




Article

Safe Reinforcement Learning for Transition Control of Ducted-Fan UAVs

Yanbo Fu ^{1,2} , Wenjie Zhao ^{1,2,*}  and Liu Liu ¹ 

¹ School of Aeronautics and Astronautics, Zhejiang University, Hangzhou 310027, China; 22124084@zju.edu.cn (Y.F.)

² Center for Unmanned Aerial Vehicles, Huanjiang Laboratory, Zhuji 311800, China

* Correspondence: zhaowenjie8@zju.edu.cn

Abstract: Ducted-fan tail-sitter unmanned aerial vehicles (UAVs) provide versatility and unique benefits, attracting significant attention in various applications. This study focuses on developing a safe reinforcement learning method for back-transition control between level flight mode and hover mode for ducted-fan tail-sitter UAVs. Our method enables transition control with a minimal altitude change and transition time while adhering to the velocity constraint. We employ the Trust Region Policy Optimization, Proximal Policy Optimization with Lagrangian, and Constrained Policy Optimization (CPO) algorithms for controller training, showcasing the superiority of the CPO algorithm and the necessity of the velocity constraint. The transition trajectory achieved using the CPO algorithm closely resembles the optimal trajectory obtained via the well-known GPOPS-II software with the SNOPT solver. Meanwhile, the CPO algorithm also exhibits strong robustness under unknown perturbations of UAV model parameters and wind disturbance.

Keywords: safe reinforcement learning; ducted fan; transition control; unmanned aerial vehicle (UAV)



Citation: Fu, Y.; Zhao, W.; Liu, L. Safe Reinforcement Learning for Transition Control of Ducted-Fan UAVs. *Drones* **2023**, *7*, 332. <https://doi.org/10.3390/drones7050332>

Academic Editor: Mostafa Hassanalian

Received: 15 April 2023

Revised: 12 May 2023

Accepted: 18 May 2023

Published: 22 May 2023



Copyright: © 2023 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

In recent years, vertical take-off and landing (VTOL) unmanned aerial vehicles (UAVs) have gained considerable attention due to their unique advantages. Distinct from traditional fixed-wing UAVs, VTOL UAVs are capable of taking off and landing vertically, eliminating the need for a runway [1]. Moreover, when compared to multi-rotor UAVs, they offer several benefits, such as a larger payload capacity, higher cruise speed, and longer flight ranges [2]. This is primarily attributed to their reliance on wings for lift production, as opposed to the multiple rotors utilized by multi-rotor UAVs. Among the various VTOL UAV configurations, such as tilt rotor, tail sitter, and vectored thrust, the ducted-fan tail-sitter fixed-wing UAV stands out as a unique design. Notably, they eliminate the need for additional moving parts to achieve VTOL capabilities, leading to a simplified mechanical design that is both easier to maintain and less prone to mechanical failures [3]. Moreover, the incorporation of a ducted fan enhances the propulsion efficiency, allowing the UAVs to cover longer distances. The ducted fan also contributes to noise reduction, making these UAVs particularly suitable for noise-sensitive environments [4].

Ducted-fan tail-sitter fixed-wing UAVs integrate the characteristics of both traditional fixed-wing and multi-rotor UAVs, enabling them to perform level flight and hover operations. However, during the transition process, the UAV experiences aerodynamic instability due to a stall occurring at the fixed wing [5]. As a result, the control strategies for transition maneuvers between these two flight modes are of paramount importance, necessitating extensive investigation and research. Generally, two prevalent methods to address this issue can be found in the current literature [3].

The first approach conceptualizes the transition process as a trajectory optimization problem. In [6], the direct collocation method was employed to obtain the optimal transition trajectory by Li et al. An optimization approach using the interior point method

was proposed that focused on altitude changes during transition in [7]. The optimal feed-forward control input for transition was computed via sequential quadratic programming by Kubo and Suzuki [8]. Banazadeh devised a gradient-based algorithm based on the classical Cauchy method to generate optimal transition trajectories [9]. Naldi and Marconi [10] utilized mixed-integer nonlinear programming to tackle the minimum-time and minimum-energy optimal transition problems.

The second approach is structured in two stages: the first stage entails devising the desired trajectory during the transition, and the second stage involves designing a controller to track the trajectory established in the first stage [3]. Jeong et al. proposed a continuous ascent transition trajectory consisting of the angle of attack and flight path angle and invoked dynamic inversion control for tracking [11]. Flores provided a desired velocity trajectory and implemented a recurrent neural network-based controller for feedback linearization [12]. Cheng and Pei established a transition corridor based on the constraint of maintaining a fixed altitude [13]. They planned the desired velocity in the corridor and utilized an adaptive controller [14].

Alternatively, reinforcement learning (RL) has emerged as a promising approach to address various challenges in the domain of UAVs, owing to its ability to learn and adapt to dynamic environments [15]. As a result, there has been growing interest in leveraging RL algorithms to tackle the transition problem of VTOL UAVs. In [16], an RL-based controller for hybrid UAVs was designed that can not only automatically complete the transition but can also be adapted to different configurations. Xu [17] proposed a soft landing control algorithm based on the RL method. Yuksek employed the deep deterministic policy gradient algorithm to address the transition flight problem for tilt-rotor UAVs [18].

In this paper, we solve the back-transition task of ducted-fan tail-sitter fixed-wing UAVs (i.e., from level flight mode to hover mode) using safe RL algorithms. While prior work [16] mainly emphasizes the successful execution of transition maneuvers, our focus is on minimizing altitude changes and the transition time, while adhering to velocity constraints. Our method, compared to [18], integrates trajectory optimization and control problems into an RL-based learning process, thereby reducing the complexity and computational effort. To the best of our knowledge, this is one of the first works in which the RL methodology is utilized to solve the back-transition control problem of ducted-fan tail-sitter UAVs.

The main contributions of our work are as follows.

1. We develop a mathematical model of ducted-fan fixed-wing UAV dynamics. Based on this model, we create a training environment for ducted-fan UAVs in OpenAI GYM [19] using the fourth-order Runge–Kutta method.
2. Taking into account the velocity constraint during the transition process, we train controllers using Trust Region Policy Optimization (TRPO) with fixed penalty, Proximal Policy Optimization with Lagrangian (PPO_{Lag}), and Constrained Policy Optimization (CPO). We assess the performance of these algorithms and demonstrate the superiority of the CPO algorithm.
3. Comparing the CPO algorithm with the optimal trajectory obtained via GPOPS-II [20], we find that the performance of CPO closely approximates the optimal trajectory. In addition, the CPO algorithm has robustness under unknown perturbations of UAV model parameters and wind disturbance, which is lacking in the GPOPS-II software.

This paper is organized as follows. In Section 2, a mathematical model of a ducted-fan fixed-wing UAV is described. In Section 3, the general structure of the RL transition controller is introduced, and the reward function, action, and observation are explained. In Section 4, comparisons between CPO and other RL algorithms are reported. We also compare the transition trajectory of CPO with GPOPS-II and verify the robustness of the CPO algorithm. In Section 5, concluding remarks and future works are reported.

2. Mathematical Modeling

In this section, we describe a three-degree-of-freedom (DOF) longitudinal model for ducted-fan fixed-wing UAVs. The model is derived from the Newton and Euler theorems and is simplified from the full 6-DOF dynamic model. This 3-DOF model is able to speed up the process of assessing the impact of different reward functions and hyper-parameter settings compared to the 6-DOF dynamic model [21]. It is equipped with four groups of control vanes as the main control surfaces, each consisting of three movable vanes. In addition, four groups of fixed vanes are situated above the main control surfaces, intended to balance the anti-torque generated by the rotor, with each fixed group consisting of two fixed vanes (see Figure 1). The four groups of control vanes are numbered 1, 2, 3, and 4 and are employed to change the attitude of the UAVs (see Figure 2). Each group of three movable vanes, controlled by a single servo, deflects by the same angle. We use δ_1 , δ_2 , δ_3 , and δ_4 to represent the deflection of group numbers 1, 2, 3, and 4.

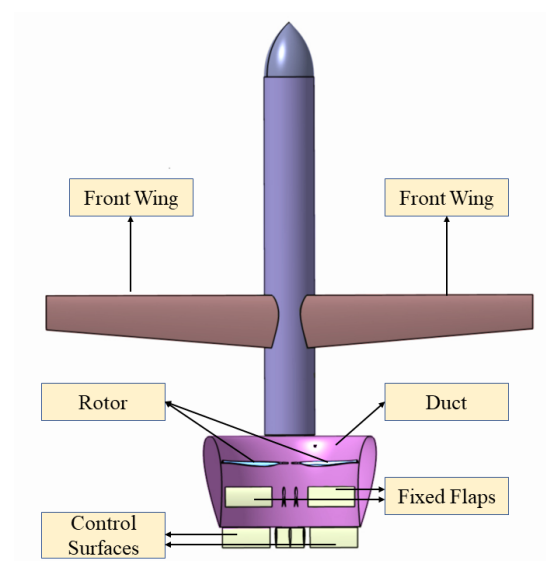


Figure 1. Ducted-fan UAV layout.

