PROGRESS IN AEROSPACE AND AVIATION TECHNOLOGY
e-ISSN: 2821-2924

PAAT

Vol. 4 No. 1 (2024) 17-35
https://publisher.uthm.edu.my/ojs/index.php/paat

Flight Control System Design and Dynamic Simulation for DJI F450 Quadcopter UAV Using Pole Placement Tuning Method

Annis Syahirah Jemsa[1], Syariful Syafiq Shamsudin[1]*, Mohammad Fahmi Pairan[1]

[1] *Research Center for Unmanned Vehicle (RECUV), Faculty of Mechanical and Manufacturing University Tun Hussein Onn Malaysia (UTHM), Batu Pahat, 86400, MALAYSIA*

*Corresponding Author: syafiq@uthm.edu.my
DOI: https://doi.org/10.30880/paat.2024.04.02.003

Article Info

Received: 22 January 2024
Accepted: 1 July 2024
Available online: 24 December 2024

Keywords

PID, quadcopter UAV, pole placement (PP) tuning, software in the loop (SIL)

Abstract

This paper presents the flight control design of a quadcopter-based unmanned aerial vehicle (UAV) using an improved PID tuning method based on pole placement (PP) to achieve the desired time response specification. The flight of a quadcopter UAV presents a difficult control problem because it is a fast-responsive and unstable nonlinear system. This study employed the X-Plane flight simulator to assess the control gain values of the autopilot system. The autopilot system was developed using the PID controller in the LabVIEW environment, utilizing the Software in the Loop (SIL) approach. A series of simulated flight experiments were carried out to verify the performance of the proposed controllers and assess the effectiveness of the control systems. The findings suggest that PID controllers, when tuned using the pole placement (PP) technique, can achieve satisfactory position tracking performance during hovering movements, with x and y positions staying within ±0.5 m limits. The controller demonstrated sufficient resilience in handling wind turbulence of 4.5 m/s during a circular path tracking simulation. It achieved position errors of 0.1306 m, 0.0832 m, and 0.2881 m in the x, y, and z positions, respectively, while maintaining a heading error accuracy of less than 3º as per the intended specification.

1. Introduction

Quadcopter-based aircraft are widely used in unmanned aerial vehicle (UAV) configurations in the aeronautical community. The aircraft are well-suited for tasks such as surveillance, aerial mapping, videography, examining high-rise buildings, and monitoring applications. To achieve automated flight in these applications, it is necessary to create an automatic flight control system (AFCS) specifically designed for quadcopter UAVs. The automatic control function is in charge of adjusting the UAV system's control inputs to produce the intended result on its outputs without requiring human involvement in the control loop. This is achieved by using control laws to compute input commands for the actuators of the UAV. These commands generate torques and forces that act on the vehicle, allowing it to regulate its motion in six degrees of freedom (DOF). The complex, under-actuated, inherently unstable, and nonlinear characteristics of quadcopter-based UAVs pose a significant difficulty for devising and executing a control strategy [1], [2]. The stability of the quadcopter-based UAV relies on low-level control mechanisms, specifically velocity feedback and attitude feedback. In the event that the UAV does not receive stabilizing control commands, even for a short duration, it is highly probable that it will lose stability and have a crash.

One of the most successful strategies for controlling a multi-rotor UAV is to use a model-free control design method, such as proportional-integral-derivative (PID) control. In most quadcopter UAV projects, a cascaded control architecture is used for the control of the vehicle [2], [3], [4]. The cascaded control architecture comprises two subsystems for control loops [2]. The inner loop control governs the orientation of the UAS by utilizing a multiple single-input, single-output (SISO) PID controller for each axis. On the other hand, the outer loop manages the UAV's translational movement. The PID controller provides an appealing design method that does not require identification of the UAS's dynamic model.

PID tuning is the process of determining the value of a PID controller's proportional, integral, and derivative gains to achieve the desired performance and correspond to design specifications. In a quadcopter-based UAS, the PID controller gain is usually adjusted by empirical tuning methods such as trial-and-error, model-based techniques, or rule-based procedures. The trial-and-error method can be applied without considering the UAV dynamic model, allowing for the empirical tuning of all controller parameters during flight. Nevertheless, the iterative adjustment of PID gains can be a lengthy process and may provide challenges, particularly for novices [5], [6], [7]. Improper selection of controller gains during flight could result in a potential danger of failure or crash.

In model based PID tuning methods, mathematical optimization algorithms are used to determine the optimal PID parameter values [8]. These methods require the development of a mathematical model of the process, which can be achieved using any system identification techniques such as those suggested in [9], [10], [11], [12]. The model is then applied to the development of a cost function that reflects the intended performance specifications. The PID parameters that minimize the cost function are searched for by the optimization method. Babu et al. [13] propose a gradient descent-based methodology to tune the PID controller parameters for an AR Drone quadrotor. In this approach, the gradient of the cost function is used to update and improve the PID gain parameters iteratively. The technique is demonstrated through two test cases of way-point navigation and leader-follower formation control. In [14], the authors present the design of a PID controller for position and attitude stabilization of a quadrotor UAV using the differential-evolution (DE) optimization algorithm. A performance index is defined to evaluate the performance of the system, which focuses on stabilizing the attitude, reaching the desired altitude quickly, and minimizing power consumption. Findings show that the performance of the DE-based PID controller is shown to be superior, indicating that the DE optimization algorithm is an effective tool for tuning PID controller gains. The optimized system, with the DE-based PID controller, stabilized its attitude considerably quicker than its counterparts. The altitude response of the optimized system achieved a faster rise time with a marginally better settling time. Overall, the performance of the DE-based PID controller was superior, highlighting the effectiveness of the DE optimization algorithm for tuning PID controller gains. In Noordin et al. [15], proposed a nonlinear dynamics model of a quadrotor using the Newton-Euler modeling technique and used a PID controller with fine-tuned parameters using the particle swarm optimization (PSO) algorithm to stabilize the quadrotor's attitude and achieve hovering. The simulation results show that the quadrotor achieves zero steady state error for hovering, with an overshoot of 18.9% and a settling time of 4.42 s. Additionally, the roll angle, pitch angle, and yaw angle converge to the set point of zero with settling times of 2.76 s, 0.1 s, and 3.2 s, respectively. Nevertheless, model-based PID tuning offers several advantages to the user, such as optimized performance and automatic tuning compared to manual tuning approaches [16], [17], however, at the expense of increased computational requirements and the need for accurate mathematical models of the system, which may not always be available or practical to obtain.

Rule-based methods use empirical formulas based on dynamic responses or characteristics. Simple experiments can be done to derive the process response or characteristics, which are then utilized to calculate the PID parameters. Classical tuning rules such as Ziegler-Nichols tuning typically suffer from several drawbacks, including a lack of ability to define control objectives or closed-loop performance requirements, and a potential degradation in controller performance in terms of oscillatory response [18], [19], [20], [21]. Pole placement (PP) based PID controller design is introduced in [21] to improve the disadvantages of classical tuning rules for PID controllers, where the PID controller parameters can be directly computed based on the dynamic model characteristics and the desired closed loop response. In the PP-based PID controller, the poles of the feedback control system are placed in specific locations in the complex plane to achieve the desired closed-loop performance [19], [25]. The desired pole locations are determined based on the desired performance of the closed-loop system. For example, if fast response and no overshoot are desired, the poles can be placed in the left half of the complex plane with a large negative real part. The PID controller gains are then tuned based on the pole locations. The proportional, integral, and derivative gains are adjusted to ensure that the closed-loop system is stable and achieves the desired performance. The method is conceptually and computationally simple, and the design methodology can be extended to various controller structures and models.

The work presented in this paper will focus on the development of a flight control system for a quadcopter UAV based on Pole Placement (PP) based PID design approaches. The PID controller design used a linear dynamic system model derived from system identification work in [28] to obtain the controller’s parameters. Finally, a set of simulated flight tests with simulated wind disturbances is conducted to validate the performance and effectiveness of the proposed controller using the X-Plane flight simulator. The proposed controller tuning methods are able to produce PID controllers that regulate and stabilize the quadcopter UAV to the desired performance level and show adequate robustness to external disturbances.

The presented work in this paper is organized as follows: In Section 2, the mathematical modeling of unmanned rotorcraft system dynamics is presented. The approach for conducting flight test simulations to evaluate the proposed control algorithms is discussed in Section 3. The proposed controller design based on PP tuning methods, is presented in Section 4. The flight simulation results are presented in Section 3, where the findings and validation analyses are discussed. Finally, Section 4 gives the concluding remarks.

2. Kinematics and Dynamics of a Quadrotor

The quadcopter-based UAV dynamics are known to be a non-linear model of a high order multiple-input multiple-output (MIMO) system. The kinematics and dynamics of the quadcopter UAV can be described using Newton-Euler equations of motion and expressed by twelve dimensional states. These state are consists of quadcopter UAV position vectors in inertial reference frame, x, y, and z; the Euler angles (roll angle ϕ, pitch angle ϕ, and yaw angles ψ) in the body reference frame; the quadcopter linear velocity in the body reference frame (u, v, and w), and quadcopter angular velocities in the body reference frame, (p, q, and r). The motion of the quadcopter expressed in the Body Reference Frame is defined in Fig. 1. The forces F_* and torques τ_* produced by each rotor are proportional to the pulsewidth modulation signals send to the motor. Therefore, the force and torque produced by each motor can be express as follows:

$$F_* = k_1 \delta_* \qquad \tau_* = k_2 \delta_* \tag{1}$$

where k_1 and k_2 are the force and torque constants that need to be determined experimentally, and δ_* is the motor command signal with subscript * represent individual motor marking (f, b, r, and l). Finally, the total forces acting on the quadcopter F, the rolling torque produced by the forces of the right and motors τ_ϕ, the pitching torque produced by the front and back motors τ_θ and yawing torque acting on the body τ_ψ can be expressed in matrix form as follows:

$$\begin{bmatrix} F \\ \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \begin{bmatrix} k_1 & k_1 & k_1 & k_1 \\ 0 & -lk_1 & 0 & lk_1 \\ lk_1 & 0 & -lk_1 & 0 \\ -k_2 & k_2 & -k_2 & k_2 \end{bmatrix} \begin{bmatrix} \delta_f \\ \delta_r \\ \delta_b \\ \delta_l \end{bmatrix} \tag{2}$$

