drones

Article

Fast Tube-Based Robust Compensation Control for Fixed-Wing UAVs

Lixin Wang [1], Sizhuang Zheng [1,*], Weijia Wang [2], Hao Wang [2], Hailiang Liu [1] and Ting Yue [1,*]

[1] School of Aeronautical Science and Engineering, Beihang University, Beijing 100191, China
[2] Flight Automatic Control Research Institute, AVIC, Xi'an 710065, China
* Correspondence: zhengsz@buaa.edu.cn (S.Z.); yueting_buaa@163.com (T.Y.)

Abstract: When considering the robust control of fixed-wing Unmanned Aerial Vehicles (UAVs), a conflict often arises between addressing nonlinearity and meeting fast-solving requirements. In existing studies, the less nonlinear robust control methods have shown significant improvements that parallel computing and dimensionality reduction techniques in real-time applications. In this paper, a nonlinear fast Tube-based Robust Compensation Control (TRCC) for fixed-wing UAVs is proposed to satisfy robustness and fast-solving requirements. Firstly, a solving method for discrete trajectory tubes was proposed to facilitate fast parallel computation. Subsequently, a TRCC algorithm was developed that minimized the trajectory tube to enhance robustness. Additionally, considering the characteristics of fixed-wing UAVs, dimensionality reduction techniques such as decoupling and stepwise approaches are proposed, and a fast TRCC algorithm that incorporates the control reuse method is presented. Finally, simulations verify that the proposed fast TRCC effectively enhances the robustness of UAVs during tracking tasks while satisfying the requirements for fast solving.

Keywords: unmanned aerial vehicles; nonlinear robust control; robust compensation control; trajectory tube; fast robust control

Citation: Wang, L.; Zheng, S.; Wang, W.; Wang, H.; Liu, H.; Yue, T. Fast Tube-Based Robust Compensation Control for Fixed-Wing UAVs. *Drones* **2023**, *7*, 481. https://doi.org/10.3390/drones7070481

Academic Editor: Mostafa Hassanalian

Received: 6 June 2023
Revised: 18 July 2023
Accepted: 20 July 2023
Published: 21 July 2023

1. Introduction

Compared to manned aircraft, fixed-wing Unmanned Aerial Vehicles (UAVs), renowned for their high agility and maneuverability, are often required to perform intense maneuvers, leading to nonlinearity in their dynamic characteristics [1]. Additionally, in trajectory tracking control tasks, fixed-wing UAVs may encounter uncertainties such as external disturbances and model perturbations. Thus, the tracking control system needs to handle robust tracking control in the face of strong nonlinearity. In addition, the UAV tracking control must also satisfy fast-solving requirements for practical application. Robust control for nonlinear systems is complex, and ensuring control performance while meeting real-time requirements is challenging. Hence, it is imperative to propose a fast generation method for the nonlinear robust control of fixed-wing UAVs.

The control methods for addressing nonlinear uncertain systems encompass linearization methods [2,3], model-based methods [4–6], learning-based methods [7–10], and adaptive control methods [11,12]. Although various nonlinear robust control methods are available for UAVs, each possesses its own advantages and disadvantages, as shown in Table 1. However, none of these methods can effectively address the challenges posed by strong nonlinearity, uncertainty, and the need for fast solving in UAV control design. Hence, an important research direction is the achievement of synergistic benefits by combining different methods to effectively address these challenges [13,14].

One advanced approach to combined control involves decomposing the complex nonlinear robust control into simpler nonlinear nominal control and Robust Compensation Control (RCC) [15,16]. In this approach, the nonlinear nominal control focuses on addressing the nonlinearity in the absence of disturbances, while the RCC is employed to

enhance the control effectiveness in the presence of disturbances. This combination control approach effectively alleviates the challenges associated with control design, simplifying the complexity of control implementation and enhancing the real-time performance of nonlinear robust control.

The design of RCC relies on quantitatively assessing the robustness of a nonlinear system. Trajectory tubes are used to depict the range of actual trajectories influenced by disturbances, with their size serving as a measure of system robustness. Consequently, trajectory tubes are extensively employed in RCC design and are known as Tube-based RCC (TRCC). In earlier studies on TRCC, trajectory tubes with non-variable cross-sectional shapes were initially used, where only the size could be adjusted to indicate robustness quantitatively [17]. Subsequent studies introduced trajectory tubes with variable shapes and sizes to enhance the TRCC efficiency [18,19]. However, the calculation of trajectory tubes described above is limited to using a linearized model near the nominal state, making it suitable for scenarios with small disturbances [17]. In situations involving significant disturbances or the coupling effects of multiple disturbances, the introduction of a nonlinear model near the nominal state becomes necessary [20,21]. Nevertheless, employing a nonlinear model in TRCC command calculations requires solving the complex Hamilton–Jacobi–Bellman (HJB) equation, which poses challenges to ensuring fast-solving performance in TRCC [22,23].

Introducing fast-solving methods for the HJB equation reduces the computational time for TRCC command generation [24,25]. However, the current direct solution approaches for the HJB equation mentioned above remain insufficient to meet the demands of online applications. In addition, an approximate HJB solution method based on Sum-of-Squares Programming (SOSP) is employed to improve the computational speed [26]. Nevertheless, SOSP converts HJB solving into semi-definite programming, which exhibits significant exponential growth in computational time as the dimensionality of the system's state increases. Consequently, while it is effective for systems with low dimensionality and weak nonlinearity, it falls short for high-dimensional and strong nonlinear fixed-wing UAVs in online applications. Reference [27] presents a method for the offline computation of a trajectory tube library, which is subsequently used for the online generation of TRCC commands. This approach effectively reduces the computational time required for online trajectory tube computation. However, when disturbances exceed the predefined range, this method will fail to promptly reconstruct the trajectory tube library and may lead to a decrease in TRCC effectiveness. Hence, existing computation methods on trajectory tubes still do not meet the TRCC requirements for online applications.

To address fast solving in nonlinear TRCC for fixed-wing UAVs, current research often emphasizes advancements in algorithms, overlooking the significant improvements that parallel computing and dimensionality reduction can bring to real-time applications. This paper proposes a fast generation method for nonlinear TRCC by incorporating parallel computing and dimensionality reduction techniques. Firstly, we developed a solving method for discrete trajectory tubes and introduced a parallelizable TRCC command solving algorithm that minimizes trajectory tubes to enhance robustness. Secondly, taking into account the characteristics of fixed-wing UAVs, we proposed two-dimensionality reduction techniques and incorporated them with the control reuse method into a fast TRCC command solving algorithm. Finally, extensive tracking simulations validated the effectiveness of the proposed TRCC in enhancing the robustness of UAV tracking control while maintaining satisfactory fast-solving performance. The key innovations presented in this paper are as follows:

1. Parallel computing method for trajectory tube computation: This paper presents a novel method for solving discrete trajectory tubes that simplifies the solution process by eliminating temporal correlations between tubes at different states. This approach enables the efficient parallel processing of trajectory tubes at different state points.
2. Dimensionality reduction technique for TRCC of fixed-wing UAVs: Efficient dimensionality reduction techniques, including decoupling and stepwise approaches, are

proposed to address higher fast-solving requirements according to fixed-wing UAV characteristics. These techniques are incorporated into a fast TRCC algorithm to enhance online applications.

The structure of this paper is organized as follows. Section 2 provides a detailed description of the methodology for TRCC. Techniques for fast TRCC are presented in Section 3. Section 4 includes simulations and evaluations. Finally, Section 5 presents the conclusions.

Table 1. The main methods for robust control of UAVs.

Methods	Robustness		Characteristics		
	External Disturbances	Dynamic Uncertainties	Nonlinearity	Fast Solving	Handling Constraint
Feedback linearization [2]		√	√	√	
Nonlinear dynamic inversion (NDI) [3]	√		√	√	
Sliding mode control (SMC) [4,5]	√		√		
Model predictive control (MPC) [6]		√		√	√
Fuzzy control [7]	√		√	√	
Artificial neural network (ANN) [8]	√		√		
Reinforcement learning (RL) [9]		√	√		
Safe RL [10]	√	√	√		√
Adaptive control [11]	√	√	√	√	
Adaptive backstepping [12]	√	√	√		
RL-based SMC [13]	√	√	√		
ANN-based adaptive NDI [14]		√	√		
LQR with robust compensation [15,16]	√	√		√	
Tube MPC [17]	√	√		√	√
Homothetic tube MPC [18]	√	√		√	√
Parameterized tube MPC [19]	√	√		√	√
Nonlinear tube MPC [20]	√	√	√		√
Robust LQR-Trees [21]	√	√	√		
TRCC [22,23]	√	√	√		√
TRCC with direct HJB solving [24,25]	√	√	√		√
TRCC with SOSP and tube library [27]	√	√	√		√

2. TRCC Algorithm Based on Discrete Trajectory Tubes

The nonlinear robust control of fixed-wing Unmanned Aerial Vehicles (UAVs) encompasses nonlinear nominal control and Robust Compensation Control (RCC), as shown in Figure 1. The trajectory tube represents the reachable set of actual trajectories for a UAV under disturbance. A smaller tube indicates that the actual trajectories closely approximate the ideal trajectory, thereby demonstrating a higher level of robustness. Consequently, the trajectory tube serves as a quantitative indicator to guide the design of RCC, known as Tube-based RCC (TRCC). Therefore, this section begins by presenting a calculation method for discrete trajectory tubes (step 1 in Figure 1) and subsequently introduces the TRCC method based on this robustness indicator (step 2 in Figure 1).

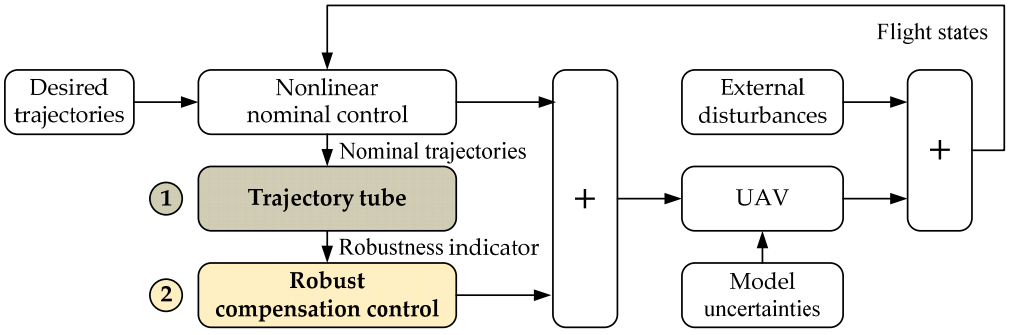

Figure 1. Structure of TRCC for UAVs.

2.1. Discrete Trajectory Tube Calculation Method

At time step t, for a nonlinear UAV, the relationship between the nominal state s_t, the nominal control a_t, and the next state s_{t+1} can be described as follows:

$$s_{t+1} - s_t = f(s_t, a_t) \tag{1}$$

Under the influence of disturbances, the actual state of the UAV is denoted as $\bar{s}_t$. The state error, $\hat{s}_t = \bar{s}_t - s_t$, can be defined accordingly. Thus, the relationship between the current state error $\hat{s}_t$ and the subsequent state error $\hat{s}_{t+1}$ can be expressed as:

$$\hat{s}_{t+1} - \hat{s}_t = f(s_t + \hat{s}_t, a_t) - f(s_t, a_t) \tag{2}$$

The initial set of disturbances is defined as $\mathcal{P}_t^0$. When $\hat{s}_t \in \mathcal{P}_t^0$, the reachable set $\mathcal{P}_{t+1}$ for the next state can be determined by $\hat{s}_{t+1} \in \mathcal{P}_{t+1}$. The combination of $\mathcal{P}_t^0$ and $\mathcal{P}_{t+1}$ constitutes the trajectory tube at time step t, as shown in Figure 2. Due to the uncertainty in the shapes of $\mathcal{P}_t^0$ and $\mathcal{P}_{t+1}$, the calculation of the trajectory tube becomes more challenging. To overcome this, the external ellipsoids of $\mathcal{P}_t^0$ and $\mathcal{P}_{t+1}$ are employed to alleviate the computational complexity. Consequently, the trajectory tube $\mathcal{T}_t$ used in practice is defined as the region formed by connecting the external ellipsoids $\mathcal{E}_t^0$ and $\mathcal{E}_{t+1}$ of $\mathcal{P}_t^0$ and $\mathcal{P}_{t+1}$, respectively.

