

## Article

# Improved Nonlinear Model Predictive Control Based Fast Trajectory Tracking for a Quadrotor Unmanned Aerial Vehicle

Hongyue Ma <sup>1</sup>, Yufeng Gao <sup>2</sup>, Yongsheng Yang <sup>1</sup> and Shoulin Xu <sup>1,\*</sup><sup>1</sup> Institute of Logistics Science and Engineering, Shanghai Maritime University, Shanghai 201306, China<sup>2</sup> Beijing Institute of Spacecraft System Engineering, Beijing 100194, China; yfgao\_hit@163.com

\* Correspondence: slxu1@shmtu.edu.cn

**Abstract:** This article studies a nonlinear model predictive control (NMPC) scheme for the trajectory tracking efficiency of a quadcopter UAV. A cost function is first proposed that incorporates weighted increments of control forces in each direction, followed by a weighted summation. Furthermore, a contraction constraint for the cost function is introduced based on the numerical convergence of the system for the sampling period of the UAV control force. Then, an NMPC scheme based on improved continuous/generalized minimum residuals (C/GMRES) is proposed to obtain acceptable control performance and reduce computational complexity. The proposed control scheme achieves efficient and smooth tracking control of the UAV while guaranteeing the closed-loop stability of the system. Finally, simulation results are presented to illustrate the effectiveness and superior performance of the proposed NMPC control scheme.

**Keywords:** continuation/generalized minimal residual (C/GMRES); nonlinear model predictive control (NMPC); trajectory tracking



**Citation:** Ma, H.; Gao, Y.; Yang, Y.; Xu, S. Improved Nonlinear Model Predictive Control Based Fast Trajectory Tracking for a Quadrotor Unmanned Aerial Vehicle. *Drones* **2024**, *8*, 387. <https://doi.org/10.3390/drones8080387>

Academic Editor: Mostafa Hassanalain

Received: 26 June 2024

Revised: 6 August 2024

Accepted: 7 August 2024

Published: 9 August 2024



**Copyright:** © 2024 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

Recently, unmanned aerial vehicles (UAVs) have aroused great interest and attention in civilian fields, including aerial photography, cargo transportation, and rescue missions [1–4]. Many studies have been conducted to explore the motion control of UAVs. The trajectory tracking problem of UAVs is one of the typical motion control problems. In particular, a UAV is a 6-degree of freedom, nonlinear, underdriven, strongly coupled system. Hence, the design of a stable and efficient trajectory-tracking controller holds high practical significance [5].

As is well known, PID controllers remain a widely utilized class of algorithms. Gao et al. [6] used cascaded PID control with a Kalman filter to remove noise. However, parameter tuning often needs more comprehensive theoretical guidance and is time-consuming. Robust control can address parameter uncertainty [7,8]. Although it effectively suppresses external disturbances during the control process, its exclusive focus on worst-case scenarios may diminish overall control performance. Sliding mode control exhibits reduced dependency on parameters and demonstrates rapid responsiveness, as the state trajectory swiftly reaches and then remains on the sliding surface in finite time [9]. Li et al. [10] designed a finite-time extended state observer based on terminal sliding mode to estimate and compensate for system faults or uncertainties in the aggregate system. Reference [11] designed a new sliding mode observer based on specified fixed-time sliding mode variables and adaptive techniques to estimate external disturbances within a specified stable time. Huang et al. [12] proposed a non-singular terminal sliding mode control method to address errors and external disturbances in UAV modeling. Using disturbance observers, they effectively mitigated vibration issues commonly associated with sliding mode control. Although these algorithms can achieve optimal control strategies [13], the challenge lies in their computational burden, making it

difficult to design and implement. Model predictive control (MPC) is an optimal feedback control technology. MPC can iteratively compute optimal control inputs using a rolling optimization strategy, while also satisfying nonlinear/linear constraints on system states and control increments of UAVs.

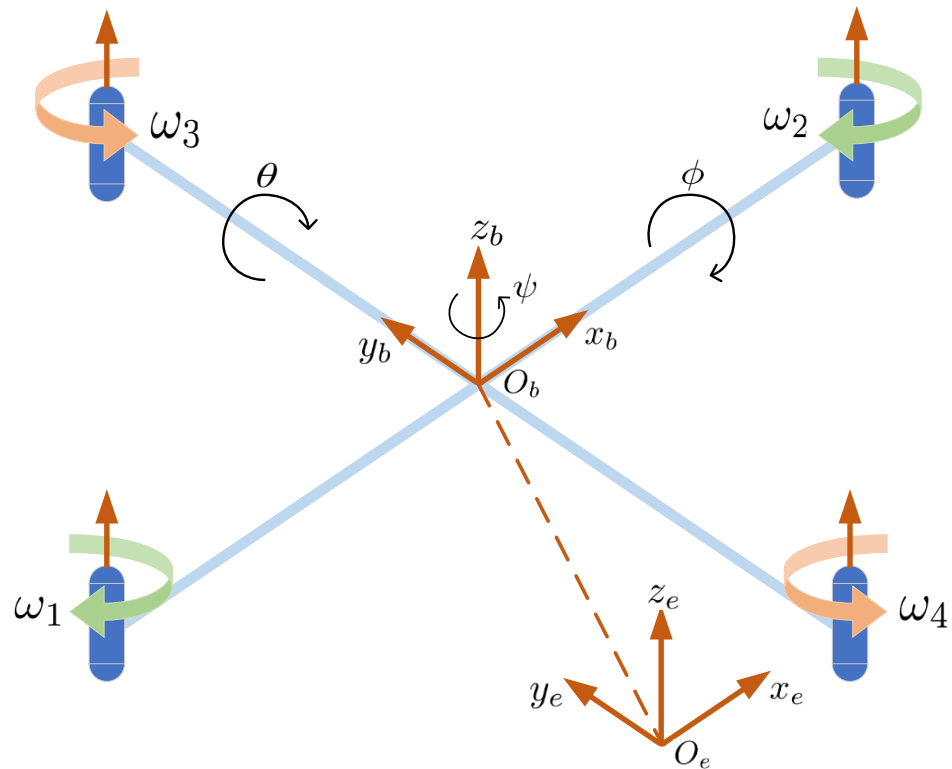
References [14–17] conducted research on MPC in the context of trajectory tracking. Wei et al. [18] developed a distributed MPC using Lyapunov theory, which allowed an autonomous underwater vehicle to maintain a desired formation and track a reference trajectory, even in the presence of external disturbances. Mohammadi et al. [19] utilized linear MPC to track the trajectory of the UAV on a ground platform, which was a straight line, and the direction of the UAV remained essentially constant. However, the limitations of linear MPC become apparent when the tracked trajectory deviates from a straight line. Vlanis et al. [20] designed an NMPC controller that considers real-time constraints, including the UAV's flight capability and collision avoidance requirements. The controller solves the constraint optimization problem online, ensuring the UAV operates effectively and safely. Pedro et al. [21] proposed a multi-model solution-based for altitude and yaw control of a quadrotor transporting an unknown constant load. Yan et al. [22] developed a multi-constraint MPC control strategy by integrating a Kalman-consistent filter with a fixed-time perturbation observer. This integration aims to enhance the applicability and robustness of the control algorithm in complex environments and improve computational efficiency. Zhang et al. [23] proposed a self-triggered adaptive MPC approach for constrained discrete-time nonlinear systems, effectively addressing challenges related to parameter uncertainties and additive disturbances. Said et al. [24] designed two optimal controllers based on linear quadratic Gaussian and MPC algorithms for a quadrotor, enabling it to autonomously track a moving platform from any initial position and achieve precise landings. Wu et al. [25] proposed a synthesis method for generating collision avoidance trajectories by categorizing trajectory planning into macro-planning and micro-planning and using model prediction to control the final arrival at the specified position within the target area.

The above analysis needs to consider improving computational efficiency and accuracy within the constraints of actual physical conditions. It also needs to address the extensive number of iterations required to solve the optimization problem. The continuous/generalized minimum residuals (C/GMRES) algorithm computes optimal parameter estimates by minimizing the residual norm. It continuously updates parameter estimates and adjusts them to minimize residuals, thereby achieving a better fit to the data [26]. However, due to its inability to handle inequality constraints, C/GMRES cannot be used in isolation. Toshiyuki [27] proposed a program based on the standard C/GMRES algorithm for automatically generating numerical solutions. Shen et al. [28] introduced a modified C/GMRES (mC/GMRES) that uses initial values based on a logarithmic barrier function, facilitating rapid convergence to the optimum. Nonetheless, this approach did not consider terminal state constraints. Wang et al. [29] incorporated a penalty term into the cost function while considering the initial optimum and analyzing its convergence and stability. However, their model was specific to certain instances and lacked applicability in practical scenarios. To address these limitations, we propose an improved NMPC solution.

- (1) Considering the actual quadrotor model, we propose an improved NMPC control scheme to achieve more precise and rapid position tracking. This improvement aims to enhance the generality and applicability control.
- (2) Compared with mC/GMRES in [28], this study enhances control stability and computational efficiency in NMPC by incorporating a weighted sum of control increments into the cost function to solve the new optimization problem.
- (3) The Lyapunov-based shrinkage constraints are added according to the optimal solution, and the set of two stabilization domains is found to ensure numerical convergence and trajectory stability throughout the entire simulation process.

## 2. System Description

The design of a quadrotor UAV involves both position and attitude control. The structure of the quadrotor is illustrated in Figure 1. The origin  $O_b$  is the center of the quadrotor UAV. The roll and pitch angles  $(\phi, \theta)$  are coupled to the global position  $(x, y)$ . To move in the  $x_e$  or  $y_e$  direction, it is usually necessary to adjust the pitch or roll angle  $(\theta, \phi)$ , while the yaw angle  $(\psi)$  changes the direction of the UAV's travel. Additionally, let  $\mathbf{X} = [x, y, z]^T$  represent the center of position of the quadrotor.