The overall dynamic of the quadcopter UAV can be divided into translational and rotational kinematics (Equations (3)-(4)), and translational and rotational dynamics (Equations (5)-(6)) as follows:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ \sin\theta & -\sin\phi\cos\theta & -\cos\phi\cos\theta \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{3}$$

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & 0 & tan\,\theta\, cos\,\phi \\ 0 & cos\,\phi & -sin\,\phi \\ 0 & \dfrac{sin\,\phi}{cos\,\theta} & \dfrac{cos\,\phi}{cos\,\theta} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{4}$$

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \begin{bmatrix} rv - qw \\ pw - rq \\ qu - pv \end{bmatrix} + \begin{bmatrix} -g\sin\theta \\ g\cos\theta\sin\phi \\ g\cos\theta\cos\phi \end{bmatrix} + \frac{1}{m}\begin{bmatrix} 0 \\ 0 \\ -F \end{bmatrix} \tag{5}$$

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \dfrac{I_{yy} - I_{zz}}{I_{xx}} qr \\ \dfrac{I_{zz} - I_{xx}}{I_{yy}} pr \\ \dfrac{I_{xx} - I_{yy}}{I_{zz}} pq \end{bmatrix} + \begin{bmatrix} \dfrac{1}{I_{xx}} \tau_\phi \\ \dfrac{1}{I_{yy}} \tau_\theta \\ \dfrac{1}{I_{zz}} \tau_\psi \end{bmatrix} \tag{6}$$

where I_{xx}, I_{yy}, and I_{zz} are the moment of inertia about body axes. For the simulation of quadcopter flight, variables F, τ_ϕ, τ_θ, and τ_ψ will be used as input to the dynamic system model.

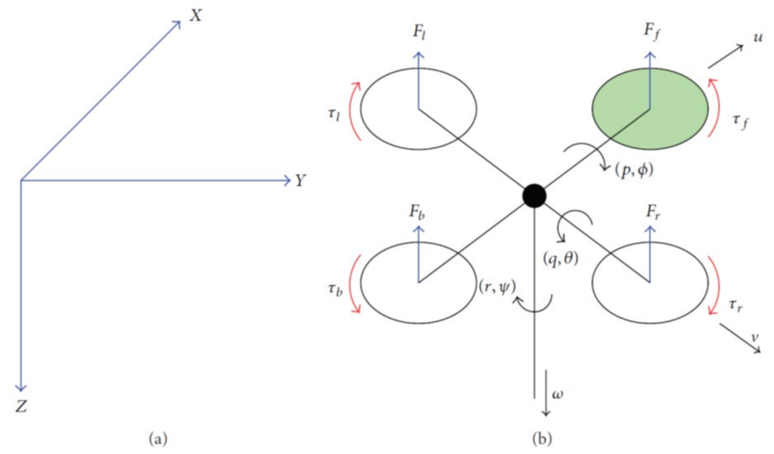


Fig. 1 *The free body diagram of a quadcopter UAV system* [3]. *(a) Inertial frame (b) The forces and moments acting on the quadcopter body*

2.1 Linearization

To design a linear controller for the quadcopter, it is important that the nonlinear differential equation presented in Equation (3)-(6) to be simplified into linear dynamic model. If the roll angle ϕ, pitch angle θ and yaw angle ψ are assumed to be small, the translational and rotational kinematic equations in Equation (3) and (4) can be simplified as:

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{z}\end{bmatrix} = \begin{bmatrix}u\\ v\\ w\end{bmatrix} \tag{7}$$

$$\begin{bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{bmatrix} = \begin{bmatrix}p\\ q\\ r\end{bmatrix} \tag{8}$$

Similarly, the translational and rotational dynamics in Equation (5) and (6) can be further simplified if the roll angle, pitch angle and the Coriolis terms qr, pr, and pq are assumed to be small to yield:

$$\begin{bmatrix}\dot{u}\\ \dot{v}\\ \dot{w}\end{bmatrix} = \begin{bmatrix}-g\theta\\ g\phi\\ g-\dfrac{F}{m}\end{bmatrix} \tag{9}$$

$$\begin{bmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix} = \begin{bmatrix}\dfrac{1}{I_{xx}}\tau_\phi\\ \dfrac{1}{I_{yy}}\tau_\theta\\ \dfrac{1}{I_{zz}}\tau_\psi\end{bmatrix} \tag{10}$$

3. Software in The Loop (SITL) Simulation

The proposed control algorithm will be evaluated in the SITL simulation, where the flight performance of the quadcopter-based UAV is analyzed. By directly connecting the flight control software to a digital plant model, engineers can utilize SITL simulation to test and modify their source code instantly and repeatedly without the use of more expensive equipment, prototypes, or test benches [22]. The test platform consists of two primary elements: the LabVIEW program, responsible for executing the control system algorithm to direct the flight control surfaces of the quadcopter, and the X-Plane flight simulator, which accurately replicates the behavior of the quadcopter-based UAV, as depicted in Fig. 2. The selection of the X-Plane flight simulator was based on its high level of accuracy in predicting the response of the UAV aircraft, as indicated by reference [23].

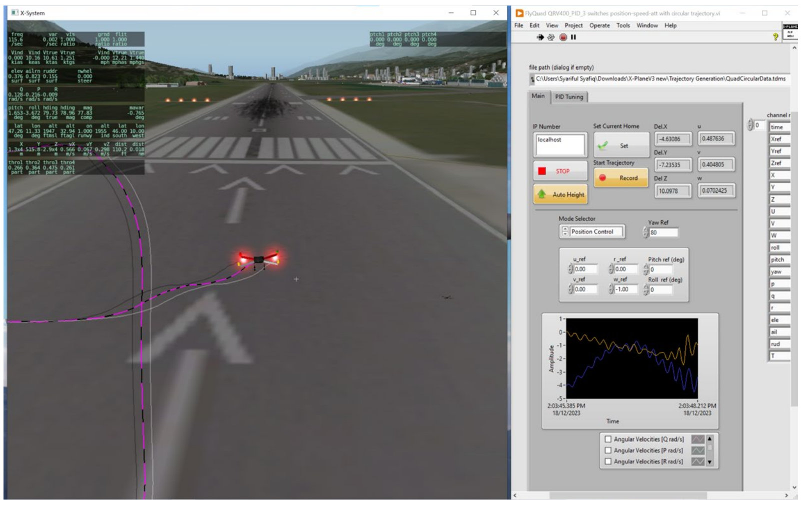
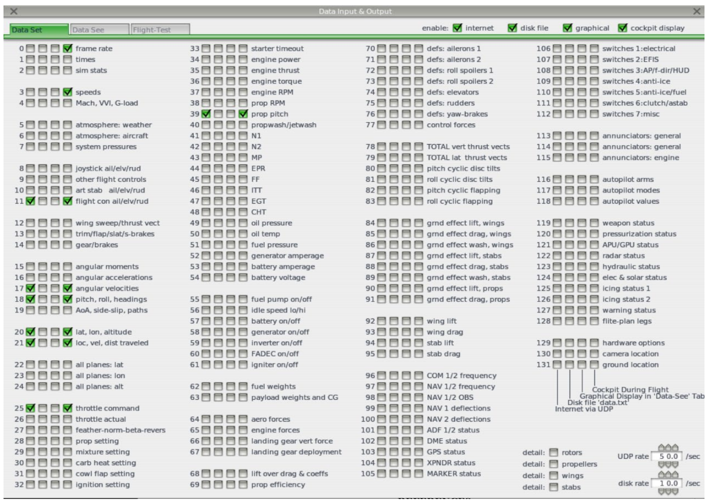


Fig. 2 *X-Plane simulation with DJI F450 quadcopter model and LabVIEW front panel*

X-Plane can be interfaced easily by transmitting the data from numerous flight data packets to the host over the Universal Data Packet (UDP) communication protocol [24]. The main advantage of the UDP communication protocol is its real-time communication ability with minimal delays. Users can create a LabVIEW program to read the received UDP data and send input commands to the X-Plane software [25]. Details about the UDP packet format can be found in [26], [27]. The IP address "127.0.0.1", which is the network card's internal address, was used in this study to create communication between X-Plane and LabVIEW on a single computer. The estimated parameters of the autopilot control system are sent to X-Plane for control of the rotorcraft's flight control surfaces. X-Plane calculates the new aircraft attitude based on the inputs from LabVIEW and provides the modified rotorcraft attitude parameters to LabVIEW, bringing the loop to a close. LabVIEW restarts the operation and continues the process flow by delivering fresh commands to the X-Plane.

Fig. 3 shows the selected input and flight sensor data that will be displayed in the cockpit and produced for the UDP data exchange. For the flight control simulation, flight data such as angular velocities, attitude angles, position, and linear velocities are used as feedback data for controller design. The calculated controller commands, $\boldsymbol{\delta} = [T \quad \tau_\theta \quad \tau_\phi \quad \tau_\psi]^T$ from PID controllers, will be sent back to X-Plane via four motor/throttle command signals. Note that the motor/throttle command signals are required to be in the range of 0 to +1. The present study considers the UDP rate at 50 Hz, with the data recording sampling time set at 10 Hz.


