# Distributed Nonlinear Model Predictive Control for a Quadrotor UAV

Bilal Mubdir[1] and Emmanuel Prempain[1]

[1]University of Leicester, University Rd

March 19, 2024

**Abstract**

A Distributed Nonlinear Model Predictive Control (DNMPC) approach is proposed to control the simplified decoupled dynamics of a quadrotor UAV. The performance of DNMPC is compared, in terms of tracking and execution time, to that of standard control configurations based on centralized MPC and PID control aiming to show the suitability of each configuration in terms of performance and the practicality of using a particular configuration in real-time applications. The results show the advantage of using DN-MPC in terms of ease of tuning and computational cost over more centralized feedback control approaches.

# Distributed Nonlinear Model Predictive Control for a Quadrotor UAV

Bilal Mubdir*, and Emmanuel Prempain
University of Leicester, University Rd, Leicester, UK



ABSTRACT

**A** Distributed Nonlinear Model Predictive Control (DNMPC) approach is proposed to control the simplified decoupled dynamics of a quadrotor UAV. The performance of DNMPC is compared, in terms of tracking and execution time, to that of standard control configurations based on centralized MPC and PID control aiming to show the suitability of each configuration in terms of performance and the practicality of using a particular configuration in real-time applications. The results show the advantage of using DNMPC in terms of ease of tuning and computational cost over more centralized feedback control approaches.

## 1 INTRODUCTION

In the last decades, different control techniques have been used in the aerospace industry. These range from basic classical control to modern artificial intelligence techniques used in many other fields. Classical controllers such as Proportional–Integral–Derivative PID [1], one of the best-known controllers, are successful in stabilizing a wide range of systems [2]. Aerospace systems such as aircraft or spacecraft, are nonlinear systems that could benefit from being controlled by advanced control systems enforcing strict stability and reliability requirements.

MPC is an advanced proactive control approach in which the control law is internally determined according to the forecasted response of the dynamic model to be controlled [2]. An open loop optimal control problem (OLOCP) is solved within a specified duration called Prediction Horizon to produce an optimal manipulated input trajectory for the optimal anticipated state trajectory. Out of the determined manipulated input trajectory, only the first control input action is applied to the system, and the whole process is repeated at every time sample. Furthermore, one of the major features of MPC is that it can handle multiple outputs and multiple inputs, or what is called MIMO systems [3], which makes it perfect for multivariable feedback control since the 1990s [4].

The testbed for this research is a Quadrotor, which is an Unmanned Aerial Vehicle (UAV) that features exceptional manoeuvrability, hovering, vertical take-off and landing capabilities.

*Email address(es): bama4@le.ac.uk

MPC is extensively used in the literature as a flight controller for quadrotors. Identifying a suitable dynamic model for a quadrotor is crucial when it comes to MPC. The literature gives formulations of MPC for different model types, such as linear time-invariant, linear time-varying, piecewise affine, nonlinear, . . . etc. Since the mathematical model is the centrepiece of MPC, the model type is pivotal in how efficiently the OLOCP will be solved by the optimization solver. Complex dynamics representations may lead to a computational burden when solving the OLOCP.

Linear dynamics were considered in designing a linear model predictive control (LMPC) for the quadrotor in [5], resulting in a good tracking performance and disturbances rejection for different trajectories. Alexies et al. [6] used a switching type MPC in controlling the quadrotor but with a linearized piecewise affine model around multiple operating points to track a trajectory with the presence of disturbances. The authors decoupled the dynamics to form what they call it "dual control scheme." A separate MPC was used to control the translational motion using the translational augmented dynamics, and another MPC was used to control the attitude of the quadrotor. As a matter of fact, the attitude dynamics (angular motion) of the quadrotor is faster in nature compared to the translational dynamics [6]. Thus, separating the controller for the translational motion and attitude is possible when the dynamics are decoupled.

A centralized nonlinear MPC was investigated in [7] based on a reduced nonlinear dynamic model of a quadrotor. It was shown that the use of a reduced nonlinear model led to a simpler MPC solution, easier to implement producing similar performance to that of an MPC solution based on a full order model. Reducing the computational complexity has been addressed effectively in [8] using machine learning to learn the control law from well-designed MPC flight data. However, this approach replaces the MPC with a clone that is limited by the conditions of the dataset used in the training stage.

Also, many researchers used the nominal MPC in conjunction with another technique. The authors in [9] proposed a two-loop controller in which the outer loop is MPC based to control the translational motion and a faster inner loop based on PID to control the attitude of the quadrotor. Although they have proved in simulation tests the effectiveness of using MPC to respect the constraints of the actuators and tracking the trajectory accurately, the disturbance effect was not introduced in their proposal. Hence, robustness concerns could

arise, and the constraints may be violated if the controller is not robust enough when the quadrotor is subjected to a disturbance. Moreover, comparative studies were conducted to quantify the performance and the control effort of the linear MPC versus PID, PD, and Linear Quadratic Regulator (LQR) in the [10]. The authors claim that MPC offers better tracking performance compared to the other techniques. Their study adopted the centralized LMPC and tested on a smooth trajectory without any sharp manoeuvres. The results are convincing but when it comes to fast and very sharp manoeuvres, the results need to be verified.

The configuration of MPC formulation and how it handles the system dynamics could be varied according to the system complexity and the performance requirements [11]. In general, there are three different MPC configurations or architectures to control the system, namely; Centralized, Decentralized, and Distributed MPC. In centralized MPC, large-scale systems are controlled using one single MPC that has one optimal problem. Although this architecture is the greatest for performance, it is considered impracticable for large systems due to its computational requirements [12]. The decentralized MPC architecture features a separate MPC for each subsystem of the dynamics. Usually, the subsystem dynamics are decoupled from each other, the mutual interaction is ignored, as well as, each MPC has its optimal problem [11, 12]. On the other hand, the distributed model predictive control (DMPC) structure shown in Figure 1, is introduced when the decentralized configuration is associated with a communication ability between the individual MPC agents to facilitate exchanging information between them [12].