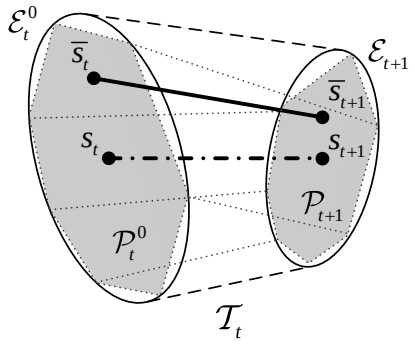

Figure 2. A discrete trajectory tube.

2.1.1. Initial Ellipsoid Calculation

The initial ellipsoid can be defined as $\mathcal{E}_t^0 = \{\hat{s}_t | 0 \le e_t^0(\hat{s}_t) \le \rho_t^0\}$, where e_t^0 represents the shape of the ellipsoid and ρ_t^0 represents its size. For solving the initial ellipsoid, e_t^0 and ρ_t^0 are iteratively optimized, and the area of the initial ellipsoid $size(\mathcal{E}_t^0)$ can be gradually decreased while ensuring that $\mathcal{E}_t^0$ contains $\mathcal{P}_t^0$. As shown in Figure 3, the $\mathcal{E}_t^0$ gradually approaches the external ellipsoid of $\mathcal{P}_t^0$.

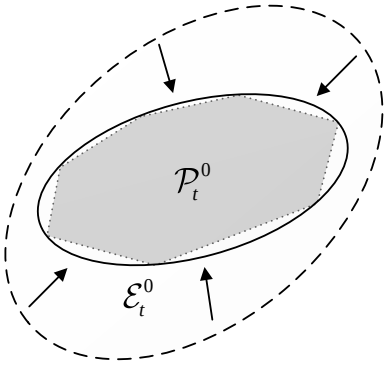

Figure 3. Approach of the initial ellipsoid calculation.

The initial error set $\mathcal{P}_t^0$ is assumed to be a semi-algebraic set that can be described by N_t polynomial inequalities, denoted as $\mathcal{P}_t^0 = \{\hat{s}_t | g_{t,i}(\hat{s}_t) \geq 0, \forall i = 1, 2, \ldots, N_t\}$. Consequently, the optimization problem for finding $\mathcal{E}_t^0$ can be formulated as follows:

$$\begin{array}{ll} \min\limits_{e_t^0, \rho_t^0} & size\left(\mathcal{E}_t^0\right) \\ \text{s.t.} & g_{t,i}(\hat{s}_t) \geq 0, \forall i = 1, 2, \ldots, N_t \Rightarrow 0 \leq e_t^0(\hat{s}_t) \leq \rho_t^0 \end{array} \tag{3}$$

2.1.2. Terminal Ellipsoid Calculation

Similarly, the terminal ellipsoid can be defined as $\mathcal{E}_{t+1} = \{\hat{s}_{t+1} | 0 \leq e_{t+1}(\hat{s}_{t+1}) \leq \rho_{t+1}\}$. For all initial states located on the initial ellipsoid, the terminal ellipsoid is determined as the smallest ellipsoid that contains the next state. Through the iterative optimization of e_{t+1} and ρ_{t+1}, the area of the terminal ellipsoid $size(\mathcal{E}_{t+1})$ gradually decreases while maintaining $e_{t+1}(\hat{s}_{t+1}) \leq \rho_{t+1}$. This approach gives the solution method of the $\mathcal{E}_{t+1}$, as depicted in Figure 4.

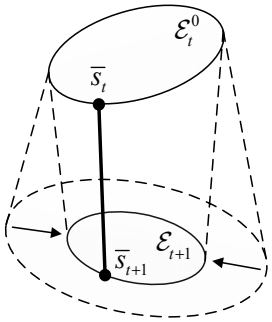

Figure 4. Approach of the terminal ellipsoid calculation.

However, $e_{t+1}(\hat{s}_{t+1})$ includes all four decision variables (e_t^0, ρ_t^0, e_{t+1}, and ρ_{t+1}), which can present challenges in terminal ellipsoid solving. Hence, the constraint $e_{t+1}(\hat{s}_{t+1}) \leq \rho_{t+1}$ is converted into an incremental form, $e_t^0(\hat{s}_t) + de_t^0(\hat{s}_t) \leq \rho_t^0 + d\rho_t^0$. Additionally, because of $e_t^0(\hat{s}_t) = \rho_t^0$, the constraint is further transformed into $de_t^0(\hat{s}_t) \leq d\rho_t^0$, where the increment can be approximated as follows:

$$\begin{cases} de_t^0(\hat{s}_t) & = \frac{\partial e_t^0}{\partial \hat{s}_t} \times \frac{d\hat{s}_t}{dt} + \frac{\partial e_t^0}{\partial t} \\ & \approx \frac{\partial e_t^0}{\partial \hat{s}_t} \times \frac{\hat{s}_{t+1} - \hat{s}_t}{\Delta t} + \frac{e_{t+1} - e_t^0}{\Delta t} \\ d\rho_t^0 & \approx \frac{\rho_{t+1} - \rho_t^0}{\Delta t} \end{cases} \tag{4}$$

In addition, when considering model uncertainties σ_t, the $f(\hat{s}_t, a_t)$ should be modified to $f(\hat{s}_t, a_t, \sigma_t)$. Similar to disturbances on the state, model uncertainties can also be described by a semi-algebraic set: $\mathcal{W}_t = \left\{\sigma_t \middle| g_{\sigma,t,j}(\sigma_t) \geq 0, \forall j = 1, 2, \ldots, N_\sigma\right\}$. Hence, the optimization problem for finding $\mathcal{E}_{t+1}$ can be formulated as follows:

$$\begin{array}{ll} \min\limits_{e_{t+1}, \rho_{t+1}} & size(\mathcal{E}_{t+1}) \\ \text{s.t.} & e_t^0(\hat{s}_t) = \rho_t^0, g_{\sigma,t,j}(\sigma_t) \geq 0, \forall j = 1, 2, \ldots, N_\sigma \Rightarrow de_t^0(\hat{s}_t) \leq d\rho_t^0 \end{array} \tag{5}$$

By combining the solution methods for $\mathcal{E}_t^0$ and $\mathcal{E}_{t+1}$, as given in Equations (3) and (5), respectively, the calculation method for $\mathcal{T}_t$ can be derived.

2.2. Sum-of-Squares Programming

Sum-of-Squares Programming (SOSP) can be utilized to verify the validity of a conditional statement [26,28]. The SOSP methodology entails converting the conditional statement into an equivalent polynomial, and its validity is established based on the non-negativity of this polynomial. Moreover, the SOSP method verifies the non-negativity

of the polynomial by examining its representability as a sum of squares. Incorporating the SOSP method necessitates adjustments to the constraints, optimization variables, and optimization objectives specified in Equations (3) and (5).

2.2.1. Constraints

Consider the conditional statement $g_{t,i}(\hat{s}_t) \geq 0, \forall i = 1, 2, \ldots, N_t \Rightarrow 0 \leq e_t^0(\hat{s}_t) \leq \rho_t^0$ in Equation (3). It is assumed that a set of non-negative multiplier polynomials $L_{t,i}$ $(i = 1, 2, \ldots, N_t)$ exists, ensuring that the constructed polynomial $\rho_t^0 - e_t^0(\hat{s}_t) - \sum_{i=1}^{N_t} L_{t,i}(\hat{s}_t) g_{t,i}(\hat{s}_t)$ is non-negative. Therefore, when the condition on the left-hand side of "⇒" holds, it can be inferred that $\rho_t^0 - e_t^0(\hat{s}_t)$ is non-negative, signifying the validity of the conclusion on the right-hand side of "⇒". By employing the SOSP approach, the constraint in Equation (3) can be modified as follows:

$$\begin{cases} \rho_t^0 - e_t^0(\hat{s}_t) - \sum_{i=1}^{N_t} L_{t,i}(\hat{s}_t) g_{t,i}(\hat{s}_t) \text{ is SOS,} \\ L_{t,i}(\hat{s}_t) \text{ are SOS, } \forall i = 1, 2, \ldots, N_t \end{cases} \tag{6}$$

Similarly, the constraint in Equation (5) can be modified as follows:

$$\begin{cases} d\rho_t^0 - de_t^0(\hat{s}_t) - \left(e_t^0(\hat{s}_t) - \rho_t^0\right) L_t(\hat{s}_t) - \sum_{j=1}^{N_\sigma} L_{\sigma,t,j}(\hat{s}_t) g_{\sigma,t,j}(\sigma_t) \text{ is SOS,} \\ L_{\sigma,t,j}(\hat{s}_t) \text{ are SOS, } \forall j = 1, 2, \ldots, N_\sigma \end{cases} \tag{7}$$

In addition, in the SOSP, it is necessary for all functions within the constraints to be represented as polynomial functions. Consequently, the Taylor expansion of $f(\hat{s}_t, a_t, \sigma_t)$ is employed instead of the original dynamics in the calculation of $de_t^0(\hat{s}_t)$, as stated in Equation (7). Typically, utilizing a third-order Taylor expansion guarantees adherence to computational error requirements [27].

2.2.2. Optimization Variables

In Equation (3), the initial ellipsoid function e_t^0 can be represented as a quadratic function with respect to $\hat{s}_t$, denoted as $e_t^0(\hat{s}_t) = \hat{s}_t^T E_t^0 \hat{s}_t$, where E_t^0 is a semi-positive definite matrix. Similarly, a semi-positive definite matrix E_{t+1} can be introduced to describe the terminal ellipsoid function e_{t+1}. Consequently, the optimization variables are transformed into E_t^0, ρ_t^0, E_{t+1}, and ρ_{t+1}. Furthermore, the optimization variables should include the non-negative multiplier polynomials in the constraints.

2.2.3. Optimization Objectives

In Equation (3), the objective function $size(\mathcal{E}_t^0) = \rho_t^0 / |E_t^0|$ is obtained based on the quadratic expression of the ellipsoid. Similarly, the objective function in Equation (5) is denoted as $size(\mathcal{E}_{t+1}) = \rho_{t+1} / |E_{t+1}|$.

Taking $size(\mathcal{E}_t^0)$ as an example, it is necessary to modify the objective function to $size(\mathcal{E}_t^0) = \rho_t^0$ since the SOSP can only handle linear objective functions. Furthermore, in order to account for the joint reduction of ρ_t^0 and $|E_t^0|$, and ensure the feasibility of the optimization problem, the constraints on E_t^0 need to be augmented. Therefore, an additional constraint $h^{\mathrm{T}}(E_t^0 - E_t^{\mathrm{R}})h \geq 0, \forall h \neq 0$ is introduced to restrict the unbounded reduction of $|E_t^0|$ during the optimization calculation. Here, E_t^{R} represents an initial feasible solution for E_t^0. The solution for E_t^0 can be obtained by solving the Hamilton–Jacobi–Bellman (HJB) differential equation, and an initial feasible solution can be provided by utilizing the Riccati algebraic equation, which is a simplification of the HJB differential equation under linear conditions [29]. Thus, E_t^{R} can be determined by the following algebraic equation:

$$E_t^{\mathrm{R}} A(s_t) + A^T(s_t) E_t^{\mathrm{R}} + Q - E_t^{\mathrm{R}} B(s_t) R^{-1} B^T(s_t) E_t^{\mathrm{R}} = 0, \tag{8}$$

where matrices A and B represent the state and control matrices, respectively, which are obtained by linearizing equations f at time t. The weight matrices Q and R are diagonal matrices, and their selection is determined by the initial disturbance of the various states.

Specifically, the weight values on the diagonal of the weight matrix are chosen to be larger for states with larger initial disturbances.

Similarly, the objective function in Equation (5) needs to be modified to $size(\mathcal{E}_{t+1}) = \rho_{t+1}$, and the constraint $h^{\mathrm{T}}(E_{t+1} - E_{t+1}^{\mathrm{R}})h \geq 0, \forall h \neq 0$ should be added.

