biomimetics

*Article*

# Trajectory Tracking Control of Variable Sweep Aircraft Based on Reinforcement Learning

**Rui Cao** [1] **and Kelin Lu** [2,*]

[1] The College of Information Engineering and Artificial Intelligence, YangZhou University, Yangzhou 225009, China; stdio@yzu.edu.cn
[2] School of Automation, Southeast University, Nanjing 210096, China
* Correspondence: klu@seu.edu.cn

**Abstract:** An incremental deep deterministic policy gradient (IDDPG) algorithm is devised for the trajectory tracking control of a four-wing variable sweep (FWVS) aircraft with uncertainty. The IDDPG algorithm employs the line-of-sight (LOS) method for path tracking, formulates a reward function based on position and attitude errors, and integrates long short-term memory (LSTM) units into IDDPG algorithm to enhance its adaptability to environmental changes during flight. Finally, environmental disturbance factors are introduced in simulation to validate the designed controller's ability to track climbing trajectories of morphing aircraft in the presence of uncertainty.

**Keywords:** morphing aircraft; deep deterministic policy gradient; path tracking; environmental disturbance



**Citation:** Cao, R.; Lu, K. Trajectory Tracking Control of Variable Sweep Aircraft Based on Reinforcement Learning. *Biomimetics* **2024**, *9*, 263. https://doi.org/10.3390/biomimetics9050263

Academic Editors: Antonio Concilio, Cristian Vendittozzi, Rosario Pecora, Salvatore Ameduri and Yu Yang

Received: 20 February 2024
Revised: 23 April 2024
Accepted: 25 April 2024
Published: 27 April 2024



## 1. Introduction

The morphing aircraft is a novel conceptual aircraft with deformable structure, which can autonomously adapt its morphology in response to changes in flight environment and mission requirements [1,2]. In comparison to traditional fixed-wing aircraft, morphing aircraft exhibit significant advantages on multiple fronts: firstly, the deformable structure of morphing aircraft can be utilized to enhance aerodynamic characteristics, reduce flight energy consumption, and expand the flight envelope; secondly, active morphing assists in maneuvering, thereby augmenting control capabilities; additionally, morphing aircraft can adapt to various flight environments and mission requirements by altering configurations, thereby broadening its applicability [3,4].

For morphing aircraft, each configuration will generate unique performance (such as speed, climb rate, range, endurance, etc.) under specific flight conditions (Mach number, altitude, angle of attack, and sideslip angle). For a given flight profile, optimal morphologies for various flight phases can typically be theoretically calculated. However, in practical flight missions, obtaining all optimal configurations within the entire flight envelope is challenging. Moreover, mission parameters may be modified or entirely changed during flight. In such cases, the deformation strategy derived from the theoretical method may not be globally optimal. Real-world mission requirements underscore the need for morphing aircraft to evolve toward intelligence and autonomy to effectively adapt to increasingly complex flight environments.

In the past three decades, artificial intelligence (AI) technologies have undergone rapid development [5,6]. Various AI approaches, such as reinforcement learning [7,8] and deep reinforcement learning [9], have provided more intelligent solutions for deformation strategies in morphing aircraft. Deep learning, a key branch of machine learning, employs multi-layer neural networks for data perception and representation, demonstrating robust capabilities in handling complex classification tasks [10]. Reinforcement learning engages in iterative learning through continual trial and error, enhancing its ability to make behavioral decisions by interacting with the environment and receiving feedback. The core of

reinforcement learning lies in modifying its own strategy based on evaluative feedback signals from the environment, thereby achieving decision optimization [11]. Common reinforcement learning algorithms include deep Q-network (DQN) [12], deep deterministic policy gradient (DDPG) [8], proximal policy optimization (PPO) [13], among others. Ref. [14] proposes a Q-learning-based adaptive control method for a swept-wing aircraft. However, this method faces limitations due to an excessively discrete action space. Additionally, both the Q-function and reward function are solely dependent on the aircraft's configuration rather than its flight state, resulting in a narrow applicability scope. The DDPG algorithm is a commonly used algorithm in deep reinforcement learning, combining deterministic policy gradient methods with DQN. It employs deep neural networks to approximate state action policies and is suitable for handling continuous action problems. Ref. [15] designs a DDPG algorithm to learn deformation strategies under both symmetric and asymmetric conditions for a morphing aircraft with simultaneous changes in wing span and sweep angle. The literature [16] introduces an enhanced DDPG algorithm for the design of deformation strategies in a morphing unmanned aerial vehicle (UAV), allowing for the deformation of wing sweep angle, wing span, wing area, and other flight structures based on environmental conditions and mission objectives.

Reinforcement learning control strategies enable aircraft to converge from various initial states to predefined endpoints. However, during the training of the learning algorithm, the large action space often results in slow training speeds. To address this issue, this study draws inspiration from trajectory tracking control design principles. In the simulated environment of reinforcement learning, a reference trajectory is introduced. Thus, the action space is defined as the error between the current action and the reference trajectory, thereby reducing the action search space. Additionally, system reference trajectories are typically based on nominal models, but actual systems may exhibit model uncertainties and obstacles [17,18]. Model uncertainty significantly affects the control effectiveness of traditional trajectory tracking control methods. Addressing the challenges posed by model uncertainty and obstacle presence, this study incorporates long short-term memory (LSTM) recursive neural networks [19,20] into the reinforcement learning algorithm to record the positions of reference trajectories and obstacles. Consequently, reinforcement learning-based trajectory tracking control can optimize an optimal trajectory and control strategy near the reference trajectory, matching the real model, and enhancing computational efficiency based on historical flight data.

This paper introduces an incremental model with uncertainties for a four-wing variable-sweep (FWVS) aircraft and redesigns the action and state spaces based on this model. In Section 3, line-of-sight (LOS) is employed as a path-tracking method with a reference trajectory, and a reward function based on position and attitude errors is designed. Section 4 integrates the LSTM into the DDPG algorithm framework to enhance its real-time adaptability to environmental changes during flight. The resulting algorithm, IDDPG, is established. In Section 5, through simulation, the IDDPG algorithm demonstrates faster convergence and achieves integrated tracking control of climb trajectories for a variant aircraft model with uncertainties using deep reinforcement learning. To assess the adaptability of the controller to environmental disturbances, such as obstacles, simulations with environmental interference are conducted. Section 6 provides a summary of the entire work.

## 2. Model Description

### *2.1. Mathematical Model of Four-Wing Variable-Sweep Aircraft*

The subject of this study is a FWVS aircraft with a tandem wing configuration. Typically, this type of tandem wing aircraft employs a passive variable-sweep angle scheme, with the main wings folding before takeoff and expanding during flight to reduce the fuselage size while ensuring sufficient lift for the aircraft [16]. This type of aircraft can control flight attitude and trajectory by adjusting the sweep angle of its four main wings, simplifying control surfaces. In principle, this design enables a stable flight without ailerons

or horizontal tails, presenting significant practical and theoretical research value. It is important to note that this morphing aircraft only considers symmetric deformation.

The control inputs for the FWVS aircraft are the variation in the sweep angles of its four main wings, as shown in Figure 1. These angles represent the rotation of the $z$-axis in each wing coordinate system with respect to the body coordinate system. Here, the $x_i$-axis is aligned with the wing coordinate system's $Ox_ay_a$ plane, and the $y_i$-axis passes through the wing rotation center, pointing to the right side of the aircraft. The sweep angles of each main wing are denoted as $a_1, a_2, a_3, a_4$.

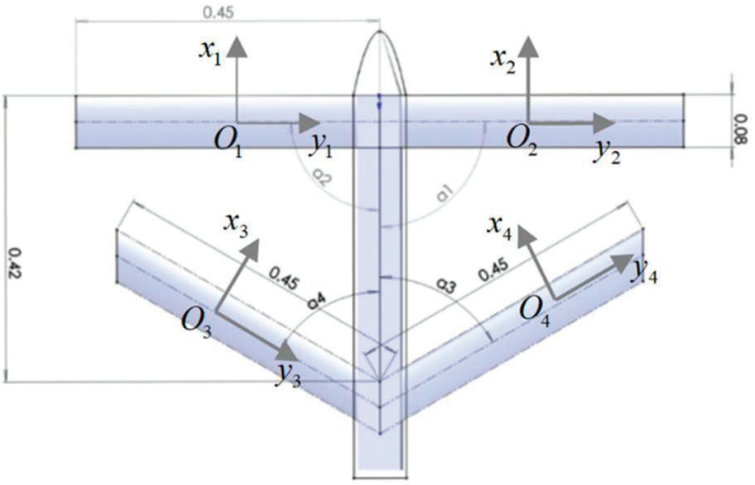



**Figure 1.** Schematic diagram of the FWVS aircraft.

Adopting clockwise rotation as positive, during symmetric deformation, $a_1 = -a_2 < 0$, $a_3 = -a_4 < 0$, and $|a_i| \in [0^o, 30^o]$. Following the definition in Ref. [21], the control variable is simplified as the deformation rate:

$$\lambda_1 = \frac{|a_1|}{30^o}, \lambda_2 = \frac{|a_3|}{30^o}, \tag{1}$$

that is $\lambda_i \in [0, 1]$. $i = 1, 2$ represent the front wing and rear wing, respectively.