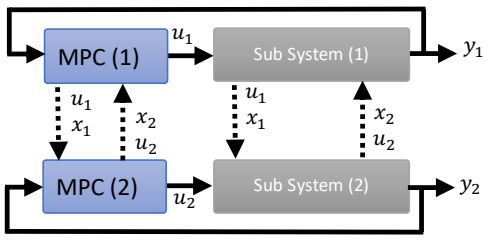


Figure 1: Typical configuration of distributed MPC.

The theory of DMPC and its stability has been extensively studied in the literature [13, 14]. Although the purpose of DMPC is to control large-scale with multiple objectives system, it has been applied successfully in controlling AC/AC converters in power applications to reduce the computational burden when centralized MPC is used [15]. The previous statement brings us to the contributions of the current research. To the best of our knowledge, no study was conducted to analyze the performance of different MPC configurations mainly when the nonlinear model is used. Furthermore, adopting DNMPC for quadrotors has gained less interest in the UAV control literature. Therefore, the contributions of this research are twofold:

- Introducing Distributed Nonlinear MPC for controlling the quadrotor by decomposing the centralized MPC into multiple agents whilst reducing the computation complexity of the overall controller.
- Evaluate various MPC configurations in terms of performance and computational complexity.

The remainder of this paper is organized as follows: Section 2 focuses on the derivation of the quadrotor UAV model and its dynamics as a part of the control system. Section 3 highlights the problem formulation and theory behind the Distributed Nonlinear MPC. Section 4 gives the design of the DNMP for the quadrotor, Section 5 demonstrates the different tests applied on the proposed DNMPC and the other configurations and the results with a brief discussion while Section 6 concludes the findings of the paper.

### *1.1 Notation*

In this paper, vectors are represented in bold lowercase letters (e.g., $\boldsymbol{x}$), and the uppercase bolded letters are used for matrices (e.g., $\mathbf{A}$). Other symbols represent scalars regardless of the letter case. The Euclidean $\mathbf{Q}$-weighted norm is denoted by $\|\boldsymbol{x}\|^2_{\mathbf{Q}} = \boldsymbol{x}^T\mathbf{Q}\boldsymbol{x}$ where $\mathbf{Q}$ is a positive definite real symmetric matrix.

## 2 QUADROTOR DYNAMIC MODEL

To model the dynamics of the quadrotor and describe its states, it is crucial to define the coordinate systems. The inertial frame is fixed on the earth, entitled "Earth Frame". This frame denoted $\mathcal{F}_E$ with an x-axis directed to the north, a y-axis directed to the east, and a z-axis directed down toward the earth, while $\mathcal{F}_B$ represents the quadrotor as a rigid body frame as shown in Figure 2. The attitude of the quadrotor is defined by the Euler angles, roll angle $\phi$, pitch angle $\theta$, and yaw angle $\psi$. Due to the time independency of the quadrotor's body inertia and its symmetric structure, the motion equations can be achieved with respect to its body frame $\mathcal{F}_B$ [16]. The

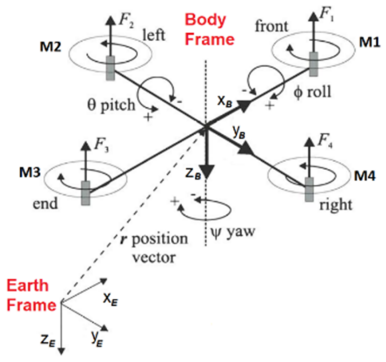


Figure 2: Frames coordinate systems.

absolute position of the quadrotor is $\boldsymbol{\xi} = [x, y, z]^T$ and the attitude denoted by $\boldsymbol{\eta} = [\phi, \theta, \psi]^T$. The model in this paper

follows the derivations made in [17] and [18]. The thrust $T_i$ is aligned with the rotor axis $i$ which rotates at angular velocity $\omega_i$ and is given by:

$$T_i = b\omega_i^2, \quad i \in 1,2,3,4 \tag{1}$$

in which $b$ is the lift constant equal to $C_T \rho A r^2$, where $\rho$ is the density of the air that surrounds the propeller of surface area $A$ and radius of $r$ and where $C_T$ is the thrust factor. Hence, the thrust along the z-direction of the quadrotor is given by:

$$T = b\sum_{i=0}^{4} \omega_i^2 = b\left(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2\right) \tag{2}$$

On the other hand, $M_\phi$ and $M_\theta$ are the pitching and rolling moments due to the thrust differences between the rotors. The net drag force due to the rotors is causing the heading moment $M_\psi$. The moments are given by:

$$M_\phi = l\left(T_4 - T_2\right) \tag{3a}$$
$$M_\theta = l\left(T_3 - T_1\right) \tag{3b}$$
$$M_\psi = d\left(-\omega_1^2 + \omega_2^2 - \omega_3^2 + \omega_4^2\right) \tag{3c}$$

where $l$ is the distance from the quadrotor's COM to the rotor axis, $d$ is the drag constant given by $C_P \rho A_r^3$, and $C_P$ is the torque coefficient of the motor. If quadrotor translational and angular velocity vectors are $\dot{\boldsymbol{\xi}} = [U, V, W]^T$ and $\dot{\boldsymbol{\eta}} = [P, S, R]^T$ respectively, the translational and angular motion equations are given by:

$$\begin{bmatrix} \dot{U} \\ \dot{V} \\ \dot{W} \\ \dot{P} \\ \dot{Q} \\ \dot{R} \end{bmatrix} = \begin{bmatrix} \left(\sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi\right)\frac{T}{m} - \frac{A_x}{m}U \\ \left(-\sin\phi\cos\psi + \cos\phi\sin\theta\sin\psi\right)\frac{T}{m} - \frac{A_y}{m}V \\ -g + \left(\cos\phi\cos\theta\right)\frac{T}{m} - \frac{A_z}{m}W \\ \frac{1}{I_{xx}}\left\{\left(I_{yy} - I_{zz}\right)\ Q\ R - J_r\ \omega_T\ Q + M_\phi - A_r\ P\right\} \\ \frac{1}{I_{yy}}\left\{\left(I_{zz} - I_{xx}\right)\ P\ R - J_r\ \omega_T\ p + M_\theta - A_r\ Q\right\} \\ \frac{1}{I_{zz}}\left\{\left(I_{xx} - I_{yy}\right)\ P\ Q + M_\psi - A_r\ R\right\} \end{bmatrix} \tag{4}$$