**Figure 1.** The structure representation of a quadrotor UAV.

The quadrotor UAV dynamics model is established as follows [13]:

$$m\ddot{\mathbf{X}} + mge_3 = \mathbf{R}(\boldsymbol{\Theta})\mathbf{U} \quad (1)$$

$$\mathbf{J}(\boldsymbol{\Theta})\ddot{\boldsymbol{\Theta}} + \mathbf{C}(\boldsymbol{\Theta}, \dot{\boldsymbol{\Theta}})\dot{\boldsymbol{\Theta}} = \boldsymbol{\tau} \quad (2)$$

with

$$\mathbf{R}(\boldsymbol{\Theta}) = \begin{bmatrix} C\psi C\theta & C\psi S\theta S\phi - S\psi C\phi & C\psi S\theta C\phi + S\psi S\phi \\ S\psi C\theta & S\psi S\theta S\phi + C\psi C\phi & S\psi S\theta C\phi - C\psi S\phi \\ -S\theta & C\theta S\phi & C\theta C\phi \end{bmatrix} \quad (3)$$

where  $e_3 = [0, 0, 1]^T$ ;  $m$  is the mass of the quadrotor UAV;  $g$  is the gravitational constant;  $\mathbf{U} = [0, 0, f]^T$  is the total thrust vector;  $\boldsymbol{\tau} \in \mathbb{R}^3$  is the moment input by the attitude system.  $S \cdot$  is the sine function  $\sin(\cdot)$  and  $C \cdot$  is the cosine function  $\cos(\cdot)$ . The positive definite diagonal matrix  $\mathbf{I} = \text{diag}([I_{11}, I_{22}, I_{33}])$ . The symmetry matrix  $\mathbf{J}(\boldsymbol{\Theta}) \in \mathbb{R}^{3 \times 3}$  and matrix  $\mathbf{C}(\boldsymbol{\Theta}, \dot{\boldsymbol{\Theta}}) \in \mathbb{R}^{3 \times 3}$  are given by

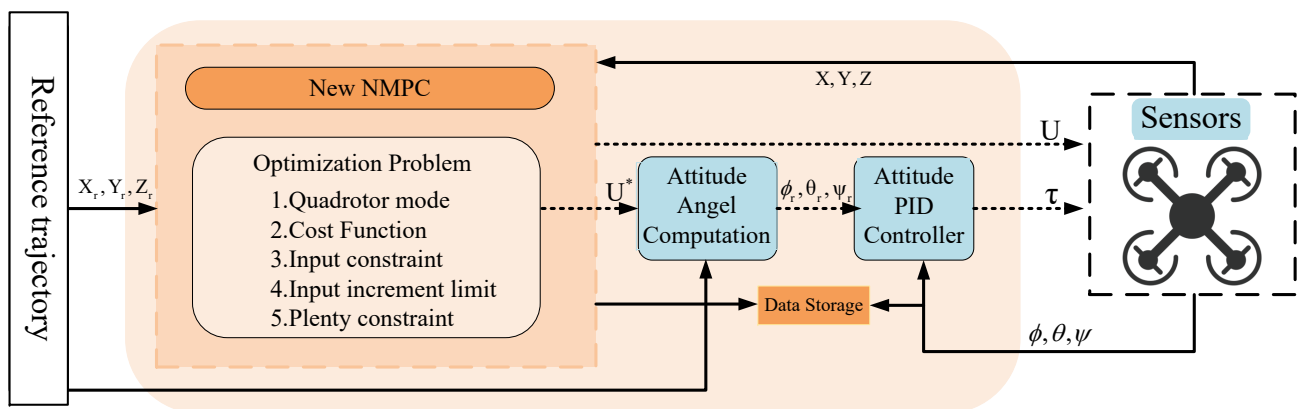
$$\mathbf{J}(\boldsymbol{\Theta}) = \begin{bmatrix} I_{11} & 0 & -I_{11}S\theta \\ 0 & I_{22}C^2\phi + I_{33}S^2\phi & (I_{22} - I_{33})C\phi S\phi C\theta \\ -I_{11}S\theta & (I_{22} - I_{33})C\phi S\phi C\theta & I_{11}S^2\theta + I_{22}S^2\phi C^2\theta + I_{33}C^2\phi C^2\theta \end{bmatrix} \quad (4)$$

$$C(\Theta, \dot{\Theta}) = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix} \quad (5)$$

with

$$\begin{aligned} c_{11} &= 0, \\ c_{12} &= (I_{22} - I_{33}) (\dot{\theta} C \phi S \phi + \dot{\psi} S^2 \phi C \theta - \dot{\psi} C^2 \phi C \theta) - I_{11} \dot{\psi} C \theta, \\ c_{13} &= (I_{22} - I_{33}) \dot{\psi} C \phi S \phi C^2 \theta, \\ c_{21} &= (I_{33} - I_{22}) (\dot{\theta} C \phi S \phi + \dot{\psi} S^2 \phi C \theta - \dot{\psi} C^2 \phi C \theta) + I_{11} \dot{\psi} C \theta, \\ c_{22} &= (I_{33} - I_{22}) \dot{\phi} C \phi S \phi, \\ c_{23} &= -I_{11} \dot{\psi} S \theta C \theta + I_{22} \dot{\psi} S^2 \phi C \theta S \theta + I_{33} \dot{\psi} C^2 \phi S \theta C \theta, \\ c_{31} &= (I_{33} - I_{22}) \dot{\psi} C \phi S \phi S \theta - I_{11} \dot{\theta} C \theta, \\ c_{32} &= (I_{33} - I_{22}) (\dot{\psi} C \phi S \phi S \theta - \dot{\phi} S^2 \phi C \theta - \dot{\phi} C^2 \phi C \theta) + I_{11} \dot{\psi} S \theta C \theta \\ &\quad - I_{22} \dot{\psi} S^2 \phi C \theta S \theta - I_{33} \dot{\psi} C \phi S \theta C \theta, \\ c_{33} &= (I_{33} - I_{22}) \dot{\phi} C \phi S \phi C^2 \theta - I_{22} \dot{\theta} S^2 \phi C \theta S \theta - I_{33} \dot{\theta} C^2 \phi S \theta C \theta \\ &\quad + I_{11} \dot{\theta} S \theta C \theta. \end{aligned} \quad (6)$$

This paper addresses the trajectory tracking problem for UAVs using NMPC with the dynamic model of the UAV described above. To ensure the UAV's control smoothness requirements, control increments are weighted in the cost function. Thus,  $\Delta u = u_i - u_{i-1} \rightarrow 0$  ( $i = 1, 2, 3, \dots$ ) is reached when  $t \rightarrow \infty$ . The C/GMRES method is employed to reduce the computational burden of NMPC. The barrier methods are used to transform the original optimization problem into an unconstrained one. Additionally, the algorithm's attraction domain intersects with a Lyapunov-based control method to ensure the system's closed-loop stability. The system is divided into inner-loop and outer-loop control, as shown in Figure 2. The outer-loop position controller uses the improved C/GMRES method to efficiently solve the nonlinear optimization problem, allowing for real-time adjustment of the system's position and ensuring accurate trajectory tracking. The inner-loop attitude controller uses a PID control strategy to quickly adjust the attitude and stabilize the system's flight state.



**Figure 2.** The schematic of the control strategy.

### 3. Trajectory Tracking Controller Design

#### 3.1. Brief Description of NMPC Controller

Assuming that the reference trajectory is appropriate and does not violate the physical limits of the UAV [30]. We consider both the UAV's force constraints and the constraints on the rate of change simultaneously. The cost function is constructed based on the UAV's position dynamics model as follows:

$$\begin{aligned} \min_{\hat{\mathbf{u}}(\cdot)} \int_t^{t+T} & \|\hat{\mathbf{x}}(s)\|_Q^2 + \|\hat{\mathbf{u}}(s)\|_R^2 ds + \|\hat{\mathbf{x}}(t+T)\|_P^2 \\ \text{s.t. } & \dot{\hat{\mathbf{x}}}(s) = f(\hat{\mathbf{x}}(s), \hat{\mathbf{u}}(s)), \quad s \in [t, t+T) \\ & \hat{\mathbf{x}}(t) = \mathbf{x}(t), \quad \hat{\mathbf{u}}(t) = \mathbf{u}(t) \\ & \|\hat{\mathbf{u}}(s)\|_\infty \leq u_{\max}, \quad s \in [t, t+T) \end{aligned} \quad (7)$$

where  $T$  is the prediction horizon;  $\hat{\mathbf{x}}(s)$  represents the current state;  $\mathbf{x}_r(s)$  represents the reference state;  $\hat{\mathbf{u}}(t)$  represents the control sequence;  $\hat{\mathbf{x}}(s) = \hat{\mathbf{x}}(s) - \mathbf{x}_r(s)$  represents the error in predicted tracking;  $u_{\max}$  and  $\Delta u_{\max}$  denote the maximum magnitudes of control input and control variations, respectively;  $Q, R, S$  are positive definite weighting matrices.  $\|\hat{\mathbf{x}}(s)\|_Q^2 = \hat{\mathbf{x}}^T(s)Q\hat{\mathbf{x}}(s)$ ,  $\|\hat{\mathbf{u}}(s)\|_R^2 = \hat{\mathbf{u}}^T(s)R\hat{\mathbf{u}}(s)$  and  $\|\hat{\mathbf{x}}(t+T)\|_P^2 = \hat{\mathbf{x}}^T(t+T)P\hat{\mathbf{x}}(t+T)$ .

The C/GMRES-based method is computed and implemented using a discrete-time formulation but remains based on continuous time  $t$ . Therefore, the discretization problem must be addressed at each continuous time  $t$ , ensuring the stability and accuracy of the discretization process. Assuming a prediction interval of  $N$  steps, the step size is  $\delta = T/N$ , which is referred to as the sampling period. The discretization of the above equation is as follows:

$$\begin{aligned} P_1(\mathbf{x}) : \min_{\hat{\mathbf{u}}_i} J &= \sum_{i=0}^{N-1} \left( \|\tilde{\mathbf{x}}_i\|_Q^2 + \|\hat{\mathbf{u}}_i\|_R^2 \right) \delta + \|\tilde{\mathbf{x}}_N\|_P^2 \\ \text{s.t. } & \dot{\hat{\mathbf{x}}}(s) = f(\hat{\mathbf{x}}(s), \hat{\mathbf{u}}(s)), \quad s \in [t, t+T) \\ & \hat{\mathbf{x}}(t) = \mathbf{x}(t), \quad \hat{\mathbf{u}}(t) = \mathbf{u}(t) \\ & \|\hat{\mathbf{u}}(s)\|_\infty \leq u_{\max}, \quad s \in [t, t+T) \end{aligned} \quad (8)$$

with the guarantee that the system is differentiable and the constraints hold, the Karush–Kuhn–Tucker (KKT) condition [31] provides a first-order necessary condition for the optimal solution  $\mathbf{U}^*$  of the optimization problem  $P_1(\mathbf{x})$ . Sequential quadratic programming and the interior point method can be used to solve the KKT condition for  $P_1(\mathbf{x})$ . However, these methods rely on continuous linearization. To address this, the C/GMRES numerical algorithm will be used to speed up the numerical computation of NMPC, considering both the onboard computational burden and computational efficiency. The standard C/GMRES can only solve optimization problems with equality constraints. Then, we relax this restriction and propose an improved strategy to compute the optimal solution under the inequality constraints, which leads to a smoother and globally stabilized control process.

#### 3.2. Improved Efficient NMPC

This section introduces a smoothing constraint on the control increment and a contraction constraint to improve the smoothness of the control process and ensure global convergence. Finally, an efficient NMPC algorithm is proposed to ensure that the solution obtained during the optimization process satisfies the conditions and constraints.

##### 3.2.1. Smoothness Constraint Term for Control Increments

The rate of change of the control input must be considered in the actual control process. Otherwise, excessive changes in the control strategy can lead to violent system oscillations.

In addition, the optimization algorithm might produce significant control input variations between successive steps, which could result in an unstable system response.