This study employs OPENVSP 3.21.1 software for aerodynamic analysis of the FWVS aircraft. Under various sweep angle deformations and flight conditions (including angle of attack, Mach number, and Reynolds number), the longitudinal aerodynamic force and moment coefficients of this aircraft are computed. Ultimately, polynomial fitting is utilized to obtain expressions for lift coefficient $C_L$, drag coefficient $C_D$, and pitch moment coefficient $C_m$.

$$\begin{aligned}
C_L = {} & 0.01 \times \Big(47.95 - 4.077\lambda_1^2 - 4.579\lambda_2^2 + 16.89\lambda_1^2\lambda_2 \\
& + 17.44\lambda_1\lambda_2^2 - 20.41\lambda_1\lambda_2 - 16.23\lambda_1^2\lambda_2^2\Big)(9.448\alpha + 0.3397), \\
C_D = {} & 0.01 \times \Big(83.58 - 5.229\lambda_2 - 0.1296\lambda_2^2 - 4.34\lambda_1 \\
& - 0.409\lambda_2 + 3.595\lambda_1\lambda_2)\Big(28.85\alpha^2 + 0.2363\alpha + 0.8429\Big), \\
C_m = {} & 0.01 \times \Big(-8.103 - 11.67\lambda_1^2 - 4.252\lambda_2^2 - 27.26\lambda_1 + 37.47\lambda_2\Big) \times \\
& \Big(-8.207\alpha^2 + 10.03\alpha + 0.34\Big) + (7.248\lambda_1 + 36.89\lambda_2 - 69.24)q,
\end{aligned} \tag{2}$$

where $\alpha$ is the angle of attack, and $q$ denotes the pitch angle rate.

During the deformation process, changes in the sweep angle induce variations in the center of mass and aerodynamic center, leading to alterations in pitch moment. The

rotational velocity and acceleration of the wing give rise to additional inertial forces $F_{xdyn}$, $F_{zdyn}$, and moment $M_{dyn}$, which can be expressed as follows [21]:

$$\begin{cases} F_z = mg\cos\theta - D\sin\alpha - L\cos\alpha \\ F_{xdyn} = 2m_a l\left(\dot{a}_1^2\cos a_1 - \ddot{a}_1\cos a_1 + \dot{a}_3^2\cos a_3 - \ddot{a}_3\cos a_3\right) \\ F_{zdyn} = 2m_a l\left(\dot{a}_1^2\sin a_1 - \ddot{a}_1\sin a_1 + \dot{a}_3^2\sin a_3 - \ddot{a}_3\sin a_3\right) \\ J_y = 0.0242 + 4\times 2.6\times 10^{-5} + 4m_a c^2 + m_a\left(l_1^2 + l_2^2 + l_3^2 + l_4^2\right) \\ M_{dyn} = 2m_a ql(\dot{a}_3 l_2\cos a_3 - \dot{a}_1 l_1\cos a_1) + \\ \qquad 2m_a l(q\sin\theta + Fz/m)(\sin a_1 + \sin a_3) - \\ \qquad 2m_a cl\left(\dot{a}_1\cos a_1 - \ddot{a}_3\cos a_3 - \dot{a}_1^2\sin a_1 + \dot{a}_3^2\sin a_3\right) \\ M_{Gdyn} = -2m_a gl\cos\theta(\sin a_1 + \sin a_3) \end{cases} \tag{3}$$

where $F_z$ represents the sum of external forces acting in the body's $z$-axis direction, $J_y$ denotes the moment of inertia, $c$ is the mean aerodynamic chord length, $l$ is the mean arm length, $l_i$ is the wingspan of the $i$th main wing, and $M_{Gdyn}$ represents the additional inertial pitch moment caused by changes in the center of mass.

This paper primarily focuses on the longitudinal motion of the FWVS aircraft. Therefore, it is assumed that the lateral attitudes are all zero. Consequently, the simplified longitudinal kinematic equations can be obtained as follows:

$$\begin{cases} \dot{x} = V\cos\gamma \\ \dot{h} = V\sin\gamma \\ \dot{V} = \left(T\cos\alpha - D - mg\sin\gamma + F_{xdyn}\cos\alpha + F_{zdyn}\sin\alpha\right)/m \\ \dot{\gamma} = \left(T\sin\alpha + L - mg\cos\gamma - F_{zdyn}\cos\alpha + F_{xdyn}\sin\alpha\right)/mV \\ \dot{\alpha} = q - \dot{\gamma} \\ \dot{q} = \left(M + M_{Gdyn} + M_{dyn}\right) \end{cases} \tag{4}$$

where $T$ is thrust, $x$ denotes horizontal displacement, $h$ represents flight altitude, $V$ indicates flight velocity, and $\gamma$ expresses the flight path angle.

### *2.2. Kinematic Incremental Model of the FWVS Aircraft*

Here, we first formalize the dynamic model of the FWVS aircraft into a general form. The aircraft controls its position and attitude by simultaneously adjusting the sweep angles of the front and rear wings, as well as the thrust magnitude. Thus, in Equation (4), $\boldsymbol{u} = [\lambda_1, \lambda_2, T]$ represents the control input and $\boldsymbol{X} = [x, h, V, \alpha, \gamma, q]$ expresses the output state variables, and we can obtain

$$\dot{\boldsymbol{X}} = f(\boldsymbol{X}, \boldsymbol{u}). \tag{5}$$

Based on the description of the aircraft kinematics in Equation (5), for a given reference trajectory $\boldsymbol{X}_r = [x_r, h_r, V_r, \alpha_r, \gamma_r, q_r]$ and $u_r = [\lambda_{1r}, \lambda_{2r}, T_r]$, the following representation can be formulated:

$$\dot{\boldsymbol{X}}_r = f(\boldsymbol{X}_r, \boldsymbol{u}_r), \tag{6}$$

among them, $r$ represents the reference state.

By subtracting Equation (5) from Equation (6), the incremental model for this morphing aircraft can be obtained

$$\Delta\dot{\boldsymbol{X}} = f(\Delta\boldsymbol{X} + \boldsymbol{X}_r, \Delta\boldsymbol{u} + \boldsymbol{u}_r) - f(\boldsymbol{X}_r, \boldsymbol{u}_r), \tag{7}$$

where $\Delta\boldsymbol{X} = \boldsymbol{X} - \boldsymbol{X}_r, \Delta\boldsymbol{u} = \boldsymbol{u} - \boldsymbol{u}_r$.

Due to the strong nonlinearity and time-varying characteristics of this morphing aircraft, obtaining an accurate mathematical model is challenging. Reinforcement learning is particularly suitable for such cases with imprecise models. Considering the uncertainty

in the model, this paper incorporates it into the model parameters. We assume uncertainty in the lift coefficient of the system, expressed in the following form:

$$\begin{aligned} C_L &= \big(0.01 \times \big(47.95 - 4.077\lambda_1^2 - 4.579\lambda_2^2 + 16.89\lambda_1^2\lambda_2 \\ &\quad +17.44\lambda_1\lambda_2^2 - 20.41\lambda_1\lambda_2 - 16.23\lambda_1^2\lambda_2^2\big)(9.448\alpha + 0.3397)\big)(1+\varepsilon) \end{aligned} \tag{8}$$

where $\varepsilon \in [-0.2, +0.2]$ represents an uncertain parameter with fixed boundaries. This indicates that the range of uncertainty for $C_L$ is between $-20\%$ and 20%. Finally, the lift coefficient $C_L$ is incorporated into the incremental model (7) to obtain the incremental model of the FWVS aircraft with uncertainty.

## 3. Reinforcement Learning Environment Design

For reinforcement learning, if the action space is designed to be excessively large, it can result in high computational complexity during network training and slow convergence speed. Considering that an aircraft follows a corresponding reference trajectory during actual flight, to address the issue of slow training convergence due to an excessively large action space, this study constrains the action space around the reference trajectory during network training. This approach aims to reduce computational complexity. In configuring the reinforcement learning environment, the incremental kinematic model based on the reference trajectory, as depicted in Equation (7), is employed.

### *3.1. Action Space and State Space Design*

To apply reinforcement learning algorithms to the FWVS aircraft, it is essential to establish the reinforcement learning environment. It includes defining and designing the state space, action space, and reward function. The reinforcement learning environment not only needs to be a Markov decision process but also must meet the mission requirements of the aircraft. In this section, the deformation process is represented as a standard Markov decision process [22], consisting of states $S$, actions $A$, rewards $R$, and discount factor $\eta$. If we analogize the reinforcement learning model to a traditional controller, the inputs can be likened to actions and outputs can be considered as state values. Hence, the state variables consist of current flight parameters describing the flight status of the aircraft, specifically flight altitude $h$, horizontal displacement $x$, flight speed $V$, pitch angle rate $q$, angle of attack $\alpha$, and flight path angle $\gamma$.

The reinforcement learning task designed in this study is trajectory tracking. To reduce exploration in the action space and expedite convergence, the action space is designed in proximity to the reference trajectory. Specifically, it is represented as $A = [\delta_{\lambda 1}, \delta_{\lambda 2}] \in [-0.1, 0.1]$, signifying the selection constraint of actions within a range of $\pm 0.1$ from the reference trajectory. Here, $\delta_{\lambda_i} = \lambda_i - \lambda_{ri}$ represents the difference between the current actual value and the reference value. Based on the flight mission and reference trajectory data for the variant aircraft, a fixed thrust value of 7.3 N is set during the actual algorithm execution.

