REPUBLIC OF TURKEY

YILDIZ TECHNICAL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

EVALUATION OF NONLINEAR CONTROL METHODS FOR FOUR ROTOR AIRCRAFT

Fahed SAYED

MASTER OF SCIENCE THESIS

Department of Control & Automation Engineering

Control & Automation Engineering Program

Advisor

Assoc. Prof. Dr. Türker TÜRKER

August, 2019

REPUBLIC OF TURKEY

YILDIZ TECHNICAL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

EVALUATION OF NONLINEAR CONTROL METHODS FOR FOUR ROTOR AIRCRAFT

A thesis submitted by Fahed SAYED in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE** is approved by the committee on 01.08.2019 in Department of Control & Automation Engineering.

Assoc. Prof. Dr. Türker TÜRKER

Yıldız Technical University

Advisor

Approved By the Examining Committee

Assoc. Prof. Dr. Türker TÜRKER, Advisor
Yıldız Technical University _____

Assist. Prof. Dr. Claudia Fernanda YAŞAR, Member
Yıldız Technical University _____

Assist. Prof. Dr. Ali Fuat ERGENÇ, Member
Istanbul Technical University _____

I hereby declare that I have obtained the required legal permissions during data collection and exploitation procedures, that I have made the in-text citations and cited the references properly, that I haven't falsified and/or fabricated research data and results of the study and that I have abided by the principles of the scientific research and ethics during my Thesis Study under the title of Location Analysis of The Emergency Service Centers Of a Case Company supervised by my supervisor, Assoc. Prof. Dr. Türker TÜRKER. In the case of a discovery of false statement, I am to acknowledge any legal consequence.

Fahed SAYED

Dedicated to my family

and my best friends

ACKNOWLEDGEMENTS

I would like to thank my advisor Assoc. Prof. Dr. Türker Türker who helped and supported me in my MSc thesis. I have benefited greatly from his experiences and knowledge.

I would also like to thank all my teachers in Yildiz technical university especially our head of department Prof. Dr. Seref Naci Engin who supported us during the three years of the master program.

Finally, I must express my very profound gratitude to my parents and my family and to all of my friends who support me throughout three years away from them or being here around me.

I'm grateful to you all.

Fahed SAYED

TABLE OF CONTENTS

LIST OF SYMBOLS

A	Blade Area
C_D	Aerodynamic Drag Coefficient
C_T	Aerodynamic Thrust Coefficient
F_B	Non Gravitational Forces Acting On the Quadrotor
I_{xx}	Area Moment Of Inertia around the X-Axis
I_{yy}	Area Moment Of Inertia around the Y-Axis
I_{zz}	Area Moment Of Inertia around the Z-Axis
J	Quadrotor's Diagonal Inertia Matrix
J_r	Rotor's Inertia
K_M	Aerodynamic Moment Constant
K_f	Aerodynamic Force Constant
K_r	Aerodynamic Rotation Coefficient Matrix
K_t	Aerodynamic Translation Coefficient Matrix
K_{mot}	Motor torque constant
l_{mot}	Motor torque constant
M_B	Moments That Affect Quadrotor
R	Rotation Matrix
R_{mot}	Motor circuit resistance
U	Control input vector
V	Lyapunov Function
X	State vector
Ω_h	Hover Angular Velocity
Ω_n	Speed of Rotor n

Ω_r	Rotor's Relative Speed
$\dot{\eta}$	Euler Rates
ω	Angular Body Rates
ϕ	Roll Angle
ϕ_d	Desired Roll
ψ	Yaw Angle
ψ_d	Desired Yaw
θ	Pitch angle
θ_d	Desired Pitch
ρ	Air Density
e	Error
g	Gravitational Acceleration g =9.81 m/s^2
K_p	Proportional gain
K_i	Integral gain
K_d	Derivative gain
l	Moment Arm
m	Mass of Quadrotor
r	Radius Of Blade
r_b	Propeller's Blade Radius
s	Sliding Surface
x_d	Desired x Position
y_d	Desired y Position
z_d	Desired z Position

LIST OF ABBREVIATIONS

ANFIS	Adaptive Network-Based Fuzzy Inference System
APSO	Adaptive Particle Swarm Optimization
FLC	Fuzzy Logic Control
HALE	High-Altitude Long-Endurance
LTI	Linear Time Invariant
MALE	Medium-Altitude Long-Endurance
MAV	Micro Unmanned Aerial Vehicle
MEMS	Micro-Electro-Mechanical Systems
MPC	Model Predictive Control
MRAC	Model Reference Adaptive Control
NAV	Nano Unmanned Aerial Vehicle
NN	Neural Networks
PID	Proportional–Integral–Derivative
PSO	Particle Swarm Optimization
RLC	Reinforcement Learning Control
SMC	Sliding Mode Control
TUAV	Tactical Unmanned Aerial Vehicle
UAV	Unmanned Aerial Vehicle
VSC	Variable Structure Control
VTOL	Vertically Take-Off and Land

