aerospace

MDPI

Article

Design and Experimental Comparison of PID, LQR and MPC Stabilizing Controllers for Parrot Mambo Mini-Drone

Mohamed Okasha [1,*], Jordan Kralev [2,*] and Maidul Islam [3]

[1] Department of Mechanical and Aerospace Engineering, College of Engineering, United Arab Emirates University, Al Ain P.O. Box 15551, United Arab Emirates
[2] Department of Systems and Control, Faculty of Automatics, Technical University of Sofia, 1700 Sofia, Bulgaria
[3] Department of Mechanical Engineering, Kuliyyah of Engineering, International Islamic University Malaysia, Kuala Lumpur 53100, Malaysia; mislam.dipu@gmail.com
* Correspondence: mokasha@uaeu.ac.ae (M.O.); jkralev@ieee.org (J.K.)

Abstract: Parrot Mambo mini-drone is a readily available commercial quadrotor platform to understand and analyze the behavior of a quadrotor both in indoor and outdoor applications. This study evaluates the performance of three alternative controllers on a Parrot Mambo mini-drone in an interior environment, including Proportional–Integral–Derivative (PID), Linear Quadratic Regulator (LQR), and Model Predictive Control (MPC). To investigate the controllers' performance, initially, the MATLAB® /Simulink™ environment was considered as the simulation platform. The successful simulation results finally led to the implementation of the controllers in real-time in the Parrot Mambo mini-drone. Here, MPC surpasses PID and LQR in ensuring the system's stability and robustness in simulation and real-time experiment results. Thus, this work makes a contribution by introducing the impact of MPC on this quadrotor platform, such as system stability and robustness, and showing its efficacy over PID and LQR. All three controllers demonstrate similar tracking performance in simulations and experiments. In steady state, the maximal pitch deviation for the PID controller is 0.075 rad, for the LQR, it is 0.025 rad, and for the MPC, it is 0.04 rad. The maximum pitch deviation for the PID-based controller is 0.3 rad after the take-off impulse, 0.06 rad for the LQR, and 0.17 rad for the MPC.

Keywords: parrot mini-drone control; PID; LQR; MPC; trajectory tracking; flight test

Citation: Okasha, M.; Kralev, J.; Islam, M. Design and Experimental Comparison of PID, LQR and MPC Stabilizing Controllers for Parrot Mambo Mini-Drone. *Aerospace* **2022**, *9*, 298. https://doi.org/10.3390/aerospace9060298

Academic Editor: Sergey Leonov

Received: 8 April 2022
Accepted: 30 May 2022
Published: 1 June 2022

1. Introduction

The improvement of technology in Micro Electronics Mechanical Systems (MEMS), particularly in sensors and microcontrollers, encourages researchers to work on multi-rotors. They are in high demand, as evidenced by a large number of research studies. It has features such as simplicity to build easier maneuverability but also myriad applications that are sometimes not easy to perform for a man and, in extreme cases, are out of a human's ability [1]. For example, surveillance, package delivery, mapping, exploration, smart agriculture [2], etc., are some applications that are used to be considered futuristic projects for human beings two decades ago, while these are not wondrous anymore rather than the reality. Interestingly, since all these applications primarily deal with trajectory tracking, the robotic community is continuously working on improving multi-rotors tracking performance to ensure smooth and collision-free flights [3].

The quadrotor is one of the most widely chosen multi-rotors among researchers because of its being compact in size and light in weight. It has a cross-configured structure with two pairs of rotors facing opposite directions. It takes flight by balancing the thrust and torque produced by the rotors. Roll, pitch, yaw, and upward–downward push are the necessary operations for quadrotor movement.

The numerous applications of quadrotors can be divided into civilian and military categories, attracting academics' interest in learning more about them. Aerial photography,

inspections of industrial pipelines [4], traffic monitoring, crop monitoring, fire detection, rescue operations, weather forecasting, news coverage, and other civilian applications are available, while military applications include border patrol, surveillance, and warfare [4]. It is worth mentioning that a quadrotor is normally not fit for carrying heavy objects since it has only four rotors that are not able to generate a sufficient amount of thrust. Thus, in those cases, hexacopters or octocopters receive the interest of the researchers.

A quadrotor is designed using fixed-pitch rotor blades that offer Vertical Take-Off Landing (VTOL), and therefore, it does not require a runway. The quadrotor is an inherently unstable and under-actuated plant. The quadrotor frame is represented as a rigid body with three translational and three rotational degrees of freedom (DoF). When the quadrotor is used for outdoor applications, additional challenging operational requirements emerge, such as high agility, smooth maneuverability, disturbance rejection, payload variation, etc. Hence, different control strategies have been introduced in control engineering that can ensure promising performance during its operations.

In order to improve quadrotor tracking performance while taking environmental factors into account, studies have introduced controllers such as Proportional–Integral–Derivative (PID), Linear Quadratic Regulator (LQR), Backstepping, Feedback Linearization Control (FLC), Sliding Mode Control (SMC), Model Predictive Control (MPC), Linear Quadratic Gaussian (LQG), neural network, H-infinity, fuzzy logic, adaptive control, etc. These controllers are divided into three categories: linear, nonlinear, and learning-based controllers [5–8]. Interestingly, numerous works on linear control systems on quadrotor platforms are available since they are easy to design and employ [9]. In addition, several studies have introduced another type of controller, hybrid controllers [10], that include two or more different control strategies together to improve different aspects of the performance of the quadrotor.

Parrot mini-drone is a widely known commercial quadrotor platform for educational purposes, particularly to learn the quadrotor's behavior in both indoor and outdoor environments. This platform is equipped with some sensors such as an ultrasonic sensor, an altimeter, an inertial measurement unit (IMU), and a camera that can describe its behaviors in different environments. It has received popularity because of its simplicity and ready-made platform at a reasonable price. Moreover, its Simulink software package is officially available, and hence, researchers can work on low-level control to test the performance in practice, which can lead to enhancing its tracking performance further. As a result, several works on controllers have been found on this platform. For example, Everett works on both LQR and LQI controller and conclude that LQI ensures better flight than regular LQR since LQI can reduce the steady-state error [11]. Glazkov and Golubev introduce an adaptive control to observe the tracking performance of the mini-drone and receive a satisfactory performance in terms of following the reference trajectory [12]. Castañeda and Gordillo adopt adaptive sliding mode control, where the adaptive approach is considered primarily to reduce the chattering problem of SMC. Thus, the platform achieves stability even in the presence of external disturbance and, at the same time, manages to follow the trajectory [13,14].

MPC, a nonlinear controller, has achieved popularity because of its robustness in tracking through dealing with noise, disturbance, and model uncertainty in quadrotor platforms. MPC and decentralized MPC are not so common for solving drone stabilization problems, and their main application is for flock optimal trajectory planning [3]. While flight control is responsible for basic self-stabilization and maneuver, flocking coordination is responsible for inter-drone synchronization.

In this study, this platform is considered to compare the tracking performances of three different controllers, such as PID, LQR, and MPC. The main contribution of this work is to introduce MPC on the parrot mini-drone in tracking the trajectory and evaluate its tracking performance with the other controller on this platform.

2. Parrot Mini-Drone

The Parrot mini-drone is a palm-sized quadrotor that can be controlled with a smartphone or a computer, making it very popular among researchers. It is a quadrotor of 55 g in weight and with three axes gyroscope, three axes accelerometer, a pressure sensor, and a 0.3-megapixel downward-facing camera of 60 fps. A 550 mAh Lithium-Polymer battery ensures a 6–8 min flight duration. Similar to regular quadrotors, four independent BLDC rotors with fixed pitched propellers are mounted at the four corners of this platform. Two of the four propellers rotate clockwise direction, while the other two rotate counterclockwise, as indicated in Figure 1.

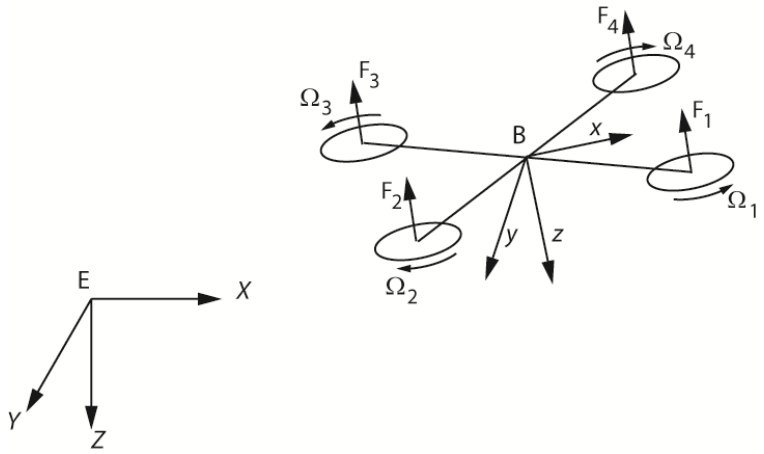

Figure 1. Configuration of the quadrotor. $E(X,Y,Z)$—Earth frame; $B(x,y,z)$—body fixed frame; Ω_i—rotor angular velocity; F_i—rotor lift force.

The dynamics of a quadrotor are generally described considering two frames of reference—an Earth-fixed frame (denoted with E) and a body frame (denoted with B)—as shown in Figure 1. We assume that the origin of the body-fixed reference frame is at the quadcopter's center of gravity, where the inertial measurement unit (IMU) is situated. Since the quadrotor frame is accelerating and rotating during its course of motion, the body-fixed reference frame is non-inertial. The Earth-fixed reference frame is assumed to behave as an inertial. The consideration of the Earth frame is required for the specification of the quadcopter's geographic position and of the gravitational force direction. The Body-fixed reference frame hosts the rotor thrust forces and the angular velocities of the quadrotor.

