MPC Controller Tuning for Least Energy Quadrotor Missions

An M. Tech Project Report Submitted
in Partial Fulfillment of the Requirements
for the Degree of

Master of Technology

by

Arya Mallick
(234102501)

under the guidance of

Dr. Chayan Bhawal

to the

DEPARTMENT OF ELECTRONICS AND ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI
GUWAHATI - 781039, ASSAM

JUNE 2025

Contents

CERTIFICATE

*This is to certify that the work contained in this thesis entitled "**MPC Controller Tuning for least energy quadrotor missions**" is a bonafide work of **Arya Mallick** (Roll No. 234102501), carried out in the Department of Electronics and Electrical Engineering, Indian Institute of Technology Guwahati under my supervision and that it has not been submitted elsewhere for a degree.*

Supervisor: Dr. Chayan Bhawal

Assistant Professor

Dept. of Electronics & Electrical Eng.

IIT Guwahati, Assam, India

Date: June, 2025

Acknowledgements

I would like to express my deepest and most heartfelt gratitude to my supervisor, Dr. Chayan Bhawal. His exceptional guidance, unwavering support, and insightful feedback were the cornerstone of this project. Prof. Bhawal's vast expertise, patient mentorship, and constant encouragement inspired me to overcome every challenge and pursue excellence at each stage of my work. His vision and commitment were instrumental in shaping this research and have left a lasting impact on my academic journey.

I am also sincerely grateful to Paraj Ganchaudhari, Research Scholar at the Department of Electronics and Electrical engineering, whose practical advice and technical expertise significantly enriched this work. His support, camaraderie, and readiness to help were invaluable throughout the process.

I owe a special debt of gratitude to my family, whose unconditional love, encouragement, and support have been my greatest motivation. Their faith in me sustained my efforts and perseverance.

I also appreciate the support and friendship of my classmates, as well as the encouragement and resources provided by the EEE department, all of which contributed meaningfully to the success of this project.

Thank you all for being part of this journey.

Arya Mallick

IIT GUWAHATI

June 2025

Abstract

Unmanned Aerial Vehicles (UAVs) are increasingly integral to a wide range of applications, making energy efficiency a key factor in ensuring sustained operational effectiveness. This work focuses on developing energy-optimized trajectory tracking for UAVs using Model Predictive Control (MPC). Minimum-energy reference trajectories are first generated by solving an Optimal Control Problem (OCP) using the GPOPS-II framework.

To track these reference trajectories, two controller architectures are implemented and evaluated through simulations:

1)A cascaded control structure combining a Linear Parameter Varying MPC (LPV-MPC) in the inner loop with a state feedback controller in the outer loop.

2)A single-layer Nonlinear MPC (NMPC) approach.

The primary objective is to identify an effective optimization formulation and tuning parameters for the NMPC that enable accurate tracking of the energy-optimal trajectories while minimizing control effort. A systematic exploration of different cost functions, weight configurations, and prediction horizons is conducted, with parameter tuning guided by genetic algorithm-based optimization. Through extensive performance comparisons, the study determines the most effective NMPC setup for the DJI Phantom 2 quadcopter, achieving a favorable balance between trajectory tracking accuracy and energy conservation

List of Figures

Chapter 1

Introduction

UAV technology has found application in various sectors of industries. So for wide-scale usage, energy efficiency and management in drones crucial.UAVs are classified into two main categories: fixed-wing and rotatory-wing UAVs, which offer significant advantages over fixed-wing UAVs since they can take off and land vertically, maintain their position at a fixed location in 3D space, have agility, and have high maneuverability.However unlike the other one, it cant glide in air and leads to failre without sufficient energy.
Hence, Energy-efficient paths are crucial for rotary wing drones as they extend flight duration, reduce battery consumption, and enable longer missions or greater payload capacity, which is essential for applications like delivery, surveillance, and search-and-rescue.

A controller ensures the drone accurately follows an energy-efficient path by regulating its motors and actuators to minimize energy usage while maintaining stability and trajectory accuracy. It optimally adjusts thrust, pitch, and yaw to reduce unnecessary maneuvers and ensure smooth transitions, aligning with the planned low-energy trajectory.

In this work we use Model Predictive Control (MPC) on a quadcopter model, which offers better predictive capabilities than conventional PID and handles sharp turns with reduced tracking error.We try to tune the MPC controller parameters for best tracking performance.

1.1 Literature Survey

We review the prior works done regarding MPC and energy optimal path planning and tracking. MPC has emerged as a promising control strategy for drones, due to its ability to handle multivariable systems and constraints effectively.

In [1], an MPC motion controller is developed based on a linearized model for waypoint tracking. Similarly, [2] employs MPC for flock control of quadrotors, incorporating collision avoidance strategies. In [3], a linear MPC framework is implemented for quadcopter navigation, enhancing trajectory precision through real-time control refinements. Building on this [4] utilized a nonlinear MPC approach using CASADI to account for the UAV's complex dynamics, allowing for precise control over 3D paths. The study presented in [5] explores an efficient nonlinear model predictive control (NMPC) approach for trajectory tracking of a quadrotor unmanned aerial vehicle (UAV), achieved by integrating the desired trajectory into a reference dynamical system.

With increase in flight durations, energy efficiency in UAV is becoming more vital.In this paper [6], the energy efficiency of a quadrotor's dynamic model is analyzed. The study proposes a method for designing the dynamic model to estimate energy consumption accurately.

This paper[7] proposed an energy-aware MPC scheme designed to optimize power usage while maintaining accurate trajectory tracking. For energy optimal path-planning , the mission is considered for a fixed final time [8], but the fixed final time may or may not be the optimal time for mission completion. So, the optimal time for mission completion need to be considered.

Chapter 2

Motivation and Problem Formulation

2.1 Motivation

While MPC had demonstrated significant effectiveness in various UAV applications, but achieving an optimal balance between energy efficiency to complete a mission while maintaining tracking accuracy remained a persistent challenge, particularly for nonlinear UAV systems.
In this work , we aim to address these gaps by developing an NMPC strategy specially tailored for tracking of least energy path for nonlinear UAV systems .

2.2 Problem Statement

We aim to track a minimum-energy reference trajectory $\mathbf{x}\text{ref}(t)$ for a quadrotor UAV while minimizing control effort and satisfying system constraints. The control inputs are total thrust u_1 and torques $\mathbf{u}_2 = [\tau\phi, \tau_\theta, \tau_\psi]^T$. Let $\mathbf{x}(t)$ be the state vector comprising position, velocity, orientation, and angular velocities.

At each time step t_k, the Nonlinear Model Predictive Control (NMPC) problem is formulated as:

$$
\begin{aligned}
\min_{\mathbf{u}(\cdot)} \quad & \int_{t_k}^{t_k+T_p} \|\mathbf{x}(t) - \mathbf{x}_{\text{ref}}(t)\|_Q^2 + \|\mathbf{u}(t)\|_R^2 \, dt \\
\text{subject to} \quad & \dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)) \\
& \mathbf{x}(t_k) = \mathbf{x}_k \\
& \mathbf{x}\text{min} \leq \mathbf{x}(t) \leq \mathbf{x}\text{max} \\
& \mathbf{u}\text{min} \leq \mathbf{u}(t) \leq \mathbf{u}\text{max} \\
& \dot{\mathbf{u}}\text{min} \leq \dot{\mathbf{u}}(t) \leq \dot{\mathbf{u}}\text{max}
\end{aligned}
$$

Where:

- T_p is the prediction horizon.
- $\mathbf{u}(t) = [u_1, \tau_\phi, \tau_\theta, \tau_\psi]^T$
- Q, R are positive semi-definite weighting matrices for state error and control effort.
- $f(\cdot)$ represents the nonlinear dynamics of the UAV.

The objective is to find an optimal control sequence $\mathbf{u}^*(t)$ that minimizes the tracking error and control effort while satisfying all physical and operational constraints. The **system dynamics used in E- frame used are as follows:**

$$
\begin{aligned}
\dot{x}_1 &= x_2, \\
\dot{x}_2 &= \frac{k_b}{m}\left(\sin x_7 \sin x_{11} + \cos x_7 \cos x_{11} \sin x_9\right) \sum_{k=13}^{16} x_k^2, \\
\dot{x}_3 &= x_4, \\
\dot{x}_4 &= \frac{k_b}{m}\left(\cos x_7 \sin x_9 \sin x_{11} - \cos x_{11} \sin x_7\right) \sum_{k=13}^{16} x_k^2, \\
\dot{x}_5 &= x_6, \\
\dot{x}_6 &= \frac{k_b}{m}\left(\cos x_9 \cos x_7\right) \sum_{k=13}^{16} x_k^2 - g, \\
\dot{x}_7 &= x_8, \\
\dot{x}_8 &= \left(\frac{I_y - I_z}{I_x}\right) x_{10} x_{12} + \frac{\ell k_b}{I_x}\left(x_{14}^2 - x_{16}^2\right) \\
&\quad - \frac{J}{I_x} x_{10}\left(x_{13} - x_{14} + x_{15} - x_{16}\right), \\
\dot{x}_9 &= x_{10}, \\
\dot{x}_{10} &= \left(\frac{I_z - I_x}{I_y}\right) x_8 x_{12} + \frac{\ell k_b}{I_y}\left(x_{15}^2 - x_{13}^2\right) \\
&\quad + \frac{J}{I_y} x_8\left(x_{13} - x_{14} + x_{15} - x_{16}\right), \\
\dot{x}_{11} &= x_{12}, \\
\dot{x}_{12} &= \left(\frac{I_x - I_y}{I_z}\right) x_8 x_{10} + \frac{\ell k_b}{I_z}\left(x_{13}^2 - x_{14}^2 + x_{15}^2 - x_{16}^2\right), \\
\dot{x}_{13} &= \alpha_1, \quad \dot{x}_{14} = \alpha_2, \quad \dot{x}_{15} = \alpha_3, \quad \dot{x}_{16} = \alpha_4.
\end{aligned}
$$

where

$\mathbf{x} = [x_1, x_2, \ldots, x_{16}]^\top \in \mathbb{R}^{16}$ is the state vector and the input vector is $\alpha = [\alpha_1, \alpha_2, \alpha_3, \alpha_4]^\top \in \mathbb{R}^4$, are :

$$
\begin{aligned}
&x_1 = x, \quad x_2 = \dot{x} = \dot{x}, \quad x_3 = y, \quad x_4 = \dot{y} = \ddot{y}, \\
&x_5 = z, \quad x_6 = \dot{z} = \ddot{z}, \quad x_7 = \phi, \quad x_8 = \dot{\phi} = \ddot{\phi}, \\
&x_9 = \theta, \quad x_{10} = \dot{\theta} = \ddot{\theta}, \quad x_{11} = \psi, \quad x_{12} = \dot{\psi} = \ddot{\psi}, \\
&x_{13} = \omega_1, \quad x_{13} = \omega_1, \quad x_{14} = \omega_2, \quad x_{14} = \omega_2, \\
&x_{15} = \omega_3, \quad x_{15} = \omega_3, \quad x_{16} = \omega_4, \quad x_{16} = \omega_4.
\end{aligned}
$$

Chapter 3

Quadcopter Dynamics and Control

3.1 Quadcopter Dynamics

Let us define the mathematical model of the quadcopter used; as done in [9].

The 4 control inputs of the drone are:

Thrust U_1 [N] (Throttle) , Moment U_2 [Nm](Roll torque),

Moment U_3 [Nm](Pitch torque), Moment U_4 [Nm](Yaw torque)

$$
\begin{aligned}
U_1 &= m\ddot{z} = c_T \cdot \left(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2\right) \\
U_2 &= I_x\ddot{\phi} = c_T \cdot l \cdot \left(\Omega_4^2 - \Omega_2^2\right) \\
U_3 &= I_y\ddot{\theta} = c_T \cdot l \cdot \left(\Omega_3^2 - \Omega_1^2\right) \\
U_4 &= I_z\ddot{\psi} = c_Q \cdot \left(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2\right) \\
\Omega_{\text{total}} &= -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4
\end{aligned}
$$

The angular velocities of rotors are denoted as Ω_1, Ω_2, Ω_3, and Ω_4 for the motors 1, 2, 3, and 4, respectively. cT and cQ are aerodynamic coefficients of thrust and drag.

A quadcopter has 6 degrees of freedom : 3 for position and 3 for attitude. The variables u, v, and w represent the velocities along the x, y, and z axes in the body frame (B-frame) [m/s], respectively. Similarly, p, q, and r denote the angular velocities about the roll (ϕ), pitch (θ), and yaw (ψ) axes in the B-frame . .Ix,Iy, Iz and Jtp are the moment of inertia around the x,y,z and propeller axis .The NEWTON EULER equations of motion in B-frame are:

$$\dot{u} = (vr - wq) + gS_\theta$$

$$\dot{v} = (wp - ur) - gC_\theta S_\phi$$

$$\dot{w} = (uq - vp) - gC_\theta C_\phi + \frac{U_1}{m}$$

$$\dot{p} = qr\frac{I_y - I_z}{I_x} - \frac{J_{TP}}{I_x}q\Omega + \frac{U_2}{I_x}$$

$$\dot{q} = pr\frac{I_z - I_x}{I_y} + \frac{J_{TP}}{I_y}p\Omega + \frac{U_3}{I_y}$$

$$\dot{r} = pq\frac{I_x - I_y}{I_z} + \frac{U_4}{I_z}$$

Let x,y and z be the translational position and ϕ, θ, and ψ be the angluar position in E-frame. Now we relate the translational and angular component in E-frame to B-frame.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = R \begin{bmatrix} u \\ v \\ w \end{bmatrix} \quad \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = T \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

where the matrices R and T are:

$$R = \begin{bmatrix} C_\theta C_\psi & S_\phi S_\theta C_\psi - C_\phi S_\psi & C_\phi S_\theta C_\psi + S_\phi S_\psi \\ C_\theta S_\psi & S_\phi S_\theta S_\psi + C_\phi C_\psi & C_\phi S_\theta S_\psi - S_\phi C_\psi \\ -S_\theta & S_\phi C_\theta & C_\phi C_\theta \end{bmatrix}$$

$$T = \frac{1}{C_\theta}\begin{bmatrix} C_\theta & S_\phi S_\theta & C_\phi S_\theta \\ 0 & C_\phi C_\theta & -S_\phi C_\theta \\ 0 & S_\phi & C_\phi \end{bmatrix}$$

T tends to the identity matrix I when ϕ and θ are small(¡15 deg) and is true for non-acrobatic drones.