Firstly, the control sequence is defined as  $\{\hat{\mathbf{u}}_i^*\}_{i=0}^N$ , where  $\mathbf{u}_i$  is the  $i$ th control input component at the current time step, and  $\mathbf{u}_{i-1}$  is the corresponding control input component at the previous time step. The sum of squares of control increments is introduced to address potential numerical instability in the search for an optimal solution to an optimization problem. When the control inputs change significantly between time steps, the penalty function penalizes these changes. This function finds a solution that satisfies the constraints during the optimization process. As a result, the final solution meets the requirements. The constraint with the penalty term is then defined as follows:

$$\vartheta(\hat{\mathbf{u}}_i) = \rho \cdot \sum_{i=1}^N (\Delta \hat{\mathbf{u}}_i)^2 \quad (9)$$

where  $\Delta \hat{\mathbf{u}}_i = \hat{\mathbf{u}}_i^* - \hat{\mathbf{u}}_{i-1}^*$ , and the penalty factor  $\rho$  is introduced to emphasize the impact of  $\vartheta(\hat{\mathbf{u}}_i)$ . A larger value of  $\rho$  causes  $\vartheta(\hat{\mathbf{u}}_i)$  to  $\Delta \hat{\mathbf{u}}_i$  more significantly, while a smaller  $\rho$  reduces this penalty. The parameter  $\rho$  is introduced to balance the attainment of an optimal solution with the satisfaction of constraints, thereby preventing the imposition of excessively stringent conditions. To facilitate the optimization process, it is typically necessary to fine-tune the value of  $\rho$ , ensuring a more effective and efficient solution. The modified optimal control problem (OCP) ( $P_2(\mathbf{x})$ ), with the penalty term incorporated into the cost function, can be formulated as follows:

$$\begin{aligned} P_2(\mathbf{x}) : \min_{\hat{\mathbf{u}}_i} J &= \sum_{i=0}^{N-1} \left( \|\hat{\mathbf{x}}_i\|_Q^2 + \|\hat{\mathbf{u}}_i\|_R^2 + \varepsilon \cdot \vartheta(\hat{\mathbf{u}}_i) \right) \delta + \|\hat{\mathbf{x}}_N\|_P^2 \\ \text{s.t. } \hat{\mathbf{x}}_{i+1} &= f(\hat{\mathbf{x}}_i, \hat{\mathbf{u}}_i) + \hat{\mathbf{x}}_i \\ \hat{\mathbf{x}}_0 &= \mathbf{x}(t) \\ \|\hat{\mathbf{u}}(s)\|_\infty &\leq \mathbf{u}_{\max}, \quad s \in [t, t+T) \end{aligned} \quad (10)$$

where  $\vartheta(\hat{\mathbf{u}}_i)$  represents a soft constraint on the control input  $\Delta \hat{\mathbf{u}}_i$ . It can moderately violate the constraints in the optimization process, making it more flexible. Especially when the constraints are difficult to handle, a feasible solution can be obtained by appropriately adjusting the strength of the penalty term, thus adapting to the different needs of the problem. The impact of different parameter values  $\varepsilon$  in (10) on numerical convergence should be discussed. Utilizing the C/GMRES method for approximate computation of solutions in  $P_2(\mathbf{x})$ , the following assumptions and lemmas are made within the context of practical applications:

**Assumption 1** ([32]). For each  $\varepsilon = \varepsilon_d > 0$ , there exists a bounded solution  $\mathbf{U}_n$ .

The C/GMRES method is based on the conditions required for solving Pontryagin's Maximum Principle (PMP) to attain the optimal solution for  $P_2(\mathbf{x})$ , rather than the KKT conditions. Therefore, the sufficiency of PMP for addressing the KKT conditions is explored first.

**Lemma 1** ([28]). A point satisfies the PMP condition of  $P_2(\mathbf{x})$ , sufficiently solving its KKT conditions.

**Proof.** The proof of Lemma 1 is presented in Appendix A.1.  $\square$

**Lemma 2.** Assume Assumption 1 holds, and  $0 < \varepsilon_{n+1} < \varepsilon_n$  is a solution to  $P_2(\mathbf{x})$  with optimal solutions  $\bar{\mathbf{U}}_{n+1}$  and  $\bar{\mathbf{U}}_n$ . Then, the following inequalities can hold as follows:

$$\begin{aligned} \bar{J}(\bar{\mathbf{U}}_{n+1}, \varepsilon_{n+1}) &\leq \bar{J}(\bar{\mathbf{U}}_n, \varepsilon_n) \\ \bar{J}(\bar{\mathbf{U}}_{n+1}) &\geq \bar{J}(\bar{\mathbf{U}}_n), \quad \vartheta(\bar{\mathbf{U}}_n) \geq \vartheta(\bar{\mathbf{U}}_{n+1}) \end{aligned}$$

**Proof.** The proof of Lemma 2 is presented in Appendix A.2.  $\square$

**Lemma 3** ([29]).  $\{\varepsilon_n\}_{n=1}^N$  is a decreasing sequence satisfying  $0 < \varepsilon_{n+1} < \varepsilon_n$ , and when  $n \rightarrow +\infty$ , there exists  $\varepsilon_n$  sufficiently small. Based on Assumption 1,  $\mathbf{U}_n$  converges to  $\mathbf{U}^*$ .

**Lemma 4** ([29]). Consider Lemmas 1 and 3.  $P_2(\mathbf{x})$  is the optimal solution. Then, the above approximately solves  $P_1(\mathbf{x})$  with a sufficiently little  $\varepsilon$  in  $P_2(\mathbf{x})$ .

### 3.2.2. C/GMRES Algorithm

For  $P_2(\mathbf{x})$ , the Hamiltonian function is defined as follows:

$$H(\mathbf{x}, \mathbf{u}, \boldsymbol{\kappa}) = \|\mathbf{x} - \mathbf{x}_r\|_Q^2 + \|\mathbf{u}\|_R^2 + \boldsymbol{\vartheta}(\mathbf{u}, \mathbf{u}_{\max}) + \boldsymbol{\kappa}^T f(\mathbf{x}, \mathbf{u})$$

where  $\boldsymbol{\kappa}$  represents the costate [28], and  $\{\hat{\mathbf{u}}_i^*\}_{i=0}^{N-1}$  is the global optimal solution to  $P_2(\mathbf{x})$ . The costate sequence  $\{\boldsymbol{\kappa}_i\}_{i=1}^N$  satisfies  $\hat{\mathbf{x}}_{i+1} = f(\hat{\mathbf{x}}_i, \hat{\mathbf{u}}_i) + \hat{\mathbf{x}}_i$  and  $\hat{\mathbf{x}}_0 = \mathbf{x}(t)$ , and the following conditions:

$$\hat{\mathbf{x}}_{i+1} = \mathbf{x}_i + f(\mathbf{x}_i, \mathbf{u}_i)\delta \quad (11)$$

$$\boldsymbol{\kappa}_N = \frac{\partial \|\tilde{\mathbf{x}}_N\|_P^2}{\partial \mathbf{x}} \quad (12)$$

$$\boldsymbol{\kappa}_i = \boldsymbol{\kappa}_{i+1} + \mathbf{H}_x(\hat{\mathbf{x}}_i, \hat{\mathbf{u}}_i^*, \boldsymbol{\kappa}_{i+1}, \varepsilon)\delta \quad (13)$$

$$\mathbf{H}_u(\hat{\mathbf{x}}_i, \hat{\mathbf{u}}_i^*, \boldsymbol{\kappa}_{i+1}, \varepsilon) = 0 \quad (14)$$

Equations (11) and (13) represent recursive relationships, while  $\hat{\mathbf{x}}_0 = \mathbf{x}(t)$  and (12) serve as boundary conditions.  $\mathbf{H}_x$  and  $\mathbf{H}_u$  are the partial derivatives of the Jacobian matrix  $\mathbf{H}$  with respect to  $\mathbf{x}$  and  $\mathbf{u}$ , respectively. The sequence  $\{\hat{\mathbf{u}}_i^*\}_{i=1}^{N-1}$  is determined by (14). Starting from  $\hat{\mathbf{x}}_0 = \mathbf{x}(t)$  and iterating through (11), equations (12) and (13) are processed in reverse order of time. The state sequence  $\{\hat{\mathbf{x}}_i\}_{i=0}^{N-1}$  and the costate sequence  $\{\boldsymbol{\kappa}_i\}_{i=1}^N$  are explicit functions of  $\hat{\mathbf{U}}^*$ . We then substitute these into (14) to construct the system of equations:

$$F(\hat{\mathbf{U}}^*, \mathbf{x}, \varepsilon) = \begin{bmatrix} \mathbf{H}_u(\hat{\mathbf{x}}_0, \hat{\mathbf{u}}_0^*, \boldsymbol{\kappa}_1, \varepsilon) \\ \mathbf{H}_u(\hat{\mathbf{x}}_1, \hat{\mathbf{u}}_1^*, \boldsymbol{\kappa}_2, \varepsilon) \\ \mathbf{H}_u(\hat{\mathbf{x}}_2, \hat{\mathbf{u}}_2^*, \boldsymbol{\kappa}_3, \varepsilon) \\ \vdots \\ \mathbf{H}_u(\hat{\mathbf{x}}_{N-1}, \hat{\mathbf{u}}_{N-1}^*, \boldsymbol{\kappa}_N, \varepsilon) \end{bmatrix} \quad (15)$$

To solve (15) at time  $t$ , standard approaches involve computing the Jacobian matrix, Hessian matrix, and their inverses using iterative methods. However, this process is computationally expensive and time-consuming. Instead, numerical continuum methods and forward difference generalized minimum residuals (FDGMRES) avoid the need to compute matrix inverses. The FDGMRES method, which combines forward difference approximation combined with GMRES, is described as follows:

$$\dot{\hat{\mathbf{U}}} = \text{FDGMRES}(\hat{\mathbf{U}}, \mathbf{x}, \dot{\mathbf{x}}, \tilde{\hat{\mathbf{U}}}, k_{\max}, \varepsilon) \quad (16)$$

where  $\dot{\mathbf{x}}$  is approximated as  $\dot{\mathbf{x}}(t) = (\mathbf{x}(t + \delta) - \mathbf{x}(t))/\delta$ ;  $k_{\max}$  represents the maximum number of iterations;  $\tilde{\hat{\mathbf{U}}}$  is the initial feasible value;  $\hat{\mathbf{U}}$  is a feasible initial control input sequence [27]. ( $P_2(\mathbf{x})$ ) can be solved efficiently by applying C/GMRES. Algorithm 1 summarizes the FDGMRES algorithm for solving  $\hat{\mathbf{U}}^*$ :

**Algorithm 1:** FDGMRES algorithm for  $\hat{U}^*$ 


---

```

1. Initialize  $t = 0, l = 0, x_0 = x(0), n = n + 1$ ;
2. Get  $\hat{U}$  by  $\|F(\hat{U}_0, x(t_0), 0)\| = 0$ ;
3. Get  $F$  from (14) and partial derivative  $F_d$ ;
4.  $b = A * F - (1/h) * F, \hat{r} = b - F_d(\hat{U}^*, x + h\dot{x}, \hat{U}, \text{zeros}(nX, 1))$ ,  $v1 = \hat{r} / \|\hat{r}\|$ ;
5. while  $k < k_{max}$  do
6.   get  $x_{k+1} = x(t + \delta)$  and  $\dot{x}_k = (x_{k+1} - x_k) / \delta$ ;
7.    $v_{k+1} = F_d(\hat{U}^*, x + h\dot{x}, v_k(nU, N), \text{zeros}(nX, 1))$ ;
8.   For  $j = 1 : 1 : k$ 
9.      $h_{j,k} = v_k^T * v_k$ ;
10.     $v_{k+1} = v_{k+1} - h_{j,k} * v_k$ ;
11.   End for
12.    $h_{k+1,k} = \|v_{k+1}\|$ ;  $v_{k+1} = v_{k+1} / \|v_{k+1}\|$ ;  $e = [1, \text{zeros}(k, 1)]$ ;
13.    $\hat{h}_k = h_k(1 : k + 1, 1 : k)$ ;  $m_k = (\hat{h}_k)^{-1} * \|\hat{r}\| * e$ ;  $\rho = \|\hat{r}\| * e - \hat{h}_k * m_k$ ;
14.   If  $\rho < \text{tol}$ 
15.     break;
16.   End if
17. End While
18.  $\hat{U}^* = \hat{U}^* + v_k * m_k$ ;
19.  $t = t + \delta, k = k + 1$ .

