mathematics

MDPI

*Article*

# Design and Real-Time Implementation of a Cascaded Model Predictive Control Architecture for Unmanned Aerial Vehicles

**Patricio Borbolla-Burillo [1], David Sotelo [1,*], Michael Frye [2], Luis E. Garza-Castañón [1], Luis Juárez-Moreno [1] and Carlos Sotelo [1]**

[1] Tecnologico de Monterrey, School of Engineering and Sciences, Ave. Eugenio Garza Sada 2501, Monterrey 64849, Mexico; a01366763@tec.mx (P.B.-B.); legarza@tec.mx (L.E.G.-C.); a01283720@tec.mx (L.J.-M.); carlos.sotelo@tec.mx (C.S.)

[2] Department of Engineering, University of the Incarnate Word, San Antonio, TX 78209, USA; mfrye@uiwtx.edu

* Correspondence: david.sotelo@tec.mx

**Abstract:** Modeling and control are challenging in unmanned aerial vehicles, especially in quadrotors where there exists high coupling between the position and the orientation dynamics. In simulations, conventional control strategies such as the use of a proportional–integral–derivative (PID) controller under different configurations are typically employed due to their simplicity and ease of design. However, linear assumptions have to be made, which turns into poor performance for practical applications on unmanned aerial vehicles (UAVs). This paper designs and implements a hierarchical cascaded model predictive control (MPC) for three-dimensional trajectory tracking using a quadrotor platform. The overall system consists of two stages: the mission server and the commander stabilizer. Different from existing works, the heavy computational burden is managed by decomposing the overall MPC strategy into two different schemes. The first scheme controls the translational displacements while the second scheme regulates the rotational movements of the quadrotor. For validation, the performance of the proposed controller is compared against that of a proportional–integral–velocity (PIV) controller taken from the literature. Here, real-world experiments for tracking helicoidal and lemniscate trajectories are implemented, while for regulation, an extreme wind disturbance is applied. The experimental results show that the proposed controller outperforms the PIV controller, presenting less signal effort fluctuations, especially in terms of rejecting external wind disturbances.

**Keywords:** quadrotor UAV; cascade hierarchical MPC; control structure design; real-time implementation; external disturbance

**MSC:** 93C05



**Citation:** Borbolla-Burillo, P.; Sotelo, D.; Frye, M.; Garza-Castañón, L.E.; Juárez-Moreno, L.; Sotelo, C. Design and Real-Time Implementation of a Cascaded Model Predictive Control Architecture for Unmanned Aerial Vehicles. *Mathematics* **2024**, *12*, 739. https://doi.org/10.3390/math12050739

Academic Editor: Zhijia Zhao

Received: 17 December 2023
Revised: 1 February 2024
Accepted: 12 February 2024
Published: 29 February 2024



## 1. Introduction

In recent years, the use of UAV technology has notably increased. Quadrotors are preferred over other UAVs because of their versatility and their stable dynamics [1]. For this reason, they are used in aerial surveillance, shipping, emergency situations, entertainment shows, and agriculture monitoring, among other areas. These applications require the quadrotors to be fast, agile, and lightweight in order to be capable of following precise trajectories in the presence of external disturbances.

To meet these requirements, some control strategies have been developed [2–7]. First, the well-known PID controller, which can handle both linear and nonlinear systems, can achieve zero steady-state errors with a simple-to-understand tuning principle [2]. A linear–quadratic regulator (LQR) control can be easily implemented and can handle MIMO systems [3]. Feedback linearization (FL) provides the capacity to linearize a model with a model inversion procedure instead of making linearization assumptions [4]. Even though these control strategies can achieve adequate reference response, they are not robust and some of them even lack the ability to achieve zero steady-state errors [3,4].

MPC is one control strategy that is useful in handling multivariable processes. This method consists of an optimal control technique that calculates the necessary inputs of a model to minimize its cost function, which represents its system dynamics using its constraints over a limited time horizon [8]. Therefore, MPC is a robust control technique capable of controlling complex nonlinear systems while meeting the system constraints and ensuring safe operation [9]. However, the application of large-scale MPC in real-time has its own challenges, mainly due to its reliability, large computational overhead, and the complexity of its software implementation [10]. A literature survey presented recent real-time applications in which the MPC strategy was used to address these concerns [10–18]. For these reasons, the control of quadrotors using MPC has become feasible in more sophisticated applications as research of their control algorithms advances.

A variety of linear and nonlinear MPCs have been developed to improve control performance in quadrotors [19–26]. In [19], a compound control technique that consists of two different control strategies was described. They used both an integral sliding mode control to reject matched disturbances and an MPC to treat the remaining disturbances for successful trajectory tracking of a quadrotor. However, one of the main problems in MPC is the large computational capacity required. Therefore, in [20], the input sequence of an MPC was approximated with the Laguerre function, decreasing the computational burden and improving the inner loop performance of the quadrotor. Moreover, in [21], a simplified optimal control problem was proposed to reduce the computational burden of traditional MPCs so their proposed 'bang-bang' MPC could be applied to smaller and lighter quadrotors with less computational capacity. In [22], the authors employed a nonlinear MPC (NMPC) for the inner control loop of a quadrotor while maintaining the computation cost that is similar to a traditional NMPC.

On the other hand, system failures are always to be expected. For that reason, in [23], the authors developed an NMPC capable of stabilizing a drone during a rotor failure. Quadrotors should be able to adapt themselves to whatever model uncertainties they are subjected too. Therefore, in [24], the authors used a hybrid adaptive NMPC to learn the model uncertainties in real time. This control algorithm improved the quadrotor's control robustness and reduced tracking error by 90% compared to a non-adaptive NMPC while keeping the computational overhead burden at a minimum. In [25], an NMPC scheme was proposed for obstacle avoidance in quadrotors; the authors applied a potential field function-based penalty term and a dynamical adjustment for the prediction horizon to cut down the heavy computational load. Similarly, in [26], an NMPC for obstacle avoidance was employed; in this case, the NMPC control was formulated in the special Euclidean group SE(3) for the quadrotor to operate in messy environments with a large number of obstacles. Simulations were performed using algebraic ellipsoids as obstacles.

Although tremendous effort has been dedicated to implementing MPC in unmanned systems [27], there still exist significant challenges in high-speed real-time applications. An MPC is designed based on the entire nonlinear dynamics model of a quadrotor UAV, which introduces several computational challenges [18,28], and its real-time applicability is reduced. To avoid this, linear control strategies have been implemented while considering the advantages of being relatively easy to understand and design [29]. The linearization procedure is carried out by first trimming the quadrotor at zero wind and then at hover [30]. Thus, it is sufficient to use a single linear time invariant (LTI) model for MPC [27,31]. In this paper, to address the drawbacks of the advanced control strategies in quadrotors mentioned in [14,32], the real-time implementation is addressed. Here, the computationally efficient control architecture for a multivariable system is obtained, which reduces the complexity of the system dynamics and improves its real-time performance [13,33]. The overall MPC strategy is decomposed into two different schemes [16,34]. The first scheme controls the translational displacements while the second scheme regulates the rotational movements of the quadrotor. This turns into a less computationally expensive scheme, which is suitable for real-time implementation. To validate the tracking performance of the proposed MPC architecture, it is compared to a PIV controller with three real-time scenarios: tracking helicoidal, tracking lemniscate, and

aggressive regulator under extreme external disturbances. All tests were carried out with a QDrone 2 quadrotor from Quanser® [28,35].

This paper is organized as follows. Section 2 states the dynamic model of the QDrone 2 quadrotor UAV. Section 3 gives an overview of the MPC control technique. Section 4 details the overall system used for implementation, including the compared PIV control structure and the proposed MPC control scheme and its algorithm. Section 5 contains all the tuning and hardware parameters and presents the tracking and regulator scenarios. The implementation results illustrate the performance improvement of the proposed cascaded MPC control structure compared to a PIV controller. Finally, in Section 6, the paper summarizes the results and future work.

## 2. Kinematics

First, to model the quadrotor dynamics with the Newton–Euler formalism taken from the literature [36], it is necessary to consider two frames, the inertial frame ($o_w x_w y_w z_w$) and the quadrotor body frame ($o_b x_b y_b z_b$). These two reference frames are needed to define both the position and orientation of the quadrotor UAV (Figure 1). Considering that each propeller contributes individually to the movement of the whole system, the force generated by each propeller and the thrust produced by the quadrotor is defined by

$$f_i = k\omega_i^2 \tag{1}$$

$$T = \begin{bmatrix} 0 \\ 0 \\ \sum_{i=1}^{4} f_i \end{bmatrix} \tag{2}$$

where $k$ is the lift constant and $\omega$ is the angular velocity of the rotor. Therefore, as the four rotors are fixed pointing upwards, the total thrust generated will be on the $z_b$ axis. The total thrust is equivalent to the sum of the forces generated by the rotors.

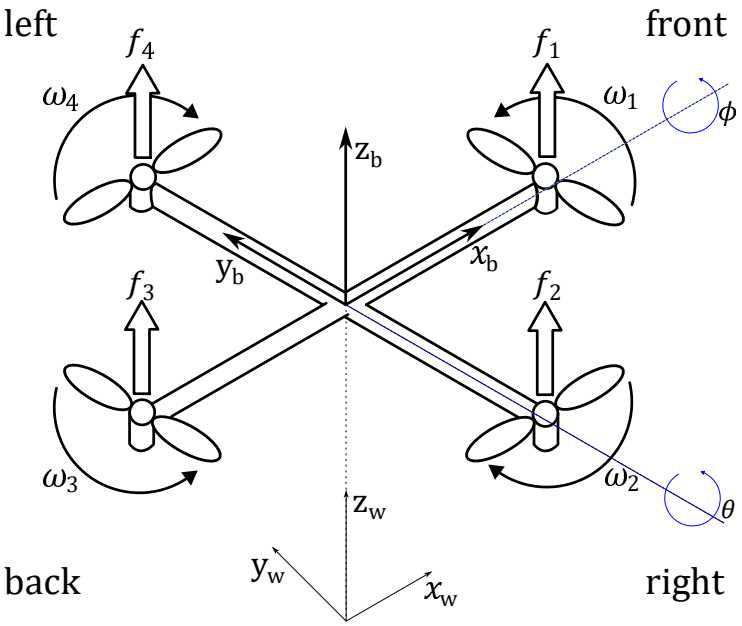



**Figure 1.** Schematic depiction of the quadrotor coordinate frame configuration.

In the inertial frame, the absolute position is defined by a linear position vector $\xi$ with the coordinates $x_w, y_w, z_w$, and an angular position vector $\eta$ with the roll, pitch, and yaw angles, expressed as $\phi, \theta, \psi$, respectively.