3.2 Drone Parameters

Figure 3.1: Fig.: DJI Phantom 2 drone

Table 3.1: Parameters of the Phantom 2 Used in the Numerical Experiments.

Parameter	Value	Unit
m_m (Motor mass)	0.025	kg
r_m (Motor radius)	0.014	m
m_p (Propeller mass)	0.0055	kg
$\dot{x}_{max} = \dot{y}_{max}$ (Maximum linear velocity x, y)	± 15	m/s
$\dot{x}_{min} = \dot{y}_{min}$ (Minimum linear velocity x, y)	± 15	m/s
$\dot{z}_{max}$ (Maximum linear velocity z)	6	m/s
$\dot{z}_{min}$ (Minimum linear velocity z)	-2	m/s
R (Resistance)	0.2	Ω
L (Inductance)	0H	-
a_{max} (Maximum acceleration)	500	rad/s^2
Ω_{max} (Maximum angular velocity)	1047.19	rad/s
ψ_{max} (Maximum yaw angle)	$+\pi$	rad
ψ_{min} (Minimum yaw angle)	$-\pi$	rad
$\phi_{max} = \theta_{max}$ (Maximum roll/pitch angle)	$+0.61$	rad
$\phi_{min} = \theta_{min}$ (Minimum roll/pitch angle)	-0.61	rad
$\dot{\phi}_{max} = \dot{\theta}_{max}$ (Maximum roll/pitch rate)	$+1.5$	rad/s
$\dot{\phi}_{min} = \dot{\theta}_{min}$ (Minimum roll/pitch rate)	-1.5	rad/s
$\dot{\psi}_{max}$ (Maximum yaw rate)	$+1.5$	rad/s
$\dot{\psi}_{min}$ (Minimum yaw rate)	-1.5	rad/s
K_{th} (Thrust constant)	3.8305×10^{-6}	Ns^2/rad^2
K_{to} (Torque constant)	2.2518×10^{-8}	Nms^2/rad^2

3.3 Solving an Optimal Control Problem (OCP) for Reference Trajectory Generation

The electrical model of a brushless DC motor is shown:

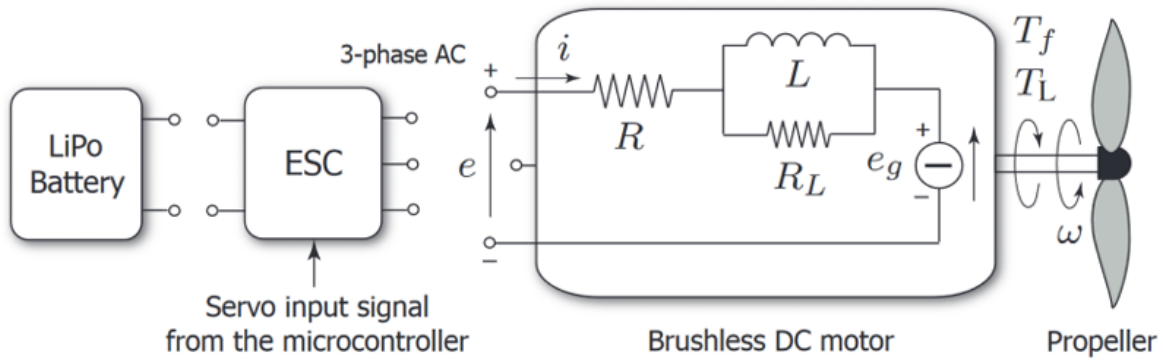

Figure 3.2: BLDC Motor equivalent model

In time [t0,tf], the total energy consumed by the four motors is given by:

$$E_M = \int_{t_0}^{t_f} \sum_{k=1}^{4} e_k(t) i_k(t) dt \tag{3.1}$$

where $e_k(t)$ and $i_k(t)$ are the instantaneous voltage across and current drawn by each motor respectively.

$$e(t) = Ri(t) + K_E \Omega(t) + L\frac{di(t)}{dt} \tag{3.2}$$

$$i(t) = \frac{1}{K_T}\left[T_f + T_L(\Omega(t)) + D_f \Omega(t) + J\frac{d\Omega(t)}{dt}\right] \tag{3.3}$$

where:

R : phase winding resistance [Ω]

L : phase winding inductance [H]

K_E : motor voltage constant [Vs/rad]

$\Omega(t)$: angular velocity of the motor shaft [rad/s]

K_T : torque constant of the motor [Nm/A] (Note: $K_T = K_E$)

K_V : velocity constant of the motor [rpm/V]

T_f : motor friction torque [Nm]

$T_L(\Omega(t))$: load friction torque [Nm] ($T_L(\Omega(t)) = k_{t0}\Omega(t)^2$)

D_f : viscous damping coefficient of the motor [Nms/rad]

J : moment of inertia [kgm^2] ($J = J_m + J_p$)

J_m : moment of inertia of the motor

J_p : moment of inertia of the propeller

Assumptions:

1)The mission starts from a hovering state and ends in a hovering state, i.e., $\Omega(t_0) = \Omega(t_f)$

2)The efficiency of the motors are 1 (i.e,ideal).

3)The onboard devices ,controller, ESC and gearbox uses negligible energy.

Using Eq. (4.2), Eq. (4.3), and Assumption 1, we rewrite Eq. (4.1) as

$$E_M = \int_{t_0}^{t_f} \sum_{i=1}^{4} \left(c_1 + c_2\Omega_i(t) + c_3\Omega_i^2(t) + c_4\Omega_i^3(t) + c_5\Omega_i^4(t) + c_6\dot{\Omega}_i^2(t) \right) dt \tag{3.4}$$

where,

$$c_1 = \frac{RT_f^2}{K_T^2}, c_2 = \frac{T_f}{K_T}\left(\frac{2RD_f}{K_T} + K_E\right), c_3 = \frac{D_f}{K_T}\left(\frac{RD_f}{K_T} + K_E\right) + \frac{2RT_f k_{t0}}{K_T^2},$$
$$c_4 = \frac{k_{t0}}{K_T}\left(\frac{2RD_f}{K_T} + K_E\right), c_5 = \frac{Rk_{t0}^2}{K_T^2}, c_6 = \frac{RJ^2}{K_T^2},$$

$c_1, c_2, \ldots, c_6$ are constants characteristic of the motor-propeller pair.

For a given mission $\mathcal{M}(x_i, y_i, z_k)$, E_M denotes the total energy consumed by the quadrotor to traverse a trajectory $\mathcal{T}(x_i(t), y_j(t), z_k(t))$.The OCP being solved here is of the form:

Objective:

$$E_M^* = \min_{u(t),t_f} E_M = \min_{u(t),t_f} \int_{t_0}^{t_f} \left[\sum_{i=13}^{16} \left(c_1 + c_2 x_i(t) + c_3 x_i^2(t) + c_4 x_i^3(t) + c_5 x_i^4(t) \right) + c_6 \sum_{k=1}^{4} \alpha_k^2(t) \right] dt \quad (3.5)$$

System Constraints:

$$\dot{x}(t) = f(x(t)) + Bu(t) \quad (3.6)$$

State Constraints:

$$x_{i,\min} < x_i < x_{i,\max} \quad (3.7)$$

Input Constraints:

$$\alpha_{k,\min} < \alpha_k < \alpha_{k,\max} \quad (3.8)$$

Boundary Conditions:

$$x_i(t_0) = x_{i,t_0}, \quad x_i(t_f) = x_{i,t_f} \quad (3.9)$$

$i \in \{1, 16\}$, $k \in \{1, 4\}$. where

$$\mathbf{X}_{\text{vector}}^\top = \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} & \phi & \theta & \psi & \dot{\phi} & \dot{\theta} & \dot{\psi} & \Omega_1 & \Omega_2 & \Omega_3 & \Omega_4 \end{bmatrix}$$

$$\mathbf{u}_{\text{vector}}^\top = \begin{bmatrix} \dot{\Omega}_1, & \dot{\Omega}_2, & \dot{\Omega}_3, & \dot{\Omega}_4 \end{bmatrix} = \begin{bmatrix} \alpha_1, & \alpha_2, & \alpha_3, & \alpha_4 \end{bmatrix}$$

$f(x) \in \mathbb{R}^{16\times 1}$ is the drone dynamics,$B \in \mathbb{R}^{16\times 4}$ is the control input matrix

$$f(x) = \begin{bmatrix} x_4 \\ x_5 \\ x_6 \\ \frac{k_{lh}}{m}(c_{x_7} c_{x_9} s_{x_8} + s_{x_7} s_{x_9}) \sum_{i=13}^{i=16} x_i^2 \\ \frac{k_{lh}}{m}(c_{x_7} s_{x_8} s_{x_9} - c_{x_9} s_{x_7}) \sum_{i=13}^{i=16} x_i^2 \\ -g + \frac{k_{lh}}{m}(c_{x_8} c_{x_7}) \sum_{i=13}^{i=16} x_i^2 \\ x_{10} \\ x_{11} \\ x_{12} \\ \frac{I_y - I_z}{I_x} x_{11} x_{12} - \frac{J}{I_x} x_{11} \Omega_5 + \frac{l \kappa_b}{I_x}(x_{14}^2 - x_{16}^2) \\ \frac{I_z - I_x}{I_y} x_{10} x_{12} + \frac{J}{I_y} x_{10} \Omega_5 + \frac{l \kappa_b}{I_y}(x_{15}^2 - x_{13}^2) \\ \frac{I_x - I_y}{I_z} x_{10} x_{11} + \frac{\kappa_t 0}{I_z}(x_{13}^2 + x_{15}^2 - x_{14}^2 - x_{16}^2) \end{bmatrix} \quad B = \begin{bmatrix} \mathbf{0}_{12\times 4} \\ I_{4\times 4} \end{bmatrix}$$

Solving it we get $u^*(t)$ and t_f^* for the least energy E_M^*. Using $u^*(t)$ and t_f^*, we compute the optimal state $x^*(t)$ and the trajectory $\mathcal{T}(x_i^*(t), y_j^*(t), z_k^*(t))$ for the mission $\mathcal{M}(x_i, y_j, z_k)$.

Table 3.2: State Constraints

$\dot{x}_{max}, \dot{y}_{max}$	$\dot{x}_{min}, \dot{y}_{min}$	$\dot{z}_{max}$	$\dot{z}_{min}$	ϕ_{max}, θ_{max}	ϕ_{min}, θ_{min}	$\dot{\phi}_{max}, \dot{\theta}_{max}$	$\dot{\phi}_{min}, \dot{\theta}_{min}$	ψ_{max}	ψ_{min}	$\dot{\psi}_{max}$	$\dot{\psi}_{min}$
+15	-15	+6	-2	+0.61	-0.61	+1.5	-1.5	$+\pi$	$-\pi$	+1.5	-1.5

3.4 Controller Model 1

The control architecture consists of two interconnected controllers: a position controller in outer and attitude controller in inner loop. The position controller is responsible for managing the position variables x, y, and z. Using state feedback linearization, it computes the thrust input U_1 and the desired roll (ϕ) and pitch (θ) angles. These outputs ensure that the UAV can track the target position (x, y, z). The computed thrust U_1 is directly applied to the open-loop system, while the desired angles ϕ and θ are provided as reference inputs to the attitude controller. This inner controller, based on an LPV-MPC (Linear Parameter Varying Model Predictive Control) framework, calculates the remaining control inputs: U_2, U_3, and U_4. These inputs control the UAV's roll, pitch, and yaw dynamics, enabling precise and stable trajectory tracking.

3.4.1 General constrained linear MPC controller (as attitude controller):

$$J = \min_{e_t, u_t} \left\{ \frac{1}{2} \sum_{k=0}^{N-1} \left(e_t^T Q e_{t+k} + u_t^T R u_{t+k} \right) + \frac{1}{2} e_t^T Q e_{t+N} \right\}, \tag{3.10}$$

where $e_k = r_k - y_k = r_k - Cx_k and u_k = u_{k-1} + \Delta u_k$.

We augment the system by including the previous input from one sample of past as an additional state.

$$\tilde{x_{k+1}} = \begin{bmatrix} x_{k+1} \\ u_k \end{bmatrix} = \begin{bmatrix} A & B \\ 0 & I \end{bmatrix} \begin{bmatrix} x_k \\ u_{k-1} \end{bmatrix} + \begin{bmatrix} B \\ I \end{bmatrix} \Delta u_k = \tilde{A}\tilde{x}_k + \tilde{B}\Delta u_k$$

$$y_k = \begin{bmatrix} C & 0 \end{bmatrix} \begin{bmatrix} x_k \\ u_{k-1} \end{bmatrix} = \tilde{C} \begin{bmatrix} x_k \\ u_{k-1} \end{bmatrix}$$

Here under small angle assumption, our drone A and B matrices in LPV(linear parameter varying) format where A and B are goverened by these following state space equations:

$$\begin{bmatrix}\dot{\phi}\\ \ddot{\phi}\\ \dot{\theta}\\ \ddot{\theta}\\ \dot{\psi}\\ \ddot{\psi}\end{bmatrix} = \begin{bmatrix}0 & 1 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & \Omega\frac{J_r P}{I_x} & 0 & \frac{I_y-I_z}{I_x}\\ 0 & 0 & 1 & 0 & 0 & 0\\ 0 & \Omega\frac{J_r P}{I_y} & 0 & 0 & 0 & \frac{I_z-I_x}{I_y}\\ 0 & 0 & 0 & 0 & 0 & 1\\ 0 & \frac{\dot{\theta}}{2}\frac{I_x-I_y}{I_z} & 0 & \frac{\dot{\phi}}{2}\frac{I_x-I_y}{I_z} & 0 & 0\end{bmatrix}\begin{bmatrix}\phi\\ \dot{\phi}\\ \theta\\ \dot{\theta}\\ \psi\\ \dot{\psi}\end{bmatrix} + \begin{bmatrix}0 & 0 & 0 & 0\\ 0 & \frac{1}{I_x} & 0 & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & \frac{1}{I_y} & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & \frac{1}{I_z}\end{bmatrix}\begin{bmatrix}U_2\\ U_3\\ U_4\end{bmatrix} \tag{3.11}$$

We calculate at each time step prediction for the entire horizon period(N). It can be compactly represented as:

$$\tilde{x} = \begin{bmatrix}\tilde{B} & & & & \\ \tilde{A}\tilde{B} & \tilde{B} & & & \\ \tilde{A}^2\tilde{B} & \tilde{A}\tilde{B} & \tilde{B} & & \\ \vdots & \vdots & \ddots & \ddots & \\ \tilde{A}^{N-1}\tilde{B} & \tilde{A}^{N-2}\tilde{B} & \cdots & \tilde{A}\tilde{B} & \tilde{B}\end{bmatrix}\Delta u + \begin{bmatrix}\tilde{A}\\ \tilde{A}^2\\ \vdots\\ \tilde{A}^N\end{bmatrix}\tilde{x}_t = \overline{\overline{C}}\Delta u + \hat{A}\tilde{x}_t. \tag{3.12}$$

$$J = \frac{1}{2}\left((\overline{\overline{C}}\Delta u + \hat{A}x_t)^T\overline{\overline{Q}}(\overline{\overline{C}}\Delta u + \hat{A}x_t) + \frac{1}{2}\Delta u^T\overline{\overline{R}}\Delta u - r^T\overline{\overline{T}}(\overline{\overline{C}}\Delta u + \hat{A}x_t)\right), \text{ where} \tag{3.13}$$

$$\overline{\overline{Q}} = \begin{bmatrix}\tilde{C}^T\,Q\tilde{C} & 0 & \cdots & 0\\ 0 & \tilde{C}^T\,Q\tilde{C} & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & \tilde{C}^T\,S\tilde{C}\end{bmatrix}, \overline{\overline{T}} = \begin{bmatrix}Q\tilde{C} & 0 & \cdots & 0\\ 0 & Q\tilde{C} & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & S\tilde{C}\end{bmatrix}, \quad \overline{\overline{R}} = \begin{bmatrix}R & 0 & \cdots & 0\\ 0 & R & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & R\end{bmatrix}$$

Finally, J becomes

$$J = \frac{1}{2}\Delta u^T\overline{\overline{H}}\Delta u + f^T\Delta u, \text{ where } \overline{\overline{H}} = \left(\overline{\overline{C}}^T\overline{\overline{QC}} + \overline{\overline{R}}\right),$$

$$f^T = \begin{bmatrix}\tilde{x}_t^T & r^T\end{bmatrix}\begin{bmatrix}\hat{A}^T\overline{\overline{QC}}\\ -\overline{\overline{TC}}\end{bmatrix} \tag{3.14}$$

We then solve the optimization problem within interval $T_s = 0.1$ sec using MATLAB's QUADPROG. Also, we subject the UAV to some constraints on $\Omega_{\max} = 1047$ and $\Omega_{\min} = 0$.