```

---

**3.2.3. Adding Stability Constraints**

Nonetheless, while it is possible to achieve the global optimal solution, the closed-loop stability of the system cannot be assured due to the limited prediction horizon. Therefore, the following solution is needed. Suppose there is a feedback control rule  $u(t) = \zeta(x(t))$ , which in some stable region ensures that the control input  $u$  satisfies a particular constraint and makes the system asymptotically stable at the origin. According to the inverse Lyapunov theorem [33], this means that we can find some specific type of function  $\gamma_i(\cdot)$  ( $i = 1, 2, 3, 4$ ) and a continuously differentiable Lyapunov function  $V(x)$  such that the system satisfies certain inequalities, thus proving its stability.  $\gamma_1\|x\| \leq V(x) \leq \gamma_2\|x\|$ ,  $\frac{\partial V}{\partial x}f(x, \zeta(x)) \leq -\gamma_3\|x\|$ ,  $\left\|\frac{\partial V(x)}{\partial x}\right\| \leq \gamma_4\|x\|$ . For all  $x \in O \in R^n$ , where  $O$  is an open neighborhood of the origin, we denote the region  $\Omega_O \in O$  as the stability region of the closed-loop system under control  $u = \zeta(x)$  [34].

Since the system uses only the first column of the control sequence  $U^*$  obtained from the prediction sequence  $[t, t + \delta]$ , only the first prediction interval is considered.

$$\begin{aligned}
 P_3(x) : \min_{\hat{u}_i} J &= \sum_{i=0}^{N-1} \left( \|\hat{x}_i\|_Q^2 + \|\hat{u}_i\|_R^2 + \varepsilon \cdot \boldsymbol{\theta}(\hat{u}_i) \right) \delta + \|\hat{x}_N\|_P^2 \\
 \text{s.t. } \hat{x}_{i+1} &= f(\hat{x}_i, \hat{u}_i) + \hat{x}_i \\
 \hat{x}_0 &= x(t) \\
 \|\hat{u}(s)\|_\infty &\leq u_{\max}, \quad s \in [t, t + T) \\
 \frac{\partial V}{\partial x}f(x(t), u(t)) &\leq \frac{\partial V}{\partial x}f(\hat{x}(t), \zeta(\hat{x}(t)))
 \end{aligned} \tag{17}$$

where  $V(\cdot)$  is the Lyapunov function and designed in Section 3.2.4;  $\zeta(\cdot)$  is the auxiliary control law based on the Lyapunov function. When  $t \in [t, t + \delta)$ , the optimal solution of the optimization problem is  $U^*$ . The constraints  $\frac{\partial V}{\partial x}f(x(t), u(t)) \leq \frac{\partial V}{\partial x}f(\hat{x}(t), \zeta(\hat{x}(t)))$  ensure that the time derivative  $V(x)$  of the Lyapunov function at time  $t$  is less than or equal to the value of the nonlinear control law  $u = \zeta(x)$ . This holds when the control law is applied in a sample-and-hold fashion within a closed-loop system. Using the state measurements obtained at each synchronized sampling time, it can be demonstrated that the control law designed in this paper inherits the stability and robustness of the Lyapunov-based control law  $\zeta(x)$ . This is true provided that system data loss is not considered [35]. One of the main properties of stability constraints is that they have the same stability domain as the

nonlinear control law  $\zeta(x)$ . That means the system is stable when the sampling time and the upper limit of perturbation are small enough [36,37]. In  $P_3(x)$ , without compromising the stability of the tracking control, a component that adaptively adjusts the prediction horizon is added to  $P_2(x)$ , which reduces the inconvenience caused by adjusting the parameters. Meanwhile, the adaptive adjustment of the prediction horizon can reduce the optimization scale of the NMPC.

### 3.2.4. Lyapunov-Based Auxiliary Control Law Design

The Lyapunov-based control method is widely used in nonlinear UAV systems, as it can demonstrate the global stability of the entire system [36]. Therefore, the fixed-time integral sliding mode control method can be adopted to construct  $\zeta(x)$ . According to [38,39], for system  $\dot{x}(t) = f(x, u)$ , if there exists a Lyapunov function on its domain that satisfies  $\dot{V}(x) \leq -aV(x)^\alpha - bV(x)^\beta$ . Here,  $a, b > 0$  and  $0 < \beta < 1 < \alpha$ . Furthermore, the system is globally fixed-time stable, and the stability time is  $T \leq \frac{1}{a(\alpha-1)} + \frac{1}{b(\beta-1)}$ .

The position loop system can be written as  $\dot{X}_1 = X_2 = \frac{u}{m} - ge_3$ , where  $X_2 = [\dot{x}, \dot{y}, \dot{z}]^T$ . Then, design a virtual controller [40]:  $u_{nom} = -k_1 \text{sig}^{p_1}(X_1) - k_1 \text{sig}^{q_1}(X_1) - k_2 \text{sig}^{p_2}(X_2) - k_2 \text{sig}^{q_2}(X_2)$ . Choose  $k_1, k_2$  to ensure that the second-order polynomial  $\lambda^2 + k_2\lambda + k_1$  is Hurwitz. And  $0 < p_1 < 1, q_1 = 2p_1/(1+p_1), p_2 > 1, q_2 = 2p_2/(1+p_2)$ .

For the error system  $\dot{X}_2 - \ddot{X}_{2r} = \ddot{X}_{2e} = \frac{\zeta}{m} - ge_3 - \ddot{X}_{2r}$ , where  $X_{2r} = [x_d, y_d, z_d]^T$  and  $\ddot{X}_{2r} = [\ddot{x}_d, \ddot{y}_d, \ddot{z}_d]^T$ . Its Lyapunov candidate function satisfies  $\lambda_1 \|x_e\|^2 \leq V(x) \leq \lambda_2 \|x_e\|^2$ , where  $\lambda_1, \lambda_2 \in \mathbb{R}$ . Select the sliding surface as follows:

$$s = X_{2e} - X_{2r}(0) + \int_0^t \left( -\frac{u_{nom}}{m} + ge_3 \right) dt \quad (18)$$

then, taking its derivative yields

$$\dot{s} = \dot{X}_{2e} - \frac{u_{nom}}{m} + ge_3 = \frac{\zeta}{m} + \frac{u_{nom}}{m} \quad (19)$$

**Theorem 1.** Considering the dynamic error system  $\dot{X}_{2e} = \frac{\zeta}{m} - ge_3 - \ddot{X}_{2r}$ , and selecting surface (18) as the sliding surface, the controller can be designed as follows:

$$\zeta = -\gamma_1 \text{sig}^{\frac{b}{c}}(s) - \gamma_2 \text{sig}^{\frac{n}{m}}(s) + u_{nom} \quad (20)$$

where  $\gamma_1, \gamma_2$  is a positive number and  $b < c, n > m$  are all positive odd integers, then there exists a fixed time  $T$  such that the system converges for  $t > T$ .

**Proof.** The proof of Theorem 1 is presented in Appendix A.3.  $\square$

**Remark 1.** According to  $\frac{\partial V}{\partial x} f(x(t), u(t)) \leq \frac{\partial V}{\partial x} f(\hat{x}(t), \zeta(\hat{x}(t)))$ , after obtaining the virtual control law based on Lyapunov, the intersection of the stability domains of the control law obtained from  $P_2(x)$  can be obtained. The final solution obtained at last is also the solution obtained from  $P_3(x)$ . The improved NMPC controller in this paper can inherit the stability and robustness of  $\zeta(x)$  [34].

**Remark 2.** Firstly, we use the control force  $U$  and the desired  $\psi_d$  obtained from the previous step to calculate the expected values  $\phi_d$  and  $\theta_d$  for the angle. Secondly, we establish the desired values for the required roll and pitch angles, where  $\phi_d = \arcsin\left(\frac{U_{a1}S\psi_d - U_{a2}C\psi_d}{\|U_a\|}\right)$  and  $\theta_d = \arctan\left(\frac{U_{a1}C\psi_d + U_{a2}S\psi_d}{U_{a3}}\right)$ . In the case of  $U_a = [U_{a1}, U_{a2}, U_{a3}]^T = \frac{1}{m}U$ , the obtained expected values of the attitude angles serve as inputs to the attitude controller. Therefore, the attitude controller is designed as  $\tau_i = K_p^i e_i + K_I^i \int_0^t e_i dt + K_D^i \frac{de_i}{dt}$ , where  $i \in \{\phi, \theta, \psi\}$ ,  $e_i$  represents the desired value of the angle minus the current value.

### 3.3. Algorithm Structure

In summary, the controller for the position loop increases the system's rapidity without compromising the stability of the system's tracking control. Then, the desired three angle values are obtained by attitude decoupling, and the design of the PID controller for the attitude loop is carried out. The control framework can be seen as follows: Firstly, the desired trajectory  $(X, Y, Z)$  is predefined and the fundamental cost function of the NMPC is formulated based on the established error model. This cost function comprises the error term, control term, and terminal error. A weighted sum of control increments is incorporated into the cost function to enhance control smoothness. Subsequently, the NMPC problem is solved using the C/GMRES method. A stability constraint is introduced to ensure that the stability region of the entire system is confined within the stability domain derived from Lyapunov's method. Finally, the resulting control forces are decoupled to determine the desired angles for the PID control. This iterative process is repeated until the trajectory tracking is completed. Algorithm 2 covers the entire process:

---

#### Algorithm 2: Efficient-NMPC algorithm for UAV trajectory

---

1. **Initialize**  $k_{max} = 0, t_{prev} = 0, \mathbf{x}_0 = \mathbf{x}(0), \mathbf{U}_0 = \mathbf{u}(0), N_{min}, N_{max}, \Delta N$ ;
  2. **While**  $k < k_{max}$  **do**
  3.  $k = k + 1$ ; find  $\tilde{\mathbf{U}}$  by  $\|F(\tilde{\mathbf{U}}, \mathbf{x}(t_0), 0)\| = 0$ ;
  4. **End while**;
  5. **While**  $t < t_{end}$  **do**
  6.  $\mathbf{x}_{k+1} = \mathbf{x}(t + \delta)$  and  $\Delta \mathbf{x}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$ ;
  7.  $\tilde{\mathbf{U}}^* = \text{FDGMRES}(\tilde{\mathbf{U}}, \mathbf{x}, \mathbf{x}/\delta, t, \tilde{\mathbf{U}}, k_{max}, \rho, \varepsilon)$ ;
  8.  $\mathbf{U}^* = \tilde{\mathbf{U}} + \tilde{\mathbf{U}}^* * \delta$ ;
  9. **If**  $\frac{\partial V}{\partial \mathbf{x}} f(\mathbf{x}(t), \mathbf{u}(t)) \leq \frac{\partial V}{\partial \mathbf{x}} f(\hat{\mathbf{x}}(t), \zeta(\hat{\mathbf{x}}(t)))$
  10.  $\mathbf{U} = \mathbf{U}^*$ ; **Else**  $\mathbf{U} = \zeta$ ;
  11. **End If**
  12. Decoupling angel  $\phi_d = \arcsin\left(\frac{U_{a1}S\psi_d - U_{a2}C\psi_d}{\|\mathbf{U}_a\|}\right), \theta_d = \arctan\left(\frac{U_{a1}C\psi_d + U_{a2}S\psi_d}{U_{a3}}\right)$ ;
  13. Attitude controller  $\tau_i = K_P^i e_i + K_I^i \int_0^t e_i dt + K_D^i \frac{de_i}{dt}$ ;
  14. **End While**
- 

