applied sciences

MDPI

Article

Error Dynamics Based Dual Heuristic Dynamic Programming for Self-Learning Flight Control

Xu Huang [1,2], Yuan Zhang [1,2], Jiarun Liu [1,2,*], Honghao Zhong [1], Zhaolei Wang [1,2] and Yue Peng [1]

[1] Beijing Aerospace Automatic Control Institute, Beijing 100854, China
[2] National Key Laboratory of Science and Technology on Aerospace Intelligence Control, Beijing 100854, China
* Correspondence: jiarunliu@163.com

Abstract: A data-driven nonlinear control approach, called error dynamics-based dual heuristic dynamic programming (ED-DHP), is proposed for air vehicle attitude control. To solve the optimal tracking control problem, the augmented system is defined by the derived error dynamics and reference trajectory so that the actor neural network can learn the feedforward and feedback control terms at the same time. During the online self-learning process, the actor neural network learns the control policy by minimizing the augmented system's value function. The input dynamics identified by the recursive least square (RLS) and output of the critic neural network are used to update the actor neural network. In addition, the total uncertainty term of the error dynamics is also identified by RLS, which can compensate for the uncertainty caused by inaccurate modeling, parameter perturbation, and so on. The outputs of ED-DHP include the rough trim surface, feedforward and feedback terms from the actor neural network, and the compensation. Based on this control scheme, the complete knowledge of system dynamics and the reference trajectory dynamics are not needed, and offline learning is unnecessary. To verify the self-learning ability of ED-DHP, two numerical experiments are carried out based on the established morphing air vehicle model. One is sinusoidal signal tracking at a fixed operating point, and the other is guidance command tracking with a morphing process at variable operating points. The simulation results demonstrate the good performance of ED-DHP for online self-learning attitude control and validate the robustness of the proposed scheme

Keywords: error dynamics; dual heuristic dynamic programming; recursive least square method; online self-learning; attitude control

check for updates

Citation: Huang, X.; Zhang, Y.; Liu, J.; Zhong, H.; Wang, Z.; Peng, Y. Error Dynamics Based Dual Heuristic Dynamic Programming for Self-Learning Flight Control. *Appl. Sci.* **2023**, *13*, 586. https://doi.org/10.3390/app13010586

Academic Editors: Mingyang Guo, Jiang Long, Qingdong Li and Zun Liu

Received: 28 November 2022
Revised: 24 December 2022
Accepted: 25 December 2022
Published: 31 December 2022

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The flight control system plays an essential role in guaranteeing the flight quality of air vehicles. However, the nonlinearity and model uncertainty of the vehicle dynamics become more significant because of increasing flight mission complexity [1,2]. Besides, it is difficult to model complex vehicle systems, and great effort is required in model identification. For these reasons, the demand for adaptive nonlinear attitude control methods with modeling inaccuracies is rising.

Reinforcement learning (RL) is a machine learning method. It possesses a self-learning capability by interacting with the environment. In recent years, RL has been applied to solve a large number of control problems in many fields, such as robotics [3], nuclear fusion [4], intelligent transportation [5], and smart grid [6]. RL is consistently used as a parameter adjuster [7], uncertainty compensator [8], and end-to-end controller [9,10] in the flight control system as well.

Adaptive/approximate dynamic programming (ADP) is one of the RL methods, bridging the gap between traditional optimal and adaptive control [11]. Different frameworks with ADP have been proposed to solve adaptive flight control problems. In some frameworks, linear/nonlinear control methods such as PID [12], sliding mode control (SMC) [13,14],

nonlinear dynamic inverse (NDI) [15], and backstepping (BS) [16] are combined with ADP to improve the control quality. Moreover, ADP provides supplementary adjustments to expedite control performance. However, the linear/nonlinear controllers are dominant parts of the control system, so the system performance can be bad or even unstable without these controllers. Moreover, they are always designed with known dynamic models and may not work when high nonlinearity and large model uncertainty exist. Some other frameworks where the ADP-based controllers are dominant have been proposed. For example, Han et al. [17] proposed an online policy iteration (PI) method for adaptive control during the vehicle reentry phase. But the convergence condition of PI is that the initial control policy must make the system stable, so the neural networks in this method should be trained offline. In addition, an ADP method called adaptive critic designs (ACDs) [18,19] have been extensively studied. However, extra structures, such as the global system model, are also vital to approximate the system dynamics in ACDs. For this reason, an offline learning phase is additionally needed in system model identification.

To reduce the dependency on prior knowledge of the system model and offline phase, one framework called incremental model-based dual heuristic dynamic programming (iDHP) was proposed by Zhou et al. [20,21]. The critic neural network (NN) in iDHP approximates the costate function, which is a state dimension vector gradient and can bring more information than the value function. In iDHP, the incremental model is identified by the RLS method, and the incremental model is provided for the neural network updating. It should be noted that the RLS works with high-frequency sample data and relatively slow-varying system assumption. Given the relatively high training failure rate in the practical application of iDHP, Li et al. [22] proposed a switching control system consisting of an iDHP-based controller and a PID controller, which was introduced as a safety control policy. Furthermore, to improve the iDHP's adaptability to a wide range of dynamic variations, Liu et al. [23] proposed an adaptive learning rate-based iDHP method where a cost function was designed to evaluate the controller stability based on the wavelet analysis. Lee et al. [24] presented two iDHP controller designs with and without the cascaded actor structure to realize the air vehicle altitude control. Heyer et al. [25] optimized the iDHP with an additional network structure to enhance the learning stability and applied the method to the high-dimensional aerospace models.

To further optimize the identification ability of RLS in iDHP and make full use of the iDHP's advantage in learning efficiency, a novel approach called error dynamics-based dual heuristic dynamic programming is proposed for online nonlinear flight control. One merit of ED-DHP is that complete knowledge of the vehicle dynamics and reference trajectory dynamics is unnecessary. Unlike iDHP, RLS is introduced to identify error dynamics containing the input distribution matrix and the total disturbance term. What's more, the state of ED-DHP is augmented by tracking error and reference trajectory, so the actor NN can learn the feedforward and feedback control terms. The contributions of this paper are summarized below:

1. An ED-DHP controller for the fast loop is designed to control the morphing air vehicle attitude system with different work points without offline learning;
2. The error dynamics of the fast loop are derived. Based on these, the control effectiveness matrix and the total disturbance term are identifiable by RLS. This enhances learning efficiency effectively. The ADP system is augmented by tracking error and reference trajectory, so the feedforward and feedback control terms can be learned by the actor NN, on the basis that the complete knowledge of system dynamics and the reference trajectory dynamics are unknown;
3. The outputs of ED-DHP include the rough trim surface, feedforward and feedback terms, and compensation. The robustness of the method can be enhanced.

The remainder of this paper is organized as follows: Section 2 describes the attitude tracking problem of the air vehicle; Section 3 presents the control structure of the system, including design and analysis of the ED-DHP framework; In Section 4, some simulation re-

sults are given to demonstrate the effectiveness of ED-DHP; Section 5 states the conclusion of this article.

2. Mathematical Model of Air Vehicle

The kinematical and dynamical equations of the air vehicle are shown below.

$$\dot{\Omega} = g_s\omega + \frac{1}{mv}F_s \tag{1}$$

$$\dot{\omega} = -\left(I^{\sigma(t)}\right)^{-1}\mu I^{\sigma(t)}\omega + \left(I^{\sigma(t)}\right)^{-1}M \tag{2}$$

where $\Omega = [\alpha, \beta, \gamma_V]^{\mathrm{T}}$ and $\omega = \left[\omega_x,\ \omega_y,\ \omega_z\right]^{\mathrm{T}}$ are the system state vectors. α, β, and γ_V denote the attack angle, sideslip angle, and bank angle separately; ω is the angular rate vector with roll rate ω_x, yaw rate ω_y, and pitch rate ω_z. m and v are the mass and the velocity of the vehicle.

