rate of FTD3 reached 78.3%, compared to 47.7% for the double deep Q-network (DDQN)-based algorithm. Compared to other algorithms, FTD3 could avoid overfitting problems with the same training step size and yielded good results in uncertain wind fields. In summary, FTD3 provides an effective solution for the trajectory planning of stratospheric airships for scheduled and large-scale movement in dynamic wind fields.

Keywords: trajectory planning; stratospheric airship; reinforcement learning; fixed time; dynamic wind field

1. Introduction

Stratospheric airships are emerging as critical platforms for long-duration applications, such as regional monitoring, communication relaying, and weather forecasting. Leveraging buoyancy from helium- or hydrogen-filled envelopes, these airships offer extended operational durations. However, their large sizes and limited propulsion capabilities introduce unique challenges. The expansive surface areas of the airships increase the wind resistance [1]. Additionally, their electrically powered propulsion systems are constrained by the energy availability, resulting in slow speeds that are often comparable to stratospheric wind speeds. Additionally, the stratospheric wind field is characterized by an uneven distribution, which creates both challenges and opportunities: while high-speed winds hinder progress, low-wind-speed regions can be strategically utilized to reduce maneuverability demands during transitions [2]. These factors underscore the critical role of trajectory planning in ensuring mission success.

A particularly challenging scenario arises when stratospheric airships must reposition to monitor moving ground or maritime targets. For example, when performing the mission of observing a distant moving target, the airship sets off at time T_0, crosses a large area, and arrives above the target's expected appearance position at time T_{target} . Achieving this involves solving a fixed-time trajectory planning problem in which both spatial and temporal constraints must be satisfied. This task is further complicated by dynamic wind conditions: headwinds can impede the airship's approach to the target, and when the airship reaches the target ahead of time, it will also affect the airship's station keeping. Addressing these problems requires trajectory planning methods capable of balancing the conflicting demands of environmental constraints and precise timing.

Trajectory planning methods have seen significant advances across various domains, including robotics, drone motion planning, and underwater vehicle motion planning [3–7]. These methods can be broadly categorized into two approaches. The methods in the first category are sampling-based methods, such as the rapidly exploring random tree (RRT) [8], particle swarm optimization (PSO) [9], probabilistic road map (PRM) [10], and artificial potential field (APF) [11] methods, which rely on pre-defined environmental models. Zhang et al. [12] designed a path planning algorithm based on PSO and the cuckoo search algorithm for point-to-point path planning missions, and they obtained a shortest and energy-optimal path. Luo et al. [13] proposed a trajectory planning algorithm based on a combination of the RRT and APF methods for regional station-keeping missions, which could solve the trajectory planning problem without target points and considered the power, energy, and thermal constraints of the airship itself.

However, traditional methods have low generalization abilities, which means that the trajectory needs to be replanned after the wind field changes, and previous experience cannot be used [14]. The methods in the second category are artificial-intelligence-based methods that leverage algorithms simulating human decision-making, such as heuristic search [15], brute force search [16], and artificial neural network [17] methods. With the rapid advancement of machine learning, deep reinforcement learning (DRL) has emerged as a promising approach due to its adaptability and ability to handle complex decision-making tasks [18–20].

DRL has been increasingly applied for stratospheric airship trajectory planning. Zheng et al. [21] introduced DRL-based trajectory planning with a reward function tailored to airship dynamics, enabling point-to-point navigation and wind field prediction. Qi et al. [22] proposed a soft actor-critic (SAC)

algorithm for time-optimal and energy-optimal point-to-point planning. Based on the same algorithm, Wang et al. [23] considered the influence of clouds on airships in addition to wind when planning airship trajectories. Liu et al. [24] developed a two-dimensional trajectory planning approach using the proximal policy optimization (PPO) algorithm, achieving smoother regional station-keeping trajectories by operating in continuous action spaces. However, previous research primarily focused on point-to-point or regional station-keeping problems and did not adequately address fixed-time point-to-point trajectory planning, which is a critical and common challenge in practical applications.

Motivated by the results mentioned above, a novel reinforcement-learning approach for fixed-time trajectory planning of stratospheric airships under dynamic wind conditions is proposed in this paper. This method utilizes the current and future wind fields as inputs to plan the motion of the airship and then obtains a trajectory that can satisfy the position and time constraints.

The main contributions of this paper are as follows:

(i) A novel reinforcement-learning approach based on the dueling double deep Q-network (D3QN) is proposed for the fixed-time trajectory planning of stratospheric airships under dynamic wind conditions.

(ii) A reward function capable of handling fixed time constraints was developed, transforming time constraints into positional constraints through time–distance mapping. This reduces the complexity of fixed-time trajectory planning and enhances the model training efficiency.

(iii) Innovative preprocessing and selection methods for wind field data significantly reduce the data complexity without losing critical information, thereby improving the training performance and avoiding prolonged training durations.

The rest of this paper is organized as follows: In Section 2, the problem modeling is presented. In Section 3, a reinforcement learning approach for trajectory planning is presented. In Section 4, comparative experiments are discussed to demonstrate the effectiveness of the proposed approach. Finally, conclusions are provided in Section 5.

2. Problem modeling

When undertaking a fixed-time mission, an airship must operate within both time constraints and a geographically limited area. To model this scenario, a 6x6 degrees of latitude and longitude region is defined as the mission area. For simplicity, the airship is treated as a particle. The airship's stability is primarily maintained by buoyancy. Since the air density at high altitudes within the mission area is nearly constant, the equilibrium between gravity and buoyancy stabilizes the airship's altitude. Based on this observation, we make the following assumption:

Assumption 2.1. *Altitude variations of the airship are negligible and can be ignored in this problem.*

The objective of the proposed trajectory planning strategy is to ensure that the airship reaches a target point at a specified time without exiting the designated mission area (see Figure 1). The process of guiding the airship to its goal is framed as a discrete-time stochastic control problem, making it well-suited for modeling using the Markov decision process (MDP) framework. For fixed-time trajectory planning, the terminal state consists of two different types of states:

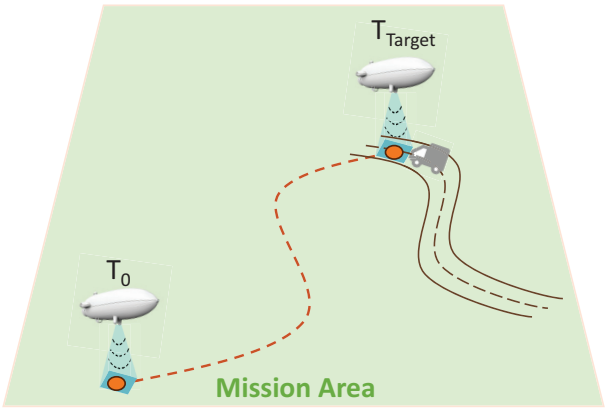


Figure 1. Example of a fixed-time trajectory planning problem in a mission area.

1) Conflict state: The agent's position (x, y) is outside the mission area:

$$\begin{cases} x < W_1 \\ x > W_2 \\ y < L_1 \\ y > L_2 \end{cases}, \tag{2.1}$$

where W_1 and W_2 denote the upper and lower bounds of the longitude of the mission area, respectively, and L_1 and L_2 denote the upper and lower bounds of the latitude of the mission area, respectively.

2) Goal state: The agent arrives at the destination when the current time t equals the target time T_{target}. The Euclidean distance between point $P(x, y)$ and point $P'(x', y')$ is calculated by

$$\text{Dis}\,(P, P') = \sqrt{(x - x')^2 + (y - y')^2}. \tag{2.2}$$

Then, the goal state is

$$\begin{cases} \text{Dis}\left(P, P^{\text{Goal}}\right) < R_{\text{target}} \\ t = T_{\text{target}} \end{cases}, \tag{2.3}$$

where the destination is defined by a circular area near the target point P^{Goal}, with a radius of R_{target}.