By substituting them, we get constraints on rotor forces and torques as $\texttt{min}, \texttt{max}(U_1, U_2, U_3, U_4)$.

3.4.2 State Feedback linearization (as position controller):

$$\ddot{x} = (C_\theta S_\phi + S_\theta S_\psi)\frac{U_1}{m} \tag{3.15}$$

$$\ddot{y} = (C_\theta S_\phi S_\psi + C_\theta C_\psi)\frac{U_1}{m} \tag{3.16}$$

$$\ddot{z} = -g + C_\theta C_\phi \frac{U_1}{m} \tag{3.17}$$

We define:

$$\mathrm{e}_x = x_R - x, \dot{e_x} = \dot{x}_R - \dot{x}, \ddot{e_x} = \ddot{x}_R - \ddot{x} = v_x,$$

$$e_y = y_R - y, \dot{e_y} = \dot{y}_R - \dot{y}, \ddot{e_y} = \ddot{y}_R - \ddot{y} = v_y,$$

$$e_z = z_R - z, \dot{e_z} = \dot{z}_R - \dot{z}, \ddot{e_z} = \ddot{z}_R - \ddot{z} = v_z$$

The variables vx, vy and vz are selected as control inputs for the linearized state feedback control strategy and are defined as:

$$v_x = -k_{1_x} e_x - k_{2_x} \dot{e_x}$$

$$v_y = -k_{1_y} e_y - k_{2_y} \dot{e_y}$$

$$v_z = -k_{1_z} e_z - k_{2_z} \dot{e_z}$$

We choose real and negative poles(-2 and -1), such that the constants(k1,k2) in these equations can be computed(k1=-2,k2=-3), that makes the error tends to zero at time tends to infinity. Thus, by setting $v_x = 0$, $v_y = 0$, and $v_z = 0$ and solving the three equations, we obtain the references θ, ϕ, and U_1.

$$\theta = \tan^{-1}(ac + bd)$$

$$\phi = \tan^{-1}((ad - bc)/\sqrt{1 + (ac + bd)^2})$$

$$U_1 = \frac{(v_z + g)m}{C_\phi C_\theta}$$

$$a = v_x/(v_z + g), \quad b = v_y/(v_z + g), \quad c = C_\psi, \quad d = S_\psi$$

The feedback linearization controller calculates the thrust input U_1, which is directly applied to the nonlinear UAV model to control altitude and overall propulsion. Simultaneously, the desired roll (ϕ) and pitch (θ) angles are generated as reference signals and passed to the MPC-based attitude controller. This approach ensures coordination between the outer and inner control loops for precise trajectory tracking.

Combined Cascaded Controller Structure

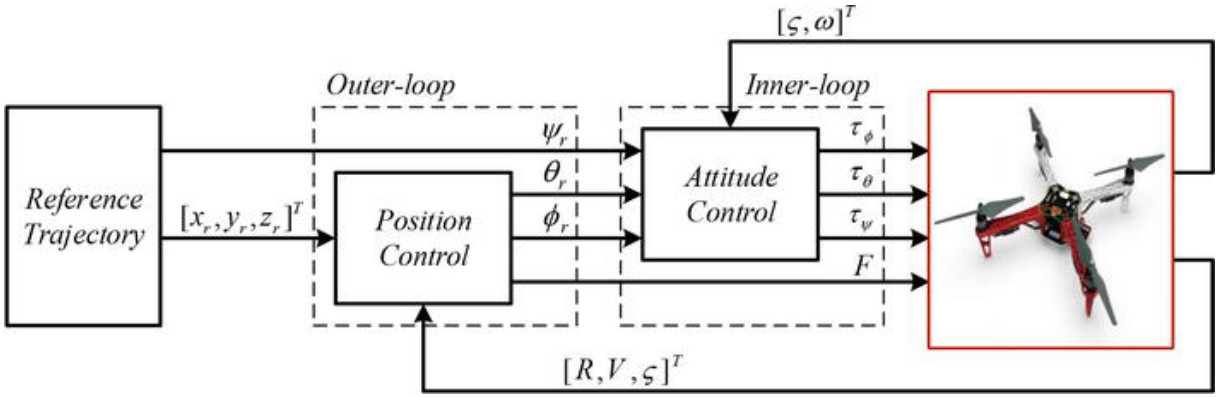

Figure 3.3: Cascade control with inner loop-MPC and outer-feedback linearization

TRACKING RESULTS WITH THIS CONTROLLER:

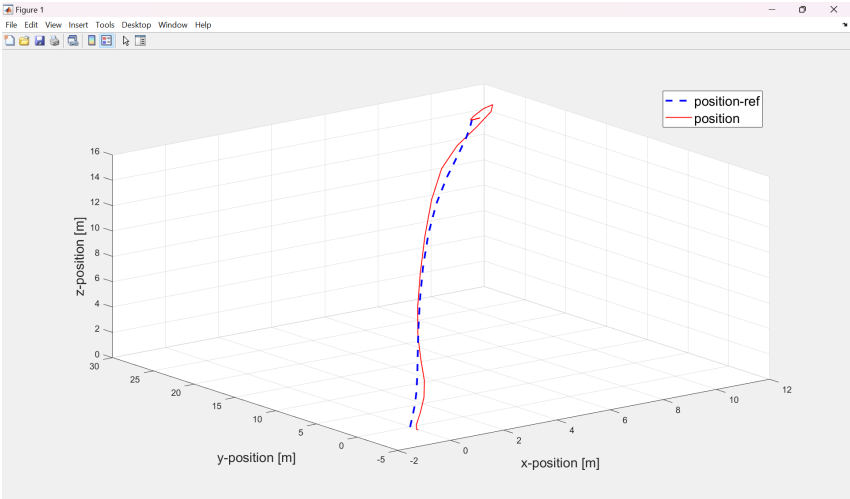

Figure 3.4: Mission (10,15,25)taking minimum of 7.2 seconds to reach whereas tf*=5.4 seconds

Remarks

We can see the controller gives very poor results in tracking a fast energy optimal trajectory which is obvious as we used hover linearized model of the attitude dynamics .

In cascaded controller, frequency of the inner loop has to be kept higher than that of the outer loop. The closed loop poles of the state feedback controller are chosen as -2 ,-1 for fast convergence of position. however inner controller is mpc that makes it quite challenging to solve. Also These are lot of hyper-parameters that needs fine tuning.

Thus this controller model isnt very suitable for fast drones.

3.5 Controller Model 2

In this control architecture, we have used a single level NMPC controller that aims to track desired position, velocities, attitudes and their rates altogether. In linear MPC, we had linear dynamics and quadratic cost that results in a Quadratic Programming (QP) problem, which is convex and can be solved efficiently using standard QP solvers (e.g., OSQP, quadprog in MATLAB). Whereas in NMPC, we have a continuous OCP with nonlinear dynamics. So first we need to transcribe it to a NLP (nonlinear program) by some numerical shooting methods which converts the continuuos,infinite dimensional OCP to a discrete finite-dimensional problem that can be solved by CASADI, ACADO, or fmincon etc.

NMPC Controller

Running (stage) Costs:

$$\ell(\mathbf{x}, \mathbf{u}) = \|\mathbf{x}_u - \mathbf{x}^{ref}\|_Q^2 + \|\mathbf{u} - \mathbf{u}^{ref}\|_R^2 \tag{3.18}$$

Optimal Control Problem (OCP):

$$\underset{\mathbf{u} \text{ admissible}}{} \quad J_N(\mathbf{x}_0, \mathbf{u}) = \sum_{k=0}^{N-1} \ell(\mathbf{x}_u(k), \mathbf{u}(k)) \tag{3.19}$$

$$\text{subject to:} \quad \mathbf{x}_u(k+1) = \mathbf{f}(\mathbf{x}_u(k), \mathbf{u}(k)), \tag{3.20}$$

$$\mathbf{x}_u(0) = \mathbf{x}_0, \tag{3.21}$$

$$\mathbf{u}(k) \in U, \quad \forall k \in [0, N-1] \tag{3.22}$$

$$\mathbf{x}_u(k) \in X, \quad \forall k \in [0, N] \tag{3.23}$$

Some Generally Used Transcription Methods

In Nonlinear Model Predictive Control (NMPC), the Optimal Control Problem (OCP) is transformed into a Nonlinear Programming (NLP) problem using various transcription methods. Below are descriptions of three commonly used methods:

- **Single Shooting**: Discretize the control input $u(t)$ into a sequence $u_0, u_1, \ldots, u_{N-1}$. Integrate the system dynamics $\dot{x} = f(x, u)$ numerically over the prediction horizon to compute the state trajectory. Optimize over the control sequence u_k. The NLP variables are the control inputs u_k.

- **Multiple Shooting**: Introduce state variables x_k at each time interval. Integrate the dynamics within each interval and enforce continuity constraints $x_{k+1} = x_k + \int_{t_k}^{t_{k+1}} f(x,u)\,dt$ to ensure the state at the end of one interval matches the start of the next. The NLP variables include both the control inputs u_k and state variables x_k.
- **Orthogonal Collocation**: Approximate the state $x(t)$ and control $u(t)$ as polynomials (e.g., using Legendre polynomials) within time intervals. Enforce the system dynamics at collocation points, such that $\dot{x}(t_c) = f(x(t_c), u(t_c))$, where t_c are the collocation points (e.g., Legendre-Gauss points, Chebyshev-Gauss points). Optimize the polynomial coefficients. The NLP variables are the coefficients of these polynomials.

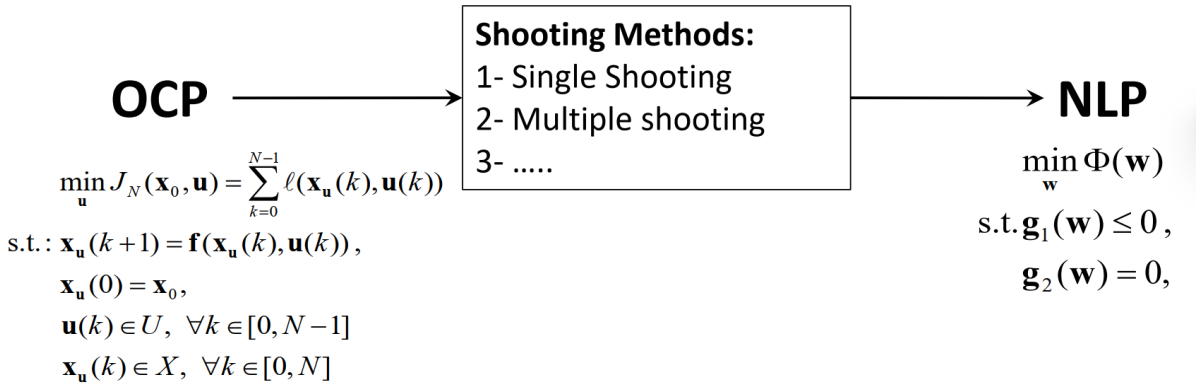

Figure 3.5: Conversion of NLP to OCP

For our quadcopter we have used multiple shooting method because it gaurantees much better convergence for a highly non linear plant like quadcopter. And also its faster computation capability than orthogonal collocation makes it a better option for real time applications.