For the slow loop subsystem (1), the concrete expressions of g_s and F_s are as follows:

$$g_s = \begin{bmatrix} -\cos\alpha\tan\beta & \sin\alpha\tan\beta & 1 \\ \sin\alpha & \cos\alpha & 0 \\ \cos\alpha\sec\beta & -\sin\alpha\sec\beta & 0 \end{bmatrix}, \quad F_s = \begin{bmatrix} -F_y/\cos\beta \\ F_z \\ (F_y\cos\gamma_V - F_z\sin\gamma_V)\tan\theta + F_y\tan\beta \end{bmatrix}$$

For the fast loop subsystem (2), the concrete expressions of $I^{\sigma(t)}$, μ, and M are as follows:

$$I^{\sigma(t)} = \begin{bmatrix} I_x^{\sigma(t)} & -I_{xy}^{\sigma(t)} & -I_{xz}^{\sigma(t)} \\ -I_{xy}^{\sigma(t)} & I_y^{\sigma(t)} & -I_{yz}^{\sigma(t)} \\ -I_{zx}^{\sigma(t)} & -I_{zy}^{\sigma(t)} & I_z^{\sigma(t)} \end{bmatrix}, \ \mu = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix},$$

$$M = \begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} = \begin{bmatrix} C_{mx}^{\sigma(t)}(Ma,\alpha,\beta,\delta)qSl \\ C_{my}^{\sigma(t)}(Ma,\alpha,\beta,\delta)qSl \\ C_{mz}^{\sigma(t)}(Ma,\alpha,\beta,\delta)qSl \end{bmatrix} = \begin{bmatrix} \left(C_{mx_basic}^{\sigma(t)}(Ma,\alpha,\beta) + C_{mx_\delta}^{\sigma(t)}(Ma,\alpha,\beta,\delta)\right)qSl \\ \left(C_{my_basic}^{\sigma(t)}(Ma,\alpha,\beta) + C_{my_\delta}^{\sigma(t)}(Ma,\alpha,\beta,\delta)\right)qSl \\ \left(C_{mz_basic}^{\sigma(t)}(Ma,\alpha,\beta) + C_{mz_\delta}^{\sigma(t)}(Ma,\alpha,\beta,\delta)\right)qSl \end{bmatrix}$$

where $\sigma(t)$ describes different morphing configurations of the air vehicle; θ is the trajectory inclination angle; F_y is the normal force and F_z is the lateral force; $I^{\sigma(t)}$ is the moment of inertia matrix; $q = \rho v^2/2$ denotes the dynamic pressure; S and l are the reference area and reference aerodynamic chord, respectively. $C_{m*_basic}^{\sigma(t)}(Ma,\alpha,\beta)$ is the aerodynamic moment coefficient, and $C_{m*_\delta}^{\sigma(t)}(Ma,\alpha,\beta,\delta)$ is the control moment coefficient induced by the control surface; Ma is the Mach number and δ is the control surface vector.

Remark. *the research object is a morphing air vehicle, and the moment of inertia $\overline{I}^{\sigma(t)}$ and the aerodynamic coefficient $C_{m*}^{\sigma(t)}$ change with different configurations, which can be derived by linear interpolation.*

However, uncertainties always present for the nominal parameters during flight. Therefore, to ensure the robustness of the air vehicle's flight control, the uncertain parameters in systems (1) and (2) cannot be neglected, described as:

$$\begin{cases} I = \overline{I}^{\sigma(t)}(1+\Delta I) \\ m = \overline{m}(1+\Delta m) \\ \rho = \overline{\rho}(1+\Delta\rho) \\ C_{m*}^{\sigma(t)} = \overline{C}_{m*}^{\sigma(t)}(1+\Delta C_m) \end{cases} \tag{3}$$

where the expressions $(\overline{\bullet})$ are the nominal value of the key parameters, and the parametric uncertainties are assumed to be bounded. Uncertain parameters can affect the system

modeling and control effect. This uncertainty model will be applied in chapter 4 to verify the effectiveness and robustness of ED-DHP.

3. Online Tracking Controller Based on ED-DHP

3.1. Control Structure of the System

Based on the multiple-timescale features, the dynamics can be divided into the slow loop subsystem and the fast loop subsystem. So the controller for each system can be designed separately. As illustrated in Figure 1, the slow loop subsystem control law is designed by dynamic inversion with the tracking differentiator (TD) in this article to get the virtual control input of the slow loop subsystem. The ED-DHP is applied to angular rate control of the pitch, yaw, and roll rate, augmented with the slow outer loop. Rate control exhibits the lowest learning complexity because the angular rates and control surfaces are under direct dynamic relation. The control surfaces consist of the rough trim value, the output of actor NN, and the uncertainty compensator proposed in Section 3.2.

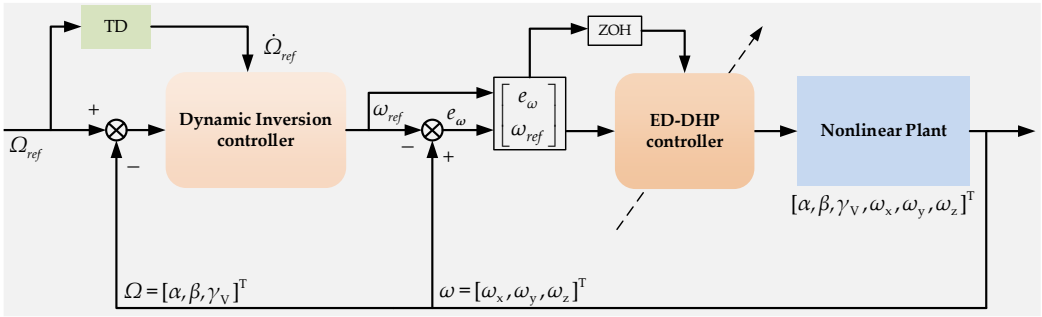


Figure 1. Schematic of flight controller with ED-DHP.

The virtual control vector of the slow loop is designed by dynamic inversion:

$$\omega_{ref} = g_s^{-1}\left(\begin{bmatrix} \dot{\alpha}_{ref} + k_\alpha(\alpha_{ref} - \alpha) \\ \dot{\beta}_{ref} + k_\beta(\beta_{ref} - \beta) \\ \dot{\gamma}_{Vref} + k_{\gamma_V}(\gamma_{Vref} - \gamma_V) \end{bmatrix} - f_s C_b^a \left(\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} + C_I^b \begin{bmatrix} g_x \\ g_y \\ g_z \end{bmatrix} \right)\right) \tag{4}$$

where $\omega_{ref} = [\omega_{x,ref} \ \ \omega_{y,ref} \ \ \omega_{z,ref}]^T$ is the reference trajectory vector of the fast loop subsystem. $[a_x \ \ a_y \ \ a_z]^T$ is the acceleration vector, which can be measured by an accelerometer. $[g_x \ \ g_y \ \ g_z]^T$ is the gravity vector. C_b^a and C_I^b are the coordinate transformation matrix calculated by attitude angles with pitch angle φ, yaw angle ψ, and roll angle γ, as well as attack angle α and sideslip angle β.

$$C_I^b = \begin{bmatrix} \cos\psi\cos\varphi & \cos\psi\sin\varphi & -\sin\psi \\ -\cos\gamma\sin\varphi + \sin\gamma\sin\psi\cos\varphi & \cos\gamma\cos\varphi + \sin\gamma\sin\psi\sin\varphi & \sin\gamma\cos\psi \\ \sin\gamma\sin\varphi + \cos\gamma\sin\psi\cos\varphi & -\sin\gamma\cos\varphi + \cos\gamma\sin\psi\sin\varphi & \cos\gamma\cos\psi \end{bmatrix}$$

$$C_b^a = \begin{bmatrix} \cos\alpha\cos\beta & \sin\beta & \sin\alpha\cos\beta \\ -\cos\alpha\sin\beta & \cos\beta & -\sin\alpha\sin\beta \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix}$$

And the matrix function f_s can be written as:

$$f_s = \frac{1}{V}\begin{bmatrix} 0 & -\sec\beta & 0 \\ 0 & 0 & 1 \\ 0 & \tan\beta + \cos\gamma_V\tan\theta & -\sin\gamma_V\tan\theta \end{bmatrix}$$

As we can see from the virtual control vector (4), the derivative of guidance commands is necessary to improve the dynamic performance of the system. To alleviate the "items of explosion" problem, the tracking differentiator was introduced [26], which is described as:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -r^2 \text{sign}(x_1 - \zeta)|x_1 - \zeta|^\tau - rx_2 \end{cases} \tag{5}$$

where ζ is the guidance command, x_2 is the differential value. r and τ are positive parameters to be designed.

