Automatika

Journal for Control, Measurement, Electronics, Computing and Communications

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/taut20

An effective proportional-double derivative-linear quadratic regulator controller for quadcopter attitude and altitude control

Mohamad Norherman Shauqee, Parvathy Rajendran & Nurulasikin Mohd Suhadis

To cite this article: Mohamad Norherman Shauqee, Parvathy Rajendran & Nurulasikin Mohd Suhadis (2021) An effective proportional-double derivative-linear quadratic regulator controller for quadcopter attitude and altitude control, Automatika, 62:3-4, 415-433, DOI: 10.1080/00051144.2021.1981527

To link to this article: https://doi.org/10.1080/00051144.2021.1981527

Published online: 23 Sep 2021.

AUTOMATIKA
2021, VOL. 62, NOS. 3–4, 415–433
https://doi.org/10.1080/00051144.2021.1981527

Taylor & Francis
Taylor & Francis Group

OPEN ACCESS Check for updates

An effective proportional-double derivative-linear quadratic regulator controller for quadcopter attitude and altitude control

Mohamad Norherman Shauqee, Parvathy Rajendran and Nurulasikin Mohd Suhadis

School of Aerospace Engineering, Universiti Sains Malaysia Nibong Tebal, Malaysia

ABSTRACT

A quadcopter control system is a fundamentally difficult and challenging problem because its dynamics modelling is highly nonlinear, especially after accounting for the complicated aerodynamic effects. Plus, its variables are highly interdependent and coupled in nature. There are six controllers studied and analysed in this work which are (1) Proportional–Integral–Derivative (PID), (2) Proportional-Derivative (PD), (3) Linear Quadratic Regulator (LQR), (4) Proportional-Linear Quadratic Regulator (P-LQR), (5) Proportional-Derivative-Linear Quadratic Regulator (PD-LQR) and lastly (6) the proposed controller named Proportional-Double Derivative-Linear Quadratic Regulator (PD2-LQR) controller. The altitude control and attitude stabilization of the quadcopter have been investigated using MATLAB/Simulink software. The mathematical model of the quadcopter using the Newton–Euler approach is applied to these controllers has illuminated the attitude (i.e. pitch, yaw, and roll) and altitude motions of the quadcopter. The simulation results of the proposed PD2-LQR controller have been compared with the PD, PID, LQR, P-LQR, and PD-LQR controllers. The findings elucidated that the proposed PD2-LQR controller significantly improves the performance of the control system in almost all responses. Hence, the proposed PD2-LQR controller can be applied as an alternative controller of all four motions in quadcopters.

ARTICLE HISTORY
Received 9 June 2020
Accepted 10 September 2021

KEYWORDS
Quadcopter; UAV controller; proportional-double derivative-linear quadratic regulator (PD2-LQR) controller; linear quadratic regulator (LQR) controller; attitude motion; altitude motion

1. Introduction

In recent years, the popularity of the small-scale Unmanned Aerial Vehicle (UAV) such as quadcopter has drastically increased due to its advantages and wide range of military and civilian applications. For instance, wildfire surveillance [1], search and rescue mission, and hazardous environment monitoring [2]. With these UAVs, the risk of human lives in a dangerous scenario can be minimized or avoided [3]. Compare to the fixed-wing UAV, a quadcopter has the ability to do a vertical take-off and landing (VTOL), or short take-off and landing (STOL) and hovering flight capabilities. With these capabilities, the quadcopter does not require a runaway for take-off and landing like a fixed-wing UAV, it just simply directs vertical take-off and landing in a limited space, plus the hovering capability give a great advantage if static image capturing is needed [4].

However, to control the quadcopter is difficult and challenging since the system is strongly coupled and nonlinear [5]. In addition, it is also severely under-actuated since it has six degrees of freedom (three translational and three rotational) that need to be controlled by only four independent inputs (rotor speeds), which brings about the complexity in its position and attitude control [3,6,7]. The precise controller design is needed to control the quadcopter system because of its high maneuverability capability. Parameter perturbation and model uncertainties have brought another difficulty with the quadcopter control system. Besides, stabilization within a short period with an acceptable precision level is needed for all parameters due to quadcopter's fast dynamics.

Before the controller can be designed, first, the dynamics model of the quadcopter must be derived and studied. Basically, there are several types of formulism can be used for the derivation, but the most commonly used by the researchers are based on the Newton–Euler or Euler–Lagrange formulism [8]. Thus, in this paper, the dynamic modelling of the quadcopter was done using the Newton–Euler formulation approach. The system that was modelled by any approach can be controlled by various methods such as PID control [8–10], LQR control [11–13], for the linear controller or Backstepping Control (BC) [14,15], Sliding Mode Control (SMC) [16–18] for the nonlinear controller to name a few. However, the linear controller like PID is the most commonly used for the quadcopter control and stabilization due to its simplicity, easy to design, and can provide sufficient performance [19].

Li and Li [20] designed a PID controller to analyse the dynamic characteristics of the quadrotor. The controller was used to control the position and orientation

CONTACT Parvathy Rajendran aeparvathy@usm.my School of Aerospace Engineering, Universiti Sains Malaysia, Engineering Campus, 14300 Nibong Tebal, Pulau Pinang, Malaysia

of the quadrotor in 3D space. The simulation conducted found that the PID controller can achieve a stable and good response in all the states. It produces a fast response with almost zero steady-state error. However, the system experiences a slight overshoot in all the states. In order to test the robustness of the controller against the disturbance, an actual experiment was conducted, and it shows that the PID controller can adjust the changes in the state when disturbed by wind or other disturbances.

Pan et al. [21] conducted a study of an optimal PID controller based on Qball-X4 quadrotor developed by Quanser company as the experimental platform for trajectory tracking control. They designed a corresponding Kalman Filter to estimate the target trajectory in consideration of tracking error and delays. The finding shows that the system can reach desired altitude height in an acceptable period without steady-state error but produce a high overshoot. The tracking error under the disturbance greatly improved when the Kalman filter was added. In another work by Tanveer et al. [22], the PID controller with Extended Kalman Filter was used to control and stabilize the quadrotor's altitude and attitude angle. Extended Kalman Filter was used to filter out the sensor noise and system noises to stabilize the quadrotor's altitude and attitude angle. In their findings, the controller can quickly stabilize the altitude of the quadrotor within 3.8 s. It can also work, handle disturbance very well, and promptly stabilize the quadrotor's attitude angle within 5.8 s.

Ahmad et al. [23] presented a comparative study of two classical PD and PID controllers for controlling quadrotor's altitude and attitude angle. The simulation result shows that both controllers have a fast response within approximately less than 1 s in all states. Still, the PD controller gives a better response in attitude control in terms of settling time, which is 2 s, while the PID controller is 11 s with slight overshoot. However, the PID controller gives a better result in altitude control with zero steady-state error, while the PD controller has a 1.25% steady-state error. Both controllers exhibit approximately a 5% overshoot in altitude motion.

Burggräf et al. [9] presented mathematical modelling of the quadrotor based on the cross configuration. They stated that cross structure allows the quadrotor to have six DOF movements while commanded by only four given inputs. In this paper, they implemented a cascaded P-PID controller to control and stabilize the quadrotor attitude. The simulation result shows that the system's overshoot is less than 25% in roll, pitch, and yaw motion, respectively, and the controller produces a very fast settling time around 1.3 s. Other than that, they also perform an experimental study to test the controller's robustness against external disturbances such as wind and collision. They found that the controller can stabilize the quadrotor in just 2.2 s after being exposed to the external disturbance.

Zhi et al. [24] proposed an optimal LQR controller combine with Kalman Filter to control the quadrotor's attitude angle. They conducted the simulation for noise-free and under noise conditions to appreciate Kalman Filter's performance in rejecting the noise. The performance of the LQR controller is then compared with the classical PID controller. The simulation result shows that both controllers can meet the system's requirement under the noise-free condition with no steady-state error. However, the LQR controller does not produce an overshoot and smoother response, while PID has a slight overshoot up to 0.2°. Under the noisy condition, the LQR controller with Kalman Filter can maintain its performance quality than the LQR controller without Kalman Filter. The attitude response error can be reduced from $\pm 0.6°$ to $+0.2$: $-0.3°$ when Kalman Filter is added.

Hernandez-Martinez et al. [25] performed a study of trajectory tracking control of the quadrotor using a two-level control strategy, which consists of an external loop to control the translational dynamics and generate the desired trajectories of the attitude orientation controlled by the inner loop. In order to achieve the maximum flight control time, energy-saving is addressed in designing the optimal control laws. The simulation result shows that the quadrotor can follow the desired trajectory with the desired attitude angle value. The tracking error converges to zero within an acceptable time.

Okyere et al. [11] presented a step-by-step design of the LQR controller for controlling the altitude motion of the quadrotor. In this paper, the authors demonstrated the effect of the weighting matrices Q and R on the controller's gain K values that simultaneously affect the system's state response. They found that if R is high, K will be small, and the response will be slower and vice-versa. If Q is high, then K will be high, and the response will be faster and vice-versa. The simulation result shows that when K is lower, the state response time will become slower to reach zero, but the controller response becomes faster, and it will produce overshoot and steady-state error. When K is higher, the state response will be faster to reach zero, but the controller response becomes slower and produce a steady-state error. Therefore, choosing a suitable gain value K is vital to get a good system performance. In this paper, the controller gain is settled on [1 1.7321] since it has no overshoot, no steady- state error, and the system was good.

In another work by Martins et al. [12], an optimal LQR controller with an Integral action is proposed for trajectory tracking control of the quadrotor. A Kalman Filter is added to estimate the quadrotor state that relies on the measurement from motion sensors installed onboard. In this paper, the quadrotor's dynamic models based on X-configuration were derived and linearized for the trim position. The simulation result shows that good trajectory tracking was achieved, and the response converges to the desired point without error. However, some error was observed in the yaw response, but it does not exceed 1/100°. More than that, some deviation within the acceptable range has occurred when implementing in the actual system.

Shah et al. [13] provided a study of trajectory tracking control for the quadrotor using Gain Scheduled Integral LQR. The integral term is added to improve the tracking performance of the quadrotor by minimizing the steady-state error of the system. In this paper, several trajectories were performed with significant variation of yaw angle. In order to appreciate the performance of the proposed controller, two simulations were conducted: (1) tracking without integral term and (2) tracking with the integral term. Based on their findings, the continuous gain control law can overcome the chattering and discontinuity problem. The tracking error can be minimized. The yaw angle's steady-state error improved by 95% (4.3854–0.222) and 56.6% (0.0865–0.0037) for cases 1 and 2, respectively. For altitude response, 94.8% decreased (0.1155–0.0059) for case 1 and a slight increase by 4% (0–0.04) for case 2.

Kumar et al. [26] presented a differential flatness-based hybrid controller for the quadrotor trajectory tracking problem. In the paper, the combination of the conventional PID and LQR controller was designed. The author indicates that the hybrid controller could balance the stability and maneuvrability of the quadrotor, which makes it suitable for the complex trajectory tracking problem. For validation purposes, a comparative study between conventional PID and LQR controllers was made by the author. The simulation results reveal that the quadrotor could track the desired trajectory with a satisfactory performance when using the hybrid controller compared to the other controller, especially the PID controller, where it cannot accurately follow the desired trajectory. The error of the position and the attitude angle is very minimal for the hybrid controller.

A hybrid P-LQR controller has been proposed by Yit and Rajendran [27]. In this paper, the hybrid P-LQR controller was designed to improve the UAV's longitudinal motion performance over the PID and LQR controllers. The simulation results show that this approach improved the rise time, settling time, and RMSE, but the overshoot and steady-state error slightly increases, still within the acceptable limit constraint. The PD-LQR is another type of hybrid controller that being studied by Yit et al. [19]. The simulation result shows that the hybrid PD-LQR controller performs with great success. The controller's improvement in rising time, settling time, RMSE, and steady-state error is 95.6%, 95.5%, 49.3%, and 0%, respectively. In terms of overshoot, the hybrid PD-LQR controller still lags behind the hybrid P-LQR controller with just a 0.001% difference, and the LQR controller has the highest percentage of overshoot with 0.351%.