Since the airship flies very slowly and a mission flight can last several days, using a small planning step would result in unnecessary computational overhead. The trajectory planning of the airship is closely tied to the wind field, making it essential to use forecast data as an input for the wind conditions during an actual flight. Most current forecast sources provide wind data with a time resolution of 1 hour, and thus, the step size for trajectory planning is selected as 1 hour. However, the accuracy of the wind field forecasts decreases significantly as the forecast time span increases. To address this, the planning horizon for each trajectory is limited to 24 hours. If the target point is beyond the airship's reachable distance within a single planning horizon, intermediate way-points can be introduced to divide the planning into manageable segments.

3. Algorithm design

3.1. Reinforcement learning environment

3.1.1. State representation

The agent gathers information about its environment through the defined MDP state. The state must contain all the information necessary for the agent to make optimal decisions. The agent's state at time t is denoted as s_t. To ensure compatibility with a deep neural network, all the parameters are normalized. The state s_t is divided into two components: $s_t = [s_t^0, s_t^1]$, where s_t^0 contains information specific to the agent and its destination, and s_t^1 includes data related to the environment. Based on Assumption 2.1, s_t^0 can be written as

$$s_t^0 = [x, y, t], \tag{3.1}$$

where x and y represent the agent's position, and t represents the time elapsed since departure. The component s_t^1 represents information associated with the environment. In the stratosphere, wind is the most significant environmental factor affecting the airship. Due to the airship's large volume, its motion is highly sensitive to wind conditions. The wind field in the mission area must be considered to achieve a globally optimal solution when performing fixed-time trajectory planning. Additionally, future wind conditions must be considered to enable the airship to make decisions in advance. For instance, the predicted wind speed at the target point determines whether the airship can arrive early and maintain its position there.

The proposed fixed-time trajectory planning algorithm uses a series of wind fields as environmental inputs. The input wind field at each moment is represented as a four-dimensional tensor:

$$\mathcal{X} \in \mathbb{R}^{H\times V\times U\times T}, \tag{3.2}$$

where H represents the horizontal spatial grid points, V represents the vertical spatial grid points, and (H, V) constitutes a two-dimensional plane area. U represents the dimension of the wind speed vector at a grid point, and T represents the number of moments included in a single input wind field. Thus, the environmental state s_t^1 can be expressed as

$$s_t^1 = \mathcal{X}. \tag{3.3}$$

Remark 1. *The stratospheric wind speed used in this paper is a two-dimensional vector, containing an east-west component and a north-south component. Because the wind field is time-varying, the input wind field usually contains the wind field at the current moment and multiple moments in the future.*

3.1.2. Action space

In fixed-time trajectory planning, the airship's actions are defined as changes in its heading angle at each step. The action space $\mathcal{A}$ is represented by a five-dimensional discrete vector, which includes four directions of motion and an option to take no action. We use a discrete action space for trajectory planning. The planning time scale of the scenario setting in this paper is 24 hours, and the planning step size is set to 1 hour. The turning time of the airship (3–5 minutes) is very small compared to the step size, so we assume that the turning process can be ignored. Compared with the continuous action space,

the discrete action space reduces the computational complexity and improves the training efficiency. Different from the precision requirements of short-time control, we believe that it is sufficient to use the discretized action space for trajectory planning under large time scales. In practical applications, the trajectory planning of the continuous action space can be further performed within the step size. Unlike conventional aircraft, a stratospheric airship relies on buoyancy for stability. Therefore, one of the possible actions is to stay afloat without providing thrust. The action space can be expressed as

$$\mathcal{A} = \{a_1, a_2, a_3, a_4, a_5\}, \tag{3.4}$$

where a_1 and a_2 correspond to movements along the x-axis (e.g., the east–west direction), a_3 and a_4 correspond to movements along the y-axis (e.g., the north–south direction), and a_5 represents staying stationary (no thrust output).

3.1.3. State transition function

The state transition function $\mathcal{T}$ describes how the agent interacts with the environment. The airship's position is influenced by both its thrust and the surrounding wind field. To simplify the problem and focus on trajectory planning, the airship is modeled as a point mass, and its position is represented using grid coordinates. If the airship moves from its current position (x, y) to the next position (x', y'), the state transition function can be expressed as [25]:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + V_A \begin{bmatrix} \hbar\,(a_1, a_2) \\ \hbar\,(a_3, a_4) \end{bmatrix} \hbar\,(1, a_5) + \begin{bmatrix} V_{\text{windx}} \\ V_{\text{windy}} \end{bmatrix}, \tag{3.5}$$

where V_A is the cruising speed of the airship, V_{windx} and V_{windy} correspond to the components of the wind speed V_{wind} along the x-axis and the y-axis, respectively. $\hbar$ is defined as

$$\hbar(p, q) = \text{sgn}(p) - \text{sgn}(q), \tag{3.6}$$

where p and q jointly determine the direction of motion. The time is updated as follows:

$$t' = t + \Delta t, \tag{3.7}$$

where Δt denotes the planning step size, and t' denotes the time of the next step.

3.1.4. Reward function

In reinforcement learning (RL), the agent's goal is to maximize the expected reward, which guides the neural network's gradient updates. When designing the reward function, we hope to give positive rewards to the agent when it takes actions that are beneficial to the mission goal, and negative rewards when it is not. Therefore, the reward function is a critical component and must be carefully designed for specific applications. A significant challenge in fixed-time planning problems is determining how to incorporate strict time constraints into the reward function. One common approach is to assign a large reward when the agent reaches the target point precisely at the designated time. However, this design does not provide sufficient guidance to the agent during training, leading to prolonged training times and slow model convergence. To address this issue, we transform the time dimension into a distance metric.

In our model, the reward function consists of the following three components: 1) the distance reward r_{dis} , 2) the collision reward r_{cls}, and 3) the goal reward r_{goal} . The transformation of the time dimension

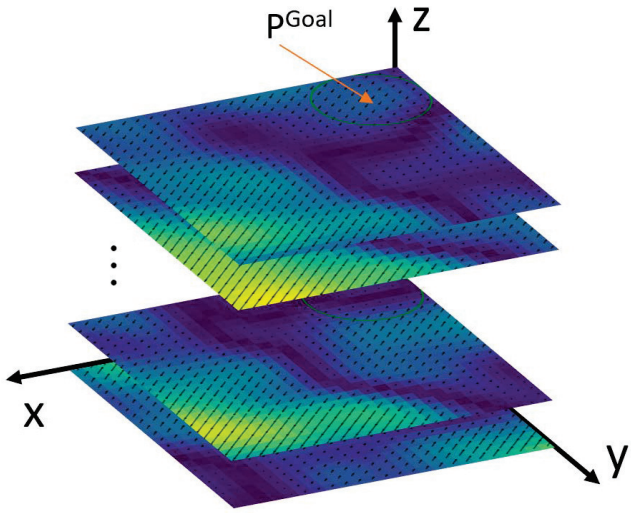


Figure 2. Wind field at each moment in 3D fixed-time trajectory planning.

into a distance is achieved through time–distance mapping. The fixed time span T of the airship's trajectory planning is interpreted as a distance D_T, which represents the distance the airship would travel at its cruising speed over time T. This mapping ensures that both the time and positional constraints are evaluated on the same scale.

As shown in Figure 2, the initial two-dimensional fixed-time trajectory planning problem is converted into a three-dimensional (3D) trajectory planning problem. In this 3D representation, the time dimension is represented by the z-axis, where the airship moves with a constant velocity V_A. Under this transformation, the position of the target point can be expressed as

$$P^{\text{Goal}} = (x_t, y_t, z_t), \tag{3.8}$$

$$z_t = V_A \cdot T. \tag{3.9}$$

The Euclidean distance between point $P(x, y)$ and point $P'(x', y')$ is calculated by

$$\text{Dis}\,(P, P') = \sqrt{(x - x')^2 + (y - y')^2 + (z - z')^2}. \tag{3.10}$$

Then, the distance reward is

$$r_{\text{dis}} = \text{Dis}\left(P', P^{\text{Goal}}\right) - \text{Dis}\left(P, P^{\text{Goal}}\right), \tag{3.11}$$

which calculates the change in the grid distance from the current position to the target point. This reward guides the airship to approach the target point. The collision reward r_{cls} is a punitive measure that imposes a negative reward when the airship moves beyond the mission area. The goal reward r_{goal} is a huge positive reward that can only be earned if the airship successfully reaches the target point, and it is a negative reward if it does not.