### *3.2. Reward Function Design Based on LOS*

Reinforcement learning is an algorithm that relies on reward functions for training, analogous to deep learning where the reward function serves as a supervisory signal. Therefore, a reasonable designed reward function is crucial. In the field of reinforcement learning, reward functions can be categorized as sparse and non-sparse rewards [23]. Sparse rewards imply that an effective reward signal is received only after completing a specific task. This implies that, during the interaction between the agent and the environment, no rewards are obtained, which is highly detrimental to exploration. Consequently, the agent receives a reward of zero for most actions due to the vast action space, leading to numerous futile explorations and slow convergence. Hence, this paper adopts a composite reward function combining sparse rewards with non-sparse rewards to expedite convergence while actively guiding the completion of flight tasks.

3.2.1. Sparse Reward Design

Initially, a sparse reward approach is employed to address the task success, designing a terminal reward. Physically, upon reaching the target point, the agent receives a positive reward, while exceeding a threshold based on the aircraft's own state results in a negative reward. If either condition is triggered, the state of the variant aircraft is reset, initiating the next flight task. The design is as follows:

Initially, a sparse reward mechanism is employed to design the terminal reward. Physically, upon reaching the target point, the aircraft receives a positive reward $r_c$, while exceeding the state threshold of aircrafts results in a negative reward $r_o$. If either condition is triggered, the state of the variable aircraft is reset, initiating the next flight task. The design of $r_c$ is as follows:

$$r_c = t \leq t_f \& \left|h - h_{\text{target}}\right| \leq \delta_h \& \left|V - V_{\text{target}}\right| \leq \delta_V \& \left|\gamma - \gamma_{\text{target}}\right| \leq \delta_\gamma \& |q| \leq \delta_q, \quad (9)$$

where $t_f$ represents the maximum time to complete the task. $h_{\text{target}}$, $V_{\text{target}}$, and $\gamma_{\text{target}}$ denote the target values for flight altitude, speed, and flight path angle, respectively. $\delta_h$, $\delta_V$, $\delta_\gamma$, and $\delta_q$ express the maximum allowable error ranges for altitude, speed, flight path angle, and pitch angle rate, respectively.

The design of $r_o$ is as follows:

$$r_o = t > t_f || h \notin [h_{\min}, h_{\max}] || V \notin [V_{\min}, V_{\max}] || |q| > q_{\max} || \gamma \notin [\gamma_{\min}, \gamma_{\max}] || |\alpha| > \alpha_{\max}, \quad (10)$$

where $h_{\max}$ and $h_{\min}$ represent the maximum and minimum values for flight altitude constraints, respectively. $V_{\max}$ and $V_{\min}$ denote the maximum and minimum values for flight speed constraints. $q_{\max}$ expresses the maximum permissible pitch angle rate and $\alpha_{\max}$ signifies the maximum allowable angle of attack. $\gamma_{\max}$ and $\gamma_{\min}$ are the maximum and minimum values for flight path angle constraints.

Furthermore, to enhance the reward for exemplary flight states and increase the diversity of samples [24], substantial coefficient factors are applied to the two rewards mentioned above. The terminal reward $R_f$ is then obtained by combining the two designed rewards:

$$R_f = 600 r_c - 100 r_o. \quad (11)$$

3.2.2. Non-Sparse Reward Design

Non-sparse reward is a deliberately crafted "dense reward", typically manifested as a reward function associated with the state. To enhance the utility of flight process samples, the following action reward function and state reward function have been designed based on the concept of non-sparse rewards:

(1) Action Reward: The morphing aircraft must ensure stable flight during the deformation process, and to achieve smoother trajectories, an action penalty function $R_a$ is designed as follows:

$$R_a = w_3 |\Delta A|, \quad (12)$$

where $\Delta A$ represents the change in actions and $w_3$ is a negative value used to guide action $\lambda_1$ and $\lambda_2$ toward slow variations.

(2) LOS-based State Reward: The reinforcement learning algorithm should guide the FWVS aircraft to track the reference trajectory and accomplish the tracking flight task. Therefore, a process reward $R_m$ based on the LOS method is designed.

The line-of-sight method [25] is a guidance approach that provides the controlled object with a visual range, aiming to perform path tracking within this range. For UAVs, the field of view is defined as a circular area with its own center of mass as the center and a radius of $R$. Assuming a decoupled vertical plane for path tracking, an appropriate radius is selected to ensure that this circle has two points with the target path. The closer intersection to the UAV is chosen as the target point. The direction from the aircraft's

current position to this point is considered as the target heading $\psi_{\text{LoS}}$, where $\Delta\psi$ represents the angle between the heading line and the $x$-axis. Assuming that the heading direction and the desired path are in the same horizontal plane, the principle of this method is illustrated in Figure 2, when there are two intersections between the expected path and the circular field of view. Let $(x_d, h_d)$ be the intersection point between the circular field of view and the desired path; then, the target heading angle can be obtained as follows:

$$\Delta\psi = \arctan \frac{y_d}{\sqrt{R^2 - y_d^2}}. \tag{13}$$

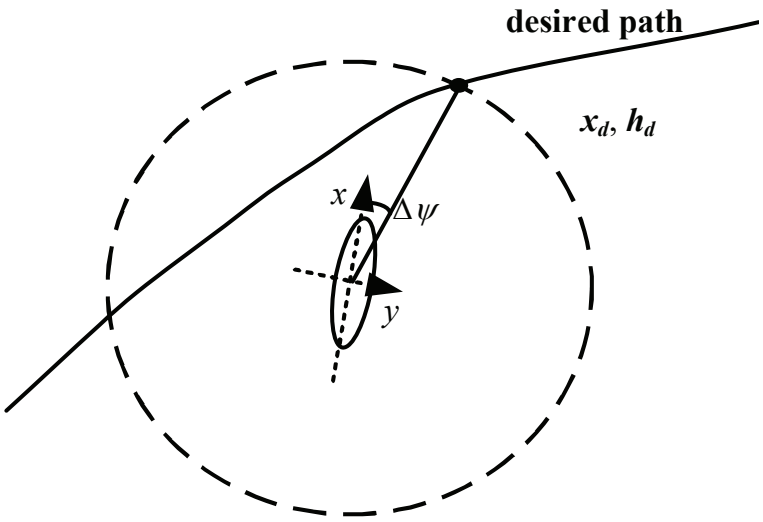



**Figure 2.** Schematic diagram of LoS path tracking method.

Additionally, when there are no intersection points between the desired path and the circular field of view, i.e., when the UAV is too far from the desired path, the vertical direction is selected as the target heading. The objective is to approach the desired path at the fastest speed.

Subsequently, based on this method, a process reward for path tracking is designed. The variables directly related to the path include the relative distance $D$ and heading angle $\psi$ between the morphing aircraft and the reference trajectory, which can be obtained at any moment. Let the position vector at the current time be denoted as $\boldsymbol{X}_{\text{UAV}} = [x_1, h_1]$, and the position vector of the target be denoted as $\boldsymbol{X}_{\text{target}} = [x_2, h_2]$. Then, the relative distance $D$ and heading angle $\psi$ can be defined as follows:

$$\begin{aligned} D &= \sqrt{(x_1 - x_2)^2 + (h_1 - h_2)^2}, \\ \psi &= \arctan \tfrac{h_1 - h_2}{x_1 - x_2}. \end{aligned} \tag{14}$$

To ensure convergence, the process reward $R_m$ is designed in the exponential function form to guide the aircraft toward the target trajectory:

$$R_m = 0.8e^{-5\times10^{-4}D} + 0.2e^{-2\times10^{-3}\psi}. \tag{15}$$

For this flight mission, the overall reward function $R_{\text{total}}$ is a synthesis of the various rewards mentioned above. It ensures that the variant aircraft consistently satisfies process constraints while completing smooth flight during the autonomous decision-making process. The form of the overall reward function $R_{\text{total}}$ is

$$R_{\text{total}} = R_f + R_a + R_m. \tag{16}$$

## 4. Morphing Aircraft Tracking Control Method Based on IDDPG

The morphing aircraft, during its flight, not only needs to track the reference trajectory but also requires obstacle avoidance. Thus, we integrate LSTM [19], a network with unique memory units, into both the Actor and Critic networks to receive environment data with temporal features. This makes the learning model better understand the dynamic changes between aircraft states.