2.1. Kinematics

The kinematics of a model establishes the relationship between the motion of the body-fixed reference frame (B) and the axes of the Earth-fixed reference frame (E). Hence, to explain the quadrotor's kinematics, a transformation matrix is necessarily considered that can map any vector from the body frame into the Earth frame. As a result, a transformation matrix, R_{EB}, is introduced, as illustrated in Equation (1), which is later used to express the Newton-Euler equations.

$$R_{EB} = R_z(\psi)R_y(\theta)R_x(\phi) = \begin{pmatrix} c_\theta c_\psi & -c_\phi s_\psi - s_\phi s_\theta c_\psi & -c_\phi s_\theta c_\psi + s_\phi s_\psi \\ c_\theta s_\psi & -s_\phi s_\theta s_\psi + c_\phi c_\psi & -c_\phi s_\theta s_\psi - s_\phi c_\psi \\ s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{pmatrix}, \quad (1)$$

where, c_α, s_α and t_α denote $\cos(\alpha)$, $\sin(\alpha)$, and $\tan(\alpha)$ trigonometric functions for some arbitrary angle α, and R_x, R_y, R_z denote rotation matrices around body frame principal axes. In addition, another transformation matrix R_r is required to describe the relationship between Euler rates, $\dot{\eta} = \left[\dot{\phi},\ \dot{\theta},\ \dot{\psi}\right]^T$, and the measured angular velocity vector of the quadrotor, $\omega = [p,\ q,\ r]^T$, as follows [15,16]

$$\eta = \begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} = \begin{pmatrix} 1 & s_\phi t_\theta & c_\phi t_\theta \\ 0 & c_\phi & -s_\phi \\ 0 & \frac{s_\phi}{c_\theta} & \frac{c_\phi}{c_\theta} \end{pmatrix} \begin{pmatrix} p \\ q \\ r \end{pmatrix} = R_r \omega. \tag{2}$$

2.2. Dynamics

Quadrotor's motion can be decomposed into a translational motion and a rotational motion. The orientation of the quadcopter in space is represented by the roll, pitch, and yaw rotations. When the drone is airborne and if its frame is tilted slightly, then translational motions along the X and Y axes arise, which are measured in the unit of meters as x, and y, respectively. Apart from that, the ascending or descending motion of the drone can be achieved by increasing or decreasing the average rotor speed [17].

To facilitate stabilizing controller design, typically, an actuator decoupling matrix is selected depending on the frame configuration. The matrix in (3) for the Parrot Mambo 'X' drone frame maps the four squared speed references Ω_1^2, Ω_2^2, Ω_3^2 and Ω_4^2 To the lift force and stabilizing torques as

$$\begin{pmatrix} F_z \\ L \\ M \\ N \end{pmatrix} = \begin{pmatrix} -k_F & -k_F & -k_F & -k_F \\ -\frac{1}{\sqrt{2}} l k_F & -\frac{1}{\sqrt{2}} l k_F & \frac{1}{\sqrt{2}} l k_F & \frac{1}{\sqrt{2}} l k_F \\ \frac{1}{\sqrt{2}} l k_F & -\frac{1}{\sqrt{2}} l k_F & -\frac{1}{\sqrt{2}} l k_F & \frac{1}{\sqrt{2}} l k_F \\ k_M & -k_M & k_M & -k_M \end{pmatrix} \begin{pmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{pmatrix}, \tag{3}$$

where F_z is the lift force along the body z dimension, L is the roll torque, M is the pitch torque, and N is the yaw torque. The control system can calculate the squares of the speed references with the help of a decoupling matrix, which is an inverse to the map from Equation (3), such that the decoupled manipulated variables $U_{1,d}$, $U_{2,d}$, $U_{3,d}$, $U_{4,d}$ to correspond to the lift force and stabilizing torques. Therefore, in this case, $U_{2,d} \approx L$, $U_{3,d} \approx M$, and $U_{4,d} \approx N$, will be responsible for the roll, pitch, and yaw movements, respectively, whereas, $U_1 \approx F_z$ will be responsible for the upward or downward movement along the z axis. The decoupling quality will depend on the symmetries between motor responses, blade geometries, and frame rigidity. Alternatively, since the mapping (3) is linear, the control system may drive the motor references directly by setting $U_i = \Omega_i$, $i = 1 \ldots 4$. The relation between decoupled, $U_{d,i}$ and direct, U_i, manipulated variables is the inverse of (3) and becomes

$$\begin{pmatrix} U_1 \\ U_1 \\ U_2 \\ U_3 \end{pmatrix} = \begin{pmatrix} -\frac{1}{4k_F} & -\frac{1}{2\sqrt{2}lk_F} & \frac{1}{2\sqrt{2}lk_F} & \frac{1}{4k_M} \\ -\frac{1}{4k_F} & -\frac{1}{2\sqrt{2}lk_F} & -\frac{1}{2\sqrt{2}lk_F} & -\frac{1}{4k_M} \\ -\frac{1}{4k_F} & \frac{1}{2\sqrt{2}lk_F} & -\frac{1}{2\sqrt{2}lk_F} & \frac{1}{4k_M} \\ -\frac{1}{4k_F} & \frac{1}{2\sqrt{2}lk_F} & \frac{1}{2\sqrt{2}lk_F} & -\frac{1}{4k_M} \end{pmatrix} \begin{pmatrix} U_{1,d} \\ U_{2,d} \\ U_{3,d} \\ U_{4,d} \end{pmatrix} \tag{4}$$

Table 1 shows the inertial parameters of the quadrotor.

Table 1. Parrot mini-drone inertial parameters [18].

Parameter	Value	Unit
Mass, m	0.068	kg
Length of an arm, l	0.062	m
Gravity, g	9.81	m/s^2
Moment of inertia along x axis, I_x	6.86×10^{-5}	kgm^2
Moment of inertia along y axis, I_y	9.2×10^{-5}	kgm^2
Moment of inertia along z axis, I_z	1.366×10^{-4}	kgm^2
Thrust coefficient, k_F	0.01	$N/(rad^2/s^2)$
Thrust coefficient, k_M	7.8263×10^{-4}	$Nm/(rad^2/s^2)$

Notably, the quadrotor dynamics can be described considering the Newton-Euler equation that deals with both forces and moments on the quadrotor. Here, the Equation (5) describes the translational motion of the quadrotor, while the Equation (6) describes the rotational motion of the quadrotor:

$$m\begin{pmatrix}\ddot{x}\\ \ddot{y}\\ \ddot{z}\end{pmatrix} = \begin{pmatrix}0\\ 0\\ mg\end{pmatrix} + R_{EB}\begin{pmatrix}0\\ 0\\ F_z\end{pmatrix}, \tag{5}$$

$$\begin{bmatrix}L\\ M\\ N\end{bmatrix} = I\dot{\omega} + \omega \times I\omega, \tag{6}$$

where $I = diag(I_x, I_y, I_z)$. From Equations (1), (2), (5), and (6), the complete mathematical model of the quadcopter can be derived as follows:

$$\ddot{x} = \frac{1}{m}\left[F_z\left(c_\phi c_\psi s_\theta + s_\phi s_\psi\right)\right], \tag{7}$$

$$\ddot{y} = \frac{1}{m}\left[F_z\left(s_\theta c_\phi s_\psi - s_\phi c_\psi\right)\right], \tag{8}$$

$$\ddot{z} = g + \frac{F_z}{m}\left(c_\theta c_\phi\right), \tag{9}$$

$$\dot{\phi} = p + r \cdot c_\phi t_\theta + q \cdot s_\phi t_\theta, \tag{10}$$

$$\dot{\theta} = q \cdot c_\phi - r \cdot s_\phi, \tag{11}$$

$$\dot{\psi} = r \cdot \frac{c_\phi}{t_\theta} + q \cdot \frac{s_\phi}{c_\theta}, \tag{12}$$

$$\dot{p} = \frac{1}{I_x}\left(L + I_y \cdot qr - I_z \cdot qr\right), \tag{13}$$

$$\dot{q} = \frac{1}{I_y}(M - I_x \cdot pr + I_x \cdot pr), \tag{14}$$

$$\dot{r} = \frac{1}{I_z}\left(N - I_x \cdot pq + I_y \cdot pq\right). \tag{15}$$

Finally, the dynamical model of the quadcopter can be described by four control inputs, $(F_z, L, M, N)^T$ and by twelve state variables, grouped in a state vector $x_s = \left(x,\ y,\ z,\ \dot{x}, \dot{y},\ \dot{z},\ \phi,\ \theta,\ \psi,\ p,\ q,\ r\right)^T$. Pay attention that Equations (7)–(15) are nonlinear due to the presence of trigonometric functions and multiplication between state variables. However, those nonlinear functions are smooth and allow linear approximations in the mini-drone flight envelope.

3. Control Strategies

3.1. PID-Based Control

PID control technique has been practiced on quadrotor platforms greatly because the quadrotor control channels can be naturally decoupled, and also, a PID controller allows for direct tuning of its parameters according to the performance requirements. Since the drone model is a Multiple Input Multiple Output (MIMO) system, a cascaded interconnection of several PID algorithms is required. The inputs to the controller have desired drone coordinates $\left(r_x, r_y, r_z\right)$. For the PID design, it is more convenient to choose the decoupled variables, $U_{d,i}$, for controller outputs, then use the Equation (4) to calculate the rotor speed references, U_i, afterward.