Data Input & Output

Data Set | Data See | Flight-Test

enable: ✓ internet ✓ disk file ✓ graphical ✓ cockpit display

	Internet via UDP	Disk file 'data.txt'	Graphical Display in 'Data-See' Tab	Cockpit During Flight	
0	☐	☐	☐	✓	frame rate
1	☐	☐	☐	☐	times
2	☐	☐	☐	☐	sim stats
3	☐	☐	☐	✓	speeds
4	☐	☐	☐	☐	Mach, VVI, G-load
5	☐	☐	☐	☐	atmosphere: weather
6	☐	☐	☐	☐	atmosphere: aircraft
7	☐	☐	☐	☐	system pressures
8	☐	☐	☐	☐	joystick ail/elv/rud
9	☐	☐	☐	☐	other flight controls
10	☐	☐	☐	☐	art stab ail/elv/rud
11	✓	☐	☐	✓	flight con ail/elv/rud
12	☐	☐	☐	☐	wing sweep/thrust vect
13	☐	☐	☐	☐	trim/flap/slat/s-brakes
14	☐	☐	☐	☐	gear/brakes
15	☐	☐	☐	☐	angular moments
16	☐	☐	☐	☐	angular accelerations
17	✓	☐	☐	✓	angular velocities
18	✓	☐	☐	✓	pitch, roll, headings
19	☐	☐	☐	☐	AoA, side-slip, paths
20	✓	☐	☐	✓	lat, lon, altitude
21	✓	☐	☐	✓	loc, vel, dist traveled
22	☐	☐	☐	☐	all planes: lat
23	☐	☐	☐	☐	all planes: lon
24	☐	☐	☐	☐	all planes: alt
25	✓	☐	☐	✓	throttle command
26	☐	☐	☐	☐	throttle actual
27	☐	☐	☐	☐	feather-norm-beta-revers
28	☐	☐	☐	☐	prop setting
29	☐	☐	☐	☐	mixture setting
30	☐	☐	☐	☐	carb heat setting
31	☐	☐	☐	☐	cowl flap setting
32	☐	☐	☐	☐	ignition setting
33	☐	☐	☐	☐	starter timeout
34	☐	☐	☐	☐	engine power
35	☐	☐	☐	☐	engine thrust
36	☐	☐	☐	☐	engine torque
37	☐	☐	☐	☐	engine RPM
38	☐	☐	☐	☐	prop RPM
39	✓	☐	☐	✓	prop pitch
40	☐	☐	☐	☐	propwash/jetwash
41	☐	☐	☐	☐	N1
42	☐	☐	☐	☐	N2
43	☐	☐	☐	☐	MP
44	☐	☐	☐	☐	EPR
45	☐	☐	☐	☐	FF
46	☐	☐	☐	☐	ITT
47	☐	☐	☐	☐	EGT
48	☐	☐	☐	☐	CHT
49	☐	☐	☐	☐	oil pressure
50	☐	☐	☐	☐	oil temp
51	☐	☐	☐	☐	fuel pressure
52	☐	☐	☐	☐	generator amperage
53	☐	☐	☐	☐	battery amperage
54	☐	☐	☐	☐	battery voltage
55	☐	☐	☐	☐	fuel pump on/off
56	☐	☐	☐	☐	idle speed lo/hi
57	☐	☐	☐	☐	battery on/off
58	☐	☐	☐	☐	generator on/off
59	☐	☐	☐	☐	inverter on/off
60	☐	☐	☐	☐	FADEC on/off
61	☐	☐	☐	☐	igniter on/off
62	☐	☐	☐	☐	fuel weights
63	☐	☐	☐	☐	payload weights and CG
64	☐	☐	☐	☐	aero forces
65	☐	☐	☐	☐	engine forces
66	☐	☐	☐	☐	landing gear vert force
67	☐	☐	☐	☐	landing gear deployment
68	☐	☐	☐	☐	lift over drag & coeffs
69	☐	☐	☐	☐	prop efficiency
70	☐	☐	☐	☐	defs: ailerons 1
71	☐	☐	☐	☐	defs: ailerons 2
72	☐	☐	☐	☐	defs: roll spoilers 1
73	☐	☐	☐	☐	defs: roll spoilers 2
74	☐	☐	☐	☐	defs: elevators
75	☐	☐	☐	☐	defs: rudders
76	☐	☐	☐	☐	defs: yaw-brakes
77	☐	☐	☐	☐	control forces
78	☐	☐	☐	☐	TOTAL vert thrust vects
79	☐	☐	☐	☐	TOTAL lat thrust vects
80	☐	☐	☐	☐	pitch cyclic disc tilts
81	☐	☐	☐	☐	roll cyclic disc tilts
82	☐	☐	☐	☐	pitch cyclic flapping
83	☐	☐	☐	☐	roll cyclic flapping
84	☐	☐	☐	☐	grnd effect lift, wings
85	☐	☐	☐	☐	grnd effect drag, wings
86	☐	☐	☐	☐	grnd effect wash, wings
87	☐	☐	☐	☐	grnd effect lift, stabs
88	☐	☐	☐	☐	grnd effect drag, stabs
89	☐	☐	☐	☐	grnd effect wash, stabs
90	☐	☐	☐	☐	grnd effect lift, props
91	☐	☐	☐	☐	grnd effect drag, props
92	☐	☐	☐	☐	wing lift
93	☐	☐	☐	☐	wing drag
94	☐	☐	☐	☐	stab lift
95	☐	☐	☐	☐	stab drag
96	☐	☐	☐	☐	COM 1/2 frequency
97	☐	☐	☐	☐	NAV 1/2 frequency
98	☐	☐	☐	☐	NAV 1/2 OBS
99	☐	☐	☐	☐	NAV 1 deflections
100	☐	☐	☐	☐	NAV 2 deflections
101	☐	☐	☐	☐	ADF 1/2 status
102	☐	☐	☐	☐	DME status
103	☐	☐	☐	☐	GPS status
104	☐	☐	☐	☐	XPNDR status
105	☐	☐	☐	☐	MARKER status
106	☐	☐	☐	☐	switches 1:electrical
107	☐	☐	☐	☐	switches 2:EFIS
108	☐	☐	☐	☐	switches 3:AP/f-dir/HUD
109	☐	☐	☐	☐	switches 4:anti-ice
110	☐	☐	☐	☐	switches 5:anti-ice/fuel
111	☐	☐	☐	☐	switches 6:clutch/astab
112	☐	☐	☐	☐	switches 7:misc
113	☐	☐	☐	☐	annunciators: general
114	☐	☐	☐	☐	annunciators: general
115	☐	☐	☐	☐	annunciators: engine
116	☐	☐	☐	☐	autopilot arms
117	☐	☐	☐	☐	autopilot modes
118	☐	☐	☐	☐	autopilot values
119	☐	☐	☐	☐	weapon status
120	☐	☐	☐	☐	pressurization status
121	☐	☐	☐	☐	APU/GPU status
122	☐	☐	☐	☐	radar status
123	☐	☐	☐	☐	hydraulic status
124	☐	☐	☐	☐	elec & solar status
125	☐	☐	☐	☐	icing status 1
126	☐	☐	☐	☐	icing status 2
127	☐	☐	☐	☐	warning status
128	☐	☐	☐	☐	flite-plan legs
129	☐	☐	☐	☐	hardware options
130	☐	☐	☐	☐	camera location
131	☐	☐	☐	☐	ground location

detail: ☐ rotors
detail: ☐ propellers
detail: ☐ wings
detail: ☐ stabs

UDP rate 5 0.0 /sec
disk rate 1 0.0 /sec

Fig. 3 *Input and output data selection in the X-Plane Data I/O window. The selected flight sensor data is used as feedback to the controller. Controller commands from LabVIEW are also sent back to X-Plane through UDP*

3.1 UAV Model Description

The dynamic simulation and controller design approach discussed in this research is based on a DJI F450 Almost Ready to Fly Quadcopter. The 3D model of the DJI F450 quadcopter is constructed using Plane Maker software, which provides a graphical user interface (GUI) to construct any aircraft according to the physical specifications. Based on the aircraft geometry and propulsion parameters defined in Plane Maker, the X-Plane simulator is able to predict the aircraft's dynamic response using forces and moments calculated several times per second using Blade Element Theory. Given that the software uses the blade element approach, the effects of the environment, airspeed, and attitude of the aircraft can be set to a certain condition and calculated on a three-dimensional geometry. The X-Plane 3D model of the DJI F450 quadcopter is shown in Fig. 4, and some key physical parameters of the DJI F450 model are given in Table 1.

Table 1 *Physical characteristics of the DJI F450 model*

Parameters	Symbols	Value	Unit
Total weight	m_t	1.15	kg
Total motor mass	m	0.245	kg
Body radius	R	0.0549	m
Arm length to x-axis	l_x	0.213	m
Arm length to y-axis	l_y	0.213	m
Arm length to z-axis	l_z	0.302	m
Moment of inertia about x-axis	I_{xx}	0.046	kg.m^2
Moment of inertia about y-axis	I_{yy}	0.046	kg.m^2
Moment of inertia about z-axis	I_{zz}	0.091	kg.m^2
Maximum speed	V_a	27	m/s
Battery		3S-4S LiPo	
Propeller diameter		24 x 12.7	cm
Motor stator size		23 x 12	mm
Motor KV		960	rpm/V
ESC maximum allowable peak current		30	A

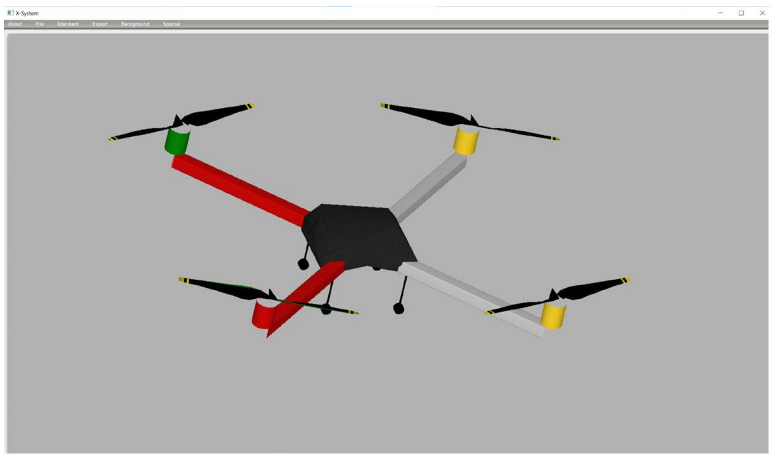


Fig. 4 *The 3D model of DJI F450 quadcopter in X-Plane's Plane Maker software*

4. Flight Controller Structure and Controller Design

The flight controller structure for quadcopter is typically constructed as multiple cascade proportional-integral-derivative (PID) control. The attitude dynamics are controlled by the inner loop controller, while the translational and yaw dynamics are controlled by the outer loop controller. In addition, the quadrotor's altitude and yaw dynamics are regulated using a single-stack PID controller. The attitude dynamics of helicopters are substantially faster than the translational dynamics. Before designing the outer loop controllers, the controllers in the inner loop should be properly tuned to make sure that the quadcopter follows attitude commands and generates the required torque controls (i.e., τ_ϕ and τ_θ). The outer loop controllers receive the position and yaw angle reference signals (i.e., $x_{ref}, y_{ref}, z_{ref}$ and ψ_{ref}) before passing the control signals to the inner loop controllers. The outer loop controllers produce the desired reference for attitude commands (i.e., θ_{ref} and ϕ_{ref}) for the inner loop controllers and generate the thrust T and yaw torque control τ_ψ directly based on normal velocity and yaw rate error. Fig. 5 below shows the overall flight control system where PID_θ and PID_ϕ are categorized as the inner loop controllers while PID_x, PID_y, PID_z and PID_ψ are categorized as the outer loop controllers. By using the same principle, the outer loop controllers can also be designed to track velocity and yaw rate reference instead of position and yaw angle reference, as suggested in [28]. A switching mechanism is introduced in our controller implementation to either track velocity-yaw rate or position-heading reference. Note that the position error is usually compared in an inertial frame, and a correction to the position error needs to be introduced to transfer the position errors into the body frame according to the respective yaw angle [29].