3.2. The Fast Loop Error Dynamics of Attitude System and Model Identification

The dynamical equations of the fast loop can be rewritten as:

$$\dot{x} = f_1 + f_2 + gu + \Delta f \tag{6}$$

The symbols in Equation (6) are given as:

$$x = \omega,\ f_1 = -\left(\overline{I}^{\sigma(t)}\right)^{-1} R \overline{I}^{\sigma(t)} \omega,$$

$$f_2 = \left(\overline{I}^{\sigma(t)}\right)^{-1} \begin{bmatrix} \overline{C}^{\sigma(t)}_{mx_basic}(Ma, \alpha, \beta) qSl \\ \overline{C}^{\sigma(t)}_{my_basic}(Ma, \alpha, \beta) qSl \\ \overline{C}^{\sigma(t)}_{mz_basic}(Ma, \alpha, \beta) qSl \end{bmatrix},\ gu = \left(\overline{I}^{\sigma(t)}\right)^{-1} \begin{bmatrix} \overline{C}^{\sigma(t)}_{mx_\delta}(Ma, \alpha, \beta, \delta) qSl \\ \overline{C}^{\sigma(t)}_{my_\delta}(Ma, \alpha, \beta, \delta) qSl \\ \overline{C}^{\sigma(t)}_{mz_\delta}(Ma, \alpha, \beta, \delta) qSl \end{bmatrix}$$

The disturbance term Δf expressed parameter uncertainty and modeling error, which causes the inaccurate trim value and external disturbance.

The trim surface is set as $u_p \approx -g^{-1} f_2$, which can be roughly obtained by the newton iteration method offline, and the standard control input $u \in \mathbb{R}^m$ is given by:

$$u = u_p + u_a \tag{7}$$

where u_a is the output vector of actor NN in ED-DHP.

Substitute Equation (7) into Equation (6) and subtract the reference trajectory dynamics $\dot{x}_{ref}$, we can obtain:

$$\begin{aligned} \dot{e} = \dot{x} - \dot{x}_{ref} &= f_1 + f_2 + gu + \Delta f - \dot{x}_{ref} \\ &= f_1 + f_2 + g(u_p + u_a) + \Delta f - \dot{x}_{ref} \\ &= f_1(x) + gu_a + \Delta f - \dot{x}_{ref} \end{aligned} \tag{8}$$

And the problem can be written in a discrete form by Euler's method:

$$\begin{cases} e(t+1) \approx e(t) - Z(t) + \Delta t f_1(x(t)) + \Delta t g(t) u_a(t) + \Delta t \Delta f \\ x_{ref}(t+1) = \phi(x_{ref}(t)) \\ Z(t) = x_{ref}(t+1) - x_{ref}(t) \end{cases} \tag{9}$$

where f_1 is known. g, Δf and reference trajectory function ϕ are unrevealed, however.

The first equation in Equation (9) can be rewritten as follows:

$$Y(t) = \varsigma(t)^{\mathrm{T}} H(t) \tag{10}$$

where

$$Y(t) = e(t) - e(t-1) + Z(t-1) - \Delta t f_1(x(t-1)),$$

$$\varsigma(t) = \begin{bmatrix} \Delta t g(t-1) \\ \Delta t \Delta f \end{bmatrix} = \begin{bmatrix} \varsigma_1(t) \\ \varsigma_2(t) \end{bmatrix} \in \mathbb{R}^{(m+1)\times n},\ H(t) = \begin{bmatrix} u_a(t-1) \\ 1 \end{bmatrix} \in \mathbb{R}^{(m+1)\times 1}.$$

$Y(t)$ can be directly calculated from the measured vehicle attitude. $H(t)$ is the information matrix, which is only related to the previous output vector of the actor NN. $\varsigma(t)$ is the parameter matrix, and it can be identified in real-time by the RLS method:

$$\varepsilon(t) = Y(t) - \hat{Y}(t) = \varsigma(t)^{\mathrm{T}} H(t) - \hat{\varsigma}(t)^{T} H(t) \tag{11}$$

$$\hat{\varsigma}(t) = \hat{\varsigma}(t-1) + \frac{Cov(t-1)H(t)\varepsilon(t)^{\mathrm{T}}}{\gamma_{\mathrm{RLS}} + H(t)^{\mathrm{T}} Cov(t-1)H(t)} \tag{12}$$

$$Cov(t) = \frac{1}{\gamma_{\mathrm{RLS}}}\left(Cov(t-1) - \frac{Cov(t-1)H(t)H(t)^{\mathrm{T}}Cov(t-1)}{\gamma_{\mathrm{RLS}} + H(t)^{\mathrm{T}} Cov(t-1)H(t)}\right) \tag{13}$$

where $\varepsilon(t) \in \mathbb{R}^{n\times 1}$ is the prediction error and γ_{RLS} is the forgetting factor. $Cov(t) \in \mathbb{R}^{(m+1)\times(m+1)}$ is the estimation covariance matrix.

3.3. ED-DHP Framework

For the system dynamics (9), the perfect control input contains the feedforward input u_d^* and the feedback input u_e^* if the system dynamics is known [27].

$$\begin{cases} u_d^* = (\Delta t g)^{-1}(\phi(x_{ref}(t)) - \Delta t f_1(x_{ref}) - \Delta t \Delta f) \\ u_e^* = -\frac{\gamma}{2} R^{-1}(\Delta t g(t))^{\mathrm{T}} \frac{\partial V(e(t+1))}{\partial e(t+1)} \\ u_a^* = u_d^* + u_e^* \end{cases} \tag{14}$$

The symbols in Equation (14) are given as follows. Function J is known as the utility and is selected based on the quadratic energy function in this paper. V is defined as a performance measure function or value function in RL. It represents the sum of expected future reward from the current state. Matrices $Q \in \mathbb{R}^{n\times n}$ and $R \in \mathbb{R}^{m\times m}$ are positive definite. γ is a discount factor.

$$J(e(t), u_e(t)) = \frac{1}{2}\left[e(t)^{\mathrm{T}} Q e(t) + u_e(t)^{\mathrm{T}} R u_e(t)\right] \tag{15}$$

$$V(t) = \sum_{i=t}^{+\infty} \gamma^{(i-t)} J(e(i), u_e(i)) \tag{16}$$

For learning the control strategy of the attitude system without some prior knowledge, the augmented system is constructed in terms of $e(t)$ and $x_{ref}(t)$ [28,29] as:

$$X(t) = \begin{bmatrix} e(t) \\ x_{ref}(t) \end{bmatrix} \in \mathbb{R}^{2n} \tag{17}$$

$$\begin{bmatrix} e(t+1) \\ x_{ref}(t+1) \end{bmatrix} = \begin{bmatrix} e(t) - Z(t) + \Delta t f_1(x(t)) \\ \phi(x_{ref}(t)) \end{bmatrix} + \begin{bmatrix} \Delta t g(t) \\ 0 \end{bmatrix} u_a(t) \equiv F(X(t)) + G(t) u_a(t) \tag{18}$$

So we could have the following controller for the fast loop subsystem as follows: (Here, the second part u_a is the output vector of actor NN, and the third $u_b = -\hat{\varsigma}_1{}^{-1}\hat{\varsigma}_2$ is designed to compensate for the total uncertainty.)

$$\begin{aligned} u^* &= u_p + u_d^* + u_e^* - (\Delta t g)^{-1}(\Delta t \Delta f) \\ &\approx u_p + \underbrace{u_d + u_e}_{u_a:\text{ED-DHP}} + \underbrace{-\hat{\varsigma}_1{}^{-1}\hat{\varsigma}_2}_{u_b:\text{RLS}} \\ &= u_p + u_a + u_b \end{aligned} \tag{19}$$

For the augmented system (18), the cost function J and value function V can be rewritten as:

$$J_X(X(t), u_a(t)) = \frac{1}{2}\left[X(t)^{T} Q_X X(t) + u_a(t)^{T} R u_a(t)\right] \tag{20}$$

$$\begin{aligned} V_X(t) &= \sum_{i=t}^{+\infty} \gamma^{(i-t)} J_X(X(t), u_a(t)) \\ &= J_X(X(t), u_a(t)) + \gamma \sum_{i=t+1}^{+\infty} \gamma^{(i-t-1)} J_X(X(i), u_a(i)) \\ &= J_X(X(t), u_a(t)) + \gamma V_X(t+1) \end{aligned} \tag{21}$$

where

$$Q_X = \begin{bmatrix} Q & 0 \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{2n \times 2n} \tag{22}$$

Figure 2 shows the schematic of ED-DHP. The critic NN approximates the derivatives of the value function $V(X(t))$ concerning the augmented state $X(t)$, defined as $\lambda(X(t))$.