where $\omega_T = -\omega_1 + \omega_2 - \omega_3 + \omega_4$, $J_r$ is the motor's rotor inertia, $A_r$ is the rotational aerodynamic drag coefficient and $A_x$, $A_y$ and $A_z$ are the linear aerodynamic drag coefficients in the x, y, and z directions respectively. Also, $I_{xx}$, $I_{yy}$, and $I_{zz}$ denote the time-invariant inertia of the quadrotor about the body axes in $\mathcal{F}_B$. As discussed before, the only variables that can be controlled are the angular velocity of the motors. Before designing a control system, it is necessary to map the control signals to state equations. Thrust and moments are related to the motors angular velocities thanks to an allocation matrix, as follows:

$$\begin{bmatrix} T \\ M_\phi \\ M_\theta \\ M_\psi \end{bmatrix} = \begin{bmatrix} b & b & b & b \\ 0 & -lb & 0 & lb \\ -lb & 0 & lb & 0 \\ -d & d & -d & d \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} \tag{5}$$

The parameters of the quadrotor UAV used in this paper are adopted from [19] and given in Table 1.

| Symbol | Description | Value (Units) |
|---|---|---|
| $g$ | Acceleration due to gravity | 9.806 (m/s2) |
| $A_r$ | Rotational aerodynamic drag coefficient | 10e-15 (Nm.s/rad) |
| $m$ | Total mass of quadcopter UAV | 0.65 (kgs) |
| $l$ | distance from the quadrotor's COM motor | 0.232 (m) |
| $A_x, A_y, A_z$ | Linear aerodynamic drag coefficient | 10e-15 (N.s/m) |
| $J_r$ | Rotor inertia | 4e-4 (kg.m2) |
| $I_{xx}$ | Moment of Inertia along x-axis | 7.5e-3 (Nm.s2/rad) |
| $I_{yy}$ | Moment of Inertia along y-axis | 7.5e-3 (Nm.s2/rad) |
| $I_{zz}$ | Moment of Inertia along z-axis | 1.3e-2 (Nm.s2/rad) |
| $\rho$ | Air Density | 1.293 (kg/m3) |
| $r$ | Propellers Radius | 0.15 (m) |
| $C_T$ | Thrust Coefficient | 0.055 |
| $C_P$ | Torque Coefficient | 0.024 |

Table 1: Quadrotor UAV parameters.

## 3 PROBLEM FORMULATION

Distributed Nonlinear MPC (DNMPC) is composed of $N_a$ agents, where, each agent can be viewed as a local model predictive controller with a separate open-loop optimal control problem (OLOCP). Considering the following nonlinear discrete-time local system model for agent $i$ [14],

$$\boldsymbol{x}_{k+1}^i = f_i\left(\boldsymbol{x}_k^i, \boldsymbol{u}_k^i\right), \quad i \in \{1, \ldots, N_a\} \tag{6}$$

Where, $\boldsymbol{x}_k^i \in \mathbb{R}^{n_i}$ and $\boldsymbol{u}_k^i \in \mathbb{R}^{m_i}$ are the local state and control vectors of agent $i$ respectively, $n$, and $m$ representing the local state, and input vector dimensions, respectively. $\boldsymbol{x}_k^i$ is the successor or the next state and $f_i : \mathbb{R}^{n_i} \times \mathbb{R}^{m_i} \to \mathbb{R}^{n_i}$ is naturally defined by the local system differential equation that equals zero at zero initial conditions $f_i\left(0, 0\right)$. The sets of constraints that represent the safety limits and physical requirements for the local state and control inputs are denoted $\mathbb{X}i \in \mathbb{R}^{n_i}$ and $\mathbb{U}^i \in \mathbb{R}^{m_i}$ respectively. They also represent the feasible state and input of the agent $i$. Therefore

$$\boldsymbol{x}_k^i \in \mathbb{X}^i, \quad \boldsymbol{u}_k^i \in \mathbb{U}^i, \quad k \geq 0, \quad i \in \{1, \ldots, N_a\} \tag{7}$$

At time step $k$, the current measure state $\boldsymbol{x}_k^i$ is used to solve an OLOCP based on the local system model (6). Whatever the formulation of OLOCP, its solution is an optimal state trajectory that stabilizes the system in addition to an optimal control input trajectory that satisfies the anticipated state trajectory. Specifying the prediction horizon is vital to formulate the OLOCP problem and hence selecting the appropriate optimization solver. According to [12], the OLOCP problem is either infinite or finite. In this paper, the upper approximate finite OLOCP version adopted with the Terminal Cost function $V_{f_i}$, and Terminal Constraint $\mathbb{X}_{f_i} \in \mathbb{R}^{n_i}$ such that:

$$\boldsymbol{x}_N^i \in \mathbb{X}_{f_i}, \quad i \in \{1, \ldots, N_a\} \tag{8}$$

The control objective is to use the receding horizon control approach to cooperatively stabilize the $N_a$ agents and

converging the state of each agent to the equilibrium point $\left(\boldsymbol{x}_e^i, \boldsymbol{u}_e^i\right)$ by minimizing the following OLOCP

$$\min_{\boldsymbol{u}} \quad J_k\left(\boldsymbol{x}_k, \boldsymbol{u}_k\right) = \sum_{k=0}^{N-1} \ell\left(\boldsymbol{x}_k, \boldsymbol{u}_k\right) + V_f\left(\boldsymbol{x}_N\right) \tag{9}$$