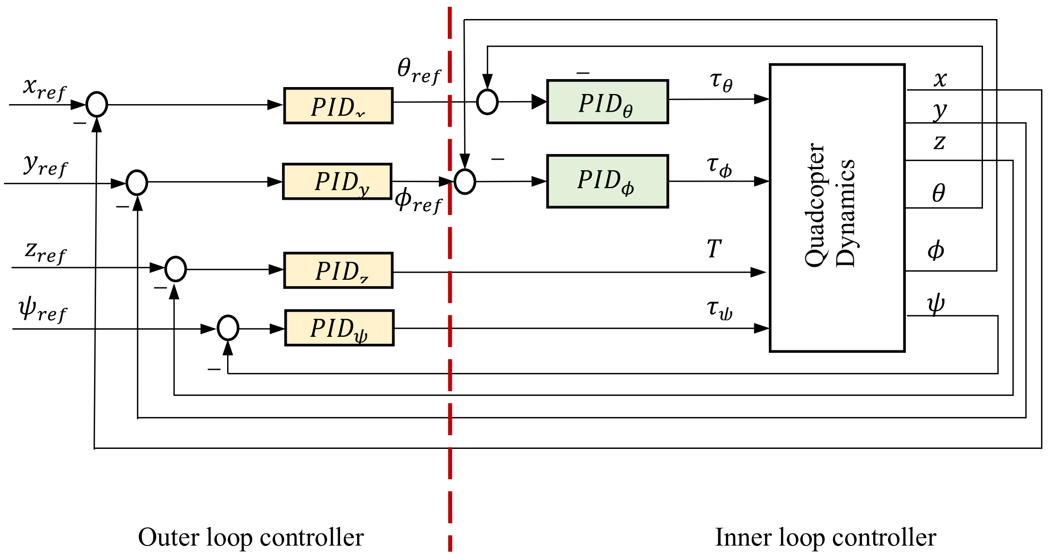


Fig. 5 *The overall flight control structure with position control in the outer loop*

Below is the linear state space equation and the equivalent transfer function model used to control the quadrotor UAV in terms of its altitude, attitude, position, and velocity.

4.1 Altitude Controller

The linear differential equations describing the altitude dynamic are as follows:

$$\dot{z} = w \tag{11}$$

$$\dot{w} = g - \frac{F}{m} \tag{12}$$

Equation (12) can be simplified into:

$$\dot{w} = T \tag{13}$$

where $T = g - F/m$. Equations (11) and (13) can be written in state space form as the followings:

$$\begin{bmatrix} \dot{z} \\ \dot{w} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} z \\ w \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} T \tag{14}$$

By applying the Laplace transform, Equation (14) can be represented in transfer function forms as follows:

$$\frac{z(s)}{T(s)} = \frac{1}{s^2} \qquad \frac{w(s)}{T(s)} = \frac{1}{s} \tag{15}$$

4.2 Position Controller

The linear differential equations responsible for position and velocity control in x direction are given as follows:

$$\dot{x} = u \qquad \dot{u} = -g\theta \tag{16}$$

For position and velocity in y direction, the linear differential equations that regulate position and velocity in y direction are:

$$\dot{y} = v \qquad \dot{v} = g\phi \tag{17}$$

The above equation from position and velocity in x and y direction can be written in the state space form:

$$\begin{bmatrix} \dot{x} \\ \dot{u} \\ \dot{y} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ u \\ y \\ v \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ -g & 0 \\ 0 & 0 \\ 0 & g \end{bmatrix} \begin{bmatrix} \theta \\ \phi \end{bmatrix} \tag{18}$$

By applying the Laplace transform, Equation (18) can be represented in transfer function forms as follows:

$$\frac{x(s)}{\theta(s)} = -\frac{g}{s^2} \qquad \frac{y(s)}{\phi(s)} = \frac{g}{s^2} \qquad \frac{u(s)}{\theta(s)} = -\frac{g}{s} \qquad \frac{v(s)}{\phi(s)} = \frac{g}{s} \tag{19}$$

4.3 Attitude Controller

In attitude control, we are interested in controlling the roll angle, pitch angle, roll rate, and pitch rate using rolling, pitching, and yawing torque inputs. The equations responsible for controlling the attitude dynamics motion are given as follows:

$$\begin{aligned} \dot{\phi} &= p \\ \dot{p} &= \frac{1}{I_{xx}}\tau_{\phi} \\ \dot{\theta} &= q \\ \dot{q} &= \frac{1}{I_{yy}}\tau_{\theta} \\ \dot{\psi} &= r \\ \dot{r} &= \frac{1}{I_{zz}}\tau_{\psi} \end{aligned} \tag{20}$$

Equation (20) can be combined and presented in the state-space form:

$$\begin{bmatrix}\dot{\phi}\\ \dot{p}\\ \dot{\theta}\\ \dot{q}\\ \dot{\psi}\\ \dot{r}\end{bmatrix}=\begin{bmatrix}0&1&0&0&0&0\\0&0&0&0&0&0\\0&0&0&0&0&0\\0&0&0&0&0&0\\0&0&0&0&0&1\\0&0&0&0&0&0\end{bmatrix}\begin{bmatrix}\phi\\ p\\ \theta\\ q\\ \psi\\ r\end{bmatrix}+\begin{bmatrix}0&0&0\\I_{xx}^{-1}&0&0\\0&0&0\\0&I_{yy}^{-1}&0\\0&0&0\\0&0&I_{zz}^{-1}\end{bmatrix}\begin{bmatrix}\tau_{\phi}\\ \tau_{\theta}\\ \tau_{\psi}\end{bmatrix} \tag{21}$$

The equivalent transfer function form of Equation (21) is as the follows:

$$\begin{array}{lll}\dfrac{\phi(s)}{\tau_{\phi}(s)}=\dfrac{I_{xx}^{-1}}{s^2} & \dfrac{\theta(s)}{\tau_{\theta}(s)}=\dfrac{I_{yy}^{-1}}{s^2} & \dfrac{\psi(s)}{\tau_{\psi}(s)}=\dfrac{I_{zz}^{-1}}{s^2}\\ \dfrac{p(s)}{\tau_{\phi}(s)}=\dfrac{I_{xx}^{-1}}{s} & \dfrac{q(s)}{\tau_{\theta}(s)}=\dfrac{I_{yy}^{-1}}{s} & \dfrac{r(s)}{\tau_{\psi}(s)}=\dfrac{I_{zz}^{-1}}{s}\end{array} \tag{22}$$

4.4 Successive Loop Closure Procedure

The successive loop closure approach is applied to design the PID controller for the attitude, altitude, and velocity/position of the RUAS. The main concept behind consecutive loop closure is to close some basic feedback loops around open-loop plant dynamics rather than building a single, complex control system. Consider the multi-loop control system with three transfer functions (i.e., $P_1(s)$, $P_2(s)$ and $P_3(s)$) as shown in Fig. 6. In the successive loop closure procedure, the closed loop feedback loop will be formed around y_1, y_2 and y_3 in sequence. The compensators (i.e., $C_1(s)$, $C_2(s)$ and $C_3(s)$) will be designed in succession. The inner loop must have the greatest frequency bandwidth as a precondition for designing successive loops, with subsequent loop bandwidths decreasing by a factor of 5-10 times in frequency [29].

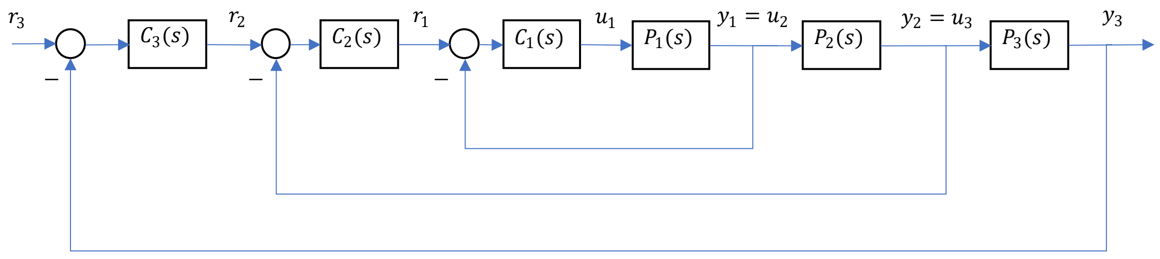


Fig. 6 *Multi loop closure design* [29]

Referring to the inner loop shown in Fig. 7, a closed loop feedback system can be formed from r_1 to y_1 with a frequency bandwidth, ω_{BW1}. The main presumption made is that the closed-loop transfer function $y_1(s)/r_1(s)$ can be approximated as a unity gain for frequencies much below ω_{BW1}. Figure 5 provides a graphic representation of this assumption. Then, the design of the second control loop is simplified by using only the plant transfer function $P_2(s)$ and the compensator C$_2$(s) when the inner loop transfer function is approximated as unity gain. The design for the third control loop can be done using the same procedure. The key to closing the loops sequentially is to design the following loop's bandwidth to be less than the previous loop, $\omega_{BW2} < \omega_{BW1}$ to ensure that the inner loop's frequency range is not violated. Due to the simplification, the conventional PID controller or lead-lag compensator design can be used effectively with tuning methods such as root locus or Ziegler-Nichols's method, which are commonly used. In this work, each PID controller in the RUAS will be tuned using either the PP approach or the LQR method.