In summary, this section has provided several literatures that perform a linear control technique for the UAV control and stabilization. The main objectives of all of these works are that to design an appropriate control technique to improve the performance of the UAV. For this particular reason, in this paper, we proposed a hybrid controller by combining the Proportional-Double Derivative (PD2) controller with the LQR controller to become the PD2-LQR controller. As far as we know, there are no such hybrid controller are being designed and applied to the quadcopter control and stabilization application in the literatures. The work that we find almost similar to our work is that proposed by [19], which using a PD-LQR controller that was applied for the fixed-wing UAV. Besides, it was proven through the simulation that by using this combination the performance of the quadcopter can be further improved. Through this work, the simulation was done by using the MATLAB/Simulink software, and all the essential parameter needed to design the dynamic model of the quadcopter was determined and collected. A comparative study of the response characteristic of the proposed controller with the conventional controller, namely PD, PID, LQR, and hybrid controller, namely P-LQR, and PD-LQR was presented and analysed to highlight the performance of the proposed controller. Four PD2-LQR controllers were designed for altitude (z) and attitude (roll, pitch, and yaw) motion. Based on the simulation results, the developed hybrid PD2-LQR controller is best-suited to control and stabilize the states of the quadcopter with a better dynamic performance in terms of rising time, settling time, percentage overshoot, steady-state error, and RMSE.

The rest of this paper is arranged as follows; in Section 2, the dynamic modelling of the quadcopter based on the Newton–Euler approach and its associated parameters are presented. The development of the control design techniques are presented in Section 3. In Section 4, the technique for the simulation is described. In Section 5, the simulation results are discussed. Lastly, a concluding remark based on the proposed controller is presented in Section 6.

2. Dynamic modelling of quadcopter

The first step in designing the dynamic model of the quadcopter is to define the inertial frame and body frame in the three-dimensional space, each with its defined right-handed coordinate system, as shown in Figure 1. In this way, the attitude and the position of the quadcopter can be controlled in a three-dimensional space. The inertial frame or earth frame, denoted by E, is used to describe the absolute position in the space. The origin of the coordinate system E is fixed on a

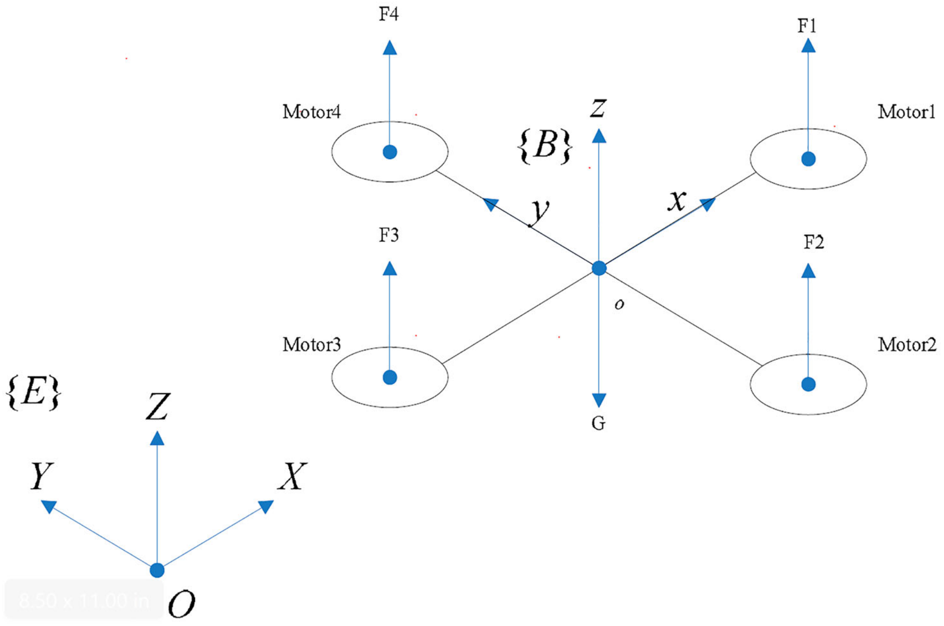

Figure 1. Structure model of the quadcopter with a reference frame [20].

ground surface or a specific point in space and the initial position of the quadcopter.

The designation of the quadcopter heading is in the positive direction of the OX-axis and perpendicular to the OYZ plane. The OY-axis is perpendicular to the OXZ plane and the OZ-axis perpendicular to the OXY plane and pointing vertically upward. The relative movement of the ground and the quadcopter is studied using this coordinate system. The centre of the quadcopter is the origin of the quadcopter coordinate system $OXYZ$ denoted by B. The positive OX-axis is pointing toward rotor 1, the positive OY-axis is pointing toward rotor 4 and OZ-axis is pointing vertically upward against the gravity. These two coordinate systems can be converted to each other through transition matrix R.

Based on this reference frame, $[xyz]^T$ is defined as the translational position and $[\phi\theta\psi]^T$ is defined as an angular position. The location of the quadcopter to the inertial frame is indicated by the translational position. The angular position is defined by the Euler angle. Roll angle, ϕ, is referred to rotating angle around the OX-axis, pitch angle, θ, is refer to rotating angle around the OY-axis and yaw angle, ψ, is refer to rotating angle around the OZ-axis. Thus, the rotational matrixs R_x, R_y, and R_z from the body frame B to the inertial frame E can be obtained as in the following equations [20]:

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \tag{1}$$

$$R_y = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{2}$$

$$R_z = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

By multiplying these three rotational matrices R_x, R_y, and R_z, we finally can obtain the rotational matrix R of the body frame relative to the inertial frame as in the following equation [20]:

$$R = R_x.R_y.R_z = \begin{bmatrix} \cos\psi\cos\phi & \cos\psi\sin\theta\sin\phi & \cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi \\ \sin\psi\cos\theta & \sin\psi\sin\theta\sin\phi & \sin\psi\sin\theta\cos\phi - \sin\phi\cos\psi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix} \tag{4}$$

Similarly, the relationship between the Euler rate $(\dot{\phi}, \dot{\theta}, \dot{\psi})$ in the inertial frame and the angular velocity in the body frame (p, q, r) can be determined through the transformation matrix given in the following equation [28]:

$$T = \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & -\sin\phi & \cos\phi\cos\theta \end{bmatrix} \tag{5}$$

On the other hand, for transforming the angular velocity from the body frame to the inertial frame, an inverse transformation matrix of Equation (5) can be performed. Around the hover position, the angle of ϕ and θ is close to 0; thus, the transformation matrix T can be approximate as a unit matrix. In this case, the relation between the Euler rate and the angular velocity in the body frame can be approximated as linear as in the following equation according to Reizenstein [29]:

$$\begin{bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{bmatrix} \cong \begin{bmatrix}p\\ q\\ r\end{bmatrix} \tag{6}$$

2.1. Mathematical derivation

Before the derivation of the mathematical model of the quadcopter can be modelled, the following assumptions need to be made [20]. The assumptions are (1) the structure of the quadcopter is symmetrically rigid, (2) the geometric centre and centroid of the quadcopter are in the same position with the origin of the inertial coordinate system, (3) flight altitude and other factors do not affect the resistance and gravity of the quadcopter and (4) thrust and drag forces are proportional to the square of the propeller speed.

2.2. Kinematic equation

Using Newton and Euler equations for the three-dimensional motions of a rigid body, the mathematical equation of the quadrotor will be presented. The vector that contains the linear and angular position of the quadrotor in the inertial frame is defined as $[XYZ\phi\theta\psi]^T$ and the vector that holds the linear and angular velocity of the quadrotor in the body frame is defined as $[uvwpqr]^T$. The relationship between the body frame and the inertial frame of the quadrotor in the three-dimensional space are linked by the following equations [30]:

$$v = R.v_B \tag{7}$$

$$\omega = T.\omega_B \tag{8}$$

where

$$v = [\dot{x}\dot{y}\dot{z}]^T \in \mathbb{R}^3$$
$$\omega = [\dot{\phi}\dot{\theta}\dot{\psi}]^T \in \mathbb{R}^3$$
$$v_B = [uvw]^T \in \mathbb{R}^3$$
$$\omega_B = [pqr]^T \in \mathbb{R}^3$$

By solving Equations (7) and (8), the kinematics equation of the quadrotor is obtained as given in the following equation [31]:

$$\begin{aligned}\dot{x} &= w[\sin\phi\sin\psi + \cos\phi\cos\psi\sin\theta]\\ &\quad - v[\cos\phi\sin\psi - \cos\psi\sin\phi\sin\theta]\\ &\quad + u[\cos\psi\cos\theta]\\ \dot{y} &= v[\cos\phi\cos\psi + \sin\phi\sin\psi\sin\theta]\\ &\quad - w[\cos\psi\sin\phi - \cos\phi\sin\psi\sin\theta]\\ &\quad + u[\cos\theta\sin\psi]\\ \dot{z} &= w[\cos\phi\cos\theta] - u[\sin\theta] + v[\cos\theta\sin\phi]\\ \dot{\phi} &= p + r[\cos\phi\tan\theta] + q[\sin\phi\tan\theta]\\ \dot{\theta} &= q[\cos\phi] - r[\sin\phi]\\ \dot{\psi} &= r\frac{\cos\phi}{\cos\theta} + q\frac{\sin\phi}{\cos\theta}\end{aligned} \tag{9}$$

2.3. Dynamic equation

The dynamics equation of the quadrotor has two components, which are translational components and rotational components. In translational components, it consists of the Z altitude, the X position, and Y position, while in rotational components, it consists of the roll (ϕ), the pitch (θ) and the yaw (ψ) motion. The derivation is done as according to the assumption stated before, and the general formalism of the Newton–Euler equation is presented as follows [31]:

$$\begin{bmatrix}\sum F\\ \sum M\end{bmatrix} = \begin{bmatrix}mI & 0\\ 0 & I\end{bmatrix}\begin{bmatrix}\dot{v}\\ \dot{\omega}\end{bmatrix} + \begin{bmatrix}\omega \times mv\\ \omega \times I\omega\end{bmatrix} \tag{10}$$

where is the mass of the quadrotor, I is the inertia, v is the linear velocity, ω is the angular velocity, F is the forces acting on the centre of the quadrotor's body, and M is the torque acting on the quadrotor's body.

2.3.1. Translational dynamic equation

Since the quadrotor is considered as a rigid body, according to Newton's second law of motion, the total forces acting on the centre of gravity of the quadrotor in the body frame can be represented as the following equation [31]:

$$F_B = m\dot{v}_B + \omega_B \times mv_B \tag{11}$$

where m is the mass of the quadrotor, and $F_B = [F_xF_yF_z]^T \in \mathbb{R}^3$ are the total forces acting on the centre of gravity of the quadrotor. Thus, solving Equation (11) yield the following equation:

$$\begin{aligned}F_x &= m(\dot{u} + qw - rv)\\ F_y &= m(\dot{v} - pw + ru)\\ F_z &= m(\dot{w} + pv - qu)\end{aligned} \tag{12}$$

2.3.2. Rotational dynamic equation

Likewise, according to Euler's equation, the total torques acting on the centre of gravity of the quadrotor in the body frame can be represented as follows [31]:

$$M_B = I.\dot{\omega}_B + \omega_B \times (I.\omega_B) \tag{13}$$

where $M_B = [M_xM_yM_z]^T \in \mathbb{R}^3$ are the total torques acting on the centre of gravity of the quadrotor, and I is the diagonal inertia matrix due to the assumption that quadrotor's structure is symmetrical. Thus, solving Equation (13) yield the following equation:

$$I = \begin{bmatrix}I_x & 0 & 0\\ 0 & I_y & 0\\ 0 & 0 & I_z\end{bmatrix} \in \mathbb{R}^{3\times3} \tag{13.1}$$

$$
\begin{aligned}
M_x &= \dot{p}I_x - qrI_y + qrI_z \\
M_y &= \dot{q}I_y + prI_x - prI_z \\
M_z &= \dot{r}I_z - pqI_x + pqI_y
\end{aligned} \tag{14}
$$

These equations are valid as long as the origin of the body frame's axes coincides with quadrotor's centre of gravity.