$$\xi = \begin{bmatrix} x_w & y_w & z_w \end{bmatrix}^T \tag{3}$$

$$\eta = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^T \tag{4}$$

On the other hand, the body frame is defined by a linear position vector $\sigma$ with $x_b, y_b, z_b$ coordinates, and an angular velocity vector $\gamma$ with $p, q, r$.

$$\sigma = \begin{bmatrix} x_b & y_b & z_b \end{bmatrix}^T \tag{5}$$

$$\gamma = \begin{bmatrix} p & q & r \end{bmatrix}^T \tag{6}$$

Both the inertial fixed frame and the body fixed frame linear-movement vectors are related by the transformation matrix $R_L$ taken from the $ZYX$ Tait–Bryan angles [37], where $C$ and $S$ denote cosine and sine functions, respectively.

$$R_L = \begin{bmatrix} C\psi \cdot C\theta & C\psi \cdot S\theta \cdot S\phi - S\psi \cdot C\phi & C\psi \cdot S\theta \cdot C\phi + S\psi \cdot S\phi \\ S\psi \cdot C\theta & S\psi \cdot S\theta \cdot S\phi + C\psi \cdot C\phi & S\psi \cdot S\theta \cdot C\phi - C\psi \cdot S\phi \\ -S\theta & C\theta \cdot S\phi & C\theta \cdot C\phi \end{bmatrix} \tag{7}$$

$$\xi = R_L \cdot \sigma \tag{8}$$

Similar to $R_L$, there is also a rotational movement transformation matrix $R_R$ that relates the angular velocity vectors from the inertial and body fixed frames [37].

$$R_R = \begin{bmatrix} 1 & 0 & -S\theta \\ 0 & C\phi & C\theta \cdot S\phi \\ 0 & -S\phi & C\theta \cdot C\phi \end{bmatrix} \tag{9}$$

$$\gamma = R_R \cdot \dot{\eta} \tag{10}$$

The translational dynamical model of the quadrotor in the inertial fixed frame based on Newton's second law of translational motion is

$$m \cdot \ddot{\xi} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} - k_d^L \dot{\xi} + R_L T \tag{11}$$

where $T$ is the thrust vector, $m$ is the mass of the quadrotor, $k_d^L$ is the thrust drag coefficient, and $g$ is the Earth's gravity constant. On the other hand, the rotational dynamical model in the body fixed frame using Euler's equation of rotational motion is the following:

$$I\dot{\gamma} = \tau - \gamma \times (I\gamma) - \mu - k_d^R \gamma \tag{12}$$

$$\tau = \begin{bmatrix} l_r(f_2 - f_4) \\ l_p(f_1 - f_3) \\ d(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{bmatrix} \tag{13}$$

$$\omega_\Gamma = \omega_1 - \omega_2 + \omega_3 - \omega_4 \tag{14}$$

$$\mu = I_R \gamma \times \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \omega_\Gamma \tag{15}$$

$$I = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{16}$$

where $\mu$ is the gyroscopic moment vector [38]; $k_d^R$ is the moment drag coefficient; $I_R$ is the gyroscopic inertia vector induced by the propellers; $d$ is the drag coefficient; $I$ is the diagonal matrix of inertia; $\omega_\Gamma$ is the residual angular speed of the rotors; and $l_r$ and $l_p$ are the lengths between the quadrotor center of mass and the rotor that corresponds to the roll and pitch angle, respectively. Then, the complete nonlinear dynamical model of the quadrotor is based on the following equations:

$$\ddot{x}_w = \frac{U_1}{m}(C\psi \cdot S\theta \cdot C\phi + S\psi \cdot S\phi) - \frac{k_d^L \dot{x}_w}{m} \tag{17}$$

$$\ddot{y}_w = \frac{U_1}{m}(S\psi \cdot S\theta \cdot C\phi - C\psi \cdot S\phi) - \frac{k_d^L \dot{y}_w}{m} \tag{18}$$

$$\ddot{z}_w = \frac{U_1}{m}(C\theta \cdot C\phi) - g - \frac{k_d^L \dot{z}_w}{m} \tag{19}$$

$$\dot{p} = \frac{U_2}{I_{xx}} + \frac{I_{yy} - I_{zz}}{I_{xx}} qr - \frac{I_R \omega_\Gamma}{I_{xx}} q - \frac{k_d^R}{I_{xx}} p \tag{20}$$

$$\dot{q} = \frac{U_3}{I_{yy}} + \frac{I_{zz} - I_{xx}}{I_{yy}} pr + \frac{I_R \omega_\Gamma}{I_{yy}} p - \frac{k_d^R}{I_{yy}} q \tag{21}$$

$$\dot{r} = \frac{U_4}{I_{zz}} + \frac{I_{xx} - I_{yy}}{I_{zz}} pq - \frac{k_d^R}{I_{zz}} r \tag{22}$$

Here, the quadrotor system consists of four input variables described in vector $U$

$$U = \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} k & k & k & k \\ 0 & l_r k & 0 & -l_r k \\ l_p k & 0 & -l_p k & 0 \\ d & -d & d & -d \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} \tag{23}$$

where $U_1$ denotes the total thrust, $U_2$ and $U_3$ stand for the torques generated about $\phi$ and $\theta$ axes, respectively, and $U_4$ is the moment produced about the $\psi$ axis. Thus, the choice of state vector is given as follows

$$X = \begin{bmatrix} x_w & y_w & z_w & \dot{x}_w & \dot{y}_w & \dot{z}_w & \phi & \theta & \psi & p & q & r \end{bmatrix}^T \tag{24}$$

Moreover, considering linear control techniques are widely used for UAV flight control [28,30,31,34,39], as they can ensure optimal performance when operating close to the equilibrium point [40,41], a linearization of the model under the following assumptions is carried out:

- Standard quadrotor structure.
- The structure is perfectly rigid.
- The center of mass of the quadrotor coincides with the origin of the body frame.
- Aerodynamics and other complex phenomena that are difficult to model are ignored.
- The quadrotor is hovering.

Then, considering that the quadrotor is assumed to be in hover position [30], the following is true:

$$\phi = \theta = \psi \approx 0, \quad \dot{\phi} = \dot{\theta} = \dot{\psi} \approx 0 \tag{25}$$

$$I_R \approx 0, \quad k_d^L \approx 0, \quad k_d^R \approx 0 \tag{26}$$

$$(\dot{\phi}, \dot{\theta}, \dot{\psi}) \approx (p, q, r) \tag{27}$$

$$U = \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} mg \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{28}$$

Given that $\ddot{x}_w$ depends directly on $\theta$ and $\ddot{y}_w$ depends directly on $\phi$, the input $U_1$, Equation (28), is substituted in (17) and (18) [42]. Then, by performing small-angle approximations and eliminating the products of two terms near zero, the linearized dynamical model used for the proposed MPC is obtained:

$$\ddot{x}_w = g\theta \tag{29}$$

$$\ddot{y}_w = -g\phi \tag{30}$$

$$\ddot{z}_w = \frac{U_1}{m} - g \tag{31}$$

$$\ddot{\phi} = \frac{U_2}{I_{xx}} \tag{32}$$

$$\ddot{\theta} = \frac{U_3}{I_{yy}} \tag{33}$$

$$\ddot{\psi} = \frac{U_4}{I_{zz}} \tag{34}$$

The linearized dynamics in (29)–(34) are decoupled for state space representation. Each of them follows the state space structure **Ax + Bu**.The altitude controller (31) is rewritten as

$$\begin{bmatrix} \dot{z}_w \\ \ddot{z}_w \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} z_w \\ \dot{z}_w \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} T + \mathbf{g} \tag{35}$$

where $m$ corresponds to the mass of the drone and

$$\mathbf{g} = \begin{bmatrix} 0 \\ -g \end{bmatrix} \tag{36}$$

For the $x_w y_w$ position controller, (29) and (30) can be represented as

$$\begin{bmatrix} \dot{x}_w \\ \ddot{x}_w \\ \dot{y}_w \\ \ddot{y}_w \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_w \\ \dot{x}_w \\ y_w \\ \dot{y}_w \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ g & 0 \\ 0 & 0 \\ 0 & -g \end{bmatrix} \begin{bmatrix} \theta^d \\ \phi^d \end{bmatrix} \tag{37}$$

And the state space representation of the rotational dynamics, (32)–(34) can be rewritten as

$$\begin{bmatrix} \dot{\phi} \\ \ddot{\phi} \\ \dot{\theta} \\ \ddot{\theta} \\ \dot{\psi} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \phi \\ \dot{\phi} \\ \theta \\ \dot{\theta} \\ \psi \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ \frac{1}{I_{xx}} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & \frac{1}{I_{yy}} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \frac{1}{I_{zz}} \end{bmatrix} \tau \tag{38}$$

The controllability of the linearized system is tested before the design of the controller. Here, the controllability matrix has full rank, which implies that the linear system is controllable. On the other hand, even though the system is unstable, linear controllers ensure system stability and optimal performance when operating close to the equilibrium point [40,41].

## 3. Control Strategy

In this section, MPCs are designed for translational and rotational dynamics. The translational subsystem considers two MPCs, one for altitude and the other for $x_w y_w$ position; meanwhile, the rotational subsystem has an MPC for attitude. Hence, the control actions are obtained in each subsystem separately.

The MPC strategies for $x_w y_w$ position and attitude are based on [43]. However, considering that $z_w$ axis depends on gravity constant $g$, the following LTI state space model is used to represent the altitude system dynamics:

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k) + \mathbf{g} \tag{39}$$

$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) \tag{40}$$

where the state vector is denoted as $\mathbf{x} \in \mathcal{R}^n$, the input vector is $\mathbf{u} \in \mathcal{R}^{n_u}$, the output vector is $\mathbf{y} \in \mathcal{R}^{n_y}$, the state matrix is $\mathbf{A} \in \mathcal{R}^{n \times n}$, the input matrix is $\mathbf{B} \in \mathcal{R}^{n \times n_u}$, the output matrix is $\mathbf{C} \in \mathcal{R}^{n_y \times n}$, and the sampling time is expressed as $k \in N$. Furthermore, the projection matrix $\mathbf{\Pi}_i^{(n,N)}$ is defined as follows:

$$\mathbf{\Pi}_i^{(n,N)} \equiv \left[ \underbrace{\mathbf{0}_{n\times n} \ldots \mathbf{0}_{n\times n}}_{(i-1)terms} \quad \mathbf{I}_{n\times n} \quad \underbrace{\mathbf{0}_{n\times n} \ldots \mathbf{0}_{n\times n}}_{(N-i)terms} \right] \tag{41}$$

where $n$ stands for the vector length, $N$ refers to the prediction horizon, and $i$ is the term desired.