At the core of the quadrotor, the PID control systems is an attitude stabilization controller, as shown in Equation (16)

$$\begin{pmatrix} U_{2,d} \\ U_{1,d} \end{pmatrix} = \begin{pmatrix} 0.013 & 0 \\ 0 & 0.01 \end{pmatrix} \begin{pmatrix} e_\theta \\ e_\varphi \end{pmatrix} + 0.01 \begin{pmatrix} x_{I,\theta}(t) \\ x_{I,\phi}(t) \end{pmatrix} + \begin{pmatrix} -0.002 & 0 \\ 0 & -0.0028 \end{pmatrix} \begin{pmatrix} \hat{q} \\ \hat{p} \end{pmatrix} \tag{16}$$

where $e_\theta = r_\theta - \hat{\theta}$, $e_\varphi = r_\varphi - \hat{\varphi}$ are the pitch and roll attitude errors, $\hat{\theta}$ is the estimated pitch angle, $\hat{\varphi}$ is the estimated roll angle, $\hat{q}$ and $\hat{p}$ are the estimated pitch rate and roll rates. The integral terms, $x_{I,\theta}$ and $x_{I,\phi}$, of the controller are calculated as

$$\begin{pmatrix} x_{I,\theta}(t+T_S) \\ x_{I,\phi}(t+T_S) \end{pmatrix} = 0.999 \begin{pmatrix} x_{I,\theta}(t) \\ x_{I,\phi}(t) \end{pmatrix} + \begin{pmatrix} e_\theta \\ e_\varphi \end{pmatrix}, \tag{17}$$

where $T_s = 5$ ms is the discrete sample time of the control algorithm. Equation (17) is an approximation of the pitch and roll integrated errors, designed with a slightly smaller than unity pole to prevent wind-up effects. The pitch and roll reference signals

$$\begin{pmatrix} r_\theta \\ r_\phi \end{pmatrix} = \begin{pmatrix} -0.24 & 0 \\ 0 & 0.24 \end{pmatrix} F_{sat} \left[R_\psi \begin{pmatrix} e_x \\ e_y \end{pmatrix} \right] + \begin{pmatrix} 0.1 & 0 \\ 0 & -0.1 \end{pmatrix} \begin{pmatrix} \hat{v}_x \\ \hat{v}_y \end{pmatrix}, \tag{18}$$

are generated by a position PID controller, which tries to minimize the position errors $e_x = r_x - \hat{x}$ and $e_y = r_y - \hat{y}$ between reference coordinates (r_x, r_y) and the estimated drone position $(\hat{x}, \hat{y})$. The matrix, $R_\psi = \begin{pmatrix} c_\psi & -s_\psi \\ s_\psi & c_\psi \end{pmatrix}$, accounts for the drone yaw and F_{sat} is a saturation nonlinearity set at ± 3 m. Along the altitude dimension, the total motor torque $U_{1,d}$ is calculated as:

$$U_{1,d} = \begin{cases} mg\left(1 + K_{tf}\right) & \hat{z} > 0.5m \\ 0.8(r_z - \hat{z}) - 0.3\hat{v}_z & \hat{z} \leq 0.5m \end{cases}, \tag{19}$$

where $K_{tf} = 0.45$ is a take-off gain, r_z is the reference altitude, $\hat{v}_z$ is the estimated vertical speed, $\hat{z}$ is the estimated drone altitude. Finally, a yaw PID controller calculates the torque around the vertical axis:

$$U_{3,d} = 0.004(r_\psi - \hat{\psi}) - 0.00012\,\hat{r}, \tag{20}$$

where $\hat{\psi}$ is the estimated yaw angle and $\hat{r}$ is the estimated yaw rate.

As it can be seen, the PID control strategy depends on a set of tunable gains, which determine the closed-loop stability and performance. Even though a lot of methods for PID tuning exist, the majority of them target single input single output systems (SISO). Furthermore, even after the selection of the initial gains, manual fine-tuning of the controller parameters is always advised. In our situation, we employed an optimization approach based on the nonlinear drone model provided in the preceding section to tune the PID controller gains.

3.2. Linear Quadratic Regulator

Because of its capacity to deal with MIMO systems, LQR is a common controller on quadrotor platforms. To reduce control effort, this controller uses a cost function minimizing approach from the optimal control theory. According to the LQR algorithm, a linearized model is required. Therefore, the quadrotor model is linearized at its hovering point denoted with $(X_{ss},\ U_{ss})$, where $X_{ss} = (0_{1x2}, z_0, 0_{1x9})^T$ and $U_{ss} = (mg,\ 0,\ 0,\ 0)^T$. The deviations around the selected operating point are denoted with δU and δX such that

$$U(t) = U_{ss} + \delta U(t), \tag{21}$$

$$X(t) = X_{ss} + \delta X(t), \tag{22}$$

The purpose of LQR is to find a feedback gain matrix K to calculate the control signal as

$$\delta U(t) = -K\delta X(t), \tag{23}$$

such that following quadratic functional

$$J_{LQR} = \sum_{k=0}^{\infty} \delta X_s(kT_s) Q \delta X_s(kT_s) + \delta U(kT_s) R \delta U(kT_s) \tag{24}$$

to be minimized, where Q is a semi-positive definite square matrix of dimension n × n, and R is a positive definite square matrix of dimension m × m, wherein n = 12 is the number of states, and m = 4 is the number of control inputs. Therefore, the closed-loop control system can be represented with

$$\delta X(t + kT_s) = (A - BK)\delta X(t). \tag{25}$$

Controller gain is calculated as

$$K = \left(B^T P B + R\right)^{-1} B^T P A, \tag{26}$$

where P is a solution of the discrete-time algebraic Riccati equation

$$A^T P A - P - A^T P B \left(B^T P B + R\right)^{-1} B^T P A + Q = 0. \tag{27}$$

In order to construct an LQR controller for the mini-drone, the nonlinear model of drone motion must be expressed in discrete-time state-space coordinates of the form:

$$x_s(t + T_S\ ) = A\ x_s(t) + B \delta U(t) \tag{28}$$

The vector $x_s(t) = \delta X(t)$ represents the set of linearized system states—position coordinates (x, y, z), velocities (u, v, w), Euler angles (φ, θ, ψ) and angular rates (p, q, r). The input signals used to stabilize the quadrotor are represented with the vector $\delta U(t)$. The sample time is chosen to be $T_s = 5$ ms. The operating point for the linearized model is numerically trimmed such that the mini-drone is held stable by hovering about 1 m above the ground. The resultant discrete-time linearized model, which is an approximate of the nonlinear model, expressed with (7)–(15), for the hovering operating point, is described with the equations:

$$x(t + T_S) = x(t) + T_S\ u(t), \tag{29}$$

$$y(t + T_S) = y(t) + T_S\ v(t), \tag{30}$$

$$z(t + T_S) = z(t) + T_S\ w(t), \tag{31}$$

$$\varphi(t + T_S) = \varphi(t) + 0.5\ p(t), \tag{32}$$

$$\theta(t + T_S) = \theta(t) + 0.5\ q(t), \tag{33}$$

$$\psi(t + T_S) = \psi(t) + 0.5\ r(t), \tag{34}$$

$$p(t + T_S) = 0.99p(t) + 5T_S(U_1 + U_2 - U_3 - U_4), \tag{35}$$

$$q(t + T_S) = 0.99q(t) + 4T_S(U_1 - U_2 - U_3 + U_4), \tag{36}$$

$$r(t + T_S) = -0.1T_S(U_1 + U_2 + U_3 + U_4), \tag{37}$$

$$u(t + T_S) = u(t) - 0.05\theta(t) + 0.15\ T_S\ q(t), \tag{38}$$

$$v(t + T_S) = v(t) + 0.05\varphi(t) - 0.15\ T_S\ p(t), \tag{39}$$

$$w(t + T_S) = w(t) + 0.1T_S(-U_1 + U_2 - U_3 + U_4) \tag{40}$$

The above Equations (29)–(40) should not be confused with their nonlinear counterparts (7)–(15), which were presented above. In the linearized model for the LQR design, there is no need to employ the decoupled controls $U_{i,d}$ because they are linearly dependent on rotor speed references U_i. As it can be seen, the linearized model is chain-like due to sparse A and B, with control inputs acting directly on angular rates and on the vertical

translation rate. All four inputs affect these states; hence, the controller aims to decouple these dependencies.

We have performed analytic linearization and time discretization of the nonlinear model for the specified hovering operating point. The quadrotor model linearization is a more-or-less standardized procedure; hence, we did not provide all the details of it. However, it can be extensively investigated with respect to approximation accuracy, which is a topic for another study. Our experiments with the drone show that the implementation of the MPC is very sensitive to the numerical roundings in the plant linear model. Hence the matrices of the linearized model are rounded in the following way to improve the numerical stability of the iterative procedure.

In order to obtain matrices Q and R, Bryson's rule is used (Table 2). The following are the weights for each state while determining the Q matrix for LQR design:

Table 2. Cost weights and its interval to determine matrix Q.