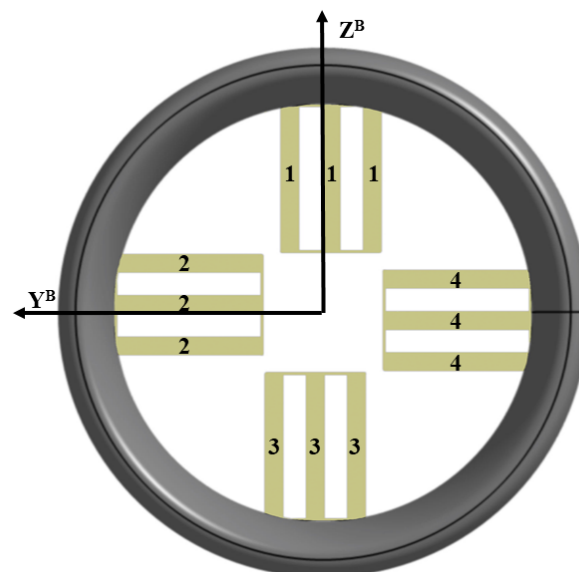


Figure 2. Control vanes.

2.1. Three-DOF Dynamics

For the ducted-fan VTOL UAV, two right-handed coordinate systems are applied to describe the states of the aircraft (see Figure 3). The inertial frame axes are denoted as $\{\Gamma^I: X^I, Y^I, Z^I\}$ and the body frame axes are denoted as $\{\Gamma^B: X^B, Y^B, Z^B\}$ with the origin located at the center of mass. Throughout this section, the superscripts $(.^I)$ and $(.^B)$ are utilized to specify whether a variable is formulated in the inertial or body frame. The position of the aircraft in Γ^I is described by $\xi = [x, y, z]^T$ and the velocity of the aircraft in Γ^B is defined as $V^B = [u, v, w]^T$. The Euler angle vector (i.e., roll, pitch, and yaw) is described by $\Theta = [\phi, \theta, \psi]^T$ and the angular velocity vector with respect to the body frame is denoted by $\Omega = [p, q, r]^T$. It should be noted that based on the above-mentioned definition, the pitch angle $\theta = 0^\circ$ at the level flight condition, and the pitch angle $\theta = 90^\circ$ at the landing and hover conditions (see Figure 4). Thus, the longitudinal dynamics of the aircraft are derived as follows:

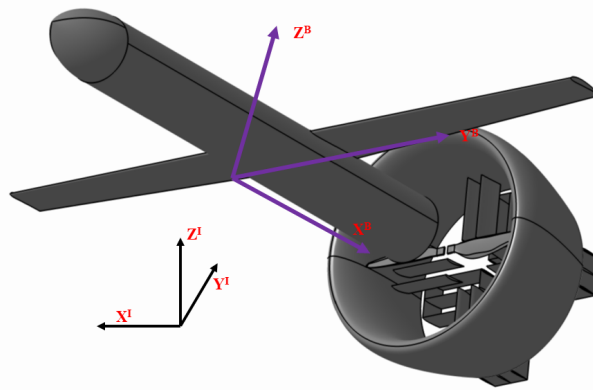


Figure 3. Inertial frame and body frame.

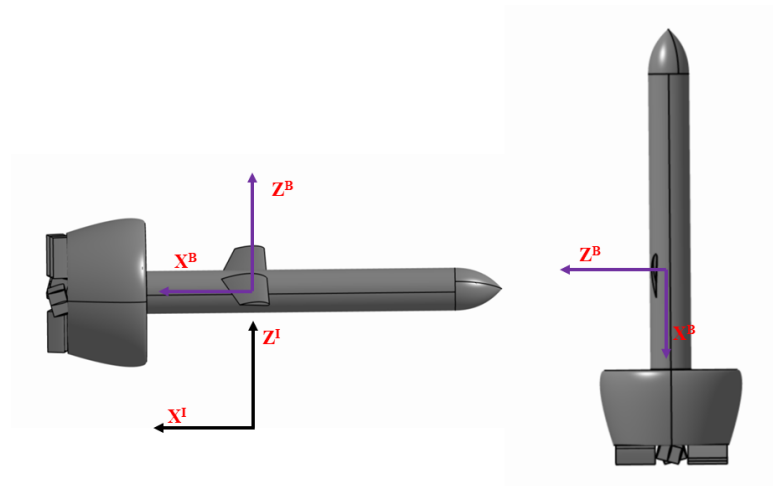


Figure 4. Left: level flight; Right: hover mode.

$$\begin{cases} \dot{u} = g \sin \theta + (D \cos \alpha - T - L \sin \alpha) / m - qw \\ \dot{w} = (F_{cw} + L \cos \alpha + D \sin \alpha + F_{cs}) / m + qu - g \cos \theta \\ \dot{x} = w \cos \theta - u \sin \theta \\ \dot{z} = u \cos \theta + w \sin \theta \\ \dot{\theta} = q \\ \dot{q} = (M_{pitch} + M_{cw} + M_{cs}) / I_{yy} \end{cases} \quad (1)$$

where g is gravity acceleration; I_{yy} is the moment of inertia on the Y^B axis; α represents the angle of attack. All the forces and moments are discussed below.

2.2. Rotor

In this subsection, the rotor model is discussed based on basic momentum theory and blade element theory [22]. Considering the airspeed along the X^B axis, the configuration of the blades, the airflow through the rotor, and the thrust of the rotor can be expressed as [23]

$$v_b = u + \frac{2}{3}\omega_r r \left(\frac{3}{4}K_{twist} \right) \quad (2)$$

$$T = \frac{1}{4}(v_b - v_i)\omega_r r^2 \rho_\infty a_0 b c_r \quad (3)$$

where $V^B = [u, v, w]^T$ is the velocity in the body frame, ρ_∞ is the air density, ω_r represents the angular velocity of the rotor, r is the radius of the rotor and K_{twist} is the twist of the blades, a_0 is the rotor lift curve slope, b is the number of blades, c_r is the chord of the rotor blade, and the induced velocity v_i and the far-field velocity v_f can be expressed as

$$v_f = \sqrt{(u - v_i)^2 + w^2 + v^2} \quad (4)$$

$$v_i = T/2\rho_\infty \pi r^2 v_f \quad (5)$$

The expressions for T and v_i can be solved iteratively through Equations (2)–(5) using the Newton–Raphson method.

2.3. Aerodynamics Model

The aerodynamic forces and moments include the lift L , the drag D , and the pitching moment M_{pitch} . They are primarily dependent on the fuselage, duct, and mostly on the wings. The angle of attack has a significant impact on the aerodynamic model, which can be expressed as follows:

$$\alpha = \begin{cases} 0 & \text{if } w = 0 \\ -\text{sgn}(w) \frac{\pi}{2} & \text{if } u = 0 \\ \arctan\left(\frac{w}{u}\right) & \text{if } u < 0 \\ \arctan\left(\frac{w}{u}\right) + \pi & \text{if } u > 0 \end{cases} \quad (6)$$

The ducted-fan UAV can be regarded as an entire lifting body, including the fuselage, wings, and duct [15]. This simplification introduces some inaccuracies in the aerodynamic data. To address these discrepancies, it is essential to incorporate compensation for the disturbances within the aerodynamic data. In order to account for the modeling errors, the lift and drag coefficients are multiplied by a perturbation factor, sampled from a uniform distribution ranging between 0.8 and 1.2.

During the transition mode, high angle of attack conditions can induce wing stall phenomena, leading to a significant reduction in lift. Traditional linear aerodynamic coefficient models are insufficient in accurately capturing the complexities of the aerodynamic behavior. As demonstrated in [24], an advanced aerodynamic model that integrates both the linear lift model and the effects of wing stall can be formulated as follows:

$$C_L(\alpha) = (1 - \sigma(\alpha)) [C_{L_0} + C_{L_\alpha} \alpha] + \sigma(\alpha) [C_{L_{max}} \sin \alpha \cos \alpha] \quad (7)$$

$$C_D(\alpha) = C_{D_p} + \frac{(C_{L_0} + C_{L_\alpha} \alpha)^2}{\pi e AR} \quad (8)$$

where

$$\sigma(\alpha) = \frac{1 + e^{-M(\alpha-\alpha_0)} + e^{M(\alpha+\alpha_0)}}{(1 + e^{-M(\alpha-\alpha_0)})(1 + e^{M(\alpha+\alpha_0)})} \tag{9}$$

$$C_{L\alpha} = \frac{\pi AR}{1 + \sqrt{1 + (AR/2)^2}} \tag{10}$$

The lift and drag coefficients are shown in Figure 5. The pitching moment coefficients will be expressed by the linear model as

$$C_m(\alpha) = C_{m_0} + C_{m_\alpha}\alpha \tag{11}$$

Thus, the lift L , drag D , and pitching moment M_{pitch} can be written as

$$\begin{aligned} L &= \frac{1}{2}\rho_\infty V^2 S C_L(\alpha) \\ D &= \frac{1}{2}\rho_\infty V^2 S C_D(\alpha) \\ M_{pitch} &= \frac{1}{2}\rho_\infty V^2 S C_m(\alpha) \end{aligned} \tag{12}$$

V is the air speed of the UAV and S is the wing area.