Then, the state predictions for the sampling time are

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k) + \mathbf{g},$$

$$\mathbf{x}(k+2) = \mathbf{A}^2\mathbf{x}(k) + \mathbf{A}\mathbf{B}\mathbf{u}(k) + \mathbf{B}\mathbf{u}(k+1) + \mathbf{A}\mathbf{g} + \mathbf{g},$$

$$\vdots$$

$$\mathbf{x}(k+i) = \left[\mathbf{A}^i\right]\mathbf{x}(k) + \begin{bmatrix} \mathbf{A}^{i-1}\mathbf{B} & \ldots & \mathbf{A}\mathbf{B} & \mathbf{B} \end{bmatrix} \begin{bmatrix} \mathbf{u}(k) \\ \vdots \\ \mathbf{u}(k+i-2) \\ \mathbf{u}(k+i-1) \end{bmatrix} + \begin{bmatrix} \mathbf{A}^{i-1}\mathbf{B} & \ldots & \mathbf{A} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{g} \\ \vdots \\ \mathbf{g} \\ \mathbf{g} \end{bmatrix},$$

$$\mathbf{x}(k+i) = \left[\mathbf{A}^{i}\right]\mathbf{x}(k) + \begin{bmatrix}\mathbf{A}^{i-1}\mathbf{B} & \dots & \mathbf{AB} & \mathbf{B}\end{bmatrix}\begin{bmatrix}\mathbf{\Pi}_1^{(n_u,N)} \\ \vdots \\ \mathbf{\Pi}_{i-1}^{(n_u,N)} \\ \mathbf{\Pi}_i^{(n_u,N)}\end{bmatrix}\widetilde{\mathbf{u}}(k) + \begin{bmatrix}\mathbf{A^{i-1}} & \dots & \mathbf{A} & \mathbf{I}\end{bmatrix}\widetilde{\mathbf{g}},$$

$$\mathbf{x}(k+i) = \mathbf{\Phi}_i\mathbf{x}(k) + \mathbf{\Psi}_i\widetilde{\mathbf{u}}(k) + \mathbf{\Theta}_i\widetilde{\mathbf{g}}, \quad \forall i \in \{1,\dots,N\} \tag{42}$$

where $\mathbf{\Phi}_i \in \mathcal{R}^{n\times n}$, $\mathbf{\Psi}_i \in \mathcal{R}^{n\times(Nnu)}$, and $\mathbf{\Theta}_i \in \mathcal{R}^{n\ \times\ (Nn)}$ stand for the N-step-ahead in LTI systems in a more compact manner [44]. Hence, the above equation can be reformulated in the following form:

$$\widetilde{\mathbf{x}}(k) = \mathbf{\Phi}\mathbf{x}(k) + \mathbf{\Psi}\widetilde{\mathbf{u}}(k) + \mathbf{\Theta}\widetilde{\mathbf{g}} \tag{43}$$

where $\mathbf{\Phi} = \begin{bmatrix}\mathbf{\Phi}_1 & \mathbf{\Phi}_2 & \dots & \mathbf{\Phi}_N\end{bmatrix}^T$, $\mathbf{\Psi} = \begin{bmatrix}\mathbf{\Psi}_1 & \mathbf{\Psi}_2 & \dots & \mathbf{\Psi}_N\end{bmatrix}^T$, $\mathbf{\Theta} = \begin{bmatrix}\mathbf{\Theta_1} & \mathbf{\Theta_2} & \dots & \mathbf{\Theta}_N\end{bmatrix}^{\mathbf{T}}$, $\widetilde{\mathbf{x}}(k) \in \mathcal{R}^{N\cdot nu}$ stands for the whole state trajectory of $\mathbf{x} = \begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_n\end{bmatrix}^T$; $\widetilde{\mathbf{u}}(k) \in \mathcal{R}^{N\cdot n_u}$ contains the computed sequence of $\mathbf{u} = \begin{bmatrix}\mathbf{u}_1 & \mathbf{u}_2 & \dots & \mathbf{u}_{n_u}\end{bmatrix}^T$; $\widetilde{\mathbf{g}} \in R^{(N\cdot n)}$ contains the gravity vectors $\mathbf{g} = \begin{bmatrix}0 & -g\end{bmatrix}^T$; and $\widetilde{\mathbf{y}}(k) \in \mathcal{R}^{N\cdot n_y}$ is the output trajectory of $\mathbf{y} = \begin{bmatrix}\mathbf{y}_1 & \mathbf{y}_2 & \dots & \mathbf{y}_{n_y}\end{bmatrix}^T$. Then, depending on the prediction horizon $N$, $\widetilde{\mathbf{x}}(k)$, $\widetilde{\mathbf{u}}(k)$, and $\widetilde{\mathbf{y}}(k)$ are expressed as

$$\widetilde{\mathbf{x}}(k) \equiv \begin{bmatrix}\mathbf{x}(k+1) \\ \mathbf{x}(k+2) \\ \vdots \\ \mathbf{x}(k+N)\end{bmatrix}, \widetilde{\mathbf{u}}(k) \equiv \begin{bmatrix}\mathbf{u}(k) \\ \mathbf{u}(k+1) \\ \vdots \\ \mathbf{u}(k+N-1)\end{bmatrix}, \widetilde{\mathbf{y}}(k) \equiv \begin{bmatrix}\mathbf{y}(k+1) \\ \mathbf{y}(k+2) \\ \vdots \\ \mathbf{y}(k+N)\end{bmatrix}. \tag{44}$$

Thus, given this formulation, any chain of inputs $\widetilde{\mathbf{u}}(k)$ has its corresponding set of future states $\widetilde{\mathbf{x}}(k)$ and outputs $\widetilde{\mathbf{y}}(k)$. Now, the state, input, and output vectors at an instant $k+i$ can be obtained with

$$\begin{aligned}\mathbf{x}(k+i) &= \mathbf{\Pi}_i^{(n,N)} \cdot \widetilde{\mathbf{x}}(k), \\ \mathbf{u}(k+i-1) &= \mathbf{\Pi}_i^{(n,N)} \cdot \widetilde{\mathbf{u}}(k), \\ \mathbf{y}(k+i) &= \mathbf{\Pi}_i^{(n_y,N)} \cdot \widetilde{\mathbf{y}}(k).\end{aligned} \tag{45}$$

Now, considering that all linear stable systems are quadratically stable [28], the cost function $J\left(\widetilde{\mathbf{u}}\middle|\mathbf{x}(k), \widetilde{\mathbf{w}}^d(k), \mathbf{u}^d\right)$ to find the best possible set of actions in the prediction horizon $[k, k+N]$ is presented [43]

$$J \equiv \sum_{i=1}^{N}\left\|\mathbf{\Pi}_i^{(n_y,N)} \cdot \widetilde{\mathbf{y}} - \mathbf{\Pi}_i^{(n_y,N)} \cdot \widetilde{\mathbf{w}}^d\right\|_{\mathbf{Q}}^2 + \sum_{i=1}^{N}\left\|\mathbf{\Pi}_i^{(n_u,N)} \cdot \widetilde{\mathbf{u}} - \mathbf{u}^d\right\|_{\mathbf{R}}^2 \tag{46}$$

In the first term, the error between the output trajectory $\widetilde{\mathbf{y}}(k) \in \mathcal{R}^{N\cdot n_y}$ and the reference $\widetilde{\mathbf{w}}^d(k) \in \mathcal{R}^{N\cdot n_y}$ is penalized by the positive definite matrix $\mathbf{Q} \in \mathcal{R}^{n_y\times n_y}$. Here, $\widetilde{\mathbf{w}}^d(k)$

contains the sequence of references $\mathbf{w}^d = \begin{bmatrix} \mathbf{w}_1^d & \mathbf{w}_2^d & \dots & \mathbf{w}_{n_y}^d \end{bmatrix}^T$, in which, depending on the prediction horizon $N$, $\widetilde{\mathbf{w}}^d(k)$ can be expressed as

$$\widetilde{\mathbf{w}}^d(k) \equiv \begin{bmatrix} \mathbf{w}^d(k+1) \\ \mathbf{w}^d(k+2) \\ \vdots \\ \mathbf{w}^d(k+N) \end{bmatrix} \tag{47}$$

Additionally, in the second term, the difference between the input vector $\widetilde{\mathbf{u}}(k)$ and the desired input $\mathbf{u}^d \in \mathcal{R}^{n_u}$ is penalized by the positive definite weighting matrix $\mathbf{R} \in \mathcal{R}^{n_u \times n_u}$. Therefore, the minimization problem is solved to find the input vector $\widetilde{\mathbf{u}}(k)$; however, only the first control input is implemented on the system. Then, the system moves to $k+1$ to solve the minimization problem again for the new control action.

The above cost function is rewritten using notations (45) and (46):

$$J \equiv \sum_{i=1}^{N} \left\| \mathbf{y}(k+i) - \mathbf{w}^d(k+i) \right\|_{\mathbf{Q}}^2 + \sum_{i=1}^{N} \left\| \mathbf{\Pi}_i^{(n_u,N)} \cdot \widetilde{\mathbf{u}} - \mathbf{u}^d \right\|_{\mathbf{R}}^2 \tag{48}$$

Hence, the general cost function implemented for the altitude controller in space state representation is as follows:

$$J \equiv \sum_{i=1}^{N} \left\| \mathbf{C}\mathbf{\Phi}_i \mathbf{x}(k) + \mathbf{C}\mathbf{\Psi}_i \widetilde{\mathbf{u}}(k) + \mathbf{C}\mathbf{\Theta}_i \widetilde{\mathbf{g}} - \mathbf{\Pi}_i^{(n_y,N)} \widetilde{\mathbf{w}}^d(k) \right\|_{\mathbf{Q}}^2 + \sum_{i=1}^{N} \left\| \mathbf{\Pi}_i^{(n_u,N)} \cdot \widetilde{\mathbf{u}} - \mathbf{u}^d \right\|_{\mathbf{R}}^2 \tag{49}$$