Where $N$ is the prediction horizon, and

$$\boldsymbol{x}_k = \left[\left(x^1\right)^T, \ldots, \left(x^{N_a}\right)^T\right]^T, \quad \boldsymbol{x}_k \in \mathbb{R}^n, \quad n = \sum_i n_i \tag{10a}$$

$$\boldsymbol{u}_k = \left[\left(u^1\right)^T, \ldots, \left(u^{N_a}\right)^T\right]^T, \quad \boldsymbol{u}_k \in \mathbb{R}^m, \quad m = \sum_i m_i \tag{10b}$$

Then, the overall dynamics can be expressed as

$$\boldsymbol{x}_{k+1} = f\left(\boldsymbol{x}_k, \boldsymbol{u}_k\right) \tag{11}$$

where $f = [f_1, \ldots, f_{N_a}]^T$, $f : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^n$ and the global equilibrium point for all agents is $(x_e, u_e)$. The OLOCP in (9) can be used in centralized MPC and is subjected to $\mathbb{X} = \mathbb{X}^1 \times \ldots \times \mathbb{X}^{N_a}$, $\mathbb{U} = \mathbb{U}^1 \times \ldots \times \mathbb{U}^{N_a}$, and $\mathbb{X}_f = \mathbb{X}_{f_1} \times \ldots \times \mathbb{X}_{f_{N_a}}$. Although this formulation slightly increases the computation cost when it comes to solving the optimization problem, the terminal cost and terminal constraints assure the stability and feasibility of the controlled system [12, 20]. There is no specific rule to determine the terminal cost, however, it captures the cost beyond $N$ up to $\infty$ to guarantee the full covering of the horizon [21].

The objective of each agent's optimization problem is to minimize the tracking error between the local state $\boldsymbol{x}_k^i$ and its reference trajectory $\bar{\boldsymbol{x}}^i$. Furthermore, to decompose the centralized MPC into a distributed MPC, the following two assumptions are made:

***Assumption 1:*** The models for all agents presented in (6) are decoupled and the communication between agents is sufficient and available as needed by any agent.

***Assumption 2:*** The condition $f_i\left(0, 0\right) = 0$ is not restrictive, since $f_i\left(x_e^i,\ u_e^i\right)$ is not zero, thus, the equilibrium point could be any aribtarary reference $\in \mathbb{X}_{f_i}$.

Accordingly, in a distributed style, problem (9) is divided into multiple OLOCPs given by

$$\begin{aligned} \min_{\boldsymbol{u}^i} \quad & J_k^i\left(\boldsymbol{u}_k^i, \boldsymbol{u}_k^i, \bar{\boldsymbol{x}}_k^i\right) = \sum_{k=0}^{N-1} \ell_i\left(\boldsymbol{x}_k^i, \boldsymbol{u}_k^i, \bar{\boldsymbol{x}}_k^i\right) + V_{f_i}\left(\boldsymbol{x}_N^i\right) \\ \text{s.t.} \quad & (6), (7), (8), \quad \boldsymbol{x}_0^i = \boldsymbol{x}_k^i \end{aligned} \tag{12}$$

Where the stage cost function $\ell_i(\boldsymbol{x}_k^i, \boldsymbol{u}_k^i)$ and terminal cost function $V_{f_i}(\boldsymbol{x}_N^i)$ are defined as:

$$\ell_i(\boldsymbol{x}_k^i, \boldsymbol{u}_k^i,\ \bar{\boldsymbol{x}}_k^i) = \|\boldsymbol{x}_k^i - \bar{\boldsymbol{x}}_k^i\|_{\mathbf{Q_i}}^2 + \|\boldsymbol{u}_k^i\|_{\mathbf{R_i}}^2 \tag{13a}$$

$$V_{f_i}(\boldsymbol{x}_N^i) = \|\boldsymbol{x}_k^i - \bar{\boldsymbol{x}}_N^i\|_{\mathbf{P_i}}^2 \tag{13b}$$

where $\mathbf{Q}_i$, $\mathbf{R}_i$, and $\mathbf{P}_i$ are positive definite weighting matrices for penalising the states, manipulated inputs, and terminal state, respectively for agent $i$. For each agent $i$, the control law at the end of each prediction iteration is

$$\boldsymbol{u}_k^i = \boldsymbol{u}_0^{i\,*}\left(x_k^i\right) \tag{14}$$

The superscript * indicates the optimal value and $u_0^i$ is the first control input action in the optimal control input trajectory obtained from the OLOCP solution that is applied to the system.

## 4 DNMPC FOR QUADROTOR UAV

When designing a controller for quadrotors, different effects could be considered in the dynamics model, the reader is referred to [22] for a list of these effects. Including all effects with a full dynamics representation for the plant will intuitively lead to an increase in computational complexity. Therefore, a simplified version of the dynamics is used such as [7] to achieve faster solution during each predication iteration. However, contrary to [9], the linear and rotational aerodynamic drag forces are considered in this paper. The resultant dynamics are driven from (4), decoupled and reduced based on the following assumptions:

***Assumption 3:*** The gyroscopic effect of the motor is ignored.

***Assumption 4:*** When the quadrotor is moving in the x-axis direction (pitching), the rolling angle is assumed zero.

***Assumption 5:*** When the quadrotor is moving in a y-axis direction or rolling, the pitching angle is assumed zero.

***Assumption 6:*** When the quadrotor moves vertically, the pitching and rolling angles are assumed zero.

***Assumption 7:*** The heading of the quadrotor is assumed not controlled and the related torque is zero.

If $T_s$ is the sampling time, the subscript $k$ denotes the value of the variables at time $kT_s$, and by using the forward Euler discretization, the simplified decoupled dynamics are given as follows

$$\boldsymbol{x}_{k+1} = \begin{bmatrix} x_k \\ U_k \\ y_k \\ V_k \\ z_k \\ W_k \\ \phi_k \\ P_k \\ \theta_k \\ Q_k \end{bmatrix} + T_s \begin{bmatrix} U_k \\ m^{-1}(\sin\theta_k T_k - A_x U_k) \\ V_k \\ m^{-1}(-\sin\phi_k T_k - A_y V_k) \\ W_k \\ -g + m^{-1}(T_k - A_z W_k) \\ P_k \\ {I_{xx}}^{-1} M_{\phi_k} - A_r P_k \\ Q_k \\ {I_{yy}}^{-1} M_{\theta k} - A_r Q_k \end{bmatrix} \tag{15}$$

Although the nonlinearity has been significantly reduced and weakened, the above dynamics in [14] without considering the linear and rotational aerodynamic drag forces, are validated in [7] when the pitching and rolling angles are limited to 0.2618 rad. Considering the linear and rotational aerodynamic drag forces, it is expected to perform better and acts more accurately compared to what was adopted in [7] with a little additional computation cost.