With the incorporation of the SOSP, the mathematical model for the optimization calculation of the initial ellipsoid, as depicted in Equation (3), can be transformed into the following form by considering the adjusted constraints, optimization variables, and objective functions:

$$\begin{aligned} \min_{E_t^0, \rho_t^0, L_{t,i}} \quad & \rho_t^0 \\ \text{s.t.} \quad & \rho_t^0 - e_t^0(\hat{s}_t) - \textstyle\sum_{i=1}^{N_t} L_{t,i}(\hat{s}_t) g_{t,i}(\hat{s}_t) \text{ is SOS}, \\ & L_{t,i}(\hat{s}_t) \text{ are SOS}, \forall i = 1, 2, \ldots, N_t, \\ & h^{\mathrm{T}}(E_t^0 - E_t^{\mathrm{R}})h \geq 0, \forall h \neq 0 \end{aligned} \tag{9}$$

In addition, the mathematical model for the optimization calculation of the initial ellipsoid, as shown in Equation (5), can be transformed into the following form:

$$\begin{aligned} \min_{E_{t+1}, \rho_{t+1}, L_t, L_{\sigma,t,j}} \quad & \rho_{t+1} \\ \text{s.t.} \quad & \mathrm{d}\rho_t^0 - \mathrm{d}e_t^0(\hat{s}_t) - (e_t^0(\hat{s}_t) - \rho_t^0) L_t(\hat{s}_t) - \textstyle\sum_{j=1}^{N_\sigma} L_{\sigma,t,j}(\hat{s}_t) g_{\sigma,t,j}(\sigma_t) \text{ is SOS}, \\ & L_{\sigma,t,j}(\hat{s}_t) \text{ are SOS}, \forall j = 1, 2, \ldots, N_\sigma, \\ & h^{\mathrm{T}}(E_{t+1} - E_{t+1}^{\mathrm{R}})h \geq 0, \forall h \neq 0 \end{aligned} \tag{10}$$

In summary, by sequentially solving Equations (9) and (10) for a specified range of disturbances, it is feasible to calculate the discrete trajectory tube at the desired state point.

2.3. TRCC Algorithm

To strengthen the robustness of the nominal control a_t against a specified range of disturbances, a polynomial TRCC command $\hat{a}_t(\hat{s})$ is introduced. By incorporating this compensation control, Equation (2) can be modified as follows:

$$\hat{s}_{t+1} - \hat{s}_t = f(s_t + \hat{s}_t, a_t + \hat{a}_t(\hat{s}_t)) - f(s_t, a_t) \tag{11}$$

The trajectory tube is utilized as a quantitative indicator to guide the design of TRCC. The feedback control command is optimized to minimize the trajectory tube within a specified range of disturbances, aiming to enhance the robustness of the UAV. Consequently, the TRCC command polynomial is introduced as an optimization variable in the calculation of the terminal ellipsoid, as illustrated in Equation (10).

The magnitude of the compensation control command is constrained by the available control capacity of the UAV after incorporating the nominal control. Therefore, it is necessary to include constraints on the magnitude of the compensation control command in the process of solving for TRCC, as detailed below:

$$e_t^0(\hat{s}_t) \leq \rho_t^0 \Rightarrow a_{\min} \leq a_t + \hat{a}_t(\hat{s}) \leq a_{\max}, \tag{12}$$

where $a_{\max}$ and $a_{\min}$ represent the upper and lower bounds of the control, respectively.

By employing the SOSP method, the conditional statement in Equation (12) can be modified as follows:

$$\begin{cases} \rho_t^0 - e_t^0(\hat{s}_t) - (a_{k,\max} - (a_{k,t} + \hat{a}_{k,t})) L_{t,au,k}(\hat{s}_t) \text{ is SOS}, \\ \rho_t^0 - e_t^0(\hat{s}_t) - ((a_{k,t} + \hat{a}_{k,t}) - a_{k,\min}) L_{t,al,k}(\hat{s}_t) \text{ is SOS}, \end{cases} \tag{13}$$

where $a_t = [a_{1,t}, a_{2,t}, \cdots, a_{k,t}, \cdots]$, and $L_{t,au,k}(\hat{s}_t)$ and $L_{t,al,k}(\hat{s}_t)$ are non-negative multiplier polynomials.

By including the constraints presented in Equation (13), the optimization calculation for the terminal ellipsoid, as illustrated in Equation (10), can be modified to:

$$\begin{aligned} \min_{E_{t+1},\rho_{t+1},\hat{a}_{k,t},\, L_t, L_{\sigma,t,j},\, L_{t,au,k}, L_{t,al,k}} \quad & \rho_{t+1} \\ \text{s.t.} \quad & \mathrm{d}\rho_t^0 - \mathrm{d}e_t^0(\hat{s}_t) - \left(e_t^0(\hat{s}_t) - \rho_t^0\right) L_t(\hat{s}_t) - \textstyle\sum_{j=1}^{N_\sigma} L_{\sigma,t,j}(\hat{s}_t) g_{\sigma,t,j}(\sigma_t) \text{ is SOS}, \\ & L_{\sigma,t,j}(\hat{s}_t) \text{ are SOS}, \forall j = 1, 2, \ldots, N_\sigma, \\ & \rho_t^0 - e_t^0(\hat{s}_t) - (a_{k,\max} - (a_{k,t} + \hat{a}_{k,t})) L_{t,au,k}(\hat{s}_t) \text{ is SOS}, \\ & \rho_t^0 - e_t^0(\hat{s}_t) - ((a_{k,t} + \hat{a}_{k,t}) - a_{k,\min}) L_{t,al,k}(\hat{s}_t) \text{ is SOS}, \\ & L_{t,au,k}(\hat{s}_t), L_{t,al,k}(\hat{s}_t) \text{ are SOS}, \\ & h^{\mathrm{T}}\left(E_{t+1} - E_{t+1}^{\mathrm{R}}\right) h \geq 0, \forall h \neq 0 \end{aligned} \tag{14}$$

Hence, a solution algorithm for TRCC based on a discrete trajectory tube can be derived, as shown in Algorithm 1. Prior to implementing the proposed TRCC algorithm, the nominal state s_t and the nominal control a_t should be given by the nominal nonlinear control.

Algorithm 1 TRCC

1: Input: $f, s_t, a_t, \mathcal{P}_t^0, \mathcal{W}_t$

2: Initialize $E_t^{\mathrm{R}}, E_{t+1}^{\mathrm{R}}$ according to Equation (8)

3: Parfor t = 1 to T

4: Calculate $E_t^0, \rho_t^0, L_{t,i}$ according to Equation (9)

5: Calculate $E_{t+1}, \rho_{t+1}, \hat{a}_t, L_t, L_{\sigma,t,j}, L_{t,au,k}, L_{t,al,k}$ according to Equation (14)

End parfor

6: Output: $\hat{a}_t, \mathcal{T}_t$

In Algorithm 1, each iteration guarantees a decrease in the tube size, forming a monotonically non-increasing sequence. With the tube size bounded below by 0, the convergence of Algorithm 1 can be inferred [27].

3. Fast TRCC for UAV

SOSP transforms the optimization problem of the TRCC command into a semi-definite programming (SDP) problem. The time complexity for solving SDP problems with decision dimensionality $n < 10^2$ is $O(n^{3.5}\log(1/\varepsilon))$, where ε represents the required tracking accuracy [30]. The computational time shows a significant increase as the decision dimensionality grows. In the case of fixed-wing UAVs, the solution of TRCC involves addressing high-dimensional SDP, which presents challenges for efficient control command generation.

This section introduces dimensionality reduction techniques aimed at improving solution speed. Two methods for reducing the solving complexity of TRCC in fixed-wing UAVs are considered: decoupling longitudinal and lateral dynamics and stepwise methods. Additionally, control reuse techniques are employed to reduce the real-time requirement in online applications. As shown in Figure 5, by incorporating these techniques, fast compensation control solving can be achieved. This section initially introduces control reuse to reduce real-time requirements and subsequently provides methods for dimensionality reduction.

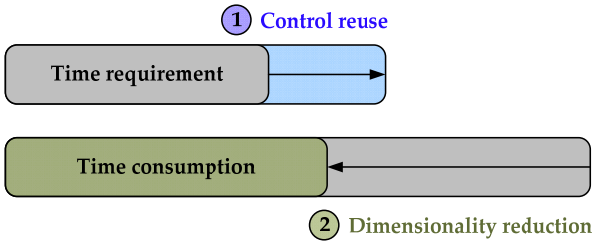

Figure 5. Techniques for fast-solving TRCC for UAVs.

3.1. Control Reuse

In general, the time interval (beat) Δt_{beat} for the TRCC command output is set to match the simulation step Δt_{step}. The calculation time interval Δt_{cal} for the TRCC command should not exceed Δt_{beat} and satisfy the condition $\Delta t_{\text{cal}} \leq \Delta t_{\text{beat}} = \Delta t_{\text{step}}$, as shown in Figure 6a. By increasing both Δt_{beat} and Δt_{step}, the limitation on Δt_{cal} can be reduced, thereby easing the fast-solving requirement. However, an excessively large Δt_{step} can compromise the simulation accuracy for UAVs. In this study, compensation control reuse was implemented to enlarge Δt_{beat} while keeping Δt_{step} constant, allowing the per-beat compensation control polynomial to be applied to multiple simulation steps. Consequently, the original requirement of $\Delta t_{\text{cal}} \leq \Delta t_{\text{step}}$ was relaxed to $\Delta t_{\text{cal}} \leq 2\Delta t_{\text{step}}$ using a one-step control reuse, as illustrated in Figure 6b. However, control reuse reduces the number of compensation control beats in a multi-step trajectory, resulting in decreased control performance under disturbances compared to no reuse.

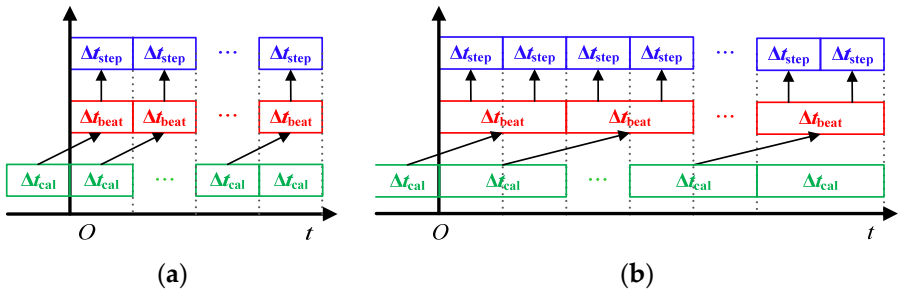

Figure 6. Approach of control reuse: (**a**) without reuse; (**b**) with one-step reuse.

3.2. Dimensionality Reduction

3.2.1. Decoupling

For the UAV, the state is defined as $s = [V, \alpha, q, \theta, \beta, p, r, \phi]$, where V represents the flight velocity, α and β are the angle-of-attack and sideslip angles, respectively; p, q, and r denote the roll, pitch, and yaw angle velocities, respectively; and θ and ϕ represent the pitch and roll angles, respectively. The drone control is denoted by $a = [\delta_{\text{e}}, \delta_{\text{th}}, \delta_{\text{a}}, \delta_{\text{r}}]$, where δ_{e} represents the elevator deflection, δ_{th} corresponds to the throttle position, δ_{a} denotes the aileron deflection, and δ_{r} represents the rudder deflection.

Despite the UAV's longitudinal and lateral dynamics being coupled, the coupling effects can be disregarded near the nominal trajectory under the action of the nominal control. Considering this assumption, the state used in compensation control calculation, originally consisting of eight dimensions, can be reduced to four dimensions of longitudinal state, $s_{\text{lon}} = [V, \alpha, q, \theta]$, and four dimensions of lateral state: $s_{\text{lat}} = [\beta, p, r, \phi]$. Similarly, the four dimensions of control can be reduced to two dimensions of longitudinal control $a_{\text{lon}} = [\delta_{\text{e}}, \delta_{\text{th}}]$ and two dimensions of lateral control $a_{\text{lat}} = [\delta_{\text{a}}, \delta_{\text{r}}]$. Based on the decoupling states and controls, the compensation control calculation can be divided into longitudinal and lateral parts (Algorithm 2). The independence of these two parts allows for parallel computation. With the reduction in state and control dimensions, the TRCC algorithm can be modified as follows:

Algorithm 2 Decoupling TRCC for UAV

Input: $f, s_t, a_t, \mathcal{P}_t^0, \mathcal{W}_t$

Initialize $E_t^{\mathrm{R}}, E_{t+1}^{\mathrm{R}}$ according to Equation (8)

Parfor t = 1 to T

  Parfor $(s, a) = \{(s_{\mathrm{lon}}, a_{\mathrm{lon}}), (s_{\mathrm{lat}}, a_{\mathrm{lat}})\}$

    Calculate $E_t^0, \rho_t^0, L_{t,i}$ according to Equation (9)

    Calculate $E_{t+1}, \rho_{t+1}, \hat{a}_t, L_t, L_{\sigma,t,j}, L_{t,au,k}, L_{t,al,k}$ according to Equation (14)

  End parfor

End parfor

Output: $\hat{a}_t, \mathcal{T}_t$

Decoupling compensation control affects performance through two factors. Firstly, neglecting longitudinal and lateral coupling results in reduced accuracy and poor control performance. However, simplifying the system to facilitate more accurate control computations under identical hardware and software conditions can ultimately enhance control performance.