Thus, the control law can be obtained by rewriting the performance index in a more compact form, as follows:

$$J \equiv \frac{1}{2} \widetilde{\mathbf{u}}^T \mathbf{H} \widetilde{\mathbf{u}} + \left[ \mathbf{F}_1 \mathbf{x}(k) + \mathbf{F}_2 \widetilde{\mathbf{w}}^d(k) + \mathbf{F}_3 \mathbf{u}^d + \mathbf{F}_4 \right]^T \widetilde{\mathbf{u}} \tag{50}$$

where

$$\begin{aligned} \mathbf{H} &\equiv 2 \sum_{i=1}^{N} \left[ \mathbf{\Psi}_i^T \mathbf{C}^T \mathbf{Q} \mathbf{C} \mathbf{\Psi}_i + \left( \mathbf{\Pi}_i^{(n_u,N)} \right)^T \mathbf{Q} (\mathbf{\Pi}_i^{(n_u,N)}) \right] \\ \mathbf{F}_1 &\equiv 2 \sum_{i=1}^{N} \left[ \mathbf{\Psi}_i^T \mathbf{C}^T \mathbf{Q} \mathbf{C} \mathbf{\Phi}_i \right] \\ \mathbf{F}_2 &\equiv -2 \sum_{i=1}^{N} \left[ \mathbf{\Psi}_i^T \mathbf{C}^T \mathbf{Q} \mathbf{\Pi}_i^{(n_y,N)} \right] \\ \mathbf{F}_3 &\equiv -2 \sum_{i=1}^{N} \left[ \mathbf{\Pi}_i^{(n_u,N)^T} \mathbf{R} \right] \\ \mathbf{F}_4 &\equiv 2 \sum_{i=1}^{N} \left[ \mathbf{\Psi}_i^T \mathbf{C}^T \mathbf{Q} \mathbf{C} \mathbf{\Theta}_i \widetilde{\mathbf{g}} \right] \end{aligned} \tag{51}$$

Then, from (50), the control law is obtained:

$$\widetilde{\mathbf{u}}^{opt}(\mathbf{x}(k)) = -\mathbf{H}^{-1} \left[ \mathbf{F}_1 \mathbf{x}(k) + \mathbf{F}_2 \widetilde{\mathbf{w}}^d(k) + \mathbf{F}_3 \mathbf{u}^d + \mathbf{F}_4 \right] \tag{52}$$

When applying the first action of the control input chain $\widetilde{\mathbf{u}}^{opt}(\mathbf{x}(k))$, the MPC state feedback can be expressed as

$$K_{MPC}(\mathbf{x}(k)) = \mathbf{\Pi}_i^{(n_u,N)} \cdot \widetilde{\mathbf{u}}^{opt} \left( \mathbf{x}(k), \widetilde{\mathbf{w}}^d(k), \mathbf{u}^d \right) \tag{53}$$

Moreover, the optimization problem below is addressed at every sample time instant; therefore, the measurements of the states and outputs of the system are updated constantly.

$$\mathbf{P}(\mathbf{x}(\mathbf{k})) : \min_{\tilde{\mathbf{u}} \in \mathcal{R}^{Nn_u}} J\left(\tilde{\mathbf{u}} \middle| \mathbf{x}(k), \tilde{\mathbf{w}}^d(k), \mathbf{u}^d\right) \tag{54}$$

## 4. Quadrotor Control Structure

The overall system structure shown in Figure 2 for this implementation consists of two stages: the mission server and the commander stabilizer. The mission server stage is equipped with an OptiTrack® Flex13 motion tracking system, providing the UAV location in real-time at 120 Hz to the main PC workstation. Here, as the communication between the server and the UAV is continuous, complex reference trajectories are sent by the user in real time. Therefore, the actual and desired coordinates serve as inputs to the commander stabilizer stage. The commander stabilizer corresponds to the main elements of the Quanser® QDrone 2 platform, such as the onboard PC, battery, and rotors. The onboard PC, NVIDIA Jetson Xavier NX SOM, contains the proposed control structure MPC, a control input PWM translator, and a battery drop compensator. Additionally, the quadrotor includes a 4S 14.8 V LiPo (3700 mAh) battery to actuate four motors with 6-inch propellers, each providing a maximum of 8.5 N lift force. Thus, the mission server and the commander stabilizer use QUARC™ 2023 SP2 software to generate real-time code directly from SIMULINK®.

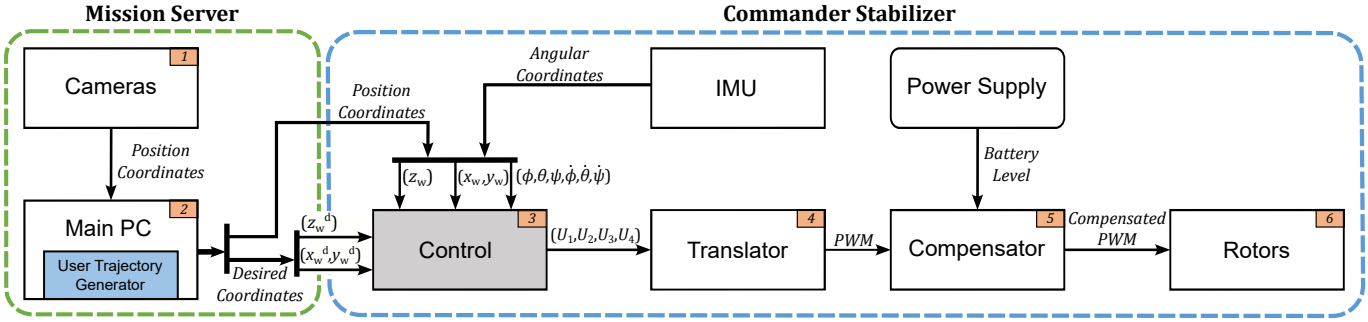



**Figure 2.** Overall quadrotor control structure.

Thus, based on Figure 2, the overall control process is described in detail:

1. The camera system provides the linear and yaw positions of the quadrotor, while the IMU of the quadrotor sends the angular status.
2. The trajectory generator commands the desired position in the inertial frame $(x_w^d, y_w^d, z_w^d)$. These will be the references for translational control.
3. The implemented cascaded control structures, PIV, and MPC receive the desired and actual position and rotational coordinates to compute $U_1, U_2, U_3, U_4$.
4. Once all the control actions are calculated, the transformation matrix provided by Quanser® is used to change the control inputs to pulse-width modulation (PWM).

$$T_m = \begin{bmatrix} 0.0489 & -0.4581 & 0.5566 & 5.1335 \\ 0.0489 & -0.4581 & -0.5566 & -5.1335 \\ 0.0489 & 0.4581 & 0.5566 & -5.1335 \\ 0.0489 & 0.4581 & -0.5566 & 5.1335 \end{bmatrix} \tag{55}$$

5. Most of the quadrotors in real-world implementations are powered by batteries [45]. Here, the reduction in power supply, named battery drainage, occurs gradually [46,47]. Therefore, the control performance of the drone is affected [48]. For this reason, we designed a compensation subsystem by measuring the altitude and battery voltage rate of change. Hence, a relationship between the PWM and the necessary battery level to reach hover position was obtained.

6. The rotors receive their compensated PWM control signal.

### *4.1. PIV Controller Design*

The main difference between the PIV controller and the well-known PID controller comes from the definition of the error [49]. In the PID, only the position error $e_p$ is considered, whereas in the PIV, the position and velocity errors, $e_p$ and $e_v$ , are considered. Here, the velocity error is obtained not from the time derivative of the position error, but from calculating the difference between the desired velocity and the actual velocity of the quadrotor. Then, the compared PIV controller can be defined with the following equation:

$$PIV = K_p e_p + K_i e_i + K_v e_v \tag{56}$$

where $e_i$ stands for the position error integral with respect to time of $e_p$, and $K_p$, $K_i$, $K_v$ represent the proportional, integrative, and velocity constants, respectively. Thus, Figure 3 shows the PIV control structure.

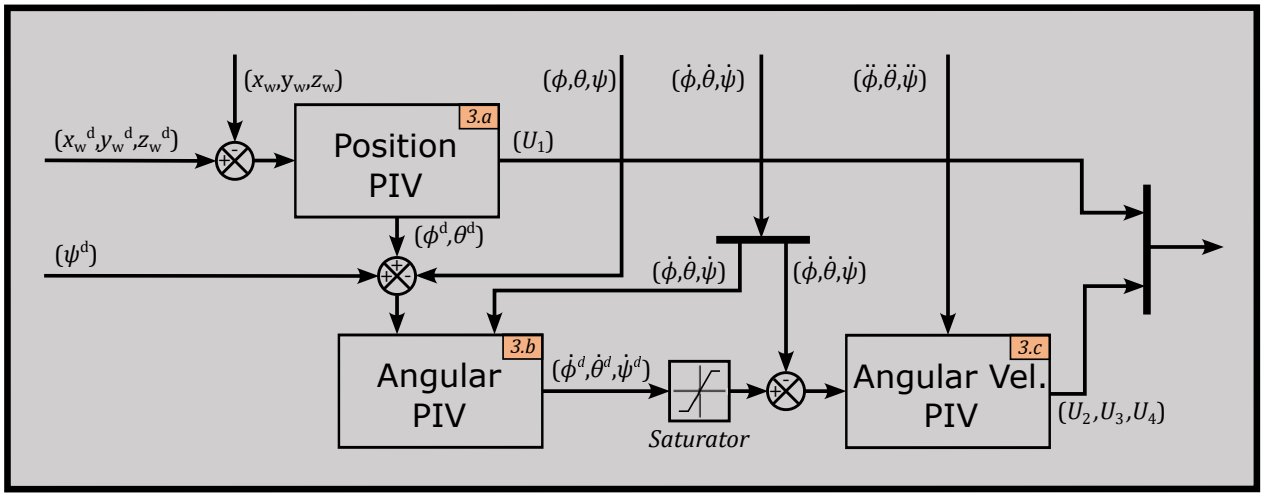



**Figure 3.** Block diagram of the PIV structure.

Hence, based on Figures 2 and 3, the overall PIV control process is described in detail:

a. The position PIV controller obtains the desired $x_w^d, y_w^d, z_w^d$ positions from the trajectory generator while the motion tracking system provides the actual $x_w, y_w, z_w$ position of the drone. Here, the thrust input $U_1$ and the angular reference $\phi^d, \theta^d$ of the angular PIV are computed as follows:

$$U_1 = k_{pz_w} e_{z_w} + k_{iz_w} \int_0^\infty e_{z_w} - k_{vz_w} \dot{z}_w \tag{57}$$

$$\phi^d = k_{py_w} e_{y_w} + k_{iy_w} \int_0^\infty e_{y_w} - k_{vy_w} \dot{y}_w \tag{58}$$

$$\theta^d = k_{px_w} e_{x_w} + k_{ix_w} \int_0^\infty e_{x_w} - k_{vx_w} \dot{x}_w \tag{59}$$

b. Then, the angular PIV receives the desired angular positions $(\phi^d, \theta^d, \psi^d)$, the actual angular positions $(\phi, \theta, \psi)$, and the actual angular velocities $(\dot{\phi}, \dot{\theta}, \dot{\psi})$. These are processed to compute the reference angular velocities $(\dot{\phi}^d, \dot{\theta}^d, \dot{\psi}^d)$ of the PIV velocities, which are saturated with a minimum and a maximum rate of $-45°$ and $45°$, respectively.

$$\dot{\phi}^d = k_{p\phi} e_\phi - k_{v\phi} \dot{\phi} \tag{60}$$

$$\dot{\theta}^d = k_{p\theta} e_\theta - k_{v\theta} \dot{\theta} \tag{61}$$

$$\dot{\psi}^d = k_{p\psi} e_\psi - k_{v\psi} \dot{\psi} \tag{62}$$

c. The angular PIV velocity gathers the desired angular velocities $(\dot{\phi}^d, \dot{\theta}^d, \dot{\psi}^d)$, the actual angular velocities $(\dot{\phi}, \dot{\theta}, \dot{\psi})$, and the actual angular accelerations $(\ddot{\phi}, \ddot{\theta}, \ddot{\psi})$. Then, it computes the optimal angular torques $(U_2, U_3, U_4)$; this is performed by considering the thrust input $(U_1)$ computed in step (a).

$$U_2 = k_{p\dot{\phi}} e_{\dot{\phi}} - k_{v\dot{\phi}} \ddot{\phi} \tag{63}$$

$$U_3 = k_{p\dot{\theta}} e_{\dot{\theta}} - k_{v\dot{\theta}} \ddot{\theta} \tag{64}$$

$$U_4 = k_{p\dot{\psi}} e_{\dot{\psi}} \tag{65}$$

The PIV controllers always stabilize with respect to the attitude commands' set point. Here, the attitude torques are added to the output of the remaining PIV controllers to generate the generalized force.