### 4.1 DNMPC Controller

Inspired by [15], with assumptions (1-7), we considered the Quadrotor UAV dynamics as a large-scale system in this paper, thanks to dynamics decoupling, (15) represented as five interconnected subsystems where each subsystem is controlled by its own MPC agent shown in Figure 3 to form the DNMPC with $N_a = 5$. Z-MPC, X-MPC, and Y-MPC form the first MPC agents' group for controlling the translational motion, z-axis, x-axis, and y-axis respectively. On the other hand, Rolling MPC, and Pitching MPC form the second MPC agents' group for controlling the angular position of $\phi$ and $\theta$ respectively.

The reference set-point is provided for each corresponding MPC agent in the first group. The role of the Z-MPC is to determine the optimal thrust force $T$, then feed it to the allocation matrix, X-MPC and Y-MPC as a cooperative input within the first group. In the proposed design, X-MPC and Y-MPC were utilized to generate the attitude trajectory for the second MPC agents' group. Thus, the two manipulated inputs $\phi$, and $\theta$, were the optimal inputs resultant from the Y-MPC, and X-MPC respectively and are provided to the second group as reference trajectory. Finally, the second group of agents, namely, Rolling MPC, and Pitching MPC, are designed to determine the optimal rotational torques, $M_{\phi_k}$, and $M_{\theta k}$ respectively and feeding them to the allocation matrix. The local subsystem's states and manipulated inputs for each MPC agent are summarized in the equations set (16). It is noteworthy to mention that according to assumption 7, no controller was used for controlling the heading of the Quadrotor UAV.

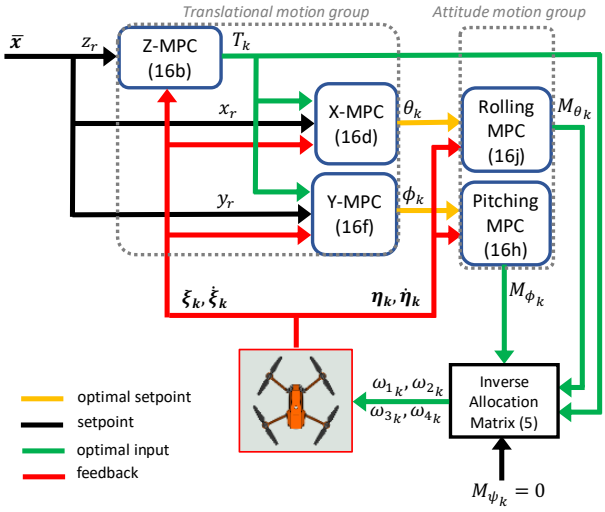


Figure 3: Block diagram for Distributed nonlinear MPC for Quadrotor UAV.

$$\boldsymbol{x}_k^1 = \begin{bmatrix} z_k \\ W_k \end{bmatrix}, \quad \boldsymbol{u}_k^1 = T_k \tag{16a}$$

$$\boldsymbol{x}_{k+1}^1 = \boldsymbol{x}_k^1 + T_s \begin{bmatrix} W_k \\ -g + m^{-1}(T_k - A_z W_k) \end{bmatrix} \tag{16b}$$

$$\boldsymbol{x}_k^2 = \begin{bmatrix} x_k \\ U_k \end{bmatrix}, \quad \boldsymbol{u}_k^2 = \theta_k \tag{16c}$$

$$\boldsymbol{x}_{k+1}^1 = \boldsymbol{x}_k^1 + T_s \begin{bmatrix} U_k \\ m^{-1}(\sin\theta_k T_k - A_x U_k) \end{bmatrix} \tag{16d}$$

$$\boldsymbol{x}_k^3 = \begin{bmatrix} y_k \\ V_k \end{bmatrix}, \quad \boldsymbol{u}_k^3 = \phi_k \tag{16e}$$

$$\boldsymbol{x}_{k+1}^3 = \boldsymbol{x}_k^3 + T_s \begin{bmatrix} V_k \\ m^{-1}(-\sin\phi_k T_k - A_y V_k) \end{bmatrix} \tag{16f}$$

$$\boldsymbol{x}_k^4 = \begin{bmatrix} \phi_k \\ P_k \end{bmatrix}, \quad \boldsymbol{u}_k^4 = M_{\phi_k} \tag{16g}$$

$$\boldsymbol{x}_{k+1}^4 = \boldsymbol{x}_k^4 + T_s \begin{bmatrix} P_k \\ {I_{xx}}^{-1} M_{\phi_k} - A_r P_k \end{bmatrix} \tag{16h}$$

$$\boldsymbol{x}_k^5 = \begin{bmatrix} \theta_k \\ Q_k \end{bmatrix}, \quad \boldsymbol{u}_k^5 = M_{\theta k} \tag{16i}$$

$$\boldsymbol{x}_{k+1}^5 = \boldsymbol{x}_k^5 + T_s \begin{bmatrix} Q_k \\ {I_{yy}}^{-1} M_{\theta k} - A_r Q_k \end{bmatrix} \tag{16j}$$

### 4.2 Handling Constraint

In order to unify the framework of the analysis and introduce a fair comparison between the various configurations, unified constraints have been applied. The constraints set for the forces (5) are calculated based on the technical parameters of the brushless motors and their propellers. Assuming that all propulsion brushless motors are identical, the minimum lifting force is equal to mg, whereas the maximum lift force is given (2). Also, by recalling (3), therefore, the maximum and the minimum value of the rotational moment can be calculated by substituting the parameters of Table 2 in (3).