3.5.1 NMPC Controller formulation 1

NMPC Problem at i^{th} Step

$$\min_{\mathbf{u0},..,\mathbf{uN-1},\mathbf{x1},..\mathbf{xN}} J(i) = \sum_{k=1}^{N} \left[\left(\mathbf{x}_{ref}^{i+k} - \mathbf{x}^{i+k} \right)^T \mathbf{Q}_{12\times 12} \left(\mathbf{x}_{ref}^{i+k} - \mathbf{x}^{i+k} \right) + \left(\mathbf{u}_{ref}^{i+k} - \mathbf{u}^{i+k} \right)^T \mathbf{R}_{4\times 4} \left(\mathbf{u}_{ref}^{i+k} - \mathbf{u}^{i+k} \right) \right] \tag{3.24}$$

where

$$\mathbf{x}_{ref}^{(i)} = \begin{bmatrix} x_{ref} \\ \dot{x}_{ref} \\ y_{ref} \\ \dot{y}_{ref} \\ z_{ref} \\ \dot{z}_{ref} \\ \phi_{ref} \\ \dot{\phi}_{ref} \\ \theta_{ref} \\ \dot{\theta}_{ref} \\ \psi_{ref} \\ \dot{\psi}_{ref} \end{bmatrix}^{(i)}_{12\times 1}, \quad \mathbf{u}_{ref}^{(i)} = \begin{bmatrix} u_{1ref} \\ u_{2ref} \\ u_{3ref} \\ u_{4ref} \end{bmatrix}^{(i)}_{4\times 1} \tag{3.25}$$

Subject to:

$$\mathbf{x} \in [\mathbf{x}_{min}, \mathbf{x}_{max}] \tag{3.26}$$

$$\mathbf{u} \in [\mathbf{u}_{min}, \mathbf{u}_{max}] \tag{3.27}$$

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) \text{ (quadcopter dynamics in E- frame)} \tag{3.28}$$

$$\mathbf{x}_0 = \mathbf{x}_0 \text{ (initial condition)} \tag{3.29}$$

Simulation Results:

1) First to start with we selected a sampling time interval long enough so that CASADI can solve decently an NLP of such size, keeping in mind not to shorten too much so that tracking performance is satisfactory . Thus we select sampling time Ts=.08 seconds and is same all subsequent problems. We selected weights and prediction horizon after prioritising tracking of attitudes and their rates with heavier weights than position and velocities . After running the genetic algorithm optimization , we got : Q=diag([0.0986, 0.7171, 0.0014, 0.0361, 0.0434, 0.3169, 2624, 635 , 77.7 , 54, 0, 0]) R=diag[(9e-8, 9e-8, 9e-8, 9e-8)], N=20

Table 3.3: Energy consumption and their deviations from optimal energy

Mission M_i	$t^*_{M_i}$ [sec]	$E^*_{M_i}$ [J]	E_{M_i} [J]	error (%)
$M_1 = M(x_{00}, y_{00}, z_{30})$	6.63	8717.20	8648	-0.79
$M_2 = M(x_{10}, y_{00}, z_{20})$	5.08	6881.63	6831	-0.74
$M_3 = M(x_{50}, y_{00}, z_{20})$	6.94	10064	10036	-0.28
$M_4 = M(x_{10}, y_{25}, z_{15})$	5.41	7818	7739	-1.01

$$M_1 = M(x_{00}, y_{00}, z_{30})$$

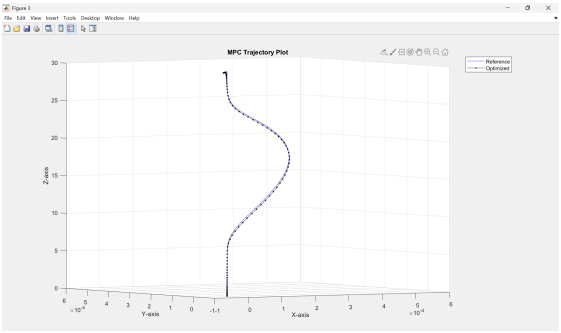

Figure 3.6: 3D trajectory tracking

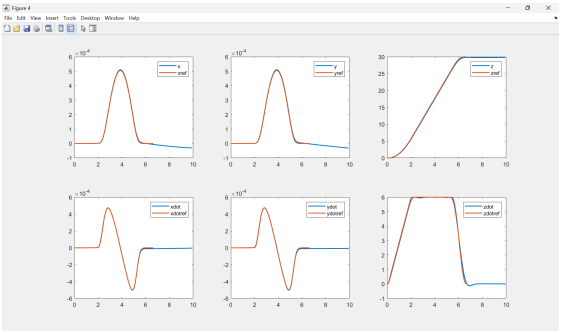

Figure 3.7: Position and velocities tracking

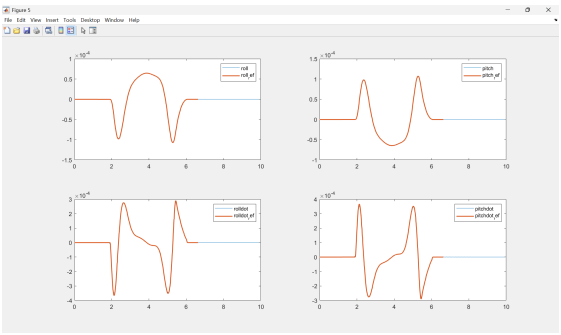

Figure 3.8: Attitude and its rates tracking

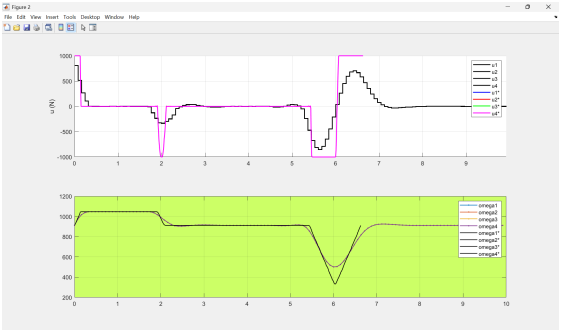

Figure 3.9: Rotor angular velocities, accelerations, and time profile

$$M_2 = M(x_{10}, y_{00}, z_{20})$$

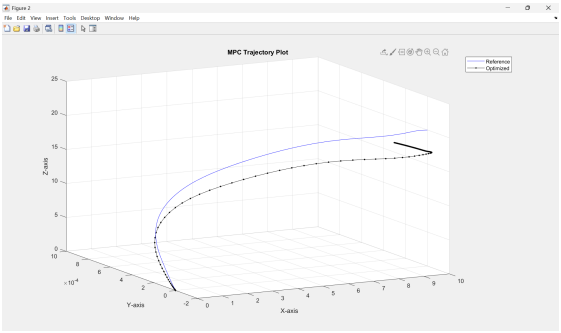

Figure 3.10: 3D trajectory tracking

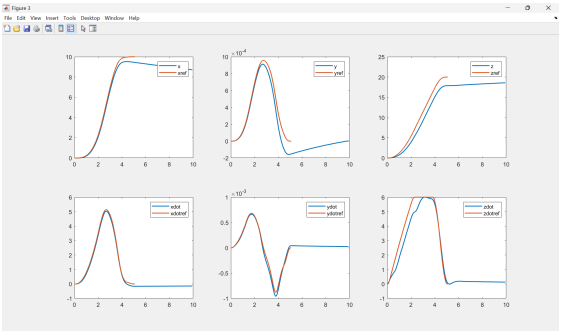

Figure 3.11: Position and velocities tracking

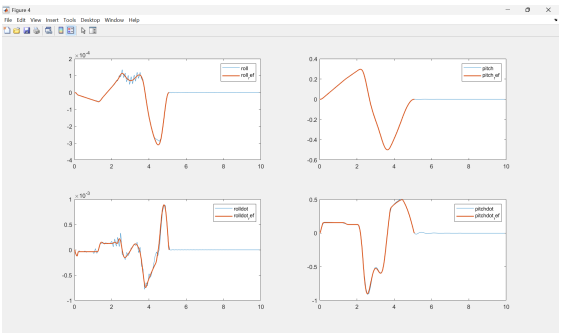

Figure 3.12: Attitude and its rates tracking

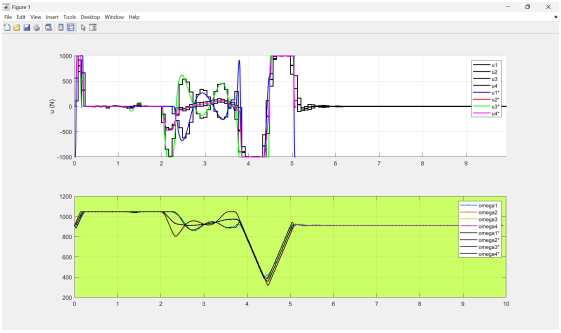

Figure 3.13: Rotor angular velocities, accelerations and time profile

$$M_3 = M(x_{50}, y_{00}, z_{20})$$

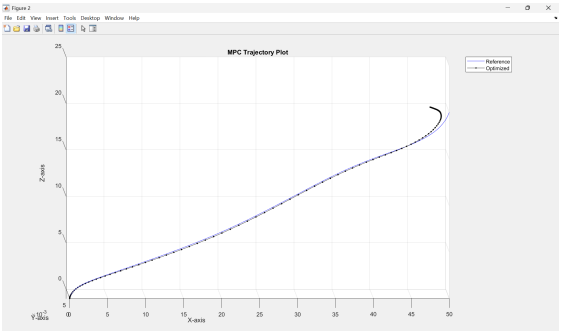

Figure 3.14: 3D trajectory tracking

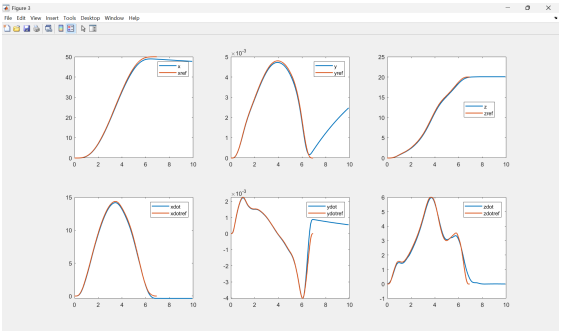

Figure 3.15: Position and velocities tracking

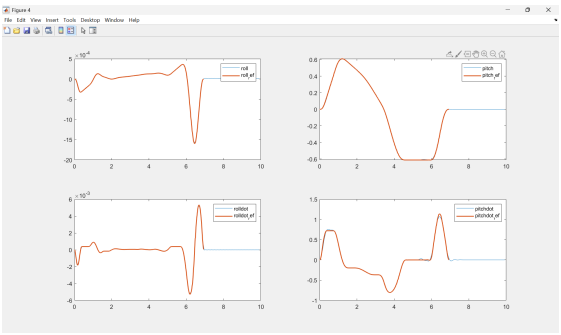

Figure 3.16: Attitude and its rates tracking

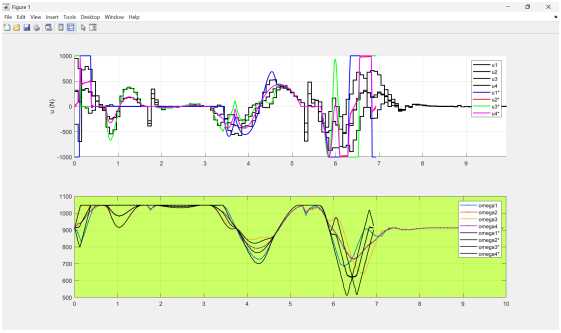

Figure 3.17: Rotor angular velocities, accelerations, and time profile

$$M_4 = M(x_{10}, y_{25}, z_{15})$$

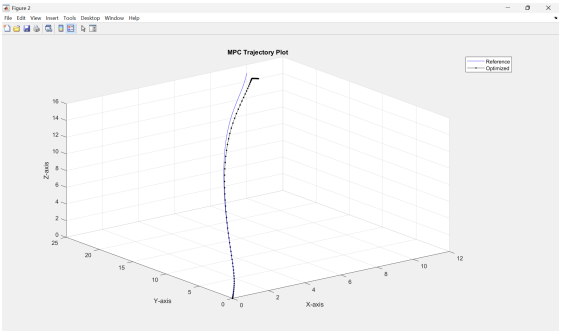

Figure 3.18: 3D trajectory tracking

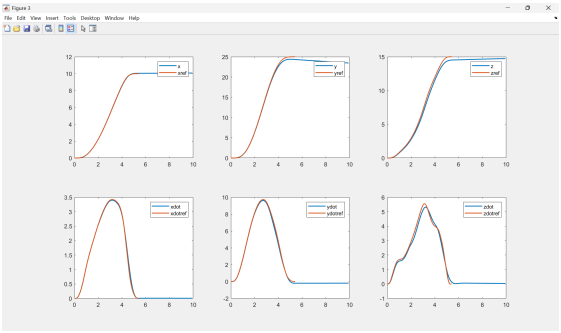

Figure 3.19: Position and velocities tracking

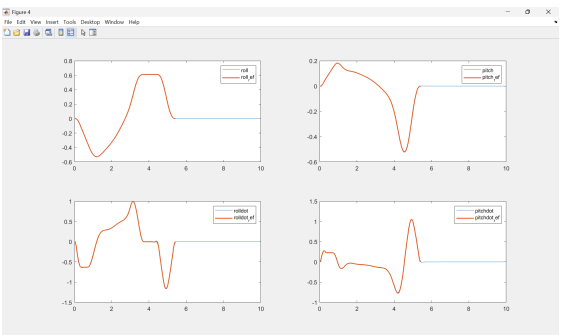

Figure 3.20: Attitude and its rates tracking

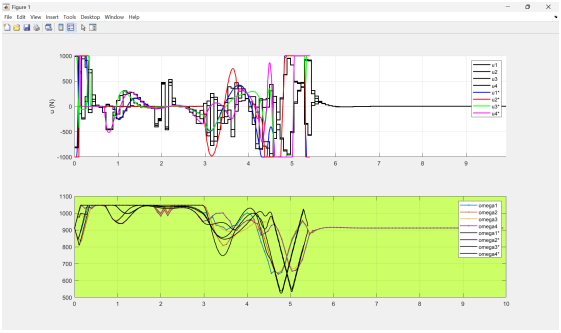

Figure 3.21: Rotor angular velocities, accelerations and time profile

REMARKS:

It does satisfactory tracking with low energy consumption of 7.7552e+03 joules (E*=7813 joules) in tf=5.42 seconds,and manages to reach endpoint(98 % band) within tf*=5.42 seconds .

Although satisfactory tracking but poor stabilization , and is clearly evident when we run the simulation for longer time.

In this formulation, we prioritized open-loop attitude references, causing suboptimal attitude calculation during disturbances. For instance, starting from $x_0 = [1,\ 0,\ 1,\ \mathbf{0}_{12\times1}]$ takes nearly $t_f = 20$ seconds to reach the 98% settling band.