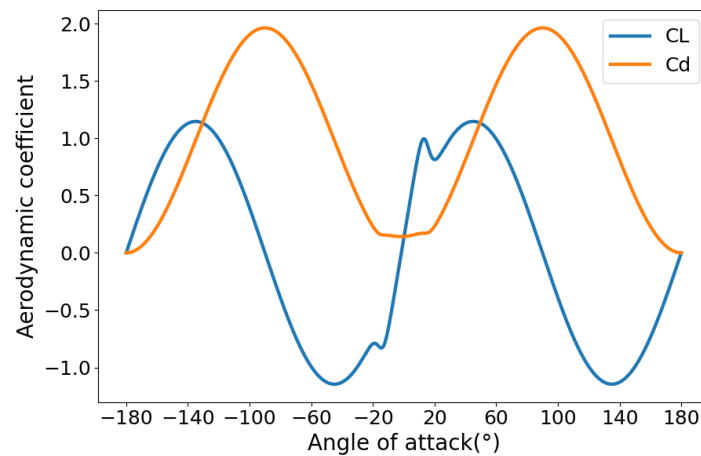


Figure 5. Lift and drag coefficient.

2.4. Momentum Drag

Due to the existence of crosswinds, the duct must generate a force to align the incoming airflow with its orientation. This results in a reaction force known as momentum drag. Moreover, crosswinds lead to the formation of a region with higher velocity over the near edge of the duct, as the air surrounding it is pulled into the duct by the rotor. This increased lift on the edge produces a moment that causes the vehicle to turn away from the crosswind, referred to as the momentum moment [22]. The formulas for the momentum drag and moment can be represented as follows:

$$F_{cw} = -v_i \rho_\infty \pi r^2 w \tag{13}$$

$$M_{cw} = C_{duct} \rho_\infty r |r| w \tag{14}$$

where w is the velocity of the Z-axis in the body frame, and v_i is determined by Equation (5).

2.5. Control Vanes

Each control vane can be modeled as an airfoil. In [3,14,15], the lift slope coefficient of vane is assumed to be constant. However, our Computational Fluid Dynamics simulations have shown that the lift coefficient of the vane remains virtually unchanged when subjected to a large angle of attack. Thus, based on the simple model in [25], the lift slope coefficient C_{lcs} of the vane can be expressed as

$$C_{lcs} = \begin{cases} 0.0625 & \text{if } |\alpha| \leq 16^\circ \\ 0 & \text{if } |\alpha| > 16^\circ \end{cases} \quad (15)$$

The dynamic pressure q_{cs} on each vane can be expressed as

$$q_{cs} = \frac{1}{2} \rho_\infty (u - v_i)^2 \quad (16)$$

where u is the velocity of the X-axis in the body frame. Based on the control allocation method [15], the equivalent vane deflection of pitch (Y-axis) is given as follows:

$$\delta_y = \delta_2 - \delta_4 \quad (17)$$

The drag forces of vanes can be neglected [15], and the vane's angle of attack only depends on the vane's deflection [3]. Thus, the force and moment generated by the control vanes are as follows:

$$F_{cs} = q_{cs} S_{cs} C_{lcs} \delta_y \quad (18)$$

$$M_{cs} = q_{cs} S_{cs} C_{lcs} l_e \delta_y \quad (19)$$

where S_{cs} represents the vane area and l_e is the arm of pitch moment.

3. Approach

3.1. Problem Formulation

In this section, we provide a mathematical expression for the UAV back-transition problem. In the back-transition process, the most straightforward strategy is that the UAV climbs at a large pitch angle while converting the kinetic energy into potential energy [26] (see Figure 6a). Although this approach offers a short transition time, it results in substantial altitude loss. In this paper, our desired trajectory, referred to as a "neat transition", aims to complete the transition as quickly as possible with minimal altitude loss [13] (see Figure 6b). To achieve this transition, we introduce an additional constraint on the Z-axis velocity, expressed as

$$\|v_z\| < 1$$

Thus, we pay attention to the altitude loss and transition time after the transition has been completed. The RL problem is defined as follows.

RL UAV Back-Transition Problem:

Minimize:

1. Terminal Velocity Error: $\|v\|$ at $t = t_f$
2. Terminal Attitude Error: $\|\theta - \pi/2\|$ at $t = t_f$
3. Terminal Angular Velocity Error: $\|\omega\|$ at $t = t_f$
4. Terminal Time: t_f
5. Height loss: $\|z_{t_f} - z_{t_0}\|$

Subject to:

1. Angular Velocity Constraint: pitch angle velocity rate $q < 180^\circ$
2. Control Constraints: $|\delta_y| < 30^\circ$ and $\omega_{rmin} < \omega_r < \omega_{rmax}$
3. Neat Transition Constraint: $\|v_z\| < 1 \text{ m/s}$

4. Equations of Motion (set by the environment)

To primarily optimize the terminal altitude loss and transition time, we opt to provide error margins for the terminal velocity, terminal angle, and terminal angular velocity. When the error falls within these margins, the transition is deemed successful. The terminal error range can be expressed as follows:

$$\begin{aligned}\|v_{t_f}\| &\leq 1 \text{ m/s} \\ \|\theta_{t_f} - \pi/2\| &\leq 0.08 \text{ rad} \\ \|\omega_{t_f}\| &\leq 0.08 \text{ rad/s}\end{aligned}$$

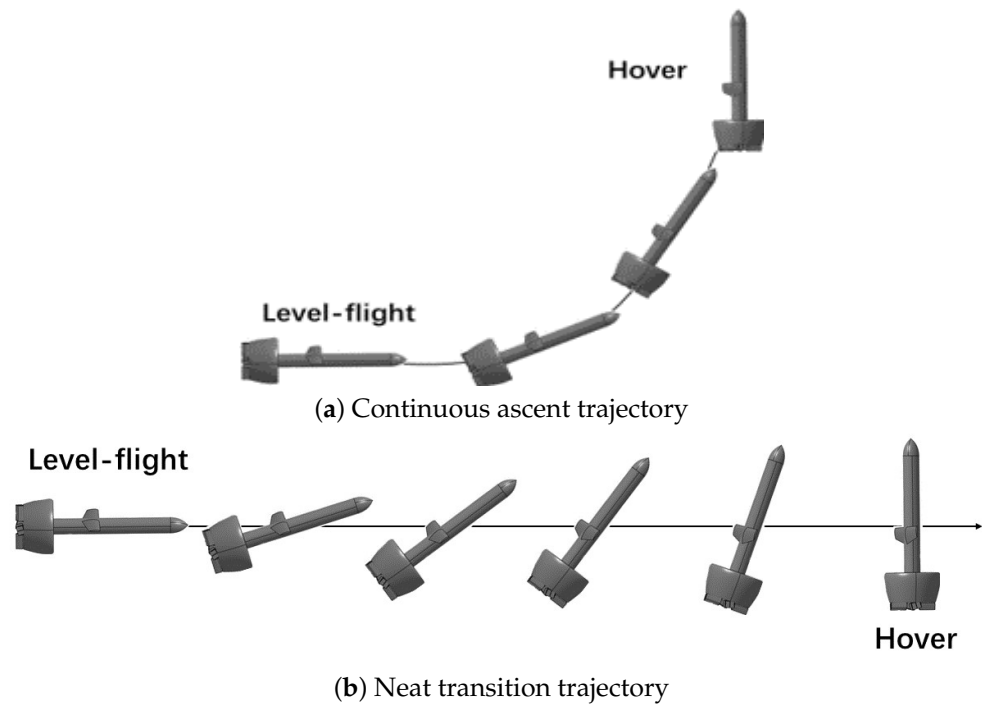


Figure 6. Different transition trajectories.

3.2. Algorithm

To address the neat transition constraint, traditional RL methods typically apply a fixed penalty through the reward function. However, if the penalty is too small, the agent may learn unsafe behavior, while an excessively severe penalty might result in the agent's inability to learn anything. In contrast, safe reinforcement learning algorithms employ the Lagrangian approach to tackle constraints, automatically balancing the weights between rewards and penalties. This ensures that the entire transition process adheres to the constraints while exploring the optimal transition performance. Among safe RL algorithms, we introduce the Constrained Policy Optimization (CPO) algorithm to solve the ducted-fan UAV back-transition problem [27].

3.2.1. Constrained Markov Decision Process (CMDP)

A Markov decision process (MDP) is a tuple, (S, A, P, γ, R) , where S is the set of states, A is the set of actions, and $R : S \times A \times S \rightarrow \mathbb{R}$ is the reward function. $P : S \times A \times S \rightarrow [0, 1]$ is the transition probability function (where $P(s' | s, a)$ is the probability of transitioning to state s' under the previous state s and the action a). $\gamma \in [0, 1]$ is the discount factor for future rewards, which represents a trade-off between immediate and future rewards. π refers to a policy, which is a function that maps a state to an action. $\pi(a|s)$ denotes the

probability of selecting action a in state s . A trajectory τ is a set of MDP tuples in one episode during which an agent interacts with the environment under policy π .

A constrained Markov decision process (CMDP) is an MDP subjected to some constraints. A set C_1, \dots, C_m (with each one as $C_i : S \times A \times S \rightarrow \mathbb{R}$) is referred to as cost functions, similar to the reward function [28]. The limits d_1, \dots, d_m represent the thresholds of the cost functions. For the policy gradient algorithm, value functions, action-value functions, and advantage functions are defined as V^π, Q^π , and A^π .

$$\begin{aligned}
 Q^\pi(s_t, a_t) &= \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k} \right] \\
 V^\pi(s_t) &= \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k} \right] \\
 A^\pi(s, a) &= Q^\pi(s, a) - V^\pi(s)
 \end{aligned}
 \tag{20}$$

Similarly, we can replace reward R in the above equation with cost C to acquire V_C^π, Q_C^π , and A_C^π .