3.2.2. Stepwise Method

The stepwise approach involves dividing decision variables into two groups, and sequentially optimizing one while keeping the other fixed. This increases the number of problems but reduces the number of decision variables in each problem, impacting computational time based on the number of variables in the original problem.

In the TRCC algorithm (Algorithm 3), the initial ellipsoid calculation, as depicted in Equation (9), involves three decision variables and is not suitable for variable grouping. In contrast, the terminal ellipsoid calculation, presented in Equation (14), with seven decision variables, suits a stepwise approach for reducing computational time. By grouping the decision variables in Equation (14) into $\{\rho_{t+1}, \hat{a}_t, L_t, L_{\sigma,t,j}, L_{t,au,k}, L_{t,al,k}\}$ and $\{E_{t+1}, \rho_{t+1}\}$, the optimal solutions for the terminal ellipsoid and compensation control can be obtained separately. This leads to a three-step solution algorithm, modifying the previous two-step approach based on the initial and terminal ellipsoid solutions. The three-step solution algorithm is shown as follows:

In the two-step algorithm (Algorithm 1), the terminal ellipsoid and the compensation control are jointly optimized, resulting in an iterative optimization process for each of them. In contrast, the three-step algorithm optimizes each of them individually only once, which leads to a reduction in calculation accuracy and poor robust control performance. In addition, similar to the two-step algorithm, the three-step algorithm is capable of generating a monotonically non-increasing sequence, ensuring the convergence of the optimization.

Algorithm 3 3-step decoupling TRCC for UAV

Input: $f, s_t, a_t, \mathcal{P}_t^0, \mathcal{W}_t$

Initialize $E_t^{\mathrm{R}}, E_{t+1}^{\mathrm{R}}$ according to Equation (8)

Parfor $t = 1$ to T

  Parfor $(s, a) = \{(s_{\mathrm{lon}}, a_{\mathrm{lon}}), (s_{\mathrm{lat}}, a_{\mathrm{lat}})\}$

    Calculate $E_t^0, \rho_t^0, L_{t,i}$ according to Equation (9)

    set $E_{t+1} = E_{t+1}^{\mathrm{R}}$

    Calculate $\rho_{t+1}, \hat{a}_t, L_t, L_{\sigma,t,j}, L_{t,au,k}, L_{t,al,k}$ according to Equation (14) when hold E_{t+1}

    Calculate E_{t+1}, ρ_{t+1} according to Equation (14) when hold $\hat{a}_t, L_t, L_{\sigma,t,j}, L_{t,au,k}, L_{t,al,k}$

  End parfor

End parfor

Output: $\hat{a}_t, \mathcal{T}_t$

4. Simulation Test for Fast TRCC Performance

4.1. UAV Nonlinear Dynamics

4.1.1. Dynamic Equations

In order to conduct a comprehensive simulation and verification analysis of TRCC, this study implemented a continuous nonlinear flight dynamics model for the UAV. This nonlinear model was also utilized in the process of TRCC command calculation; however, it was necessary to discretize the model based on the form of Equation (1).

The quaternion method is utilized in the UAV nonlinear flight dynamics model to prevent singularities that may arise when using the Euler angle method. Equation (15) demonstrates the resulting dynamics equations.

$$\begin{cases} m\begin{bmatrix} \dot{u}+qw-rv \\ \dot{v}+ru-pw \\ \dot{w}+pv-qu \end{bmatrix} = \begin{bmatrix} X+2mg(q_1q_3-q_0q_2) \\ Y+2mg(q_2q_3+q_0q_1) \\ Z+mg\left(q_0^2-q_1^2-q_2^2+q_3^2\right) \end{bmatrix} \\ \begin{bmatrix} L \\ M \\ N \end{bmatrix} = \begin{bmatrix} I_x\dot{p}+(I_z-I_y)qr-I_{zx}\left(pq+\dot{r}\right) \\ I_y\dot{q}+(I_x-I_z)rp+I_{zx}\left(p^2-r^2\right) \\ I_z\dot{r}+(I_y-I_x)pq+I_{zx}\left(qr-\dot{p}\right) \end{bmatrix} \\ \begin{bmatrix} \dot{q}_0 \\ \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 0 & -p & -q & -r \\ p & 0 & r & -q \\ q & -r & 0 & p \\ r & q & -p & 0 \end{bmatrix}\begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} \end{cases}, \tag{15}$$

where m represents the mass of the body; I_{x}, I_{y}, I_{z}, and I_{zx} denote the rotational inertia of the body; g represents gravitational acceleration; the vector $[u, v, w]^{\mathrm{T}}$ represents the components of velocity in the body-axis; q_0, q_1, q_2, and q_3 refer to quaternions; and $[X, Y, Z]^{\mathrm{T}}$ and $[L, M, N]^{\mathrm{T}}$ represent the external forces and moments, which primarily account for the influence of aerodynamic forces and engine thrust.

Equation (16) gives the kinematic equations for a UAV in the ground coordinate system.

$$\begin{cases} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} q_0^2+q_1^2-q_2^2-q_3^2 & 2(q_1q_2-q_0q_3) & 2(q_1q_3+q_0q_2) \\ 2(q_1q_2+q_0q_3) & q_0^2-q_1^2+q_2^2-q_3^2 & 2(q_2q_3-q_0q_1) \\ 2(q_1q_3-q_0q_2) & 2(q_2q_3+q_0q_1) & q_0^2-q_1^2-q_2^2+q_3^2 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} \\ \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} = \begin{bmatrix} \cos\frac{\phi}{2}\cos\frac{\theta}{2}\cos\frac{\psi}{2}+\sin\frac{\phi}{2}\sin\frac{\theta}{2}\sin\frac{\psi}{2} \\ \sin\frac{\phi}{2}\cos\frac{\theta}{2}\cos\frac{\psi}{2}-\cos\frac{\phi}{2}\sin\frac{\theta}{2}\sin\frac{\psi}{2} \\ \cos\frac{\phi}{2}\sin\frac{\theta}{2}\cos\frac{\psi}{2}+\sin\frac{\phi}{2}\cos\frac{\theta}{2}\sin\frac{\psi}{2} \\ \cos\frac{\phi}{2}\cos\frac{\theta}{2}\sin\frac{\psi}{2}-\sin\frac{\phi}{2}\sin\frac{\theta}{2}\cos\frac{\psi}{2} \end{bmatrix} \end{cases}, \quad (16)$$

where $[x, y, z]^{\mathrm{T}}$ represents the components of the position in the ground-axis, while ψ denotes the yaw angle.

In addition, the velocity and aerodynamic angle of the UAV are determined by the following equation.

$$\begin{cases} V = \sqrt{u^2+v^2+w^2} \\ \alpha = \arctan(w/u) \\ \beta = \arcsin(v/V) \end{cases} \quad (17)$$

The UAV's control inputs include elevator, throttle, aileron, and rudder. The elevator deflection range is $[-20^\circ, 20^\circ]$, the throttle stick position is [0, 1], the aileron deflection range is $[-21.5^\circ, 21.5^\circ]$, and the rudder deflection range is $[-30^\circ, 30^\circ]$.

4.1.2. Aerodynamic Characteristics

The nonlinear aerodynamic forces and moments on the UAV can be calculated from the aerodynamic coefficients, denoted as $[X_{\text{aero}}, Y_{\text{aero}}, Z_{\text{aero}}, L_{\text{aero}}, M_{\text{aero}}, N_{\text{aero}}]^{\mathrm{T}} = 0.5 \times \rho \times S \times V^2 \times [-C_D, C_Y, -C_L, b \times C_l, c \times C_m, b \times C_n]^{\mathrm{T}}$, where ρ is the air density, S is the wing area, b is the aircraft spread length, and c is the average aerodynamic chord length. The variation curves depicting the aerodynamic coefficients mentioned above with respect to the angle-of-attack within the range of -10° to 25° is presented in Figure 7.

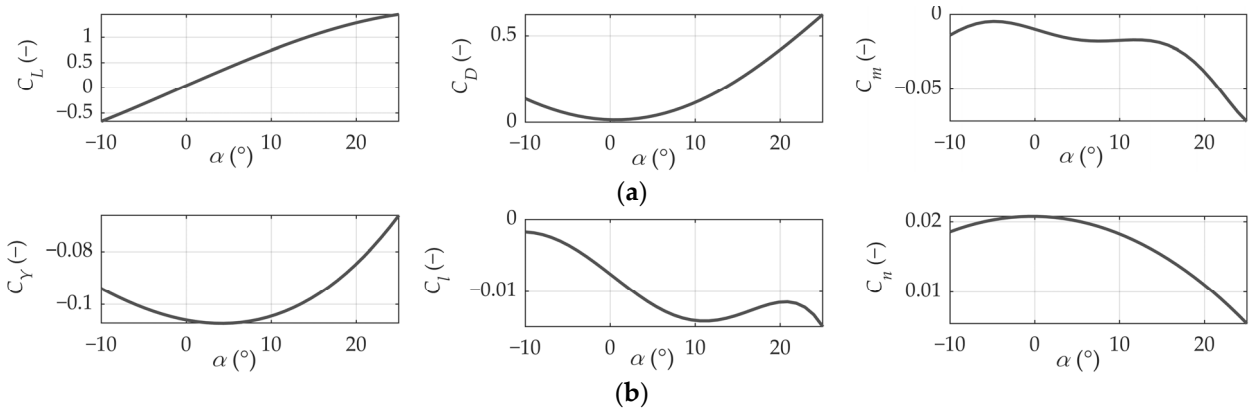

Figure 7. UAV nonlinear aerodynamic coefficients: (**a**) longitudinal aerodynamic coefficients with $\beta = 0^\circ$, $q = 0^\circ/\text{s}$ and $\delta_{\text{e}} = 0^\circ$; (**b**) lateral aerodynamic coefficients with $\beta = 5^\circ$, $p = 0^\circ/\text{s}$, $r = 0^\circ/\text{s}$, $\delta_{\text{a}} = 0^\circ$ and $\delta_{\text{r}} = 0^\circ$.

4.2. Performance Analysis for TRCC

The performance analysis of the TRCC involves verifying the trajectory tube and conducting simulations of the tracking task under disturbances. The nominal trajectory generated by constant control was used, with its initial state and control outlined in Table 2. The range of disturbance, determined based on sensor noise, is also presented in Table 2. The flight altitude is denoted by H, while σ_m represents the mass perturbation in the UAV model.

Table 2. Initial state, control, and disturbance for tube calculation.

Parameter	Value	Parameter	Value
V (m/s)	185	$\hat{V}$ (m/s)	$[-5, 5]$
α (°)	19.3	$\hat{\alpha}$ (°)	$[-0.5, 0.5]$
q (°/s)	28.2	$\hat{q}$ (°/s)	$[-0.5, 0.5]$
θ (°)	-50.5	$\hat{\theta}$ (°)	$[-1, 1]$
β (°)	0.7	$\hat{\beta}$ (°)	$[-0.5, 0.5]$
p (°/s)	-19.7	$\hat{p}$ (°/s)	$[-1, 1]$
r (°/s)	-9.4	$\hat{r}$ (°/s)	$[-0.5, 0.5]$
ϕ (°)	0	$\hat{\phi}$ (°)	$[-1, 1]$
H (km)	3	δ_e (°)	-5.3
δ_{th} (-)	0.75	δ_a (°)	3.0
δ_r (°)	0	σ_m (% m_{max})	$[-5, 5]$

4.2.1. Trajectory Tube

Figure 8 displays the calculated trajectory tubes with and without TRCC. Due to the hyperelliptic characteristic of the trajectory tube in the high-dimensional state, it is essential to project the trajectory tube onto a three-dimensional space composed of any two states and time to observe its shape accurately.