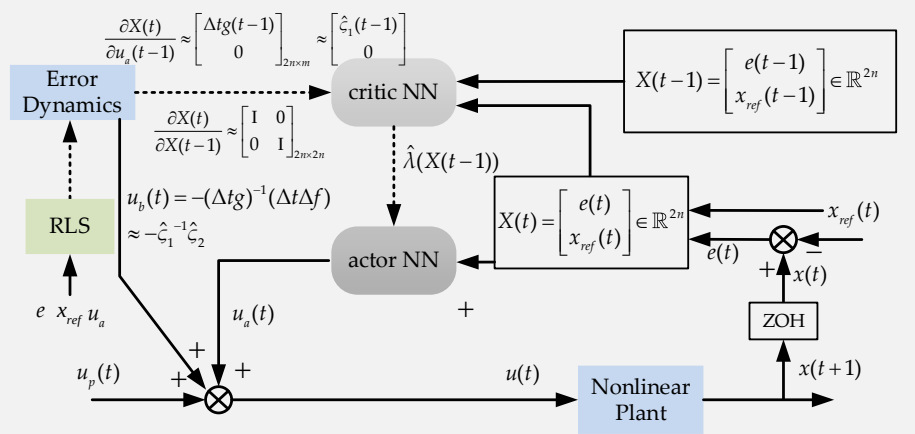


Figure 2. Schematic of ED-DHP.

As can be observed from this figure, the previous derivatives of the value function $\hat{\lambda}(X(t-1))$ and the augmented state $X(t-1)$ are required to be stored and used in updating the NNs, without predicted ones. The control policy estimated by the actor NN is updated by minimizing the value function. A two-layer neural network with one hidden layer containing 10 neurons is used as the critic NN and actor NN in this article, shown in Figure 3. The hidden-layer activation function is tanh. For the critic NN, $w_{\text{critic1},ij}$, and $w_{\text{critic2},jk}$ represent the weights from the ith input node to the jth hidden layer node and the jth hidden node to the kth output node, respectively. For the actor NN, $w_{\text{actor1},ij}$, and $w_{\text{actor2},jk}$ represent the weights from the ith input node to the jth hidden layer node and the jth hidden node to the kth output node. The intermediate variables for the jth hidden layer node and the outputs of the neural networks can be written as:

$$p_{c,j} = \sum_{i=1}^{n} e_i(t) w_{\text{critic1},ij}(t) + \sum_{i=n+1}^{2n} x_{ref,i-n}(t) w_{\text{critic1},ij}(t), \quad j = 1, \ldots, 10 \tag{23}$$

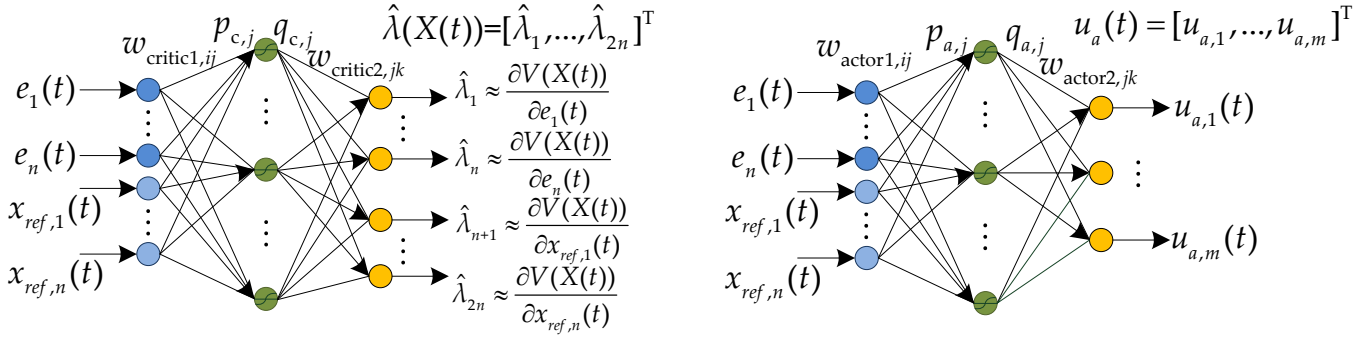
$$q_{c,j} = \tanh(p_{c,j}) = \frac{e^{p_{c,j}} - e^{-p_{c,j}}}{e^{p_{c,j}} + e^{-p_{c,j}}}, \quad j = 1, \ldots, 10 \tag{24}$$

$$\hat{\lambda}_k = \sum_{j=1}^{10} q_{c,j} w_{\text{critic2},jk}(t), \quad k = 1, \ldots, 2n \tag{25}$$

$$p_{a,j} = \sum_{i=1}^{n} e_i(t) w_{\text{actor1},ij}(t) + \sum_{i=n+1}^{2n} x_{ref,i-n}(t) w_{\text{actor1},ij}(t), \quad j = 1, \ldots, 10 \tag{26}$$

$$q_{a,j} = \tanh(p_{a,j}) = \frac{e^{p_{a,j}} - e^{-p_{a,j}}}{e^{p_{a,j}} + e^{-p_{a,j}}}, \quad j = 1, \ldots, 10 \tag{27}$$

$$u_{a,k} = \sum_{j=1}^{10} q_{aj} w_{\text{actor2},jk}(t), \quad k = 1, \ldots, m \tag{28}$$



(**a**) critic neural network (**b**) actor neural network

Figure 3. The critic NN and actor NN.

It is desired to update the critic NN by minimizing the square Bellman error:

$$E_C(t) = \frac{1}{2} e_C^{\mathrm{T}}(t) e_C(t) \tag{29}$$

where

$$\begin{aligned} e_C(t) &= \frac{\partial[V_X(t-1) - J_X(X(t-1), u_a(t-1)) - \gamma V_X(t)]}{\partial X(t-1)} = \lambda(X(t-1)) - \frac{\partial J_X(X(t-1), u_a(t-1))}{\partial X(t-1)} - \gamma \frac{\partial V_X(t)}{\partial X(t-1)} \\ &= \lambda(X(t-1)) - \left(X(t-1)^{\mathrm{T}} Q_X + u_a(t-1)^{\mathrm{T}} R \frac{\partial u_a(t-1)}{\partial X(t-1)} \right) - \gamma \lambda(X(t)) \left(\frac{\partial X(t)}{\partial X(t-1)} + \frac{\partial X(t)}{\partial u_a(t-1)} \frac{\partial u_a(t-1)}{\partial X(t-1)} \right) \end{aligned} \tag{30}$$

In Equation (30), $\lambda(X(t-1))$ and $\lambda(X(t))$ can be approximated by critic NN. $\partial u_a(t-1)/\partial X(t-1)$ is calculated by actor NN. The transition matrices $\partial X(t)/\partial X(t-1)$ and $\partial X(t)/\partial u_a(t-1)$ can be obtained from Equation (9). Because the partial derivative f_1 with respect to the angle rate of other channels is close to 0, we can obtain the transition matrices as follows: (The input dynamics are identified by RLS.)

$$\frac{\partial X(t)}{\partial X(t-1)} \approx \begin{bmatrix} \mathrm{I} & 0 \\ 0 & \mathrm{I} \end{bmatrix}_{2n \times 2n} \tag{31}$$

$$\frac{\partial X(t)}{\partial u_a(t-1)} \approx \begin{bmatrix} \Delta t g(t-1) \\ 0 \end{bmatrix}_{2n \times m} \approx \begin{bmatrix} \hat{\varsigma}_1(t-1) \\ 0 \end{bmatrix}_{2n \times m} \tag{32}$$

The weights of critic NN are updated by the gradient descent method with a learning rate η_C.