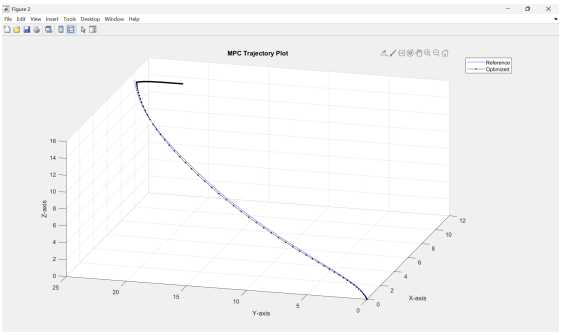

Figure 3.22: Running simulation for 1 minute

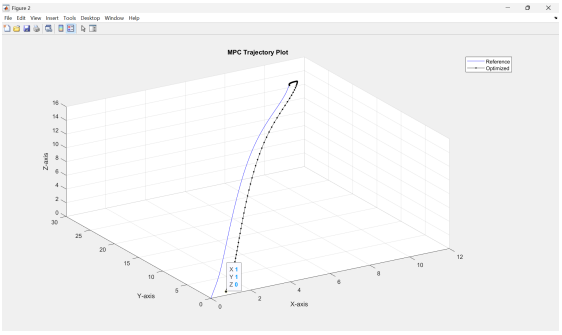

Figure 3.23: Tracking starting from $x_0 = [1,\ 1,\ 0]$

2) Next we selected weights and prediction horizon after prioritising tracking of linear position and velocities with heavier weights than attitudes and their rates . After running the genetic algorithm optimization , we got :

Q=diag([149700, 5741400, 61100, 161200, 325100, 879200, 1020, 413, 2054, 611, 0, 0])

R=diag[(.001, .001, .001, .001)],

N=14

Simulation Results:

Table 3.4: Energy consumption and their deviations from optimal energy

Mission M_i	$t^*_{M_i}$ [sec]	$E^*_{M_i}$ [J]	E_{M_i} [J]	error (%)
$M_1 = M(x_{00}, y_{00}, z_{30})$	6.63	8717.20	8639.5	-0.89
$M_2 = M(x_{10}, y_{00}, z_{20})$	5.08	6881.63	6823	-0.85
$M_3 = M(x_{50}, y_{00}, z_{20})$	6.94	10064	10124	0.60
$M_4 = M(x_{10}, y_{25}, z_{15})$	5.41	7813.28	7728	-1.09

$$M_1 = M(x_{00}, y_{00}, z_{30})$$

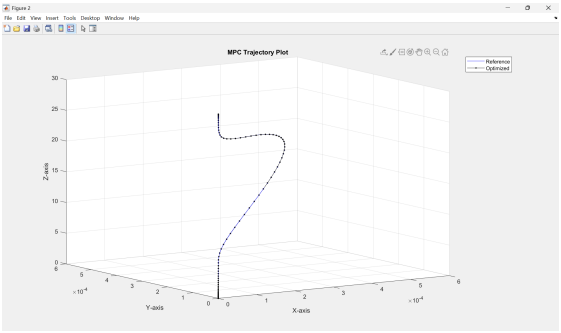

Figure 3.24: 3D trajectory tracking

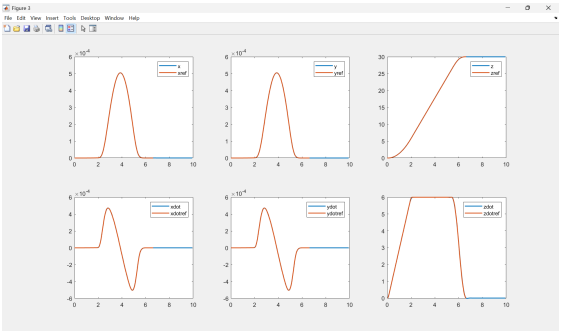

Figure 3.25: Position and velocities tracking

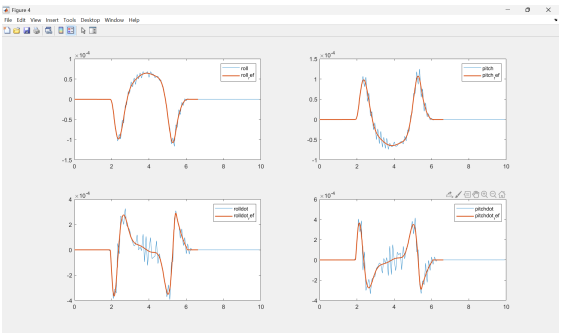

Figure 3.26: Attitude and its rates tracking

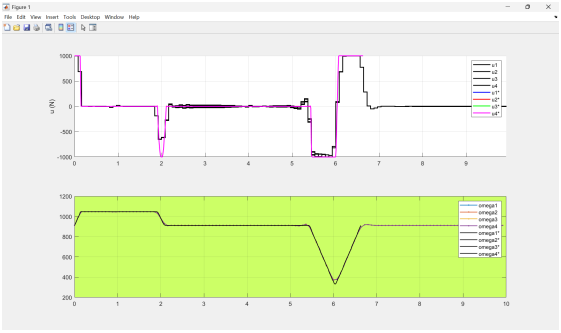

Figure 3.27: Rotor angular velocities, accelerations, and time profile

$$M_2 = M(x_{10}, y_{00}, z_{20})$$

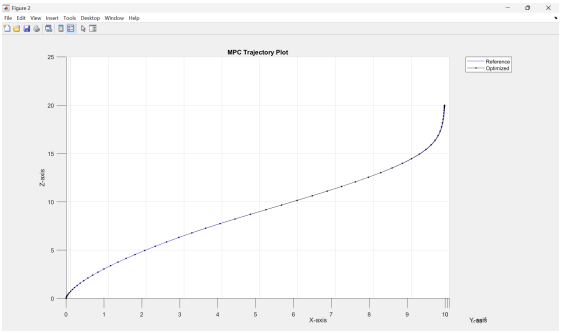

Figure 3.28: 3D trajectory tracking

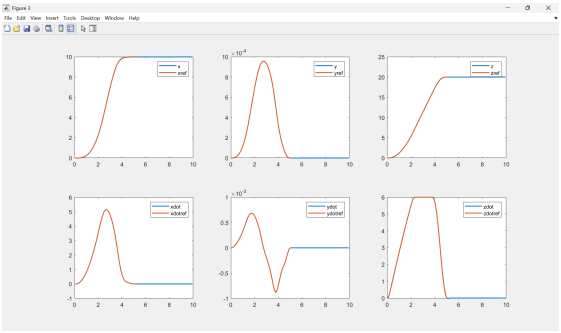

Figure 3.29: Position and velocities tracking

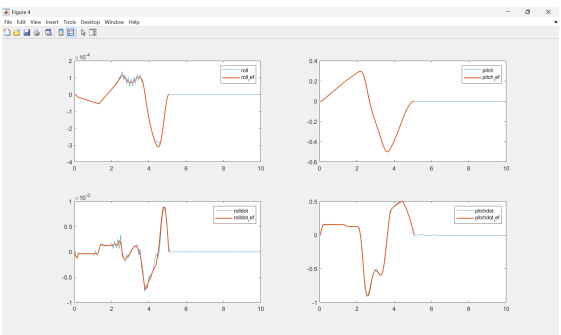

Figure 3.30: Attitude and its rates tracking

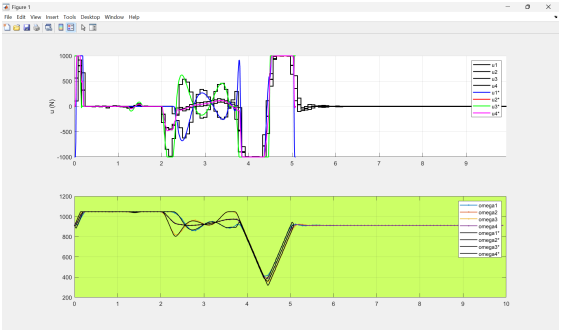

Figure 3.31: Rotor angular velocities, accelerations and time profile

$$M_3 = M(x_{50}, y_{00}, z_{20})$$

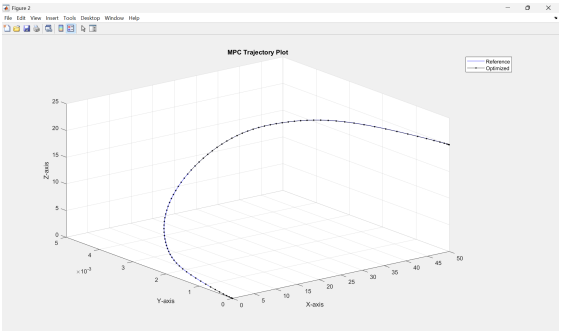

Figure 3.32: 3D trajectory tracking

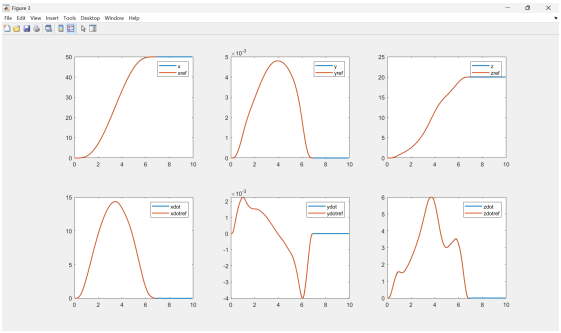

Figure 3.33: Position and velocities tracking

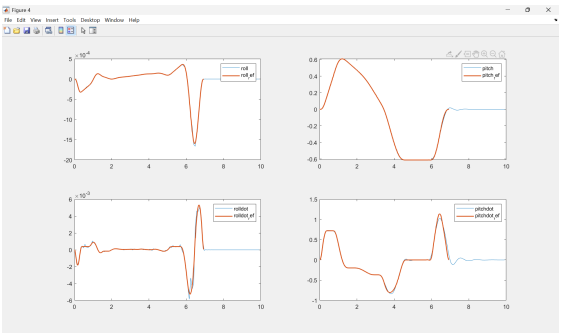

Figure 3.34: Attitude and its rates tracking

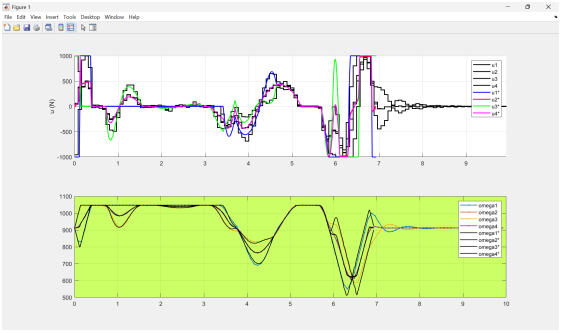

Figure 3.35: Rotor angular velocities, accelerations and time profile

$$M_4 = M(x_{10}, y_{25}, z_{15})$$

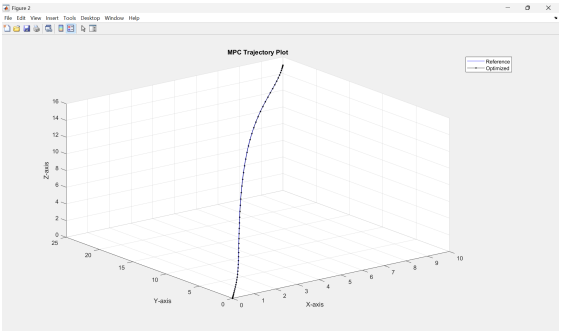

Figure 3.36: 3D trajectory tracking

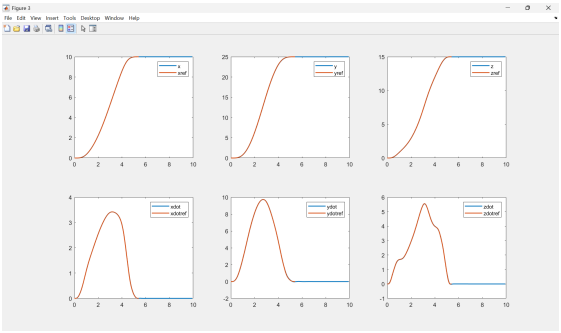

Figure 3.37: Position and velocities tracking

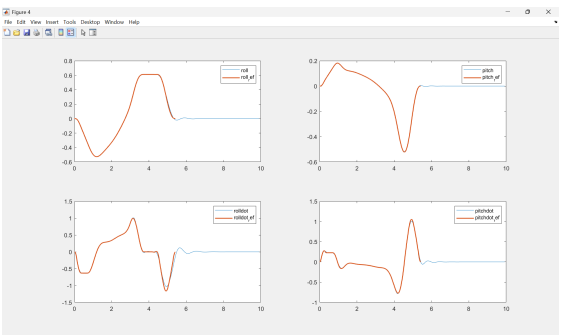

Figure 3.38: Attitude and its rates tracking

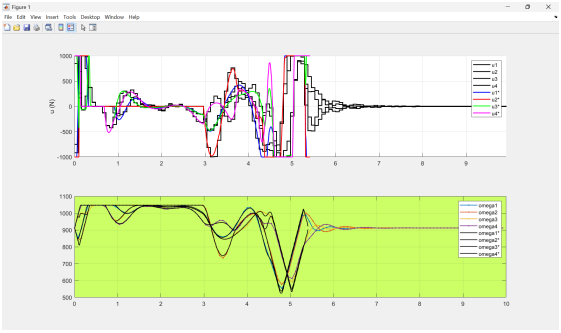

Figure 3.39: Rotor angular velocities, accelerations and time profile

REMARKS:

It does satisfactory tracking with lower energy consumption of 7.7282e+03 joules (E*=7813 joules) in tf=5.42 seconds,and manages to reach endpoint(98 % setlling band) within tf*=5.42 seconds . This controller is giving both satisfactory tracking and good stabilization , and is clearly evident when the simulation runs for longer time.

In this formulation also, we give some priority to open-loop attitude references, causing suboptimal attitude calculation during disturbances. For instance, starting from $x_0 = [1,\ 0,\ 1,\ \mathbf{0}_{12\times1}]$ takes nearly $t_f = 25$ seconds to reach the 98% settling band.