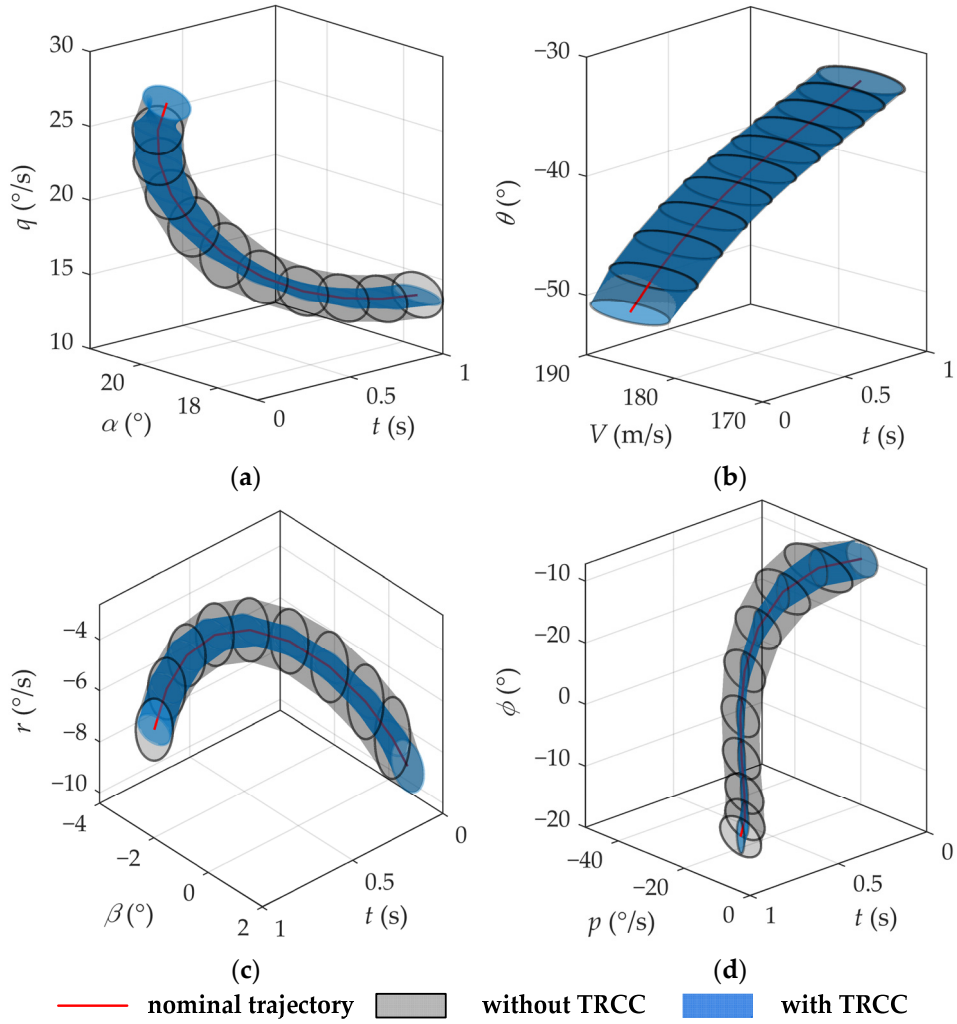

Figure 8. Calculated trajectory tubes: (**a**) projection at angle-of-attack and pitch angle velocity; (**b**) projection at velocity and pitch angle; (**c**) projection at sideslip angle and yaw angle velocity; (**d**) projection at roll angle and roll angle velocity.

In Figure 8a,b, tube projections for angle-of-attack, velocity, and pitch angles remained unchanged with TRCC, except for a decrease in the pitch angle velocity. Lateral states showed similar variations to longitudinal states, with less variation in sideslip and roll angles, and a significant reduction in yaw and roll angle velocities, as shown in Figure 8c,d. Pitch, roll, and yaw angle velocities are more sensitive to disturbances and exhibit a wider range of variation under disturbances. Hence, TRCC prioritizes the suppression of disturbance effects on angle velocity but fails to fully address angle and velocity disturbances in minimizing the trajectory tube size.

4.2.2. Performance in Tracking Tasks

The effectiveness of TRCC is theoretically demonstrated through a reduction in trajectory tube size. In addition, conducting tracking task simulations under disturbance enables a quantitative analysis of TRCC's performance disturbance suppression.

Results from 100 tracking task simulations, both with and without TRCC, were recorded separately and displayed in Figure 9. Each simulation considered random combinations of disturbances and modeling inaccuracies generated within the range specified in Table 2. The plotted area in Figure 9 illustrates the upper and lower boundaries of the tracking error for each state influenced by disturbances.

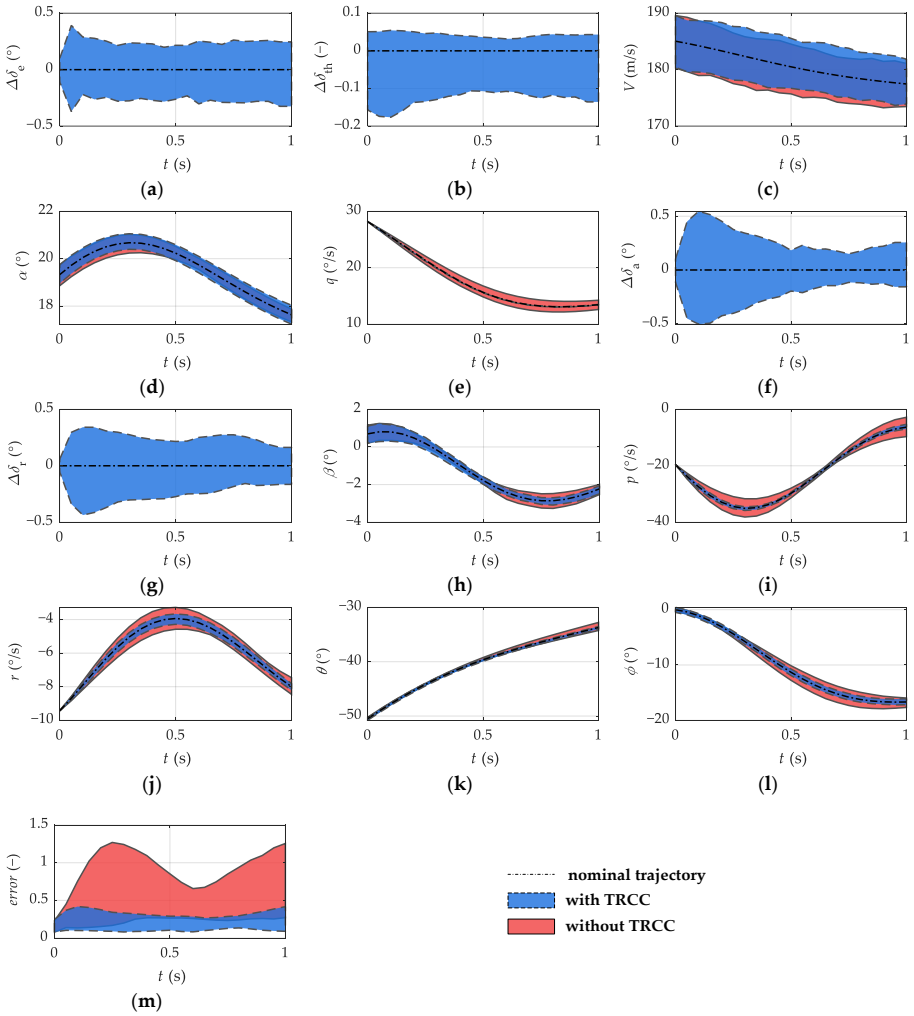

Figure 9. Tracking task simulations with and without TRCC: (**a**) elevator; (**b**) throttle; (**c**) velocity; (**d**) angle-of-attack; (**e**) pitch angle velocity; (**f**) aileron; (**g**) rudder; (**h**) sideslip angle; (**i**) roll angle velocity; (**j**) yaw angle velocity; (**k**) pitch angle; (**l**) roll angle; (**m**) sum of errors.

In Figure 9a,b, TRCC produces an elevator increment of $-0.2°$ to $0.2°$ and a throttle increment of -0.2 to 0.1 under continuous random disturbances. The overall velocity error range exhibited a slight upward shift but maintained a similar magnitude compared to the case without TRCC. Additionally, angle-of-attack errors primarily occurred in the first half of the trajectory. The tracking error ranges for each longitudinal state corresponded to the calculated results for the trajectory tube, as illustrated in Figure 9c–e, as well as in Figure 8a,b.

Furthermore, TRCC produced an aileron increment of $-0.5°$ to $0.5°$ and a rudder increment of $-0.3°$ to $0.3°$ under continuous random disturbances, as shown in Figure 9f,g. Aileron and rudder actions resulted in a slight decrease in the error of the sideslip angle during the second half of the trajectory. In addition, the error ranges for roll and yaw angle velocities significantly decreased. The tracking error ranges for each lateral state corresponded to the calculated results for the trajectory tube, as shown in Figure 9h–j, as well as in Figure 8c,d.

In Figure 9k,l, TRCC effectively reduced the pitch angle error during the second half of the trajectory, while the roll angle error was significantly diminished. Although the tracking error range deviated from the calculated trajectory tube, it highlighted the effectiveness of TRCC in mitigating pitch and roll disturbances.

The variable error was introduced to comprehensively describe the state tracking error in the presence of disturbances. It is defined as the root mean square (RMS) of the tracking error across all eight states. Mathematically, it can be expressed as:

$$error = \sqrt{\frac{\sum e_i^2}{\sum i}},\ \left(i = \frac{\hat{V}}{10^2}, \hat{\alpha}, \hat{q}, \hat{\theta}, \hat{\beta}, \hat{p}, \hat{r}, \hat{\phi}\right), \tag{18}$$

where the velocity error is divided by 10^2 to account for the significant magnitude difference compared with other states.

In Figure 9m, without TRCC, the average maximum tracking error (the average upper boundary of the tracking error over time) was 0.93. With TRCC, it decreased to 0.32, a 66% reduction. This demonstrates the effective improvement in UAV robustness achieved by the proposed TRCC in conjunction with nominal control.

4.3. Performance Analysis for TRCC with Control Reuse

To assess the influence of control reuse, 100 tracking task simulations were conducted. Random combinations of disturbances and modeling errors within the range specified in Table 2 were employed. The simulations compared scenarios without control reuse and with one-step control reuse, as illustrated in Figure 10.

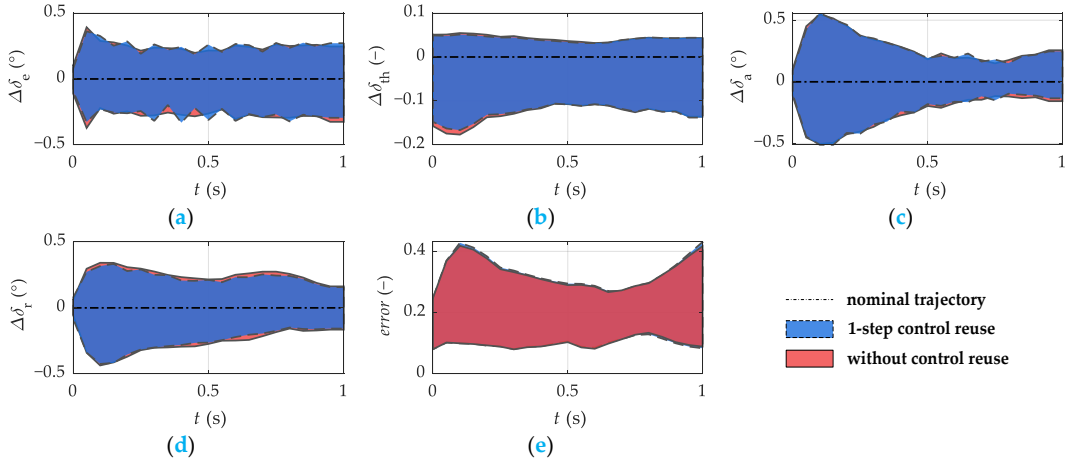

Figure 10. Tracking simulations under TRCC with and without control reuse: (**a**) elevator; (**b**) throttle; (**c**) aileron; (**d**) rudder; (**e**) sum of errors.