$$\begin{aligned} & w_{\text{critic2}}(t+1) = w_{\text{critic2}}(t) - \eta_C e_C(t)^{\mathrm{T}} \frac{\partial \hat{\lambda}(X(t-1))}{\partial w_{\text{critic2}}(t)} \\ & \approx w_{\text{critic2}}(t) - \eta_C \left(\hat{\lambda}(X(t-1)) - \left(X(t-1)^{\mathrm{T}} Q_X + u_a(t-1)^{\mathrm{T}} R \frac{\partial u_a(t-1)}{\partial X(t-1)} \right) - \gamma \hat{\lambda}(X(t)) \left(\begin{bmatrix} \mathrm{I} & 0 \\ 0 & \mathrm{I} \end{bmatrix} + \begin{bmatrix} \hat{\varsigma}_1(t-1) \\ 0 \end{bmatrix} \frac{\partial u_a(t-1)}{\partial X(t-1)} \right) \right)^{\mathrm{T}} \frac{\partial \hat{\lambda}(X(t-1))}{\partial w_{\text{critic2}}(t)} \end{aligned} \tag{33}$$

$$\begin{aligned} & w_{\text{critic1}}(t+1) = w_{\text{critic1}}(t) - \eta_C e_C(t)^{\mathrm{T}} \frac{\partial \hat{\lambda}(X(t-1))}{\partial w_{\text{critic1}}(t)} \approx w_{\text{critic1}}(t) - \\ & \eta_C \left(\hat{\lambda}(X(t-1)) - \left(X(t-1)^{\mathrm{T}} Q_X + u_a(t-1)^{\mathrm{T}} R \frac{\partial u_a(t-1)}{\partial X(t-1)} \right) - \gamma \hat{\lambda}(X(t)) \left(\begin{bmatrix} \mathrm{I} & 0 \\ 0 & \mathrm{I} \end{bmatrix} + \begin{bmatrix} \hat{\varsigma}_1(t-1) \\ 0 \end{bmatrix} \frac{\partial u_a(t-1)}{\partial X(t-1)} \right) \right)^{\mathrm{T}} \frac{\partial \hat{\lambda}(X(t-1))}{\partial q_c} \frac{\partial q_c}{\partial p_c} \frac{\partial p_c}{\partial w_{\text{critic1}}(t)} \end{aligned} \tag{34}$$

To minimize the value function in Equation (21), we can update the weights of actor NN by the gradient descent method, similarly. The learning framework of ED-DHP is shown in Algorithm 1.

$$\begin{aligned} w_{\text{actor2}}(t+1) &= w_{\text{actor2}}(t) - \eta_A \frac{\partial V_X(t-1)}{\partial u_a(t-1)} \frac{\partial u_a(t-1)}{\partial w_{actor2}(t)} \approx w_{\text{actor2}}(t) - \eta_{\text{A}} \left[\frac{\partial J(t-1)}{\partial u_a(t-1)} + \gamma \hat{\lambda}(X(t)) \frac{\partial X(t)}{\partial u_a(t-1)} \right] \frac{\partial u_a(t-1)}{\partial w_{\text{actor2}}(t)} \\ &\approx w_{\text{actor2}}(t) - \eta_A \left[u_a(t-1)^T R + \gamma \hat{\lambda}(X(t)) \begin{bmatrix} \hat{\varsigma}_1(t-1) \\ 0 \end{bmatrix}_{2n \times m} \right] \frac{\partial u_a(t-1)}{\partial w_{\text{actor2}}(t)} \end{aligned} \tag{35}$$

$$\begin{aligned} w_{\text{actor1}}(t+1) &= w_{\text{actor1}}(t) - \eta_A \frac{\partial V_X(t-1)}{\partial u_a(t-1)} \frac{\partial u_a(t-1)}{\partial w_{actor1}(t)} \approx w_{\text{actor1}}(t) - \eta_{\text{A}} \left[\frac{\partial J(t-1)}{\partial u_a(t-1)} + \gamma \hat{\lambda}(X(t)) \frac{\partial X(t)}{\partial u_a(t-1)} \right] \frac{\partial u_a(t-1)}{\partial w_{\text{actor1}}(t)} \\ &\approx w_{\text{actor1}}(t) - \eta_{\text{A}} \left[u_a(t-1)^{\text{T}} R + \gamma \hat{\lambda}(X(t)) \begin{bmatrix} \hat{\varsigma}_1(t-1) \\ 0 \end{bmatrix}_{2n \times m} \right] \frac{\partial u_a(t-1)}{q_a} \frac{\partial q_a}{\partial p_a} \frac{\partial p_a}{\partial w_{\text{actor1}}(t)} \end{aligned} \tag{36}$$

Algorithm 1 Learning Framework

Require:
Simulation parameters: sampling time Δt and the total simulation time T;
The controller parameters for the slow loop subsystem;
The forgetting factor of RLS;
The positive definite matrices;
The discount factor and the learning rates of two neural networks.
Initialize:
The initial states and inputs of the air vehicle;
The parameter matrix and the estimation covariance matrix of RLS;
Initialize the parameters of two neural networks randomly.
Online Learning:
1: **for** t = 0 to $(T/\Delta t) - 1$ **do**;
2: Calculate reference angular rate vector ω_{ref} of the fast loop subsystem with (4);
3: Obtain the angular rate vector ω through the sensors;
4: Update the RLS model with (11)–(13);
5: Update the parameters of the critic NN with (33) and (34);
6: Update the parameters of the actor NN with (35) and (36);
7: Calculate the output vector of the actor NN u_a with (26)–(28);
8: Calculate the control input vector of the air vehicle with (19);
9: Get the next system states by taking action u;
10: **end for**.

4. Numerical Simulation and Discussion

In this section, the ED-DHP's capability of online self-learning without complete knowledge of system dynamics and reference trajectory dynamics is demonstrated. Two numerical simulations are conducted to verify the design of the control scheme, sinusoidal signal tracking at a fixed operating point and command tracking with a morphing process at variable operating points.

4.1. Example 1

The scheme firstly controls the angle of attack α, sideslip angle β, and bank angle γ_V to track the sinusoidal reference signals under the condition of a fixed operating point. The initial conditions are $\alpha(0) = 2^\circ$, $\beta(0) = \gamma_V(0) = 0^\circ$, and $\omega_x(0) = \omega_y(0) = \omega_z(0) = 0^\circ/\text{s}$. The primary control surfaces are $\delta_z(0) = -11^\circ$ and $\delta_x(0) = \delta_y(0) = 0^\circ$, which are close to the sketchy trim values. The reference commands are designed as:

$$\begin{cases} \alpha_{ref} = 2 + 2 \times \sin(0.4\pi t) \\ \beta_{ref} = 1 \times \sin(0.4\pi t) \\ \gamma_{Vref} = 1 \times \sin(0.4\pi t) \end{cases} \tag{37}$$

The controller parameters for the slow loop subsystem are set as $K_\alpha = K_\beta = K_{\gamma_V} = 3.5$. The positive parameters of TD are designed as $r = 120$ and $\tau = 0.9$. The forgetting factor

of RLS is $\gamma_{\mathrm{RLS}} = 1.0$, to ensure consistency of the identification. It was found that if γ_{RLS} is set to a value less than 1, the identification speed is improved with the reduction of stability of ED-DHP. So the parameters identified onlineshould have the correct sign and, to some extent, the correct relative magnitude in the self-learning process. The initial parameter matrix of RLS is set as follows, (which only conforms to the polarity of the vehicle):

$$\varsigma(0) = \begin{bmatrix} -0.3 & 0.0 & 0.0 \\ 0.0 & -0.3 & 0.0 \\ 0.0 & 0.0 & -0.3 \\ 0.05 & 0.05 & 0.05 \end{bmatrix} \in \mathbb{R}^{(m+1)\times n}|_{m=n=3}$$

The initial estimation covariance matrix is set as:

$$Cov(0) = \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 100 \end{bmatrix} \in \mathbb{R}^{(m+1)\times(m+1)}|_{m=n=3}$$

The ED-DHP is implemented with the sampling time $\Delta t = 0.01s$, and the integration period of RK4 is 0.001s. The weights of the critic NN and actor NN are randomly initialized in $[-0.5, 0.5]$. The positive definite matrices of ED-DHP are $Q = \mathrm{diag}\{800, 800, 800\}$ and $R = \mathrm{diag}\{0.1, 0.2, 0.2\}$. The learning rate of critic NN is $\eta_C = 0.004$ and of the actor NN is $\eta_{\mathrm{A}} = 0.002$, respectively. The discount factor γ is set as 0.99.

As is shown in Figures 4–7, both α, β, and γ_V can follow the reference signals after less than 2 s. From Figure 4, we see that during the learning process, the max dynamic tracking errors of α, β, and γ_V are 0.6 degrees, 1 degree, and 0.8 degrees, respectively.