### *4.1. LSTM Recurrent Neural Network*

The LSTM network is an improved type of recurrent neural network, consisting mainly of four components: the forget gate, input gate, output gate, and memory cell. LSTM not only addresses the short-term memory issue but also mitigates problems such as gradient vanishing and exploding in the loss function. The algorithmic framework is illustrated in Figure 3, where $x_t$ represents the input, $\sigma$ and tanh denote the sigmoid function and hyperbolic tangent function. $f_t$, $i_t$, and $o_t$ express the computed results of the forget gate, input gate, and output gate, respectively. $c'_t$ represents the candidate for the current memory cell, $c_t$ is the updated cell state, and $h_t$ is the state of the hidden layer. $w_f$, $w_i$, $w_c$, and $w_o$ indicate the weight matrices for the respective components. Thus, $c_t$ and $h_t$ are expressed as follows:

$$\begin{cases} c_t = f_t^* c_{t-1} + i_t^* c'_t \\ h_t = o_t \tanh(c_t) \end{cases} \tag{17}$$

where

$$\begin{aligned} f_t &= \sigma\left(w_f \cdot [h_{t-1}, x_t] + b_f\right) \\ i_t &= \sigma(w_i \cdot [h_{t-1}, x_t] + b_i) \\ c'_t &= \tanh(w_c \cdot [h_{t-1}, x_t] + b_c) \\ o_t &= \sigma(w_o \cdot [h_{t-1}, x_t] + b_o) \end{aligned} \tag{18}$$

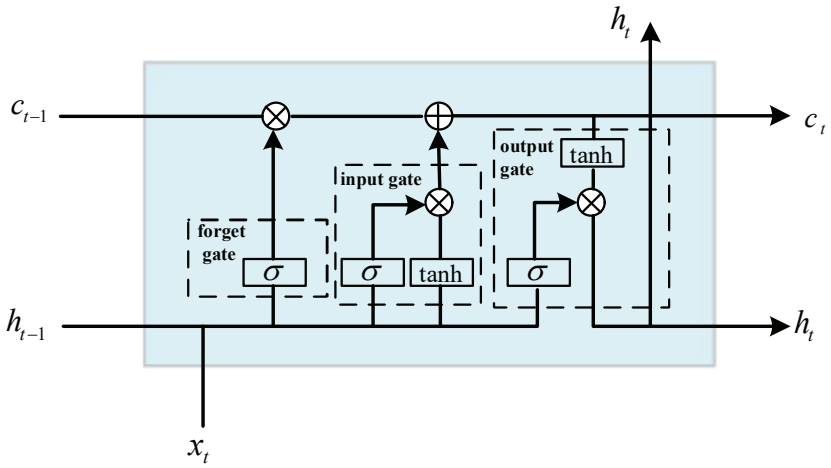



**Figure 3.** LSTM algorithm framework.

### *4.2. The IDDPG Algorithm Design Process*

For complex environments and tasks, learning the optimal policy from scratch after specifying a reward function involves a computationally intensive process. Therefore, a pre-training approach [23] is employed, utilizing expert data to facilitate faster convergence during training.

Specifically, the IDDPG algorithm is based on the incremental kinematic model of a morphing aircraft, dividing the entire reinforcement learning process into two phases: pre-training and DDPG training. Pre-training essentially involves supervised learning,

utilizing high-quality data for behavioral cloning to construct the initial action policy. The learning objective expression is

$$\mathcal{L}(a^*, \pi_\theta(s)) = \|a^* - a_\theta\|^2 \tag{19}$$

where $\pi_\theta(s)$ represents the reinforcement learning policy, $\theta$ is the hyperparameter for reinforcement learning, $a^*$ expresses the expert action, and $a_\theta$ is the action output by the learning network. $\mathcal{L}$ indicates the error loss function. The closer the output action is to the expert action based on the loss function, the better the learning outcome.

The initial policy network parameters obtained from pre-training are input into the Actor network in the second phase for online training. This stage employs the DDPG algorithm, and as the actions at each time step are deterministic, noise is introduced to enhance policy exploration.

$$a_t = \mu(s_t|\theta^\mu) + \mathcal{N}_t, \tag{20}$$

where $s_t$ and $a_t$ represent the state and action at time $t$, respectively. $\mu(s_t|\theta^\mu)$ signifies the parameterized policy network responsible for obtaining actions corresponding to the state $s_t$. $\mathcal{N}_t$ denotes time-dependent Ornstein–Uhlenbeck (OU) noise. OU noise differs from Gaussian noise in that the difference between adjacent steps is not significantly large; instead, it tends to explore a certain distance in the direction of the mean based on inertia from the previous step, either positively or negatively. This is advantageous for exploration in a specific direction.

For the studied FWVS aircraft's flight states in this paper, estimation is performed using a differentiable function approximator $Q\left(s_t, a_t|\theta^Q\right)$. At each time step $t$, $N$ mini-batch data are sampled from the buffer to update the parameters of the Actor and Critic networks. The parameter updates for the $Q$ network employ the temporal-difference algorithm, minimizing the mean squared error of the loss function $\mathcal{J}$ at each time step $t$.

$$\mathcal{J} = \frac{1}{N}\sum_t \left(y_t - Q\left(s_t, a_t|\theta^Q\right)\right)^2. \tag{21}$$

In reinforcement learning, the objective is to find the optimal policy $\pi^*$ that maximizes the expected return. In the case of continuous control, the gradient $\nabla_{\theta^\mu}\mathcal{J}$ is taken with respect to the hyperparameters $\theta$ to update the policy network. By using the expression in Equation (21), the expression for $\nabla_{\theta^\mu}\mathcal{J}$ is obtained as shown in Equation (22).

$$\nabla_{\theta^\mu}\mathcal{J} \approx \frac{1}{N}\sum_t \nabla_a Q\left(s_t, a_t|\theta^Q\right)|_{s=s_t, a=\mu(s_t|\theta^\mu)} \nabla_{\theta^\mu}\mu(s_t|\theta^\mu)|_{s_t}. \tag{22}$$

The IDDPG algorithm adopts the framework of DQN with dual networks. It requires simultaneous updating of the hyperparameters for both the $Q$ network $Q(s, a|\theta^Q)$ and policy network $\mu(s|\theta^\mu)$. This is typically achieved through the following soft update method:

$$\begin{cases} \theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'} \\ \theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'} \end{cases} \tag{23}$$

where $\theta^{Q'}$ and $\theta^{\mu'}$ represent the corresponding network parameters after updating the $Q$ network and the policy network, respectively. $\tau$ denotes the soft update rate, with values in the range of [0, 1]. Figure 4 illustrates the learning training framework of the IDDPG policy.

In Figure 4, the environmental model within the entire algorithm framework is the previously established FWVS aircraft. The aircraft, based on the policy network, adjusts the variable-sweep angles, resulting in new flight states. These states are fed back into the policy network, creating a closed loop. Considering the need for the morphing aircraft to choose deformation strategies based on different flight states during training, OU noise is introduced to enhance exploration. The action space is defined as the increments in

the variable rates of wings $\delta_{\lambda_1}$ and $\delta_{\lambda_2}$, and the state space is defined as a continuous six-dimensional flight state. Additionally, the IDDPG algorithm incorporates LSTM networks into both the Actor and Critic networks, enhancing memory for reference trajectories and obstacle positions to further improve algorithm performance. Furthermore, the algorithm stores experiences $[s_t, a_t, r_t, s_{t+1}]$ in a memory cell during the training process (where $r_t$ is the single-step reward generated under the action at time $t$). The learning process maps the current state to the optimal action, and based on the received rewards, computes gradients to update the neural network parameters, ultimately obtaining the optimal deformation strategy during the acceleration climb.

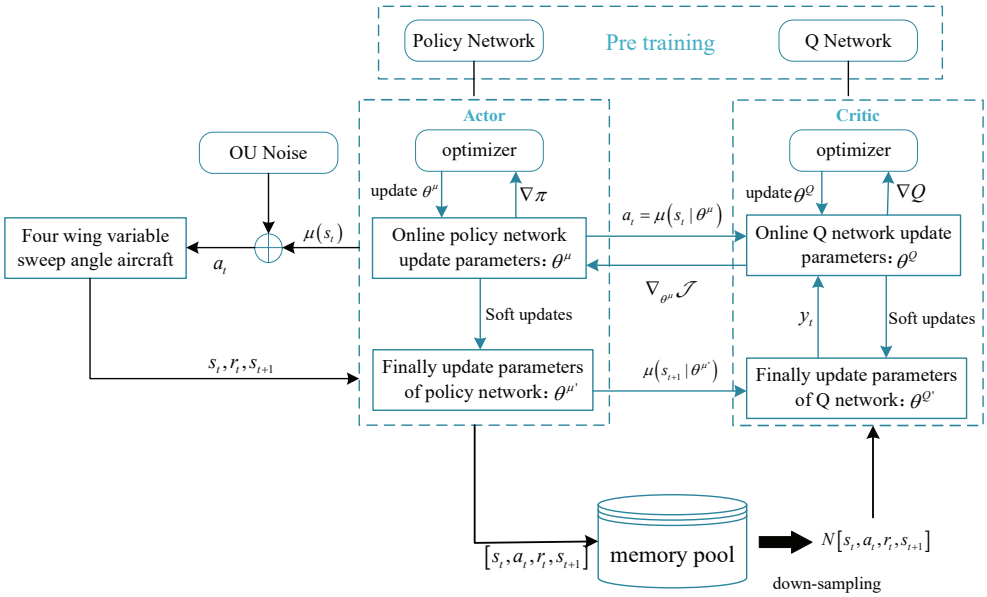



**Figure 4.** Learning and training framework of FWVS aircraft.

### *4.3. Design and Training of the IDDPG Network*

#### 4.3.1. Design of the IDDPG Network

The environmental data received by the aircraft exhibit temporal characteristics during task execution. The LSTM network, with its unique memory units, is better suited for capturing the positions of obstacles and reference trajectories [26].

The environmental data received by the aircraft exhibit temporal characteristics during task execution. The LSTM network, with its unique memory units, is better suited for capturing the positions of obstacles and reference trajectories [26]. The overall network framework of the IDDPG algorithm is illustrated in Figure 5. The Actor network comprises two LSTM layers and one fully connected layer (denoted as FC), responsible for generating control signals, specifically the variable wing sweep rate. The Critic network consists of two parts. The first part, denoted as FC1, is a fully connected network responsible for processing outputs from the Actor network. The second part is an LSTM network, mirroring the structure of the Actor network, tasked with handling environmental state information. Finally, the results from these two parts are processed through another fully connected network, FC2, to calculate and output the $Q$ value of the Actor network. Considering the task environment, 128 hidden layer neurons were chosen for the LSTM network.