## 5 Simulation Results & Discussion

In this section, the simulating environment under which the proposed DNMPC controller and the other three different configurations are tested is explained. The other configurations are the Centralized Nonlinear MPC with full dynamics (CNMPC-F) built with dynamics (4), Centralized Nonlinear MPC with reduced decoupled dynamics (CNMPC-R) built with dynamics (15), and Nonlinear MPC with PID (MPC-PID) in [9].

### 5.1 Simulation Setup

The modelling of the quadrotor UAV has been done in MATLAB/Simulink while the model predictive controllers in all configurations have been carried out in MATLAB in MATLAB/Simulink by integrating CasADi, an open-source optimization toolkit with MATLAB interface [23]. The nonlinear OLOCP has been solved using the Interior Point Optimizer (Ipopt) [24], which is natively available in CasADi. The weighting matrices for each configuration, prediction hori-

zon, control horizon, and sampling time are summarized in Table 2.

| | |
|---|---|
| CNMPC Sampling Time $T_s$ | 0.05 s |
| DNMPC Translation Group $T_s$ | 0.1 s |
| DNMPC Attitude Group $T_s$ | 0.05 s |
| MPC in MPC-PID $T_s$ | 0.05 s |
| PID Sampling time $T_s$ | 0.01 s |
| Prediction horizon | 40 |
| Weighting matrices for CNMPC | $\mathbf{Q} = \mathbf{P} = diag(15, 15, 150)$<br>$\mathbf{R} = diag(0.5, 8, 8, 0.01)$ |
| Z-MPC weight in DNMPC | $\mathbf{Q} = \mathbf{P} = 10, \mathbf{R} = 0.09$ |
| X-MPC weight in DNMPC<br>Y-MPC weight in DNMPC | $\mathbf{Q} = \mathbf{P} = 35, \mathbf{R} = 50$ |
| Rolling MPC weight in DNMPC<br>Pitching MPC weight in DNMPC | $\mathbf{Q} = \mathbf{P} = 200, \mathbf{R} = 20$ |
| Weighting matrices for CNMPC | $\mathbf{Q} = \mathbf{P} = diag(5, 5, 80)$<br>$\mathbf{R} = diag(0.1, 7, 7)$ |
| Z-MPC weight in DNMPC | $\mathbf{Q} = \mathbf{P} = 10, \mathbf{R} = 0.09$ |
| PID controller parameters | $KP_\theta = KP_\phi = 25$<br>$KI_\theta = KI_\phi = 0.1$<br>$KD_\theta = KD_\phi = 15$ |

Table 2: Controllers parameters for all configurations.

### 5.2 *Step Response Results*

All configurations were tested by applying a step input of 5m on the x-axis while the quadrotor is at a hovering point (0, 0, 4m). The step response characteristics for each configuration have been summarized in Table 3.

| | CNMPC-F | CNMPC-R | DNMPC | MPC-PID |
|---|---|---|---|---|
| **Rise Time** | 1.4207 | 1.567 | 2.1466 | 2.3924 |
| **Transient Time** | 2.5409 | 2.7869 | 3.4209 | 3.509 |
| **Settling Time** | 2.5409 | 2.7869 | 3.4209 | 3.509 |
| **Settling Min** | 4.5022 | 4.5288 | 4.5387 | 4.5608 |
| **Settling Max** | 5.0001 | 5.0001 | 5.0000 | 5.0159 |
| **Overshoot** | 0.0017 | 0.0015 | 0 | 0.3157 |
| **Peak** | 5.0001 | 5.0001 | 5.0000 | 5.0159 |
| **Peak Time** | 5.65 | 5.40 | 7.00 | 4.15 |

Table 3: Step Response characteristics for each configuration.

### 5.3 *Tracking Results*

Two trajectories were used to test the tracking performance of each configuration. Figures (4) and (5) illustrate the tracking response for the infinity shape and the multiple short rectangles' trajectory respectively. The corresponding motors' angular velocities for each tracking response are illustrated in Figures (6) and (7). It is seen how the DNMPC configuration provides fewer changes for all trajectories, especially for the sharp manoeuvres. Moreover, it is obvious that the MPC-PID configuration has the worst variation, and this is expected as the only optimal input is the left force, where the other forces are generated from the PID. To investigate the practicability and performance of each configuration, the solution time for the OLOCP in each prediction cycle. In addition, the root mean square error RMSE has been determined for all trajectories. Table 5.3 shows the solution time and RMSE of each configuration for the tested trajectories.

| **Infinity Shape Trajectory (40 s)** | | | | |
|---|---|---|---|---|
| **RMSE x-axis (m)** | 0.3556 | 0.29982 | 0.29688 | 0.22453 |
| **RMSE y-axis (m)** | 0.35302 | 0.30021 | 0.2982 | 0.23771 |
| **RMSE z-axis (m)** | 0.40636 | 0.40672 | 0.53857 | 0.29981 |
| **Total Solving time (s)** | 68.3392 | 14.9918 | 25.0127 | 10.8356 |
| **Average Solving time (ms)** | 85.42 | 18.74 | 31.22 | 13.54 |
| **Multiple Short Rectangles' Trajectory (26 s)** | | | | |
| **RMSE x-axis (m)** | 0.96495 | 1.0071 | 1.0551 | 1.0633 |
| **RMSE y-axis (m)** | 0.68085 | 0.67747 | 0.71234 | 0.712 |
| **RMSE z-axis (m)** | 0.42955 | 0.42866 | 0.52551 | 0.38389 |
| **Total Solving time (s)** | 36.0099 | 14.7204 | 17.6784 | 9.0684 |
| **Average Solving time (ms)** | 69.25 | 28.30 | 33.93 | 17.43 |

Table 4: Step Response characteristics for each configuration.