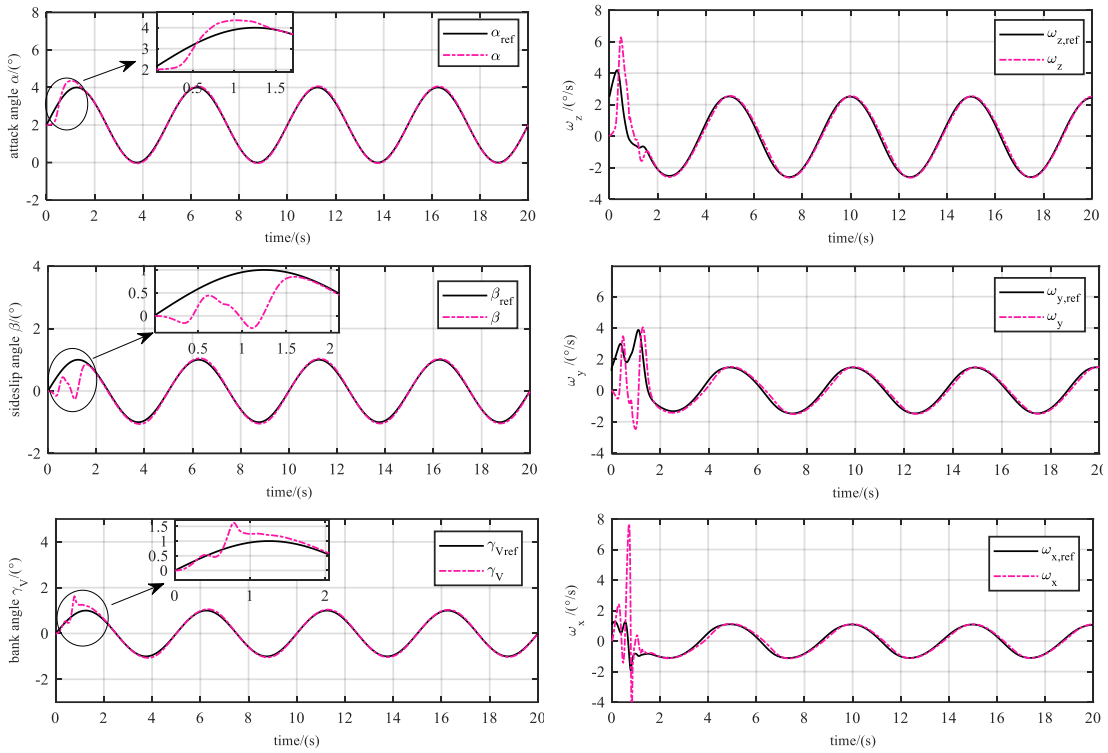


Figure 4. Tracking performance. The attack angle, sideslip angle, and bank angle response (**left**), and angular rates responses (**right**). It can be seen that the angles can follow the reference signals in about 2 s. There are certain oscillations in each channel to fully excite the identification and optimize the learning process.

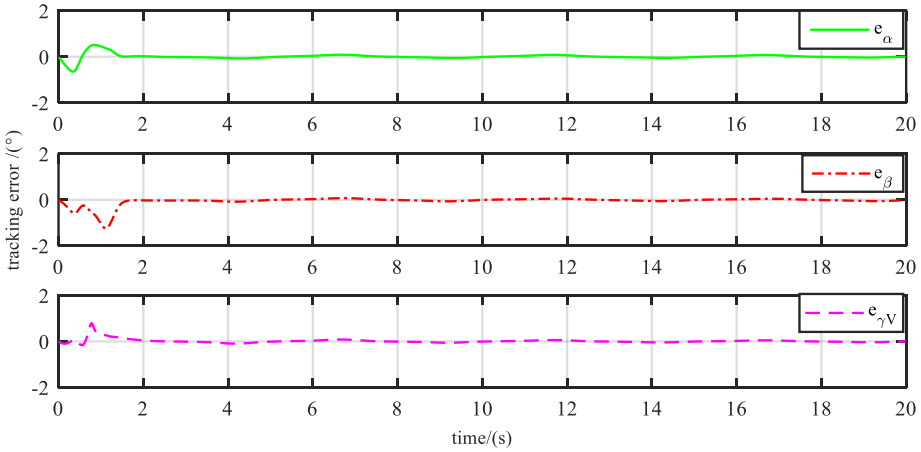


Figure 5. Tracking errors. It takes a little longer to learn to control the sideslip angle and bank angle than the attack angle. This can be attributed to the stronger coupling characteristics between the roll and yaw channels.

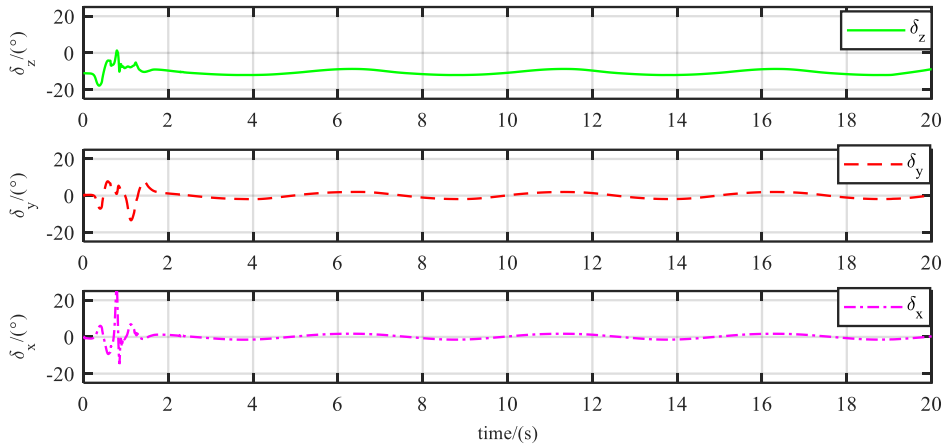


Figure 6. Control signals. It is worth noting that the equivalent surface of the pitch channel is initialized near the rough trim value of –11 degrees. If the trim value of your plant is close to 0, you can initialize it at 0 degrees.

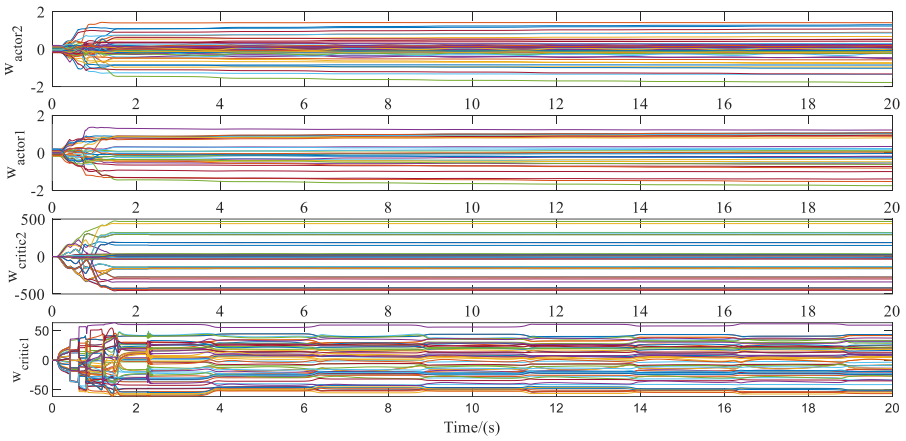


Figure 7. Weights updating of actor NN and critic NN. The parameters of neural networks converge to a steady state at about 1.4 s.

After the parameters of the neural networks converge to a steady state, the tracking errors of the three angles are maintained within 0.05 degrees. The same observation is made from the parameters of two NNs, as illustrated in Figure 7. The ED-DHP shows good

stability even though the learning rate does not change throughout the tracking process. However, it is worth noting that the angular rate loop has a large amplitude oscillation, especially the yaw rate loop and the roll rate loop. The phenomenon is mainly aroused by the cross-channel coupling and characteristics of the air vehicle's roll channel.

4.2. Example 2

To further verify the effectiveness and robustness of the proposed control scheme, the guidance command tracking with morphing process at variable operating points is considered, as well as the model uncertainty.

The initial tracking errors of the three channels are 1 degree, and the control parameters of the fast loop and slow loop are all the same as in Section 4.1. Three cases with different parameters perturbation are carried into the numerical simulation, shown in Table 1. The switch of the air vehicle's configuration occurs between 20 s~23 s. To ensure the stability of initial learning, TD starts to act on the system at 5 s.

Table 1. The parameter uncertainty for 3 cases.

Case 0	Case 1	Case 2
$\Delta I = 0\%$	$\Delta I = -5\%$	$\Delta I = 5\%$
$\Delta m = 0\%$	$\Delta m = -5\%$	$\Delta m = 5\%$
$\Delta \rho = 0\%$	$\Delta \rho = 10\%$	$\Delta \rho = -10\%$
$\Delta C_m = 0\%$	$\Delta C_m = 30\%$	$\Delta C_m = -30\%$

As is shown in Figure 8, the whole duration of the simulation is 30 s, and the guidance commands are not concrete expressions. The controller makes the vehicle track the reference value of the attack angle of 2 degrees and maintains a sideslip angle and bank angle of around 0 degrees. After that, the vehicle is lifted at an attack angle of 8 degrees, morphing and maintained at the current angle of attack. Additionally, in this simulation condition, the flight speed of the vehicle is fast, and the dynamic pressure changes dramatically within the simulation time.