### *4.2. MPC Controller Design*

The proposed cascaded MPC translational and rotational structure is shown in Figure 4, which considers the dependency between the linear displacement and the rotation of the drone.

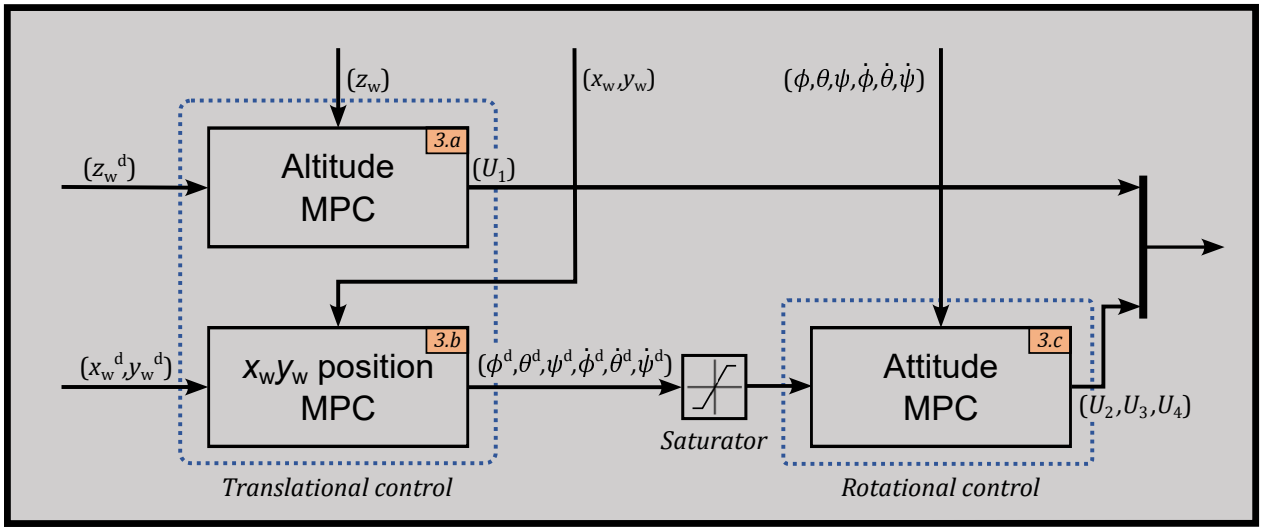



**Figure 4.** Block diagram of the cascaded MPC structure.

The translational control (outer loop) tracks the altitude while computing the thrust input $(U_1)$, and the $x_w y_w$ position, feeding the reference orientations to the rotational control. Then, the rotational control (inner loop) tracks the attitude, applying feasible torques $(U_2, U_3, U_4)$ to the quadrotor.

The proposed cascaded MPC used to fulfill the linear drone dynamics works as follows:

a. The altitude controller obtains the reference $z_w^d$ from the trajectory generator while the motion tracking system provides the actual $z_w$ position of the drone. Here, the thrust input $U_1$ is computed.
b. On the other hand, $x_w$ and $y_w$ accelerations depend directly on the rotation of the drone. Thus, the dynamics are dependent, and the controllers must be related. This

relation is described in the dynamical models (29), (30), where it is shown that the product between gravity and the respective angle gives a linear displacement. Then, the output of the $x_w y_w$ MPC algorithm is saturated with a minimum and maximum of $-45°$ and $45°$, respectively, and the desired angles are sent to the attitude controller.

c. The attitude MPC receives the following: from the motion tracking system, the angular positions $(\phi, \theta, \psi)$ and the angular velocities $(\dot{\phi}, \dot{\theta}, \dot{\psi})$; and from the $x_w y_w$ MPC algorithm, the saturated orientation references $(\phi^d, \theta^d, \psi^d, \dot{\phi}^d, \dot{\theta}^d, \dot{\psi}^d)$. Then, the attitude MPC uses this information to obtain the optimal angular torques $(U_2, U_3, U_4)$; factoring into the calculations the thrust input $(U_1)$ computed in step (a).

Additionally, for the translational and rotational control described in detail in this section, each MPC controller works according to the following algorithm in Table 1.

**Table 1.** Methodology that describes each MPC controller implemented.

| **MPC Methodology** | |
|---|---|
| Input: | Actual states of the quadrotor and desired values |
| Output: | Optimal manipulation |
| 1 | Define the LTI mathematical model of the physical system in state space representation |
| 2 | Define the sampling time and discretize the state space representation |
| 3 | Set the sampling instants $N$ for the prediction horizon |
| 4 | Compute the matrices $\mathbf{\Phi}_i$,$\mathbf{\Psi}_i$ and $\mathbf{\Theta}_i, \forall i \in 1, \ldots, N$ used to represent the $N$-step-ahead prediction map for LTI systems |
| 5 | Define weight matrices $\mathbf{Q}$ and $\mathbf{R}$ |
| 6 | Determine the saturations |
| 7 | Compute the online matrices $\mathbf{H}$ and $\mathbf{F}$ |
| 8 | Obtain the control law $\tilde{\mathbf{u}}^{opt}$ and apply to the system the first control input of the best control input sequence |

## 5. Real-Time Implementation

### *5.1. Experimental Setup*

In this section, flight experiments are carried out to validate the effectiveness of the proposed cascaded MPC strategy compared to the traditional PIV control scheme. Table 2 shows the quadrotor parameters of the QDrone 2 for the real-time implementation programmed using MATLAB® SIMULINK® [50].

**Table 2.** Parameters of the drone used for testing.

| **QDrone 2 Parameters** | | |
|---|---|---|
| Dimensions | $50 \times 50 \times 15$ cm | - |
| Mass | 1500 g | - |
| Max. Payload | 300 g | - |
| $K_v$ | 2100 $^{\text{RPM}}/_{\text{V}}$ | - |
| Max. Continuous Current | 25 A | - |
| $K_{v,eff}$ | 1295.4 $^{\text{RPM}}/_{\text{V}}$ | Effective motor speed constant |
| $W_c$ | 2132.6 RPM | Voltage-to-angular velocity offset |
| $W_f$ | 1004.5 RPM | Angular velocity-to-force offset |
| $C_t$ | $2.0784 \times 10^{-8}$ $^{\text{N}}/_{\text{RPM}^2}$ | Motor force constant |
| $F_b$ | $-0.2046$ N | Motor force offset |
| $k_T$ | 81.0363 $^{\text{N}}/_{\text{Nm}}$ | Motor thrust–torque constant |

The constant values for the PIV implementation are shown in Table 3 [28,35]. Here, the PIV controller parameters $K_p$, $K_i$, and $K_v$ are defined for the positions $(x_w, y_w, z_w)$, the angular positions $(\phi, \theta, \psi)$, and the angular velocities $(\dot{\phi}, \dot{\theta}, \dot{\psi})$ of the drone.

**Table 3.** PIV tuning parameters.

| PIV Parameters | Position PIV $[x_w \quad y_w \quad z_w]$ | Angular PIV $[\phi \quad \theta \quad \psi]$ | Angular Vel. PIV $[\dot{\phi} \quad \dot{\theta} \quad \dot{\psi}]$ |
|---|---|---|---|
| $K_p$ | $[\frac{\pi}{2} \ \frac{\pi}{2} \ 35]$ | [12 12 15] | [0.188 0.154 0.040] |
| $K_i$ | [0 0 6] | - | - |
| $K_v$ | $[\frac{\pi}{2} \ \frac{\pi}{3} \ 28]$ | [0.1 0.1 4] | [0.003 0.003 0] |

On the other hand, the tunning parameters for a desired closed loop performance of each MPC controller are presented in Table 4.

**Table 4.** MPC tuning parameters.

| MPC Parameters | Altitude MPC | $x_w y_w$ Position MPC | Attitude MPC |
|---|---|---|---|
| **Q** matrix | 10 | [1000 500] | [1500 750 1500] |
| **R** matrix | 0.00001 | [75 50] | [200 500 250] |
| Prediction horizon $N$ | 10 | 10 | 15 |
| Sample time $k$ | 0.25 | 0.05 | 0.025 |

### *5.2. Experimental Results*

To validate the performance of the proposed cascaded MPC controllers over the conventional PIV control structure, the following three scenarios are real-time implemented:

- Scenario 1. Helicoidal trajectory;
- Scenario 2. Lemniscate trajectory;
- Scenario 3. Wind disturbance rejection.