In a CMDP, the actor-critic framework consists of three parts: the actor, the critic, and the cost critic. The actor is responsible for selecting actions based on the current state of the agent, while the critic and cost critic estimate the expected discounted reward and cost based on the current state. Typically, these three parts are usually approximated by neural networks (NN). In the case of CMDP, let $J(\pi)$ denote the expected discounted reward:

$$J(\pi) = \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{\infty} \gamma^t R_t \right]
 \tag{21}$$

Similarly, the cumulative cost can be described as

$$J_{C_i}(\pi) = \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{\infty} \gamma^t C_i(s_t, a_t, s_{t+1}) \right]
 \tag{22}$$

Thus, the set of feasible policies is

$$\Pi_C \doteq \{ \pi \in \Pi : \forall i, J_{C_i}(\pi) \leq d_i \}
 \tag{23}$$

The objective in a CMDP is to find a policy that maximizes $J(\pi)$ in Π_C :

$$\pi^* = \arg \max_{\pi \in \Pi_C} J(\pi)
 \tag{24}$$

3.2.2. Constrained Policy Optimization

CPO adheres to the monotonic improvement theory proposed by Trust Region Policy Optimization (TRPO) [29]. By constraining the difference between the old and new strategies to a small step size, it becomes possible to approximate the lower bound of the objective function and the upper bound of the cost objective function between the old and new policies, resulting in the following expression [27]:

$$J(\pi') - J(\pi) \approx \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{\infty} A^\pi(s_t, a_t) \frac{\pi'(a_t|s_t)}{\pi(a_t|s_t)} \right]
 \tag{25}$$

$$J_{C_i}(\pi') - J_{C_i}(\pi) \approx \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{\infty} A_{C_i}^\pi(s_t, a_t) \frac{\pi'(a_t|s_t)}{\pi(a_t|s_t)} \right]
 \tag{26}$$

Due to the approximation of the difference and the requirement in importance sampling that the two strategy distributions must not be too far apart, a constraint is added on the average Kullback–Leibler (KL) divergence between the old and new policies:

$$\bar{D}_{KL}(\pi\|\pi') \leq \delta \tag{27}$$

where δ is a step size. By employing the minorization-maximization (MM) algorithm, policy boosting can be guaranteed through maximizing an approximate lower bound on the difference between the old and new policies (often called the alternative objective function). Similarly, if the sum of the upper bound of the difference between the cost function of the old and the new strategy and the cost function of the old strategy satisfies the constraint, then the cost objective function of the new policy can also satisfy the constraint. As a result, this optimization problem can be expressed as

$$\begin{aligned} & \max_{\theta} \mathbb{E}_{(s_t, a_t) \sim \pi} \left[\frac{\pi_{\theta}(a_t | s_t)}{\pi(a_t | s_t)} \hat{A}^{\pi}(s_t, a_t) \right] \\ \text{s.t. } & J_{C_i}(\pi) + \mathbb{E}_{(s_t, a_t) \sim \pi} \left[\frac{\pi_{\theta}(a_t | s_t)}{\pi(a_t | s_t)} \hat{A}_{C_i}^{\pi}(s_t, a_t) \right] \leq d_i \quad \forall i \\ & \bar{D}_{KL}(\pi\|\pi^{\theta}) \leq \delta \end{aligned} \tag{28}$$

where θ is the parameter of the policy network and \hat{A} is the advantage value function estimated by generalized advantage estimation [30]. Since the KL divergence limits the difference in the policy distribution to a small step size, the objective and constraint functions in the above equation can be expanded in the first order, while the KL divergence constraint is expanded in the second order. Let the gradient of the objective function be g , the gradient of constraint i be b_i , the Hessian of KL divergence be H , and c be defined as $J_{C_i}(\pi) - d_i$. Thus, the approximate problem is

$$\begin{aligned} & \theta_{k+1} = \arg \max_{\theta} g^T(\theta - \theta_k) \\ \text{s.t. } & c_i + b_i^T(\theta - \theta_k) \leq 0 \quad i = 1, \dots, m \\ & \frac{1}{2}(\theta - \theta_k)^T H(\theta - \theta_k) \leq \delta \end{aligned} \tag{29}$$

Due to the computational complexity of the Hessian matrix, CPO uses a Fisher information matrix to approximate H , which makes the problem above a convex optimization problem [27]. To solve this convex optimization problem, we denote the Lagrangian multipliers λ and ν . Then, when the original problem has a feasible solution, a dual to (29) can be expressed as

$$\max_{\substack{\lambda \geq 0 \\ \nu \geq 0}} \frac{-1}{2\lambda} \left(g^T H^{-1} g - 2\nu^T b^T H^{-1} g + \nu^T b^T H^{-1} b \nu \right) + \nu^T c - \lambda \delta \tag{30}$$

If λ^* and ν^* are the optimal solution to the dual problem, the policy can be updated as follows:

$$\theta^{k+1} = \theta^k + \frac{1}{\lambda^*} H^{-1} (g - b \nu^*) \tag{31}$$

When there is no feasible solution to the original problem, CPO will execute the recovery method, which will change the search direction. The recovery method is as follows:

$$\theta^{k+1} = \theta^k - \sqrt{\frac{2\delta}{b^T H^{-1} b}} H^{-1} b \tag{32}$$

Inspired by TRPO, the line search method is used to adjust the step size of the update to avoid any error caused by the approximation.

3.2.3. Implementation Details

In this section, we give the corresponding design of the state space, action space, and reward function in RL. The state and action spaces are described as

$$s = (\sin \theta, \cos \theta, v_x, v_z, q, t_{left}, I_{v_x}, I_{v_z}, I_\theta, z)^T \in R^{10}, \tag{33}$$

$$a = (\delta_y, \omega_r)^T \in R^2.$$

where δ_y represents the vane deflection of pitch, ω_r is the angular velocity of the rotor, θ is the pitch angle, v_x and v_z are the velocity of the X-axis and Z-axis in the inertial frame, q is the angular velocity of the pitch angle, t_{left} is the remaining time during one episode (the maximum time for an episode is 10 s), and z is the height. $I_{v_x}, I_{v_z}, I_\theta$ is the integral error of v_x, v_z, θ , which can be expressed as

$$e_\theta = \theta - \pi/2, e_{v_x} = v_x, e_{v_z} = v_z \tag{34}$$

$$I_n = \eta I_{n-1} + e_n \tag{35}$$

where η is the integral coefficient, and the integration of the error is approximated by such an incremental approach as Equation (35). The design of the reward function is a key factor affecting the performance of the reinforcement learning algorithm. The reward function R and the cost function C are given by the following:

$$\Phi(s_t) = k_1 \sqrt{v_x^2 + v_z^2} + k_2 |\theta - \pi/2| \tag{36}$$

$$R = \kappa + (\Phi(s_{t+1}) - \Phi(s_t)) \times (0.5)^{t/10} - \chi I(s) - \beta(q > \pi) - \xi(|z| > 2) - \tau \tag{37}$$

$$C = 1 \text{ if } \|v_z\| \geq 1 \text{ else } 0 \tag{38}$$

where the various terms are described as follows.

1. κ is a bonus reward for successful transition, where the terminal velocity, pitch angle, and angular velocity are all within specified limits. In addition, different amounts of bonuses are also given based on the loss of height when a successful transition state is reached. We use Z_T to represent the terminal height loss.

$$\kappa = \begin{cases} 1000 & \text{if } Z_T \leq 0.125 \text{ m} \\ 100 - 100 \times \log_2 Z_T & \text{if } Z_T \leq 1 \text{ m} \\ 100 - 90 \times \log_2 Z_T & \text{if } Z_T \leq 2 \text{ m} \\ 10 & \text{else} \end{cases} \tag{39}$$

2. $\Phi(s_{t+1}) - \Phi(s_t)$ is in the form of reward shaping. The aim of reward shaping is to accelerate the learning process, where k_1 and k_2 denote the weights of the speed and angle rewards. The term $(0.5)^{t/10}$, which is gradually decaying over time, is designed to reduce the transition time.
3. χ is constant and $I(s) = \|0.02 \times I_{v_x}^2 + I_\theta^2 + I_{v_z}^2\|$, β is a penalty for an angle exceeding the limit, and ξ is a penalty when the UAV loses more than two meters in altitude. τ is a fixed time penalty.

The cost limit d is set at 5. The terminal condition of an episode is either meeting the termination condition specified in Section 3.1 or reaching the maximum time of 10 s.

4. Simulation and Results

In this section, we present the experimental results. In our experiments, the controller update and sensor data download frequency was set as 50 HZ. For our experiments, the initial flight mode was set to level flight, with the initial angle of attack sampled from a uniform distribution ranging between 6° and 15° . The initial pitch angle was set equal to the angle of attack, the initial height was set at 50 m, and the corresponding horizontal velocity could be determined using the following equation:

$$v_{level} = -\sqrt{\frac{mg}{\frac{1}{2}C_L(\alpha)\rho_\infty S}} \quad (40)$$

4.1. Comparing CPO and TRPO with Fixed Penalty

To demonstrate the superiority of safe reinforcement learning compared to other conventional reinforcement learning algorithms with a fixed penalty, we conducted three different sets of experiments, which were named CPO, TRPO with penalty 1, and TRPO with penalty 5. The experimental results are shown in Figure 7.