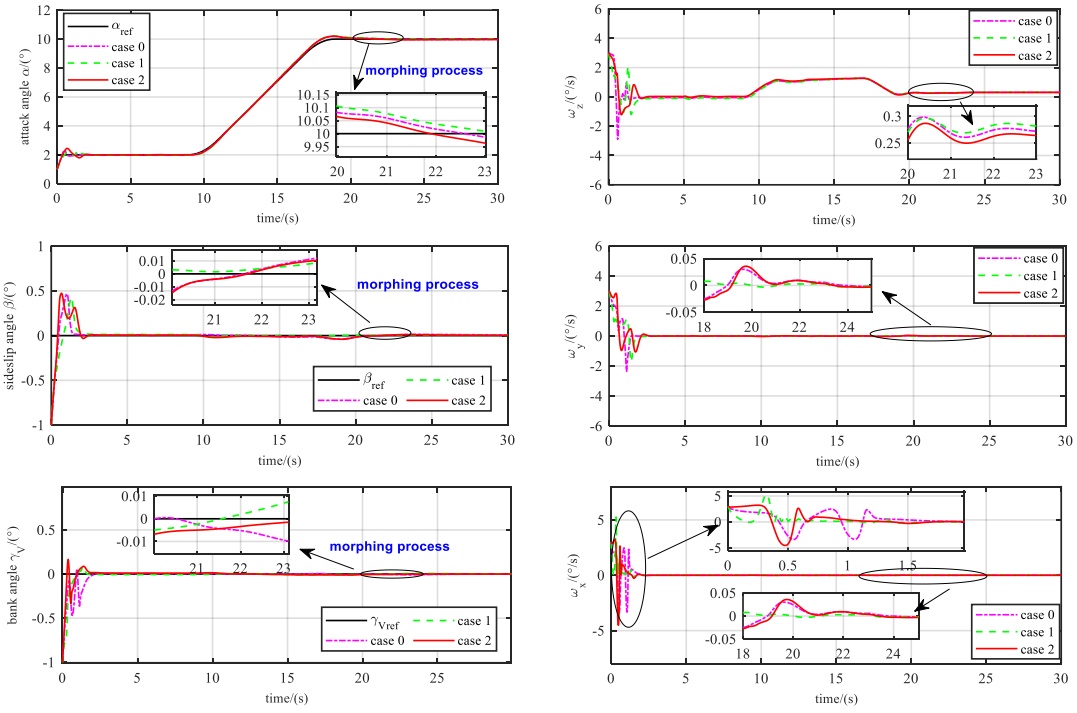


Figure 8. Tracking performance under three cases. In case 1, the dynamic pressure and moment coefficients are larger. So the system owns stronger control efficiency, causing higher requirements for the servo.

α, β, and γ_V can follow the reference signals after about 2 s. In Figure 9, the max tracking errors of α, β, and γ_V are both within 0.5 degrees before 2 s. The tracking errors of the attack angle are shown in Table 2.

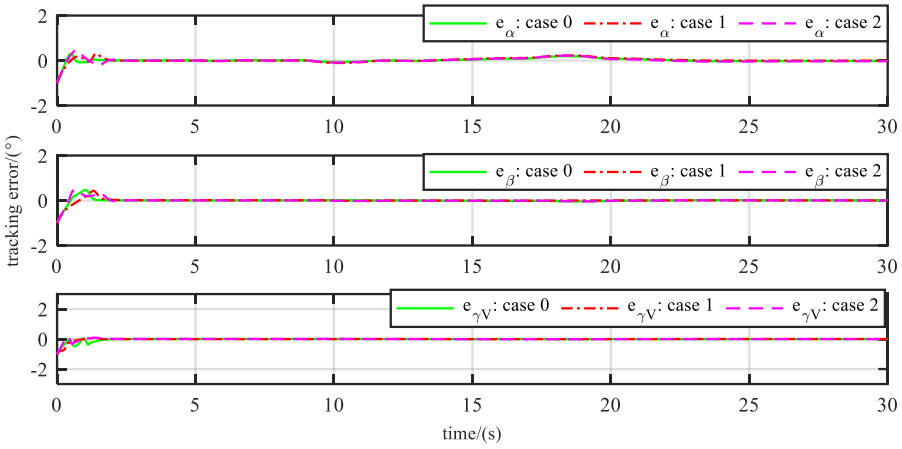


Figure 9. Tracking errors under three cases.

Table 2. Tracking errors of attack angle at different time periods.

Time	Case 0	Case 1	Case 2
2 s~10 s	$< 0.005°$	$< 0.01°$	$< 0.0025°$
10 s~20 s	$< 0.19°$	$< 0.22°$	$< 0.21°$
20 s~25 s	$< 0.08°$	$< 0.1°$	$< 0.06°$
25 s~30 s	$< 0.02°$	$< 0.005°$	$< 0.04°$

Figure 10 shows that the drastic changes in control surfaces are mainly concentrated in the first two seconds of the learning phase. In the lifting process, the symbol change of the pitch channel's equivalent control surface has occurred, which means the desperate transformation of the operating points. This can be furthermore reflected in the sharp change of critic NN parameters in Figure 11, while the parameters of actor NN are only slightly adjusted.

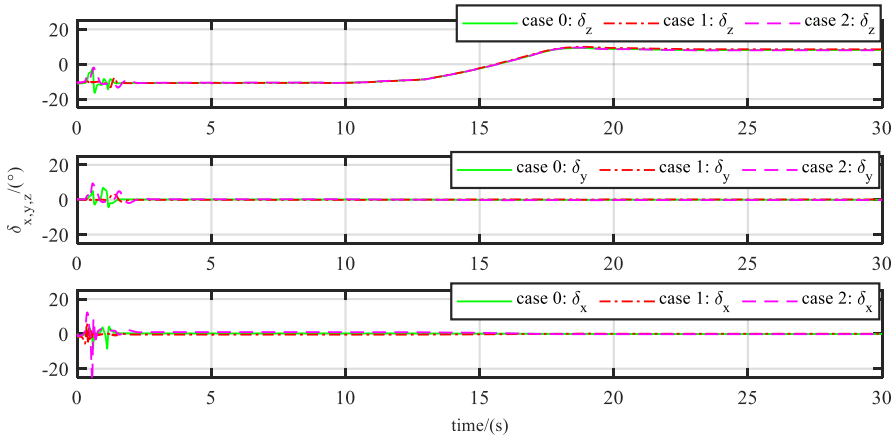


Figure 10. Control signals under three cases.

The parameters of critic NN are obviously affected during the morphing process. However, the actor NN is adjusted slightly. It shows that the updating of NNs is mainly related to the tracking error. When actor NN owns strong control ability, that is, the tracking errors are very small during the morphing, the network parameters can adapt to the process only by fine-tuning.

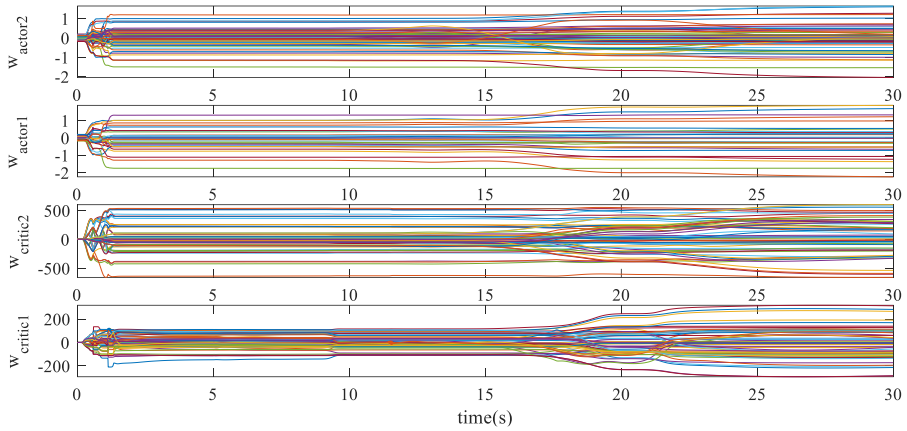


Figure 11. Weights updating of actor NN and critic NN.