3.5.2 NMPC Controller formulation 2

NMPC Problem at i^{th} Step

$$\begin{aligned}\min_{\mathbf{u}_0,\ldots,\mathbf{u}_{N-1},\mathbf{x}_1,\ldots,\mathbf{x}_N,\omega_1,\ldots,\omega_N} J(i) = &\sum_{k=1}^{N}\left(\left(\mathbf{X}_{\text{ref}}^{(i+k)}-\mathbf{X}^{(i+k)}\right)^T\mathbf{Q}_{6\times6}\left(\mathbf{X}_{\text{ref}}^{(i+k)}-\mathbf{X}^{(i+k)}\right)\right)\\ &+\sum_{k=1}^{N}\left(\left(\boldsymbol{\omega}^{(i+k)}-\omega_{\text{hover}}\mathbf{1}_4\right)^T\mathbf{R}_{4\times4}\left(\boldsymbol{\omega}^{(i+k)}-\omega_{\text{hover}}\mathbf{1}_4\right)\right)\\ &+T\left\{\sum_{k=1}^{N}\left[\left(\left(\omega_2^{(i+k)}\right)^2-\left(\omega_{2,\text{ref}}^{(i+k)}\right)^2\right)-\left(\left(\omega_4^{(i+k)}\right)^2-\left(\omega_{4,\text{ref}}^{(i+k)}\right)^2\right)\right]\right.\\ &\left.+\sum_{k=1}^{N}\left[\left(\left(\omega_3^{(i+k)}\right)^2-\left(\omega_{3,\text{ref}}^{(i+k)}\right)^2\right)-\left(\left(\omega_1^{(i+k)}\right)^2-\left(\omega_{1,\text{ref}}^{(i+k)}\right)^2\right)\right]\right\}\end{aligned} \tag{3.30}$$

where

$$\mathbf{x}_{ref}^{(i)} = \begin{bmatrix} x_{ref}\\ \dot{x}_{ref}\\ y_{ref}\\ \dot{y}_{ref}\\ z_{ref}\\ \dot{z}_{ref}\end{bmatrix}^{(i)}_{6\times1}, \quad \boldsymbol{\omega}_{ref}^{(i)} = \begin{bmatrix}\omega_{1ref}\\ \omega_{2ref}\\ \omega_{3ref}\\ \omega_{4ref}\end{bmatrix}^{(i)}_{4\times1}, \quad \mathbf{u}^{(i)} = \begin{bmatrix}u_1\\ u_2\\ u_3\\ u_4\end{bmatrix}^{(i)}_{4\times1}, \quad \omega_{\text{hover}} = \sqrt{\frac{mg}{4k_b}} = 912 rad/s$$

Subject to:

$$\mathbf{x} \in [\mathbf{x}_{min}, \mathbf{x}_{max}] \tag{3.31}$$

$$\mathbf{u} \in [\mathbf{u}_{min}, \mathbf{u}_{max}] \tag{3.32}$$

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) \text{ (quadcopter dynamics in E- frame)} \tag{3.33}$$

$$\mathbf{x}_0 = \mathbf{x}_0 \text{ (initial condition)} \tag{3.34}$$

After trying genetic algorithm it was found high gains on position and velocities gives good tracking, keeping some gain on R helps improve stability and minimize energy. Further a small gain on T (torque in B-frame matrix) helps in improving performance and reduces oscillations. Further fine tuning and tweaking the gains is done manually to ensure that the optimization is solved within Ts=0.08 seconds, we have these values:

Q=[90000000 , 90000000 , 90000000 , 90000000 90000000 , 90000000]

R=[5, 5 , 5 , 5] , T=0.0001, N= 12

Simulation Results:

This NMPC controller achieves the best possible performance in reaching mission endpoints starting from $(0, 0, 0)$, while adhering to the quadcopter's actuator constraints. The controller enables the drone to achieve 98% of the settling level within the optimal time $t^*_{M_i}$, hence our comparison will focus solely on the differences in energy consumption during this duration.

Table 3.5: Energy consumption and their deviations from optimal energy

Mission M_i	$t^*_{M_i}$ [sec]	$E^*_{M_i}$ [J]	E_{M_i} [J]	error (%)
$M_1 = M(x_{00}, y_{00}, z_{30})$	6.63	8717.20	8780.5	0.73
$M_2 = M(x_{10}, y_{00}, z_{20})$	5.08	6881.63	6925	0.63
$M_3 = M(x_{50}, y_{00}, z_{20})$	6.94	10064	10124	0.60
$M_4 = M(x_{10}, y_{25}, z_{15})$	5.41	7813.28	7841	0.36

$M_1 = M(x_{00}, y_{00}, z_{30})$

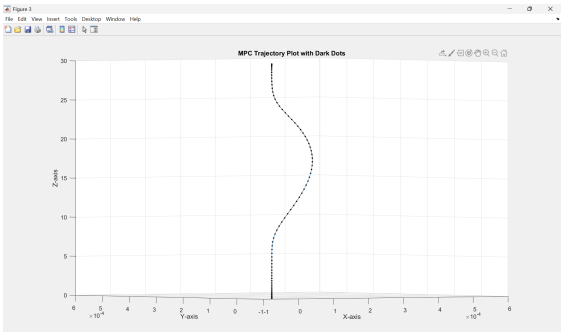

Figure 3.40: 3D trajectory tracking

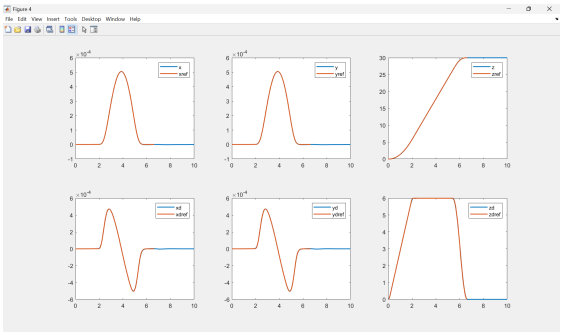

Figure 3.41: Position and velocities tracking

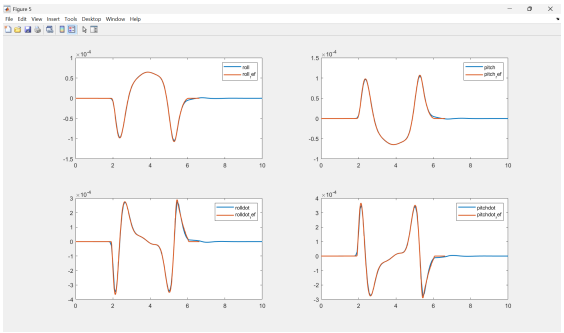

Figure 3.42: Attitude and its rates tracking

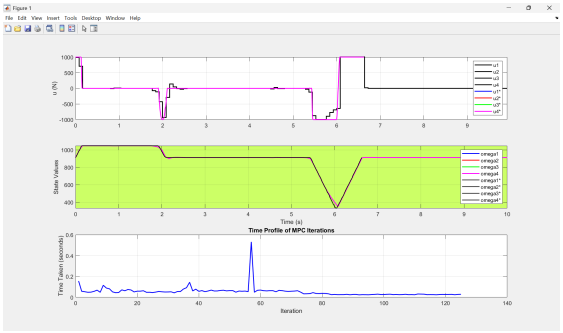

Figure 3.43: Rotor angular velocities, accelerations, and time profile

$M_2 = M(x_{10}, y_{00}, z_{20})$

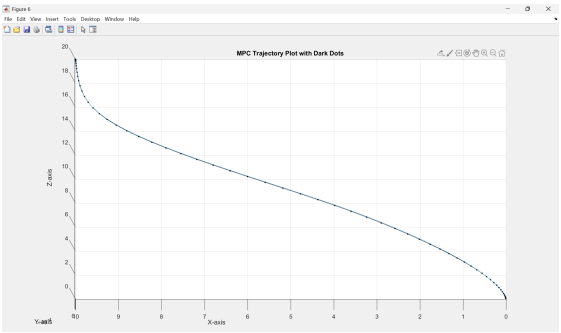

Figure 3.44: 3D trajectory tracking

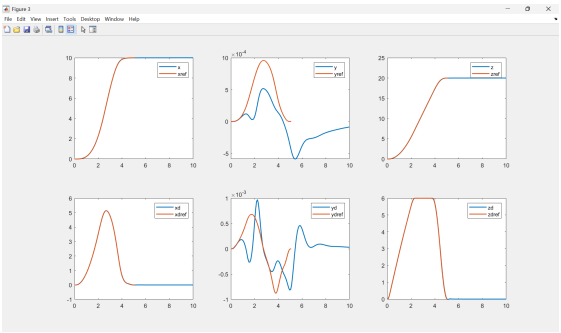

Figure 3.45: Position and velocities tracking

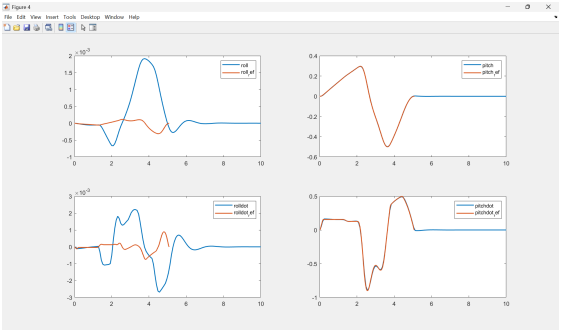

Figure 3.46: Attitude and its rates tracking

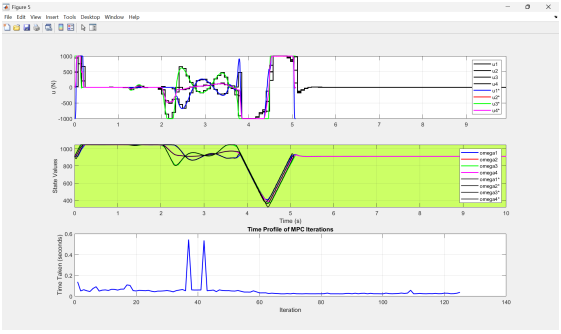

Figure 3.47: Rotor angular velocities, accelerations and time profile

$M_3 = M(x_{50}, y_{00}, z_{20})$

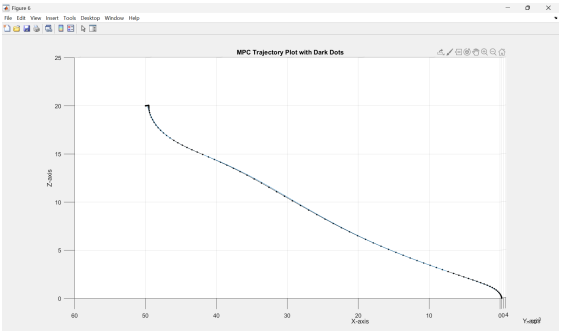

Figure 3.48: 3D trajectory tracking

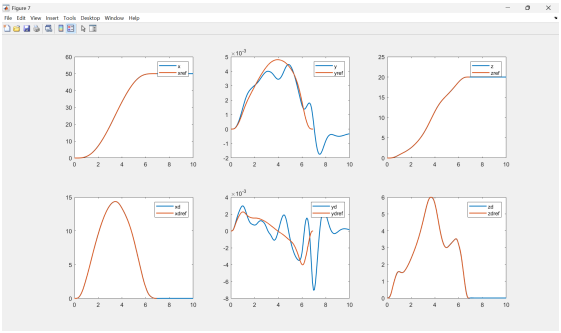

Figure 3.49: Position and velocities tracking

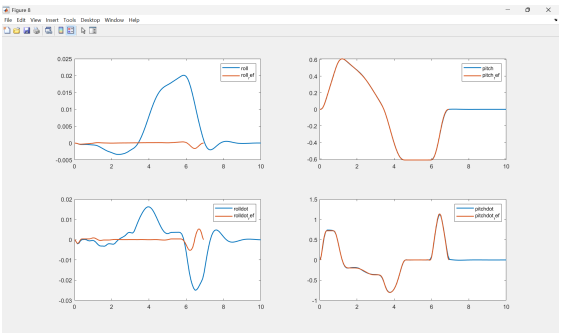

Figure 3.50: Attitude and its rates tracking

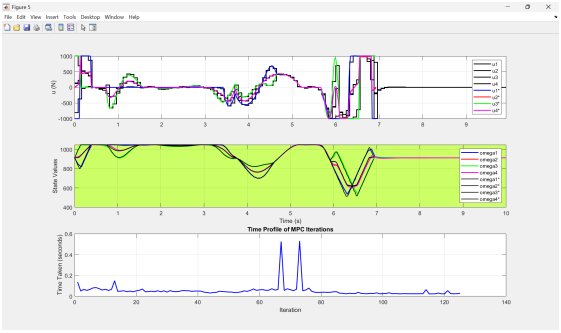

Figure 3.51: Rotor angular velocities, accelerations and time profile

$M_4 = M(x_{10}, y_{25}, z_{15})$

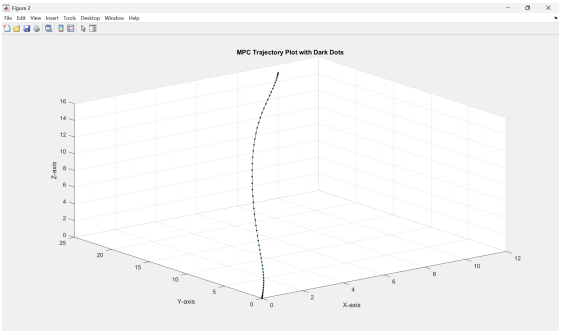

Figure 3.52: 3D trajectory tracking

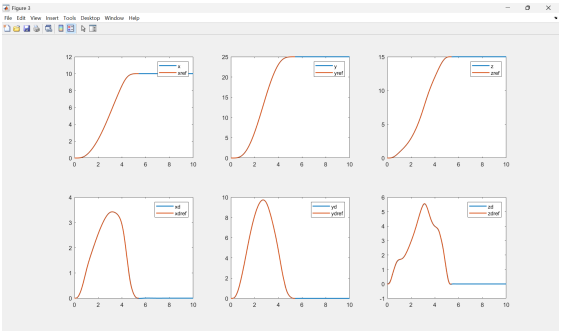

Figure 3.53: Position and velocities tracking

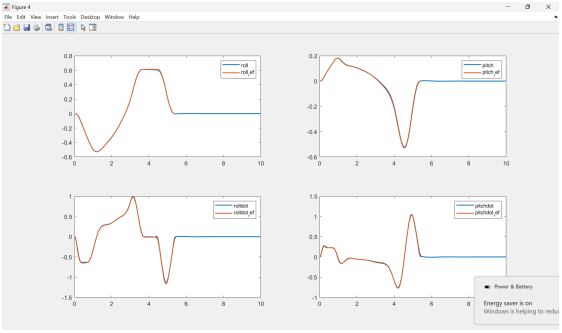

Figure 3.54: Attitude and its rates tracking

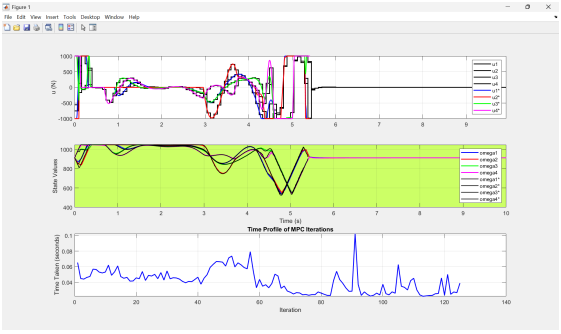

Figure 3.55: Rotor angular velocities, accelerations and time profile

REMARKS:

It does satisfactory tracking with slightly high energy consumption than optimal energy (E* joules) in optimal time(tf* seconds),and manages to reach endpoint(98 % setlling band) within tf* seconds .

This controller is giving both excellent tracking and good stabilization , and is clearly evident when the simulation runs for longer time.