From the above algorithmic process, it is evident that the IDDPG algorithm is based on the Actor-Critic network. The structure of the network significantly influences the algorithm's performance, necessitating design optimization.

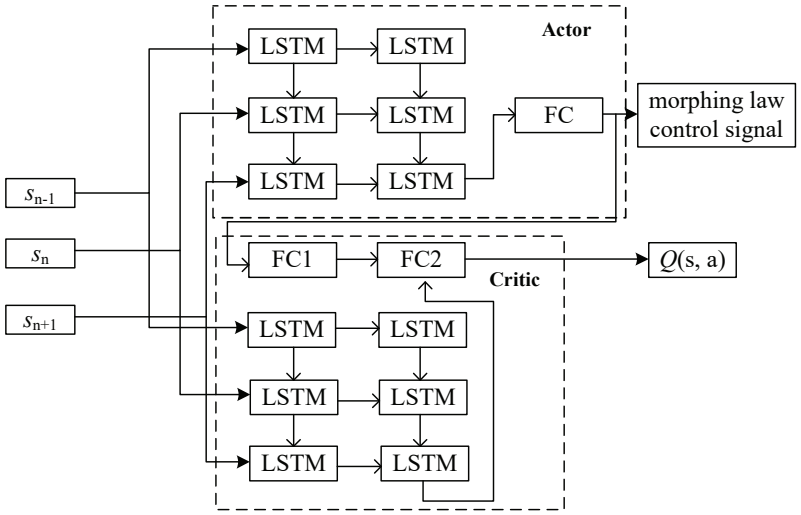



**Figure 5.** LFramework of the IDDPG network.

The Actor network is the policy network, established through a neural network to map from the current state to the next action (Figure 6). In the IDDPG algorithm, the input to the Actor network consists of the aircraft's state variables and altitude error, forming a seven-dimensional array $s_t = [x, h, V, \gamma, \alpha, q, \Delta h]$. This network consists of two hidden layers, with the first layer being an LSTM network housing 128 hidden neurons to effectively process environmental information, enhancing real-time performance [27]. The second layer employs fully connected architecture with 256 hidden neurons. Additionally, the rectified linear unit (Relu) function is chosen as the activation function between each hidden layer to reduce the probability of gradient vanishing issues [28]. Data from the first two layers flow directly into the subsequent layer through the activation function, and the final layer connects to the two-dimensional action space. Considering that the actions (sweep angle deformation rate) are constrained, the Tanh activation function is chosen for the final layer to limit the output within a specified range.

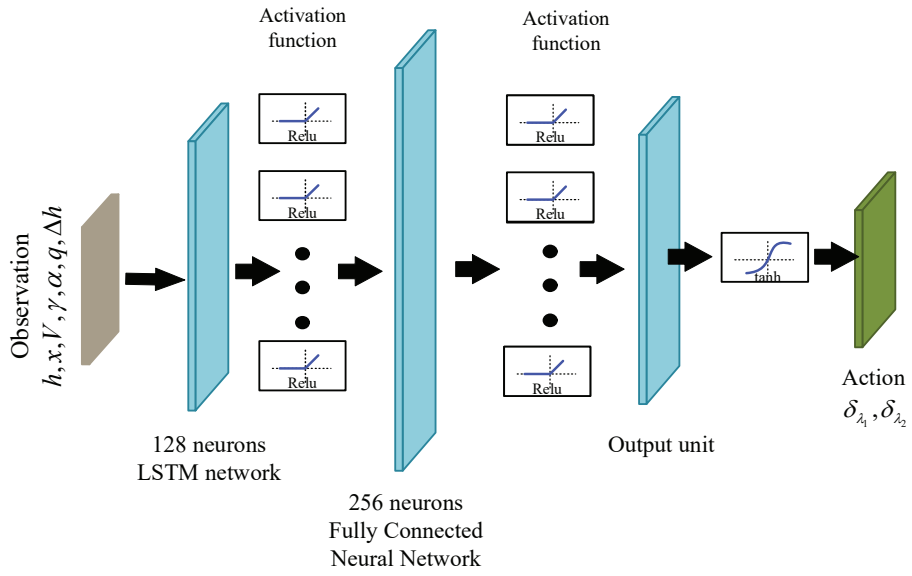



**Figure 6.** Actor network structure diagram.

In the IDDPG algorithm, the Critic network establishes a mapping from actions and states to the $Q$ function through a neural network, with the structure designed as depicted in Figure 7. The input layer of the Critic network is composed of state variables and the output actions from the Actor network, forming a nine-dimensional array. The output

represents the $Q$-function, denoted as $Q(s, a)$. There are two fully connected hidden layers, with the first and second layers having 128 and 200 hidden neurons, respectively. Unlike the Actor network, the final layer is directly connected to a one-dimensional output layer, which is further used to iteratively update the Bellman equation in reinforcement learning to maximize the output $Q$ value.

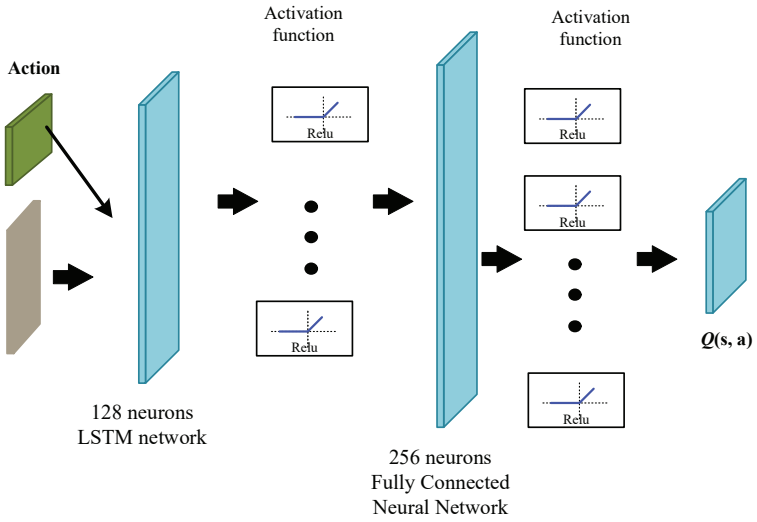



**Figure 7.** Critic network structure diagram.

#### 4.3.2. IDDPG Network Parameter Settings

In the IDDPG algorithm, apart from the Actor–Critic network structure, another crucial factor influencing algorithm performance is the setting of hyperparameters. This includes the total number of steps, total number of episodes, Actor learning rate, Critic learning rate, noise variance, and discount factor, among others. Based on empirical findings in the references and extensive trial-and-error simulations, the final values and definitions of all hyperparameters are presented in Table 1.

**Table 1.** The hyperparameter of IDDPG algorithm.

| Hyperparameter | Parameter Value |
|---|---|
| Target network update rate | 0.001 |
| Actor network learning rate | 0.0001 |
| Critic network learning rate | 0.001 |
| Experience replay pool capacity | $10^6$ |
| Number of small batch samples | 256 |
| Discount factor | 0.991 |
| Maximum number of episodes | 80,000 |
| Maximum steps per episode | 800 |
| Sampling time (s) | 0.02 |

It should be noted that the above parameter settings were obtained through extensive trial and error.

Here, an episode is defined as the process in which the agent executes a particular policy in the environment from start to finish. The IDDPG algorithm defaults to a maximum of 55,000 episodes, with the first 20,000 episodes constituting an offline pre-training process using expert data to avoid inefficient random exploration in the early stages of the IDDPG algorithm. Due to a sampling time of 0.02 s, the maximum number of steps per episode is set to 800, ensuring that the total time for each episode does not exceed 16 s. The Actor learning rate and Critic learning rate define the step size for gradient updates during network parameter optimization. After multiple trials, the Actor and Critic learning rates are determined to be 0.0001 and 0.001, respectively. The buffer size for storing past

experiences is set to 1,000,000, and the batch size of samples used during each gradient update is determined to be 256.

Another innovation of the IDDPG algorithm is the introduction of noise to increase sample diversity. In this study, OU noise with a variance of 0.15 is employed. Finally, considering that the initial actions may not necessarily lead to the expected state, the weights for future rewards need to decrease gradually. Therefore, a discount factor is introduced to calculate the expected future returns during the learning process and adjust the impact of future rewards, typically chosen as 0.991.

## 5. Simulation and Analysis

### *5.1. IDDPG Algorithm Network Training*

To validate the proposed IDDPG algorithm framework, simulations are conducted using the reinforcement learning toolbox in Matlab. Initially, without incorporating the kinematic incremental model, the impact of integrating LSTM networks into the DDPG algorithm will be assessed. The learning environments for both DDPG and LSTM–DDPG are constructed using the nonlinear expression (4) of the FWVS aircraft model, along with the state and action spaces, and reward function proposed in Section 3.2. However, the non-sparse reward state is replaced with the following form:

$$R'_m = w_4 r_h + w_5 r_V + w_6 r_q, \tag{24}$$

where $r_h = e^{-5\times10^{-4}\sqrt{(h-h_{\text{target}})}}$ represents altitude reward, $r_V = e^{-2\times10^{-3}\sqrt{(V-V_{\text{target}})}}$ is flight speed reward, and $r_q = e^{-5\sqrt{(q-q_{\text{target}})}}$ denotes pitch angle rate reward. The weights $w_i$ assigned to each reward are designed as $[0.5, 0.4, 0.1]$. When both velocity and altitude reach the target endpoint, the corresponding rewards reach their maximum values, and the weighted state reward reaches 1. Table 2 displays the reward function parameters determined after multiple simulation trials.