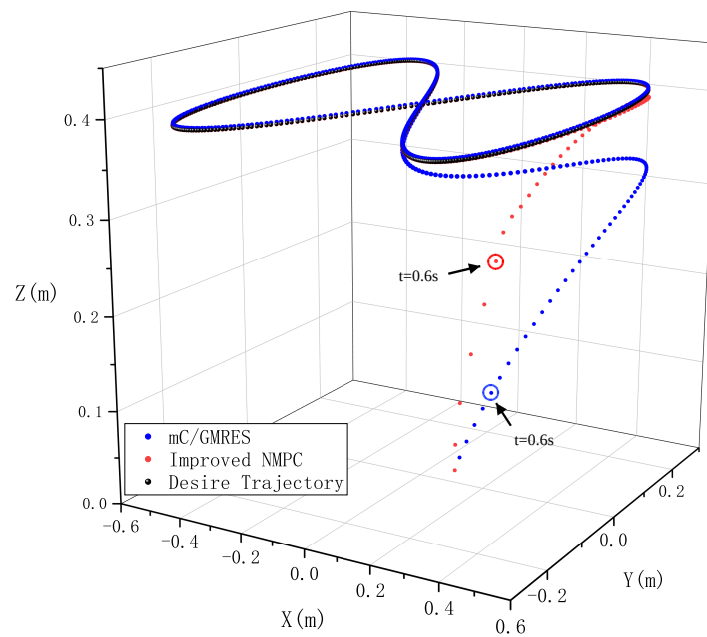
## 4. Simulation

### 4.1. Parameter Selection

In the simulation process, the mass of the quadrotor is 1.12 kg. Inertia moments are  $I_{11} = 0.01 \text{ kg} \cdot \text{m}^2, I_{22} = 0.0082 \text{ kg} \cdot \text{m}^2, I_{33} = 0.0148 \text{ kg} \cdot \text{m}^2$ . The initial position is set as  $\mathbf{X}_0 = [0.1; 0.01; 0.01]$ . The system's sampling period is  $t = 0.1 \text{ s}$ . The desired trajectory is set as  $\mathbf{X}_d = [0.5 \sin(0.3t); 0.3 \sin(0.6t); 0.4]$ . These trajectories are typical tracking control test trajectories with different reference tangential velocities and angular velocities, which pose a challenge to the tracking task. We can choose the following main parameters for the fixed-time integral sliding mode controller  $\gamma_1 = 0.02, \gamma_2 = 0.02, b = 3, c = 2, n = 4, m = 3$ . Since the initial solution of the FDGMRES function at the previous sampling time is often reasonable, the system can converge when the maximum number of iterations  $k_{max}$  is reached. The weighting matrices  $\mathbf{R} = \text{diag}([1, 1, 1]), \mathbf{Q} = \text{diag}([100, 100, 320, 300, 340, 160])$  and  $\mathbf{P} = \text{diag}([110, 130, 100, 300, 80, 40])$ . The coefficient is  $\omega = 0.4$ , and the prediction increment is  $\Delta N = 5$ .

### 4.2. Total Trajectory Simulation

When the sampling time  $t = 0.1 \text{ s}$ , the system tracking results are plotted in Figure 3. The blue trajectory represents the improved NMPC algorithm proposed in this paper and the red trajectory represents the algorithm used in reference [28]. The black trajectory represents the desired trajectory. Observing the trajectories at  $t = 0.6 \text{ s}$ , it can be seen that our proposed improved algorithm exhibits a significant improvement in rapidity, tracking the desired trajectory in a shorter time, with better performance and higher accuracy under the same conditions.



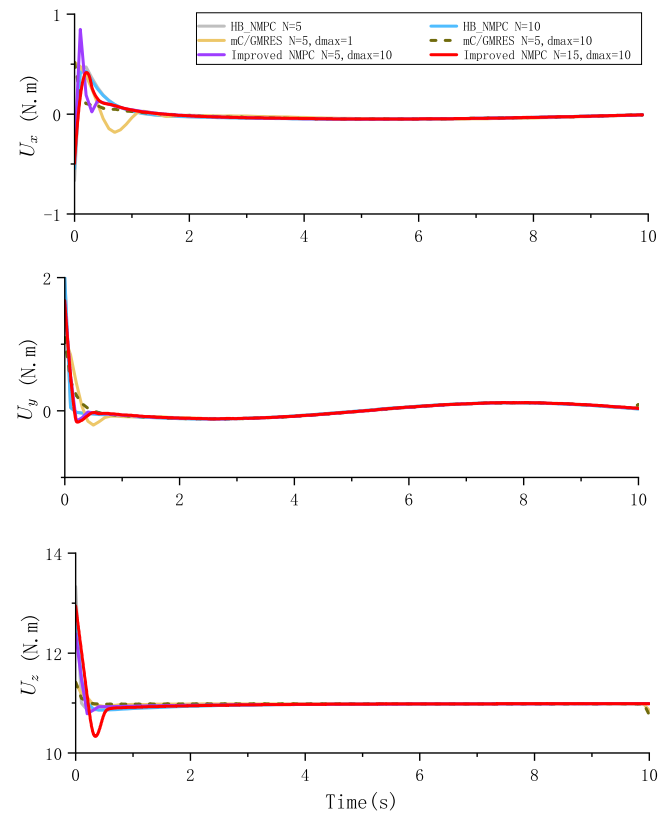
**Figure 3.** Overall trajectory comparison.

#### 4.3. Improved NMPC Tracking Trajectory Performance and Efficiency Comparison

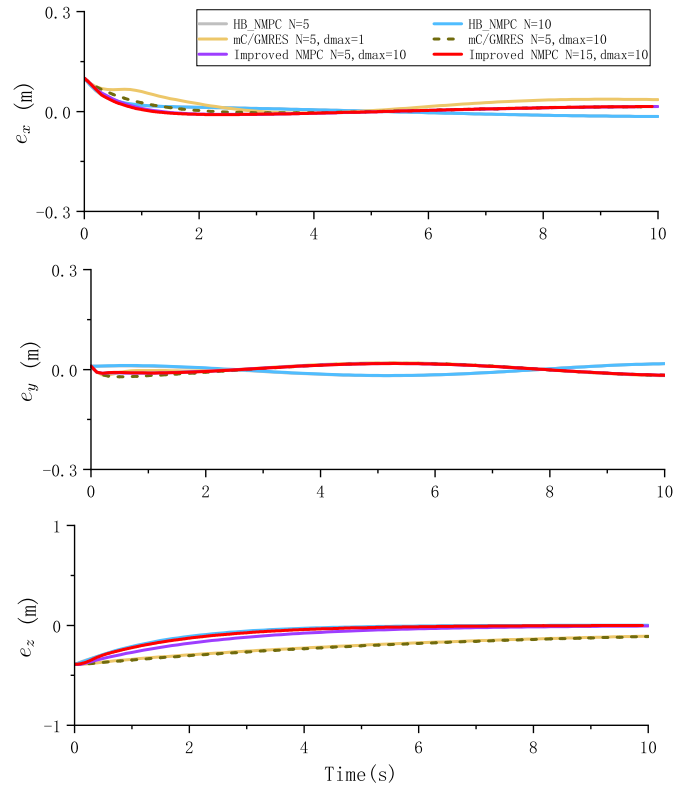
The NMPC algorithm utilizing the Hamiltonian solution method is employed to solve the position trajectory, with a sampling time of  $t = 0.2$  s. The grey trajectory curves for  $N = 5$  and light blue trajectory curves for  $N = 10$  are plotted. For a sampling time of  $t = 0.1$  s and  $N = 5$ , the yellow trajectory curve with  $d_{max} = 1$  and the brown trajectory curve with  $d_{max} = 10$  are solved using the mC/GMRES method [28]. Additionally, with the same sampling time of  $t = 0.1$  s, the improved NMPC solution proposed in this paper is computed for various values of  $N$  and  $d_{max}$ . In Table 1, it can be seen that the average running time of the algorithms increases as  $N$  increases. However, when comparing the two algorithms with the same  $N$ , it can be seen that the average solution speed of the improved NMPC proposed in this paper is higher than that of the Hamiltonian-based NMPC. As shown in Figures 4 and 5, the improved NMPC exhibits faster responsiveness and tracking with faster stabilization. As  $N$  and  $d_{max}$  increase, the  $e_x$  error rapidly converges close to 0. In Figure 4, the C/GMRES method does not achieve a steady-state trend as quickly as the improved NMPC, which can also be observed in  $e_y$ . It is observed that with the same  $d_{max}$ , the system response is faster when  $N = 10$ ; however, when both  $N$  and  $d_{max}$  are lower, the tracking effect is not so good. In Figure 5, larger  $N$  and  $d_{max}$  can make the system error smaller and the system response faster. So when  $N$  and  $d_{max}$  exist relatively correctly, the system performance improves more obviously. The response time is shorter and the system error is smaller. It can be synthesized that the proposed improved NMPC demonstrates better overall performance within the tracking system, but comparisons reveal that different values of  $N$  and  $d_{max}$  still result in slight variations in tracking effectiveness.

**Table 1.** The time required for each iteration of different algorithms.

| Numerical Algorithm    | Prediction Horizon | Average Time (s) |
|------------------------|--------------------|------------------|
| Improved NMPC          | $N = 5$            | 0.0012           |
|                        | $N = 10$           | 0.0041           |
|                        | $N = 15$           | 0.0065           |
| Hamiltonian-based NMPC | $N = 5$            | 0.15             |
|                        | $N = 10$           | 0.19             |
|                        | $N = 15$           | 1.28             |



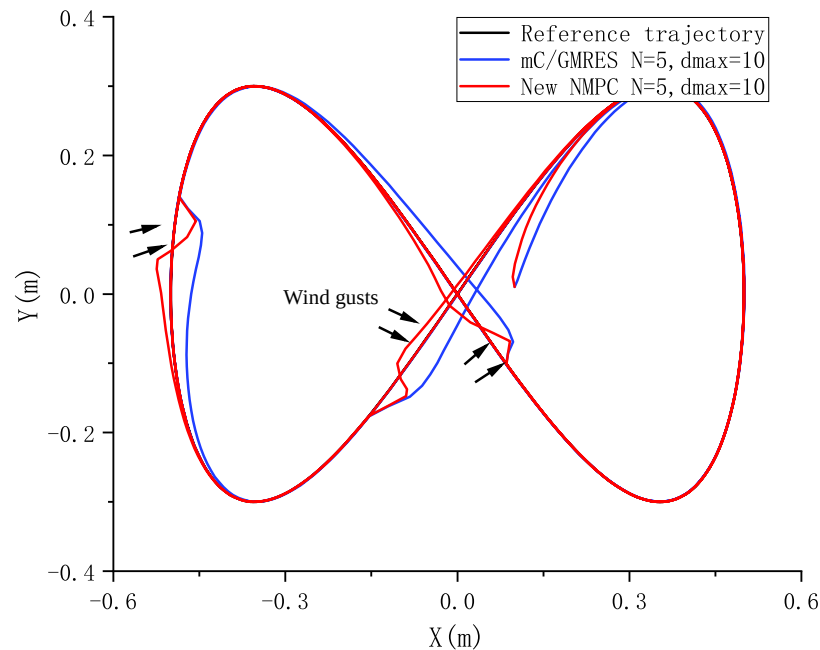
**Figure 4.** Optimal solutions of different algorithms.