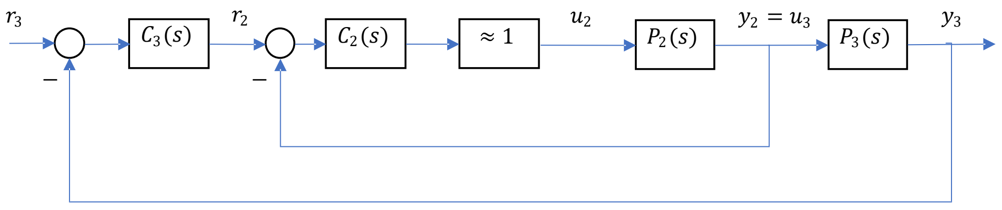


Fig. 7 *Design of successive loop closure with inner loop assumed as unity gain* [29]

4.5 PID Control Design Based on Pole Placement

The tuning method used to design the PID controller for the quadcopter is based on the PP method. The desired characteristic equation and dynamic parameters from the transfer function are used to determine the controller gains. This method must be used with the assumption that the system is controllable. There are only two types of models that are employed in the design of PID controllers: the first-order model and the second-order model. An approximation will be used if the plant dynamics give a higher order model to generate a first-order or second-order model, allowing a model-based method to be used to develop a PID controller [21]. For the first-order model, a PI controller will be used. However, the second-order model, PD controller, or PID controller can be used. Consider the block diagram shown in Fig. 8 as an example, where $R(s)$, $E(s)$, $U(s)$, and $Y(s)$ stand in for the Laplace transform variables of the reference input, error signal, control signal, and output signal, respectively.

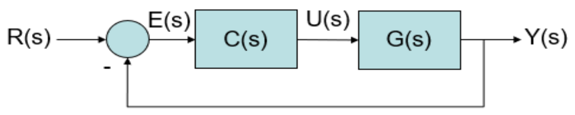


Fig. 8 *General control system block diagram*

The plant transfer function G(s) and the controller transfer function C(s) are assumed to be:

$$G(s) = \frac{b}{s+a} \tag{23}$$

$$C(s) = K_p + \frac{K_i}{s} + \frac{K_d s}{\tau_f s + 1} \tag{24}$$

where K_i is the integral gain and K_p is the proportional gain. The controller transfer function in Equation (24) represents a general PID controller form with a derivative filter. Considering the derivative gain $K_d = 0$, for PI controller form, yields:

$$C(s) = \frac{K_p s + K_i}{s} \tag{25}$$

The closed-loop transfer function from the reference signal to the output signal is expressed as:

$$\frac{Y(s)}{R(s)} = \frac{G(s)C(s)}{1 + G(s)C(s)} = \frac{b(K_p s + K_i)}{s^2 + (a + bK_p)s + bK_i} \tag{26}$$

The characteristics equation of the closed-loop system can be matched with the desired characteristics equation using the following relationship:

$$s^2 + (a + bK_p)s + bK_i = As^2 + Bs + C \tag{27}$$

Hence, a simple formulation between PID controller gain and the plant parameters (a, b) and the desired characteristics equation parameters can be established as follows:

$$a + bK_p = B$$

$$K_p = \frac{B - a}{b} \tag{28}$$

$$bK_i = C$$

$$K_i = \frac{C}{b} \tag{29}$$

The same calculation procedures can be easily repeated for PD and PID controllers with derivative filters for second-order systems, as listed in Table 2. A derivative filter included in the PID controller design is essential because it avoids the amplification of measurement noise, hence improving the PID control system's robustness [21].

Table 2 *PID tuning equation parameters for first and second order transfer function model*

First Order Transfer Function Model $G(s) = \frac{b}{s+a}$	**Controller:** PI	**Desired Characteristic Equation:** $As^2 + Bs + C$
Proportional Gain	$K_p = \frac{B-a}{b}$	
Integral Gain	$K_i = \frac{C}{b}$	
Second Order Transfer Function Model $G(s) = \frac{b}{s^2 + a_1 s + a_2}$	**Controller:** PD	**Desired Characteristic Equation:** $As^2 + Bs + C$
Proportional Gain	$K_p = \frac{C - a_2}{b}$	
Derivative Gain	$K_d = \frac{B - a_1 - bK_p\tau_f}{b}$	
Second Order Transfer Function Model $G(s) = \frac{b}{s^2 + a_1 s + a_2}$	**Controller:** PID	**Desired Characteristic Equation:** $As^4 + Bs^3 + Cs^2 + Ds + E$
Proportional Gain	$K_p = \frac{(D - bK_i)\tau_f - a_2}{b}$	
Integral Gain	$K_i = \frac{E\tau_f}{b}$	
Derivative Gain	$K_d = \frac{(C - a_2 - bK_p)\tau_f - a_1}{b}$	
Filter Time Constant	$\tau_f = \frac{1}{B - a_1}$	

5. Results and Discussion

This section presents the simulation results of the AFCS design using the PID gain values tuned using the pole placement technique. After completion of the PID gain calculation, two trajectory tracking simulations for the DJI F450 quadcopter were carried out in the SITL simulation environment for hover and circular trajectories. In the hover trajectory simulation, the proposed flight controller is simulated under ideal weather conditions without wind gusts, and the circular trajectory is simulated with a light wind speed of 10 knots (4.5 m/s) at 0º (left crosswind) with Level 2 turbulence. The trajectory tracking results were obtained using X-Plane data output over network commands. The flight data obtained was processed using Microsoft Excel or LabVIEW software and validated in a series of flight tests to analyze its flight performance characteristics and determine the efficiency of the control systems.

5.1 PID Gains Results

Table 3 displays the results of the PID gain tuning using the PP method, along with the desired pole locations, the proportional gain, K_p, integral gain, K_i, derivative gain, K_d, and the filter time constant, τ_f values. The derivative filter is utilized mainly in the design of pitch and roll angle controllers to filter out rapid PID output changes.

Table 3 *The PID gains obtained using Pole Placement (PP) method*

Transfer function [28]	**Pole Placement (PP) method**			**Controller output range**
	PI controller	**PID controller**	**Desired pole location**	
$\frac{\phi(s)}{\tau_\phi(s)} = \frac{21.74}{s^2}$	-	$K_{p_\phi} = 0.69$ $K_{i_\phi} = 0.736$ $K_{d_\phi} = 0.233$ $\tau_f = 0.063$	$-4, -4, -4, -4$	± 1
$\frac{\theta(s)}{\tau_\theta(s)} = \frac{21.74}{s^2}$	-	$K_{p_\theta} = 0.69$ $K_{i_\theta} = 0.736$ $K_{d_\theta} = 0.233$ $\tau_f = 0.063$	$-4, -4, -4, -4$	± 1
$\frac{\psi(s)}{\tau_\psi(s)} = \frac{10.99}{s^2}$	-	$K_{p_\psi} = 0.273$ $K_{i_\psi} = 0.091$ $K_{d_\psi} = 0.273$	$-1, -1, -1$	± 1
$\frac{r(s)}{\tau_\psi(s)} = \frac{10.99}{s}$	$K_{p_r} = 0.364$ $K_{i_r} = 0.364$	-	$-2, -2$	±1
$\frac{w(s)}{T(s)} = \frac{3.478}{s}$	$K_{p_w} = 0.575$ $K_{i_w} = 0.2875$	-	$-1, -1$	$T_{max} = 3$ $T_{min} = -1$
$\frac{u(s)}{\theta(s)} = -\frac{9.81}{s}$	$K_{p_u} = -0.2039$ $K_{i_u} = -0.1019$	-	$-1, -1$	± 0.5 rad
$\frac{v(s)}{\phi(s)} = \frac{9.81}{s}$	$K_{p_v} = 0.2039$ $K_{i_v} = 0.1019$	-	$-1, -1$	± 0.5 rad
$\frac{x(s)}{\theta(s)} = -\frac{9.81}{s^2}$	-	$K_p = -0.076$ $K_i = -0.013$ $K_d = -0.153$	$-0.5, -0.5$, -0.5	± 0.5 rad
$\frac{y(s)}{\phi(s)} = \frac{9.81}{s^2}$	-	$K_p = 0.076$ $K_i = 0.013$ $K_d = 0.153$	$-0.5, -0.5, -0.5$	± 0.5 rad
$\frac{z(s)}{T(s)} = \frac{1}{s^2}$	-	$K_{p_z} = 0.750$ $K_{i_z} = 0.125$ $K_{d_z} = 1.5$	$-0.5, -0.5, -0.5$	$T_{max} = 3$ $T_{min} = -1$

5.2 Trajectory Tracking Simulation

To evaluate the PID controller's ability to maintain its position, hover flight tests were performed with the following position references (x_d, y_d and z_d):

$$x_d = 0, \qquad \forall t \tag{30}$$

$$y_d = 0, \qquad \forall t \tag{31}$$

$$z_d = 15\,m, \qquad 0 \le t \le 15s \tag{32}$$

$$\psi_d = 80°, \qquad \forall t \tag{33}$$

The circular trajectory references are given by Equations (34)-(37) to evaluate the controller's performance in a low-speed forward flight condition.