Finally, the total reward is expressed as

$$r = c_{\text{dis}}\, r_{\text{dis}} + I_{\text{cls}}\, r_{\text{cls}} + I_{\text{goal}}\, r_{\text{goal}}\,, \tag{3.12}$$

where c_{dis} is a weighting factor. Based on testing, the following values were designed: $c_{\text{dis}} = -1, r_{\text{cls}} = -10, r_{\text{goal}} = 100$ or -10, $I_{\text{cls}} = 0$ or 1, which depends on whether the airship is beyond the mission area, and $I_{\text{done}} = 0$ or 1, which depends on whether the airship reaches the target at the target time. Under the regulation of the reward function, the airship can be guided to reach the target point at the target time.

3.2. Wind field data preprocessing

High-altitude wind fields are characterized by uncertainty, and the susceptibility of airships to wind requires the wind effects to be accounted for during trajectory planning. To address this, trajectory planning algorithms for airships must be highly flexible. Incorporating wind field factors into trajectory planning significantly enhances the intelligence and adaptability of these algorithms. Therefore, we designed an RL-based algorithm called FTD3.

Wind fields affect trajectory planning throughout the planning period, and they vary dynamically in both time and space. To ensure the generalizability of the training results, the inputs for FTD3 must include wind field data that represent both current and future conditions across the entire mission area. However, an excessive number of input parameters can increase the training complexity of the D3QN algorithm, introduce irrelevant information, and lead to overfitting. To address these challenges, we reduce the amount of input data while maintaining forward-looking planning by selecting wind field data at two specific moments: the current time T_i and six hours later T_{i+6}, for a total planning time span of $T = 24$ hours. The time interval of the input wind field data cannot be too short, otherwise the trajectory will lose predictability and fall into a local optimum. However, a time interval that is too long will result in too much irrelevant wind field data being input after the target time in the latter part of the mission. This preprocessing approach balances the need for foresight with computational efficiency, as illustrated in Figure 3.

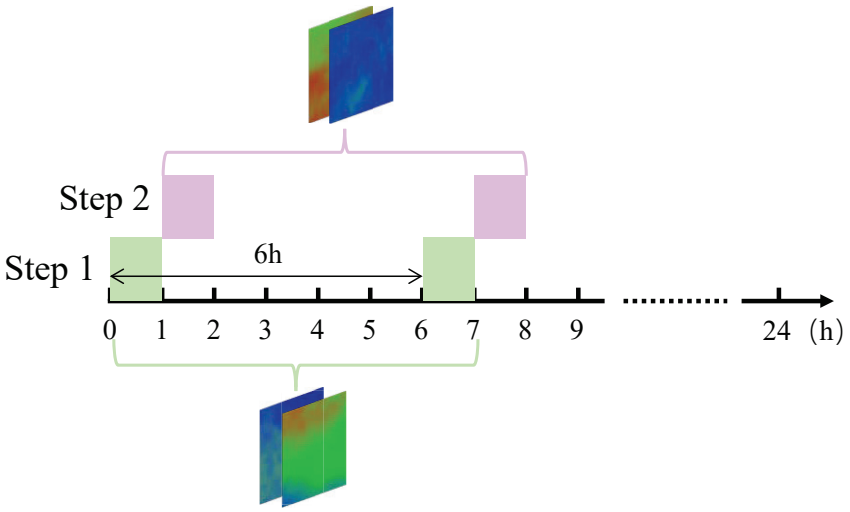


Figure 3. Wind field data input at each step.

Then, the input series of the wind fields are preprocessed. At each moment, a wind field $\boldsymbol{s}_1$ with

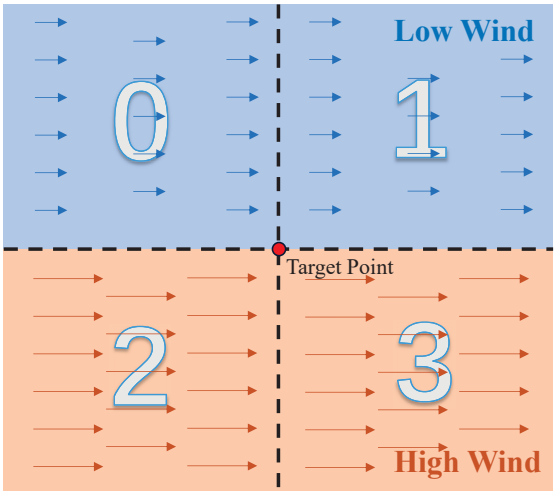


Figure 4. Wind field preprocessing based on location and wind direction.

dimensions of $m \times n$ in two directions can be represented as

$$\boldsymbol{s}_{1x} = \begin{bmatrix} w_{x11} & \cdots & w_{x1n} \\ \vdots & \ddots & \vdots \\ w_{xm1} & \cdots & w_{xmn} \end{bmatrix}_{m\times n}, \tag{3.13}$$

$$\boldsymbol{s}_{1y} = \begin{bmatrix} w_{y11} & \cdots & w_{y1n} \\ \vdots & \ddots & \vdots \\ w_{ym1} & \cdots & w_{ymn} \end{bmatrix}_{m\times n}, \tag{3.14}$$

where w_x and w_y denote the wind speeds in two directions. Since the wind field has two dimensions, a large number of input parameters are introduced to the D3QN algorithm. Additionally, these parameters are continuous, which poses significant challenges for RL. These challenges include prolonged learning times and poor training results. To mitigate these issues, it is necessary to preprocess and reduce the complexity of the wind field data. To achieve this reduction, wind fields are classified based on their favorability, depending on the location and wind direction, as illustrated in Figure 4. This classification simplifies the data while retaining essential information about how the wind affects the trajectory planning [26], and it is expressed mathematically as follows:

$$\boldsymbol{s}_2(i,j) = \begin{cases} 0, V_{\text{wind}} < V_A \,\&\, \cos(\chi_w - \phi) \geq 0 \\ 1, V_{\text{wind}} < V_A \,\&\, \cos(\chi_w - \phi) < 0 \\ 2, V_{\text{wind}} \geq V_A \,\&\, \cos(\chi_w - \phi) \geq 0 \\ 3, V_{\text{wind}} \geq V_A \,\&\, \cos(\chi_w - \phi) < 0 \end{cases}, \tag{3.15}$$

where χ_w is the wind direction angle, and ϕ is the direction angle from the position of the airship to the target point.