2.3.3. Forces Applied on the quadcopter

The gravitational force naturally exists in the inertial frame and acts at the centre of quadrotor's centre of gravity. This force only exists in the Z-axis and pointing in the downward direction and defined as follows:

$$
F_g = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} \tag{15}
$$

To transform the gravitational force from inertial frame to body frame, Equation (15) is multiplied with the rotation matrix R and represented as follows:

$$
F_g^B = R_B^E \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} = -mg \begin{bmatrix} -\sin\theta \\ \sin\phi\cos\theta \\ \cos\phi\cos\theta \end{bmatrix} \tag{16}
$$

where m is the mass of the quadrotor and g is the gravitational acceleration.

The second force exerted on the quadrotor is the thrust force. Thrust force is the lifting force produced by the rotation of the propellers that move the quadrotor in the positive vertical direction of the Z-axis. Since the quadrotor has four rotors that drive the vehicle, and to achieve a vertical motion, all rotors must produce the same amount of thrust, and the total thrust produce is simply the addition of the thrust in each rotor. The total thrust is defined as follows:

$$
F_T = \sum_{i=1}^{4} T_i = T_1 + T_2 + T_3 + T_4 \tag{17}
$$

where T_i is the thrust produced by the ith propeller. Since the thrust is generated from the rotation speed of the propellers, thus the equation of the thrust is defined as follows [31]:

$$
T_i = C_T \rho A r^2 \Omega_i^2 \tag{18}
$$

However, since the quadrotor is flying at a limited altitude, the air density can be considered constant, and to comply with the assumption stated earlier, Equation (18) can be simplified as follows [32]:

$$
T_i = b\Omega_i^2 \tag{19}
$$

where C_T is the thrust coefficient, ρ is the air density, A is the area of the blade, r is the radius of the blade, Ω_i is the angular speed of the ith rotor, and b is the thrust factor. Therefore, the total thrust force expressed in the body frame is as follows:

$$
\begin{aligned}
F_T &= b\Omega_1^2 + b\Omega_2^2 + b\Omega_3^2 + b\Omega_4^2 \\
&= b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2)
\end{aligned} \tag{20}
$$

2.3.4. Torque Applied on the quadcopter

Since the rotor is located at a certain distance from quadrotor's centre, it will create torque in a different rotation axis. As seen in Figure 2, each rotor will create a moment in the direction opposite to the rotor's rotation.

Around the x-axis, by using the right-hand rule with the axis of rotation, rotors 1 and 3 did not produce any moment on the x-axis, but rotors 2 and 4 did. Thus, the total moment about the x-axis can be expressed as follows:

$$
\begin{aligned}
M_{roll} &= -lT_2 + lT_4 = -lb\Omega_2^2 + lb\Omega_4^2 \\
&= lb(-\Omega_2^2 + \Omega_4^2)
\end{aligned} \tag{21}
$$

Around the y-axis, by using the right-hand rule with the axis of rotation, rotors 2 and 4 did not produce any moment on the y-axis, but rotors 1 and 3 did. Thus, the total moment about the y-axis can be expressed as follows:

$$
\begin{aligned}
M_{pitch} &= -lT_1 + lT_3 = -lb\Omega_1^2 + lb\Omega_3^2 \\
&= lb(-\Omega_1^2 + \Omega_3^2)
\end{aligned} \tag{22}
$$

where l is the distance of the rotor to the centre of gravity of the quadrotor. Around the z-axis, the moment generated is the result of the rotation of the rotors itself and not because of the thrust force produce by the propellers. The reaction torque of the rotor is given as follows:

$$
Q_i = d\Omega_i^2 \tag{23}
$$

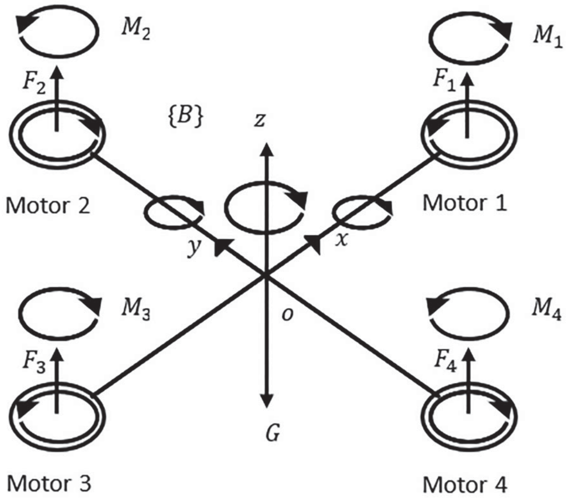

Figure 2. Forces and moments acting on the quadrotor.

Thus, the total moment about the z-axis can be expressed as follows:

$$\begin{aligned} M_{yaw} &= T_1 - T_2 + T_3 - T_4 \\ &= d\Omega_1^2 - d\Omega_2^2 + d\Omega_3^2 - d\Omega_4^2 \\ &= d(\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2) \end{aligned} \tag{24}$$

where d is the drag factor.

The gyroscopic moment is a physical effect produced by the propellers' rotation and the rotation of quadrotor's body. The gyroscopic moment is given by

$$M_G = \omega \times J_r \begin{bmatrix} 0 \\ 0 \\ -(1)^i\Omega_i \end{bmatrix} = \begin{bmatrix} qJ_r\Omega_r \\ -pJ_r\Omega_r \\ 0 \end{bmatrix} \tag{25}$$

where ω is the propeller's speed, J_r is the rotor's inertia, and Ω_r is the resultant angular velocity defined as $\Omega_r = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4$.

2.4. Final quadcopter model

The complete quadrotor mathematical model including the control input vector is presented in this subsection.

2.4.1. Control input vector, U

By controlling the rotational speed of the rotor, one can control the movement of the quadrotor. The rotational speed of the rotor is chosen as the input to the quadrotor system. Four control inputs that controlled the quadrotor system are defined as follows:

$$U = \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} \tag{26}$$

where

$$\begin{aligned} U_1 &= F_t = b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\ U_2 &= M_{roll} = lb(-\Omega_2^2 + \Omega_4^2) \\ U_3 &= M_{pitch} = lb(-\Omega_1^2 + \Omega_3^2) \\ U_4 &= M_{yaw} = d(\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2) \end{aligned} \tag{27}$$

The control inputs in Equation (27) can be represented in matrix form as

$$\begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} b & b & b & b \\ 0 & -lb & 0 & lb \\ -lb & 0 & lb & 0 \\ d & -d & d & -d \end{bmatrix} \begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} \tag{28}$$

From the above matrix Equation (28), U_1 is the control input responsible for the altitude motion of the quadrotor, consisting of the four rotors' speed which caused the upward thrust force. U_2 is the control input responsible for the roll motion of the quadrotor; it is the difference in thrust between rotor 2 and rotor 4. U_3 is the control input responsible for the pitch motion of the quadrotor; it is the difference in thrust between rotor 1 and rotor 3. Finally, U_4 is the control input responsible for the yaw motion of the quadrotor; it is the difference in torque between the pair of clockwise rotating rotors and the pair of counter-clockwise rotating rotors. If the rotor's speed is required to be calculated from the control inputs, an inverse relationship between the rotor's speed and control input can be performed by inverting the above matrix Equation (28) and can be represented as follows:

$$\begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} = \begin{bmatrix} \frac{1}{4b} & 0 & -\frac{1}{2b} & \frac{1}{4d} \\ \frac{1}{4b} & -\frac{1}{2b} & 0 & -\frac{1}{4d} \\ \frac{1}{4b} & 0 & \frac{1}{2b} & \frac{1}{4d} \\ \frac{1}{4b} & \frac{1}{2b} & 0 & -\frac{1}{4d} \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} \tag{29}$$

2.4.2. Translational equation of motion

The complete translational dynamics equation, after considering the gravitational force and thrust force is become,

$$\sum F = ma = m \begin{bmatrix} \ddot{X} \\ \ddot{Y} \\ \ddot{Z} \end{bmatrix} = F_g + F_T \tag{30}$$

Substituting the value for F_g and F_T from Equations (16) and (20) into Equation (30), the following equation is obtained:

$$m \begin{bmatrix} \ddot{X} \\ \ddot{Y} \\ \ddot{Z} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + R_E^B \begin{bmatrix} 0 \\ 0 \\ b\sum_{i=1}^{4}\Omega_i^2 \end{bmatrix} \tag{31}$$

Finally, rewriting Equation (31) and solve for the linear acceleration, we can obtain in the following equation:

$$\begin{aligned} \ddot{X} &= (\cos\phi\cos\psi\sin\theta + \sin\phi\sin\psi)\frac{U_1}{m} \\ \ddot{Y} &= (\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi)\frac{U_1}{m} \\ \ddot{Z} &= (\cos\theta\cos\phi)\frac{U_1}{m} - g \end{aligned} \tag{32}$$

2.4.3. Rotational equation of motion

The complete rotational dynamics equation, after considering the torque acting on the quadrotor and the gyroscopic effect is become,

$$\sum M = I\dot{\omega} + \omega \times I\omega = M_B + M_G \tag{33}$$

where $M_B = [U_2 U_3 U_4]^T$. Substituting the value for M_B and M_G from Equations (21), (22), (24), and (25) into

Equation (33), the following equation is obtained:

$$\begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix} \begin{bmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} + \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \times \begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}$$

$$= \begin{bmatrix} U_2 \\ U_3 \\ U_4 \end{bmatrix} + \begin{bmatrix} qJ_r\Omega_r \\ -pJ_r\Omega_r \\ 0 \end{bmatrix} \tag{34}$$

Finally, rewriting Equation (34) and solve for the angular acceleration, we can obtain in the following equation:

$$\begin{aligned} \ddot{\phi} &= \left(\frac{I_y - I_z}{I_x}\right)\dot{\theta}\dot{\psi} + \left(\frac{J_r\Omega_r}{I_x}\right)\dot{\theta} + \frac{U_2}{I_x} \\ \ddot{\theta} &= \left(\frac{I_z - I_x}{I_y}\right)\dot{\phi}\dot{\psi} - \left(\frac{J_r\Omega_r}{I_y}\right)\dot{\phi} + \frac{U_3}{I_y} \\ \ddot{\psi} &= \left(\frac{I_x - I_y}{I_z}\right)\dot{\phi}\dot{\theta} + \frac{U_4}{I_z} \end{aligned} \tag{35}$$

Grouping together the translational and rotational equation of motion, the complete nonlinear dynamic equation of the quadrotor becomes

$$\begin{aligned} \ddot{X} &= (\cos\phi\cos\psi\sin\theta + \sin\phi\sin\psi)\frac{U_1}{m} \\ \ddot{Y} &= (\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi)\frac{U_1}{m} \\ \ddot{Z} &= (\cos\theta\cos\phi)\frac{U_1}{m} - g \\ \ddot{\phi} &= \left(\frac{I_y - I_z}{I_x}\right)\dot{\theta}\dot{\psi} + \left(\frac{J_r\Omega_r}{I_x}\right)\dot{\theta} + \frac{U_2}{I_x} \\ \ddot{\theta} &= \left(\frac{I_z - I_x}{I_y}\right)\dot{\phi}\dot{\psi} - \left(\frac{J_r\Omega_r}{I_y}\right)\dot{\phi} + \frac{U_3}{I_y} \\ \ddot{\psi} &= \left(\frac{I_x - I_y}{I_z}\right)\dot{\phi}\dot{\theta} + \frac{U_4}{I_z} \end{aligned} \tag{36}$$

However, since only four control inputs are available, thus only four motions were chosen to be control, which are the altitude (Z) and the attitudes (ϕ, θ, ψ). Therefore, the dynamic Equation (36) can be reduced to Equation (37) by omitting the translational movement in X and Y directions from the system.

$$\begin{aligned} \ddot{Z} &= (\cos\theta\cos\phi)\frac{U_1}{m} - g \\ \ddot{\phi} &= \left(\frac{I_y - I_z}{I_x}\right)\dot{\theta}\dot{\psi} + \left(\frac{J_r\Omega_r}{I_x}\right)\dot{\theta} + \frac{U_2}{I_x} \\ \ddot{\theta} &= \left(\frac{I_z - I_x}{I_y}\right)\dot{\phi}\dot{\psi} - \left(\frac{J_r\Omega_r}{I_y}\right)\dot{\phi} + \frac{U_3}{I_y} \\ \ddot{\psi} &= \left(\frac{I_x - I_y}{I_z}\right)\dot{\phi}\dot{\theta} + \frac{U_4}{I_z} \end{aligned} \tag{37}$$

From the nonlinear Equation (37), the system's solution is difficult to find because the trigonometric functions are related to each other in a no-elementary way [33]. Due to this factor, a simplification of the system based on a small angle approximation can be done to obtain a linear dynamic equation for the system. This simplification is made by approximating the sine function with its argument ($\sin\alpha = \alpha$), and the cosine function with the unity ($\cos\alpha = 1$). This approximation is valid as long as the alpha is small, where $\alpha = \phi, \theta, \psi$.