Cost Name	Weight (%)	Interval
$W_{pos,X}$	0.08	[0, 1]
$W_{pos,Y}$	0.08	[0, 1]
$W_{pos,Z}$	100.5	[0, 1]
$W_{\varphi,\theta,\psi}$	0.175	[0, 0.35]
$W_{u,v,w}$	0.0583	[0, 1]
$W_{p,q,r}$	0.8	[0, 1]

The resulting matrix Q is with elements only on the main diagonal denoted as:

$$Q = diag\left(Q_x, Q_y, Q_z, Q_\varphi, Q_\theta, Q_\psi, Q_p, Q_q, Q_r, Q_u, Q_v, Q_w\right) \tag{41}$$

where $Q_{x,y} = 0.0008$, $Q_z = 0.97$, $Q_{\varphi,\theta,\psi} = 0.0048$, $Q_{p,q,r} = 0.008$ and $Q_{u,v,w} = 0.0006$. The matrix $R = 8 \times 10^{-5} \cdot I_4$, where I_4 is the identity matrix of the fourth order, and the state feedback matrix, K:

$$K = \begin{pmatrix} -1.5 & 1.5 & -54 & 19 & 19 & -4 & 5 & 6 & -12 & -3 & 3 & -51 \\ 1.5 & 1.5 & 54 & 19 & -19 & -4 & 5 & -6 & -12 & 3 & 3 & 51 \\ 1.5 & -1.5 & -54 & -19 & -19 & -4 & -5 & -6 & -12 & 3 & -3 & -51 \\ -1.5 & -1.5 & 54 & -19 & 19 & -4 & -5 & 6 & -12 & -3 & -3 & 51 \end{pmatrix} \tag{42}$$

3.3. Model-Predictive Controller

MPC for quadrotor platforms is gaining popularity in control engineering because it includes characteristics such as handling control constraints, state estimation, predictive behavior, noise, and disturbance rejection. More specifically, MPC gained popularity as a result of its predictive behavior, which may reduce closed system uncertainty [19]. MPC employs the following steps (Figure 2):

1. Construct a discrete-time state-space model.
2. Over a predefined prediction horizon, it predicts future plant states and outputs for a given control signal sequence over a predefined control horizon.
3. Quadratic optimization procedure selects a control sequence, which minimizes the closed-loop cost function.
4. Only the first sample from the optimal control signal is applied to the plant.
5. The method is then repeated from step 2 sequentially.

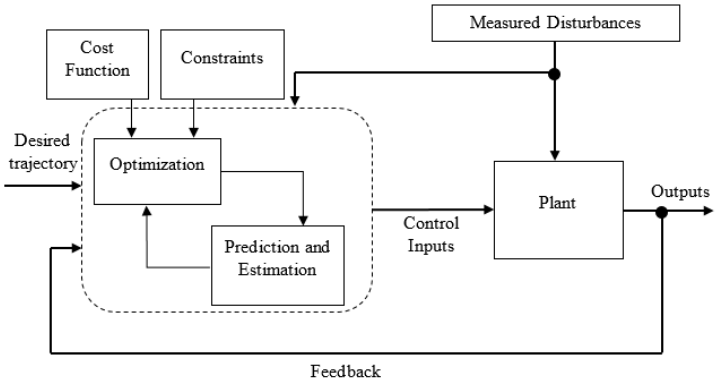

Figure 2. A block diagram of MPC workflow.

3.3.1. Plant Model

The design procedure of MPC takes the linearized discrete-time mini-drone model, specified with Equations (29)–(40), for the hovering operating point. The MPC continually predicts the behavior of the states within a prediction horizon, N [16], as

$$\begin{pmatrix} x_{s,0} \\ x_{s,1} \\ x_{s,2} \\ \vdots \\ x_{s,N-1} \end{pmatrix} = \begin{pmatrix} I \\ A \\ A^2 \\ \vdots \\ A^{N-1} \end{pmatrix} x_{s,0} + \begin{pmatrix} 0 & 0 & \cdots & 0 & 0 \\ B & 0 & \cdots & 0 & 0 \\ AB & B & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ A^{N-2}B & A^{N-3}B & \cdots & B & 0 \end{pmatrix} \begin{pmatrix} \delta U_0 \\ \delta U_1 \\ \delta U_2 \\ \vdots \\ \delta U_{N-1} \end{pmatrix}, \tag{43}$$

where $x_{s,k} = x_s(t + kT_s)$ is the predicted state k sample steps after the present moment t, δU_k are the future values of the control signal, A and B are the matrices of the linearized system (28).

3.3.2. Control Design

The MPC optimizer minimizes a quadratic cost function using two different weighting matrices $\hat{W}_U$ and $\hat{W}_X$ of appropriate dimension, which can be stated as

$$J_{MPC} = \sum_{k=0}^{N} \hat{W}_X (x_{s,k} - x_{s,r})^2 + \hat{W}_U \delta U_k, \tag{44}$$

where $x_{s,r}$ are the desired state values. Unlike the quaternion orientation-based quadrotor model, the Euler angle-orientation-based quadrotor model uses algebraic subtraction to determine the cost function; hence, this equation can be used to directly reduce the cost function [20–22]. Quadratic programming is used to solve optimization problems by minimizing the cost function in a feasible search direction δU [23,24].

3.3.3. Constrain Handling

In general, a motor has the ability to generate a certain amount of force that relies on the rpm and propeller design parameters. As a result, the control inputs are bound to the limit of motors' properties in generating forces. Hence, the constraints must be considered at control input to design the control algorithm properly. However, when an upper bound, u_{ub}, and a lower bound, u_{lb}, at the control inputs where $u_{lb} \leq U_i \leq u_{ub}$. The numerical representation of the discrete-time state-space model is for the fixed sample time of 5 ms. However, simulations of the nonlinear model with various MPC designs confirmed that required prediction horizon to achieve stable flight conditions was between 40–50 samples, with a control horizon of roughly 5 samples. However, the hardware platform running the controller implementation at the Parrot mini-drone did not allow such high order of prediction and control horizons due to insufficient resources. Hence, to reduce the prediction horizon samples, we decided to resample the discrete time state space model

with a sample time of Ts = 100 ms. That way, the required prediction horizon for successful hover is about 10 samples, and the control horizon is about 2 samples. This is a limitation coming from what the Parrot Mambo drone microcontroller can sustain without violating real-time execution constraints; hence, it can be understood as a hardware limitation. The resampling operation over the state space model is achieved with the help of Tustin bilinear approximation. The match of the resampled and the original model is checked in the time and frequency domain for the frequencies up to 5 rad/s.

In the presented MPC design prediction horizon is set to 10 samples, and the control horizon is set to 2 samples. The nominal control vector, U = (237–237 237–237), and the nominal state vector Y = 0 correspond to the selected operating point of the linearized model. All control inputs are saturated in the bounds [10 500] for motors 1 and 3 and in the bounds [−500 −10] for motors 2 and 4. The weights for the control signal are set to $\widehat{W}_{U} = 0.000001$, and control signal rates weights are set to 0.0005. The weights are set to these values through iterative experimentation with the Parrot Mambo drone. However, these values are not the only possible stabilizing solution.

The critical for the control performance is the value of the state weights. In the present design, we assume all states are measurable as

$$\widehat{\mathrm{W}}_{\mathrm{X}} = \begin{pmatrix} 1 & 1 & 0.5 & 0.1 & 0.1 & 0.1 & 0.05 & 0.05 & 0.05 & 0.7 & 0.9 & 0.05 \end{pmatrix} \tag{45}$$

Output weights vector is normalized to a unit vector. A white noise model is assumed to act on all outputs with a variance 0.01. That is introduced to smooth further the generated controller signal. Since the MPC design will be implemented in real-time, that requires tight bounds on the optimization iterations. As a result, sub-optimal solutions are tolerated, and the maximum number of iterations is limited to 10.

4. Simulation Results

The controller variants are evaluated in simulation as well as in experiments. Typically controller performance is best understood in the frequency domain. However, the MPC is a nonlinear control approach, and frequency response has to be obtained through approximation. Since there is not a unique way to define such approximation, that will be a question for separate research. Therefore, we decided to compare the controllers in the time domain alone in this study.

The desired drone trajectory is selected as a rectangular path, which is stretched by a low-frequency sinusoidal wave in each spatial dimension to increase the trajectory complexity. The resultant trajectory is defined by the following expressions:

$$X_{ref}(t) = \int_0^t V_{\mathrm{X},0}(\tau)d\tau + 2sin(0.1\pi t + 0.3)\ 1(t-10) \tag{46}$$

$$Y_{ref}(t) = \int_0^t V_{\mathrm{Y},0}(\tau)d\tau + 2sin(0.1\pi t + 0.6)\ 1(t-10) \tag{47}$$

$$Z_{ref}(t) = -1.5 + 0.1sin(0.02\pi t + 0.3)\ 1(t-10) \tag{48}$$

$$V_{\mathrm{X},0}(t) = \begin{cases} -0.5, & t \in [25, 35] \\ 0.5, & t \in [55, 65] \\ 0 \end{cases} \tag{49}$$

$$V_{\mathrm{Y},0}(t) = \begin{cases} -0.5, & t \in [10, 20] \\ 0.5, & t \in [40, 50] \\ 0 \end{cases} \tag{50}$$

The yaw, pitch, and roll reference signals are held at zero. At low altitudes, the ground rotor effect is strongly disturbing the stability of the drone; hence, initially, take-off power impulse is applied to the motors such that commanded torque is $U_{\mathrm{i}} = -(1 + K_{tf})\mathrm{mg}$, where $K_{tf} = 0.45$ is empirically selected take-off gain. Figure 3 overviews the comparison between

PID, LQR, and MPC drone control systems during tracking of the desired trajectory. Both LQR and MPC demonstrate similar tracking performance keeping aircraft close enough to the reference. However, the PID-based closed-loop system response is oscillatory during some of the trajectory segments, even though the aircraft stability is recovered after that. During simulation, safety-critical sensor conditions for ± 10 m flight zone boundaries and velocity estimation error $E_{x,y}^{vel}$ at ± 6 m/s are monitored such that simulation stops if asserted. The velocity estimation error $E_{x,y}^{vel} = V_{x,y}^{cam} - \hat{V}_{x,y}$ is between velocity $V_{x,y}^{cam}$ from optical flow processing and velocity $\hat{V}_{x,y}$ from the complementary filter. None of the controllers drive the system close to these conditions, and the estimation errors of state coordinates are negligible. Landing is not simulated in this scenario; however, the controller handles landing by descending at a near ground altitude of -0.6 m, in which spinning down the rotors is safe for the drone.