When determining whether a wind field is favorable for the mission, the wind speed is a key factor. The wind speed determines whether the airship's position is controllable. If $V_w < V$, where V_w is the wind speed and V is the airship's cruising speed, the wind is favorable for the airship to accomplish its

mission. Additionally, the wind direction relative to the direction of the target point from the airship is another important factor. If the wind is directed toward the target, i.e., $\cos(\chi_w - \phi) \geq 0$, the airship can successfully reach the target point, even if its position is uncontrollable. Therefore, the wind field $\boldsymbol{s}_1$ can be simplified to $\boldsymbol{s}_2$, expressed as

$$\boldsymbol{s}_2 = \begin{bmatrix} 0 & \cdots & \cdots & 3 & 2 & 0 \\ 3 & \ddots & 2 & 0 & 1 & 0 \\ \vdots & \ddots & 1 & 2 & \ddots & \vdots \\ \vdots & \ddots & 2 & 3 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 1 & \cdots & 3 & 1 & 0 & 1 \end{bmatrix}_{m \times n}. \tag{3.16}$$

3.3. Reinforcement learning network

We design the reinforcement learning network based on the dueling double deep Q-network (D3QN). The D3QN is an improvement of the deep Q-network (DQN). The DQN introduces a neural network to calculate the Q-value of all actions taken in the current state, and through training, the action with the greatest reward has the largest Q-value. The DQN contains two neural networks, the current network $Q(s, a; \theta)$ and the target network $Q'(s', a'; \theta')$, where θ and θ' are the parameters of the respective neural networks. The current network is used to participate in the forward propagation to update network parameters and estimate the Q-value of the current state-action pair (s, a), and the target network is used to calculate the maximum Q-value of the next state-action pair (s', a'). The loss function used for updating is calculated as

$$\text{Loss} = E\left[\left(r + \gamma \max_{a'} Q'(s', a'; \theta') - Q(s, a; \theta)\right)^2\right] \tag{3.17}$$

where $E[\cdot]$ represents the expectation, r represents the reward for taking the current action pair, and γ is the reward discount. Then, the parameters of the current network are updated through gradient descent as

$$\nabla_\theta \text{Loss} = E\left[\left(r + \gamma \max_{a'} Q'(s', a'; \theta') - Q(s, a; \theta)\right) \nabla_\theta Q(s, a; \theta)\right]. \tag{3.18}$$

A DQN makes the Q-value converge quickly to the possible optimization target by maximizing it, but it is easy to cause the problem of overestimating the Q-value, which makes the model have a large deviation.

A double deep Q-network (DDQN) is an improved version of the DQN that solves the problem of the DQN algorithm overestimating the value of actions. The double DQN algorithm does not directly select all possible Q-values calculated by the target network by maximizing them, but first selects the action a_{max} corresponding to the maximum Q-value through the estimation network, which is expressed as

$$a_{max} = \arg\max_a Q(s_{t+1}, a; \theta). \tag{3.19}$$

Then, in the loss function calculation, by decoupling the action selection and evaluation process, the systematic deviation of the Q-value is significantly reduced, which is expressed as

$$\text{Loss} = E\left[\left(r + \gamma Q'(s', a_{max}; \theta') - Q(s, a; \theta)\right)^2\right]. \tag{3.20}$$

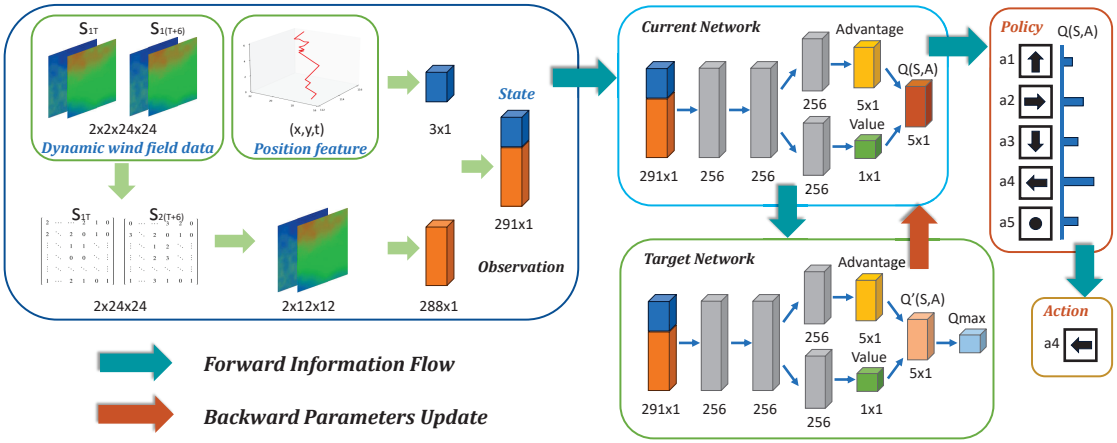


Figure 5. Pipeline of FTD3.

The D3QN decomposes the Q-value into the state value $V(s)$ and the advantage function $A(s,a)$ based on the DDQN, separating the contribution of state and action, reducing repeated learning, and improving generalization ability. In summary, the D3QN combines the evaluation decoupling and the value decomposition, solving the problems of overestimation and inefficient state modeling, and can achieve more stable, efficient, and generalized strategy learning in complex tasks.

The stratospheric wind field, unlike the low-altitude wind field, is less influenced by the ground and thus does not vary significantly with position on small scales. To address this, FTD3 reduces the amount of data using a pooling layer of a convolutional neural network. As shown in Figure 5, the preprocessed wind field data first passes through the pooling layer, then it merges with the airship's state information, and finally, it is fed into the fully connected layer [27]. The preprocessing step already performs feature extraction, which is why a convolutional layer is not introduced.

Due to the complexity of the wind field, fixed-time trajectory planning is more challenging than planning in other scenarios, where the optimization goal is typically to minimize the time or distance. FTD3 prioritizes finding feasible trajectories rather than optimizing for the shortest time or distance. To achieve this, we define the exploration rate ε as follows:

$$\varepsilon = \varepsilon_{\text{final}} + (\varepsilon_{\text{start}} - \varepsilon_{\text{final}}) \times e^{-k \cdot \frac{b}{m}}, \tag{3.21}$$

where $\varepsilon_{\text{final}}$ is the final exploration rate, $\varepsilon_{\text{start}}$ is the exploration rate at the start of training, m is the size of the minibatch, b denotes the number of successful attempts in the recent minibatch, and k denotes the exploration decay factor. The action a is determined by

$$a = \begin{cases} \text{random } a \in \mathcal{A}, & \varepsilon \\ \arg\max_a Q(s,a;\theta), & 1-\varepsilon \end{cases}. \tag{3.22}$$

Overall, during trajectory planning, FTD3 considers the global wind field over the entire time period, ensuring that the trajectory is globally optimal. By downscaling the wind field data, the training difficulty is reduced without losing crucial information, and the impact of irrelevant data on the results is minimized. Additionally, by decoupling the action from the state and separating the action selection from the value prediction, the training efficiency is improved, and the problem of overestimation is

mitigated. The algorithm pipeline is shown in Figure 5, and the complete training algorithm is presented in Algorithm 1.

Algorithm 1: FTD3 Algorithm

Input: Wind field data $W(h, w)$, Q-network Q, target network $\hat{Q}$, replay buffer B, discount factor γ, learning rate α, exploration rate ϵ
Output: Trained Q-network
Initialize Q-network Q with random weights
Initialize target network $\hat{Q}$ with weights $\hat{Q} \leftarrow Q$
Initialize replay buffer B
Learn step counter $c = 0$
Wind preprocess
for *episode = 1, maxEpisode* **do**
  Reset environment;
  for *t = 1, Targetime* **do**
    Integrate the position state s^0 and preprocessed global wind fields s^1
    $s = \left[s^0, s^1\right]$;
    Estimate action value
    $Q(s, a; (\theta_s, \theta_a)) = q(s; \theta_s) + q(s, a; \theta_a) - \mathrm{AVE}(s)$;
    Choose action $a = \begin{cases} \text{random } a \in \mathcal{A}, & \varepsilon \\ \arg\max_a Q(s, a; \theta), & 1 - \varepsilon; \end{cases}$
    Transfer state according to $\mathcal{T}$;
    Store transition (s, a, r, s') in replay buffer B;
    if *B is large enough* **then**
      Sample random minibatch m of transitions (s_j, a_j, r_j, s'_j) from B;
      for *each m* **do**
        Select action $a' = \arg\max Q(s'_j, a')$;
        Set target $y_j = r_j + \gamma \hat{Q}(s'_j, a')$;
        Perform a gradient descent step on $(y_j - Q(s_j, a_j))^2$ to update Q;
      end
    end
    if *learn step counter c mod* f_c *== 0* **then**
      Update target network $\hat{Q} \leftarrow Q$;
    end
    $s \leftarrow s'$;
  end
end