3. Controller design

The complex nonlinear dynamic of the system makes the control of the quadcopter becomes a difficult task, thus, a linear dynamics equation was used for the simulation. In this paper, several control strategies namely the conventional PID controller, PD controller, LQR controller, a hybrid P-LQR controller, and hybrid PD-LQR controller are studied and compared with the proposed controller, a hybrid PD2-LQR controller. These techniques are elaborated in the following section. The general form of a linear state space representation is given as $\dot{x} = Ax + Bu$ with $\dot{x}$ is the states vector, and u is the inputs vector as given in Equations (38) and (39).

State vector,

$$x = [z, \dot{z}, \phi, \dot{\phi}, \theta, \dot{\theta}, \psi, \dot{\psi}]^T \tag{38}$$

Input vector,

$$u = [u_1, u_2, u_3, u_4]^T \tag{39}$$

In a straight flight path, the longitudinal motion equation of the quadcopter is defined in Equation (40), and the coefficient matrix is given in Equation (41) [20].

$$\dot{x}_g = A_g x_g + B_g u_g \tag{40}$$

where state vector, $x_g = [\dot{x}, \dot{z}, q, \theta]^T$ and input vector, $u_g = [u_1, u_4]^T$.

$$A_g = \begin{bmatrix} x_{\dot{x}} & x_{\dot{z}} & x_q & x_\theta \\ z_{\dot{x}} & z_{\dot{x}} & z_q & z_\theta \\ m_{\dot{x}} & m_{\dot{z}} & m_q & m_\theta \\ 0 & 0 & 1 & 0 \end{bmatrix} B = \begin{bmatrix} x_t & x_\delta \\ 0 & z_\delta \\ 0 & m_\delta \\ 0 & 0 \end{bmatrix} \tag{41}$$

In a horizontal flight path, the motion equation of the quadcopter is defined in Equation (42), and the coefficient matrix is given in Equation (43) [20], where state vector, $x_h = [v, p, r, \phi, \psi]^T$ and input vector, $u_s = u_2$.

$$\dot{x}_h = A_h x_h + B_h u_h \tag{42}$$

$$A_g = \begin{bmatrix} y_v & y_p & y_r & y_\phi 0 \\ l_v & l_p & l_r & l_\phi 0 \\ n_v & n_p & n_r & n_\phi 0 \\ 0 & 1 & 0 & 0\,0 \\ 0 & 0 & 1 & 0\,0 \end{bmatrix} B_g = \begin{bmatrix} y_\delta \\ l_\delta \\ n_\delta \\ 0 \\ 0 \end{bmatrix} \tag{43}$$

The transfer function of the quadcopter in the roll, pitch, yaw, and altitude motion are given in Equations

Table 1. Quadcopter parameter.

Parameter Name	Parameter	Value	Unit
Mass	m	1.2	kg
Length	l	0.2	m
Thrust factor	k_t	3.13 e−5	Ns^2
Drag factor	k_d	7.5 e−7	Nms^2
Moment inertia x-axis	I_x	2.353 e−3	kg m^2
Moment inertia y-axis	I_y	2.353 e−3	kg m^2
Moment inertia z-axis	I_z	5.262 e−2	kg m^2

(44)–(47), respectively, based on the quadcopter parameter in Table 1 and according to the transfer function $G(s) = C(sI - A)^{-1}B + D$ [34].

$$G_2 = \frac{\phi}{u_2} = \frac{65s + 4560}{s^3 + 109s^2 + 1023s + 2935} \tag{44}$$

$$G_3 = \frac{\theta}{u_3} = \frac{56.95s + 4391}{s^3 + 105s^2 + 870s + 4430} \tag{45}$$

$$G_4 = \frac{\psi}{u_4} = \frac{105}{s^2 + 413s} \tag{46}$$

$$G_1 = \frac{z}{u_1} = \frac{\dot{z}}{su_1} = \frac{1.63}{s(s+5)} \tag{47}$$

3.1. PID controller

PID controllers are one of the most commonly used controllers in the industrial application due to its robustness and simple structure configuration. For the controller implementation, the parallel structure of PID in a continuous-time domain is chosen. The PID controller attempts to minimize the error by adjusting the input value to reduce the contrast between the measured value and the desired value. Proportional, integral, and derivative gains are the three involved parameter gains in this structure, as shown in Figure 3. The PID controller gains used in this paper are shown in Table 2. The structure of the PID controller can be expressed in the following equation:

Table 2. Gains for the PID controller.

Motion	K_p	K_i	K_d
Pitch	13.9934	7.3931	1.7023
Roll	4.6094	4.0755	0.5097
Yaw	100	1	1
Z	98.9629	1.7538 e−15	12.6053

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)\mathrm{d}\tau + K_d \frac{de(t)}{\mathrm{d}t} \tag{48}$$

where u is controller output, K_p is the proportional gain, K_i is the integral gain and K_d is the derivative gain.

3.2. PD controller

Other than the PID controller, the quadcopter system also can be stabilized using only the Proportional-Derivative (PD) controller configuration. As compared to the PID controller, it shows that the PD controller is better in terms of robustness [3]. Thus, the PD controller can be used to replace the PID controller (Figure 4) since its structure is simpler to design, and it gives the easiest way to control the highly uncontrolled quadcopter vehicle. The gains of the PD controller are shown in Table 3. The structure of the PD controller in its respective motion can be designed as in the following equations:

$$u_\phi = K_p(\phi_{ref} - \phi) + K_d(\dot{\phi}_{ref} - \dot{\phi}) \tag{49}$$

$$u_\theta = K_p(\theta_{ref} - \theta) + K_d(\dot{\theta}_{ref} - \dot{\theta}) \tag{50}$$

$$u_\psi = K_p(\psi_{ref} - \psi) + K_d(\dot{\psi}_{ref} - \dot{\psi}) \tag{51}$$

$$u_z = K_p(z_{ref} - z) + K_d(\dot{z}_{ref} - \dot{z}) \tag{52}$$

3.3. LQR controller

The LQR controller is a feedback controller with an optimal control technique. The state or output of the

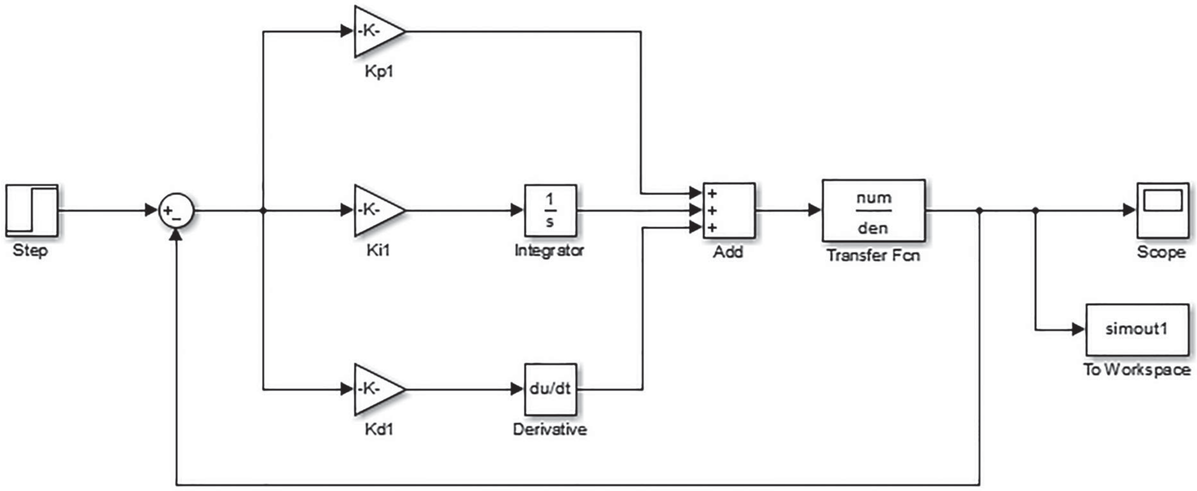

Figure 3. Simulink model of the PID controller.

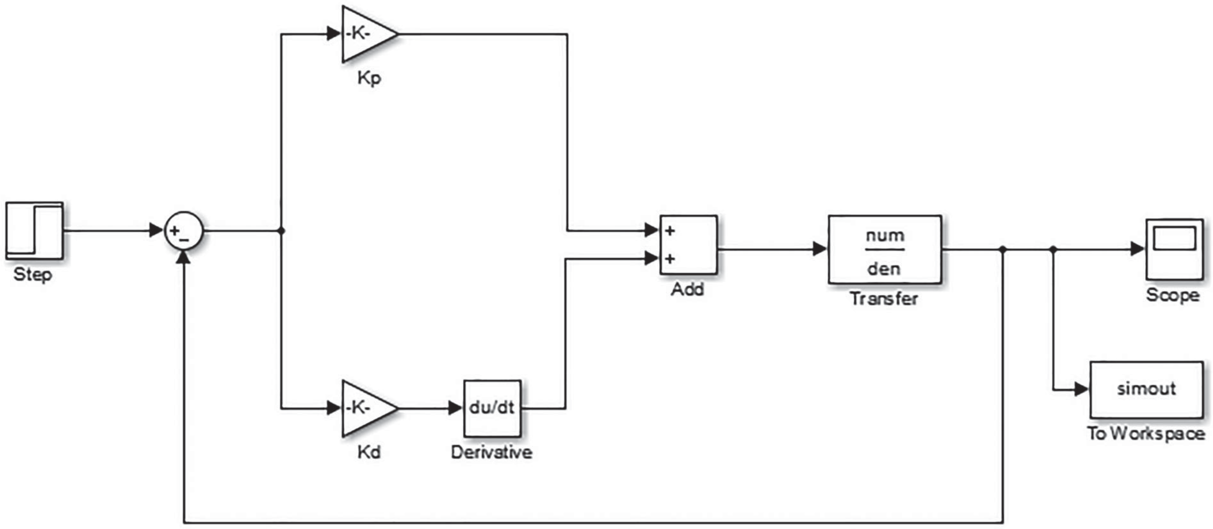

Figure 4. Simulink model of the PD controller.

Table 3. Gains for the PD controller.

Motion	K_p	K_d
Pitch	102.8174	10.6424
Roll	73.4042	8.1973
Yaw	30.8183	0.0820
Z	27.8877	3.1904

Table 4. Gains for the LQR controller.