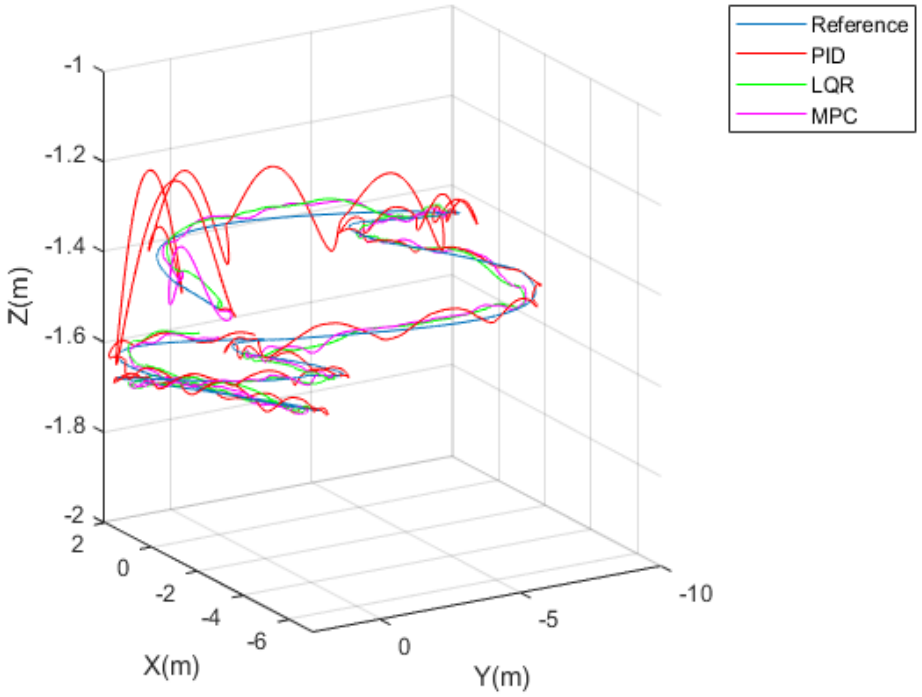

Figure 3. Tracking performances of PID, LQR and MPC during simulation.

Table 3 calculates 2-norms of the tracking errors between reference position signals and actual drone positions, as well as the 2-norms of the yaw, pitch, and roll angles, for the simulation duration $T_{sim} = 100$ s. The PID-based control system demonstrates the best tracking performance along with X and Y dimensions; however, it is worst along the critical for stability Z dimension. The drone altitude is tightly associated with averaged rotor velocity, which is determinant for the current operating point and respective linear approximation of the equations of motion. Along with the pitch and roll components, the PID-based controller is also with the worst performance compared to LQR and MPC systems. On the other hand, the LQR controller performs best in keeping drone orientation, such that for the yaw angle, the 2-norm is 10 times less for the LQR compared to the PID-based. The tracking performance along X, Y, and Z coordinates are similar for LQR and MPC closed loops, and the same is true for the pitch and roll angles. One drawback of the MPC is evident in the large yaw angle component compared to PIC and LQR. That does not affect the position tracking performance. Eventually, that yaw error can be minimized by modifying the respective weights during the specification of the MPC cost function.

Table 3. Tracking errors of different controllers.

Dim	PID	LQR	MPC
X error	29.005	111.7	103.61
Y error	51.654	150.14	103.61
Z error	21.895	8.9773	6.047
ϕ error	13.146	2.2553	4.1943
θ error	3.7284	1.6177	2.9035
ψ error	0.5375	0.0372	12.24

Figure 3 shows the resultant trajectory of the drone center of gravity in Cartesian coordinates for the three examined controllers. It is clear that the PID-based controller is unable to keep the aircraft on track, particularly after take-off around position (0, 0). Furthermore, it can be seen that there is no significant difference in tracking performance between LQR and MPC controllers. This can be attributed to the similarity in cost criteria utilized in these controllers. Both LQR and MPC are based on the quadratic cost with state and control weighting matrices. However, in the case of LQR, the controller minimizes the quadratic cost over an infinite time horizon, and MPC solves the optimization problem iterative for the finite prediction horizon. According to the trajectory tracking experiment, the optimum performance is obtained when the drone altitude is close to the operational point value used to build the linearized model.

Figures 4 and 5 show the detailed transient response of the three controllers during trajectory tracking along the X and Y dimensions. Because the trajectory segments approximate a constant velocity reference signal, the steady state error will be proportional to the number of integrator elements in the loop. The PID-based control loop works with zero steady-state errors but with pronounced oscillations with amplitudes around ± 0.5 m. The LQR and MPC work with a constant steady state error around 2 m during constant velocity trajectory segments, which is the reason for the higher 2-norms of the respective tracking errors. However, the velocity error is zero, and the magnitude of the position error can be controlled with the slope of the position reference. Furthermore, the response of the LQR and MPC is not oscillatory. Generally, both controllers demonstrate quite similar behavior, which can be explained by the shared quadratic cost.

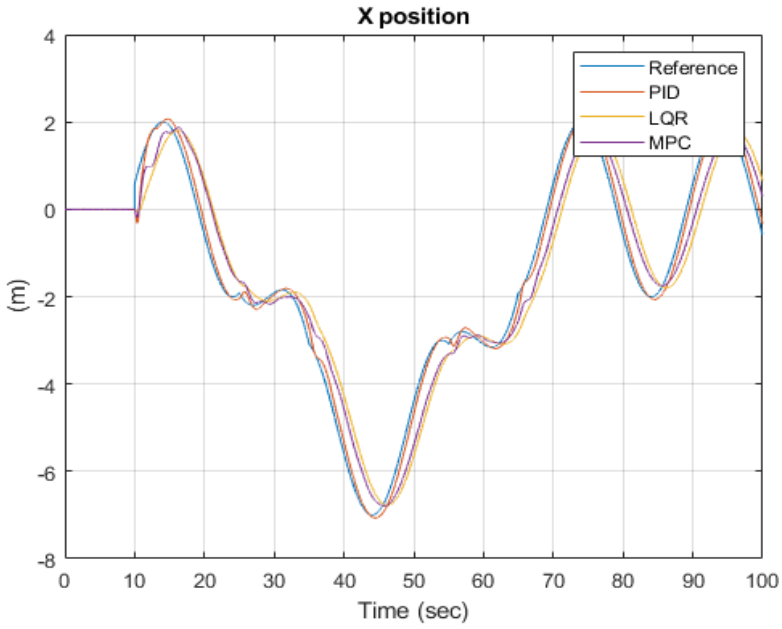

Figure 4. Performance of the controllers along x axis in simulation.

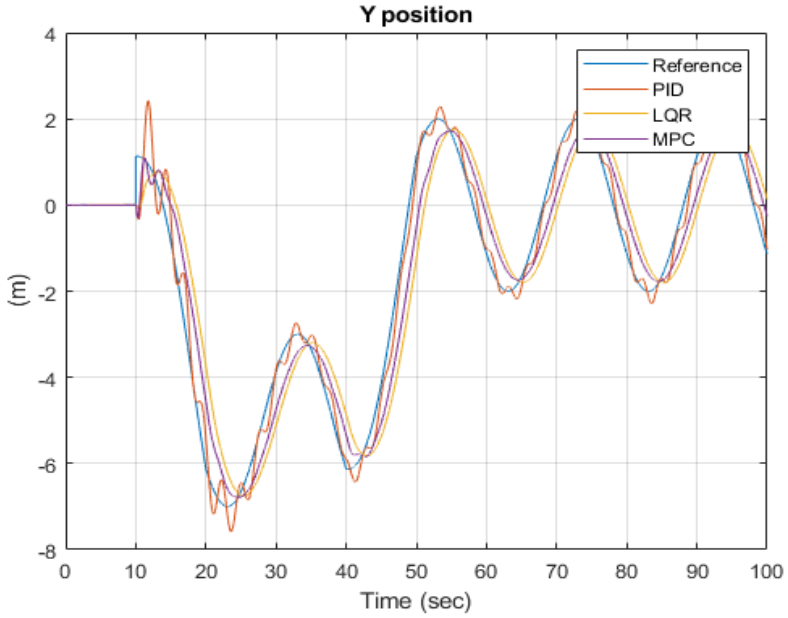

Figure 5. Performance of the controllers along y axis in simulation.