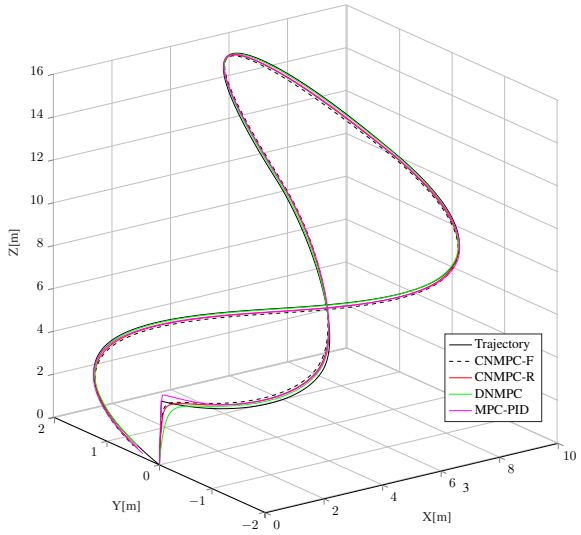


Figure 4: Tracking response for an infinity shape trajectory.

### 5.4 *Discussion*

It is seen from Table 5.3 that decomposing the dynamics and reducing its complexity in the DNMPC led to a significant computation time reduction. That is expected as computational complexity and thus the computation cost can be reduced when the problem dimension is scaled down, for instance, the factorization in Newton-type solver requires $2n^3$ flops (floating point operations) for a state equation of dimension $n$ [25]. Therefore, when scaling down dynamics (4) used in the CNMPC-F to dimension 2 in the individual state equations of (16), the number of flops required for factorization is then reduced to 1/216 (where $n$ divided by 6). Concerning tuning, the different configurations have been tuned by a simple strategy where all weighting parameters are initiated with 1, and then manipulated in steps until an acceptable step response performance is achieved. As a matter of fact, tuning the centralized MPC is challenging. However, the DNMPC configuration was the most flexible one for tuning. This is because each agent in the proposed DNMPC is independent of the other in terms of its cost function and penalty weight.

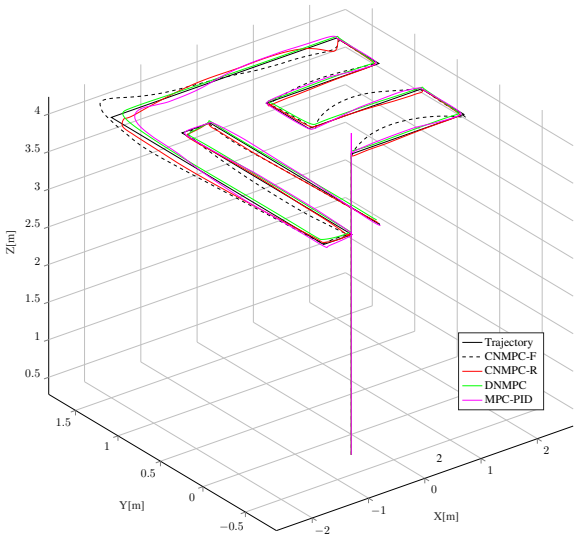


Figure 5: Tracking response for multiple short rectangles' trajectory.

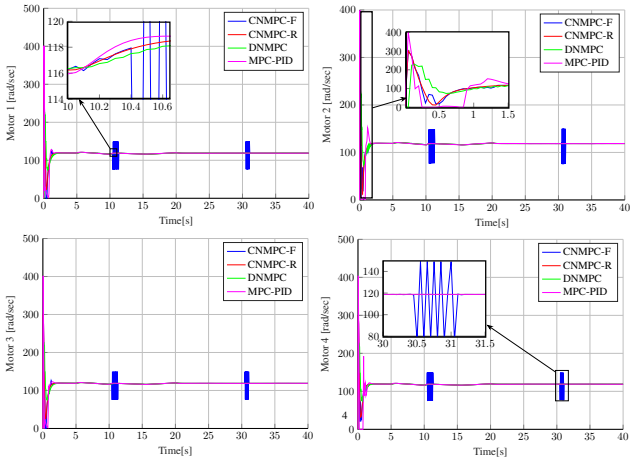


Figure 6: Motors' angular velocities for an infinity shape trajectory.

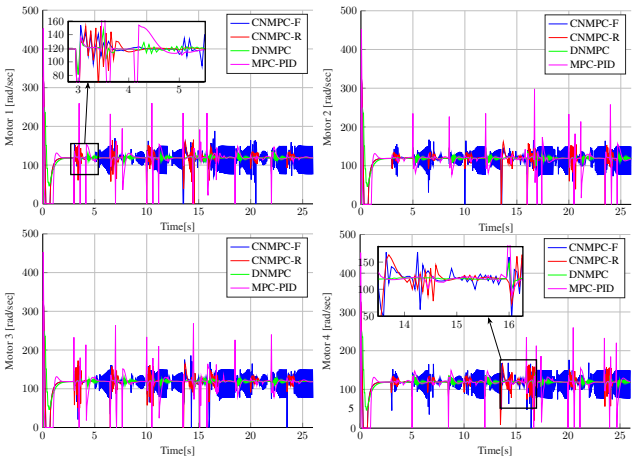


Figure 7: Motors' angular velocities for multiple short rectangles' trajectory.

The results reveal that the MPC-PID configuration is faster than the other configurations since the time needed for solving the OLOCP in its MPC is 84% better than the nominal CNMPC-F configuration. In terms of practicability and real-time application, the CNMPC-R, DNMPC, and MPC-PID configurations are more suitable than using the CNMPC-F. The step response characteristics in Table 3 provide a plain understanding of the response for each configuration and it is discernible that the DNMPC is slower compared to the other configurations but steadier in reaching the set-point.