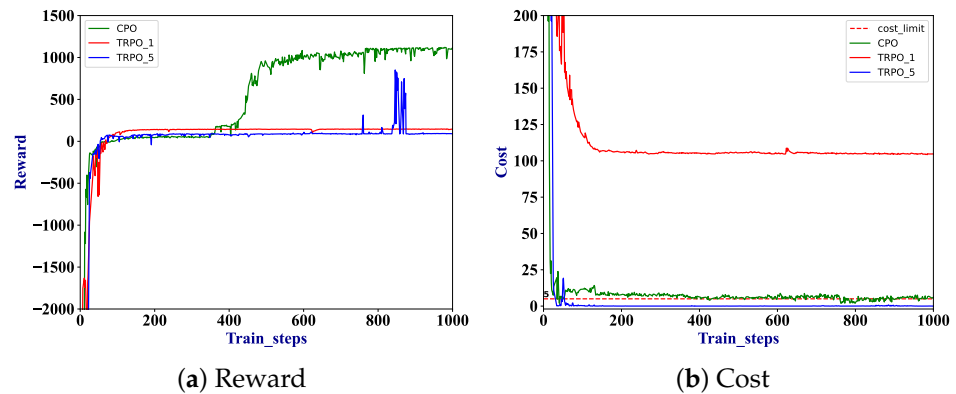


Figure 7. Comparison of learning curve.

During training, we took into account the randomization of the initial state and the perturbation of the aerodynamic coefficients. However, in evaluating the transition performance of CPO and TRPO with penalty, we have not considered these factors. Three sets of experiments with the same initial state (angle of attack at 6°) for the sake of comparison are shown in Figure 8, where the arrows represent the terminal state.

In Figures 7 and 8, we can observe that the trajectory corresponding to TRPO with penalty 1 converges to a locally optimal continuous ascending path, resulting in a fast transition time but significant altitude loss. In contrast, the trajectory corresponding to TRPO with penalty 5 consistently satisfies the constraints, but its terminal angle and speed exceed the predefined range, ultimately failing to complete the task. The CPO algorithm, however, effectively meets the constraints while accomplishing the transition with a minimal altitude loss of 0.1 m, which complies with the requirements of a neat transition.

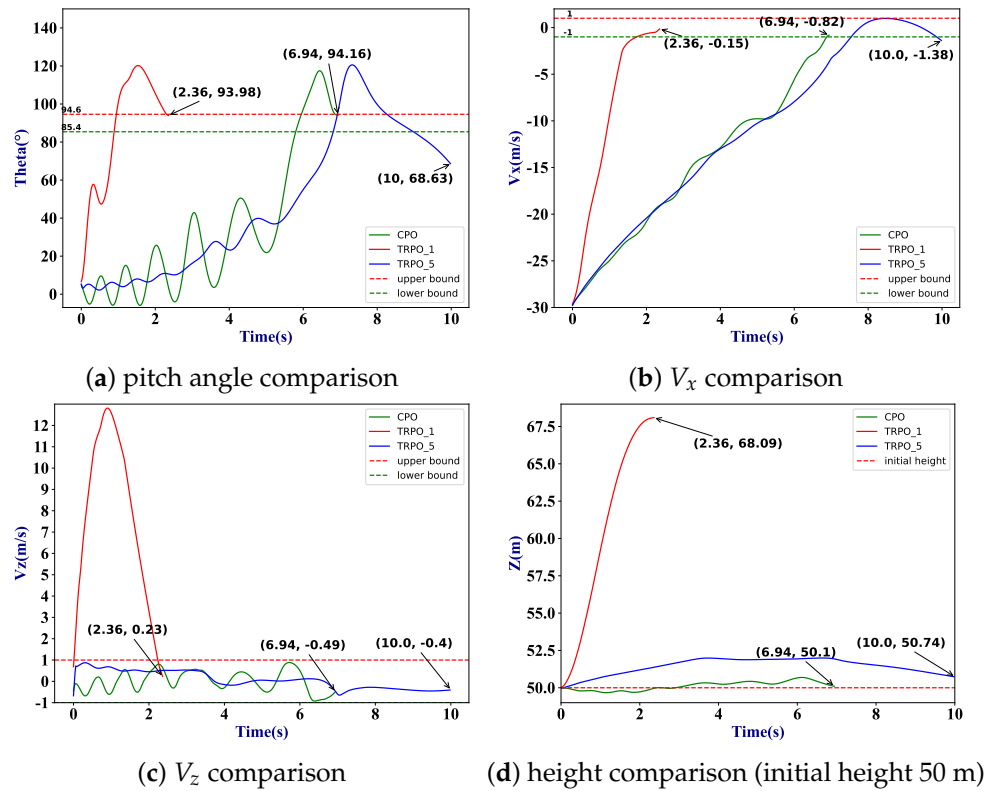


Figure 8. Trajectory comparison.

4.2. Comparing CPO and PPOLag

In the domain of safe reinforcement learning, numerous algorithms can address constraint-related issues. For the sake of code development convenience, we have chosen to compare the CPO algorithm with Proximal Policy Optimization with Lagrangian, also called PPOLag (see Figure 9).

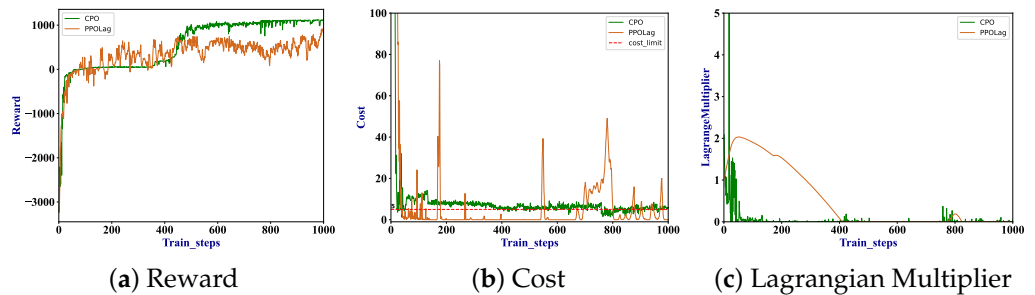


Figure 9. CPO vs. PPOLag.

PPOLag is a variant of the Proximal Policy Optimization (PPO) algorithm, which employs a Lagrangian relaxation approach to handle constraints. Constraints are integrated into the objective function using a penalty term that is computed with the help of the Lagrange multiplier. The Lagrange multiplier is updated during the training process based on the discrepancy between the current constraint value and its cost limit, which serves as a weighting factor to balance the trade-off between the reward function and the constraints.

PPOLag is a simpler algorithm in principle and easier to implement. However, as shown in Figure 9b, this method may cause oscillations near the cost limit of the constraint, leading to poor performance of the agent (as shown in Figure 9a). Consequently, although PPOLag can complete the transition, its performance is inferior to that of CPO. The best-trained model with PPOLag (at an angle of attack of 6°) has a terminal height loss of 1.2 m

and a transition time of 7.84 s, but CPO has a terminal height loss of 0.1 m and a transition time of 6.94 s.

4.3. Comparison with GPOPS-II

According to the problem formulation in Section 3.1, this problem can also be considered as an optimal control problem. As a result, we choose to employ GPOPS-II when computing the optimal trajectory. In this case, the control variables for GPOPS-II are the same as in our approach—namely, the angular velocity of the rotor and elevator deflection. The experimental results without perturbations at an initial angle of attack 10° are compared in Figures 10 and 11. The cost function of GPOPS-II is the following:

$$J = 0.5t_f + 0.2|Z_{t_f}| \tag{41}$$

where t_f is the transition time and Z_{t_f} is the terminal height loss.

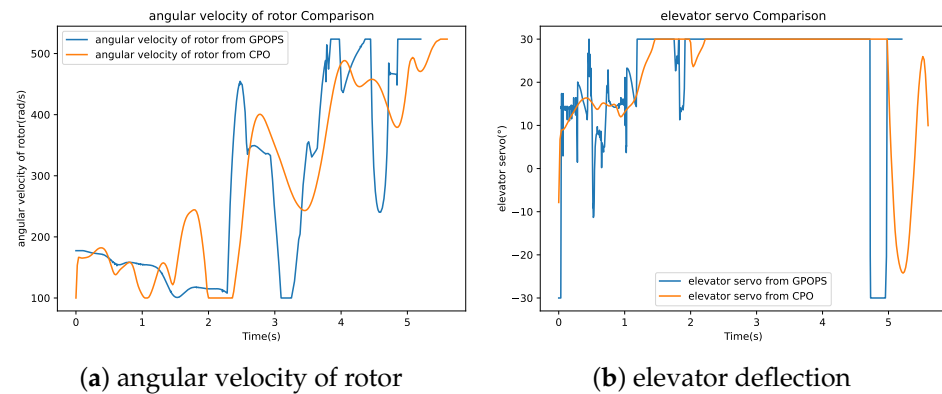


Figure 10. Control input comparison for CPO and GPOPS-II.