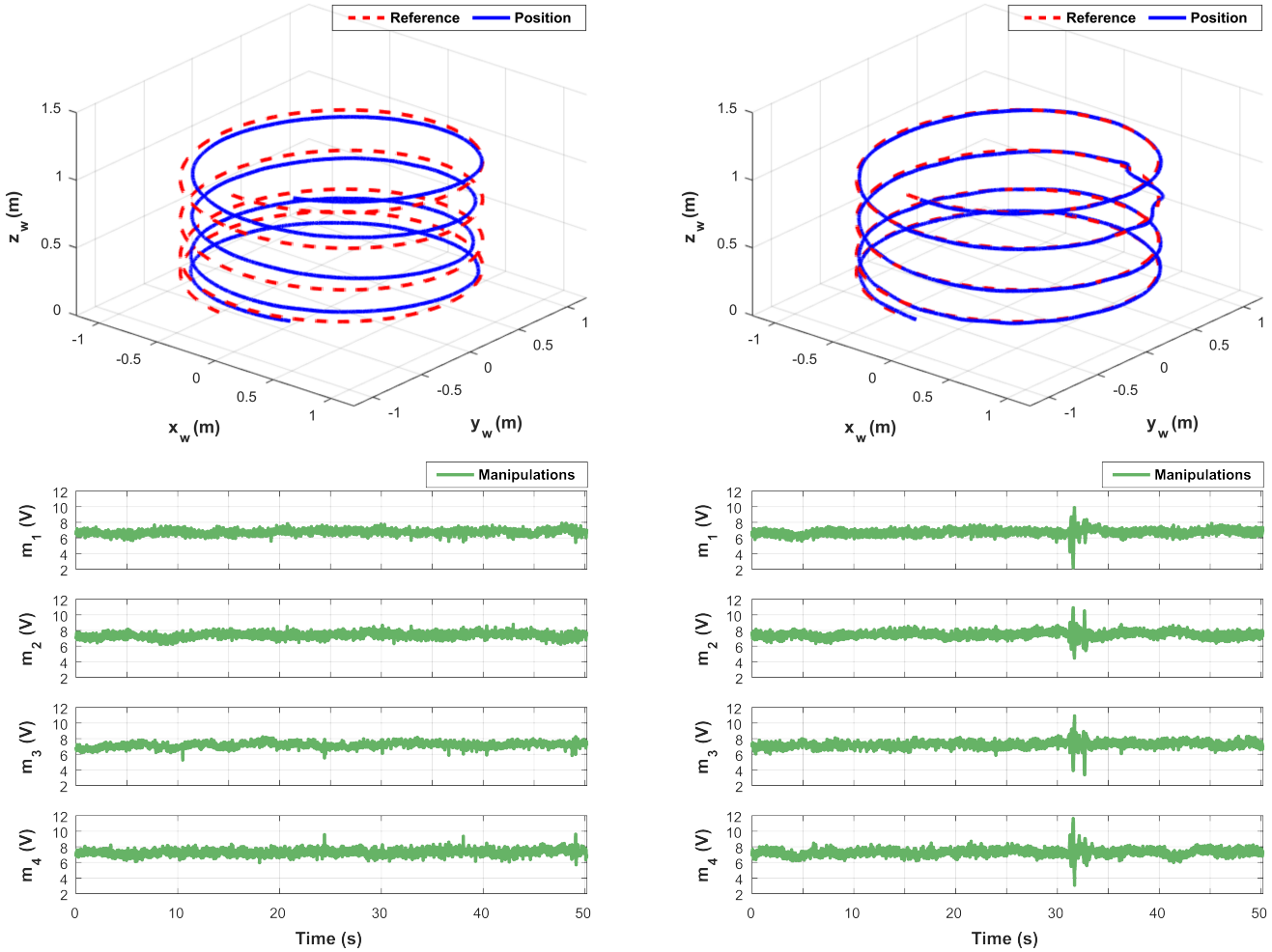
Hence, to demonstrate the ability of a quadrotor in automatic flight mode, scenario 1 and scenario 2 are selected, considering that they are commonly used as a standard benchmark for tracking control [37]. Meanwhile, to test the performance of the quadrotor under external disturbances, scenario 3 is selected for regulatory control [51]. Therefore, the positions $(x_w, y_w, z_w)$ and the PWM signals in voltage of each rotor $(m_1, m_2, m_3, m_4)$ based on the control inputs $(U_1, U_2, U_3, U_4)$ must be plotted.

#### 5.2.1. Scenario 1: Helicoidal Trajectory

Regarding the first scenario, the helicoidal trajectory is given by Equation (66). Figure 5 shows the comparison between the proposed MPC controller performance with respect to PIV control structure.

$$\begin{aligned} x_w^d &= \sin(0.5t) \\ y_w^d &= \cos(0.5t) \\ z_w^d &= 0.5\sin(0.05t) \end{aligned} \tag{66}$$

Here, even though the quadrotor under the MPC control strategy had a communication error between the main PC workstation and the UAV at 35 s, the comparison indicates that the MPC does not present steady-state errors compared to the PIV control method.




(**a**) PIV tracking (**b**) MPC tracking

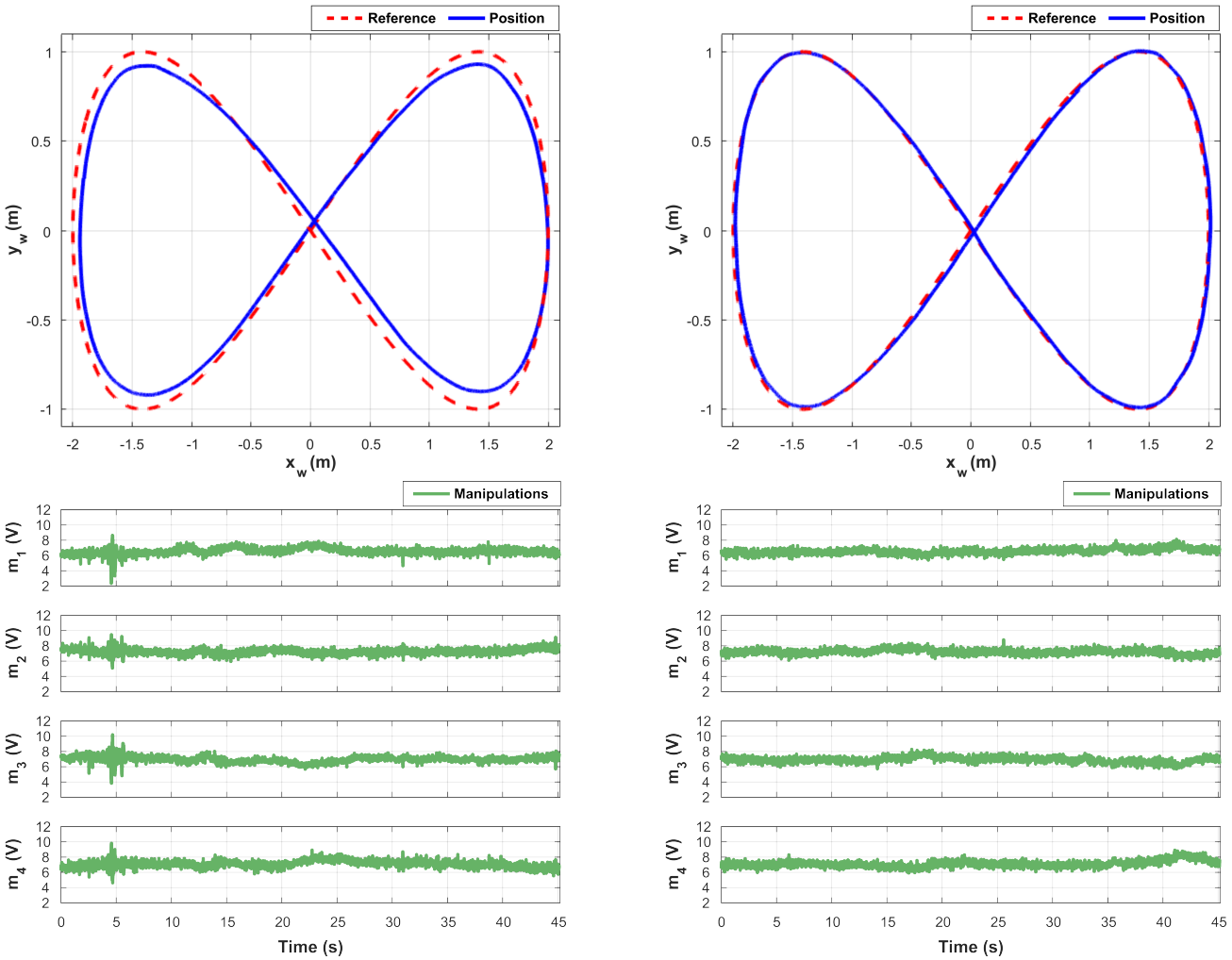
**Figure 5.** Helicoidal trajectory scenario.

#### 5.2.2. Scenario 2: Lemniscate Trajectory

For the second scenario, a lemniscate trajectory is programmed in the main PC workstation based on the Equation (67). Figure 6 shows the comparison between the proposed MPC controller performance with respect to PIV control strategy.

$$\begin{aligned} x_w^d &= 2\cos\left(\tfrac{1}{45}t\right) \\ y_w^d &= \sin\left(\tfrac{2}{45}t\right) \end{aligned} \tag{67}$$

Similarly to scenario 1, the tracking results for scenario 2 show that the PIV control presents steady-state errors.




(**a**) PIV tracking (**b**) MPC tracking

**Figure 6.** Lemniscate trajectory scenario.

#### 5.2.3. Scenario 3: Disturbance Rejection

To assess the performance of the MPC and the PIV under external disturbances, the third scenario is considered in the presence of wind gusts. Initially, the quadrotor is in a hovering position according to the inertial frame (Equation (68)).