4. Experiments

4.1. Basic settings and evaluation indicators

The mission area for the trajectory planning was set in the South China Sea (16°N–22°N, 111°E–117°E, at an altitude of 20,000 m). The airship started at the initial position of 17°N, 112°E,

with the target point located at 21°N, 116°E. The time constraint was 24 hours, and the airship needed to fly from the starting point to the target point within this time frame, without leaving the mission area. The wind field data was sourced from the public historical data of the Global Forecast System, National Centers for Environmental Prediction, and the US National Weather Service. The wind field had a sampling interval of 1 hour with a resolution of 0.25°, and a 24×24 grid was extracted from the mission area. The parameter values for FTD3 are listed in Table 1. The deep Q-network (DQN) and double deep Q-network (DDQN) algorithms used for comparison employed the same default parameters as FTD3. The time step for the planning algorithms was set to 1 hour, which corresponded to the wind field's sampling interval. Our experiments were conducted on the Windows 11 system, with 32GB of RAN and NVIDIA RTX 4060 GPU, using Python 3.9. For standardization, uniform grid coordinates were used for convenience.

Table 1. Parameter values.

Parameter	Value
Batch size m	256
Learning rate η	1×10^{-4}
Action dimention	5
Time step interval t_{step}	1 h
Wind field size $h \times w$	24×24
Gamma γ	0.99
Target network update frequence f	1000
Maximum learning episode M	2×10^4
Replay buffer capacity $\mathbb{N}$	2×10^4
Optimizer	Adam

4.2. Effectiveness

We selected November 2022 as the training period. Figure 6 shows the trend of the average reward over 20,000 training steps. The average reward was positively correlated with the mission success rate, meaning that a higher average reward indicated that the airship was closer to the target point at the target time and less likely to be penalized for leaving the mission area. FTD3, based on the D3QN, exhibited a significant advantage in the fixed-time airship trajectory planning. Both the D3QN and DDQN successfully converged, while the DQN failed to learn a trajectory planning strategy. Due to the overestimation problem of the DQN, the airship may over-prefer those overestimated actions, resulting in poor results in action selection and failure to converge at the maximum number of training steps. Compared to the DDQN, D3QN achieved a higher average reward by the end of training through efficient state modeling.

We next used November 2023 as the test scenario to evaluate the performances of the D3QN and DDQN. We selected the same month, since the stratospheric wind field tends to follow seasonal trends, and the mission area and settings were consistent with the training scenario. By randomly selecting an hour within the 30×24 hours of November as the mission start time, the wind field during the mission was determined. After conducting 1000 experiments, the mission success rate for the D3QN was 78.3%,

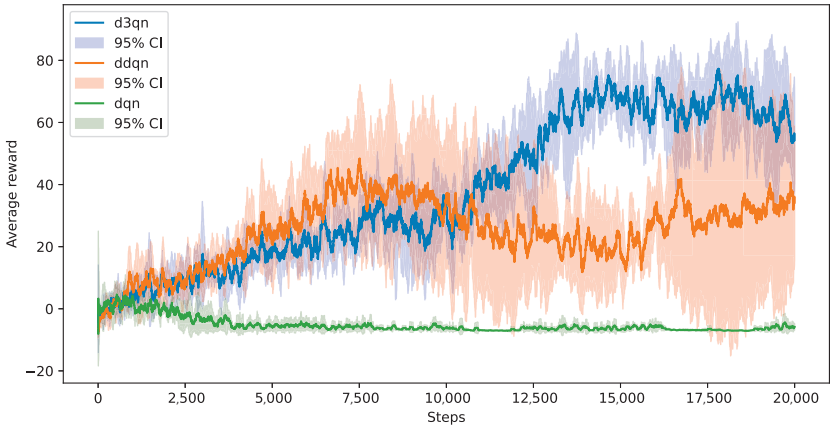


Figure 6. Average reward changes during training for the deep Q-network (DQN), double deep Q-network (DDQN), and dueling double deep Q-network (D3QN).

while the DDQN achieved a success rate of 47.7%. Due to the use of actual historical wind fields, this result includes extreme weather scenarios such as strong winds exceeding the airship's maximum airspeed for extended periods of time, which is clearly beyond the capabilities of trajectory planning, and our goal is to maximize mission success as much as possible within the airship's capabilities. During the training process, the D3QN has a higher average reward, which means that the trained strategy is better and can complete the mission more effectively. The D3QN's stronger generalization ability can help the airship perform well in test environments that it has never encountered before.

Remark 2. *Overfitting is a common challenge in RL. In our validation, we used the wind field data from 2023 as the test set and achieved a success rate of 78.3%. Since the wind fields in the training and test datasets were completely different, and extreme weather conditions are unavoidable, we believe that the performance of the trained model meets expectations, and no overfitting had occurred.*

These results were based on the assumption that the wind field was accurate. However, in real-world applications, most wind field data comes from meteorological forecasting agencies, and the predicted wind field may not be fully accurate. We introduced a random variation to the original wind speed in the state transition function to test the performance of the proposed approach with inaccurate wind fields, expressed as

$$V^*_{\text{wind}} = V_{\text{wind}} + \eta \text{Random}(-1, 1), \tag{4.1}$$

where V^*_{wind} is the wind field after adding a random variation, η represents the magnitude of the random variation, and Random(-1,1) denotes a random decimal value drawn from a uniform distribution between -1 and 1. Figure 7 shows the performance of FTD3 when a random wind field was incorporated. When the maximum random wind speed reached 15 m/s, FTD3 still achieved a 50.3% success rate over 1000 experiments. This indicated that, even when there were deviations in the forecasted wind field, our approach could still maintain a reasonable level of feasibility.

To assess the practicality of FTD3, we selected a scenario with an initial time of 17:00 on November 1 as an example. The 3D trajectory obtained from the airship's planning is shown in Figure 8, where the

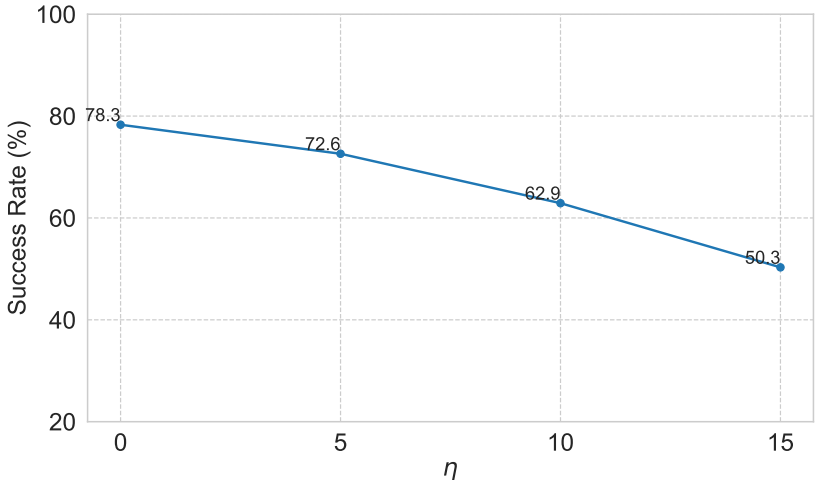


Figure 7. Success rate of FTD3 under different random wind magnitudes.

z-axis represents the time-converted distance, as described in Section 3.1.4. The airship reached the vicinity of the target point after 24 hours while adhering to the fixed-time constraint.