Figure 6 compares the altitude tracking performance between the controllers where two transients are most evident. The first is right after take-off when the -1.5 m operating height must be stabilized. Before lowering to the required altitude, a PID-based controller is characterized by a large overshoot of more than 50% or reaching –3.3 m. Due to developed overshot, the drone motors almost stop causing a free fall. With the LQR, the overshot is considerably damped to around 30%, and finally, with the MPC, the ascent is aperiodic without overshot. After reaching the operating altitude of –1.5 m a small sinusoidal reference variation is commanded. All controllers succeed in following that reference, but the transient for the PID-based controller is oscillatory.

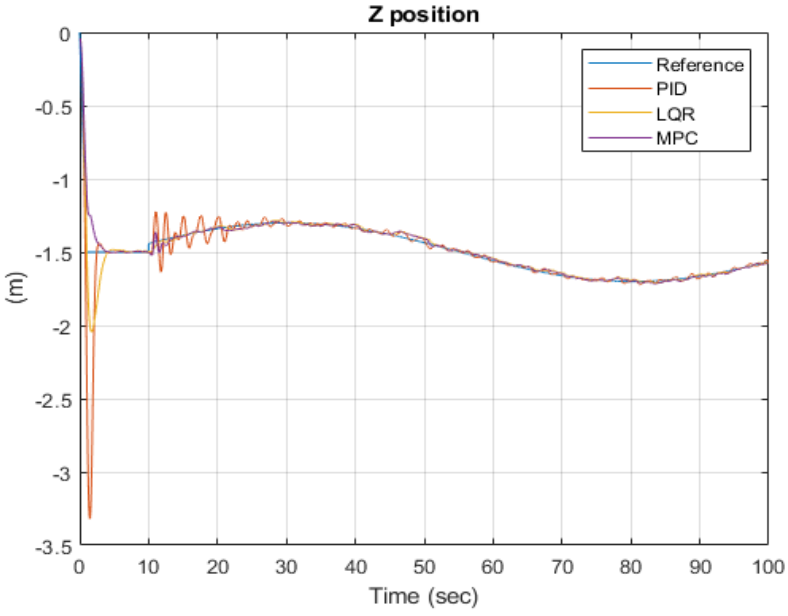

Figure 6. Performance of the controllers along z axis in simulation.

Figure 7 demonstrates the performance in roll angle stabilization for the controllers. On the one hand, small deviations in the roll and pitch angles are necessary to accomplish drone translation in Cartesian space, inducing a projection of the motor thrust vector along

the horizontal plane. On the other hand, roll and pitch angle amplitudes are critical for aircraft stability in the presence of external disturbances such as wind, ground effects, asymmetries in rotor wakes, etc.

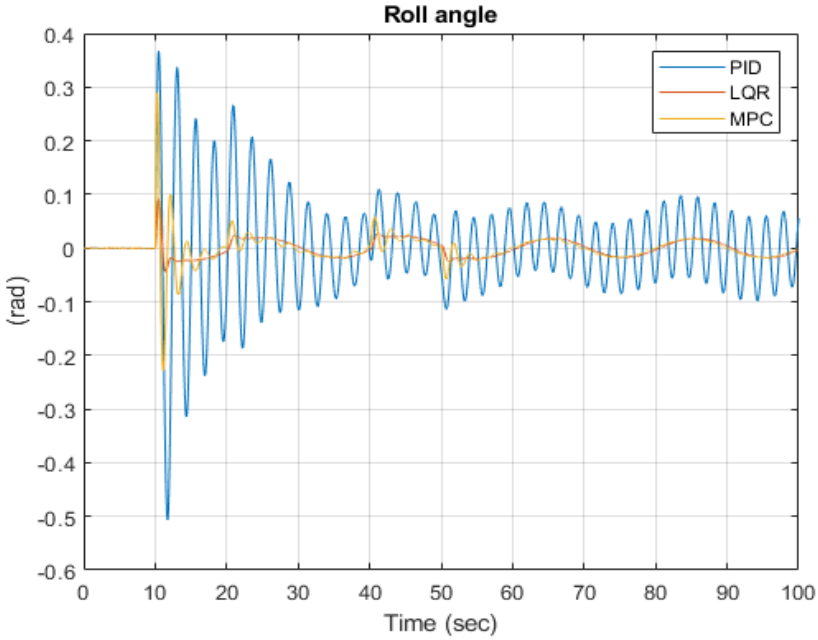

Figure 7. Performance of the controllers in roll angle in simulation.

The PID-based closed-loop system's behavior is highly oscillatory, as can be observed, with a steady state amplitude of roughly 0.1 rad and a frequency of about 0.4 Hz. Even though the aircraft remains stable, such oscillations indicate small stability margins. The control system based on LQR is aperiodic, with a maximum deviation of roughly 0.025 rad. Similar behavior is obtained with the MPC algorithm. However, small oscillations appear during transients along the trajectory with amplitude up to 0.05 rad.

Sensitivity of the closed-loop to disturbances is most evident after the take-off, where the control algorithm is enabled at t = 10 s in the presence of non-zero initial conditions for the plant due to impulsive motor trust required to lift the drone high enough to overcome the ground effect. Integral terms of the controller are still not developed. Hence, large swings in the roll and pitch angles appear. For the PID, maximal swing amplitude is as large as 0.5 rad, for LQR, it is 0.2 rad, and for MPC, it is 0.3 rad.

The simulated pitch angle is depicted in Figure 8. As can be seen, the pitch angle variations using the PID-based controller are far fewer than those seen in the roll angle. This indicates that the cause of the PID oscillatory behavior is related to different drone inertia along pitch and roll axes. Hence, additional tuning of the PID could improve its performance. As noted in the introduction, a lot of PID tuning rules exist for SISO systems, but not so much for multivariate systems as is the case with the drone. On the contrary, tuning of the MPC- or LQR-based regulators is more direct with specifying weighting matrices for the output and control variables.

In the steady state, the maximal pitch deviation for the PID controller is 0.075 rad, for the LQR, it is 0.025 rad, and for the MPC, it is 0.04 rad. The maximum pitch deviation for the PID-based controller is 0.3 rad after the take-off impulse, 0.06 rad for the LQR, and 0.17 rad for the MPC.

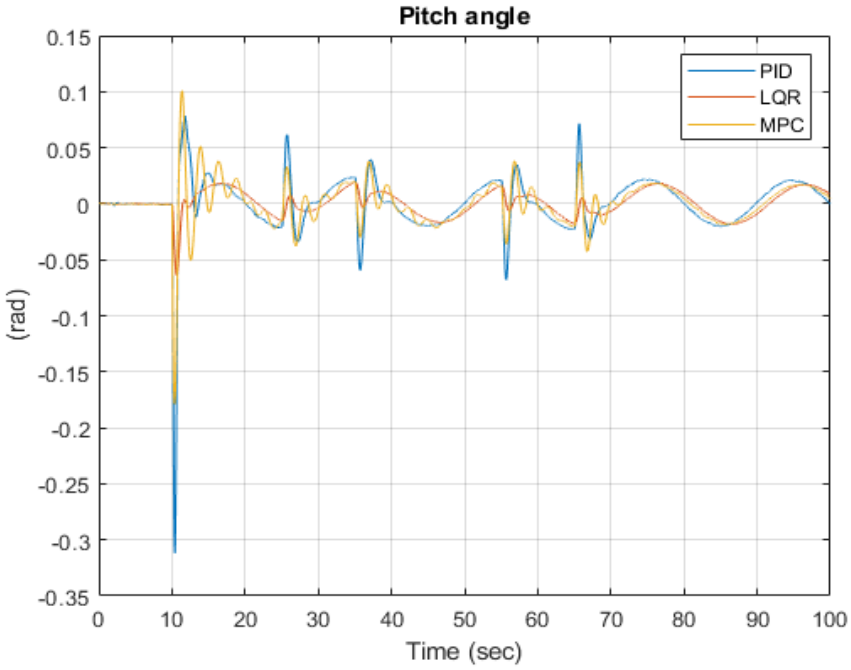

Figure 8. Performance of the controllers in pitch angle in simulation.

5. Experimental Results

Figures 9–12 present the experimental results with the designed PID, LQR, and MPC controllers. For the implementation of the controllers in the Parrot Mambo mini-drone, the authors have employed the Simulink Coder Toolbox capabilities to automatically convert the controller model to the MISRA compatible C code. The generated code can be validated in software in the loop simulations. For the particular mini-drone, we are relying on the Mathworks supplied framework models for Parrot, which is extended with the developed controllers. We also provide the Simulink diagram used for code generation for the 3 controllers as a Supplementary Material.

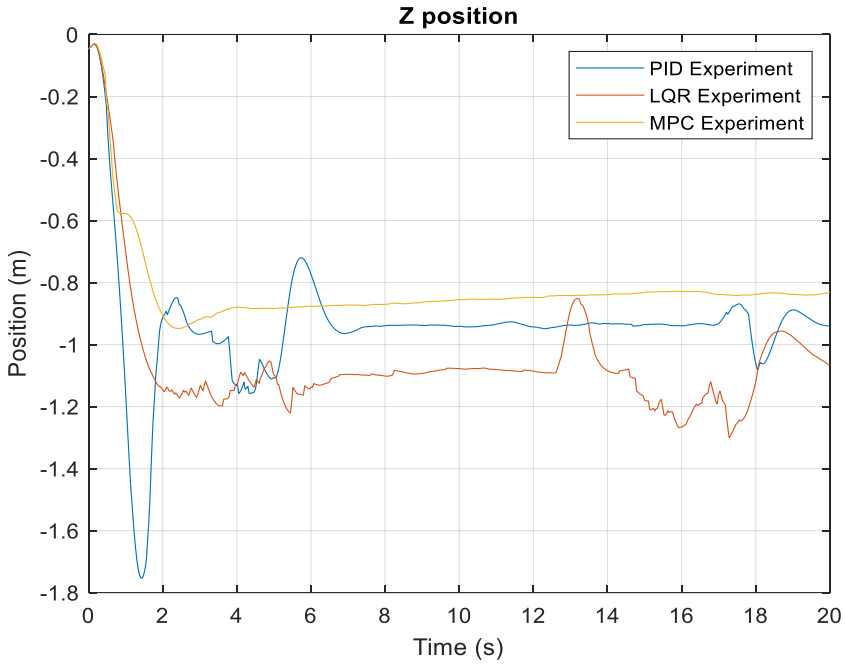

Figure 9. Performance of the controllers along z axis in real-time.