5. Conclusions

This article presents the ED-DHP method for online nonlinear flight attitude control. ED-DHP defines the augmented system state, which contains tracking error and reference trajectory, to solve the tracking problem. This can make the actor neural network learn the feedforward and feedback control terms simultaneously without requiring complete knowledge of the vehicle dynamics and the reference trajectory dynamics. An RLS method is used to identify not only the input dynamics but also the total uncertainty based on the error dynamics in real-time.

This proposed method is applied in two numerical experiments based on a morphing air vehicle model: sinusoidal signal tracking at a fixed operating point and guidance command tracking with morphing process at variable operating points. The results indicate the robustness and adaptiveness of ED-DHP. They also reveal that ED-DHP can learn the multichannel attitude control policy online without offline training.

In the meantime, there are some questions worth pondering:

1. Augmented state increases the dimension of the system state space, which may affect the convergence speed of the algorithm without offline training;
2. The systems with input constraints can be further considered by designing the new cost function to make this method better apply to real vehicle systems;
3. ED-DHP is suitable for the control of fast loop subsystems. It is necessary to optimize the error dynamic and RLS method to expand the application scenarios of ED-DHP.

Author Contributions: Conceptualization, X.H. and J.L.; methodology, X.H., H.Z. and Y.P.; software, X.H.; validation, Y.Z., J.L. and Z.W. formal analysis, X.H., Y.Z. and J.L.; writing—original draft preparation, X.H.; writing—review and editing, Y.Z., J.L., H.Z., Z.W. and Y.P.; visualization, Y.Z.; supervision, J.L. and Z.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China, grant number U21B2028.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Chai, R.; Tsourdos, A.; Savvaris, A. Review of advanced guidance and control algorithms for space/aerospace vehicles. *Prog. Aerosp. Sci.* **2021**, *122*, 100696. [CrossRef]
2. Ding, Y.B.; Yue, X.K.; Chen, G.S. Review of control and guidance technology on hypersonic vehicle. *Chin. J. Aeronaut.* **2022**, *35*, 1–18. [CrossRef]
3. Kober, J.; Peters, J. Reinforcement Learning in Robotics: A Survey. In *Learning Motor Skills. Springer Tracts in Advanced Robotics*; Springer: Cham, Switzerland, 2014; Volume 97, pp. 9–15.
4. Degrave, J.; Felici, F.; Buchli, J. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature* **2022**, *602*, 414–419. [CrossRef] [PubMed]
5. Tang, X.; Qin, Z.T.; Zhang, F. A Deep Value-network Based Approach for Multi-Driver Order Dispatching. In Proceedings of the 25th ACM SIGKDD International Conference, Anchorage, AK, USA, 4–8 August 2019; pp. 1780–1790.
6. Zeng, S.; Kody, A.; Kim, Y. A Reinforcement Learning Approach to Parameter Selection for Distributed Optimization in Power Systems. Available online: https://arxiv.org/abs/2110.11991 (accessed on 22 October 2021).
7. Dong, C.Y.; Liu, C.; Wang, Q. Switched Adaptive Active Disturbance Rejection Control Of Variable Structure Near Space Vehicles Based On Adaptive Dynamic Programming. *Chin. J. Aeronaut.* **2019**, *32*, 1684–1694. [CrossRef]
8. Liu, C.; Dong, C.Y.; Zhou, Z.J. Barrier Lyapunov Function Based Reinforcement Learning Control For Air-Breathing Hypersonic Vehicle With Variable Geometry Inlet. *Aerosp. Sci. Technol.* **2019**, *96*, 105537. [CrossRef]
9. Jia, C.H.; Gong, Q.H.; Huang, X. Synchronization Control of Nonlinear Chaotic Systems with Deep Reinforcement Learning Algorithm. In Proceedings of the International Conference on Guidance, Navigation and Control, Tianjin, China, 23–25 October 2020; pp. 1673–1682.
10. Hui, J.P.; Wang, R.; Yu, Q.D. Generating new quality flight corridor for reentry aircraft based on reinforcement learning. *Acta Aeronaut. Astronaut. Sin.* **2022**, *43*, 325960.
11. Kiumarsi, B.; Vamvoudakis, K.G.; Modares, H. Optimal and Autonomous Control Using Reinforcement Learning: A Survey. *IEEE T. Neur. Net. Lear.* **2018**, *29*, 2042–2062. [CrossRef]
12. Ferrari, S.; Stengel, R. Online adaptive critic flight control. *J. Guid. Control. Dynam.* **2004**, *27*, 777–786. [CrossRef]
13. Mu, C.X.; Ni, Z.; Sun, C.Y. Air-breathing hypersonic vehicle tracking control based on adaptive dynamic programming. *IEEE T. Neur. Net. Lear.* **2017**, *28*, 584–598. [CrossRef] [PubMed]
14. Xia, R.; Chen, M.; Wu, Q. Neural Network based Integral Sliding Mode Optimal Flight Control of Near Space Hypersonic Vehicle. *Neurocomputing* **2020**, *379*, 41–52. [CrossRef]
15. Huang, Y.; Liu, D. Neural-network-based optimal tracking control scheme for a class of unknown discrete-time nonlinear systems using iterative ADP algorithm. *Neurocomputing* **2014**, *125*, 46–56. [CrossRef]
16. Wang, Q.; Gong, L.G.; Dong, C.Y. Morphing aircraft control based on switched nonlinear systems and adaptive dynamic programming. *Aerosp. Sci. Technol.* **2019**, *93*, 105325. [CrossRef]
17. Han, X.; Zheng, Z.; Liu, L. Online policy iteration ADP-based attitude-tracking control for hypersonic vehicles. *Aerosp. Sci. Technol.* **2020**, *106*, 106233. [CrossRef]
18. Bertsekas, D.P.; Homer, M.L.; Logan, D.A. Missile Defense and Interceptor Allocation By Neuro-Dynamic Programming. *T-SMCA* **2000**, *1*, 42–51. [CrossRef]
19. Van Kampen, E.; Chu, Q.P.; Mulder, J.A. Continuous Adaptive Critic Flight Control Aided with Approximated Plant Dynamics. In Proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit, Keystone, CO, USA, 21–24 August 2006.
20. Zhou, Y.; van Kampen, E.J.; Qi, P.C. Nonlinear Adaptive Flight Control Using Incremental Approximate Dynamic Programming and Output Feedback. *J. Guid Control. Dynam.* **2016**, *40*, 1–8. [CrossRef]
21. Zhou, Y.; van Kampen, E.J.; Qi, P.C. Incremental model based online dual heuristic programming for nonlinear adaptive control. *Control. Eng.Pract.* **2018**, *73*, 13–25. [CrossRef]
22. Li, H.X.; Sun, L.G.; Tan, W.Q. Switching Flight Control for Incremental Model-Based Dual Heuristic Dynamic Programming. *J. Guid, Control. Dynam.* **2020**, *43*, 1–7. [CrossRef]
23. Liu, J.H.; Shan, J.Y.; Rong, J.L. Incremental reinforcement learning flight control with adaptive learning rate. *J. Astronaut.* **2022**, *43*, 111–121.
24. Lee, J.H.; van Kampen, E.J. Online reinforcement learning for fixed-wing aircraft longitudinal control. In Proceedings of the AIAA Scitech 2021 Forum, Online, 11–15 January 2021.
25. Heyer, S.; Kroezen, D.; Van Kampen, E. Online Adaptive Incremental Reinforcement Learning Flight Control for a CS-25 Class Aircraft. In Proceedings of the AIAA Scitech 2020 Forum, Orlando, FL, USA, 6–10 January 2020.
26. Han, J.Q. *Auto Disturbance Rejection Control Technology*; National Defense Industry Press: Beijing, China, 2008; pp. 64–67.
27. Lewis, F.L.; Vrabie, D. Reinforcement learning and adaptive dynamic programming for feedback control. *IEEE Circ. Syst. Mag.* **2009**, *9*, 32–50. [CrossRef]

28. Kiumarsi, B.; Lewis, F.L. Actor-critic-based optimal tracking for partially unknown nonlinear discrete-time systems. *IEEE Trans. Neural Netw. Learn. Syst.* **2015**, *26*, 140–151. [CrossRef]
29. Modares, H.; Lewis, F.L. Optimal tracking control of nonlinear partially-unknown constrained-input systems using integral reinforcement learning. *Automatic* **2014**, *50*, 1780–1792. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.