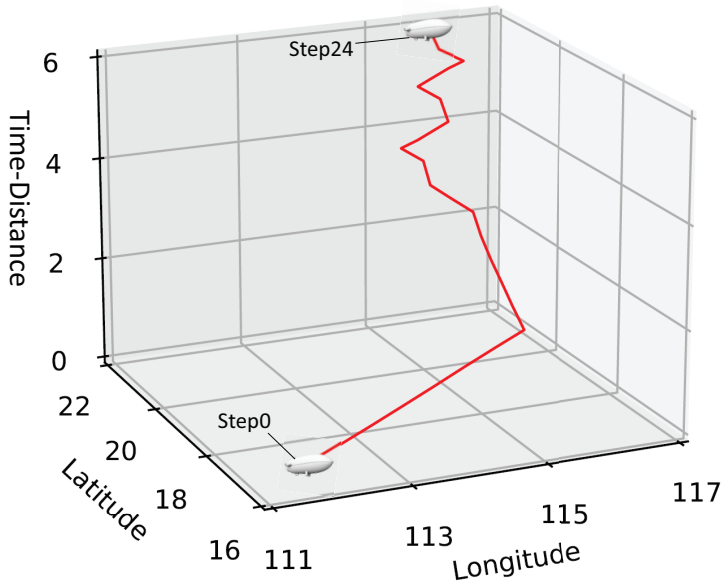


Figure 8. Three-dimensional trajectories generated by FTD3.

Figure 9 shows the wind field at various moments during the flight. At Step 1, the flight strategy suggested by FTD3 was to fly east. The wind speed near the airship was high, and the wind direction was not favorable for the airship to fly toward the target. Therefore, it would be a wise choose to exit the strong wind area as soon as possible. By Step 4, the airship was almost out of the strong wind area, and the flight strategy suggested by FTD3 was to continue flying east. At Step 7, the airship reached the southeastern part of the mission area, where the wind speed was lower than that in the northern part. This lower wind speed improved the airship's position control accuracy, which was advantageous

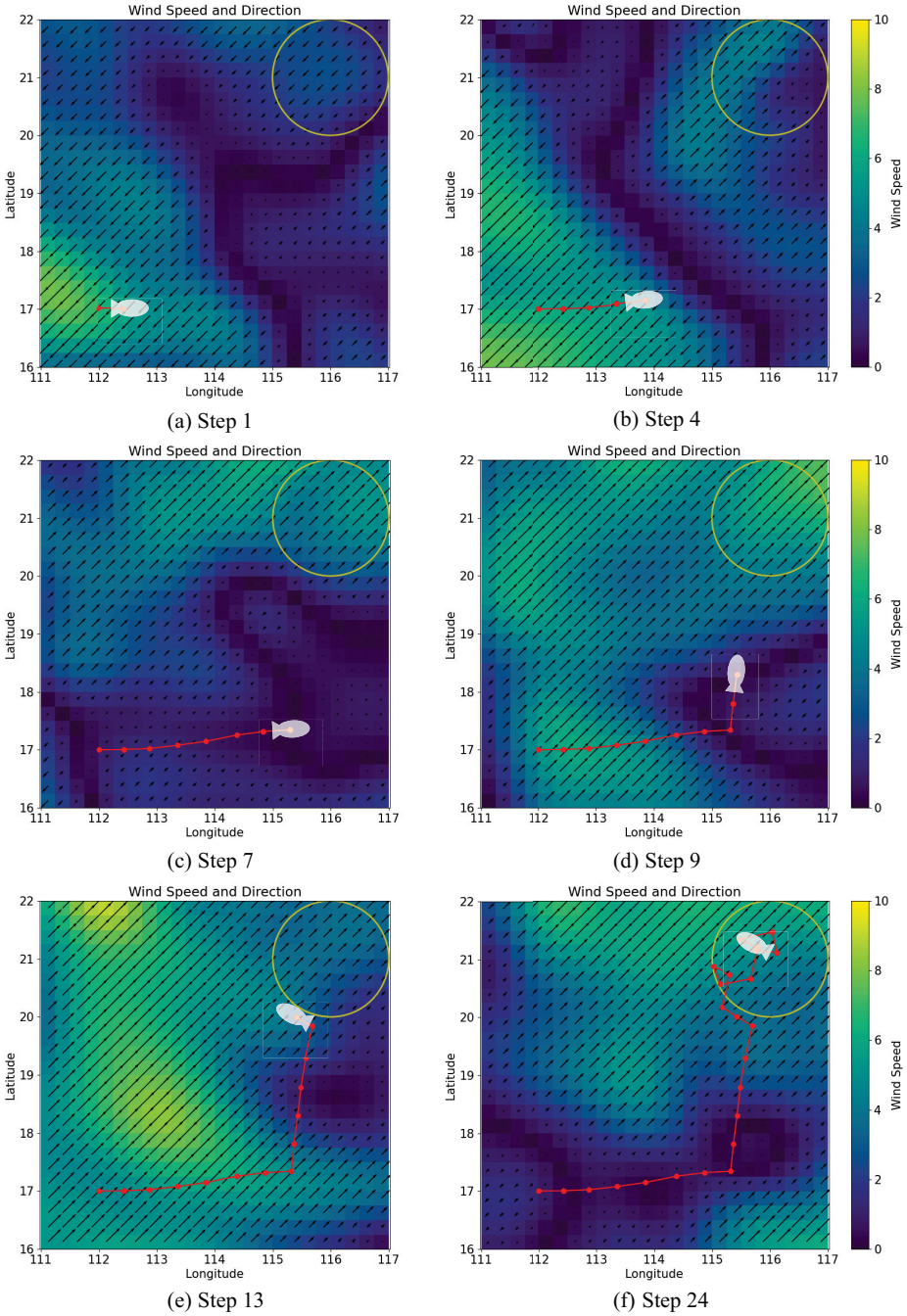
for completing the mission. Since FTD3 considered both current and future wind fields, it initially suggested flying east. By Step 7, the strategy changed to flying north. At Step 9, the airship was still in a favorable low-wind region, and FTD3 suggested continuing north. By Step 13, the airship was approaching the target area and chose to fly upwind to avoid being blown out of the mission area by the strong winds. Between Steps 15 and 24, the airship entered the target area. Despite the winds blowing to the northeast, FTD3 opted to continue flying southeast in the mission area to avoid future strong winds until the target time was reached. From this series of flight strategies, it is evident that FTD3 had successfully learned effective fixed-time trajectory planning through extensive training. It also demonstrated the ability to make informed decisions using wind field data, and thus, it is a practical solution for real-world applications.

In order to verify the generalization of the model under different time constraints (refer to (2.3)), we modified the test time constraint to test the performance of the model under time constraints of 12, 24, and 48 h. Since the time constraint is defined during training, the flight strategy given to the airship by the model during testing is based on the training time constraint. Therefore, changing the test time constraint when testing the model will lead to different degrees of performance degradation. In order to verify the above inference, we further trained the 48-h model based on the training time constraint of 48 h, and the test results are shown in Figure 10. When the planned distance remains unchanged, changing the time constraint means a change in the difficulty of the mission. A shorter time constraint means that the airship has less time to adjust its position to avoid the impact of strong winds. In order to avoid delayed arrival, the airship needs to always move toward the target point. Therefore, when the mission constraint is 12 hours, it has the lowest mission success rate. On the contrary, a longer time constraint allows the airship to flexibly choose the flight direction during the mission, and avoid the impact of strong winds by staying, turning back, and other actions without worrying about delayed arrival. The success rate of all models except the 24-h D3QN model increases with the extension of the test time constraint, which is the effect of the reduction in mission difficulty. The success rate of the 24-h D3QN decreases when the test time constraint increases from 24 h to 48 h because it is better at trajectory planning in the same environment as the training set. Interestingly, the 48-h DDQN outperforms the 24-h DDQN when the test time constraint is 24 h, possibly because the 48-h DDQN has accumulated more experience during training than the 24-h DDQN under the same episode. In summary, under the premise of sufficient training, the same model can still guarantee a certain generalization under different test time constraints, but in practical applications, it is recommended to train models with the same time constraint according to the actual mission time constraint.