In this formulation also, we dont use open-loop attitude references, thus giving the controller some extra degree of freedom for finding optimal attitude . hence its more robust to disturbances. For instance, starting from $x_0 = [1,\ 1, 0]$ takes nearly $t_f = 6$ seconds to reach the 98% settling band.

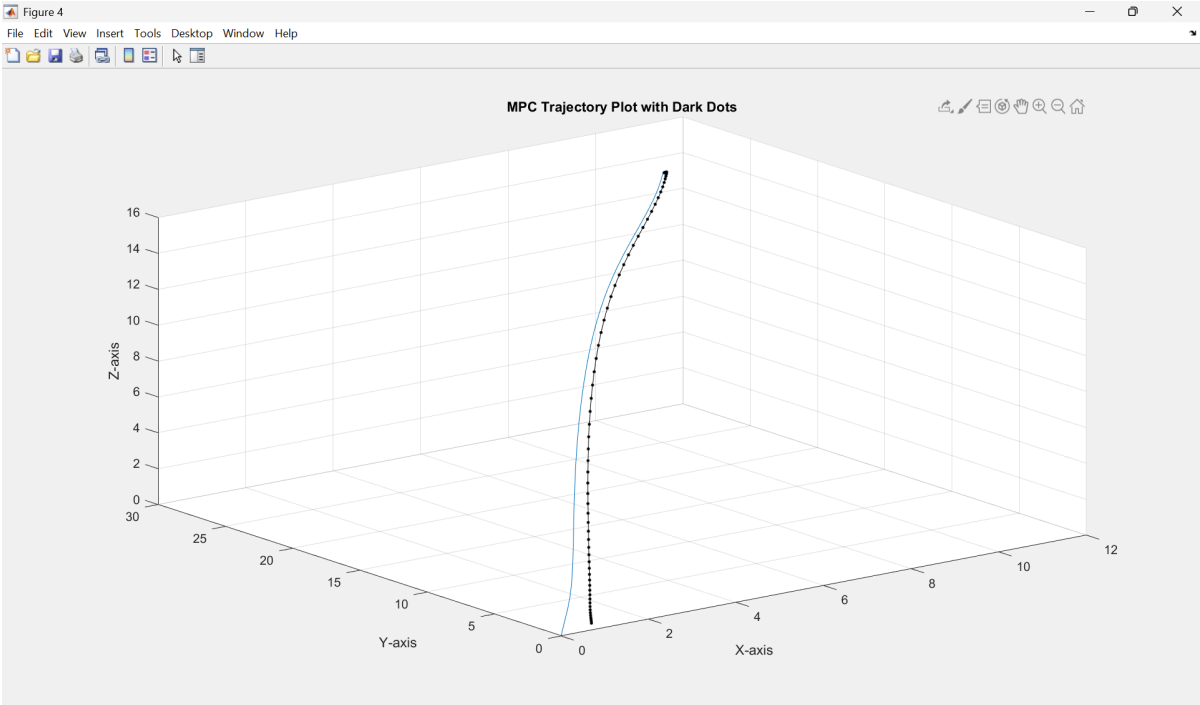

Figure 3.56: satisfactory tracking in presence of disturbance

SOME OTHER 3D MISSIONS:

$M_5 = M(x_{30}, y_5, z_{40})$

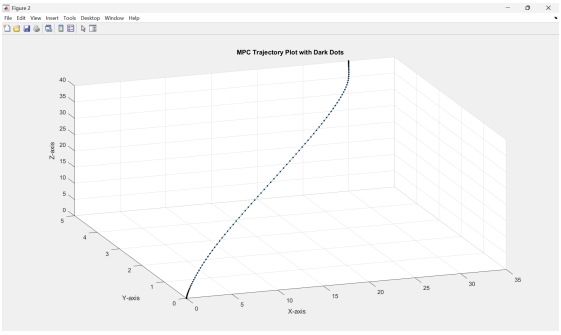

Figure 3.57: 3D trajectory tracking

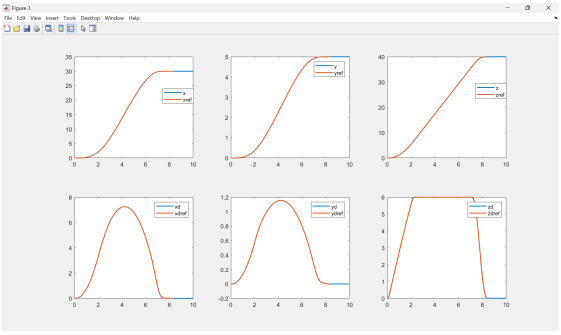

Figure 3.58: Position and velocities tracking

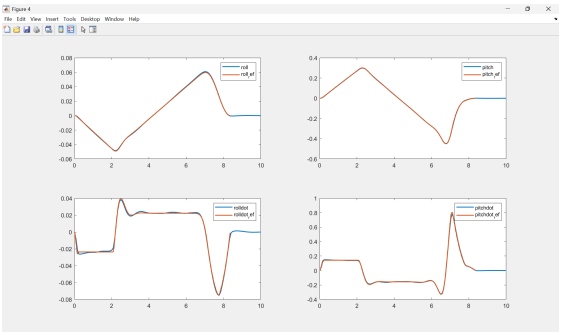

Figure 3.59: Attitude and its rates tracking

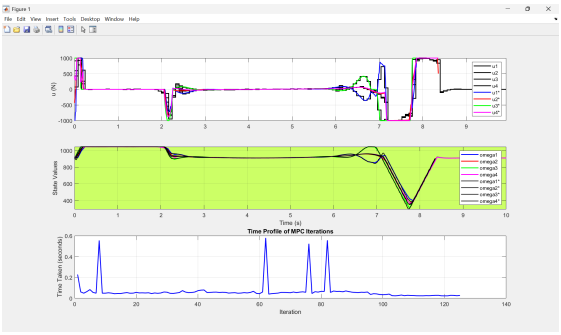

Figure 3.60: Rotor angular velocities, accelerations and time profile

$$M_6 = M(x_{10}, y_{20}, z_{30})$$

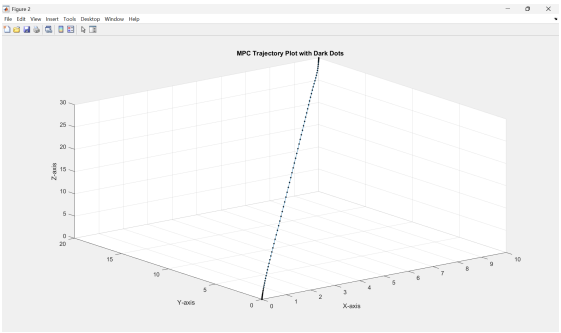

Figure 3.61: 3D trajectory tracking

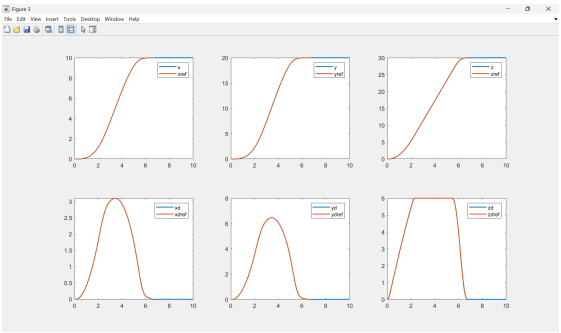

Figure 3.62: Position and velocities tracking

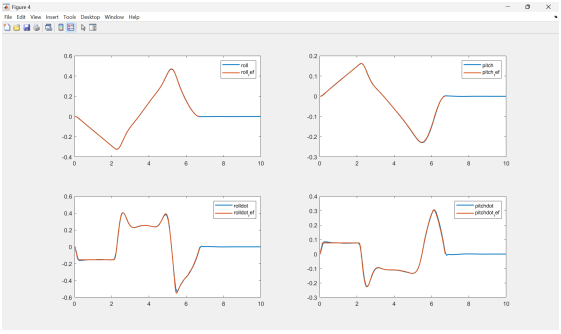

Figure 3.63: Attitude and its rates tracking

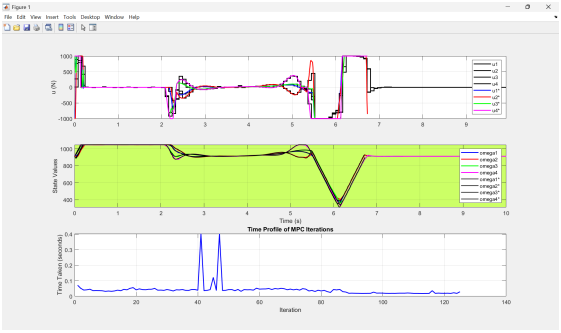

Figure 3.64: Rotor angular velocities, accelerations and time profile

$M_7 = M(x_{30}, y_{20}, z_{10})$

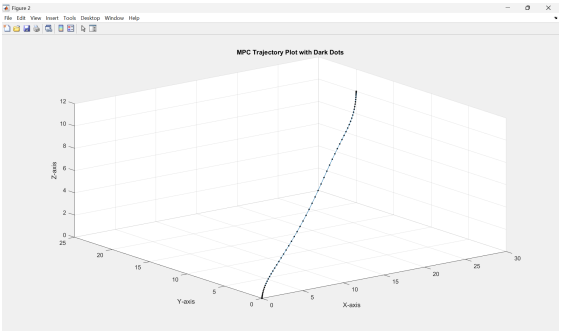

Figure 3.65: 3D trajectory tracking

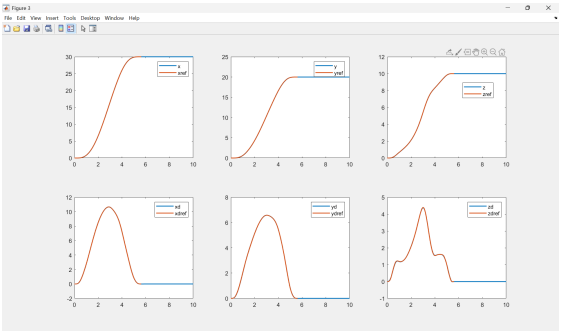

Figure 3.66: Position and velocities tracking

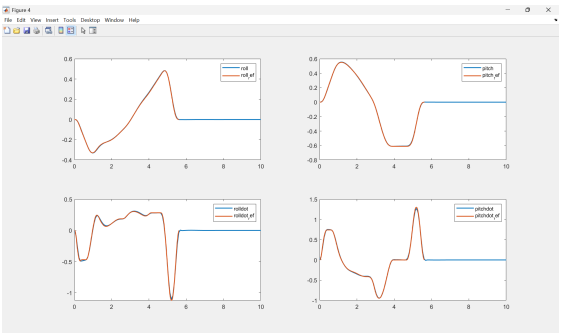

Figure 3.67: Attitude and its rates tracking

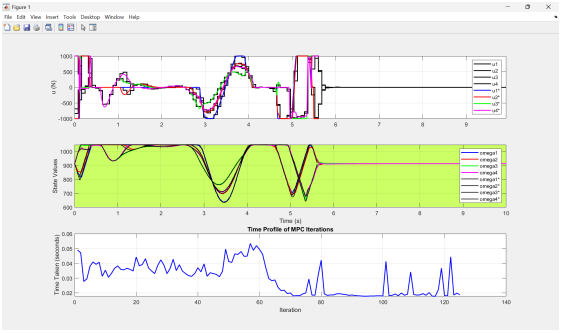

Figure 3.68: Rotor angular velocities, accelerations and time profile

$M_8 = M(x_{20}, y_{50}, z_0)$

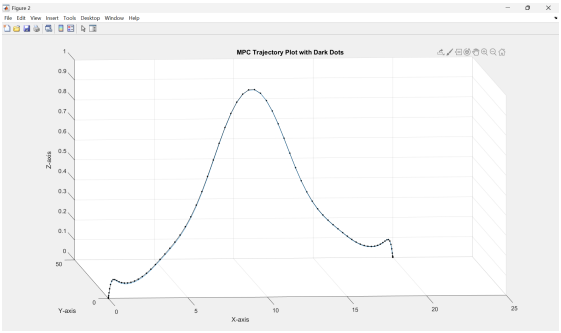

Figure 3.69: 3D trajectory tracking

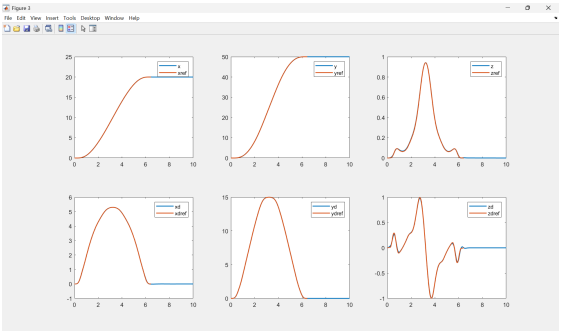

Figure 3.70: Position and velocities tracking

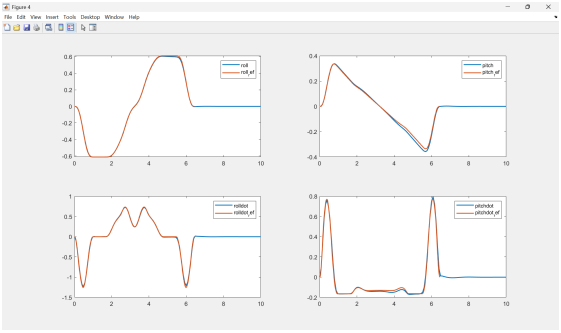

Figure 3.71: Attitude and its rates tracking

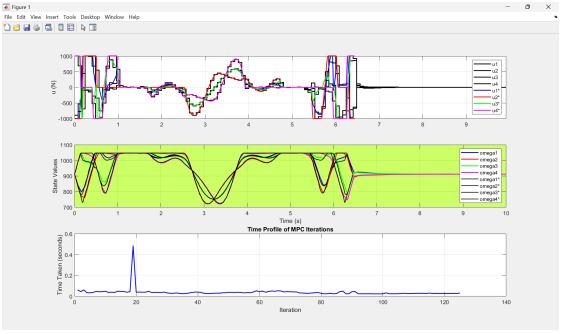

Figure 3.72: Rotor angular velocities, accelerations and time profile

Again a table of comparison for these 3D missions is shown below based on the differences in energy consumption during the optimal time duration ($t^*_{M_i}$).