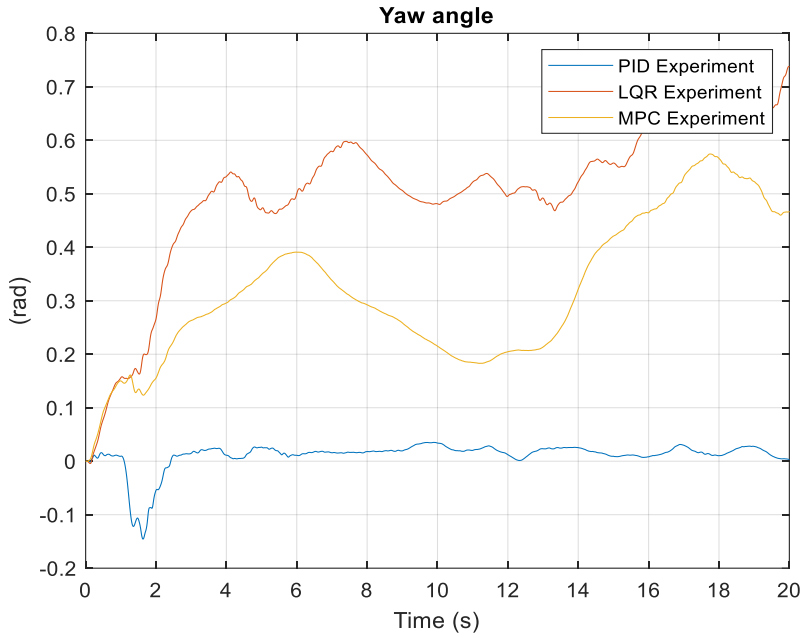

Figure 10. Performance of the controllers for yaw angle in real-time.

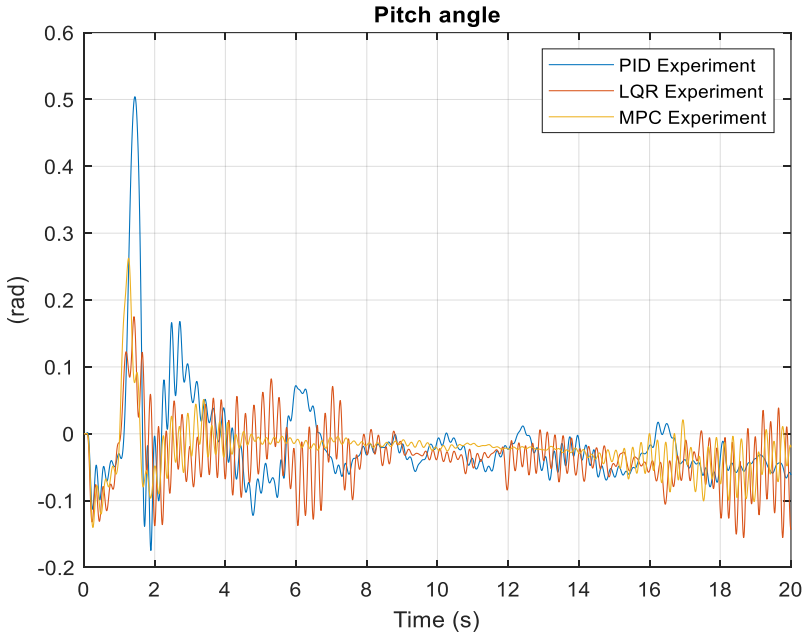

Figure 11. Performance of the controllers for pitch angle in real-time.

This framework model takes operator command, raw sensor signals, and image data from the drone camera as inputs and calculates the four motor speed reference commands. The sensor signals are composed of three accelerometer signals, three gyroscope signals, optical flow data, ultrasound altitude signal, and pressure signal. Accelerometer, gyroscope, and pressure data are first calibrated by removing the constant bias terms detected during sensor initialization. Low pass filters are applied to the accelerometer data and yaw rate data. Then the complementary filter estimates the mini-drone orientation from the accelerometer, gyroscope signals, and yaw estimate from the optical flow driver. The aim of the complementary filter is to produce a converging estimate for the drone orientation through minimization of the error between accelerometer or optical flow and gyroscope integrated Euler angles. The altitude and vertical speed are then estimated from orientation

estimate, gravity vector, and ultrasonic signal through a Kalman filter. Furthermore, the translation velocity of the drone is estimated from the optical flow velocity estimate, altitude estimate, and orientation estimate. Finally, the drone position is estimated with the help of the Kalman filter by comparing the integrated velocity and optical flow estimate.

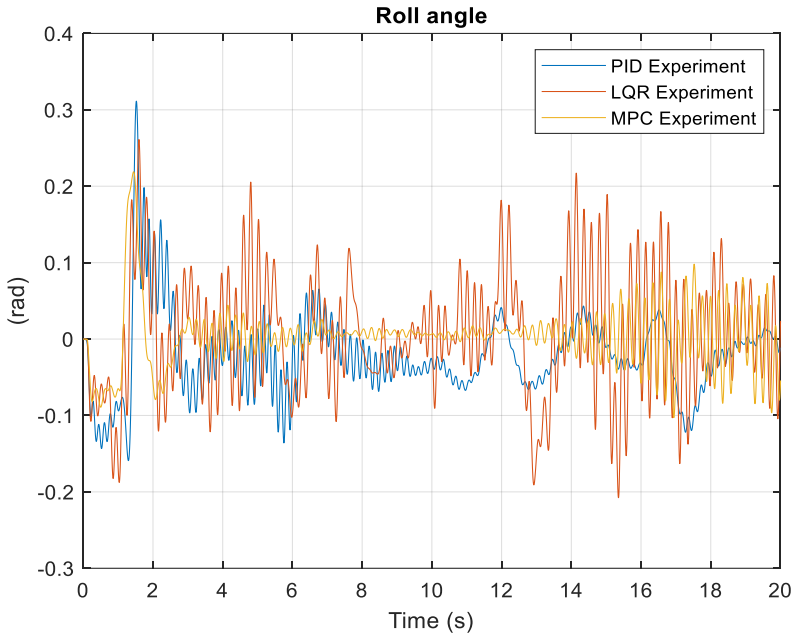

Figure 12. Performance of the controllers for roll angle in real-time.

The following figures compare the experimental performance of the designed regulators. An important note is that experiments are performed indoors, which is well recognized that magnetometer, ultrasonic and optical signals can be disturbed. The general observation is that LQR and MPC controllers demonstrate similar performance, with MPC being a bit smoother in response.

In Figure 9, along the Z position, the PID hovering error is settled at around 20 cm; however, it is increasing. With the LQR or MPC controllers, the hovering error is steady at around 30 cm but with a decreasing trend.

The controllers do not show the same response in their rotational motion as they show in simulation. PID controller shows significant overshoot in the pitch angle response and undershoots in yaw angle response. However, PID response is better compared to LQR and MPC in yaw movement. Along both roll and pitch movement, MPC and LQR show jittery responses, while PID is better in response compared to them.

This work introduces the simulated and experimental results of the Parrot mini-drone flight with three different controllers such as MPC, PID, and LQR. This study compares the performance of the controllers from several aspects such as tracking error, stability, and robustness. From the tracking error aspect in simulation, LQR performs well compared to PID and MPC. However, in terms of stability and robustness in real-time, MPC shows promising performance while both PID and LQR show fluctuations in their responses. Theoretically, MPC should perform well because it is a nonlinear controller and can estimate the system's behavior. However, it could not show its strength since the Parrot mini-drone is a low-budget commercial quadrotor equipped with some sensors that are poor in response. Positions along the X and Y axes are measured by a camera instead of a GPS sensor, and hence, the responses become worse when it moves over a plane ground that is without a black-white box pattern. During the experiments, it was discovered that the sensor's performance degrades over time, and so the quadrotor cannot be run for a longer period of time to fully appreciate the controller's performance. In addition, the microcontroller that is used in this quadrotor is not suitably efficient to offer a suitable platform to show a controller's strength effectively. As a result, environmental factors may greatly affect

the quadrotor's performance in that case where the controllers have limited opportunity to show their robustness. Thus, the system shows inconsistency in both simulated and experimental results.

Both MPC and LQR are suitable controllers for the quadrotor platform because they have six states that are controlled by four inputs. However, PID is suitable for dealing with the SISO system, and hence, X and Y positions are coupled with roll and pitch movement to be controlled by PID controllers, respectively, though finally, six different PID controllers have been adopted for the states. Hence, it is time-consuming to find out suitable gains. On the other hand, LQR is a low computational controller that relies on Q and R matrices.