$$x_d = 5\,sin\,\omega t, \qquad 0 \le t \le 45s \tag{34}$$

$$y_d = 5\cos\omega t\,, \qquad 0 \le t \le 45s \tag{35}$$

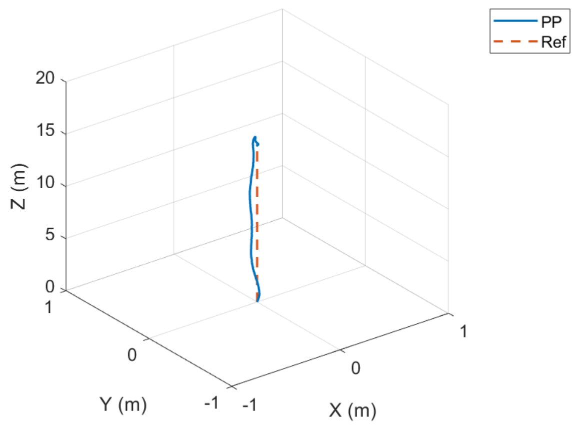
$$z_d = 10\ m, \qquad \forall t \tag{36}$$

$$\psi_d = 80°, \qquad \forall t \tag{37}$$

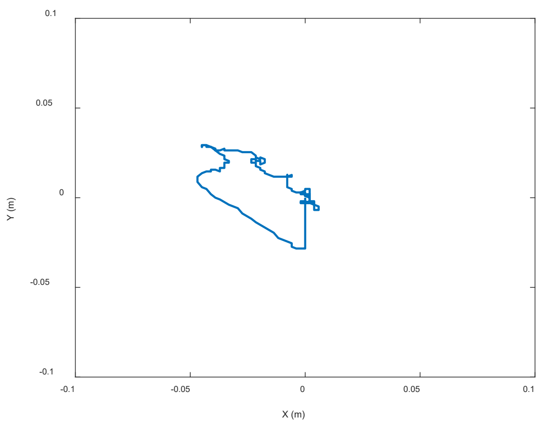
where ω represents the angular velocity $= 0.5$ rad/s.

5.2.1 Results for Hover Control

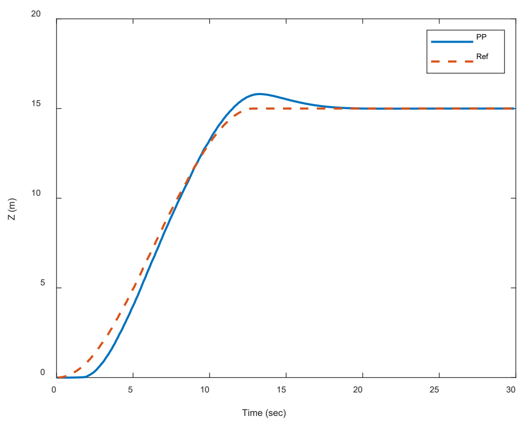
Fig. 9 shows the 3D position trajectory for hovering maneuver simulation with the PID controller tuned using the PP method and the time histories of position trajectories and attitude angle responses for each axis. Based on responses recorded in Fig. 9, PID controllers tuned by the PP method can track the desired hovering trajectories with good accuracy. The PID controllers maintain the desired position states within ±0.05 m for x and y-positions with a small tracking error, as shown in Table 4. The proposed PID controllers are found to be capable of stabilizing the roll, pitch, and yaw angles of the quadcopter within the specified state constraints (±0.5 rad or ±28°) in less than 10 seconds. Fig. 10 shows the controller input response produced by the PID controller tuned by the PP tuning method, which includes the quadcopter's rotor thrust, yawing moment input, rolling moment input, and pitching moment input. The pitching moment control input is used to move the quadcopter forward and backward during hover flight, rolling moment control input is used to move the quadcopter sideways, while the rotor thrust and yawing moment input control the altitude and heading changes during hover flight. Results show that the proposed PID controllers produce control input responses within the specified actuator limit range. Therefore, in terms of trajectory tracking accuracy, it can be concluded that for hovering maneuvers, the proposed PID controller tuned by the PP method can produce satisfactory tracking accuracy with minimal control effort.



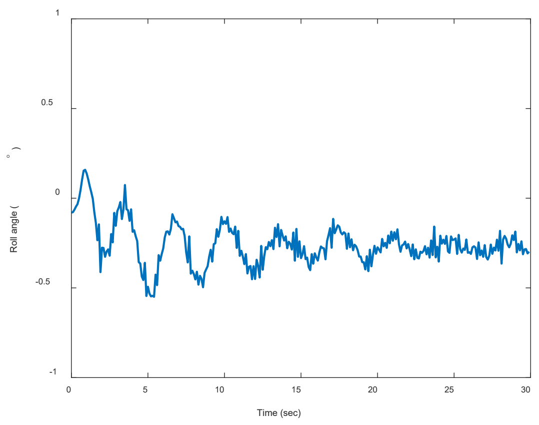
(a)



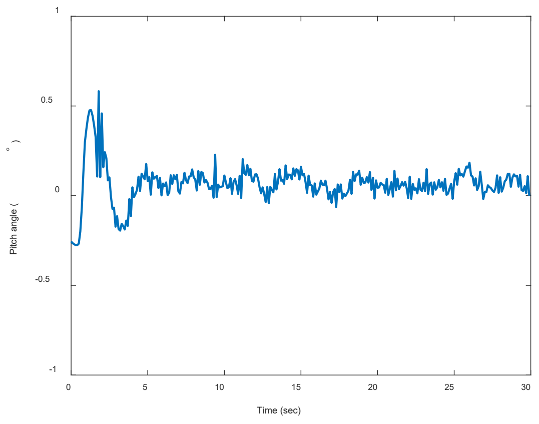
(b)



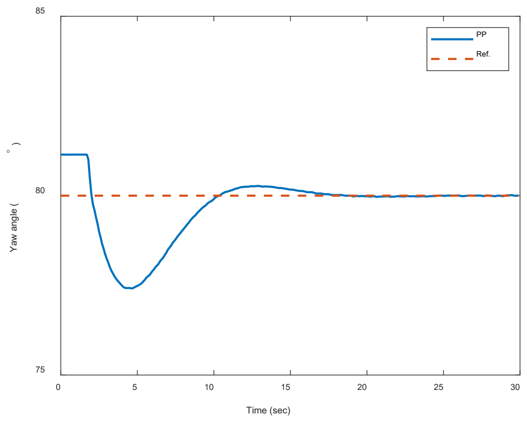
(c)



(d)



(e)

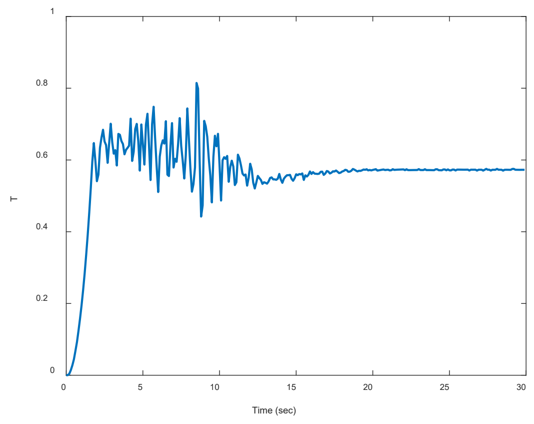


(f)

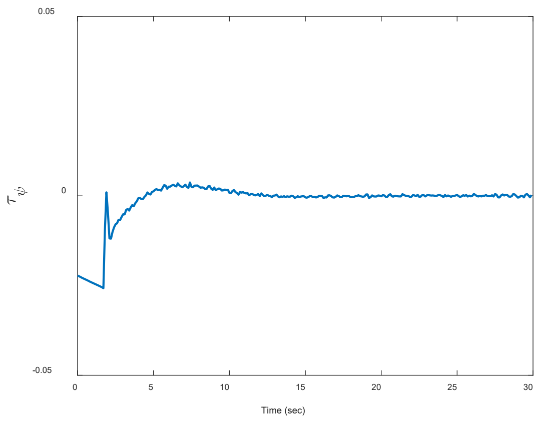
Fig. 9 *Hovering tracking performance for PP based PID tuning. (a) 3D-position (b) xy-position (c) z-position (d) roll angle (e) pitch angle (f) yaw angle*

Table 4 *Hover performance evaluation*

Specification	Desired Level	Pole Placement (PP) based PID
Time taken to stabilized hover	≤ 20 s	18 s
Longitudinal position error	≤ 0.5 m	0.0105 m
Lateral position error	≤ 0.5 m	0.0090 m
Heading error	≤ 3°	0.4927°
Altitude error	≤ 3 m	0.3014 m



(a)



(b)