$$x_w^d = -1.987 \text{ [m]}$$

$$y_w^d = 0.21 \text{ [m]} \tag{68}$$

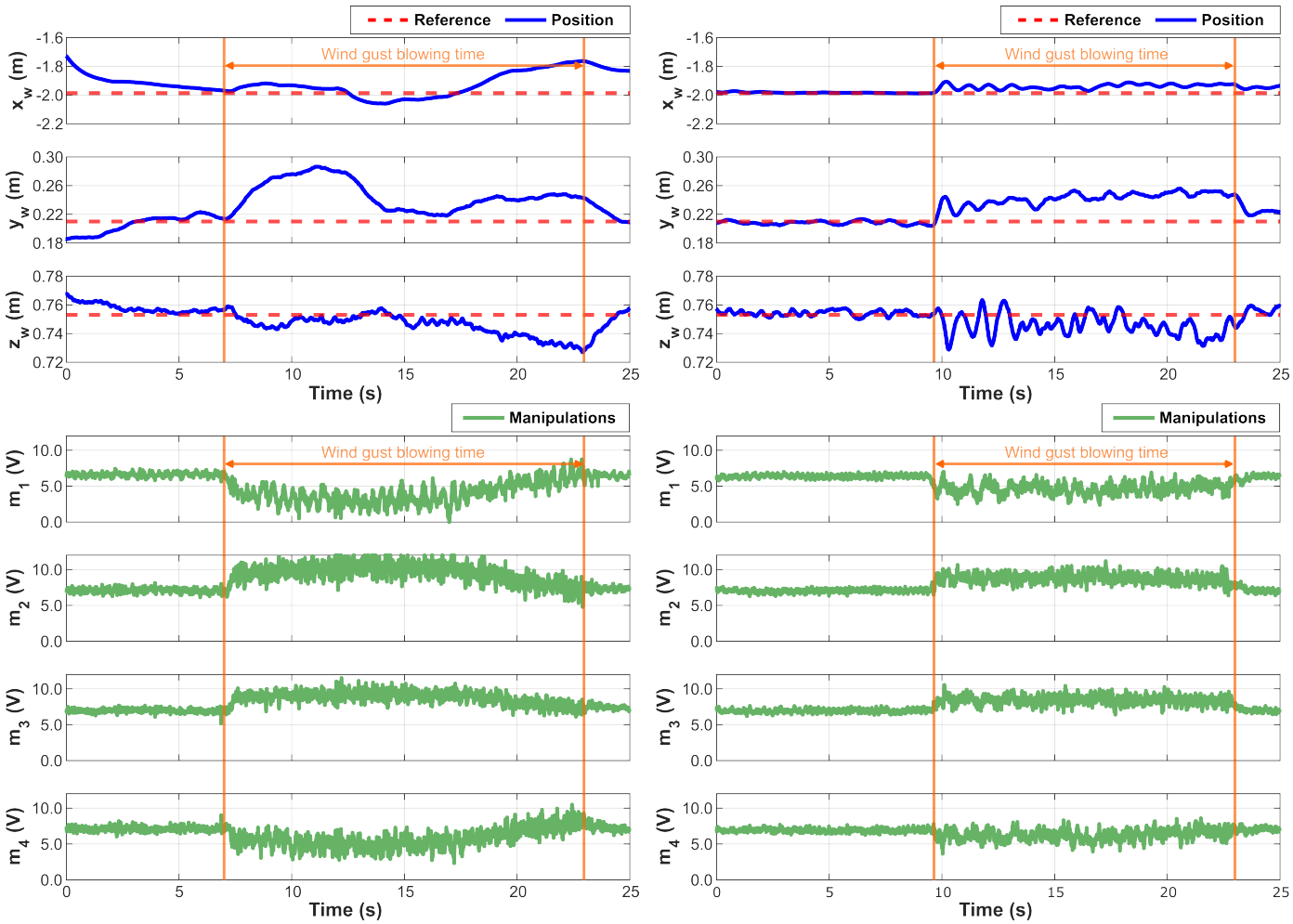
$$z_w^d = 0.753 \text{ [m]}$$

A leaf blower with a blowing rate of 900 [$m^3$/h] at a speed of 177 [km/h] is placed at a distance of 3 [m] away at a 45° angle. The duration of the random disturbance applied for the quadrotor under the PIV control structure is 15.96 [s], in which 3193 time steps are collected; this is considering a sample time of 5 [ms]. Meanwhile, the duration of the random disturbance under the MPC control structure is 13.355 [s], which implies the collection of 2672 time steps of 5 [ms]. Table 5 shows the specific time in which the disturbance is applied to the quadrotor under both control structures. As is shown in Figure 7, the wind gusts produced greater disturbance against the PIV method controller

than the proposed cascaded MPC. In addition, when using the proposed cascaded MPC, the signal effort presents less fluctuations.

**Table 5.** Time lapses of the wind disturbance.

| Quadrotor | Start | End |
|---|---|---|
| Under PIV | 7.005 s | 22.965 s |
| Under MPC | 9.645 s | 23 s |



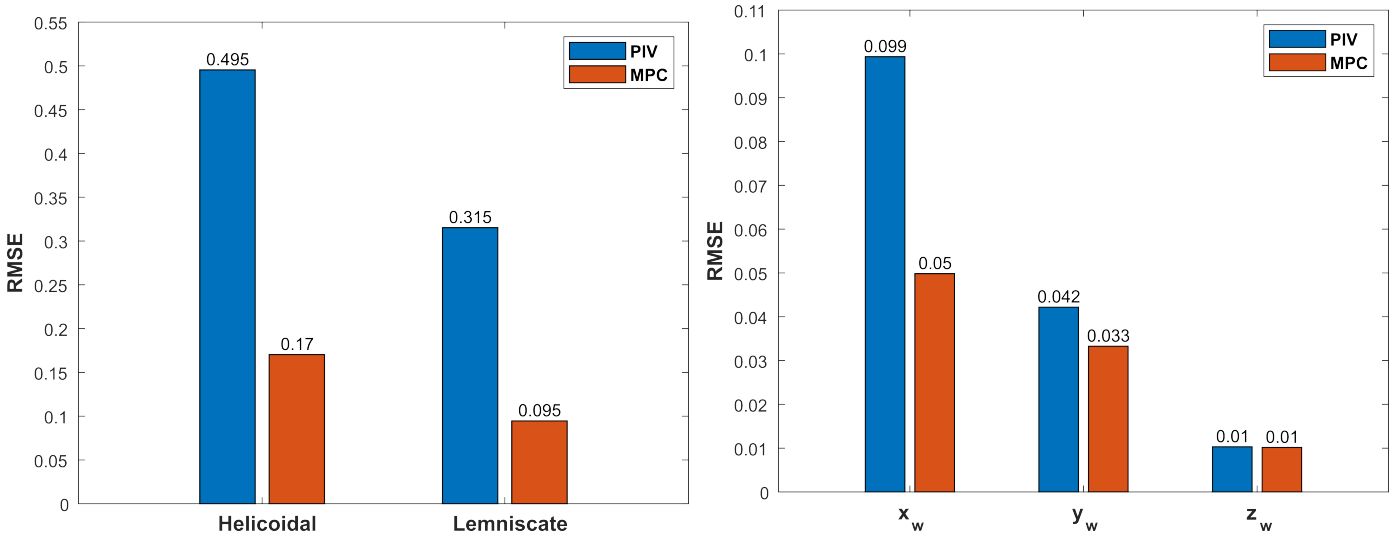

(**a**) PIV (**b**) MPC

**Figure 7.** Disturbance rejection scenario.

Thus, to verify the efficiency of both controllers, a root-mean-squared error (RMSE) comparison of tracking performance for each scenario is made (Equation (69)).

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{\left(y_i^d - y_i\right)^2}{n}} \tag{69}$$

where $n$ is the number of time steps, $y_i^d$ is the reference value, and $y_i$ is the measured value. Figure 8 shows the RMSE comparison, which includes 10,039 and 9026 collected time steps of 5 [ms] for the helicoidal and lemniscate trajectories, respectively. The lower the RMSE value, the better the tracking performance.




(**a**) Tracking comparative (**b**) Regulator comparative

**Figure 8.** RMSE comparison.

Based on the result of the practical scenarios, it can be concluded that the proposed MPC controller improves the overall tracking performance in comparison with the PIV control method. The RMSE comparison analysis from the experimental results indicates that the proposed cascaded MPC architecture outperforms the PIV method. During the helicoidal trajectory, an RMS value of 0.17 is achieved for the proposed MPC, while 0.495 is obtained using the PIV strategy. Moreover, at the end of the lemniscate trajectory, RMS values of 0.095 and 0.315 are obtained for the proposed MPC and PIV structures, respectively. Therefore, it can be seen that the PIV controller cannot adequately control the quadrotor [52].

## 6. Conclusions

This paper proposes a multivariable cascaded MPC structure for real-time implementation in quadrotors. The proposed architecture is used to test the translational and attitude control of the QDrone 2 by Quanser®. For tracking control, helicoidal and lemniscate trajectories are chosen to demonstrate the ability of the quadrotor in automatic flight mode. Additionally, for regulatory control, wind gusts are applied to the quadrotor in a hovering position to validate disturbance rejection. For tracking control, using the proposed cascaded MPC architecture, the RMS values are reduced by more than a half. Similarly, lower RMS values for the three positions of the quadrotor are achieved in the presence of external disturbances, especially for $x_w$ position, which receives the wind gusts directly. Thus, higher tracking and regulatory performances are obtained. Our future work will be a maneuver control of multiple drones by using the proposed method and applying particle swarm optimization.

**Author Contributions:** Conceptualization, P.B.-B., D.S. and C.S.; methodology, D.S., M.F., L.E.G.-C., L.J.-M. and C.S.; validation, D.S., L.J.-M. and C.S.; investigation, D.S., M.F., L.E.G.-C. and C.S.; resources, D.S., M.F. and C.S.; writing—original draft, P.B.-B., D.S., L.J.-M. and C.S.; writing—review and editing, D.S., M.F., L.E.G.-C., L.J.-M. and C.S.; supervision, D.S., M.F., L.E.G.-C. and C.S.; project administration, D.S., M.F. and L.E.G.-C. and C.S.; funding acquisition, D.S, M.F. and C.S. All authors have read and agreed to the published version of the manuscript.

**Funding:** This material is based upon work supported by, or in part by, the U. S. Army Research Laboratory and the U. S. Army Research Office under contract/grant number W911NF2010260.

**Data Availability Statement:** The data presented in this study are available on request from the corresponding author. The data are not publicly available to provide specific detail upon request.

**Acknowledgments:** The authors would like to acknowledge the use of the SWARM Lab at the University of the Incarnate Autonomous Vehicle Systems Research Laboratories.