**Figure 5.** Tracking errors under different algorithms.

#### 4.4. Performance under Wind Disturbance

When the system faces irregular and sudden wind disturbances during flight, its ability to recover quickly and accurately is essential for stability. To evaluate this recovery capability, wind disturbances of  $ds = [2; -0.1; 1]$  m/s lasting for 0.2 s are applied at 10 s, 15 s, and 20 s, respectively. As shown in Figure 6, the mC/GMRES method demonstrates poor recovery performance in response to wind disturbances, while the improved NMPC proposed in this paper recovers more rapidly.

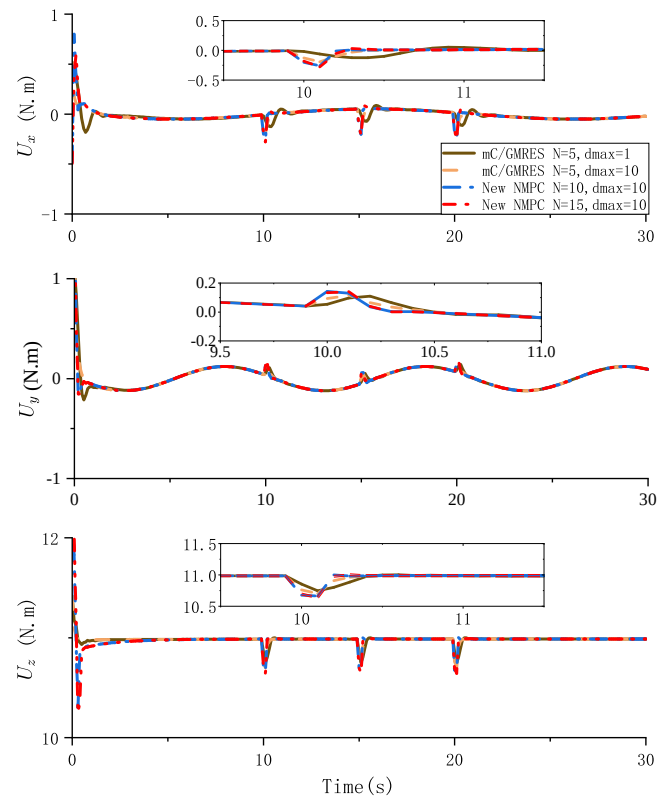


**Figure 6.** Comparison of tracking images under strong disturbances with different algorithms.

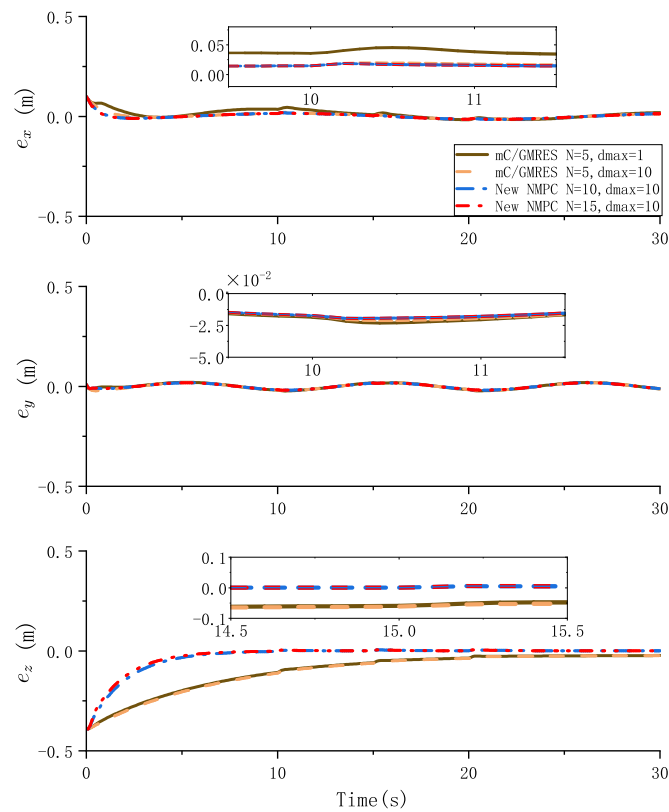
When subjected to smaller wind disturbances of  $ds = [0.2; -0.1; 0.3]$  m/s, lasting 0.2 s at 10 s, 15 s, and 20 s, respectively, the recovery performance and the magnitude of the resulting errors are compared. This comparison includes cases where  $d_{max}$  for the mC/GMRES method is 1 and 10, and where  $d_{max}$  for the improved NMPC method varies. As shown in Figure 7, the improved NMPC proposed in this paper handles wind disturbances more effectively and recovers to stable values more quickly. Figure 8 illustrates that the error in the solution process of the improved NMPC is smoother and converges faster in response to disturbances compared to the mC/GMRES algorithm. In summary, the improved NMPC significantly enhances perturbation resistance and stability compared to the mC/GMRES method. Additionally, Figure 9 shows the variation in linear velocity under different algorithms.

#### 4.5. Attitude Trajectory and Error Simulation

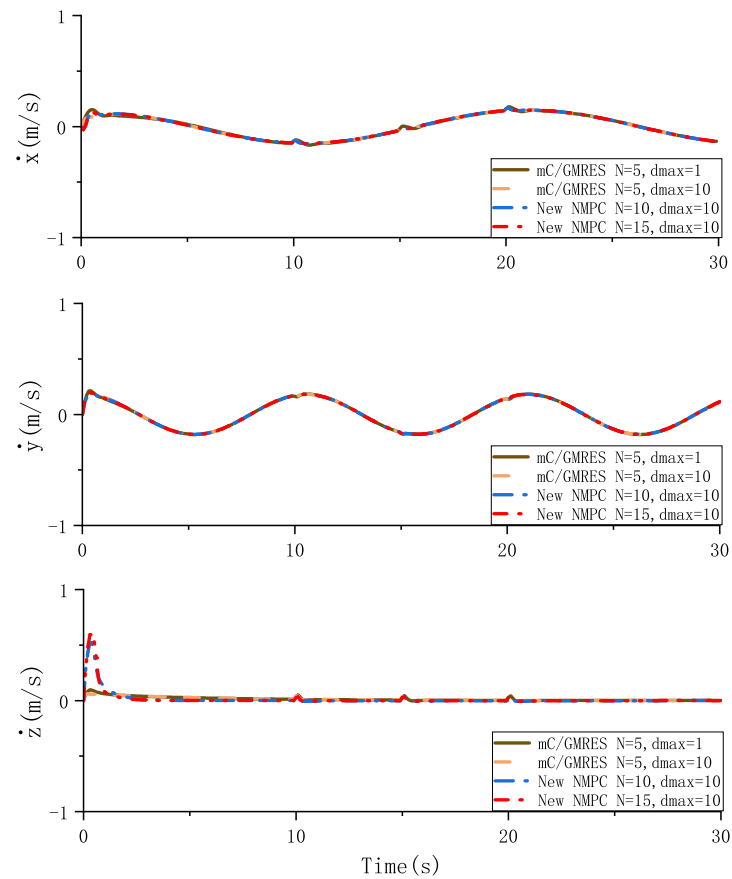
Given the decoupling of the three attitude angles, which are obtained under conditions of rapidity and stability in the position loop, the expected values also adhere to physical characteristics and exhibit strong continuity and stability. Based on this, PID control is employed for the UAV's attitude loop, ensuring more accurate control with enhanced stability. In the absence of external wind disturbances, Figures 10–12 illustrate the images for different angular velocities and error values, respectively. It can be seen that the error  $\phi_e$ ,  $\theta_e$ , and  $\psi_e$  can quickly reach a steady state. The entire trajectory tracking simulation can be viewed at this link (<https://github.com/MHY-design/Dissertation-Video-Verification/blob/main/Real-time%20UAV%20trajectory.gif>, accessed on 5 August 2024). It demonstrates that the UAV can track the entire trajectory in real time, achieve a stable state quickly, and maintain stable control performance.



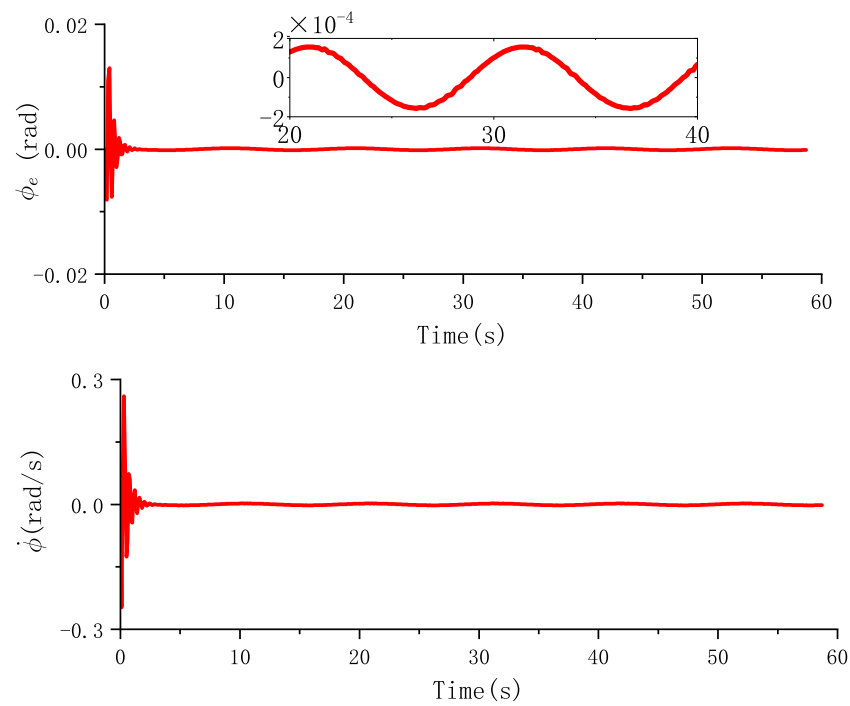
**Figure 7.** Optimal solutions of different algorithms under interference.



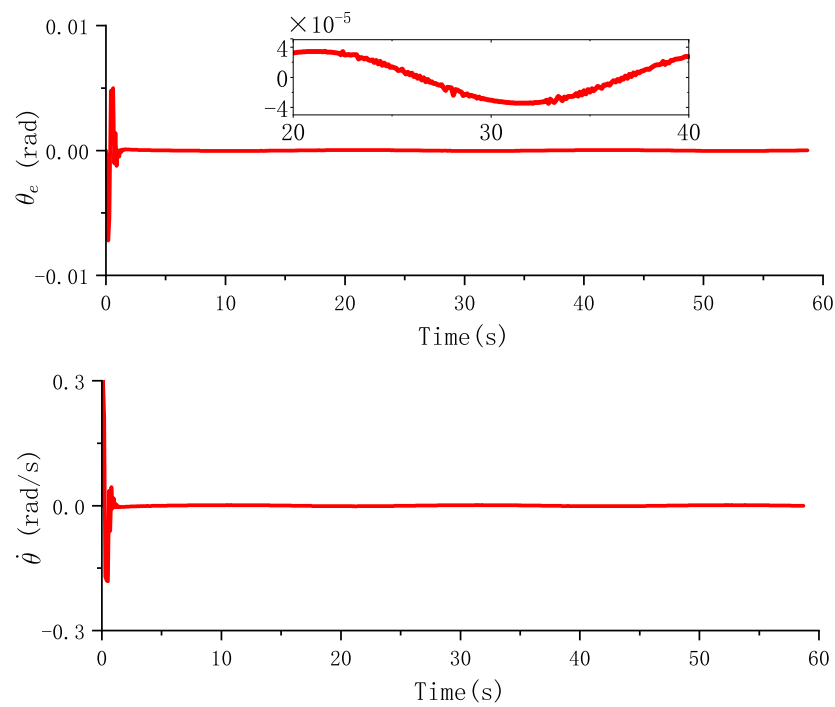
**Figure 8.** Tracking errors under different algorithms in the presence of disturbances.