Motion	Gain K	Q Matrix	R Matrix
Pitch	[0.0192 0.1162 0.0001]	$Q =$ diag([3.8 32.14 0.922])	$R =$ [1]
Roll	[0.0180 0.0675 0.0002]	$Q =$ diag([3.8 32.14 0.922])	$R =$ [1]
Yaw	[0.0183 5.6692]	$Q =$ diag([3.8 32.14])	$R =$ [1]
Z	[1.3355 5.6692]	$Q =$ diag([3.8 32.14])	$R =$ [1]

LQR controller is feedback and compared to the set-point value. The output is feedback through the controller's gain K [35]. The LQR system (Figure 5) can be expressed in a state variable form in the following equation [36,37]:

$$\dot{x} = Ax + Bu \tag{53}$$

To achieve the desired output, the controller's gain K needs to be obtained [35]. The optimal control vector for the state space form in the system is expressed as follows [36]:

$$u = -Kx \tag{54}$$

In order to determine an optimal control input, the minimization of the cost function is needed as shown in Equation (55) [38], where Q is a positive semi-definite symmetric matrix and R is a positive definite symmetric matrix. Q and R weighting matrices are chosen to control each state effectively using little control effort according to the performance index in the following equation:

$$J = \frac{1}{2}\int_0^\infty (x^TQx + u^TRu)\mathrm{d}t \tag{55}$$

The algebraic Riccati Equation (56) must be solved in order to calculate the optimal gain K [39], where the controller's gain K can be obtained using Equation (57) [40]. By using MATLAB software, gain K can be obtained using *LQR (A, B, C, D, Q, and R)* command. The gains of the controller are given in Table 4.

$$A^TP + PA + Q - PBR^{-1}B^TP = 0 \tag{56}$$

$$K = R^{-1}B^TP \tag{57}$$

3.4. P-LQR controller

To improve the performance of the quadcopter control system, a hybrid controller called the P-LQR controller

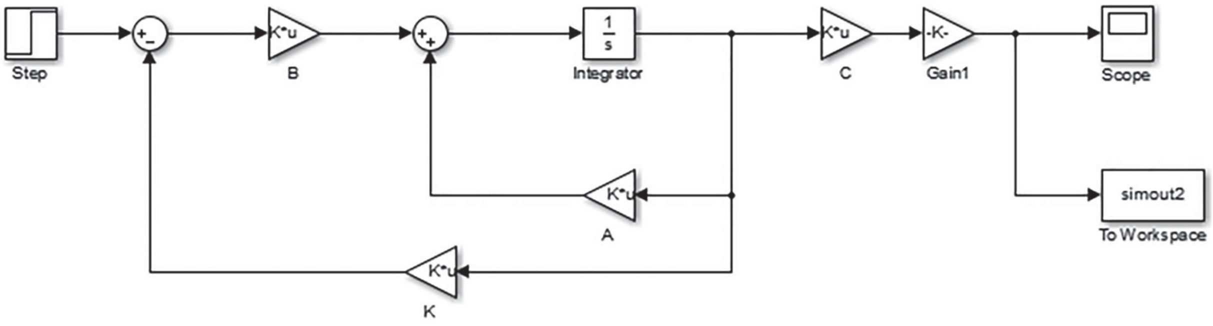

Figure 5. Simulink model of the LQR controller.

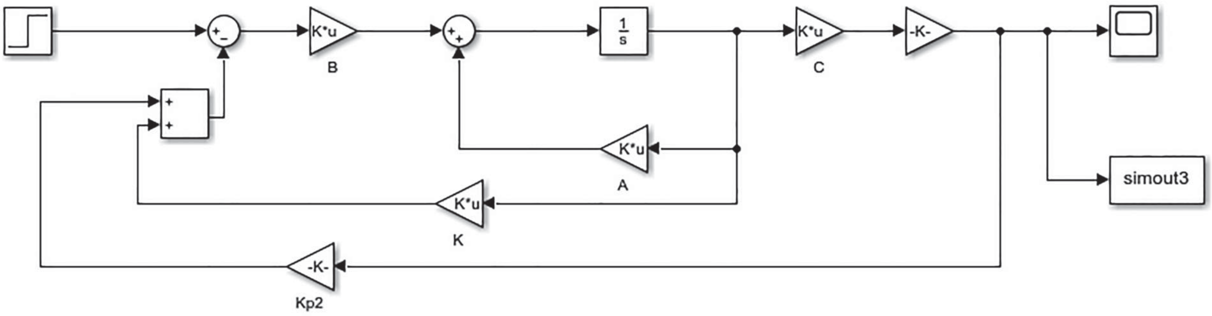

Figure 6. Simulink model of the P-LQR controller.

Table 5. Proportional gains for the P-LQR controller.

Motion	K_p
Pitch	0.0029
Roll	0.4058
Yaw	0.9890
Z	0.8305

Table 6. Proportional and derivative gains for the PDQR controller.

Motion	K_p	K_d
Pitch	0.9680	0.1238
Roll	0.8809	0.1047
Yaw	0.9907	0.1000
Z	0.9348	0.1188

was developed [41], as shown in Figure 6. This controller consists of the combination of the Proportional (P) controller with the LQR controller. This controller was designed by adding another close-loop as negative feedback in the LQR system. The LQR controller's gain K will be added with the P controller's gain K_P and form negative feedback to the system. This type of controller can perform better than the conventional LQR controller by shorten the rise time and setting time and improved the RMSE. However, this controller tends to increase the overshoot response of the system [41]. The proportional gains used for this controller are shown in Table 5.

3.5. PD-LQR controller

The PD-LQR controller is the combination of the Proportional-Derivative (PD) controller with the LQR controller (Figure 7). This controller was designed by adding another close-loop as negative feedback in the LQR system. The LQR controller's gain .. will be added with the PD controller's gain K_P, K_d and form negative feedback to the system. The PD-LQR controller was designed to improve the performance of the LQR controller and P-LQR controller in terms of its stability and overshoot response without affecting the desired output. PD controller was able to improve the response performance while maintaining the overshoot within a limit constraint [41]. The proportional and derivative gains in all states are given in Table 6.

3.6. PD2-LQR controller (Proposed controller)

The entire controller mentioned above can control the quadcopter; however, the autonomous UAVs need to have a fast response in rising time, settling time and very low overshoot. Besides, the steady-state error and RMSE must be close to zero as much as possible. Thus, various methods and approaches have been studied, and numerous simulations have been done. In all simulations done, it has been observed that the performance of the quadcopter could be improved by adding another Derivative (D) controller in the closed-loop as negative feedback in the PD-LQR controller system.

The proposed controllers (Figure 8) are the combination of the proportional-double derivative (PD2) controller with the conventional LQR controller becomes the PD2-LQR controller. By tuning the proportional and derivative gains of the controller accurately,

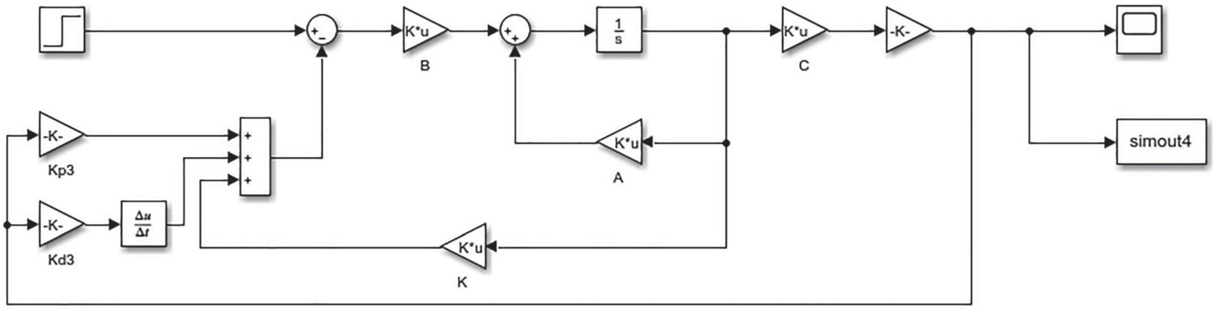

Figure 7. Simulink model of the PD-LQR controller.

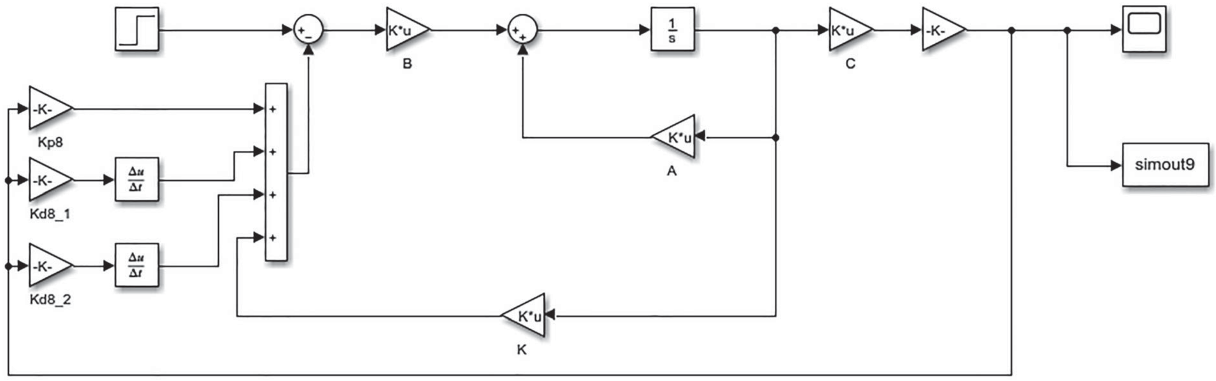

Figure 8. Simulink model of the PD2-LQR controller.

Table 7. Proportional and derivative gains for the PD-2LQR controller.

States	K_p	K_{d1}	K_{d2}
Pitch	0.9012	0.0548	6.6312e−4
Roll	0.9780	0.0069	0.0335
Yaw	0.9996	6.0278e−5	0.0010
Z	0.9977	0.0223	0.0058

the result obtained in Tables 8–12 in Section 5 shows that the proposed PD2-LQR controller can improve the quadcopter performance by reducing the rise time and settling time, the overshoot is within the limit constraint, and the steady-state error and RMSE are approximately close to zero as compared to other controllers. The gains used in this controller are shown in Table 7.

4. Simulation technique

In this section, the simulation technique used to determine the performance of each controller is discussed. First, the controllers' structure, as shown in Figures 3–8 was design using the Simulink block diagram. Then, a MATLAB program code was written to link up the MATLAB interface with the Simulink interface. Note that the simulation was run with a default sample time which is 0. The desired performance specification of the system is that, the overshoot requirement of the system aimed to be less than 10% [42] and the control precision to be lower than ±1% [43]. Moreover, the ideal performance for rising time, settling time, overshoot, steady-state error, and root mean square error (RMSE) to be close to zero as much as possible. The expected level of precision aimed to be ±0.001 s [44] due to its small size of the quadcopter. Each controller's gain parameters were tuned using the Gradient Descent Optimization method with Sequential Quadratic Programming [45], where this is an optimization algorithm based on a convex function with multi-variable parameters defined. This method refines its parameters iteratively to minimize a given function as far as possible to its local minimum and if the function satisfied within the pre-given constraint to ensure the convergence to a local minimum.

Initially, the transfer function of the quadcopter and the initial gain parameters values must be defined. These gain values must be defined as a set or array of value with a lower and upper limit, where in this work we have set between ±150. Then, the limit constraint of the required step response must be defined so that if the function did not satisfy the requirement, it would automatically stop the searching, where in this work the upper bound is limit to amplitude of 1 with 1% overshoot and the lower bound is limit to amplitude of 0 for $0 < t < 0.5$, 0.9 for $0.5 < t < 1$, and 0.99 for $1 < t < 3$. Next, the simulation to obtain the first set of gain value is executed. In this step, the gradient descent optimization method was used. At this stage, the optimum gain values that satisfy the constraint are being searched by iteratively adjusting the value until it finds the feasible solution that satisfy all constraints to within specific tolerances. After the first sets of gain values are obtained, these steps must be repeated to find a better step response with the optimum parameters gain.