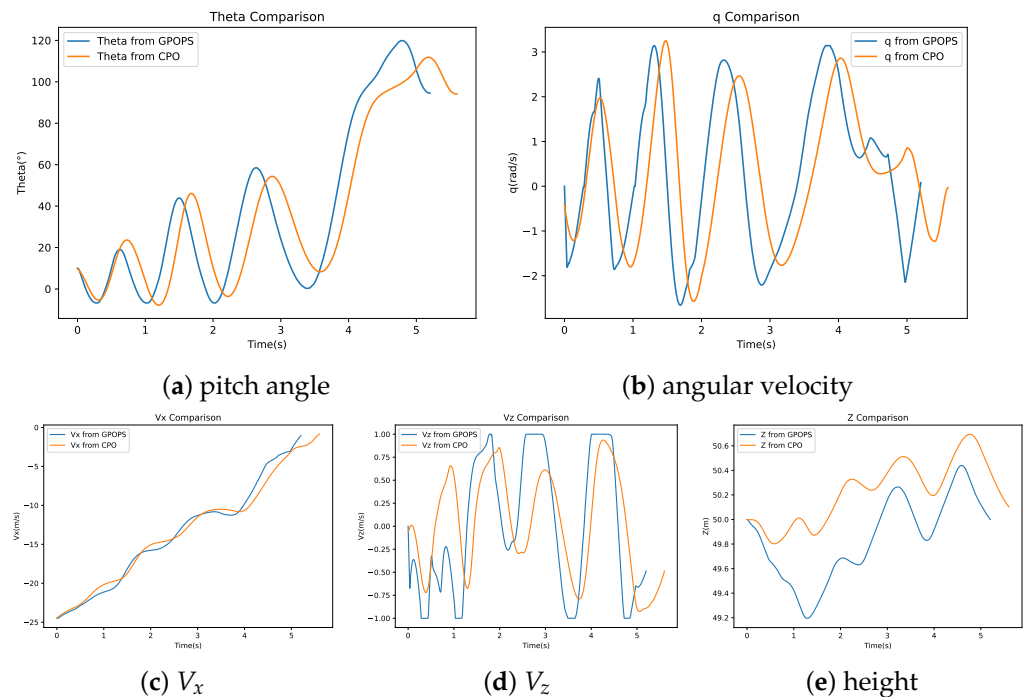


Figure 11. State comparison for CPO and GPOPS-II.

The height loss of GPOPS-II is 0.01 m with a transition time of 5.21 s, while the height loss of CPO is 0.103 m with a transition time of 5.62 s. As can be seen from the figures, the performance of the CPO algorithm closely approximates that of GPOPS-II. However, GPOPS-II has two main drawbacks. Firstly, as a model-based approach, it provides only optimal feed-forward control input, which is an ideal solution. When the dynamics model of the UAV changes, it requires re-solving based on the altered model. In contrast, the CPO algorithm is highly robust and can still perform the transition task with high performance despite certain modeling errors and wind disturbance, as discussed later in Sections 4.4 and 4.5. Secondly, as micro-controllers onboard aerial robots generally only have limited computational power, the optimization of GPOPS-II can be only executed offline. However, RL algorithms such as CPO can solve the transition problem online after a policy model has been trained, which saves a great deal of computational resources.

4.4. Robustness Validation

During the training process, the randomness of the UAV is only determined by the randomness of the initial state and aerodynamic parameters. Therefore, in order to assess the robustness of the system, we chose to randomize various factors, including the mass, moment of inertia, and aerodynamic parameters of lift and drag (with a larger range), by employing a uniform distribution. Furthermore, we took sensor noise into account because measurement errors in the height can potentially lead to a significant decline in the system's performance, especially for terminal height loss. To validate the robustness of sensor noise, Gaussian noise was introduced into the height measurements. It is important to note that we did not consider these perturbations during training, meaning that the UAV had not been exposed to them before.

To enable a fair comparison, we only randomized the initial state and tested each perturbation individually. Specifically, we conducted 500 experiments for each perturbation, while maintaining the same transition success condition as in Section 3.1. We report the average performance of the UAV across all experiments (see Table 1). At the same time, the trajectories under each disturbance group (50 in total) are also provided. We chose to use the pitch angle and velocity of the UAV during the transition process to draw 2D plane curves (see Figures 12–15), where the different color simply indicates that these trajectories are different. Through the figures, we can find that the UAV can still complete the transition task with excellent performance despite the change in UAV model parameters. At the same time, we can observe that even when the UAV does not transition successfully, its terminal velocity and pitch angle mostly remain close to our desired range.

Table 1. Experimental results under different perturbations.

| Perturbation Type | Parameter Range | Success Rate | Transition Time | Terminal Height Loss | Terminal Velocity | Terminal Pitch Angle |
|-------------------|-----------------------------------------------|--------------|-----------------|----------------------|-------------------|----------------------|
| Mass | 21% | 94% | 6.30 s | 0.27 m | 1.01 m/s | 95.09° |
| Lift and Drag | 40% | 97.0% | 6.09 s | 0.109 m | 0.966 m/s | 93.76° |
| Inertia | 40% | 100% | 6.02 s | 0.072 m | 0.80 m/s | 93.42° |
| Sensor Noise | $ \mu \leq 0.2, 0.005 \leq \sigma \leq 0.05$ | 100% | 6.54 s | 0.133 m | 0.95 m/s | 93.05° |

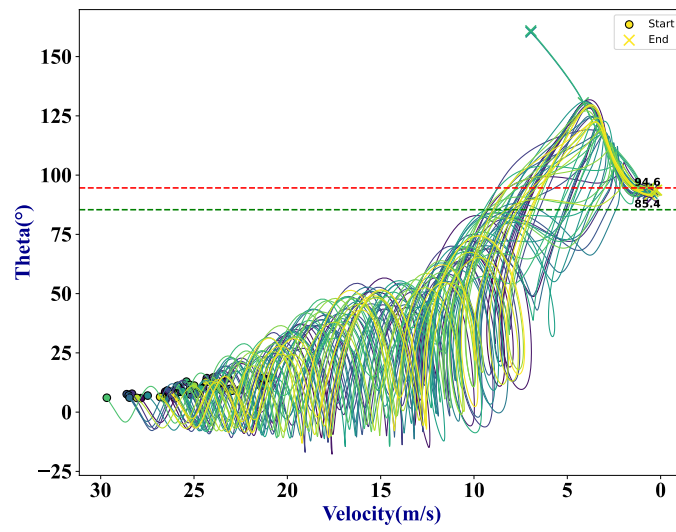


Figure 12. Mass perturbation.

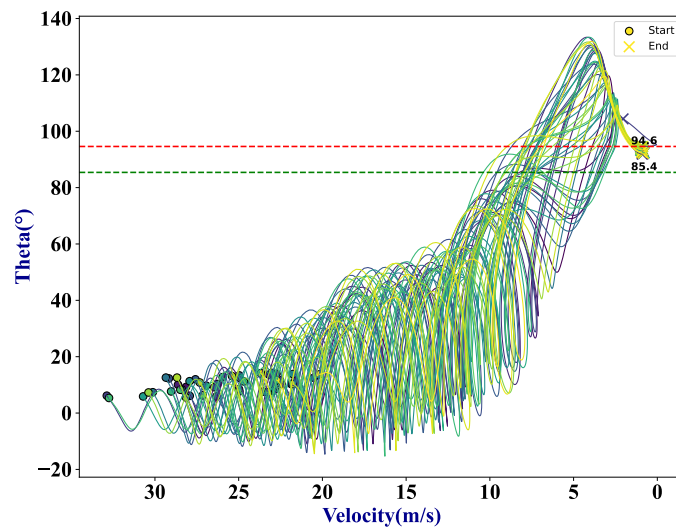


Figure 13. Lift and drag perturbation.

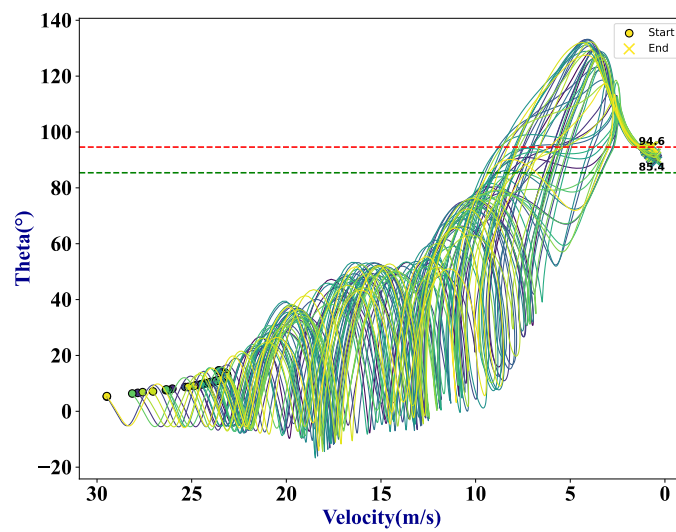


Figure 14. Inertia perturbation.