Figure 10a–d demonstrates that one-step control reuse widens the gap between successive TRCC commands, resulting in significant variation during TRCC switching. The elevator increment was particularly affected by the reuse, exhibiting noticeable jaggedness, while the throttle, aileron, and rudder displayed relatively slight jaggedness.

With one-step control reuse, the average maximum tracking error increased from 0.32 to 0.33, resulting in just a 3% increment compared to the case without control reuse, as shown in Figure 10e. Therefore, employing control reuse enables a reduction in fast-solving requirements while still satisfying the robustness requirement.

4.4. Time Consumption and Performance Analysis for Decoupling TRCC

4.4.1. Time Consumption

To analyze the computational time consumption of decoupling TRCC, the algorithms with and without dimensionality reduction were utilized to calculate the TRCC command 100 times. The computational time was recorded, and its distribution is depicted in Figure 11.

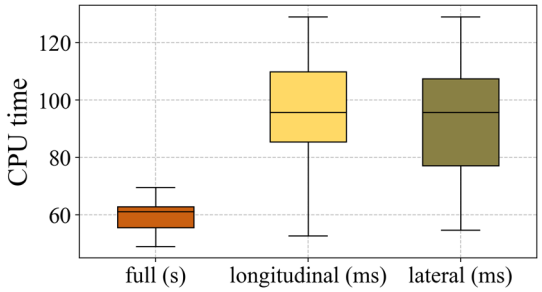

Figure 11. CPU time distribution under TRCC with and without decoupling.

The computational time analysis employed an SOSP solver based on the MATLAB R2020a platform SOSTOOLS v4.00 [31] and SEDUMI v1.05 [32]. All analysis results were obtained using an "Intel Xeon Gold 6230" 40-thread CPU (2.10 GHz) with 128GB RAM.

As illustrated in Figure 11, the computational time needed for TRCC is typically in the order of seconds, with a median value of approximately 61 s. This duration falls short of meeting the requirement for rapid generation. However, by employing the decoupling TRCC algorithm, the computation time for solving both the longitudinal and lateral subproblems is reduced to the millisecond range, with medians of around 94 ms and 95 ms, and maximums of about 129 ms and 128 ms, respectively. Hence, the decoupling TRCC algorithm can satisfy the fast generation criteria of 100 ms per beat, although a small number of cases slightly exceed this threshold.

4.4.2. Performance in Tracking Tasks

To assess the influence of decoupling and dimensionality reduction, 100 tracking task simulations were conducted. Random combinations of disturbances and modeling errors within the range specified in Table 2 were employed. The simulations compared scenarios with and without decoupling, as illustrated in Figure 12.

Figure 12a,b demonstrates the increased variation of elevator and throttle when utilizing decoupling TRCC, facilitating a more aggressive suppression of longitudinal disturbances. Consequently, Figure 12c,d exhibits slight reductions in velocity and angle-of-attack errors. Similarly, Figure 12e,f showcases expanded aileron variation, enabling the aggressive reduction of the roll angle velocity error. The variation in the rudder remained unchanged, as depicted in Figure 12g, and no significant differences were observed in other state error responses. Decoupling TRCC reduces the average maximum tracking error from 0.33 to 0.30 when compared to TRCC without decoupling, as shown in Figure 12h.

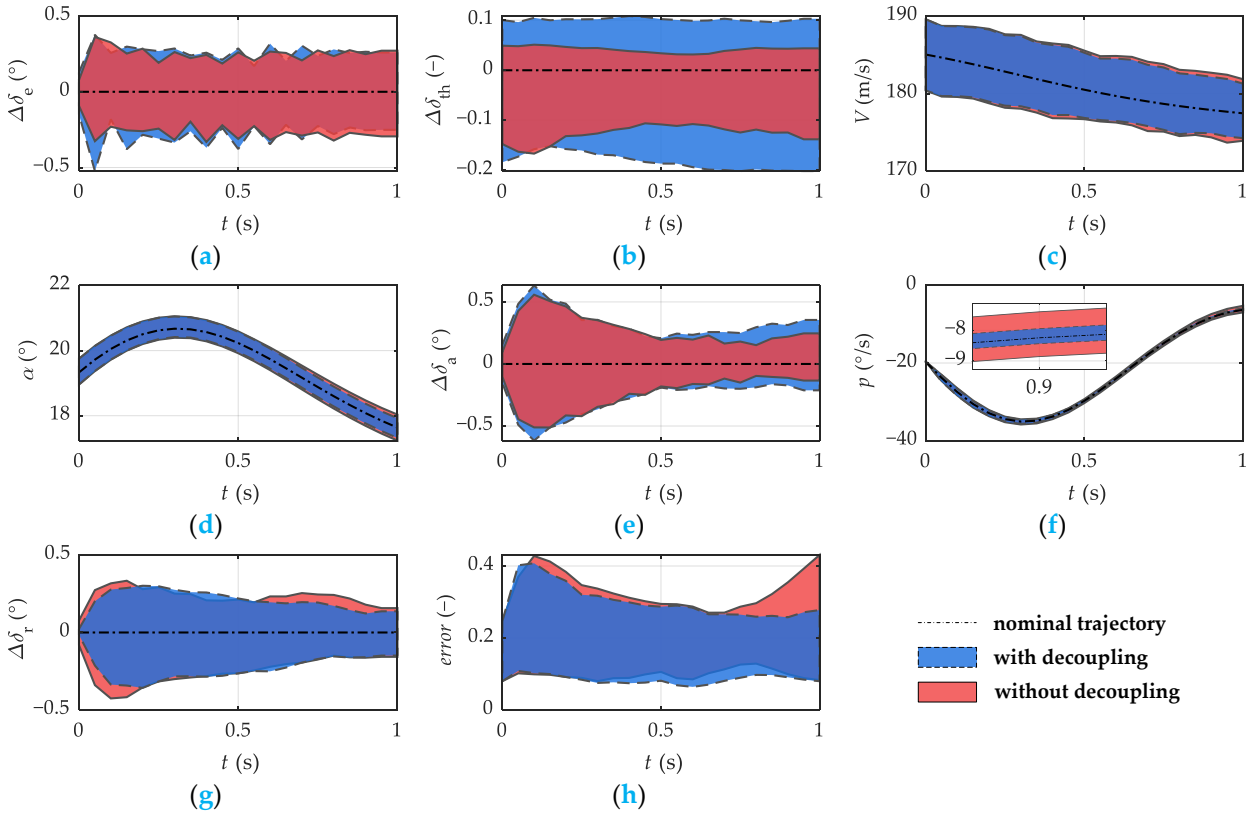

Figure 12. Tracking simulations under TRCC with and without decoupling: (**a**) elevator; (**b**) throttle; (**c**) velocity; (**d**) angle-of-attack; (**e**) aileron; (**f**) roll angle velocity; (**g**) rudder; (**h**) sum of errors.

Therefore, within a given UAV decoupling TRRC algorithm, the simplification of the problem after decoupling significantly enhanced computational accuracy, outweighing the decrease from neglecting longitudinal and lateral coupling. The decoupling TRRC effectively reduced the computational time and improved the control performance.

4.5. Time Consumption and Performance Analysis for Stepwise TRCC

4.5.1. Time Consumption

In addition to the incorporation of the decoupling algorithm, two-step and three-step algorithms were employed to calculate the TRCC command 100 times. The computational time was recorded, and its distribution is depicted in Figure 13.

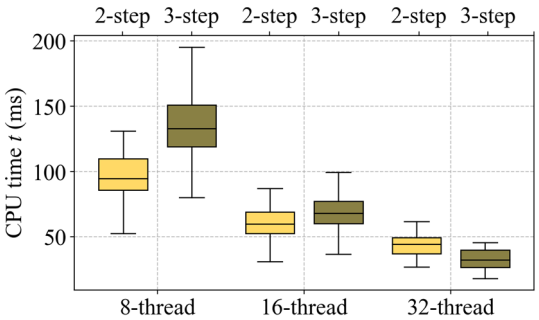

Figure 13. CPU time distribution under two-step and three-step TRCC algorithms.

The median computation times for solving TRCC were approximately 95 ms and 133 ms using the two-step and three-step algorithms, respectively, with eight parallel computing threads. In addition, based on Welch's *t*-test at a 5% significance level, three-

step algorithms exhibited a significantly higher average computation time than two-step algorithms, by 41.5 ms.

Increasing the number of threads to 16, the median computation times were approximately 60 ms and 68 ms using the two-step and three-step algorithms, respectively. Additionally, based on Welch's *t*-test at a 5% significance level, three-step algorithms exhibited a slightly higher average computation time than two-step algorithms, by 7.5 ms.

On further raising the number of threads to 32, the median computation times were approximately 44 ms and 33 ms using the two-step and three-step algorithms, respectively. In addition, based on Welch's *t*-test at a 5% significance level, three-step algorithms exhibited a moderately lower average computation time than two-step algorithms, by 10.6 ms. As the number of parallel computation threads increased, the three-step algorithm showed superiority.

4.5.2. Performance in Tracking Tasks

To assess the influence of the stepwise method, 100 tracking task simulations were conducted. Random combinations of disturbances and modeling errors within the range specified in Table 2 were employed. The simulations compared scenarios with two-step and three-step algorithms, as illustrated in Figure 14.

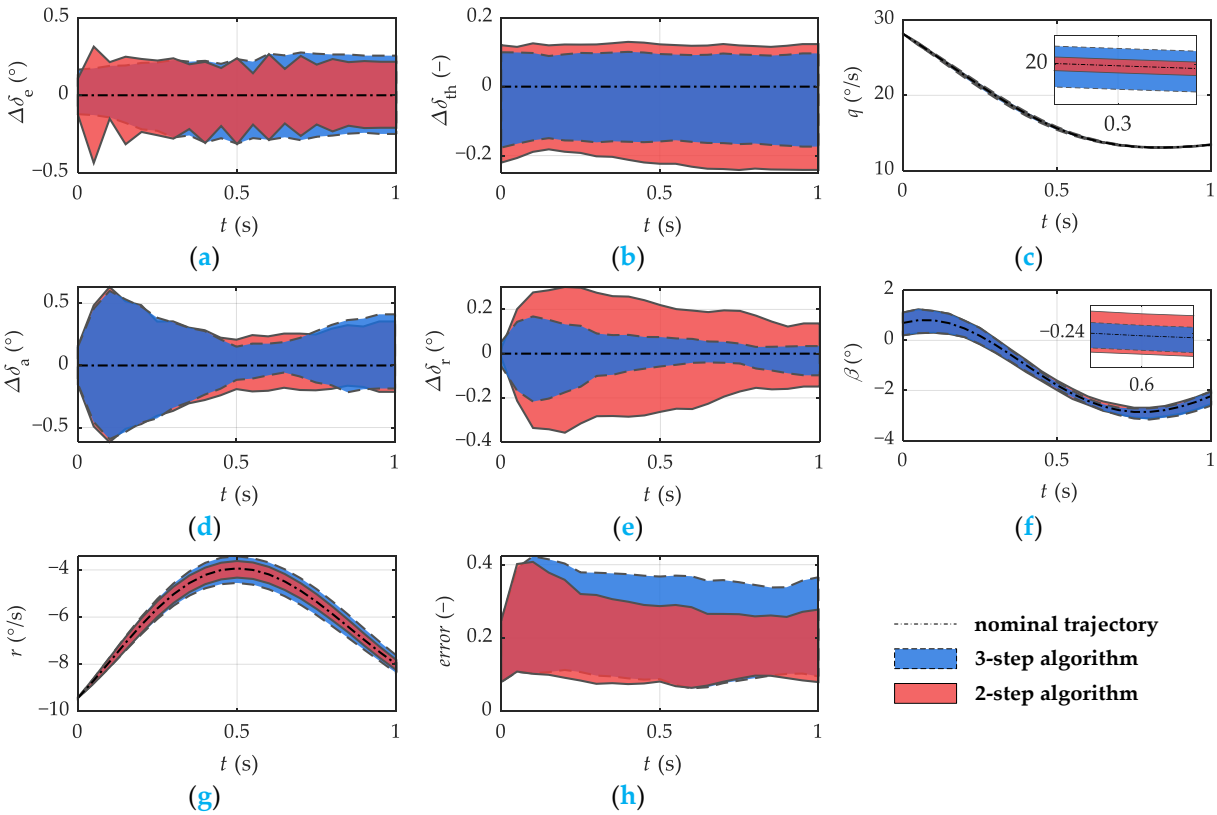

Figure 14. Tracking simulations with two-step and three-step TRCC: (**a**) elevator; (**b**) throttle; (**c**) pitch angle velocity; (**d**) aileron; (**e**) rudder; (**f**) sideslip angle; (**g**) roll angle velocity; (**h**) sum of errors.