Normally, the wind field extracts feature parameters through the convolution layer and then uses it as the input for training. However, we use wind field data preprocessing instead of convolution. We use a 2*2 convolution kernel for convolution processing with a step size of 1, and then use a layer of pooling to ensure that the input dimension is the same as the wind field data preprocessing. After 20,000 learning episodes, the test results are shown in Table 2. Under limited episodes, our method is better than the

Table 2. Parameter values.

Feature extraction method	Success rate
convolution	58.6%
wind field data preprocessing	78.3%



(a) Step 1 (b) Step 4

(c) Step 7 (d) Step 9

(e) Step 13 (f) Step 24

Figure 9. Wind fields and trajectories at some steps in the test scene.

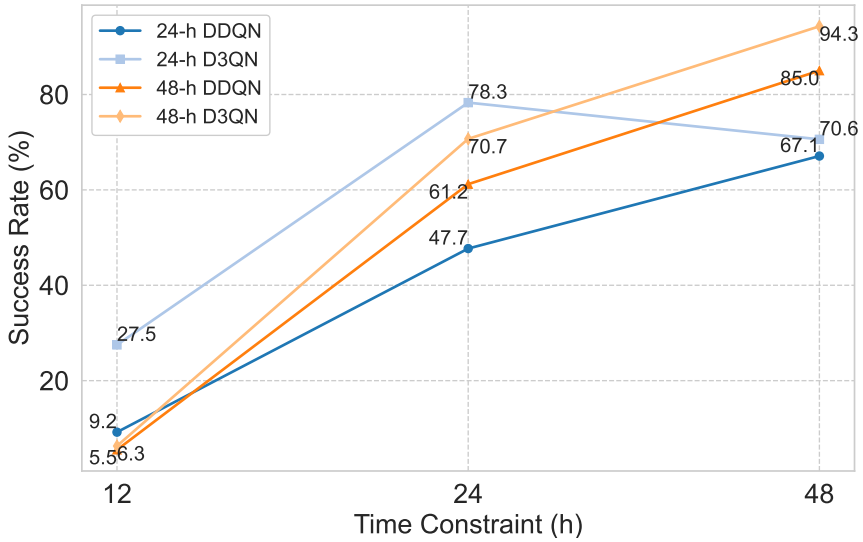


Figure 10. Success rate of the 24-hour model and 48-hour model based on the DDQN and D3QN under different time constraints.

method using convolutional layers. The purpose of both convolution and our preprocessing is to extract useful feature parameters in the wind field to guide the algorithm to make decisions. The difference is that during the training process, the model with convolutional layers needs to learn the weight matrix of the fully connected layer and the weight of the convolution kernel through back propagation, which will prolong the training time. In the mission scenario of this paper, we designed the wind field data preprocessing to obtain feature parameters. Experiments have proved that wind field data preprocessing can obtain useful feature parameters. If useful feature parameters cannot be obtained by other methods, convolution is a reliable method and sufficient training is required.

Table 3. Time consumption.

Model	Training time	Test scenario	Inference time
24-h DDQN	1731 s	12 h	1.58 s
		24 h	1.62 s
		48 h	1.56 s
24-h D3QN	2091 s	12 h	1.57 s
		24 h	1.61 s
		48 h	1.59 s
48-h DDQN	1978 s	12 h	1.57 s
		24 h	1.54 s
		48 h	1.57 s
48-h D3QN	2325 s	12 h	1.56 s
		24 h	1.59 s
		48 h	1.60 s

To ensure real-time application, we tested the training time and inference time of different models. The test results are shown in Table 3. The training time of all models is less than 1 hour, and historical wind field data can be used for training before actual flight. The inference time is independent of the model and is around 1.6 seconds, which can meet the needs of real-time application.

In order to test the impact of different wind data intervals on the reinforcement learning, we tested the models with intervals of 2 and 10 h. The model training process is shown in Figure 11. Compared with other intervals, the model training stability is the best when the interval is 6 h, indicating that the moderate interval selection will maintain a certain degree of predictability while avoiding the introduction of too much irrelevant wind field data after the target time in the later stage of each mission.

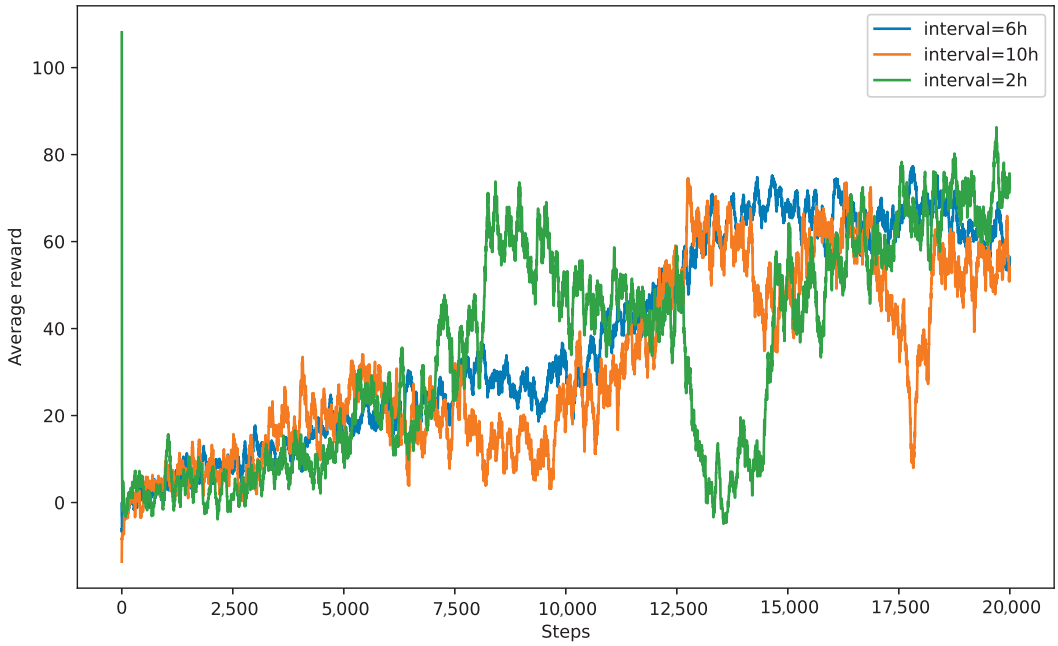


Figure 11. Average reward changes during training at intervals of 2, 6, and 10 h.

5. Conclusions

This paper proposes a novel trajectory planning algorithm for stratospheric airships on fixed-time missions. We designed a reinforcement-learning environment, preprocessed global wind field data (including both current and future wind information), and input it into a reinforcement-learning framework tailored for fixed-time missions. After training, the algorithm could plan a trajectory that satisfied the mission constraints, and its performance was verified under varying wind field conditions. The main conclusions are as follows:

(1) Our proposed FTD3 can well solve the problem of fixed-time trajectory planning for airships under a dynamic wind field. Under the historical wind field, the airship can reach the designated position accurately after 24 h.

(2) FTD3 effectively avoids the overfitting problem often encountered in reinforcement learning.

The comparison results showed that FTD3, based on the D3QN, outperformed the DDQN and DQN in terms of the learning efficiency and success rate. While the DQN-based approach failed to converge, the D3QN achieved a success rate of 78.3%, surpassing the DDQN's success rate of 47.7%.

(3) FTD3 has good robustness and can cope well with the uncertainty of wind field forecast in practical applications.

FTD3 addresses the key challenge of trajectory planning for stratospheric airships, but several challenges remain. For instance, to speed up convergence, we used a discrete action space, but a more continuous action set would be required for real-world applications. Additionally, to reduce the training time, we only considered the current wind field and the wind field 6 hours later, which, while practical, was not sufficient for obtaining the globally optimal trajectory. Future research will focus on improving these aspects.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grants (No. 52402509 and No. 62173016).

Conflict of interest

The authors declare there is no conflicts of interest.