The step for the optimization process is illustrated in Figure 9.

5. Result and discussion

In this section, the simulation result of all controllers in pitch, roll, yaw, and altitude motion are presented. In this research work, a countless effort is dedicated to finding a suitable controller with better performance on controlling the quadcopter. Conventional PID control, PD control, LQR control, P-LQR control, PD-LQR control, and PD2-LQR control algorithm are the six controllers assessed.

5.1. Pitching motion

The PID, PD, LQR, P-LQR, PD-LQR, and PD2-LQR controller in pitch motion response are shown in Figure 10. From the figure, the LQR controllers have the slowest response in terms of the rise time, while the

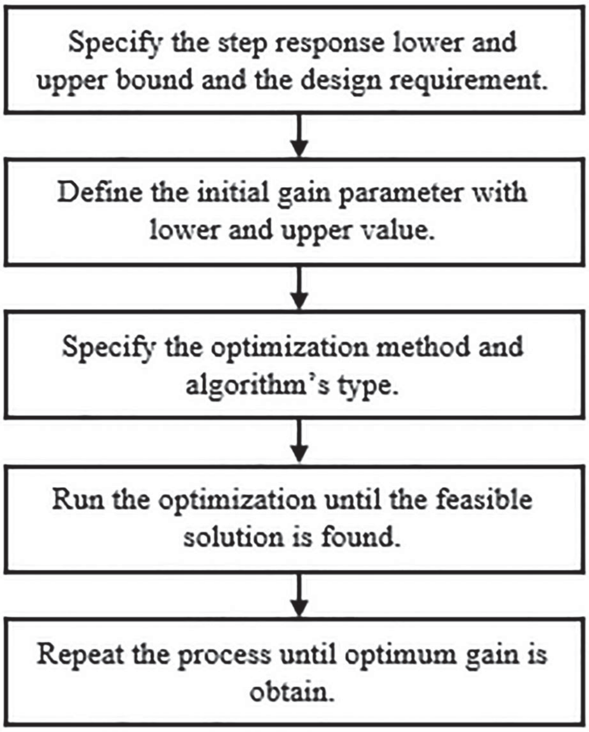
Specify the step response lower and upper bound and the design requirement.
Define the initial gain parameter with lower and upper value.
Specify the optimization method and algorithm's type.
Run the optimization until the feasible solution is found.
Repeat the process until optimum gain is obtain.

Figure 9. The steps for the optimization process.

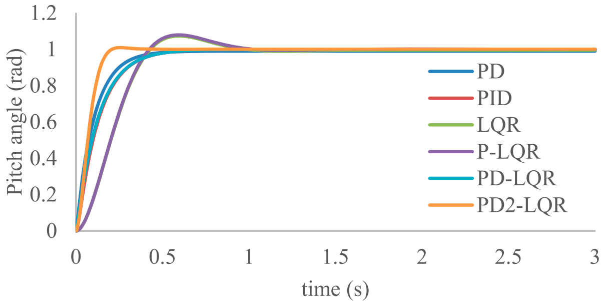

Figure 10. Response of various controllers in pitch motion.

PD2-LQR controller is the fastest. The PID and PD controller has an excellent performance in overshoot and RMSE respectively compared to other controllers. Even though the PID controller outperforms the PD2-LQR controller in terms of overshoot it worth to mention that the overshoot of the PD2-LQR controller is still within the acceptable limit constraint less than 10% in overshoot.

Table 8 present the comparison of the rise time, settling time, percentage overshoot, steady-state error, and RMSE of all the controllers in pitch motion. From Table 8, we can see that the PD2-LQR controller has the lowest rise time with only 0.11860 s, while the LQR controller is the highest with 0.28512 s. For the PD, PID, P-LQR, and PD-LQR controller, the rise time is 0.22726, 0.27697, 0.28436, and 0.27859 s, respectively. From these values, we can see that the percentage improvement of the PD2-LQR controller is as big as 58.4% in the rise time. The bold value in the table represents the best value of the respective controller with the respective response.

Table 8. Performance comparison various controllers in pitch motion.

Step info Controller	Rise time, s	Settling time, s	Overshoot, %	Steady-state error	RMSE
PD	0.22726	0.40718	0.00055	0.00968	**0.25126**
PID	0.27697	0.50334	**0**	5.36208e−05	0.34999
LQR	0.28512	0.88143	7.75500	0.00394	0.34661
P-LQR	0.28436	0.88006	7.81045	0.00087	0.34630
PD-LQR	0.27859	0.50166	0.00872	8.38531e−05	0.30598
PD2-LQR	**0.11860**	**0.18235**	0.91639	**2.58940e-05**	0.28600

In terms of settling time, the LQR controller is the highest with 0.88143 s, and the P-LQR controller is the second highest with only 0.00137 s indifference which is 0.88006 s. The settling time of the PD controller is 0.40718 s, which is lower by 0.09616 s than the PID controller. The settling time of the PD-LQR controller and PD2-LQR controller are 0.50166 and 0.18235 s, respectively, which make the PD2-LQR controller has a better performance in term of settling time. In terms of overshoot, the PID controller has the lowest percentage overshoot as compared to other controllers with 0% overshoot. In comparison, the P-LQR controller has the highest percentage overshoot with 7.81045% then followed by the LQR, PD2-LQR, PD-LQR, and PD controller with 7.75500%, 0.91639%, 0.00872%, and 0.00055%, respectively. However, the percentage overshoot of the proposed PD2-LQR is still within a 10% constraint limit that can be acceptable.

The steady-state error value of the PD, PID, LQR, P-LQR, PD-LQR, and PD2-LQR controller are 0.00968, 5.36208e−5, 0.00394, 0.00087, 8.38531e−5, and 2.58940e−5, respectively. From these values, we can see that the PD2-LQR controller has the lowest steady-state error as compared to other controllers, while the PD controller has the highest steady-state error. The RMSE of the PD controller is the lowest with 0.25126, followed by the PD2-LQR, PD-LQR, P-LQR, LQR, and PID controller, respectively. The bold value in the table represents the best value of the respective controller with the respective response.

The comparison of the performance of the controller is made for all step responses, rise time, settling time, percentage overshoot, steady-state error, and RMSE. From the table, we can see that the proposed PD2-LQR controller is performing better than other controllers. In order to see in better display, the P-LQR controller and the PD2-LQR are compared; the rise time of the PD2-LQR controller is improved by 58.3%, 79.3% improvement in settling time was achieved by the PD2-LQR controller, percentage overshoot of the controller has improved by 88.3%, the minimization of the steady-state error is up to 97.0%, and the RMSE reduce by 17.4%. The result also shows that the percentage overshoot of the PD, PID, and PD-LQR controller are better than that PD2-LQR controller. However, the overshoot value of the PD2-LQR controller is still within the acceptable limit constraint. Thus, the performance of the quadcopter control system in the pitch motion can be improved using the PD2-LQR controller.

5.2. Rolling motion

The PID, PD, LQR, P-LQR, PD-LQR, and PD2-LQR controllers' response in the roll motion is shown in Figure 11. From the figure, the LQR controller has the slowest response in terms of the rise time and settling time, and highest steady-state error value while the PD2-LQR controller is the fastest. However, the PID controller has an excellent performance in overshoot, and the PD controller is best in RMSE compared to other controllers. Even though the PID and PD controller outperform the PD2-LQR controller in terms of overshoot and RMSE respectively, it is worth to mention that the overshoot of the PD2-LQR controller is still within the acceptable limit constraint less than 10% in overshoot, and the difference in RMSE is very small.

Table 9 presents the rise time, settling time, percentage overshoot, the steady-state error, and RMSE of all the controllers in roll motion. The PD2-LQR controller has the lowest rise time with only 0.08388 s, while the LQR controller is the highest with 0.54730 s. For the PD, PID, P-LQR, and PD-LQR controller, the rise time is 0.24698, 0.28322, 0.30698, and 0.26790 s, respectively. So, the percentage improvement of the PD2-LQR controller is as high as 84.7% in the rise time compared to the LQR controller. In terms of settling time, the LQR controller also is the highest with 0.90246 s, followed by the P-LQR controller with only 0.0679 s indifference which is 0.83456 s. The settling time of the PID controller is 0.51770 s, which is higher than the PD controller by 0.07535 s. The settling time of the PD-LQR controller and PD2-LQR controller are 0.48197 and 0.13786 s, respectively, which make the PD2-LQR controller has a better performance in term of settling time than other controllers.

In terms of overshoot, the PID controller has the lowest percentage overshoot as compared to other controllers with 0% overshoot. In comparison, the P-LQR controller has the highest percentage overshoot with 4.02312% then followed by the LQR, PD2-LQR, PD, and PD-LQR controller with 0.05790%, 0.04306%, 0.01195%, and 0.00045%, respectively. However, the percentage overshoot of the proposed PD2-LQR is still within a 10% constraint limit that can be acceptable. The PD2-LQR controller has the lowest steady-state error as compared to other controllers, while the LQR controller has the highest steady-state error. As for the proposed PD2-LR controller, the value is 1.21465e−6. The RMSE of the PD, PID, LQR, P-LQR, PD-LQR, and PD2-LQR controller are 0.26446, 0.33522, 0.37263, 0.34307, 0.31744, and 0.28503, respectively, which make the PD controller has the lowest RMSE. However, the difference between the PD and PD2-LQR controller is minimal, which is 0.02057; thus, it still can be accepted.

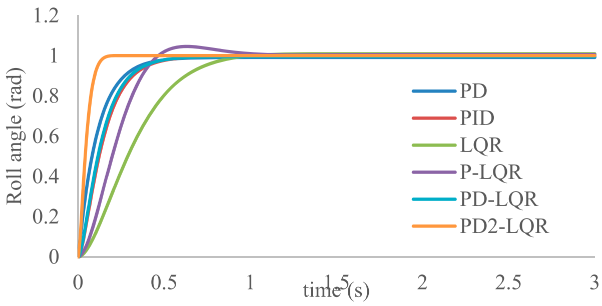

Figure 11. Response of various controllers in roll motion.

Table 9. Performance comparison various controllers in roll motion.

Step info Controller	Rise time, s	Settling time, s	Overshoot, %	Steady-state error	RMSE
PD	0.24698	0.44235	0.01195	0.00872	**0.26446**
PID	0.28322	0.51770	**0**	0.00027	0.33522
LQR	0.54730	0.90246	0.05790	0.00941	0.37263
P-LQR	0.30698	0.83456	4.02312	0.00557	0.34307
PD-LQR	0.26790	0.48197	0.00045	0.00349	0.31744
PD2-LQR	**0.08388**	**0.13796**	0.04306	**1.21465e−06**	0.28503

Overall, the proposed PD2-LQR controller is performing better than other controllers. In comparison between the PD-LQR and the PD2-LQR controller; the rise time of the PD2-LQR controller is improved by 68.7%, 71.4% improvement in settling time was achieved by the PD2-LQR controller, percentage overshoot of the controller has improved by 98.9%, the difference of the steady-state error between the PD-LQR controller and PD2-LQR controller is minimal thus the value can be negligible, and the RMSE reduce by 10.2%. The result also shows that the percentage overshoot of the PID, PD, and PD-LQR controller is better than that PD2-LQR controller. However, the overshoot value of the PD2-LQR controller is still within the acceptable limit constraint. Thus, the performance of the quadcopter control system in the roll motion can be improved using the PD2-LQR controller.