**Table 2.** Reward function parameters.

| Reward Function Parameters | Parameter Value |
|---|---|
| $h_{\text{target}}$ | 232 m |
| $V_{\text{target}}$ | 32 m/s |
| $q_{\text{target}}$ | 0 rad/s |
| $\delta_h$ | 1 m |
| $\delta_V$ | 0.5 m/s |
| $\delta_\gamma$ | 0.1 rad |
| $\delta_q$ | 0.1 rad/s |
| $h_{\max}, h_{\min}$ | 235 m, 198 m |
| $V_{\max}, V_{\min}$ | 33 m/s, 20 m/s |
| $\alpha_{\max}$ | 1 rad/s |
| $q_{\max}$ | $\pi/4$ rad |

The network training performance of the LSTM–DDPG algorithm is compared with the traditional DDPG algorithm [29], and the results are illustrated in Figure 8. It is evident that the LSTM–DDPG algorithm converges more rapidly to a higher average reward and exhibits greater stability. The incorporation of LSTM networks enables the DDPG algorithm to better discern valuable data, leading to optimal results and avoiding unnecessary exploration during the climb phase of the FWVS aircraft. This highlights the excellent performance of the LSTM network.

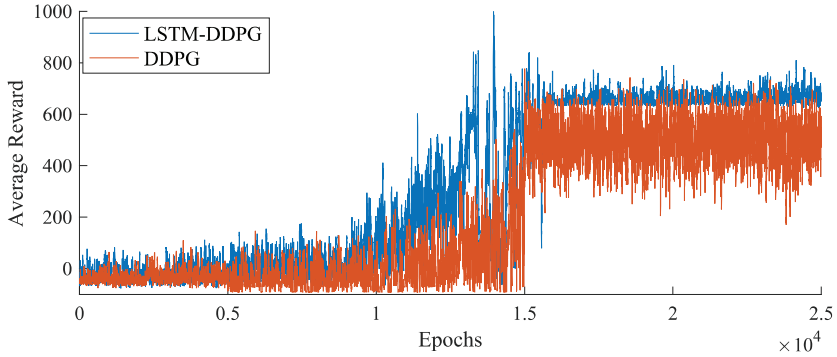



**Figure 8.** Comparison of average rewards for LSTM–DDPG and DDPG algorithm training.

Subsequently, the IDDPG algorithm, generated based on the kinematic increment model incorporating a reference trajectory, is employed to validate whether the inclusion of trajectory constraints accelerates the convergence speed of the network. In this simulation, the reinforcement learning environment utilizes the incremental model (7) of the morphing aircraft, with an action space $a = [\delta_{\lambda_1}, \delta_{\lambda_2}] \in [-0.1, 0.1]$ and hyperparameters referenced in Table 2. Furthermore, in comparison to LSTM–DDPG, the IDDPG algorithm incorporates the state reward function $R_m$, as depicted in Equation (15), within the reward function module. The simulation results are illustrated in Figure 9. Table 3 presents the average single-training time, total training time, and average reward for three algorithms: DDPG [29], LSTM–DDPG, and IDDPG.

**Table 3.** Algorithm comparison.

| Algorithms | DDPG [29] | LSTM–DDPG | IDDPG |
|---|---|---|---|
| Single-training time | 0.31 s | 0.35 s | 0.35 s |
| Total training time | 37.5 h | 30.8 h | 18.2 h |
| Average reward | 531 | 787 | 791 |

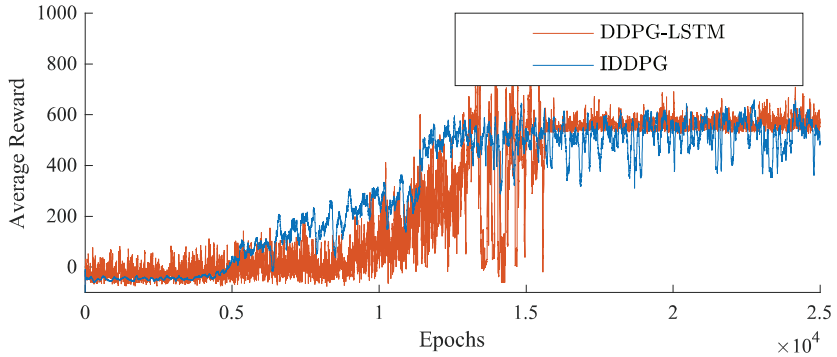



**Figure 9.** Algorithm training average reward comparison chart.

The simulation platform of each algorithm in Table 3 is the Mac system M2 chip, and the Matlab version is 2023b.

Based on the simulated training process and Table 3, it can be observed that IDDPG significantly reduces training time compared to DDPG, while achieving substantial improvements in training effectiveness and reward values. Additionally, as depicted in Figure 9, the LSTM–DDPG algorithm, without environmental modifications, achieves successful episodes around 8000, with the average reward converging after approximately 15,000 episodes. In contrast, the IDDPG algorithm, after redesigning the action space, achieves successful episodes before 5000 and starts converging to the maximum average reward just beyond 10,000 episodes. This indicates that the incremental model and action space derived from the reference trajectory can reduce the exploration space for actions, thereby accelerating the convergence rate (Table 3).

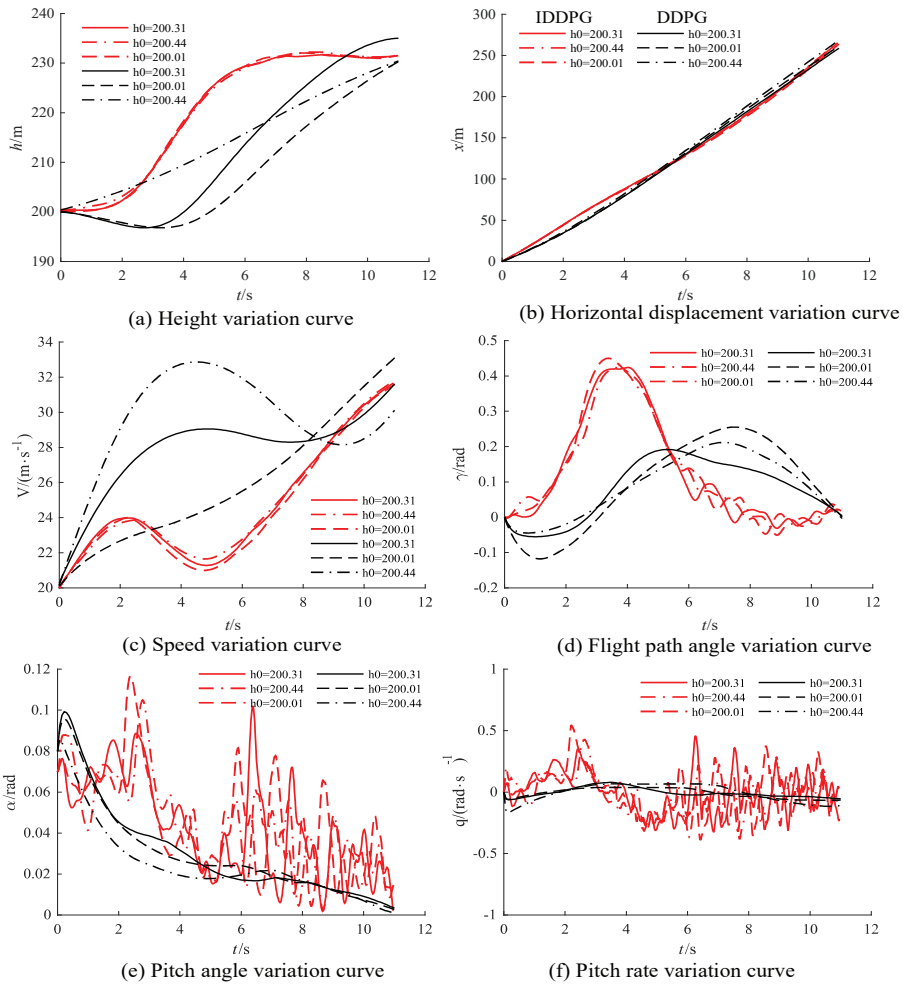
### *5.2. Flight Control Simulation Based on IDDPG Algorithm*

Upon completion of the training for the FWVS aircraft control strategy, simulation testing is conducted. The testing set is configured for 1000 episodes to assess the deformation effectiveness and success rate of accomplishing the tasks for the FWVS aircraft. The initial state is set to $s_0 = [0\ \text{m}, 200\ \text{m}, 20\ \text{m/s}, 0\ \text{rad}, 0.0799\ \text{rad}, 0\ \text{rad/s}]$, and the target states, represented by $h_f = 232$ m, $V_f = 32$ m/s, and $q_f = 0$ rad/s are defined. The objective is to complete the process of accelerating climb within a total time not exceeding 16 s. To evaluate the algorithm's adaptability to different initial environments, random initial heights are selected:

$$h_0 = 200 + rand(0.5), \tag{25}$$

where $rand(\zeta)$ represents a random number within the range of $\zeta$. In this simulation, the uncertain parameter $\varepsilon$ in Equation (8) is set to 0.1, indicating a 10% deviation in the lift coefficient under the current conditions. To validate the proposed algorithm's effectiveness compared to the traditional DDPG method [29], in this example, we trained and simulated both the proposed IDDPG and the DDPG methods separately.