MPC is a powerful nonlinear controller that can ensure stability and robustness to the system, and interestingly, in this experimental work, it offers stability in the system compared to the other controllers as well. It couples estimator with cost function approach that makes more computational, however. Hence, designing a suitable controller is not as stressful as PID.

6. Conclusions

In conclusion, PID, LQR, and MPC controllers have been applied to Parrot mini-drones, and their strengths and weakness have been analyzed in this platform. These controllers have been chosen considering the existing works where MPC on Parrot mini-drone is not available. The most important findings of this study can be summarized as:

- The employment of a PID-based controller on a quadrotor platform is straightforward, given the accumulated research on the topic. The PID approach is advantageous because it allows manual parameter tuning but does not provide a unified, systematic way to optimize the system performance. We do not claim that simulated and experimental PID performance will be as poor with all the possible parameter tunings. However, for our time spent tuning the PID controller, we could not find tunings that led to a better result. The tuning of multivariate PID controllers is mostly a heuristic task trying to balance the requirements of several feedback loops with the aim of keeping internal stability.
- The application of LQR for mini-drone stabilization is also extensively investigated on various platforms. It generally shows better performance in tracking and stability than PID-based controllers, which was confirmed by our simulations and experiments. The tuning of LQR requires less effort than tuning of PID because the existence of a solution to the algebraic Riccati equation guarantees the closed-loop stability. Interestingly, in experimental work, both MPC and LQR show similar tracking errors along the altitude.
- This work brings novelty to the Parrot Mambo platform by offering MPC and introducing its characteristics and suitability. This work substantiates that MPC can be considered to ensure the stability and robustness of the system; however, simulation work shows poor performance in tracking than LQR. The explanation can be found that LQR provides an infinite time horizon quadratic cost minimization, while the MPC is always limited to a finite time horizon. Moreover, the iterative nature of MPC solution calculation may not always converge to a solution that minimizes the cost function. Practical tuning of the MPC weights to run on a Parrot Mambo drone proved a very hard task, requiring a lot of experiments.
- The practical implementation of the proposed controllers was facilitated by the automatic code generation capabilities offered by the Simulink Coder. That mostly eliminated the need to perform lower-level system programming. The microcontroller installed in the Parrot Mambo drone is powerful enough to support each of the designed controllers. However, we have reached some limitations with the implementation of MPC. Due to its intensive computation requirements, we had to limit the prediction horizon to 10 samples and increase its sample time to 100 ms in order to perform a successful experiment. The LQR is far more efficient in implementation

than MPC. The greatest benefit of MPC is that it allows explicit account for the control signal amplitude and rate constraints.

- A relatively large mismatch between the simulation and experimental results is evident in the present research. We have explored a vast number of controller parameter values, which led us to the conclusion that there is inherent uncertainty in the linearized model, which can explain the differences between the simulation and experimental results. That can be compensated by performing a special identification experiment with random signals to extract statistical estimates about model parameters or estimate a state-space model directly using subspace identification methods.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/aerospace9060298/s1, Simulink flow diagrams for the controller implementation.

Author Contributions: Supervision and project administration, M.O.; conceptualization, M.O. and M.I.; methodology, M.O and J.K.; software, M.O. and J.K.; experiments, J.K.; writing—review and editing, M.O., J.K. and M.I. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the United Arab University grant (G00003527), UAE.

Data Availability Statement: Not applicable.

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

References

1. Olejnik, A.; Kiszkowiak, Ł.; Rogólski, R.; Chmaj, G.; Radomski, M.; Majcher, M.; Omen, Ł. The Use of Unmanned Aerial Vehicles in Remote Sensing Systems. *Sensors* **2020**, *20*, 2003. [CrossRef] [PubMed]
2. Mehmood, Y.; Aslam, J.; Ullah, N.; Chowdhury, S.; Techato, K.; Alzaed, A. Adaptive Robust Trajectory Tracking Control of Multiple Quad-Rotor UAVs with Parametric Uncertainties and Disturbances. *Sensors* **2021**, *21*, 2401. [CrossRef] [PubMed]
3. Ganesan, R.; Raajini, X.M.; Nayyar, A.; Sanjeevikumar, P.; Hossain, E.; Ertas, A.H. BOLD: Bio-Inspired Optimized Leader Election for Multiple Drones. *Sensors* **2020**, *20*, 3134. [CrossRef] [PubMed]
4. Sun, C.; Liu, M.; Liu, C.; Feng, X.; Wu, H. An Industrial Quadrotor UAV Control Method Based on Fuzzy Adaptive Linear Active Disturbance Rejection Control. *Electronics* **2021**, *10*, 376. [CrossRef]
5. Zulu, A.; John, S. A Review of Control Algorithms for Autonomous Quadrotors. *arXiv* **2016**, arXiv:1602.02622. [CrossRef]
6. Junior, J.C.V.; Paula, J.C.; De Paula, J.C.; Leandro, G.V.; Bonfim, M.C. Stability Control of a Quad-rotor Using a PID Controller. *Braz. J. Instrum. Control* **2013**, *1*, 15–20. [CrossRef]
7. ElKholy, H. Dynamic Modeling and Control of a Quadrotor Using Linear and Nonlinear Approaches. Master's Thesis, The American University in Cairo, New Cairo, Egypt, 2014.
8. Kendoul, F. Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems. *J. Field Robot.* **2012**, *29*, 315–378. [CrossRef]
9. Baek, J.; Jung, J. A Model-Free Control Scheme for Attitude Stabilization of Quadrotor Systems. *Electronics* **2020**, *9*, 1586. [CrossRef]
10. Patel, B.; Patle, B. Analysis of Firefly–Fuzzy Hybrid Algorithm for Navigation of Quad-Rotor Unmanned Aerial Vehicle. *Inventions* **2020**, *5*, 48. [CrossRef]
11. Everett, M.F. LQR with Integral Feedback on a Parrot mini-drone. Massachusetts Institute of Technology, Tech. Rep. 2015. Available online: http://mfe.scripts.mit.edu/portfolio/img/portfolio/16.31/16.31longreport.pdf (accessed on 7 April 2022).
12. Glazkov, T.V.; Golubev, A.E. Using Simulink Support Package for Parrot mini-drones in nonlinear control education. In *AIP Conference Proceedings*; AIP Publishing LLC: Melville, NY, USA, 2019; Volume 2195, p. 020007.
13. Castañeda, H.; Gordillo, J. Embedded Flight Control Based on Adaptive Sliding Mode Strategy for a Quadrotor Micro Air Vehicle. *Electronics* **2019**, *8*, 793. [CrossRef]
14. Islam, M.; Okasha, M.; Idres, M.M. Trajectory tracking in quadrotor platform by using PD controller and LQR control approach. In *IOP Conference Series: Materials Science and Engineering*; IOP Publishing: Bristol, UK, 2017; Volume 260, p. 012026. [CrossRef]
15. Sabatino, F. Quadrotor Control: Modeling, Nonlinearcontrol Design, and Simulation. Master's Thesis, Electrical Engineering, KTH Royal Institute of Technology, Stockholm, Sweden, 2015.
16. Islam, M.; Okasha, M.; Idres, M.M. Dynamics and control of quadcopter using linear model predictive control approach. In *IOP Conference Series: Materials Science and Engineering*; IOP Publishing: Bristol, UK, 2017; Volume 270, p. 012007. [CrossRef]
17. Rodríguez-Abreo, O.; Garcia-Guendulain, J.M.; Hernández-Alvarado, R.; Flores Rangel, A.; Fuentes-Silva, C. Genetic Algorithm-Based Tuning of Backstepping Controller for a Quadrotor-Type Unmanned Aerial Vehicle. *Electronics* **2020**, *9*, 1735. [CrossRef]
18. Lyu, H. *Multivariable Control of a Rolling Spider Drone*; University of Rhode Island: Kingston, RI, USA, 2017.
19. Islam, M.; Okasha, M.; Sulaeman, E. A Model Predictive Control (MPC) Approach on Unit Quaternion Orientation Based Quadrotor for Trajectory Tracking. *Int. J. Control Autom. Syst.* **2019**, *17*, 2819–2832. [CrossRef]

20. Murillo, M.H.; Limache, A.C.; Fredini, P.S.R.; Giovanini, L.L. Generalized nonlinear optimal predictive control using iterative state-space trajectories: Applications to autonomous flight of UAVs. *Int. J. Control Autom. Syst.* **2015**, *13*, 361–370. [CrossRef]
21. Saif, A.W.A.; Aliyu, A.; Dhaifallah, M.A.; Elshafei, M. Decentralized backstepping control of a quadrotor with tilted-rotor under wind gusts. *Int. J. Control Autom. Syst.* **2018**, *16*, 2458–2472. [CrossRef]
22. Islam, M.; Okasha, M. A Comparative Study of PD, LQR and MPC on Quadrotor Using Quaternion Approach. In Proceedings of the 2019 7th International Conference on Mechatronics Engineering (ICOM), Putrajaya, Malaysia, 30–31 October 2019; pp. 1–6.
23. Mathworks. QP Solver. 2018. Available online: https://www.mathworks.com/help/mpc/ug/qp-solver.html (accessed on 7 April 2022).
24. Yuan, Q.; Zhan, J.; Li, X. Outdoor flocking of quadcopter drones with decentralized model predictive control. *ISA Trans.* **2017**, *71*, 84–92. [CrossRef] [PubMed]