5.3. Yawing motion

The apparent response comparison of the PID, PD, LQR, P-LQR, PD-LQR, and PD2-LQR controller in yaw motion is shown in Figure 12. From the figure, we can see that the PD-LQR controller have the slowest response in terms of the rise time and settling time, while the PD2-LQR controller is the fastest. However, the PD2-LQR controller has slightly increased in steady-state error compared to PD controllers, but for the other controller, PD2-LQR still performs better. Though PD controller outperforms the PD2-LQR controller in terms of steady-state error, it is worth to

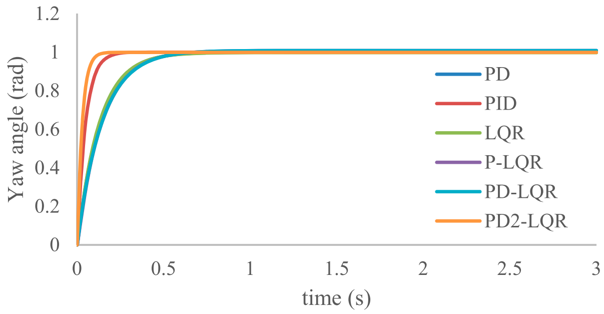

Figure 12. Response of various controllers in yaw motion.

Table 10. Performance comparison various controllers in yaw motion.

Step info Controller	Rise time, s	Settling time, s	Overshoot, %	Steady-state error	RMSE
PD	0.28114	0.50296	0.00046	**6.33704e−06**	0.21469
PID	0.10479	0.18866	0.00838	0.00047	0.21418
LQR	0.27685	0.49542	0.00199	0.00176	0.19483
P-LQR	0.31195	0.55778	0.00151	0.00912	0.20034
PD-LQR	0.31452	0.55901	0.02265	0.00875	0.21993
PD2-LQR	**0.06103**	**0.11072**	**0**	1.82156e−05	**0.14441**

mention that the steady-state error of the PD2-LQR controller is still within the acceptable limit constraint, plus the value differences are minimal that can be approximated to zero.

Table 10 presents the rise time, settling time, percentage overshoot, steady-state error, and RMSE of all the controllers in yaw motion. The PD2-LQR controller has the fastest rise time response with only 0.06103 s, while the PD-LQR controller has the highest response with 0.31452 s. For the PD, PID, LQR, and P-LQR controller, the rise time is 0.28114, 0.10479, 0.27685, and 0.31195 s, respectively. From these values, the percentage improvement of the PD2-LQR controller is as big as 80.6% in the rise time compared to the PD- LQR controller.

In terms of settling time, the PD-LQR controller also is the highest with 0.55901 s, and the P-LQR controller is the second highest with only 0.00123 s indifference, which is 0.55778 s. The settling time of the PD controller is 0.50296 s, which is higher than the PID controller by 0.3143 s. The settling time of the LQR controller and PD2-LQR controller are 0.49542 and 0.11072 s, respectively, which make the PD2-LQR controller has a better performance in term of settling time than other controllers.

In terms of overshoot, the PD2-LQR controller has the lowest percentage overshoot as compared to other controllers with 0% overshoot. In comparison, the PD-LQR controller has the highest percentage overshoot with 0.02265% then followed by the PID, LQR, P-LQR, and PD controller with 0.00838%, 0.00199%, 0.00151%, and 0.00046%, respectively. The value of steady-state error of the PD, PID, LQR, P-LQR, PD-LQR, and PD2-LQR controller are 6.33704e−6, 0.00047, 0.00176, 0.00912, 0.00875, and 1.82156e−5, respectively.

From these values, the PD controller has the lowest steady-state error as compared to other controllers, while the P-LQR controller has the highest steady-state error. As for the proposed PD2-LR controller, the value can be approximated to zero since the differences in the value are very small. The RMSE of the PD2-LQR controller is the lowest with 0.14441 followed by the LQR, P-LQR, PID, PD, and PD-LQR controller, respectively.

Generally, the proposed PD2-LQR controller is performing better than other controllers. In comparison between PD-LQR controller and the PD2-LQR, the rise time of the PD2-LQR controller is improved by 80.6%, 80.2% improvement in settling time was achieved by the PD2-LQR controller, percentage overshoot of the controller has improved by 100%, the difference of the steady-state error between the PD-LQR controller and PD2-LQR controller is minimal thus the value can be negligible, and the RMSE reduce by 34.3%. Thus, the performance of the quadcopter control system in the yaw motion can be improved using the PD2-LQR controller.

5.4. *Altitude motion*

The apparent response comparison of the PID, PD, LQR, P-LQR, PD-LQR, and PD2-LQR controller in altitude motion are shown in Figure 13. From the figure, we can see that the PD and LQR controller has the slowest response in terms of the rise time and settling time respectively, while the PD2-LQR controller is the fastest in rising time and settling time. However, the PD2-LQR controller also has slightly increased in overshoot and steady-state error compared to the PID and PD-LQR controller with 0% overshoot. Though the PID and PD-LQR controller outperforms the PD2-LQR controller in terms of overshoot, the PD2-LQR controller is still within the limit constraint of 10%. Other than that, PID also has the lowest value of steady-state error with only 8.08649e-11, while the PD2-LQR with 0.00034. As for the RMSE, the PD2-LQR controller outperformed the other controllers.

The rise time, settling time, percentage overshoot, steady-state error, and RMSE of all the controllers in altitude motion are presented in Table 11. Here, the

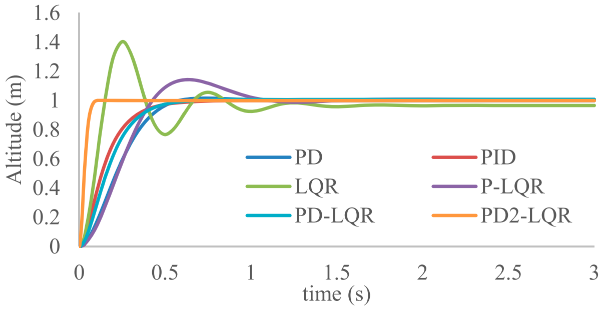

Figure 13. Response of various controllers in altitude motion.

Table 11. Performance comparison various controllers in altitude motion.

Step info Controller	Rise time, s	Settling time, s	Overshoot, %	Steady-state error	RMSE
PD	0.34965	0.52671	1.59856	6.46821e−08	0.41045
PID	0.29385	0.53652	**0**	**8.08649e−11**	0.42895
LQR	0.09783	1.08887	45.34163	0.03368	0.31462
P-LQR	0.29485	0.98465	13.49222	0.00752	0.34269
PD-LQR	0.31883	0.56058	**0**	0.00766	0.35388
PD2-LQR	**0.04941**	**0.08132**	0.18594	0.00034	**0.25820**

PD2-LQR controller has the fastest rise time response with only 0.04941 s, while the PD controller has the slowest response with 0.34965 s. For the PID, LQR, P-LQR, and PD-LQR controllers, the rise time is 0.29385, 0.09783, 0.29485, and 0.31883 s, respectively.

In terms of settling time, the LQR controller has the highest settling time with 1.08887 s, and the P-LQR controller is the second highest with only 0.10422 s indifference which is 0.98465 s. The settling time of the PID controller is 0.53652 s, which is higher than the PD controller by 0.00981 s. The settling time of the PD-LQR controller and PD2-LQR controller are 0.56058 and 0.08132 s, respectively, which make the PD2-LQR controller has a better performance in term of settling time than other controllers.

In terms of overshoot, the PID and PD-LQR controller has the lowest percentage overshoot as compared to other controllers with both 0% overshoot. In comparison, the LQR controller has the highest percentage overshoot with 45.34163% then followed by the P-LQR, PD, and PD2-LQR controllers with 13.49222%, 1.59856%, and 0.18594%, respectively. The steady-state error of the PD, PID, LQR, P-LQR, PD-LQR, and PD2-LQR controller are 6.46821e−8, 8.08649e−11, 0.03368, 0.00752, 0.00766, and 0.00034, respectively.

From these values, the PID controller has the lowest steady-state error as compared to other controllers, while the LQR controller has the highest steady-state error. As for the proposed PD2-LR controller, the value can be approximate to zero since the value is minimal. The PD2-LQR controller has the lowest RMSE with 0.25820 and followed by the LQR (0.31462), P-LQR (0.34269), PD-LQR (0.35388), PD (0.41045), and PID (0.42895) controller, respectively.

Collectively, the proposed PD2-LQR controller is performing better than other controllers. When the PD-LQR controller and the PD2-LQR are compared, the rise time of the PD2-LQR controller is improved by 84.5%, 85.5% improvement in settling time was achieved by the PD2-LQR controller. The difference in percentage overshoot of the controller is minimal and still within the limit; thus, it still can be acceptable. The steady-state error was improved by 95.6% and the RMSE reduce by 27%. Therefore, the performance of the quadcopter control system in attitude motion can be improved using the PD2-LQR controller.

5.5. Summary of findings

In the previous subsection, the simulation result of the PD, PID, LQR, P-LQR, PD-LQR, and PD2-LQR controller in all four motions, pitch, roll, yaw, and altitude have been presented and discussed. From all the controller design approaches, a comparative study has been done to elucidate a collective performance. The comparisons of the entire controller in all motion and percentage improvement of PD2-LQR controller against other controllers are presented in Table 12. In summary, the result shows that the PD2-LQR controller has a better performance in all motions in most of the step responses than other controllers.

From Table 12, we can see that the PD2-LQR controller can perform better than the other controllers in terms of rising time in all motion. In an average of 69.35% improvement can be made if the proposed PD2-LQR controller is used to stabilize the quadcopter.

In terms of settling time, the performance of the PD2-LQR controller is excellent since it can improve all the responses in all motions. The proposed PD2-LQR controller can improve the settling time up to 76% on average. For the overshoot response, the PD2-LQR controller gives a better improvement than most of the other controllers. In an average of 15.7% improvement in overshoot can be achieved using the PD2-LQR controller. However, in pitch motion, the PD, PID, and PD-LQR controllers outperformed the PD2-LQR controller by 99.9%, 100%, and 99.1% improvement, respectively. In contrast, in roll motion, the PD, PID, and PD-LQR controllers give better improvement with 72.4%, 100%, and 98.9%, respectively than PD2-LQR. Though the PD, PID, and PD-LQR controllers are outperformed the PD2-LQR controller in pitch and roll motion, the overshoot of the PD2-LQR controller is still within the acceptable limit constraint that is less than 10% in overshoot.

In pitch motion, the PD2-LQR controller gives an outstanding result in steady-state error compared to PD, PID, LQR, P-LQR, and PD-LQR with 99.73%, 51.7%, 99.3%, 97.0%, and 69.1% improvement, respectively. In roll motion, the steady-state error was improved by 99.8% on average using the PD2-LQR controller. In yaw motion, only the PD controller outperformed the PD2-LQR controller with 65.2% improvement in steady-state error but compared to the PID, LQR, P-LQR, and PD-LQR controllers it still lags by 96.1%, 98.9%, 99.8%, and 99.8%, respectively. In altitude motion, only PD and PID controller outperforms the PD2-LQR controller with both 99.9% improvement in steady-state error. However, the value of the steady-state error in all motion is minimal between the proposed PD2-LQR controllers with the other controllers. Therefore, the value of the steady-state error for the entire controllers can be approximated to be zero.