Figure 14a,b shows that the three-step algorithm reduced the variation of the elevator and throttle compared to the two-step algorithm. Hence, this reduction led to ineffective compensation for longitudinal disturbances, increasing the pitch angle velocity error, as shown in Figure 14c. Additionally, the three-step algorithm led to minor aileron variation but significantly reduced the rudder variation, as shown in Figure 14d,e. Therefore, the rudder failed to adequately compensate for lateral disturbances, resulting in increased sideslip angle and yaw angle velocity errors, as shown in Figure 14f,g. Figure 14h demonstrates that the average maximum tracking error increased from 0.30 to 0.37 when employing

the three-step algorithm instead of the two-step algorithm. Thus, although the three-step algorithm reduced the computational time, it resulted in an approximately 23% reduction in TRCC performance.

4.6. Runtime Simulation

To validate the real-time performance of the Fast TRCC, a runtime simulation was conducted. The hardware device supported up to 40 parallel computational threads, with 16 threads each used for longitudinal and lateral RCC computation. The computation time analysis (Figure 13) demonstrated a consumption below 100 ms for the 16-thread environment, effectively meeting the 50 ms real-time requirement with a one-step delay. Consequently, the simulation step was set at 50 ms in the runtime simulation. Before initiating the runtime simulation, the TRCC command for the first 16 beats was pre-calculated. Throughout the runtime simulation, the algorithm handles the calculation of RCC commands for the subsequent 16 beats, as depicted in Figure 15. The results of the runtime simulation are presented in Figure 16.

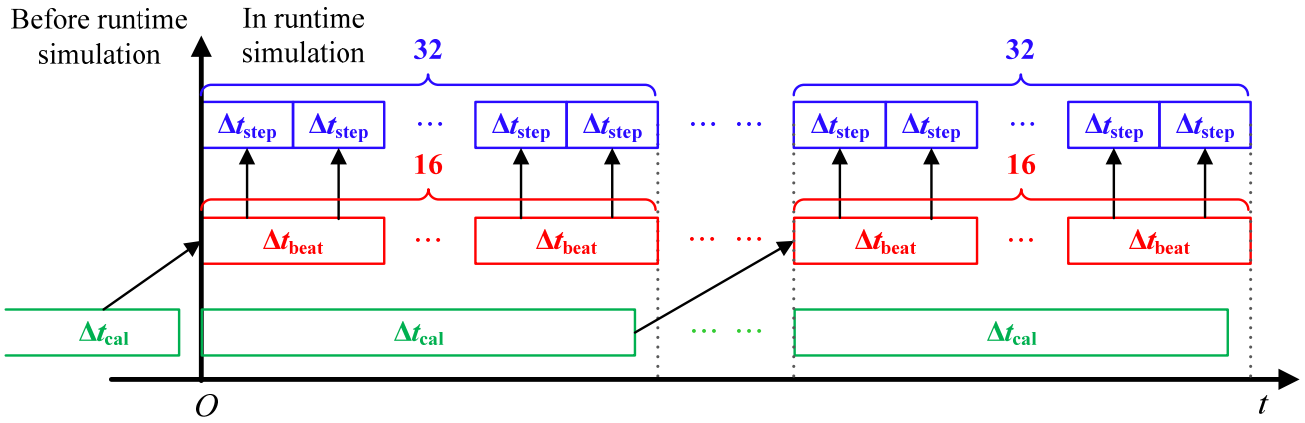

Figure 15. Time sequences for runtime simulation.

In the runtime test, the velocity initially deviated by 3 m/s, as depicted in Figure 16a. Since the actual velocity was lower than the nominal velocity, TRCC generated a positive throttle compensation to minimize the velocity error. The compensated throttle consistently satisfied the constraint, as illustrated in Figure 16b. Furthermore, TRCC effectively suppressed the measurement noise of the velocity and angle-of-attack through throttle and elevator compensation, reducing the tracking error of pitch angle velocity and pitch angle, as indicated in Figure 16a,f. In addition, in the runtime simulation, an angle-of-attack of 7° was achieved, in which the aerodynamic characteristics presented moderate nonlinearity, as shown in Figure 7.

As shown in Figure 16g, the sideslip angle exhibited an initial deviation of 0.3° and was affected by continuous measurement noise. Consequently, TRCC generated a larger rudder compensation, leading to a reduction in the tracking error for the sideslip angle and yaw angle velocity, as depicted in Figure 16g–i. Although TRCC produced a smaller aileron compensation, it still effectively reduced the tracking error for the roll angle velocity and roll angle, as shown in Figure 16j–l.

The combined tracking error curves, with and without the effect of TRCC, are presented in Figure 16m. To effectively evaluate the control effect of TRCC, the Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) metrics of the tracking error were used [33]. By utilizing TRCC to suppress the effect of initial state deviation and measurement noise, the MAE of the tracking error was reduced from 1.21 to 0.25, and the RMSE decreased from 1.30 to 0.27 after the implementation of TRCC, indicating improved robustness.

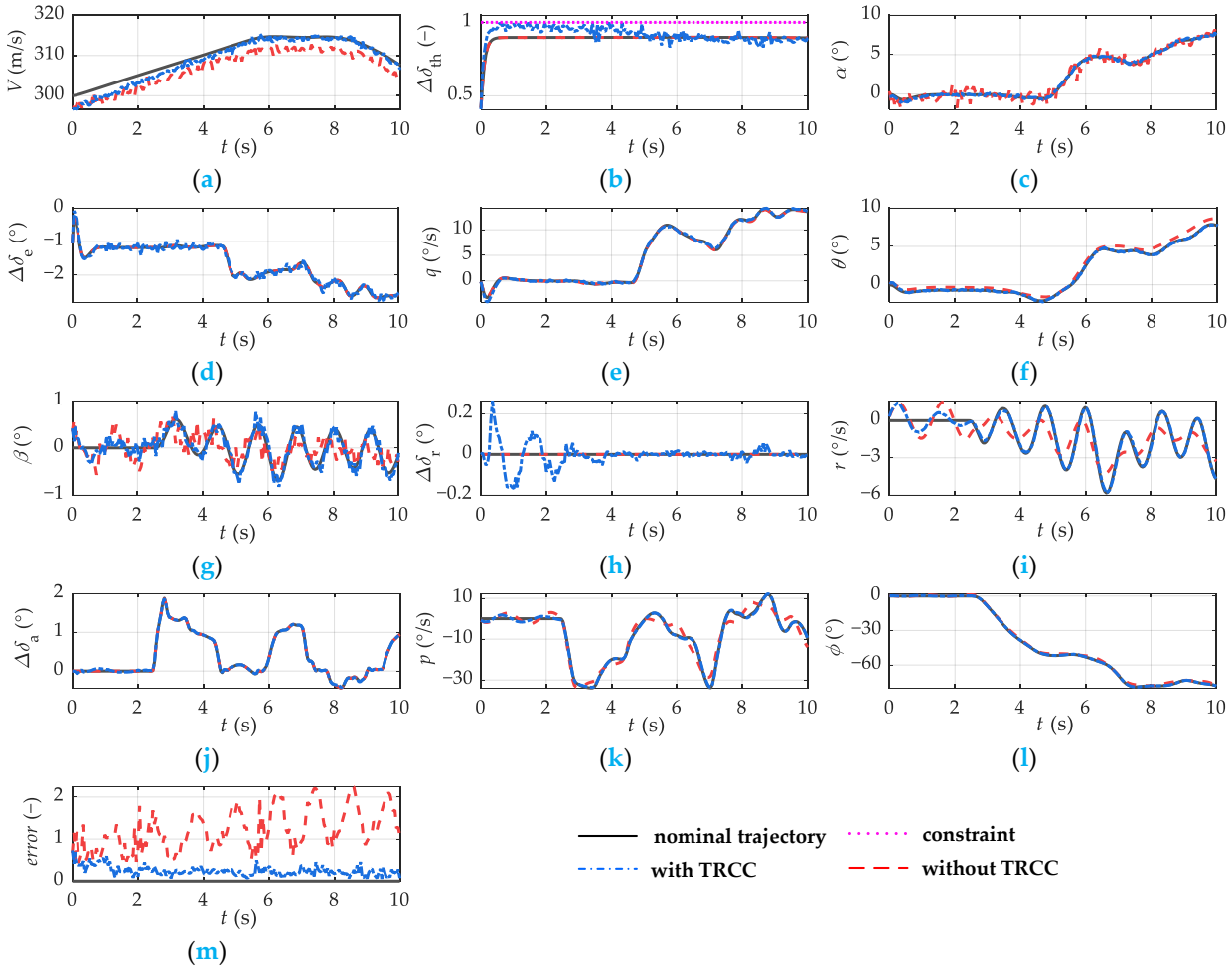

Figure 16. Runtime simulation of fast-solving TRCC: (**a**) velocity; (**b**) throttle; (**c**) angle-of-attack; (**d**) elevator; (**e**) pitch angle velocity; (**f**) pitch angle; (**g**) sideslip angle; (**h**) rudder; (**i**) yaw angle velocity; (**j**) aileron; (**k**) roll angle velocity; (**l**) roll angle; (**m**) sum of errors.

4.7. Sensitivity Analysis

Sensitivity analysis identifies key nominal states and actions affecting the TRCC command. The recalculation of the compensation control command polynomials is conducted only when these key factors experience significant changes, reducing the computational burden of TRCC implementation.

In the sensitivity analysis, two sets of 100 test nominal state and control inputs were randomly sampled within the ranges specified in Table 3. The sensitivity was determined by computing $\Delta C = C_1 - C_2$, where C_1 and C_2 are the coefficients of the compensation command polynomials obtained under the two sets of test inputs. A larger ΔC indicates that the input has a significant influence on the TRCC command, indicating high sensitivity. The sensitivity analysis results are shown in Figure 17.

Table 3. Range of nominal states and controls for sensitivity analysis.

Parameter	Range	Parameter	Range
V (m/s)	[200, 350]	α (°)	[−10, 25]
q (°/s)	[−30, 30]	θ (°)	[−60, 60]
β (°)	[−10, 10]	p (°/s)	[−50, 50]
r (°/s)	[−10, 10]	ϕ (°)	[−90, 90]
δ_e (°)	[−15, 15]	δ_{th} (-)	[0.2, 0.8]
δ_a (°)	[−15, 15]	δ_r (°)	[−25, 25]

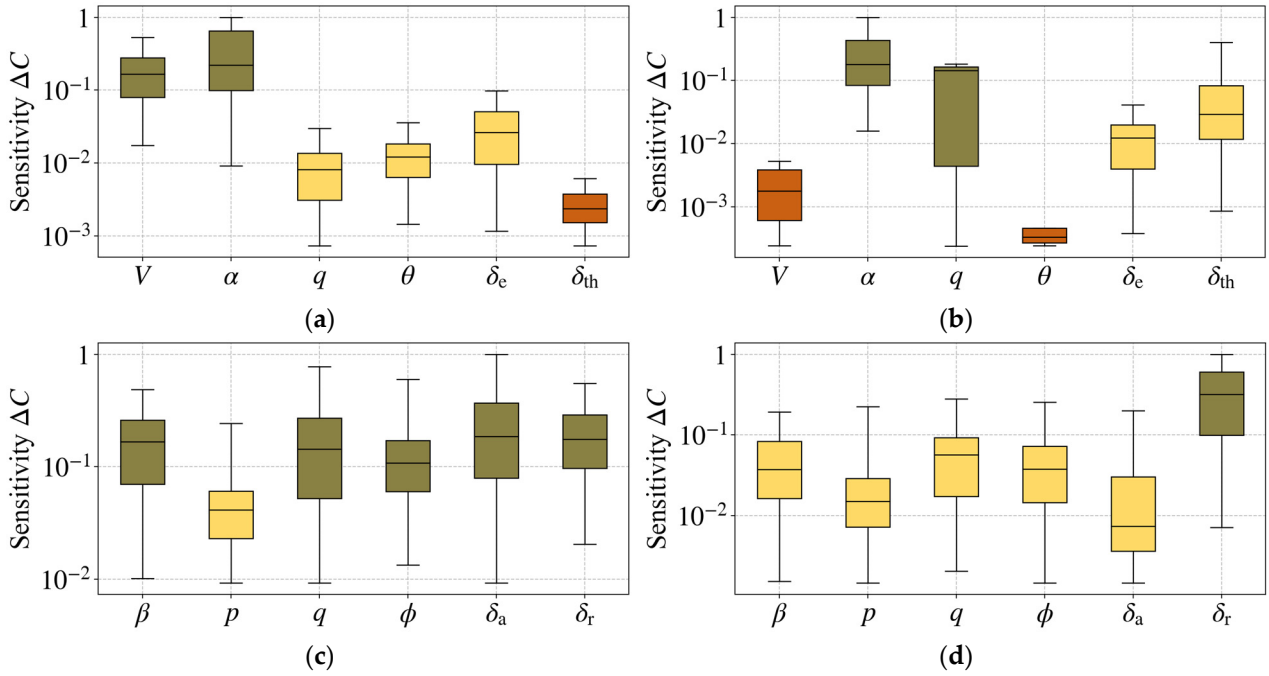

Figure 17. Sensitivity analysis on TRCC commands: (**a**) elevator; (**b**) throttle; (**c**) aileron; (**d**) rudder.