References

1. J. Gonzalo, D. Domínguez, A. García-Gutiérrez, A. Escapa, On the development of a parametric aerodynamic model of a stratospheric airship, *Aerosp. Sci. Technol.*, **107** (2020), 106316. https://doi.org/10.1016/j.ast.2020.106316
2. Z. Zuo, J. Song, Z. Zheng, Q. L. Han, A survey on modelling, control and challenges of stratospheric airships, *Control Eng. Pract.*, **119** (2022), 104979. https://doi.org/10.1016/j.conengprac.2021.104979
3. R. Chai, H. Niu, J. Carrasco, F. Arvin, H. Yin, B. Lennox, Design and experimental validation of deep reinforcement learning-based fast trajectory planning and control for mobile robot in unknown environment, *IEEE Trans. Neural Networks Learn. Syst.*, **35** (2022), 5778–5792. https://doi.org/10.1109/TNNLS.2022.3209154
4. Y. Yang, X. Xiong, Y. Yan, Uav formation trajectory planning algorithms: A review, *Drones*, **7** (2023), 62. https://doi.org/10.3390/drones7010062
5. Y. Li, B. Li, W. Yu, S. Zhu, X. Guan, Cooperative localization based multi-auv trajectory planning for target approaching in anchor-free environments, *IEEE Trans. Veh. Technol.*, **71** (2021), 3092–3107. https://doi.org/10.1109/TVT.2021.3137171

6. E. Zhang, R. Zhang, N. Masoud, Predictive trajectory planning for autonomous vehicles at intersections using reinforcement learning, *Transp. Res. Part C Emerging Technol.*, **149** (2023), 104063. https://doi.org/10.1016/j.trc.2023.104063
7. L. Liu, Q. Shan, Q. Xu, Usvs path planning for maritime search and rescue based on pos-dqn: Probability of success-deep q-network, *J. Mar. Sci. Eng.*, **12** (2024), 1158. https://doi.org/10.3390/jmse12071158
8. C. Dong, Y. Zhang, Z. Jia, Y. Liao, L. Zhang, Q. Wu, Three-dimension collision-free trajectory planning of uavs based on ads-b information in low-altitude urban airspace, *Chin. J. Aeronaut.*, **38** (2025), 103170. https://doi.org/10.1016/j.cja.2024.08.001
9. X. Wu, Y. Yang, Y. Sun, Y. Xie, X. Song, B. Huang, Dynamic regional splitting planning of remote sensing satellite swarm using parallel genetic pso algorithm, *Acta Astronaut.*, **204** (2023), 531–551. https://doi.org/10.1016/j.actaastro.2022.09.020
10. J. Kikuchi, R. Nakamura, S. Ueda, Comparison of transfer trajectory to nrho and operation plan for logistics resupply mission to gateway, *Acta Astronaut.*, **223** (2024), 577–584. https://doi.org/10.1016/j.actaastro.2024.07.038
11. J. Fan, X. Chen, X. Liang, Uav trajectory planning based on bi-directional apf-rrt* algorithm with goal-biased, *Expert Syst. Appl.*, **213** (2023), 119137. https://doi.org/10.1016/j.eswa.2022.119137
12. Y. Zhang, K. Yang, T. Chen, Z. Zheng, M. Zhu, Integration of path planning and following control for the stratospheric airship with forecasted wind field data, *ISA Trans.*, **143** (2023), 115–130. https://doi.org/10.1016/j.isatra.2023.08.026
13. Q. C. Luo, K. W. Sun, T. Chen, Y. f. Zhang, Z. W. Zheng, Trajectory planning of stratospheric airship for station-keeping mission based on improved rapidly exploring random tree, *Adv. Space Res.*, **73** (2024), 992–1005. https://doi.org/10.1016/j.asr.2023.10.002
14. A. Gasparetto, P. Boscariol, A. Lanzutti, R. Vidoni, Path planning and trajectory planning algorithms: A general overview, in *Motion and Operation Planning of Robotic Systems. Mechanisms and Machine Science*, Springer, Cham, **29** (2015), 3–27. https://doi.org/10.1007/978-3-319-14705-5_1
15. H. Jin, R. Xu, P. Cui, S. Zhu, H. Jiang, F. Zhou, Heuristic search via graphical structure in temporal interval-based planning for deep space exploration, *Acta Astronaut.*, **166** (2020), 400–412. https://doi.org/10.1016/j.actaastro.2019.10.002
16. R. Ueda, L. Tonouchi, T. Ikebe, Y. Hayashibara, Implementation of brute-force value iteration for mobile robot path planning and obstacle bypassing, *J. Rob. Mechatron.*, **35** (2023), 1489–1502. https://doi.org/10.20965/jrm.2023.p1489
17. P. Gupta, D. Isele, D. Lee, S. Bae, Interaction-aware trajectory planning for autonomous vehicles with analytic integration of neural networks into model predictive control, in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, (2023), 7794–7800. https://doi.org/10.1109/ICRA48891.2023.10160890
18. Y. Qin, Z. Zhang, X. Li, W. Huangfu, H. Zhang, Deep reinforcement learning based resource allocation and trajectory planning in integrated sensing and communications uav network, *IEEE Trans. Wireless Commun.*, **22** (2023), 8158–8169. https://doi.org/10.1109/TWC.2023.3260304

19. F. Wang, H. Zhang, S. Du, M. Hua, G. Zhong, C-sppo: A deep reinforcement learning framework for large-scale dynamic logistics uav routing problem, *Chin. J. Aeronaut.*, **38** (2025), 103229. https://doi.org/10.1016/j.cja.2024.09.005

20. C. Wu, W. Yu, G. Li, W. Liao, Deep reinforcement learning with dynamic window approach based collision avoidance path planning for maritime autonomous surface ships, *Ocean Eng.*, **284** (2023), 115208. https://doi.org/10.1016/j.oceaneng.2023.115208

21. X. Zheng, J. Cao, B. Zhang, Y. Zhang, W. Chen, Y. Dai, et al., Path planning of prm based on artificial potential field in radiation environments, *Ann. Nucl. Energy*, **208** (2024), 110776. https://doi.org/10.1016/j.anucene.2024.110776

22. L. Qi, X. Yang, F. Bai, X. Deng, Y. Pan, Stratospheric airship trajectory planning in wind field using deep reinforcement learning, *Adv. Space Res.*, **75** (2025), 620–634. https://doi.org/10.1016/j.asr.2024.08.057

23. Y. Wang, B. Zheng, W. Lou, L. Sun, C. Lv, Trajectory planning of stratosphere airship in wind-cloud environment based on soft actor-critic, in *2024 IEEE International Conference on Artificial Intelligence in Engineering and Technology (IICAIET)*, IEEE, (2024), 401–406. https://doi.org/10.1109/IICAIET62352.2024.10730558

24. S. Liu, S. Zhou, J. Miao, H. Shang, Y. Cui, Y. Lu, Autonomous trajectory planning method for stratospheric airship regional station-keeping based on deep reinforcement learning, *Aerospace*, **11** (2024), 753. https://doi.org/10.3390/aerospace11090753

25. M. Xi, J. Yang, J. Wen, H. Liu, Y. Li, H. H. Song, Comprehensive ocean information-enabled auv path planning via reinforcement learning, *IEEE Internet Things J.*, **9** (2022), 17440–17451. https://doi.org/10.1109/JIOT.2021.3137742

26. A. G. S. Junior, D. H. Santos, A. P. F. Negreiros, J. M. V. B. S. Silva, L. M. G. Gonçalves, High-level path planning for an autonomous sailboat robot using q-learning, *Sensors*, **20** (2020), 1550. https://doi.org/10.3390/s20061550

27. S. Woo, J. Park, J. Park, L. Manuel, Wind field-based short-term turbine response forecasting by stacked dilated convolutional LSTMs, *IEEE Trans. Sustainable Energy*, **11** (2019), 2294–2304. https://doi.org/10.1109/TSTE.2019.2954107

AIMS Press