## 6 CONCLUSION

This paper explores the possible configuration for designing MPC based flight controller for a quadrotor UAV. The dynamics of the quadrotor UAV were modelled using its equations of motion and used as an oriented model for the MPC. Four configurations were studied based on two architectures, centralized and distributed architecture. A nonlinear flight controller based on the distributed MPC was proposed, in which five separate MPC agents were used to stabilise the quadrotor. All configurations were tested with a step input in addition to tracking two different trajectories. The results disclose that the use of simplified decoupled dynamics reduces the computational complexity of the overall control law. The proposed DNMPC unfolds promising performance, especially for the sharp manoeuvres trajectories. Furthermore, the DMPC is more flexible and easier to tune than the conventional MPC. Further investigations could be conducted to evaluate the practicability and the suitability of each configuration, where real-time optimization is limited by the capabilities of the on-board microcontroller.

## ACKNOWLEDGEMENTS

This research is funded by the Higher Committee for Education Development in Iraq (HCED) and supported by Sulaimani Polytechnic University, Sulaymaniyah, Iraq.

## REFERENCES

[1] Ang Kiam Heong, G. Chong, and Li Yun. Pid control system analysis, design, and technology. *IEEE Transactions on Control Systems Technology*, 13(4):559–576, 2005.

[2] Max Schwenzer, Muzaffer Ay, Thomas Bergs, and Dirk Abel. Review on model predictive control: an engineering perspective. *The International Journal of Advanced Manufacturing Technology*, 117(5-6):1327–1349, 2021.

[3] C. H. F. Silva, H. M. Henrique, and L. C. Oliveira-Lopes. Experimental application of predictive controllers. *Journal of Control Science and Engineering*, 2012:1–18, 2012.

[4] Manfred Morari. Model predictive control: Multivariable control technique of choice in the 1990s? *Oxford science publications*, 1993.

[5] M. Islam, M. Okasha, and M. M. Idres. Dynamics and control of quadcopter using linear model predictive control approach. *IOP Conference Series: Materials Science and Engineering*, 270, 2017.

[6] K. Alexis, G. Nikolakopoulos, and A. Tzes. On trajectory tracking model predictive control of an unmanned quadrotor helicopter subject to aerodynamic disturbances. *Asian Journal of Control*, 16(1):209–224, 2014.

[7] M. Abdolhosseini, Y. M. Zhang, and C. A. Rabbath. An efficient model predictive control scheme for an unmanned quadrotor helicopter. *Journal of Intelligent & Robotic Systems*, 70(1-4):27–38, 2012.

[8] Guanrui Li, Alex Tunchez, and Giuseppe Loianno. Learning model predictive control for quadrotors. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 5872–5878, 2022.

[9] Huan Cheng and Yanhua Yang. Model predictive control and pid for path following of an unmanned quadrotor helicopter. In *2017 12th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, pages 768–773, 2017.

[10] Mohamed Okasha, Jordan Kralev, and Maidul Islam. Design and experimental comparison of pid, lqr and mpc stabilizing controllers for parrot mambo minidrone. *Aerospace*, 9(6), 2022.

[11] Yang Shi and Kunwu Zhang. Advanced model predictive control framework for autonomous intelligent mechatronic systems: A tutorial overview and perspectives. *Annual Reviews in Control*, 52:170–196, 2021.

[12] Utku Eren, Anna Prach, Başaran Bahadır Koçer, Saša V. Raković, Erdal Kayacan, and Behçet Açıkmeşe. Model predictive control in aerospace systems: Current state and opportunities. *Journal of Guidance, Control, and Dynamics*, 40(7):1541–1566, 2017.

[13] William B. Dunbar. Distributed receding horizon control of dynamically coupled nonlinear systems. *IEEE Transactions on Automatic Control*, 52(7):1249–1263, 2007.

[14] Wei Shanbi, Chai Yi, and Li Penghua. Distributed model predictive control of the multi-agent systems with improving control performance. *Journal of Control Science and Engineering*, 2012:1–8, 2012.

[15] Luca Tarisciotti, Giovanni Lo Calzo, Alberto Gaeta, Pericle Zanchetta, Felipe Valencia, and Doris Saez. A distributed model predictive control strategy for back-to-back converters. *IEEE Transactions on Industrial Electronics*, 63(9):5867–5878, 2016.

[16] Zoran Benic, Petar Piljek, and Denis Kotarski. Mathematical modelling of unmanned aerial vehicles with four rotors. *Interdisciplinary Description of Complex Systems*, 14(1):88–100, 2016.

[17] Hossein Bolandi, Mohammad Rezaei, Reza Mohsenipour, Hossein Nemati, and S. M. Smailzadeh. Attitude control of a quadrotor with optimized pid controller. *Intelligent Control and Automation*, 04(03):335–342, 2013.

[18] Harikrishnan Suresh, Abid Sulficar, and Vijay Desai. Hovering control of a quadcopter using linear and nonlinear techniques. *International Journal of Mechatronics and Automation*, 6(2/3), 2018.

[19] Armando S. Sanca, Pablo J. Alsina, and Jés de Jesus F. Cerqueira. Dynamic modelling of a quadrotor aerial vehicle with nonlinear inputs. In *2008 IEEE Latin American Robotic Symposium*, pages 143–148, 2008.

[20] J.B. Rawlings, D.Q. Mayne, and M. Diehl. *Model Predictive Control: Theory, Computation, and Design*. Nob Hill Publishing, second edition, 2020.

[21] Tor Aksel N. Heirung, Joel A. Paulson, Jared O'Leary, and Ali Mesbah. Stochastic model predictive control — how does it work? *Computers & Chemical Engineering*, 114:158–170, 2018.

[22] S. Norouzi Ghazbi, Y. Aghli, M. Alimohammadi, and A. A. Akbari. Quadrotors unmanned aerial vehicles: A review. *International Journal on Smart Sensing and Intelligent Systems*, 9(1):309–333, 2016.

[23] Joel A. E. Andersson, Joris Gillis, Greg Horn, James B. Rawlings, and Moritz Diehl. Casadi: a software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1):1–36, 2019.

[24] Andreas Wächter and Lorenz T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, 2005.

[25] Rien Quirynen. *Numerical Simulation Methods for Embedded Optimization*. Dissertation, University of Freiburg, 2017.