After validation, employing the agent trained through IDDPG as the controller for the aircraft resulted in a success rate of 91.32%, whereas DDPG achieved a success rate of 78.26%. For different initial altitudes, Figure 10 illustrate the state variations for the FWVS aircraft under the training agents of IDDPG and DDPG.



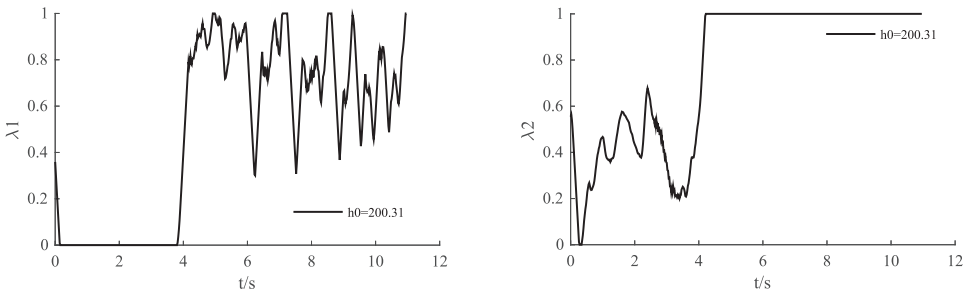

(a) Height variation curve
(b) Horizontal displacement variation curve
(c) Speed variation curve
(d) Flight path angle variation curve
(e) Pitch angle variation curve
(f) Pitch rate variation curve

**Figure 10.** State variable variation curves under different initial states (the red curve is IDDPG, the black curve is DDPG).

It should be noted that, once the learning network is trained, it can rapidly output action commands based on the current state of FWVS. The time required for the trained learning network to output action commands based on the current state of FWVS is on the order of $10^{-2}$ seconds (this is the result of running on the Mac M2 chip, Matlab2023b).

From Figure 10, it can be observed that, under the DDPG−trained agent, the FWVS aircraft (all black curves) deviates from the target point under different initial altitudes and lift coefficient uncertainties. In contrast, the FWVS aircraft under the influence of the IDDPG−trained agent (all red curves) reaches the target point despite slight oscillations compared to the corresponding DDPG curves. Furthermore, all state variables throughout the entire flight process satisfy the imposed constraints, successfully accomplishing the task of ascending to level flight.

Figure 11 illustrates the control variations of IDDPG for completing the flight task. It is observed that, during the climbing phase, the primary adjustments are made to the trailing−edge wing, while in the acceleration phase, the leading−edge wing plays a predominant role after reaching a certain altitude. However, the results indicate noticeable oscillations in control inputs, suggesting a potential issue with a large action space. Figure 12 visually presents the aircraft shape during the flight process. In the climbing phase, the trailing−edge sweep angle gradually increases, generating a pitching moment to increase lift. After reaching the target altitude, the trailing−edge sweep angle reaches its maximum value. At this point, the leading edge sweep angle is employed to control the pitch−down moment, maintaining a level flight state.




(a) deformation curve of leading-edge sweep angle (b) deformation curve of trailing-edge sweep angle

**Figure 11.** Action variation curve.

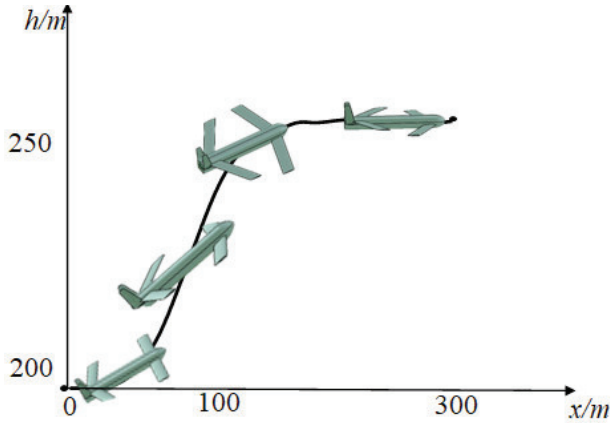
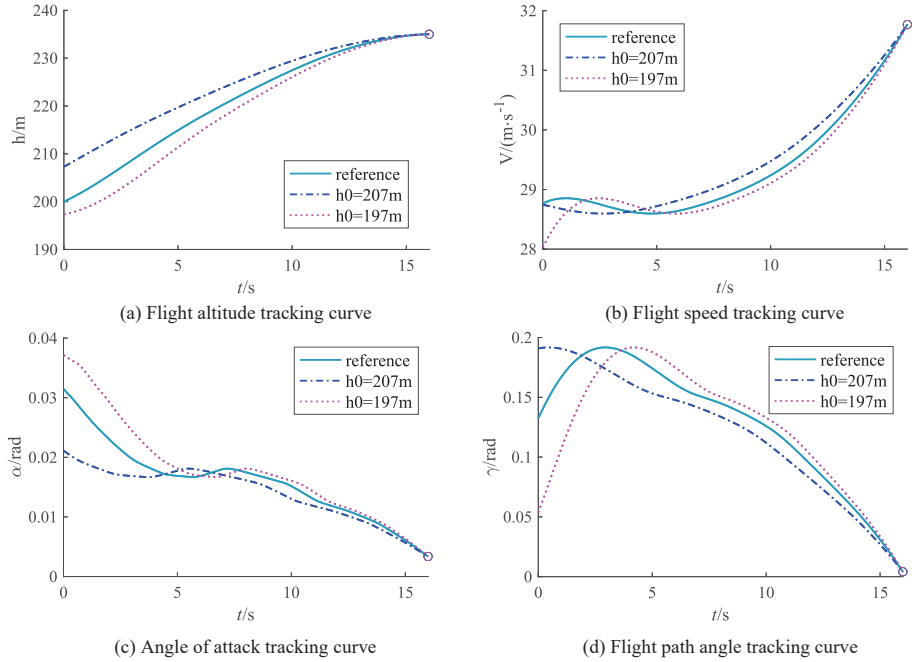



**Figure 12.** Outline diagram of the aircraft during flight.

Furthermore, employing the trained network, simulation experiments are conducted for trajectory tracking control and obstacle avoidance control. A test set of 1000 episodes is utilized to evaluate the effectiveness of tracking a reference trajectory. The various initial values, distributed on either side of the reference trajectory, are chosen to assess the tracking performance. Additionally, an uncertainty coefficient $\varepsilon$ for lift is set to 0.1. The reference trajectory is depicted by the solid blue line in Figure 13.



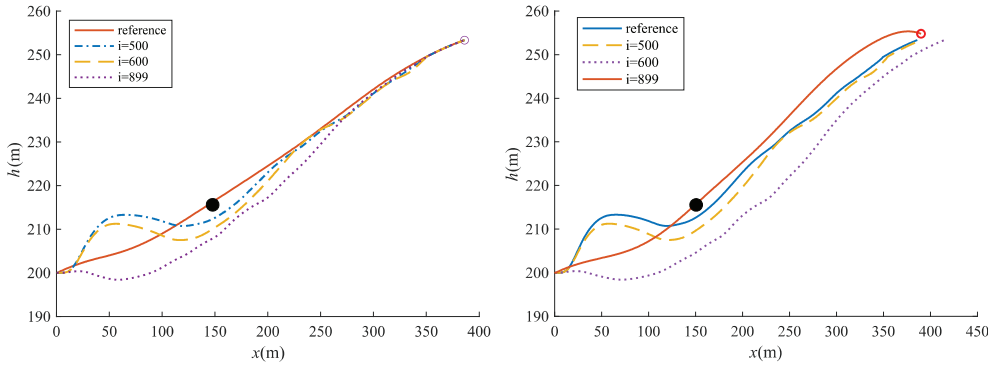

(a) Flight altitude tracking curve (b) Flight speed tracking curve

(c) Angle of attack tracking curve (d) Flight path angle tracking curve

**Figure 13.** Tracking curves of reinforcement learning controllers under different initial values.

As depicted in Figure 13, post−training with the IDDPG algorithm, the FWVS aircraft demonstrates rapid and accurate trajectory tracking to accomplish the task of accelerated climb, even in the presence of initial state errors and lift coefficient uncertainty. This validates the robustness of the IDDPG strategy to uncertainties in the deformation process.

Moreover, due to the unique memory units of the LSTM network, the UAV can better record obstacle positions and consider information from previous time steps when selecting actions. To further validate the information processing capabilities of the IDDPG network, obstacles are introduced into the flight environment to create different scenarios. A comparison is then made with the traditional DDPG method to assess its performance against the proposed approach. The initial state is set to $s_0 = [0\ \text{m}, 200\ \text{m}, 20\ \text{m/s}, 0\ \text{rad}, 0.0799\ \text{rad}, 0\ \text{rad/s}]$, with the reference trajectory represented by the solid orange line in Figure 14. The obstacle is positioned at $x = 150$ m, $h = 215$ m, and the circular obstacle has a radius of 1 m, as depicted by the black circle in Figure 14. Trajectory test for obstacle avoidance is conducted with the IDDPG network, achieving a success rate of at least 78.5% after numerous tests, whereas DDPG achieved only 48% success in a comparable scenario. Figure 14 illustrates the flight trajectories for obstacle avoidance during the 500th, 600th, and 899th instances.