Table 12. Performance comparison in Percentage among the various in all four motions

Motion	Parameters	PD vs. PD2-LQR	PID vs. PD2-LQR	LQR vs. PD2-LQR	*P*-LQR vs. PD2-LQR	PD-LQR vs PD2-LQR
Pitch	Rise time, s	🟩 47.8	🟩 57.2	🟩 58.4	🟩 58.3	🟩 57.4
	Settling time, s	🟩 55.2	🟩 63.8	🟩 79.3	🟩 79.3	🟩 63.7
	Overshoot, %	🟥 99.9	🟥 100	🟩 88.2	🟩 88.3	🟥 99.1
	Steady-state error	🟩 99.73	🟩 51.7	🟩 99.3	🟩 97.0	🟩 69.1
	RMSE	🟩 12.1	🟩 18.3	🟩 17.5	🟩 17.4	🟩 6.5
Roll	Rise time, s	🟩 66.0	🟩 70.4	🟩 84.7	🟩 72.7	🟩 68.7
	Settling time, s	🟩 68.8	🟩 73.3	🟩 84.7	🟩 83.5	🟩 71.4
	Overshoot, %	🟥 72.4	🟥 100	🟩 25.7	🟩 98.9	🟥 98.9
	Steady-state error	🟩 99.9	🟩 99.5	🟩 99.9	🟩 99.9	🟩 99.9
	RMSE	🟥 7.2	🟩 15.0	🟩 23.5	🟩 16.9	🟩 10.2
Yaw	Rise time, s	🟩 78.3	🟩 41.8	🟩 78.0	🟩 80.4	🟩 80.6
	Settling time, s	🟩 78.0	🟩 41.3	🟩 77.7	🟩 80.2	🟩 80.2
	Overshoot, %	🟩 100	🟩 100	🟩 100	🟩 100	🟩 100
	Steady-state error	🟥 65.2	🟩 96.1	🟩 98.9	🟩 99.8	🟩 99.8
	RMSE	🟩 32.7	🟩 32.6	🟩 25.9	🟩 27.9	🟩 34.3
Altitude	Rise time, s	🟩 85.9	🟩 83.2	🟩 49.5	🟩 83.2	🟩 84.5
	Settling time, s	🟩 84.6	🟩 84.8	🟩 92.5	🟩 91.7	🟩 85.5
	Overshoot, %	🟩 88.4	🟥 100	🟩 99.6	🟩 94.2	🟥 100
	Steady-state error	🟩 99.9	🟩 99.9	🟩 99.0	🟩 95.5	🟩 95.6
	RMSE	🟩 37.1	🟩 39.8	🟩 17.9	🟩 24.7	🟩 27.0

∗ The cell in red means the performance of the controller is not improved and the ones in green means improved.

Lastly, in terms of RMSE, only in roll motion where the PD controller outperforms the PD2-LQR controller with only 7.2% improvement. For the others, PD2-LQR gives an excellent performance than the other controllers. In average, a 69.5% improvement can be made if the proposed PD2-LQR controller is used in all motions.

6. Conclusion

In this paper, modelling and control of the quadcopter in all four motions, pitch, roll, yaw, and altitude have been done. The dynamic model of the quadcopter used in this research was derived based on the Newton–Euler approach. Six controllers, including the proposed controller, namely PD, PID, LQR, P-LQR, PD-LQR, and PD2-LQR controllers, have been designed to access how it stabilizes the quadcopter. Their performance in terms of rising time, settling time, percentage overshoot, steady-state error, and RMSE has been compared to find the best-suited controller for the quadcopter. The simulation results show that the proposed PD2-LQR controller was able to improve the performance of the quadcopter in almost all step responses than other controllers. Though some controllers can outperform the PD2-LQR controller in certain responses, it is worth to mention that the differences between those values are minimal and some of them can be approximately equal to zero. Plus, all responses of the PD2-LQR controller is within the acceptable limit constraint. Therefore, based on this result, the proposed PD2-LQR controller is suitable to be used to controlling the quadcopter in all pitch, roll, yaw, and altitude motion.

Acknowledgements

The authors thank the editor and anonymous reviewers for their helpful comments and valuable suggestions.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

The authors would like to acknowledge the funding support provided by RU Grant Universiti Sains Malaysia 1001/PAERO/8014108 in carrying out this work.

Data availability statement

The authors confirm that the data supporting the findings of this study are available within the article.

ORCID

Parvathy Rajendran http://orcid.org/0000-0002-7430-1389
Nurulasikin Mohd Suhadis http://orcid.org/0000-0002-2695-8167

References

[1] Badr S, Mehrez O, Kabeel AE. A design modification for a quadrotor UAV: modeling, control and implementation. Adv Robot. 2019;33(1):13–32.

[2] Wei Y, Blake MB, Madey GR. An operation-time simulation framework for UAV swarm configuration and mission planning. Procedia Comput Sci. 2013;18:1949–1958.

[3] Khatoon S, Nasiruddin I, Shahid M. Design and simulation of a hybrid PD-ANFIS controller for attitude tracking control of a quadrotor UAV. Arab J Sci Eng. 2017;42(12):5211–5229.

[4] Han J, Di L, Coopmans C, et al. Pitch loop control of a VTOL UAV using fractional order controller. J Intell Robot Syst. 2014;73:187–195.

[5] Hasseni S-E-I, Abdou L, Glida H-E. Parameters tuning of a quadrotor PID controllers by using nature-inspired algorithms. Evol Intell. 2021;14:61–73.

[6] Xiu C, Liu F, Xu G. General model and improved global sliding mode control of the four-rotor aircraft. Proc Inst Mech Eng I J Syst Control Eng. 2018;232(4):383–389.

[7] Kuantama E, Vesselenyi T, Dzitac S, et al. PID and fuzzy-PID control model for quadcopter attitude with disturbance parameter. Int J Comput Commun Control. 2017;12(4):519–532.

[8] Erkol HO. Attitude controller optimization of four-rotor unmanned air vehicle. Int J Micro Air Veh. 2017;10(1):42–49.

[9] Burggräf P, Perez Martinez AR, Roth H, et al. Quadrotors in factory applications: design and implementation of the quadrotor's P-PID cascade control system. SN Appl Sci. 2019;1(7):722.

[10] Abdelhay S, Zakriti A. Modeling of a quadcopter trajectory tracking system using PID controller. Procedia Manuf. 2019;32:564–571.

[11] Okyere E, Bousbaine A, Poyi GT, et al. LQR controller design for quad-rotor helicopters. J Eng. 2019;2019(17): 4003–4007.

[12] Martins L, Cardeira C, Oliveira P. Linear quadratic regulator for trajectory tracking of a quadrotor. IFAC-PapersOnLine. 2019;52(12):176–181.

[13] Shah J, Okasha M, Faris W. Gain scheduled integral linear quadratic control for quadcopter. Int J Eng Technol. 2018;7(4.13):81–85.

[14] Jia Z, Yu J, Mei Y, et al. Integral backstepping sliding mode control for quadrotor helicopter under external uncertain disturbances. Aerosp Sci Technol. 2017;68:299–307.

[15] Basri A, Husain A, Danapalasingam KA. Nonlinear control of an autonomous quadrotor unmanned aerial vehicle using backstepping controller optimized by particle swarm optimization. J Eng Sci Technol Rev. 2015;8:39–45.

[16] Mofid O, Mobayen S. Adaptive sliding mode control for finite-time stability of quad-rotor UAVs with parametric uncertainties. ISA Trans. 2018;72:1–14.

[17] Castillo-Zamora JJ, Camarillo-Gomez KA, Perez-Soto GI, et al. Comparison of PD, PID and sliding-mode position controllers for V–tail quadcopter stability. IEEE Access. 2018;6:38086–38096.

[18] Xiu C, Liu F, Xu G. General model and improved global sliding mode control of the four-rotor aircraft. Proc Inst Mech Eng I J Syst Control Eng. 2017;232(4):383–389.

[19] Yit KK, Rajendran P, Wee LK. Proportional-derivative linear quadratic regulator controller design for improved longitudinal motion control of unmanned aerial vehicles. Int J Micro Air Veh. 2016;8(1):41–50.

[20] Li J, Li Y.. Dynamic analysis and PID control for a quadrotor. Proceedings of the 2011 IEEE International Conference on Mechatronics and Automation August 7-10, Beijing, China. 2011.

[21] Pan F, Liu L, Xue D. Optimal PID controller design with Kalman filter for Qball-X4 quad-rotor unmanned aerial vehicle. Trans Inst Meas Control. 2016;39(12):1785–1797.

[22] Tanveer M, Ahmed SF, Desa H, et al. Stabilized controller design for attitude and altitude controlling of quad-rotor under disturbance and noisy conditions. Amer J Appl Sci. 2013;10:819–831.

[23] Ahmad F, Kumar P, Patil PP. Modeling and simulation of a quadcopter with altitude and attitude control. Nonlinear Stud. 2018;25:287–299.

[24] Zhi Y, Li G, Song Q, et al. Flight control law of unmanned aerial vehicles based on robust servo linear quadratic regulator and Kalman filtering. Int J Adv Rob Syst. 2017;14(1). DOI:10.1177/1729881416686952.

[25] Hernandez-Martinez EG, Fernandez-Anaya G, Ferreira ED, et al. Trajectory tracking of a quadcopter UAV with optimal translational control. IFAC-PapersOnLine. 2015;48(19):226–231.

[26] Kumar R, Dechering M, Pai A, et al. Differential Flatness Based Hybrid PID/LQR Flight Controller for Complex Trajectory Tracking in Quadcopter UAVs 27-30 June 2017 Dayton, OH, USA2017 IEEE National Aerospace and Electronics Conference (NAECON).

[27] Yit KK, Rajendran P. Enhanced longitudinal motion control of UAV simulation by using P-LQR method. Int J Micro Air Veh. 2015;7(2):203–210.

[28] Walle SM. Dynamic modeling and trajectory tracking control of hexacopter using sliding mode controller; 2019. Master Dissertation

[29] Reizenstein A. Position and trajectory control of a quadcopter using PID and LQ controllers; 2017. Master Dissertation

[30] Siciliano B, Sciavicco L, Villani L, et al. Robotics: modelling, planning and control. Springer-Verlag London, London: Springer Science & Business Media; 2010.

[31] Numan M. Controller design for attitude and position control of quadrotor. Pakistan: Department of Electrical Engineering, Capital University of Science & Technology Islamabad; 2017.

[32] Nagaty A, Saeedi S, Thibault C, et al. Control and navigation Framework for quadrotor helicopters. J Intell Robot Syst. 2013;70(1):1–12.

[33] Sabatino F. Quadrotor control: modeling, nonlinearcontrol design, and simulation; 2015. Master Dissertation

[34] Nise NS. Control systems engineering: international student version. 6th ed. Hoboken: John Wiley & Sons; 2011.

[35] Beebi M. L, Vishakh KH, N N. Control of longitudinal dynamics in the reentry phase of a reusable launch vehicle. Int J Adv Res Electr Electron Instrum Eng. 2012;1(4):438–442.

[36] Vinodh Kumar E, Jerome J. Robust LQR controller design for stabilizing and trajectory tracking of inverted pendulum. Procedia Eng. 2013;64:169–178.

[37] Das S, Pan I, Halder K, et al. LQR based improved discrete PID controller design via optimum selection of weighting matrices using fractional order integral performance index. Appl Math Model. 2013;37(6):4253–4268.

[38] Khoygani MRR, Ghasemi R, Sanaei D.. Design controller for a class of nonlinear pendulum dynamical system. IAES International Journal of Artificial Intelligence. 2013;2:159–168.

[39] Hajiyev C. LQR controller with Kalman estimator Applied to UAV longitudinal dynamics. Positioning. 2013;4:36–41.

[40] Kim GB, Nguyen TK, Budiyono A, et al. Design and development of a class of rotorcraft-based UAV. Int J Adv Rob Syst. 2013;10(2):131.

[41] Kok Kai Y, Rajendran P, Lim Kah W. Proportional-derivative linear quadratic regulator controller design for improved longitudinal motion control of unmanned aerial vehicles. Int J Micro Air Veh. 2016;8(1): 41–50.

[42] Wael MA, Quan Q. Robust hybrid control for ballistic missile longitudinal autopilot. Chin J Aeronaut. 2011;24(6):777–788.

[43] Gong H, Zhen Z, Li X, et al. Design of automatic climbing controller for large civil aircraft. J Franklin Inst. 2013;350(9):2442–2454.

[44] Austin R. Unmanned aircraft systems. AIAA education series. Chichester: American Institute of Aeronautics and Astronautics; Wiley; 2010.

[45] MathWorks. Simulink design optimization: user's guide. 2021; [cited 2021 July 7]. Available from https://www.mathworks.com/help/pdf_doc/sldo/sldo_ug.pdf.