Table 3.6: Energy consumption and their deviations from optimal energy

Mission M_i	$t^*_{M_i}$ [sec]	$E^*_{M_i}$ [J]	E_{M_i} [J]	error (%)
$M_5 = M(x_{30}, y_5, z_{40})$	8.36	11235	11217	- 0.16
$M_6 = M(x_{10}, y_{20}, z_{30})$	6.73	9144	9204	0.66
$M_7 = M(x_{30}, y_{20}, z_{10})$	5.62	8310	8353	0.52
$M_8 = M(x_{20}, y_{50}, z_0)$	6.44	9667	9714	0.49

3.6 Genetic Algorithm-Based Controller Tuning

We saw Nonlinear Model Predictive Control (NMPC) is a powerful strategy for controlling complex systems like quadrotors, but its performance hinges on tuning the state weighting matrix $\mathbf{Q}$, control weighting matrix $\mathbf{R}$, and prediction horizon N. A Genetic Algorithm (GA) was employed to optimize these parameters, balancing tracking accuracy and energy efficiency in a closed-loop NMPC framework.

The GA is a bio-inspired optimization method that evolves a population of candidate solutions through selection, crossover, mutation, and elitism. Each individual is represented as a tuple $(\mathbf{Q}, \mathbf{R}, N)$, where $\mathbf{Q} = \text{diag}(q_1, \ldots, q_n)$ weights state errors, $\mathbf{R} = \text{diag}(r_1, \ldots, r_m)$ weights control efforts, and N defines the prediction horizon length. The GA processes are summarized as follows:

- **Selection**: Chooses fitter individuals based on a fitness function using tournament selection.
- **Elitism**: Preserves the top-performing individuals across generations to retain optimal solutions.
- **Crossover**: Combines parameters from two parents to create offspring, promoting solution diversity.
- **Mutation**: Introduces random changes to parameters to explore new solutions and avoid local optima.

3.7 CONTROLLER TUNING PIPELINE

At first, the GA selects an individual$(\mathbf{Q}, \mathbf{R}, N)$ to the quadrotor NMPC controller. For each individual $(\mathbf{Q}, \mathbf{R}, N)$, a closed-loop simulation is run using the NMPC cost function as shown in the below figure:

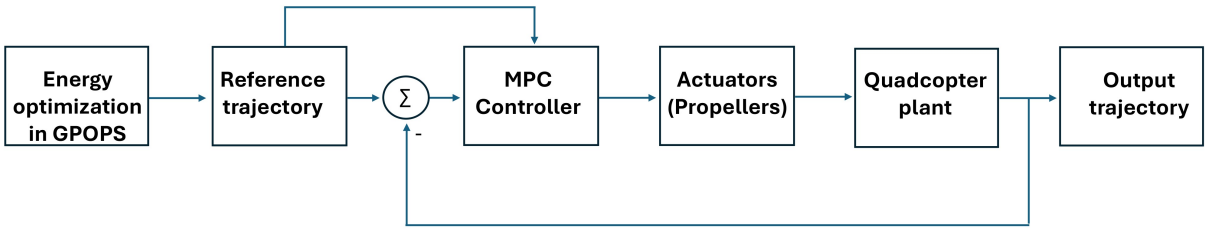

Figure 3.73: NMPC controler with quadcopter in closed loop

Thereafter , the GA calculates the fitness function for that individual$(\mathbf{Q}, \mathbf{R}, N)$.

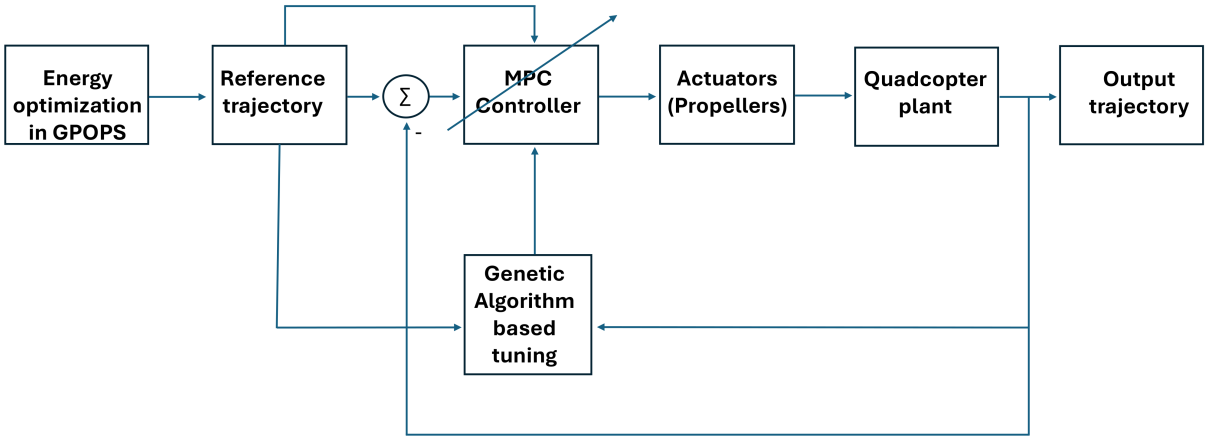

Figure 3.74: NMPC controller with quadcopter in closed loop along with the GA tuner

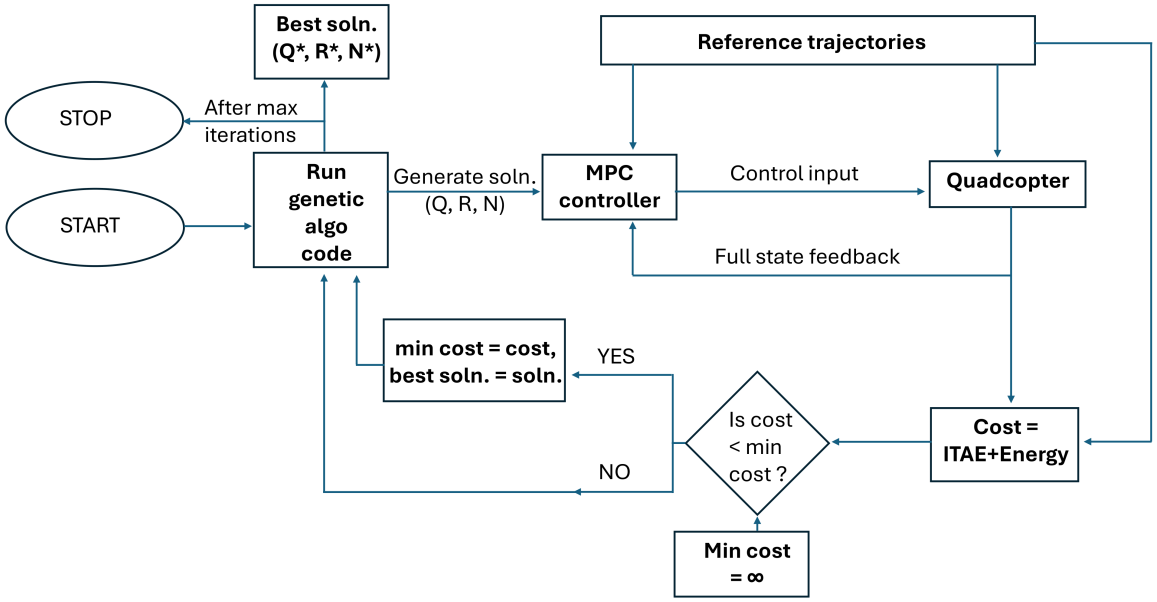

Figure 3.75: Controller tuning pipeline

$$
\begin{aligned}
[\mathbf{Q}^*, \mathbf{R}^*, \mathbf{N}^*] = & \textbf{For } i = 1 \textbf{ to } \text{population\_size} \times \text{max\_generations} \\
& \arg\min \left[\text{fitness\_function}\left(\mathbf{X}_{\text{ref}}, \mathbf{X}(\mathbf{i}), \mathbf{U}(\mathbf{i})\right)\right] \\
& \textbf{such that} \\
& \quad \textbf{For } t = 0 \textbf{ to } t_{\text{simulation}} \\
& \quad\quad \mathbf{X}(\mathbf{i}, \mathbf{t}), \mathbf{U}(\mathbf{i}, \mathbf{t}) \leftarrow \arg\min \left[\text{MPC\_objective\_function}\left(\mathbf{Q}, \mathbf{R}, \mathbf{N}, \mathbf{X}_{\text{ref}}(\mathbf{t}), \mathbf{X}(\mathbf{i}, \mathbf{t}), \mathbf{U}(\mathbf{i}, \mathbf{t})\right)\right] \\
& \quad\quad \textbf{subject to:} \\
& \quad\quad\quad \dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) \quad (\textbf{quadcopter dynamics in E-frame}) \\
& \quad\quad\quad \mathbf{x}(0) = \mathbf{x_0} \quad (\textbf{initial condition}) \\
& \quad\quad\quad \mathbf{X} \in [\mathbf{X}_{\min}, \mathbf{X}_{\max}] \\
& \quad\quad\quad \mathbf{U} \in [\mathbf{U}_{\min}, \mathbf{U}_{\max}] \\
& \quad\quad\quad \mathbf{Q} \in [\mathbf{Q}_{\min}, \mathbf{Q}_{\max}] \\
& \quad\quad\quad \mathbf{R} \in [\mathbf{R}_{\min}, \mathbf{R}_{\max}] \\
& \quad\quad\quad \mathbf{N} \in [\mathbf{N}_{\min}, \mathbf{N}_{\max}], \quad \mathbf{N} \in \mathbb{I} \\
& \quad\quad \textbf{end} \\
& \quad \textbf{end}
\end{aligned}
$$

The fitness function is a weighted sum of the Integral of Time-weighted Absolute Error (ITAE), $\int_0^T t\|\mathbf{x}_{\text{ref}}(t)-\mathbf{x}(t)\|\,dt$, and the energy consumption function. Closed-loop simulations evaluate each individual's performance, and the GA iteratively minimizes the fitness function $f = w_1 \cdot \text{ITAE} + w_2 \cdot \text{Energy}$, yielding optimal $\mathbf{Q}$, $\mathbf{R}$, and N for precise tracking and energy-efficient control.

Algorithm 1 Genetic Algorithm for NMPC Controller Tuning

Initialize population with random $(\mathbf{Q}_i, \mathbf{R}_i, N_i)$, where $\mathbf{Q}_i = \text{diag}(q_1, \ldots, q_n)$, $\mathbf{R}_i = \text{diag}(r_1, \ldots, r_m)$, and $N_i \in [N_{\min}, N_{\max}]$.
for generation = 1 to `max_generations` **do**
    for each individual $(\mathbf{Q}_i, \mathbf{R}_i, N_i)$ **do**
        Run closed-loop NMPC simulation with cost function $J(i)$.
        Compute fitness: $f_i = w_1 \cdot \int_0^T t\|\mathbf{x}_{\text{ref}}(t) - \mathbf{x}(t)\|\,dt + w_2 \cdot \int_0^T \|\mathbf{u}(t)\|^2\,dt$.
    end for
    Elitism: Copy top 5% individuals to the next generation.
    while next generation not full **do**
        Select parents via tournament selection.
        Apply crossover (arithmetic blend, probability $p_c = 0.8$).
        Apply mutation (Gaussian noise, probability $p_m = 0.05$).
        Add offspring to new population.
    end while
end for
return optimal $(\mathbf{Q}^*, \mathbf{R}^*, N^*)$.

Chapter 4

Conclusion

The work presented in this report has done significant exploration in the area of energy optimal trajectory tracking to complete a given mission.And for our drone DJI Phantom2.Its giving excellent performance in reaching mission within optimal time while tracking the trajectory alongside maintaining low energy consumption .

This work also addresses robustness of the NMPC controller to disturbances like when DRONE isnt aligned with the refrernce trajectory at the begining which is equivalent to wind gusts displacing the drone . Our NMPC controller could sucessfully calculate the model's predictions and merge with our refernce trajectory with passage of time.

There is lot of room for further exploration in this area of least energy tracking. We assumed full state feedback was availabe. So future work could be to integrate a Kalman filter into the closed loop to account for sensor noise—affecting measured states for real world implementation.In contrast to traditional NMPC controller for drones where we get the thrust and torques as outputs of mpc controller, in our case we get rotor angular accelerations as output of the NMPC controller ,which implies by intregrating them we can obtain 4 reference omegas for the 4 BLDC motors of the drone. hence for implementing our controller in real life means we would require very good speed control to track these continuously varying speeds.

Lastly with these points , testing the entire control strategy on a real UAV would provide valuable insights into its practical feasibility and performance. MATLAB code could be converted to C/C++ and implemented on an actual drone to evaluate the control system's trajectory-tracking capabilities under real-world conditions.

Chapter 5

Bibliography

[1] N. Michel, S. Bertrand, S. Olaru, G. Valmorbida, and D. Dumur, "Design and flight experiments of a tube-based model predictive controller for the ar. drone 2.0 quadrotor," *IFAC-PapersOnLine*, vol. 52, no. 22, pp. 112–117, 2019.

[2] Y. Lyu, J. Hu, B. M. Chen, C. Zhao, and Q. Pan, "Multivehicle flocking with collision avoidance via distributed model predictive control," *IEEE transactions on cybernetics*, vol. 51, no. 5, pp. 2651–2662, 2019.

[3] M. Islam, M. Okasha, and M. Idres, "Dynamics and control of quadcopter using linear model predictive control approach," in *IOP conference series: materials science and engineering*, vol. 270, p. 012007, IOP Publishing, 2017.

[4] M. Elhesasy, T. N. Dief, M. Atallah, M. Okasha, M. M. Kamra, S. Yoshida, and M. A. Rushdi, "Non-linear model predictive control using CASADI package for trajectory tracking of quadrotor," *Energies*, vol. 16, no. 5, p. 2143, 2023.

[5] D. Wang, Q. Pan, Y. Shi, J. Hu, and C. Zhao, "Efficient nonlinear model predictive control for quadrotor trajectory tracking: Algorithms and experiment," *IEEE Transactions on Cybernetics*, vol. 51, no. 10, pp. 5057–5068, 2021.

[6] M. Jacewicz, M. Żugaj, R. Głebocki, and P. Bibik, "Quadrotor model for energy consumption analysis," *Energies*, vol. 15, no. 19, p. 7136, 2022.

[7] J. A. Robbins, A. F. Thompson, S. Brennan, and H. C. Pangborn, "Energy-aware predictive motion planning for autonomous vehicles using a hybrid zonotope constraint representation," *arXiv preprint arXiv:2411.03189*, 2024.

[8] F. Morbidi, R. Cano, and D. Lara, "Minimum-energy path generation for a quadrotor UAV," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1492–1498, IEEE, 2016.

[9] C. Trapiello Fernández, "Control of an UAV using LPV techniques," Master's thesis, Universitat Politècnica de Catalunya, 2018.