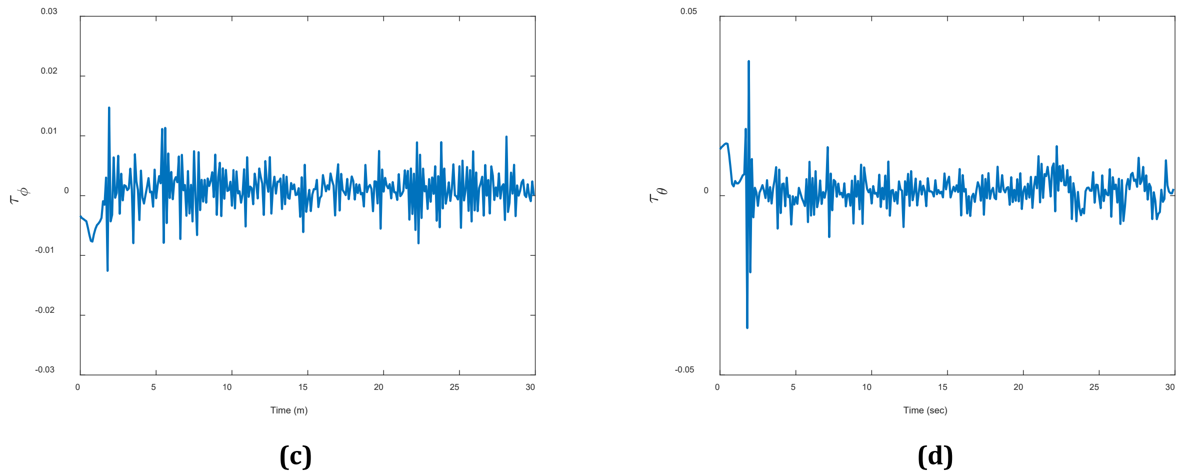
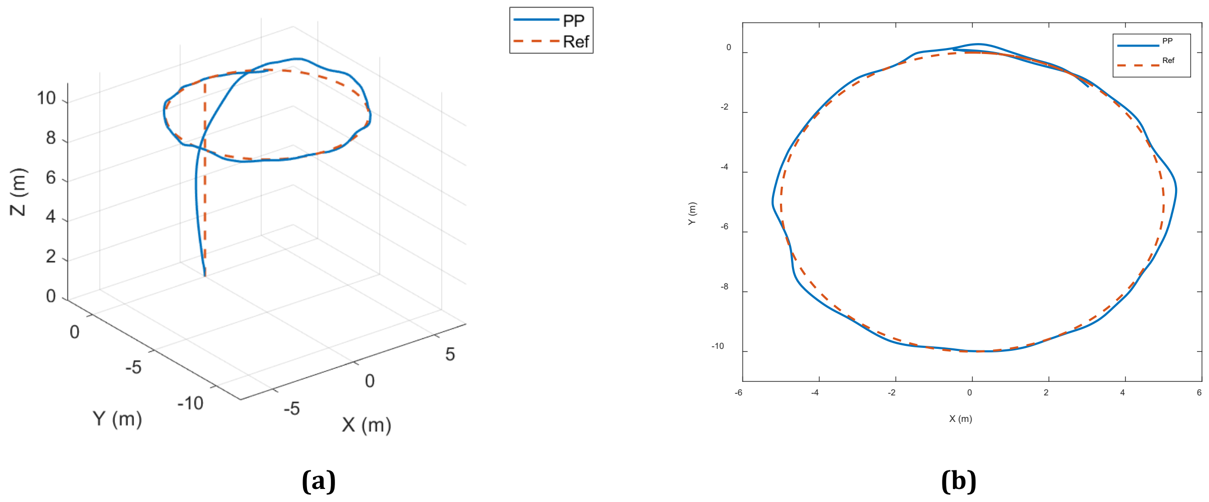
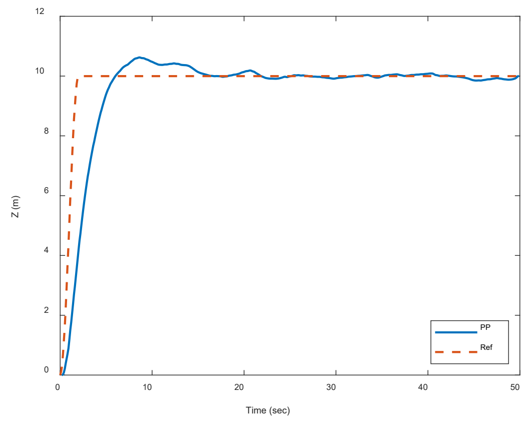


Fig. 10 *Controller input responses produced by PID controllers tuned using PP method during hovering maneuver. (a) Thrust (b) Yawing torque input (c) Rolling torque input (d) Pitching torque input*

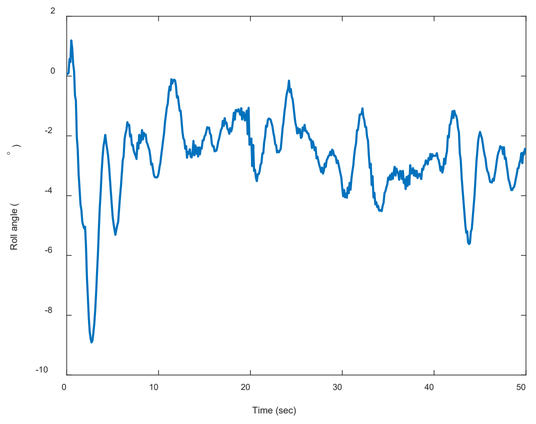
5.2.2 Results for Circular Path Tracking

Fig. 11 shows the circular path tracking simulation using the proposed PP tuning method. The simulation results indicate that the PID controller tuned by the PP method can track the desired circular trajectories with wind disturbances with good accuracy. Table 5 shows the error evaluation of the controller response. The controller can achieve position tracking accuracy with 0.1306 m and 0.0832 m in longitudinal and lateral position errors, respectively. The proposed controllers can also stabilize the roll and pitch angles of the quadcopter within the specified state constraints (±0.5 rad or ±28°). Fig. 12 shows the controller input responses produced by the proposed PID tuning for the circular path maneuver. It is evident that the controller utilizing the PP tuning method, effectively generates a control input response that remains within the specified actuator limit and is consistently maintained within a reasonable limit.

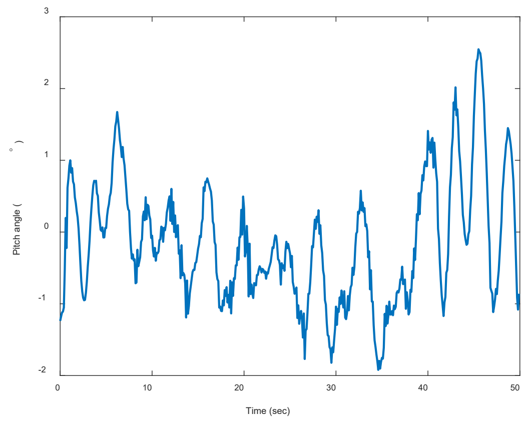




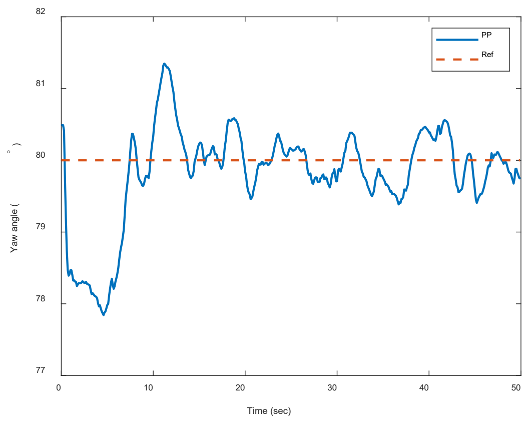
(c)



(d)



(e)



(f)

Fig. 11 *Circular path tracking performance for PP based PID tuning. (a) 3D-position; (b) xy-position; (c) z-position; (d) roll angle; (e) pitch angle; (f) yaw angle*

Table 5 *Circular trajectory tracking performance evaluation*

Specification	Desired Level	Pole Placement (PP) based PID
Maneuver completion time	≤ 60 s	$50\ s$
Longitudinal position error	≤ 0.5 m	$0.1306\ m$
Lateral position error	≤ 0.5 m	$0.0832\ m$
Heading error	$\leq 3°$	$0.4218°$
Altitude error	≤ 3 m	$0.2881\ m$

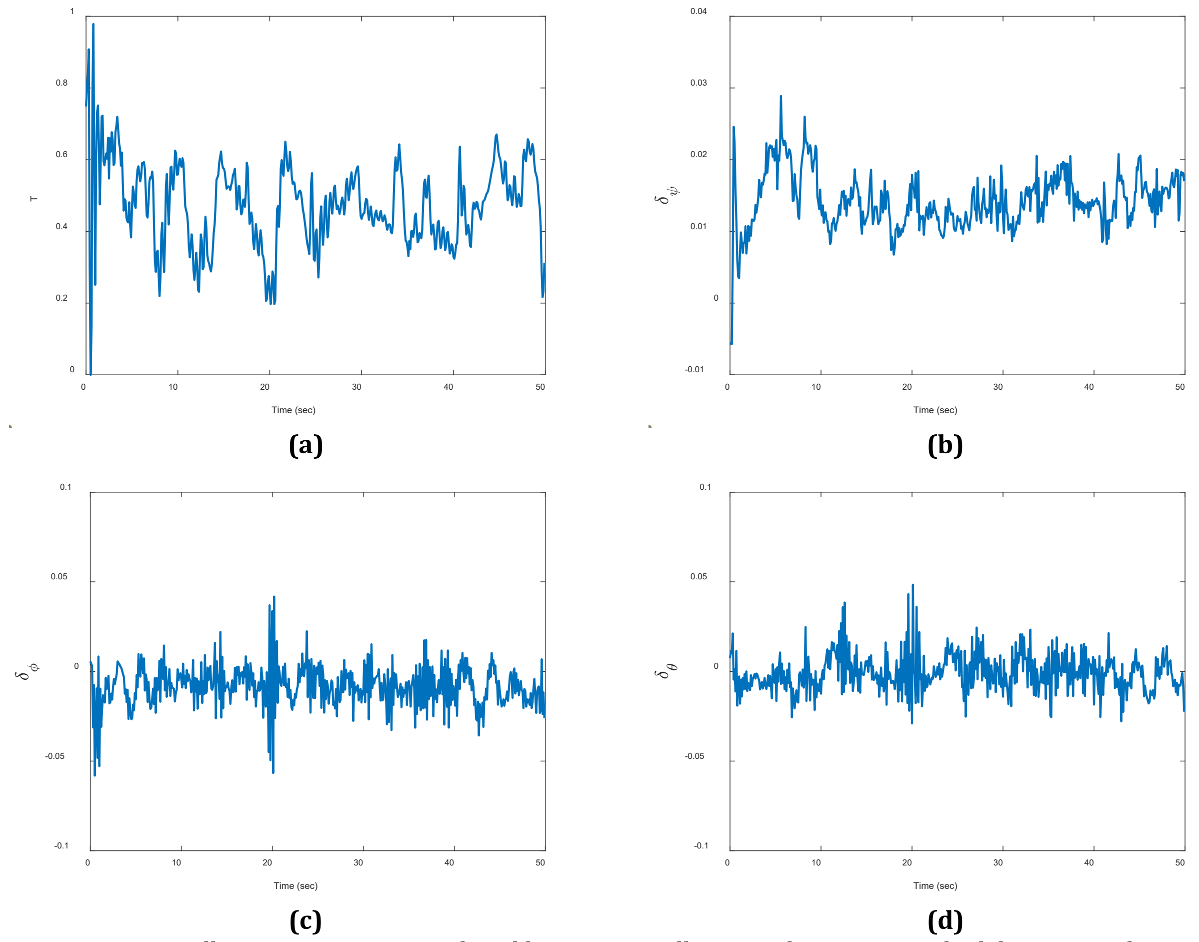


Fig. 12 *Controller input response produced by PID controllers tuned using PP method during circular path maneuver. (a) Thrust; (b) Yawing moment input; (c) Roll moment input; (d) Pitching moment input*

6. Conclusion

The research study focuses on the development of a flight control design simulation for an unmanned quadcopter system and the evaluation of the flight controller's performance in hovering and circular trajectory tracking. This paper presents a simple and practical controller design approach for quadcopter controller design based on PID gain tuning using the Pole Placement (PP) method. The controller design was implemented using LabVIEW software, while the X-Plane flight simulator was used to simulate the response of the RUAS. The obtained results show that the PID controller tuned using the PP method is capable of producing good tracking accuracy for hovering and circular path trajectories and showed adequate robustness in dealing with wind turbulence in a circular path tracking simulation. It is recommended that future work utilize the hardware-in-the loop (HIL) simulation approach for the validation of the proposed controller before actual flight tests. By adopting the HIL simulation approach, the reliability of the autopilot hardware and closed-loop system performance can be safely tested in a virtual environment.