In the sensitivity analysis, Welch's *t*-tests at a 5% significance level were used to evaluate the differences in average sensitivity ΔC. Figure 17a shows that elevator compensation control had the highest sensitivity to nominal velocity and angle-of-attack, with the average sensitivity at least one order of magnitude higher than other states and controls. Similarly, Figure 17b demonstrates the significant sensitivity of throttle compensation control to the nominal angle-of-attack and pitch angle velocity. Figure 17c indicates comparable average sensitivity results for aileron compensation control across lateral nominal states and controls, except for the roll angle velocity. Lastly, Figure 17d reveals the substantial sensitivity of rudder compensation control to nominal rudder deflection, exceeding other nominal states and controls by at least one order of magnitude.

5. Conclusions

This study introduced a method for calculating a discrete trajectory tube and developed a parallel TRCC algorithm that enhanced robustness by minimizing the tube. Additionally, two efficient methods for solving TRCC commands for UAVs were presented: dimensionality reduction techniques (decoupling and stepwise) and control reuse. Through runtime simulations under external disturbances and dynamic uncertainties, it was verified that the proposed fast TRCC effectively enhances the robustness of UAVs during tracking tasks while satisfying the requirements for fast solving, as supported by the following two key points:

1. The TRCC method for UAVs reduces the RMS tracking error by 66% compared to the uncompensated control, significantly enhancing robustness during maneuver trajectory tracking.
2. By utilizing one-step reuse, UAV decoupling, and a three-step algorithm, the TRCC fast generation requirement of 50 ms per beat can be achieved in a 16-thread environment. Simulations demonstrate that the fast TRCC method reduces the RMS tracking error by 60% compared to the uncompensated control. However, when subjected to the same range of disturbances, the fast TRCC exhibits an approximate 9% increase in the RMS tracking error compared to the slow nominal TRCC, indicating lower robustness.

In addition, the fast-solving method for trajectory tubes proposed in this paper enables real-time assessment of robustness in nonlinear UAVs. In future work, its application can extend beyond UAV control to benefit trajectory planning and situation assessment, addressing disturbances in practical scenarios.

Author Contributions: Conceptualization, L.W. and S.Z.; methodology, L.W. and S.Z.; software, S.Z., W.W. and H.W.; validation, L.W., S.Z. and T.Y.; formal analysis, H.L., W.W. and H.W.; investigation, T.Y.; resources, H.L., W.W. and H.W.; data curation, H.L.; writing—original draft preparation, S.Z.; writing—review and editing, L.W.; visualization, S.Z.; supervision, T.Y.; project administration, H.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Fundamental Research Funds for Central Universities, China (Funding number: YWF-23-SDHK-L-011).

Data Availability Statement: Data are available on request from the authors.

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

References

1. Wang, M.; Wang, L.; Yue, T.; Liu, H. Influence of unmanned combat aerial vehicle agility on short-range aerial combat effectiveness. *Aerosp. Sci. Technol.* **2020**, *96*, 105534. [CrossRef]
2. Pereira, L.A.A.; Pimenta, L.C.A.; Raffo, G.V. MPC Based Feedback-Linearization Strategy of a Fixed-Wing UAV. *Automática* **2020**, *2*, 1. [CrossRef]
3. Pfeifle, O.; Fichter, W. Cascaded incremental nonlinear dynamic inversion for three-dimensional spline-tracking with wind compensation. *J. Guid. Control Dyn.* **2021**, *44*, 1559–1571. [CrossRef]
4. Olguin-Roque, J.; Salazar, S.; González-Hernandez, I.; Lozano, R. A Robust Fixed-Time Sliding Mode Control for Quadrotor UAV. *Algorithms* **2023**, *16*, 229. [CrossRef]
5. Wang, Q.; Wang, W.; Suzuki, S.; Namiki, A.; Liu, H.; Li, Z. Design and implementation of UAV velocity controller based on reference model sliding mode control. *Drones* **2023**, *7*, 130. [CrossRef]
6. Mothes, F. Trajectory planning in time-varying adverse weather for fixed-wing aircraft using robust model predictive control. *Aerospace* **2019**, *6*, 68. [CrossRef]
7. Melo, A.G.; Andrade, F.A.A.; Guedes, I.P.; Carvalho, G.F.; Zachi, A.R.L.; Pinto, M.F. Fuzzy gain-scheduling PID for UAV position and altitude controllers. *Sensors* **2022**, *22*, 2173. [CrossRef] [PubMed]
8. Alsaade, F.W.; Jahanshahi, H.; Yao, Q.; Al-Zahrani, M.S.; Alzahrani, A.S. A new neural network-based optimal mixed H2/H∞ control for a modified unmanned aerial vehicle subject to control input constraints. *Adv. Space Res.* **2023**, *71*, 3631–3643. [CrossRef]
9. Wan, K.; Gao, X.; Hu, Z.; Wu, G. Robust motion control for UAV in dynamic uncertain environments using deep reinforcement learning. *Remote Sens.* **2020**, *12*, 640. [CrossRef]
10. Fu, Y.; Zhao, W.; Liu, L. Safe Reinforcement Learning for Transition Control of Ducted-Fan UAVs. *Drones* **2023**, *7*, 332. [CrossRef]
11. Popov, A.M.; Kostrygin, D.G.; Shevchik, A.A.; Andrievsky, B. Speed-Gradient Adaptive Control for Parametrically Uncertain UAVs in Formation. *Electronics* **2022**, *11*, 4187. [CrossRef]
12. Lungu, M. Auto-landing of UAVs with variable centre of mass using the backstepping and dynamic inversion control. *Aerosp. Sci. Technol.* **2020**, *103*, 105912. [CrossRef]
13. Wang, Q.; Namiki, A.; Asignacion, A., Jr.; Li, Z.; Suzuki, S. Chattering Reduction of Sliding Mode Control for Quadrotor UAVs Based on Reinforcement Learning. *Drones* **2023**, *7*, 420. [CrossRef]
14. Cao, S.; Shen, L.; Zhang, R.; Yu, H.; Wang, X. Adaptive incremental nonlinear dynamic inversion control based on neural network for UAV maneuver. In Proceedings of the 2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), Hong Kong, China, 8–12 July 2019; IEEE: Tampa, FL, USA, 2019. [CrossRef]
15. Liu, H.; Lu, G.; Zhong, Y. Robust LQR attitude control of a 3-DOF laboratory helicopter for aggressive maneuvers. *IEEE Trans. Ind. Electron.* **2012**, *60*, 4627–4636. [CrossRef]
16. Liu, H.; Wang, X.; Zhong, Y. Quaternion-based robust attitude control for uncertain robotic quadrotors. *IEEE Trans. Ind. Inform.* **2015**, *11*, 406–415. [CrossRef]
17. Mayne, D.Q.; Seron, M.M.; Raković, S.V. Robust model predictive control of constrained linear systems with bounded disturbances. *Automatica* **2005**, *41*, 219–224. [CrossRef]
18. Raković, S.V.; Kouvaritakis, B.; Findeisen, R.; Cannon, M. Homothetic tube model predictive control. *Automatica* **2012**, *48*, 1631–1638. [CrossRef]
19. Raković, S.V.; Kouvaritakis, B.; Cannon, M.; Panos, C.; Findeisen, R. Parameterized tube model predictive control. *IEEE Trans. Autom. Control* **2012**, *57*, 2746–2761. [CrossRef]
20. Mayne, D.Q.; Kerrigan, E.C.; Van Wyk, E.J.; Falugi, P. Tube-based robust nonlinear model predictive control. *Int. J. Robust Nonlinear Control* **2011**, *21*, 1341–1353. [CrossRef]

21. Moore, J.; Cory, R.; Tedrake, R. Robust post-stall perching with a simple fixed-wing glider using LQR-Trees. *Bioinspir. Biomim.* **2014**, *9*, 025013. [CrossRef]
22. Majumdar, A.; Ahmadi, A.A.; Tedrake, R. Control design along trajectories with sums of squares programming. In Proceedings of the 2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, 6–10 May 2013; IEEE: Tampa, FL, USA, 2013. [CrossRef]
23. Tobenkin, M.M.; Manchester, I.R.; Tedrake, R. Invariant funnels around trajectories using sum-of-squares programming. *IFAC Proc. Vol.* **2011**, *44*, 9218–9223. [CrossRef]
24. Herbert, S.L.; Bansal, S.; Ghosh, S.; Tomlin, C.J. Reachability-based safety guarantees using efficient initializations. In Proceedings of the 2019 IEEE 58th Conference on Decision and Control (CDC), Nice, France, 11–13 December 2019; IEEE: Tampa, FL, USA, 2019. [CrossRef]
25. Seo, H.; Son, C.Y.; Kim, H.J. Fast funnel computation using multivariate Bernstein polynomial. *IEEE Robot. Autom. Lett.* **2021**, *6*, 1351–1358. [CrossRef]
26. Cunis, T.; Legat, B. Sequential sum-of-squares programming for analysis of nonlinear systems. In Proceedings of the 2023 American Control Conference (ACC), San Diego, CA, USA, 31 May–2 June 2013; IEEE: Tampa, FL, USA, 2013. [CrossRef]
27. Majumdar, A.; Tedrake, R. Funnel libraries for real-time robust feedback motion planning. *Int. J. Robot. Res.* **2017**, *36*, 947–982. [CrossRef]
28. Prajna, S.; Papachristodoulou, A.; Parrilo, P.A. Introducing SOSTOOLS: A general purpose sum of squares programming solver. In Proceedings of the 41st IEEE Conference on Decision and Control, Las Vegas, NV, USA, 10–13 December 2002; IEEE: Tampa, FL, USA, 2002. [CrossRef]
29. Mitchell, I.M.; Bayen, A.M.; Tomlin, C.J. A time-dependent Hamilton-Jacobi formulation of reachable sets for continuous dynamic games. *IEEE Trans. Autom. Control* **2005**, *50*, 947–957. [CrossRef]
30. Waldspurger, I.; d'Aspremont, A.; Mallat, S. Phase recovery, maxcut and complex semidefinite programming. *Math. Program.* **2015**, *149*, 47–81. [CrossRef]
31. Papachristodoulou, A.; Anderson, J.; Valmorbida, G.; Prajna, S.; Seiler, P.; Parrilo, P.; Peet, M.M.; Jagt, D. SOSTOOLS version 4.00 sum of squares optimization toolbox for MATLAB. *arXiv* **2013**, arXiv:1310.4716.
32. Sturm, J.F. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optim. Methods Softw.* **1999**, *11*, 625–653. [CrossRef]
33. El-kenawy, E.S.M.; Ibrahim, A.; Bailek, N.; Bouchouicha, K.; Hassan, M.A.; Jamei, M.; Al-Ansari, N. Sunshine duration measurements and predictions in Saharan Algeria region: An improved ensemble learning approach. *Theor. Appl. Climatol.* **2022**, *147*, 1015–1031. [CrossRef]