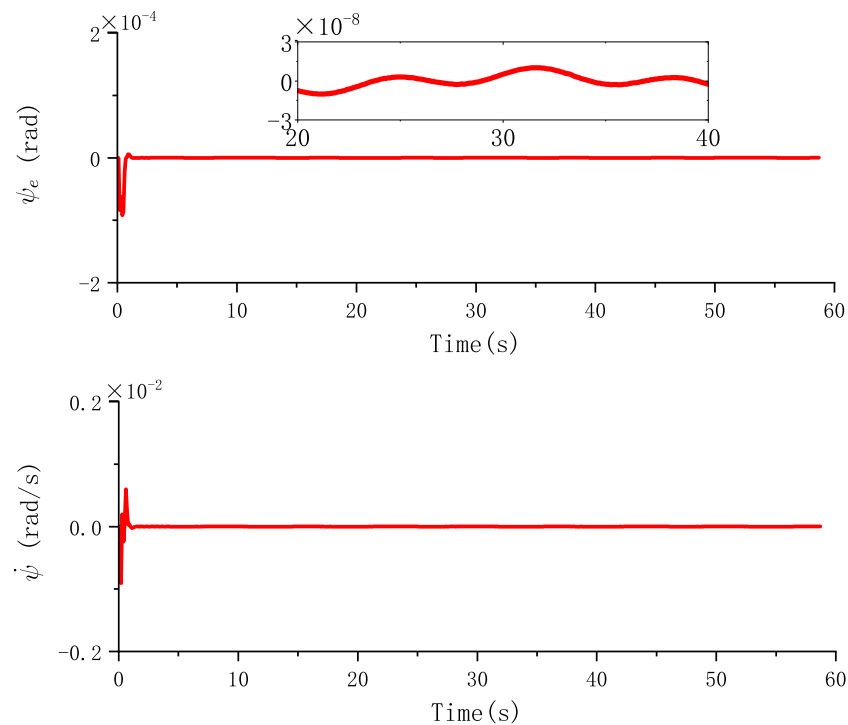
**Figure 9.** Linear velocity under different algorithms in the presence of disturbances.



**Figure 10.** Attitude loop  $\phi_e$  error and  $\dot{\phi}$ .



**Figure 11.** Attitude loop  $\theta_e$  error and  $\dot{\theta}$ .



**Figure 12.** Attitude loop  $\psi_e$  error and  $\dot{\psi}$ .

## 5. Conclusions

An improved NMPC control algorithm is proposed for a quadrotor UAV under practical conditions. This algorithm simultaneously considers inequality constraints, control variation increments, and time constraints. A fixed-time integral sliding mode controller is employed as an auxiliary controller for stability constraints to ensure closed-loop system stability. Simulations using algorithms with varying numerical values to track the reference trajectory indicate that the improved NMPC proposed in this paper is more suitable for ef-

ficiently tracking quadrotors. The simulation experiments clearly show that the parameters significantly impact the system's performance.

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

**Funding:** This work was supported in part by the National Natural Science Foundation of China (Grant Number 62073212), and in part by the Natural Science Foundation of Shanghai (Grant Number 23ZR1426600).

**Data Availability Statement:** The data presented in this study are available on request from the corresponding author.

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

## Appendix A

### Appendix A.1

**Proof of Lemma 1.** The detailed derivation of the KKT conditions for  $P_2(x)$  is provided below [28]:

$$\begin{aligned} \bar{J} = & \sum_{i=0}^{N-1} X(x_i, u_i, p_i) \delta + X_N(x_N, p_N) + \sum_{i=0}^{N-1} \bar{\kappa}_{j+1}^T \times (f_d(x_j, u_j, p_j) - x_{j+1}) \\ & + \sum_{i=0}^{N-1} \varepsilon \cdot \theta(u_i, U_{\max}(i)) \delta + \bar{\kappa}_0^T (x(t_0) - x_0) \end{aligned} \quad (A1)$$

where  $f_d(x_i, u_i, p_i) = x_i + f(x_i, u_i, p_i) \delta$ , and the Hamiltonian sequence formula is defined as  $H_i = X(x_i, u_i, p_i) \delta + \bar{\kappa}_{i+1}^T f_d(x_i, u_i, p_i) + \varepsilon \cdot \theta(u_i, U_{\max}(i)) \delta$ , and then simplifying  $\bar{J}$  to  $\bar{J} = X_N(x_N, p_N) - \bar{\kappa}_N^T x_N + \sum_{i=0}^{N-1} (H_i - \bar{\kappa}_i^T x_i) + H_0 + \bar{\kappa}_0^T (x(t_0) - x_0)$ .

If  $X = \text{col}(x_0, \dots, x_N)$  holds true, then  $\nabla \bar{J} = 0$  is equivalent to  $d\bar{J} = \frac{\partial \bar{J}}{\partial X} dX + \frac{\partial \bar{J}}{\partial U} dU = 0$ . After simplifying  $d\bar{J}$ , we can obtain the following:

$$\begin{aligned} d\bar{J} = & \left( \frac{\partial X_N}{\partial X} - \bar{\kappa}_N^T \right) dX + \sum_{i=0}^{N-1} \left\{ \left( \frac{\partial H_i}{\partial x_i} - \bar{\kappa}_i^T \right) dx_i + \frac{\partial H_i}{\partial u_i} du_i \right\} \\ & + \frac{\partial H_0}{\partial x_0} dx_0 + \frac{\partial H_0}{\partial u_0} du_0 - \bar{\kappa}_0^T dx_0 = 0 \end{aligned} \quad (A2)$$

if the selected costate is

$$\bar{\kappa}_i^T - \frac{\partial H_i}{\partial x_i} = 0 \quad (A3a)$$

$$\frac{\partial X_N}{\partial X} - \bar{\kappa}_N^T = 0 \quad (A3b)$$

given  $x_0 = x(t_0)$ , then  $d\bar{J} = 0$  is equivalent to

$$d\bar{J} = \sum_{i=0}^{N-1} \frac{\partial H_i}{\partial u_i} du_i = 0 \quad (A4)$$

In this context, we can see that Equations (A3a) and (A3b) are equal to Equations (13) and (12), and (14) is sufficient to solve (A4). Currently, the sufficiency of the PMP in addressing the KKT conditions has been proven.

Let  $\bar{\mathbf{U}}_n$  represent the solution to  $P_2(\mathbf{x})$  when  $\varepsilon = \varepsilon_n$ , then  $P_2(\mathbf{x})$  can be expressed as follows:

$$\bar{J}(\bar{\mathbf{U}}_n, \varepsilon_n) = J(\bar{\mathbf{U}}_n) + \varepsilon_n \cdot \vartheta(\bar{\mathbf{U}}_n)\delta \quad (\text{A5})$$

□

### Appendix A.2

**Proof of Lemma 2.** [29] When  $\varepsilon_{n+1} < \varepsilon_n$ , combining with Equation (A5), we have

$$\bar{J}(\bar{\mathbf{U}}_{n+1}, \varepsilon_{n+1}) \leq \bar{J}(\bar{\mathbf{U}}_{n+1}) + \varepsilon_n \cdot \vartheta(\bar{\mathbf{U}}_{n+1})\delta = \bar{J}(\bar{\mathbf{U}}_{n+1}, \varepsilon_n) \quad (\text{A6a})$$

$$\bar{J}(\bar{\mathbf{U}}_{n+1}, \varepsilon_{n+1}) \leq \bar{J}(\bar{\mathbf{U}}_n, \varepsilon_{n+1}) \quad (\text{A6b})$$

similarly, we have

$$\bar{J}(\bar{\mathbf{U}}_{n+1}, \varepsilon_n) \leq \bar{J}(\bar{\mathbf{U}}_n, \varepsilon_n) \quad (\text{A7})$$

from Equations (A6a) and (A7), we can derive

$$\bar{J}(\bar{\mathbf{U}}_{n+1}, \varepsilon_{n+1}) \leq \bar{J}(\bar{\mathbf{U}}_n, \varepsilon_n) \quad (\text{A8})$$

substituting (A5) into (A6b) and (A7), we obtain

$$\bar{J}(\bar{\mathbf{U}}_{n+1} - \bar{\mathbf{U}}_n) \geq \varepsilon_{n+1}(\vartheta(\bar{\mathbf{U}}_n) - \vartheta(\bar{\mathbf{U}}_{n+1})) \quad (\text{A9})$$

$$\bar{J}(\bar{\mathbf{U}}_{n+1} - \bar{\mathbf{U}}_n) \leq \varepsilon_n(\vartheta(\bar{\mathbf{U}}_n) - \vartheta(\bar{\mathbf{U}}_{n+1})) \quad (\text{A10})$$

combining (A9) and (A10), we obtain

$$(\varepsilon_n - \varepsilon_{n+1})(\vartheta(\bar{\mathbf{U}}_n) - \vartheta(\bar{\mathbf{U}}_{n+1})) \geq 0 \quad (\text{A11})$$

therefore, we can obtain

$$\vartheta(\bar{\mathbf{U}}_n) \geq \vartheta(\bar{\mathbf{U}}_{n+1}) \quad (\text{A12})$$

when  $0 < \varepsilon_{n+1} < \varepsilon_n$  holds, substituting (A12) into (A9) yields

$$\bar{J}(\bar{\mathbf{U}}_{n+1}) \geq \bar{J}(\bar{\mathbf{U}}_n) \quad (\text{A13})$$

□

### Appendix A.3

**Proof of Theorem 1.** Constructing the Lyapunov candidate function as  $V = \frac{1}{2}\mathbf{s}^2$ , its derivative can be obtained:

$$\begin{aligned} \dot{V} &= \mathbf{s}\dot{\mathbf{s}} \\ &= \frac{\mathbf{s}}{m}(\zeta - \mathbf{u}_{nom}) \\ &= \frac{\mathbf{s}}{m}(-\gamma_1 \text{sig}^{\frac{b}{c}}(\mathbf{s}) - \gamma_2 \text{sig}^{\frac{n}{m}}(\mathbf{s})) \\ &\leq -\frac{1}{m}(\gamma_1 \mathbf{s}^{\frac{b+c}{c}} + \gamma_2 \mathbf{s}^{\frac{n+m}{m}}) \\ &\leq -\beta_1 2^{\hat{\gamma}_1} V^{\hat{\gamma}_1} - \beta_2 2^{\hat{\gamma}_2} V^{\hat{\gamma}_2} \end{aligned} \quad (\text{A14})$$

where  $\beta_1 = \frac{\gamma_1}{m}$ ,  $\beta_2 = \frac{\gamma_2}{m}$ ,  $\hat{\gamma}_1 = \frac{b+c}{2c}$ ,  $\hat{\gamma}_2 = \frac{n+m}{2m}$ . The system trajectory is fixed-time stable. Moreover, it can converge within time  $T \leq \frac{1}{\beta_1 2^{\hat{\gamma}_1} (1-\hat{\gamma}_1)} + \frac{1}{\beta_2 2^{\hat{\gamma}_2} (\hat{\gamma}_2-1)}$ . This means that the sliding variable  $\mathbf{s}$  remains within a small boundary of zero, which ensures the true sliding mode. As a result, the state variables of the position system will converge to the origin along the sliding surface [41]. □