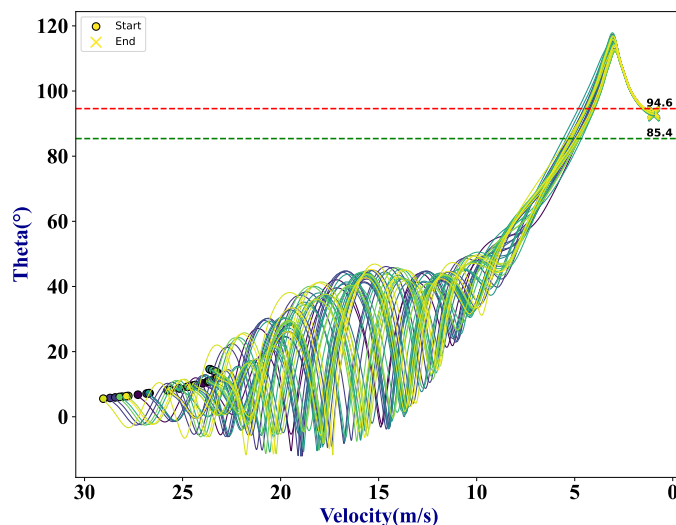


Figure 15. Sensor noise perturbation.

4.5. Transition under Wind Disturbance

In this section, we aim to verify the wind disturbance rejection ability of our method. In RL-based robotic control design, the sim-to-real gap is a challenging problem because there are inevitable mismatches between the simulator setting and real-world setting in the UAV control problem. This is not only due to the unknown perturbations of the UAV parameters but also the presence of adversarial disturbances such as wind in the real world. To account for wind disturbance, we introduce two different wind scenarios: constant-magnitude wind (ranging from -5 m/s to 5 m/s) and sinusoidal wind (with an amplitude of 5 m/s and time periods of 2 s, 3 s, 4 s, and 5 s), both along the horizontal direction, which is the X^l axis. Furthermore, the dynamics of ducted-fan fixed-wing UAVs in the presence of wind are described in detail in [3,14]. Therefore, we recreate the dynamic equations of the UAV using their modeling of wind.

To address the wind disturbance, we employ domain randomization [31] and retrain the RL agent under three different conditions: the two wind scenarios mentioned above and a no-wind scenario. Domain randomization is an approach used to overcome the sim-to-real gap by randomizing the simulator environment during the training of the RL control policy. By adapting the domain randomization approach, we aim to retrain a UAV controller that is robust against wind disturbance.

The trajectory comparison among the constant-magnitude wind (5 m/s), sinusoidal wind (time period 2 s), and no-wind conditions is shown in Figure 16, where the arrows represent the terminal state and all initialized from the angle of attack 6° .

From Figures 8 and 16, we can see that compared to the controller trained without domain randomization, the terminal height loss of the drone retrained with domain randomization remains nearly the same, while the time taken increases by 1.52 s. This phenomenon can be intuitively understood as follows: the RL algorithm selects actions with the highest expected returns under various wind disturbance scenarios, rather than actions that can achieve high returns in windless environments, but may fail to complete the task in windy conditions. Thus, a certain degree of performance loss is reasonable.

In order to better evaluate the resistance of our method to wind disturbances, we applied the UAV in 100 experiments under three conditions and from different initial states, in addition to considering the perturbation of the aerodynamic coefficients during the transition (a uniform distribution ranging between 0.8 and 1.2). The experimental results are shown in Figure 17 (success rate 90%), where the different color simply indicates that these trajectories are different.

In Figure 17, we observe that in most cases, the UAV is able to withstand wind disturbances and complete the transition task. However, for those trajectories that do not

satisfy the terminal constraint in Section 3.1, we find that the trajectories generally terminate with an attitude close to vertical hover (90°) and a small velocity. This suggests that the UAV is less resistant to interference in the hovering state. In conventional approaches, a common method is to switch to the hover controller when the UAV transitions from level flight to near-hover. Wind disturbance can be resisted by switching between the two controllers and enhancing the hover controller’s ability to handle wind disturbance. In RL, resistance to wind disturbance in hovering should be treated as a separate task, and the problem should be addressed using multi-task RL or meta-learning, which will be the focus of our future work.

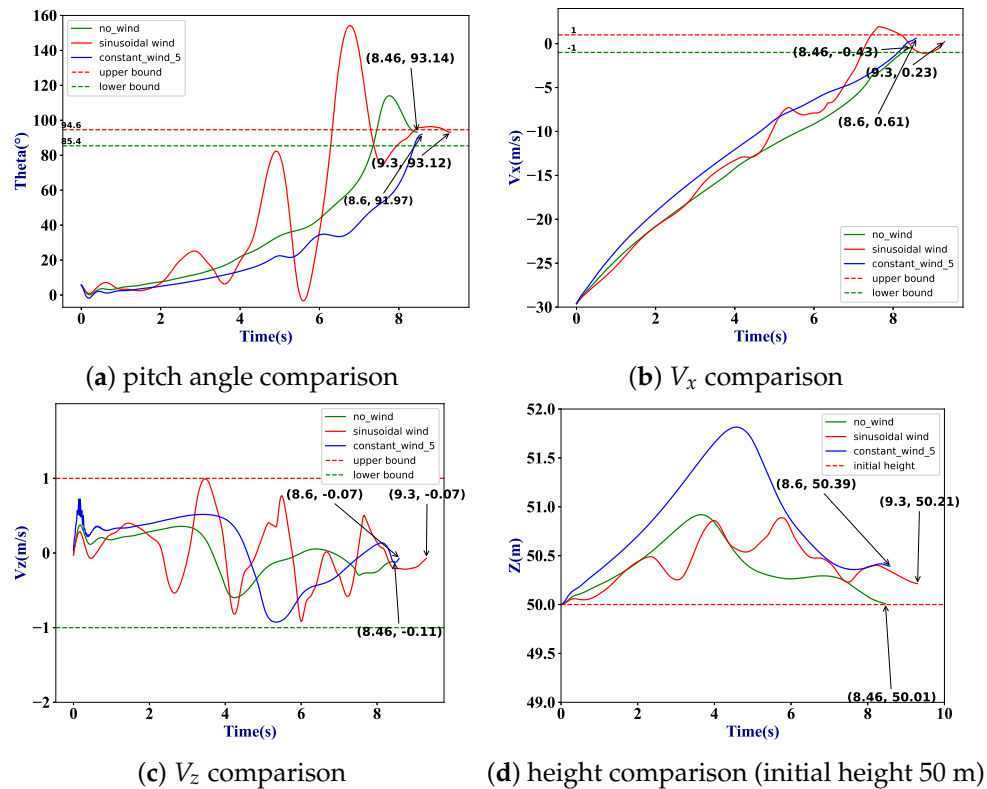


Figure 16. Trajectory comparison under wind disturbance.

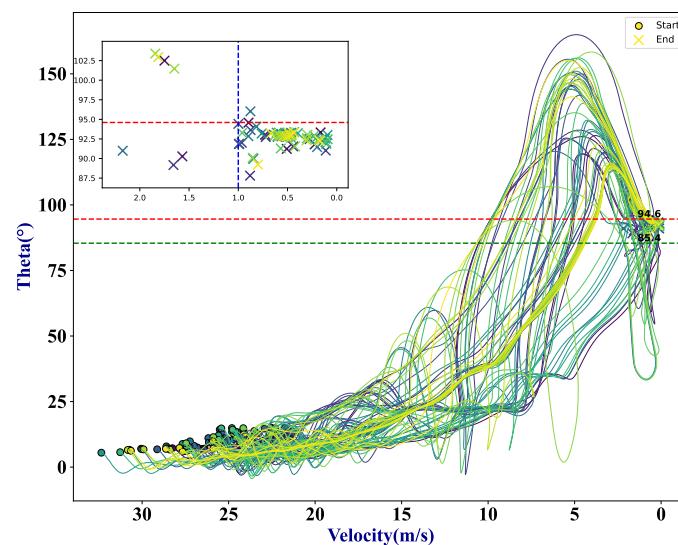


Figure 17. Trajectory under wind disturbance.

5. Conclusions

In this study, we have successfully developed a safe reinforcement learning-based approach for neat transition control during the back-transition process of ducted-fan fixed-wing UAVs. By constructing a three-degree-of-freedom longitudinal model and implementing the CPO algorithm, our method effectively addresses the challenges of integrating trajectory optimization and control methods. By comparison, we found that the introduction of a velocity constraint leads to better performance compared to adding a penalty to the reward. Furthermore, we also found that our method closely resembles GPOPS-II's performance without the need for prior knowledge. Additionally, we confirmed the robustness of the CPO algorithm and found that even when the transition was not successful, the terminal conditions remained close to our desired terminal range. Future research directions include enabling the UAV to complete multi-tasks (i.e., from hover to level flight, hover under wind disturbance, and level flight), ensuring robustness against wind disturbance, and validating the approach in the real world.

Author Contributions: Conceptualization, Y.F. and W.Z.; Data curation, Y.F. and L.L.; Formal analysis, Y.F.; Funding acquisition, W.Z.; Investigation, Y.F. and W.Z.; Methodology, Y.F.; Project administration, W.Z.; Resources, W.Z. and L.L.; Software, Y.F.; Supervision, W.Z.; Validation, Y.F.; Visualization, Y.F. and L.L.; Writing—original draft, Y.F.; Writing—review and editing, Y.F. and W.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the 1912 project, the Key Research and Development Program of Zhejiang Province, China (Grant No. 2020C05001), and the Fundamental Research Funds for the Central Universities, China (Grant No. 2021QNA4030).

Data Availability Statement: The data presented in this study are available on request.