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

## References

1. Idrissi, M.; Salami, M.; Annaz, F. A review of quadrotor unmanned aerial vehicles: Applications, architectural design and control algorithms. *J. Intell. Robot. Syst.* **2022**, *104*, 22. [CrossRef]
2. Al Tahtawi, A.R.; Yusuf, M. Low-cost quadrotor hardware design with PID control system as flight controller. *TELKOMNIKA (Telecommun. Comput. Electron. Control.)* **2019**, *17*, 1923–1930. [CrossRef]
3. Shakeel, T.; Arshad, J.; Jaffery, M.H.; Rehman, A.U.; Eldin, E.T.; Ghamry, N.A.; Shafiq, M. A Comparative Study of Control Methods for X3D Quadrotor Feedback Trajectory Control. *Appl. Sci.* **2022**, *12*, 9254. [CrossRef]
4. Rinaldi, M.; Primatesta, S.; Guglieri, G. A comparative study for control of quadrotor uavs. *Appl. Sci.* **2023**, *13*, 3464. [CrossRef]
5. Chovancová, A.; Fico, T.; Duchoň, F.; Dekan, M.; Chovanec, L.; Dekanova, M. Control methods comparison for the real quadrotor on an innovative test stand. *Appl. Sci.* **2020**, *10*, 2064. [CrossRef]
6. Benaddy, A.; Bouzi, M.; Labbadi, M. Comparison of the different control strategies for Quadrotor Unmanned Aerial Vehicle. In Proceedings of the 2020 International Conference on Intelligent Systems and Computer Vision (ISCV), Fez, Morocco, 9–11 June 2020; pp. 1–6.
7. Kang, C.; Park, B.; Choi, J. Scheduling PID attitude and position control frequencies for time-optimal quadrotor waypoint tracking under unknown external disturbances. *Sensors* **2021**, *22*, 150. [CrossRef]
8. Nazin, A.; Alazki, H.; Poznyak, A. Robust Tracking as Constrained Optimization by Uncertain Dynamic Plant: Mirror Descent Method and ASG—Version of Integral Sliding Mode Control. *Mathematics* **2023**, *11*, 4112. [CrossRef]
9. Bøhn, E.; Gros, S.; Moe, S.; Johansen, T.A. Reinforcement learning of the prediction horizon in model predictive control. *IFAC-PapersOnLine* **2021**, *54*, 314–320. [CrossRef]
10. Veksler, A.; Johansen, T.A.; Borrelli, F.; Realfsen, B. Dynamic positioning with model predictive control. *IEEE Trans. Control Syst. Technol.* **2016**, *24*, 1340–1353. [CrossRef]
11. Mohamed, O.; Wang, J.; Al-Duri, B.; Lu, J.; Gao, Q.; Xue, Y.; Liu, X. Predictive control of coal mills for improving supercritical power generation process dynamic responses. In Proceedings of the 2012 IEEE 51st IEEE Conference on Decision and Control (CDC), Maui, HI, USA, 10–13 December 2012; pp. 1709–1714.
12. Polisano, F.; Ryals, A.D.; Pannocchia, G.; Landi, A. MPC based optimization applied to treatment of HCV infections. *Comput. Methods Programs Biomed.* **2021**, *210*, 106383. [CrossRef]
13. Bardaro, G.; Bascetta, L.; Ceravolo, E.; Farina, M.; Gabellone, M.; Matteucci, M. MPC-based control architecture of an autonomous wheelchair for indoor environments. *Control Eng. Pract.* **2018**, *78*, 160–174. [CrossRef]
14. Skjong, E.; Johansen, T.A.; Molinas, M. Distributed control architecture for real-time model predictive control for system-level harmonic mitigation in power systems. *ISA Trans.* **2019**, *93*, 231–243. [CrossRef] [PubMed]
15. Carlet, P.G.; Toso, F.; Favato, A.; Bolognani, S. A speed and current cascade Continuous Control Set Model Predictive Control architecture for synchronous motor drives. In Proceedings of the 2019 IEEE Energy Conversion Congress and Exposition (ECCE), Baltimore, MD, USA, 29 September–3 October 2019; pp. 5682–5688.
16. Palmieri, A.; Rosini, A.; Procopio, R.; Bonfiglio, A. An MPC-sliding mode cascaded control architecture for PV grid-feeding inverters. *Energies* **2020**, *13*, 2326. [CrossRef]
17. Kumar, P.; Rawlings, J.B.; Carrette, P. Modeling proportional–integral controllers in tracking and economic model predictive control. *J. Process Control* **2023**, *122*, 1–12. [CrossRef]
18. Benotsmane, R.; Vásárhelyi, J. Towards optimization of energy consumption of tello quad-rotor with mpc model implementation. *Energies* **2022**, *15*, 9207. [CrossRef]
19. Xue, R.; Dai, L.; Huo, D.; Xie, H.; Sun, Z.; Xia, Y. Compound tracking control based on MPC for quadrotors with disturbances. *J. Frankl. Inst.* **2022**, *359*, 7992–8013. [CrossRef]
20. Eskandarpour, A.; Sharf, I. A constrained error-based MPC for path following of quadrotor with stability analysis. *Nonlinear Dyn.* **2020**, *99*, 899–918. [CrossRef]
21. Westenberger, J.; De Wagter, C.; de Croon, G.C. Efficient Bang-Bang Model Predictive Control for Quadcopters. *Unmanned Syst.* **2022**, *10*, 395–405. [CrossRef]
22. Schlagenhauf, J.; Hofmeier, P.; Bronnenmeyer, T.; Paelinck, R.; Diehl, M. Cascaded nonlinear mpc for realtime quadrotor position tracking. *IFAC-PapersOnLine* **2020**, *53*, 7026–7032. [CrossRef]
23. Nan, F.; Sun, S.; Foehn, P.; Scaramuzza, D. Nonlinear MPC for quadrotor fault-tolerant control. *IEEE Robot. Autom. Lett.* **2022**, *7*, 5047–5054. [CrossRef]
24. Hanover, D.; Foehn, P.; Sun, S.; Kaufmann, E.; Scaramuzza, D. Performance, precision, and payloads: Adaptive nonlinear mpc for quadrotors. *IEEE Robot. Autom. Lett.* **2021**, *7*, 690–697. [CrossRef]
25. Zhao, C.; Wang, D.; Hu, J.; Pan, Q. Nonlinear model predictive control-based guidance algorithm for quadrotor trajectory tracking with obstacle avoidance. *J. Syst. Sci. Complex.* **2021**, *34*, 1379–1400. [CrossRef]

26. Pereira, J.C.; Leite, V.J.; Raffo, G.V. Nonlinear model predictive control on SE (3) for quadrotor aggressive maneuvers. *J. Intell. Robot. Syst.* **2021**, *101*, 1–15. [CrossRef]
27. Kamel, M.A.; Hafez, A.T.; Yu, X. A review on motion control of unmanned ground and aerial vehicles based on model predictive control techniques. *J. Eng. Sci. Mil. Technol.* **2018**, *2*, 10–23.
28. Wang, S.; Polyakov, A.; Zheng, G. Generalized homogenization of linear controllers: Theory and experiment. *Int. J. Robust Nonlinear Control* **2021**, *31*, 3455–3479. [CrossRef]
29. Lambert, P.; Reyhanoglu, M. Observer-based sliding mode control of a 2-DOF helicopter system. In Proceedings of the IECON 2018—44th Annual Conference of the IEEE Industrial Electronics Society, Washington, DC, USA, 21–23 October 2018; pp. 2596–2600.
30. Mendez, A.P.; Whidborne, J.F.; Chen, L. Wind Preview-Based Model Predictive Control of Multi-Rotor UAVs Using LiDAR. *Sensors* **2023**, *23*, 3711. [CrossRef] [PubMed]
31. Jiang, Y.; Hu, S.; Damaren, C.; Luo, L.; Liu, B. Trajectory Planning with Collision Avoidance for Multiple Quadrotor UAVs Using DMPC. *Int. J. Aeronaut. Space Sci.* **2023**, *24*, 1403–1417. [CrossRef]
32. Landolfi, E.; Natale, C. An adaptive cascade predictive control strategy for connected and automated vehicles. *Int. J. Adapt. Control Signal Process.* **2023**, *37*, 2725–2751. [CrossRef]
33. Sajjadi, S.; Mehrandezh, M.; Janabi-Sharifi, F. A Cascaded and Adaptive Visual Predictive Control Approach for Real-Time Dynamic Visual Servoing. *Drones* **2022**, *6*, 127. [CrossRef]
34. 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]
35. Quanser. Qdrone—Quanser. 2020. Available online: https://www.quanser.com/products/qdrone (accessed on 6 May 2020).
36. Alaiwi, Y.; Mutlu, A. Modelling, simulation and implementation of autonomous unmanned quadrotor. *Mach. Technol. Mater.* **2018**, *12*, 320–325.
37. Kapnopoulos, A.; Alexandridis, A. A cooperative particle swarm optimization approach for tuning an MPC-based quadrotor trajectory tracking scheme. *Aerosp. Sci. Technol.* **2022**, *127*, 107725. [CrossRef]
38. Islam, M.; Okasha, M.; Idres, M. Dynamics and control of quadcopter using linear model predictive control approach. In *Proceedings of the IOP Conference Series: Materials Science and Engineering*; IOP Publishing: Bristol, UK, 2017; Volume 270, p. 012007.
39. Yan, D.; Zhang, W.; Chen, H. Design of a multi-constraint formation controller based on improved MPC and consensus for quadrotors. *Aerospace* **2022**, *9*, 94. [CrossRef]
40. Roy, R.; Islam, M.; Sadman, N.; Mahmud, M.A.P.; Gupta, K.D.; Ahsan, M.M. A Review on Comparative Remarks, Performance Evaluation and Improvement Strategies of Quadrotor Controllers. *Technologies* **2021**, *9*, 37. [CrossRef]
41. Saeed, A.S.; Younes, A.B.; Islam, S.; Dias, J.; Seneviratne, L.; Cai, G. A review on the platform design, dynamic modeling and control of hybrid UAVs. In Proceedings of the 2015 International Conference on Unmanned Aircraft Systems (ICUAS), Denver, CO, USA, 9–12 June 2015; pp. 806–815. [CrossRef]
42. Zhang, X.; Li, X.; Wang, K.; Lu, Y. A survey of modelling and identification of quadrotor robot. *Abstr. Appl. Anal.* **2014**, *2014*, 320526. [CrossRef]
43. Alamir, M. *A Pragmatic Story of Model Predictive Control: Self Contained Algorithms and Case-Studies*; CreateSpace Independent Publishing Platform: Scotts Valley, CA, USA, 2013.
44. Chen, K.; Zhu, Z.; Zeng, X.; Wang, J. Distributed Observers for State Omniscience with Stochastic Communication Noises. *Mathematics* **2023**, *11*, 1997. [CrossRef]
45. Liu, Z.; Yuan, C.; Zhang, Y. Active fault-tolerant control of unmanned quadrotor helicopter using linear parameter varying technique. *J. Intell. Robot. Syst.* **2017**, *88*, 415–436. [CrossRef]
46. Mehmet, E. Neural network assisted computationally simple pid control of a quadrotor UAV. *IEEE Trans. Ind. Inform.* **2011**, *7*, 354–361.
47. Efe, M.Ö. Battery power loss compensated fractional order sliding mode control of a quadrotor UAV. *Asian J. Control* **2012**, *14*, 413–425. [CrossRef]
48. Pi, C.H.; Hu, K.C.; Cheng, S.; Wu, I.C. Low-level autonomous control and tracking of quadrotor using reinforcement learning. *Control Eng. Pract.* **2020**, *95*, 104222. [CrossRef]
49. Alkomy, H.; Shan, J. Vibration reduction of a quadrotor with a cable-suspended payload using polynomial trajectories. *Nonlinear Dyn.* **2021**, *104*, 3713–3735. [CrossRef] [PubMed]
50. Labbadi, M.; Chatri, C.; Boubaker, S.; Kamel, S. Fixed-Time Controller for Altitude/Yaw Control of Mini-Drones: Real-Time Implementation with Uncertainties. *Mathematics* **2023**, *11*, 2703. [CrossRef]
51. Aliyari, M.; Wong, W.K.; Bouteraa, Y.; Najafinia, S.; Fekih, A.; Mobayen, S. Design and implementation of a constrained model predictive control approach for unmanned aerial vehicles. *IEEE Access* **2022**, *10*, 91750–91762. [CrossRef]
52. Utkin, A.V.; Utkin, V.A.; Krasnova, S.A. Synthesis of a Control System for a Waste Heat Boiler with Forced Circulation under Restrictions on Control Actions. *Mathematics* **2022**, *10*, 2397. [CrossRef]