## References

1. Zuo, Z. Trajectory tracking control design with command-filtered compensation for a quadrotor. *IET Control Theory Appl.* **2010**, *4*, 2343–2355. [\[CrossRef\]](#)
2. Cho, S.W.; Park, H.J.; Lee, H.; Shim, D.H.; Kim, S.Y. Coverage path planning for multiple unmanned aerial vehicles in maritime search and rescue operations. *Comput. Ind. Eng.* **2021**, *161*, 107612. [\[CrossRef\]](#)
3. Shao, X.; Liu, J.; Cao, H.; Shen, C.; Wang, H. Robust dynamic surface trajectory tracking control for a quadrotor UAV via extended state observer. *Int. J. Robust Nonlinear Control* **2018**, *28*, 2700–2719. [\[CrossRef\]](#)
4. Chen, T.; Shan, J. A novel cable-suspended quadrotor transportation system: From theory to experiment. *Aerosp. Sci. Technol.* **2020**, *104*, 105974. [\[CrossRef\]](#)
5. He, Y.; Zhang, S. Trajectory tracking control for quad-rotor system in the presence of velocity constraint. *Int. J. Adv. Robot. Syst.* **2020**, *17*, 1729881420931682. [\[CrossRef\]](#)
6. Gao, p.; Liu, y.; Zhang, H.; Wang, l. Quadrotor helicopter Attitude Control using cascade PID. In Proceedings of the 28th Chinese Control and Decision Conference, CCDC 2016, Yinchuan, China, 28–30 May 2016; pp. 5158–5163.
7. Li, C.; Jing, H.; Bao, J.; Sun, S.; Wang, R. Robust  $H_\infty$  fault tolerant control for quadrotor attitude regulation. *Proc. Inst. Mech. Eng. Part I J. Syst. Control. Eng.* **2018**, *232*, 1302–1313.
8. Rekabi, F.; Shirazi, F.A.; Sadigh, M.J.; Saadat, M. Nonlinear  $H_\infty$  Measurement Feedback Control Algorithm for Quadrotor Position Tracking. *J. Frankl. Inst.* **2020**, *357*, 6777–6804. [\[CrossRef\]](#)
9. Li, B.; Liu, H.; Ahn, C.K.; Wang, C.; Zhu, X. Fixed-Time Tracking Control of Wheel Mobile Robot in Slipping and Skidding Conditions. *IEEE/ASME Trans. Mechatron.* **2024**, 1–11. [\[CrossRef\]](#)
10. Li, B.; Hu, Q.; Yang, Y. Continuous finite-time extended state observer based fault tolerant control for attitude stabilization. *Aerosp. Sci. Technol.* **2019**, *84*, 204–213. [\[CrossRef\]](#)
11. Li, B.; Gong, W.; Yang, Y.; Xiao, B.; Ran, D. Appointed Fixed Time Observer-Based Sliding Mode Control for a Quadrotor UAV Under External Disturbances. *IEEE Trans. Aerosp. Electron. Syst.* **2022**, *58*, 290–303. [\[CrossRef\]](#)
12. Huang, Y.; Zhu, M.; Zheng, Z.; Feroskhan, M. Fixed-time autonomous shipboard landing control of a helicopter with external disturbances. *Aerosp. Sci. Technol.* **2019**, *84*, 18–30. [\[CrossRef\]](#)
13. Li, B.; Liu, H.; Ahn, C.K.; Gong, W. Optimized intelligent tracking control for a quadrotor unmanned aerial vehicle with actuator failures. *Aerosp. Sci. Technol.* **2024**, *144*, 108803. [\[CrossRef\]](#)
14. Hanover, D.; Foehn, P.; Sun, S.; Kaufmann, E.; Scaramuzza, D. Performance, Precision, and Payloads: Adaptive Nonlinear MPC for Quadrotors. *IEEE Rob. Autom. Lett.* **2022**, *7*, 690–697. [\[CrossRef\]](#)
15. Zheng, R.; Lyu, Y. Nonlinear tight formation control of multiple UAVs based on model predictive control. *Def. Technol.* **2023**, *25*, 69–75. [\[CrossRef\]](#)
16. Xu, Z.; Fan, L.; Qiu, W.; Wen, G.; He, Y. A Robust Disturbance-Rejection Controller Using Model Predictive Control for Quadrotor UAV in Tracking Aggressive Trajectory. *Drones* **2023**, *7*, 557. [\[CrossRef\]](#)
17. Yi, F.; Zhang, C.; Rawashdeh, S.; Baek, S. Autonomous Landing of a UAV on a Moving Platform Using Model Predictive Control. *Drones* **2018**, *2*, 34. [\[CrossRef\]](#)
18. Wei, H.; Shen, C.; Shi, Y. Distributed Lyapunov-Based Model Predictive Formation Tracking Control for Autonomous Underwater Vehicles Subject to Disturbances. *IEEE Trans. Syst. Man Cybern. Syst.* **2021**, *51*, 5198–5208. [\[CrossRef\]](#)
19. Mohammadi, A.; Feng, Y.; Zhang, C.; Rawashdeh, S.; Baek, S. Vision-Based Autonomous Landing Using an MPC-Controlled Micro UAV on a Moving Platform. In Proceedings of the 2020 International Conference on Unmanned Aircraft Systems, ICUAS 2020, Athens, Greece, 1–4 September 2020; pp. 771–780.
20. Vlantis, P.; Marantos, P.; Bechlioulis, C.P.; Kyriakopoulos, K.J. Quadrotor landing on an inclined platform of a moving ground vehicle. In Proceedings of the IEEE International Conference on Robotics and Automation, Seattle, WA, USA, 26–30 May 2015; pp. 2202–2207.
21. Outeiro, P.; Cardeira, C.; Oliveira, P. Multiple-model adaptive control architecture for a quadrotor with constant unknown mass and inertia. *Aerosp. Sci. Technol.* **2021**, *117*, 106899. [\[CrossRef\]](#)
22. Yan, D.; Zhang, W.; Chen, H.; Shi, J. Robust control strategy for multi-UAVs system using MPC combined with Kalman-consensus filter and disturbance observer. *Isa Trans.* **2023**, *135*, 35–51. [\[CrossRef\]](#)
23. Zhang, K.; Liu, C.; Shi, Y. Self-triggered adaptive model predictive control of constrained nonlinear systems: A min-max approach. *Automatica* **2022**, *142*, 110424. [\[CrossRef\]](#)
24. Cengiz, S.K.; Uzun, L. Optimal controller design for autonomous quadrotor landing on moving platform. *Simul. Model. Pract. Theory* **2022**, *119*, 102565. [\[CrossRef\]](#)
25. Wu, X.; Xiao, B.; Wu, C.; Guo, Y. Centroidal voronoi tessellation and model predictive control-based macro-micro trajectory optimization of microsatellite swarm. *Space Sci. Technol.* **2022**, *2022*, 9802195. [\[CrossRef\]](#)
26. Boyd, S.; Vandenberghe, L. *Convex Optimization*; Cambridge University Press: Cambridge, UK, 2004.
27. Ohtsuka, T. A tutorial on C/GMRES and automatic code generation for nonlinear model predictive control. In Proceedings of the 2015 European Control Conference, ECC 2015, Linz, Austria, 15–17 July 2015; pp. 73–86.
28. Shen, C.; Buckham, B.; Shi, Y. Modified C/GMRES Algorithm for Fast Nonlinear Model Predictive Tracking Control of AUVs. *IEEE Trans. Control Syst. Technol.* **2017**, *25*, 1896–1904. [\[CrossRef\]](#)

29. Wang, D.; Pan, Q.; Shi, Y.; Hu, J.; Records, C.Z. Efficient Nonlinear Model Predictive Control for Quadrotor Trajectory Tracking: Algorithms and Experiment. *IEEE Trans. Cybern.* **2021**, *51*, 5057–5068. [\[CrossRef\]](#)
30. Singh, B.K.; Kumar, A. Model predictive control using LPV approach for trajectory tracking of quadrotor UAV with external disturbances. *Aircraft Eng. Aerosp. Technol.* **2023**, *95*, 607–618. [\[CrossRef\]](#)
31. Giorgi, G.; Jimenez, B.; Novo, V. Approximate Karush-Kuhn-Tucker Condition in Multiobjective Optimization. *J. Optim. Theory Appl.* **2016**, *171*, 70–89. [\[CrossRef\]](#)
32. Lasdon, L.; Waren, A.; Rice, R. An interior penalty method for inequality constrained optimal control problems. *IEEE Trans. Autom.* **1967**, *12*, 388–395. [\[CrossRef\]](#)
33. Khalil, K.H. *Nonlinear Systems*; Prentice-Hall: Hoboken, NJ, USA, 2002.
34. Christofides, P.D.; Jinfeng, L.; de la Peña, D.M. *Networked and Distributed Predictive Control: Methods and Nonlinear Process Network Applications*; Advances in Industrial Control; Springer: London, UK, 2011. [\[CrossRef\]](#)
35. Munoz de la Pena, D.; Christofides, P.D. Lyapunov-Based Model Predictive Control of Nonlinear Systems Subject to Data Losses. *IEEE Trans. Automat. Control* **2008**, *53*, 2076–2089. [\[CrossRef\]](#)
36. Shen, C.; Shi, Y. Distributed implementation of nonlinear model predictive control for AUV trajectory tracking. *Automatica* **2020**, *115*, 108863. [\[CrossRef\]](#)
37. Shen, C.; Shi, Y.; Buckham, B. Lyapunov-based model predictive control for dynamic positioning of autonomous underwater vehicles. In Proceedings of the 2017 IEEE International Conference on Unmanned Systems, ICUS 2017, Beijing, China, 27–29 October 2017; pp. 588–593.
38. Ba, D.; Li, Y.X.; Tong, S. Fixed-time adaptive neural tracking control for a class of uncertain nonstrict nonlinear systems. *Neurocomputing* **2019**, *363*, 273–280. [\[CrossRef\]](#)
39. Polyakov, A. Nonlinear Feedback Design for Fixed-Time Stabilization of Linear Control Systems. *IEEE Trans. Autom.* **2012**, *57*, 2106–2110. [\[CrossRef\]](#)
40. Li, B.; Zhang, H.; Xiao, B.; Wang, C.; Yang, Y. Fixed-time integral sliding mode control of a high-order nonlinear system. *Nonlinear Dyn.* **2022**, *107*, 909–920. [\[CrossRef\]](#)
41. Song, T.; Fang, L.; Wang, H. Model-free finite-time terminal sliding mode control with a novel adaptive sliding mode observer of uncertain robot systems. *Asian J. Control* **2022**, *24*, 1437–1451. [\[CrossRef\]](#)

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