Acknowledgement

This research was supported by Universiti Tun Hussein Onn Malaysia (UTHM) through Tier 1 (Vot. Q112) research grant.

Conflict of Interest

The authors declare that there is no conflict of interest regarding the publication of the paper.

Author Contribution

The authors confirm their contribution to the paper as follows: ***study conception and design:*** *Syariful Syafiq Shamsudin, Mohammad Fahmi Pairan;* ***data collection:*** *Annis Syahirah Jemsa, Syariful Syafiq Shamsudin;* ***analysis and interpretation of results:*** *Annis Syahirah Jemsa, Syariful Syafiq Shamsudin, Mohammad Fahmi Pairan;* ***draft***

***manuscript preparation:** Annis Syahirah Jemsa, Syariful Syafiq Shamsudin, Mohammad Fahmi Pairan. All authors reviewed the results and approved the final version of the manuscript.*

References

[1] F. Kendoul, Z. Yu, and K. Nonami, “Guidance and nonlinear control system for autonomous flight of minirotorcraft unmanned aerial vehicles,” *J Field Robot*, vol. 27, no. 3, pp. 311–334, 2010, doi: 10.1002/rob.20327.

[2] I. Lopez-Sanchez and J. Moreno-Valenzuela, “PID control of quadrotor UAVs: A survey,” *Annu Rev Control*, vol. 56, no. 1, p. 100900, 2023, doi: 10.1016/j.arcontrol.2023.100900.

[3] M. Chipofya, D. J. Lee, and K. T. Chong, “Trajectory Tracking and Stabilization of a Quadrotor Using Model Predictive Control of Laguerre Functions,” *Abstract and Applied Analysis*, vol. 2015, pp. 1–11, 2015, doi: 10.1155/2015/916864.

[4] Q. Quan, *Introduction to Multicopter Design and Control*, 1st ed. Singapore: Springer Singapore, 2017. doi: 10.1007/978-981-10-3382-7.

[5] A. Salih, M. Moghavvemi, M. Haf, and K. Gaeid, “Flight PID Controller Design for a UAV Quadrotor,” *Scientific research and essays*, vol. 5, pp. 3660-3667., Dec. 2010.

[6] A. Sheta, M. Braik, D. R. Maddi, A. Mahdy, S. Aljahdali, and H. Turabieh, “Optimization of PID Controller to Stabilize Quadcopter Movements Using Meta-Heuristic Search Algorithms,” *Applied Sciences*, vol. 11, no. 14, p. 6492, Jul. 2021, doi: 10.3390/app11146492.

[7] Z. He and L. Zhao, “A Simple Attitude Control of Quadrotor Helicopter Based on Ziegler-Nichols Rules for Tuning PD Parameters,” *The Scientific World Journal*, vol. 2014, p. 280180, 2014, doi: 10.1155/2014/280180.

[8] S. B. Joseph, E. G. Dada, A. Abidemi, D. O. Oyewola, and B. M. Khammas, “Metaheuristic algorithms for PID controller parameters tuning: review, approaches and open problems,” *Heliyon*, vol. 8, no. 5, p. e09399, May 2022, doi: 10.1016/j.heliyon.2022.e09399.

[9] G. Cai, B. M. Chen, and T. H. Lee, *Unmanned Rotorcraft Systems*, 1st ed. in Advances in Industrial Control. Springer-Verlag London, 2011. doi: 10.1007/978-0-85729-635-1.

[10] M. B. Tischler and R. K. Remple, *Aircraft and Rotorcraft System Identification, Second Edition*. 2012. doi: 10.2514/4.868207.

[11] S. S. Shamsudin and X. Chen, “Recursive Gauss-Newton based training algorithm for neural network modelling of an unmanned helicopter dynamics,” in *Mechatronics and Machine Vision in Practice (M2VIP), 2012 19th International Conference*, 2012, pp. 92–99.

[12] M. F. Pairan, S. S. Shamsudin, M. F. Yaakub, and M. S. M. Anwar, “Real-time system identification of an unmanned quadcopter system using fully tuned radial basis function neural networks,” *Int J Model Identif Control*, vol. 37, no. 2, p. 128, 2021, doi: 10.1504/IJMIC.2021.120209.

[13] V. M. Babu, K. Das, and S. Kumar, “Designing of self tuning PID controller for AR drone quadrotor,” in *2017 18th International Conference on Advanced Robotics (ICAR)*, IEEE, Jul. 2017, pp. 167–172. doi: 10.1109/ICAR.2017.8023513.

[14] J. O. Pedro, M. Dangor, and P. J. Kala, “Differential evolution-based PID control of a quadrotor system for hovering application,” in *2016 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, Jul. 2016, pp. 2791–2798. doi: 10.1109/CEC.2016.7744141.

[15] A. Noordin, M. A. Mohd Basri, Z. Mohamed, and A. F. Zainal Abidin, “Modelling and PSO Fine-tuned PID Control of Quadrotor UAV,” *Int J Adv Sci Eng Inf Technol*, vol. 7, no. 4, p. 1367, Aug. 2017, doi: 10.18517/ijaseit.7.4.3141.

[16] C. Sravan Bharadwaj, T. Sudhakar Babu, and N. Rajasekar, “Tuning PID Controller for Inverted Pendulum Using Genetic Algorithm,” in *Advances in Systems, Control and Automation. Lecture Notes in Electrical Engineering*, vol. 442, 2018, pp. 395–404. doi: 10.1007/978-981-10-4762-6_38.

[17] M. Rathinam, W. I. Maria Siluvairaj, and A. Ramaveerapathiran, “Tuning of Robust PID Controller with Filter for SISO System Using Evolutionary Algorithms,” *Studies in Informatics and Control*, vol. 26, no. 3, Sep. 2017, doi: 10.24846/v26i3y201703.

[18] L. J. Mpanza and J. O. Pedro, “Optimised Tuning of a PID-Based Flight Controller for a Medium-Scale Rotorcraft,” *Algorithms*, vol. 14, no. 6, p. 178, Jun. 2021, doi: 10.3390/a14060178.

[19] P. Poksawat and L. Wang, “Automatic tuning of hexacopter attitude control systems with experimental validation,” in *2017 21st International Conference on System Theory, Control and Computing (ICSTCC)*, IEEE, Oct. 2017, pp. 753–758. doi: 10.1109/ICSTCC.2017.8107127.

[20] V. Bagyaveereswaran, Subhashini, A. Sahu, and R. Anitha, “Optimal Control of Roll Axis of Aircraft Using PID Controller,” in *Soft Computing for Problem Solving. Advances in Intelligent Systems and Computing*, vol. 1048, 2020, pp. 961–969. doi: 10.1007/978-981-15-0035-0_76.

[21] L. Wang, *PID Control System Design and Automatic Tuning using MATLAB/Simulink*. Wiley, 2020. doi: 10.1002/9781119469414.

[22] S. Anshuman, “Modeling, Design and Control of a 6 D-O-F Quadcopter Fleet with Platooning Control,” Master of Science, Arizona State University, 2021. Accessed: Dec. 18, 2023. [Online]. Available: https://hdl.handle.net/2286/R.2.N.161257

[23] M.-H. Do, C.-E. Lin, and Y.-C. Lai, “Validation of the Flight Dynamics Engine of the X-Plane Simulator in Comparison with the Real Flight Data of the Quadrotor UAV Using CIFER,” *Drones*, vol. 7, no. 9, p. 548, Aug. 2023, doi: 10.3390/drones7090548.

[24] A. Meyer, “X-Plane 9 Operation Manual.” X-Plane, pp. 1–229, 2014. Accessed: May 03, 2023. [Online]. Available: https://www.x-plane.com/files/manuals/X-Plane_Desktop_manual.pdf

[25] A. Bellchambers, “Creation of a LabVIEW based autopilot to control X-Plane.” GitHub, 2017. Accessed: May 03, 2023. [Online]. Available: https://github.com/bellcham/LV_XP_AP

[26] L. Yu, G. He, S. Zhao, X. Wang, and L. Shen, “Design and Implementation of a Hardware-in-the-Loop Simulation System for a Tilt Trirotor UAV,” *J Adv Transp*, vol. 2020, pp. 1–17, Oct. 2020, doi: 10.1155/2020/4305742.

[27] C. S. Jamadagni, C. U. Chethan, Y. Jeppu, S. B. Kamble, and V. H. Desai, “System simulation approach for helicopter autopilot,” in *2014 International Conference on Contemporary Computing and Informatics (IC3I)*, IEEE, Nov. 2014, pp. 404–408. doi: 10.1109/IC3I.2014.7019616.

[28] Y. Ben, “Development of a miniature low-cost UAV helicopter autopilot platform and formation flight control of UAV teams,” Department of Electrical and Computer Engineering, National University of Singapore, 2010. [Online]. Available: http://scholarbank.nus.edu.sg/handle/10635/17758

[29] Cunjia Liu, “Advanced control for miniature helicopters: modelling, design and flight test,” Doctoral Thesis, Loughborough University, 2011. Accessed: Jan. 01, 2023. [Online]. Available: https://repository.lboro.ac.uk/account/articles/9219725