(a) IDDPG (b) DDPG

**Figure 14.** Tracking path of morphing aircraft with different iterations (The black dot is the obstacle added during algorithm training).

From Figure 14a, it can be observed that the FWVS aircraft trained with the IDDPG algorithm successfully resumes tracking the reference trajectory after avoiding obstacles. However, under the DDPG−trained agent (Figure 14b), the FWVS aircraft deviates from the target trajectory while avoiding obstacles and struggles to reach the target point. Here, "*i*" denotes the number of successful episodes.

## 6. Conclusions

This paper focuses on the trajectory tracking control of a FWVS aircraft using reinforcement learning, proposing the IDDPG algorithm. This algorithm not only utilizes LOS for target point tracking, enhancing the algorithm's realization rate in target point tracking, but also integrates LSTM units to improve the algorithm's adaptability to environmental changes during flight. Subsequently, through a significant amount of training, it is concluded that the IDDPG algorithm has a faster convergence speed than the DDPG algorithm, and it also achieves a good tracking control effect for the FWVS with uncertainties. Lastly, to further validate the adaptability of this controller to environmental disturbances, simulations are conducted by introducing environmental perturbations. The results indicate that the FWVS aircraft can successfully navigate around obstacles and retrace the reference trajectory.

**Author Contributions:** Conceptualization, K.L. and R.C.; methodology, K.L. and R.C.; software, R.C.; validation, K.L. and R.C.; formal analysis, K.L. and R.C.; investigation, K.L. and R.C.; writing—original draft preparation, R.C.; writing—review and editing, K.L. and R.C. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the Natural Science Foundation of Jiangsu Province of China (No. BK20230560), and National Natural Science Foundation of China (grant number No. 62303400, 62373101, 62073075).

**Institutional Review Board Statement:** Not applicable.

**Data Availability Statement:** Data are contained within the article.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## Abbreviations

The following abbreviations are used in this manuscript:

| | |
|---|---|
| FWVS | Four Wing Variable Sweep Aircraft |
| LOS | Line of Sight |
| LSTM | Long Short Term Memory |
| DDPG | Deep Deterministic Policy Gradient |
| IDDPG | Incremental Deep Deterministic Policy Gradient |

## References

1. Christina, H.; Lawren, L.G.; Christian, R.B.; Douglas, F.H.; James, J.J.; Daniel, J.I. A review of avian-inspired morphing for UAV flight control. *Prog. Aerosp. Sci.* **2022**, *132*, 100825.
2. Jo, B.W.; Majid, T. Enhanced range and endurance evaluation of a camber morphing wing aircraft. *Biomimetics* **2023**, *8*, 34. [CrossRef]
3. Ajaj, R.; .Jankee, G.K. The transformer aircraft: A multisession unmanned aerial vehicle capable of symmetric and asymmetric span morphing. *Aerosp. Sci. Technol.* **2018**, *76*, 512–522. [CrossRef]
4. Afonso, F.; Vale, J.; Lau, F.; Suleman, A. Performance based multidisciplinary design optimization of morphing aircraft. *Aerosp. Sci. Technol.* **2017**, *67*, 1–12. [CrossRef]
5. Moens, F. Augmented aircraft performance with the use of morphing technology for a turboprop regional aircraft wing. *Biomimetics* **2019**, *4*, 64. [CrossRef] [PubMed]
6. Suraj, B.; Roberto, S.; Alessandro, G. Advances in intelligent and autonomous navigation systems for small UAS. *Prog. Aerosp. Sci.* **2020**, *115*, 100617.
7. Wang, Y.; Wang, J.; Kang, S.; Yu, J. Target-following control of a biomimetic autonomous system based on predictive reinforcement learning. *Biomimetics* **2024**, *9*, 33. [CrossRef] [PubMed]

8. Li, Y.; Chen, Z.; Wu, C.; Mao, H.; Sun, P. A hierarchical framework for quadruped robots gait planning based on DDPG. *Biomimetics* **2023**, *8*, 382. [CrossRef]
9. Zhao, J.; Liu, H.; Sun, J.; Wu, K.; Cai, Z.; Ma, Y.; Wang, Y. Deep reinforcement learning-based end-to-end control for UAV dynamic target tracking. *Biomimetics* **2022**, *7*, 197. [CrossRef]
10. Lecun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef]
11. Littman, M.L. Reinforcement learning improves behaviour from evaluative feedback. *Nature* **2015**, *521*, 445–451. [CrossRef] [PubMed]
12. Mnih, V.; Kavukcuoglu, K.; Silver, D. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533. [CrossRef] [PubMed]
13. Gu, Y.; Cheng, Y.; Yu, K.; Wang, X. Anti-martingale proximal policy optimization. *IEEE Trans. Cybern.* **2023**, *53*, 6421–6432. [CrossRef] [PubMed]
14. Gong, L.; Wang, Q.; Hu, C.; Liu, C. Switching control of morphing aircraft based on Q-learning. *Chin. J. Aeronaut.* **2020**, *33*, 672–687. [CrossRef]
15. Lu, K.; Fu, Q.; Cao, R.; Peng, J.; Wang, Q. Asymmetric airfoil morphing via deep reinforcement learning. *Biomimetics* **2022**, *7*, 188. [CrossRef] [PubMed]
16. Li, R.; Wang, Q.; Liu, Y.; Dong, C. Morphing Strategy Design for UAV based on Prioritized Sweeping Reinforcement Learning. In Proceedings of the IECON 2020 the 46th Annual Conference of the IEEE Industrial Electronics Society, Singapore, 18–21 October 2020; pp. 2786–2791.
17. Lekkas, A.M.; Fossen, T.I. A quaternion-based LOS guidance scheme for path following of AUVs. In Proceedings of the IFAC Conference on Control Applications in Marine Systems, Osaka, Japan, 17–20 September 2014; pp. 245–250.
18. Israel, L.; Nesrin, S.K. A review of uncertainty in flight vehicle structural damage monitoring, diagnosis and control: Challenges and opportunities. *Prog. Aerosp. Sci.* **2010**, *46*, 247–273.
19. Coto-Jiménez, M. Improving post-filtering of artificial speech using pre-trained LSTM neural networks. *Biomimetics* **2019**, *4*, 39. [CrossRef] [PubMed]
20. Kuo, C.T.; Lin, J.J.; Jen, K.K.; Hsu, W.L.; Wang, F.C.; Tsao, T.C.; Yen, J.Y. Human posture transition-time detection based upon inertial measurement unit and long short-term memory neural networks. *Biomimetics* **2023**, *8*, 471. [CrossRef]
21. Gao, L.; Jin, H.; Zhao, J.; Cai, H.; Zhu, Y. Flight Dynamics Modeling and Control of a Novel Catapult Launched Tandem-Wing Micro Aerial Vehicle With Variable Sweep. *IEEE Access* **2018**, *6*, 42294–42308. [CrossRef]
22. Jayaweera, S.K. *Markov Decision Processe*; Cognitive Radios, Wiley: Hoboken, NJ, USA, 2015; pp. 207–268.
23. Wang, C.; Wang, J.; Wang, J.; Zhang, X. Deep-Reinforcement-Learning-Based Autonomous UAV Navigation With Sparse Rewards. *IEEE Internet Things J.* **2018**, *7*, 6180–6190. [CrossRef]
24. Li, L.; Li, D.; Song, T.; Xu, X. Actor–Critic Learning Control With Regularization and Feature Selection in Policy Gradient Estimation. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *7*, 1217–1227. [CrossRef] [PubMed]
25. Kim, J.; Cha, S.H.; Ryu, M.; Jo, M. Pre-training framework for improving learning speed of reinforcement learning based autonomous vehicles. In Proceedings of the 2019 International Conference on Electronics, Information, and Communication (ICEIC), Auckland, New Zealand, 22–25 January 2019; pp. 1–2.
26. Bo, L.; Yang, Z.; Chen, D.; Liang, S.; Ma, H. Maneuvering target tracking of UAV based on MN-DDPG and transfer learning. *Def. Technol.* **2021**, *17*, 457–466.
27. Yu, Y.; Si, X.; Hu, C.; Zhang, J. A review of recurrent neural networks: LSTM cells and network architectures. *Neural Comput.* **2019**, *31*, 1235–1270. [CrossRef] [PubMed]
28. Chao, W.; Han, D.; Jie, X. Multi-rotor UAV autonomous tracking and obstacle avoidance based on improved DDPG. In Proceedings of the 2021 2nd International Conference on Artificial Intelligence and Computer Engineering (ICAICE), Hangzhou, China, 5–7 November 2021; pp. 261–267.
29. Hao, G.; Zhuang, F.; Xin, F.; Gong, Z.; Chen, P.; Wang, D.; Wang, W.B.; Si, Y. A deep deterministic policy gradient approach for vehicle speed tracking control with a robotic driver. *IEEE Trans. Autom. Sci. Eng.* **2022**, *19*, 2514–2525. [CrossRef]