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

Nomenclature

| Symbol | Description | Units |
|-----------------------------|-----------------------------------------------------------------|---------------------|
| x, y, z | position of UAV | m |
| u, v, w | velocity in the body frame | m/s |
| ϕ, θ, ψ | roll, pitch, and yaw angle | rad |
| p, q, r | angular velocity | rad/s |
| g | gravity acceleration | m/s ² |
| α | angle of attack | rad |
| m | mass | kg |
| I_{yy} | moment of inertia | kg · m ² |
| L, D, T | aerodynamic lift, drag, and thrust | N |
| F_{cw}, F_{cs} | momentum drag and control vane force | N |
| $M_{pitch}, M_{cw}, M_{cs}$ | pitch, momentum, and control vane moment | N · m |
| ρ_{∞} | air density | Kg/m ³ |
| ω_r | angular velocity of the rotor | rad/s |
| r | radius of the rotor | m |
| K_{twist} | twist of blades | |
| a_0 | rotor lift curve slope | |
| b | number of blades | |
| c_r | chord of the rotor blade | m |
| v_b, v_i, v_f | airflow through rotor, induced velocity, and far-field velocity | m/s |
| C_L, C_D, C_m | lift, drag, and pitch moment coefficient | |
| V | air speed | m/s |
| C_{lcs} | lift slope coefficient of vane | |

| | | |
|----------|----------------------------|--------------------------------|
| q_{cs} | dynamic pressure | $\text{kg/m} \cdot \text{s}^2$ |
| S_{cs} | vane area | m^2 |
| l_e | arm of pitch moment | m |
| s | state space | |
| R | reward function | |
| C | cost function | |
| a | action space | |
| P | transition probability | |
| γ | discount factor | |
| V^π | value function | |
| A^π | advantage value function | |
| $J(\pi)$ | expected discounted reward | |

References

- Ozdemir, U.; Aktas, Y.O.; Vuruskan, A.; Dereli, Y.; Tarhan, A.F.; Demirbag, K.; Erdem, A.; Kalaycioglu, G.D.; Ozkol, I.; Inalhan, G. Design of a commercial hybrid VTOL UAV system. *J. Intell. Robot. Syst.* **2014**, *74*, 371–393. [\[CrossRef\]](#)
- Okulski, M.; Ławryńczuk, M. A Small UAV Optimized for Efficient Long-Range and VTOL Missions: An Experimental Tandem-Wing Quadplane Drone. *Appl. Sci.* **2022**, *12*, 7059. [\[CrossRef\]](#)
- Argyle, M.E. Modeling and Control of a Tailsitter with a Ducted Fan. Ph.D. Thesis, Ira A. Fulton College of Engineering and Technology, Provo, UT, USA, 2016.
- Graf, W.E. Effects of Duct Lip Shaping and Various Control Devices on the Hover and Forward Flight Performance of Ducted Fan UAVs. Ph.D. Thesis, Virginia Tech, Blacksburg, VA, USA, 2005.
- Oosedo, A.; Abiko, S.; Konno, A.; Uchiyama, M. Optimal transition from hovering to level-flight of a quadrotor tail-sitter UAV. *Auton. Robot.* **2017**, *41*, 1143–1159. [\[CrossRef\]](#)
- Li, B.; Sun, J.; Zhou, W.; Wen, C.Y.; Low, K.H.; Chen, C.K. Transition optimization for a VTOL tail-sitter UAV. *IEEE/ASME Trans. Mechatronics* **2020**, *25*, 2534–2545. [\[CrossRef\]](#)
- Verling, S.; Stastny, T.; Bättig, G.; Alexis, K.; Siegwart, R. Model-based transition optimization for a VTOL tailsitter. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 3939–3944.
- Kubo, D.; Suzuki, S. Tail-sitter vertical takeoff and landing unmanned aerial vehicle: transitional flight analysis. *J. Aircr.* **2008**, *45*, 292–297. [\[CrossRef\]](#)
- Banazadeh, A.; Taymourtash, N. Optimal control of an aerial tail sitter in transition flight phases. *J. Aircr.* **2016**, *53*, 914–921. [\[CrossRef\]](#)
- Naldi, R.; Marconi, L. Optimal transition maneuvers for a class of V/STOL aircraft. *Automatica* **2011**, *47*, 870–879. [\[CrossRef\]](#)
- Jeong, Y.; Shim, D.; Ananthkrishnan, N. Transition Control of Near-Hover to Cruise Transition of a Tail Sitter UAV. In Proceedings of the AIAA Atmospheric Flight Mechanics Conference, Toronto, ON, Canada, 2–5 August 2010; p. 7508.
- Flores, A.; Flores, G. Transition control of a tail-sitter UAV using recurrent neural networks. In Proceedings of the 2020 International Conference on Unmanned Aircraft Systems (ICUAS), Athens, Greece, 1–4 September 2020; pp. 303–309.
- Cheng, Z.H.; Pei, H.L. Transition analysis and practical flight control for ducted fan fixed-wing aerial robot: Level path flight mode transition. *IEEE Robot. Autom. Lett.* **2022**, *7*, 3106–3113. [\[CrossRef\]](#)
- Cheng, Z.; Pei, H.; Li, S. Neural-networks control for hover to high-speed-level-flight transition of ducted fan uav with provable stability. *IEEE Access* **2020**, *8*, 100135–100151. [\[CrossRef\]](#)
- Zhang, W.; Quan, Q.; Zhang, R.; Cai, K.Y. New transition method of a ducted-fan unmanned aerial vehicle. *J. Aircr.* **2013**, *50*, 1131–1140. [\[CrossRef\]](#)
- Xu, J.; Du, T.; Foshey, M.; Li, B.; Zhu, B.; Schulz, A.; Matusik, W. Learning to fly: Computational controller design for hybrid uavs with reinforcement learning. *ACM Trans. Graph. (TOG)* **2019**, *38*, 1–12. [\[CrossRef\]](#)
- Xu, X.; Chen, Y.; Bai, C. Deep reinforcement learning-based accurate control of planetary soft landing. *Sensors* **2021**, *21*, 8161. [\[CrossRef\]](#) [\[PubMed\]](#)
- Yukse, B.; Inalhan, G. Transition Flight Control System Design for Fixed-Wing VTOL UAV: A Reinforcement Learning Approach. In Proceedings of the AIAA SCITECH 2022 Forum, San Diego, CA, USA, 3–7 January 2022; p. 0879.
- Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; Zaremba, W. Openai gym. *arXiv* **2016**, arXiv:1606.01540.
- Patterson, M.A.; Rao, A.V. GPOPS-II: A MATLAB software for solving multiple-phase optimal control problems using hp-adaptive Gaussian quadrature collocation methods and sparse nonlinear programming. *ACM Trans. Math. Softw. (TOMS)* **2014**, *41*, 1–37. [\[CrossRef\]](#)
- Gaudet, B.; Linares, R.; Furfaro, R. Deep reinforcement learning for six degree-of-freedom planetary landing. *Adv. Space Res.* **2020**, *65*, 1723–1741. [\[CrossRef\]](#)
- Johnson, E.N.; Turbe, M.A. Modeling, control, and flight testing of a small-ducted fan aircraft. *J. Guid. Control Dyn.* **2006**, *29*, 769–779. [\[CrossRef\]](#)

23. Heffley, R.K.; Mnich, M.A. Minimum-complexity helicopter simulation math model. Technical Report; Manudyne Systems, Inc.: Los Altos, CA, USA, 1988.
24. Beard, R.W.; McLain, T.W. *Small Unmanned Aircraft: Theory and Practice*; Princeton University Press: Princeton, NJ, USA, 2012.
25. Puopolo, M.; Reynolds, R.; Jacob, J. Comparison of three aerodynamic models used in simulation of a high angle of attack UAV perching maneuver. In Proceedings of the 51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition, Grapevine, TX, USA, 7–10 January 2013; p. 242.
26. Kikumoto, C.; Urakubo, T.; Sabe, K.; Hazama, Y. Back-Transition Control with Large Deceleration for a Dual Propulsion VTOL UAV Based on Its Maneuverability. *IEEE Robot. Autom. Lett.* **2022**, *7*, 11697–11704. [[CrossRef](#)]
27. Achiam, J.; Held, D.; Tamar, A.; Abbeel, P. Constrained policy optimization. In Proceedings of the International Conference on Machine Learning, PMLR, Sydney, Australia, 6–11 August 2017; pp. 22–31.
28. Yang, T.Y.; Rosca, J.; Narasimhan, K.; Ramadge, P.J. Projection-based constrained policy optimization. *arXiv* **2020**, arXiv:2010.03152.
29. Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; Moritz, P. Trust region policy optimization. In Proceedings of the International Conference on Machine Learning PMLR, Lille, France, 6–11 July 2015; pp. 1889–1897.
30. Schulman, J.; Moritz, P.; Levine, S.; Jordan, M.; Abbeel, P. High-dimensional continuous control using generalized advantage estimation. *arXiv* **2015**, arXiv:1506.02438.
31. Tobin, J.; Fong, R.; Ray, A.; Schneider, J.; Zaremba, W.; Abbeel, P. Domain randomization for transferring deep neural networks from simulation to the real world. In Proceedings of the 2017 IEEE/RSJ international conference on intelligent robots and systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; pp. 23–30.

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.