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Evaluation Of Nonlinear Control Methods For Four Rotor Aircraft

Fahed SAYED

Department of Control & Automation Engineering

Master of Science Thesis

Advisor: Assoc. Prof. Dr. Türker TÜRKER

In this thesis, a nonlinear mathematical model of the quadrotor system taking into account aerodynamic affects is presented in detail. The nonlinear model of the quadrotor dynamics is extracted using the Newton–Euler equations. The quadrotor dynamics can be separated into translational and rotational dynamics. The translational dynamics are responsible for controlling the position of the quadrotor (altitude z, x, and y), while the rotational dynamics are responsible for controlling the roll, pitch, and yaw angles. Although the quadrotor system is under actuated with six degrees of freedom and four actuators, the derived rotational dynamics is fully actuated, while the translational dynamics is still under actuated.

Linear and nonlinear control methods are developed to stabilize and control the roll, pitch, yaw, and altitude movements of the quadrotor system. The applied control methods are linear PID control, nonlinear Backstepping control, and Sliding Mode Control (SMC) approach.

The parameters of the PID controllers were tuned using particle swarm optimization algorithm (PSO) to improve their performance. A new adaptive particle swarm optimization (APSO) algorithm is proposed to improve the

performance of the standard PSO algorithm. It can adapt the inertia weight parameter to balance between the global and local search processes that give better search competence and convergence speed. Simulation-based results were compared to evaluate the performance of the developed control approaches in terms of dynamic performance, stability and the effect of external disturbances.

Keywords: Nonlinear Control, UAV, Backstepping Control, Sliding Mode Control, APSO, PID.

ÖZET

Dört Rotorlu Hava Araçlarıiçin Doğrusal Olmayan Kontrol Yöntemlerinin İncelenmesi

Fahed SAYED

Kontrol ve Otomasyon Mühendisliği Bölümü

Yüksek Lisans Tezi

Danışman: Doç. Dr. Türker TÜRKER

Bu tezde, aerodinamik etkileri hesaba katan dört rotorlu sistemin doğrusal olmayan bir matematiksel modeli ayrıntılı olarak sunulmuştur. Dört rotorlu sistem dinamiğinin doğrusal olmayan modeli, Newton-Euler denklemleri kullanılarak elde edilmiştir. Dört rotorlu sistem dinamikleri ötelenme ve dönme dinamikleri olarak 2 kısma ayrılabilir. Ötelenme dinamikleri, dört rotorlu sistemin pozisyonunu (irtifa z, x, y) kontrol etmekten sorumluyken, dönme dinamikleri yatış, yunuslama ve dönme açılarını kontrol etmekten sorumludur. Her ne kadar dört rotorlu sistem, altı serbestlik derecesi ve dört aktüatör ile eksik tahrikli olsa da, öteleme dinamiği halen eksik tahrikli olduğunda elde edilen dönme dinamiği tam tahriklidir.

Dört rotorlu sistemin yatış, yunuslama, dönme ve irtifa hareketlerini dengelemek ve kontrol etmek üzere doğrusal ve doğrusal olmayan kontrol metodları geliştirilmiştir. Uygulanan kontrol yöntemleri doğrusal PID Kontrol, doğrusal olmayan Geri Adımlamalı Kontrol ve Kayan Kipli Kontrol (KKK) yaklaşımıdır.

Bahsedilen PID kontrolörlerin parametreleri, performanslarını artırmak üzere parçacık sürüsü optimizasyon algoritması (PSO) kullanılarak ayarlanmıştır.

Standart PSO algoritmasının performansını geliştirmek amacıyla yeni bir uyarlamalı parçacık sürüsü optimizasyonu (APSO) algoritması önerilmiştir. Önerilen algoritma, atalet ağırlık parametresini daha iyi arama yeteneği ve yakınsama hızı sağlayan genel ve yerel arama süreçleri arasındaki dengeye adapte edebilmektedir. Simülasyon temelli sonuçlar geliştirilen kontrol yaklaşımlarının dinamik performans, kararlılık ve harici bozucuların etkisi açısından performansını değerlendirmek üzere karşılaştırılmıştır.

Anahtar Kelimeler: Doğrusal Olmayan Kontrol, İHA, Geribeslemeli Kontrol, Kayan Kipli Kontrol, APSO, PID.

1
INTRODUCTİON

1.1 Literature Review

1.1.1 Unmanned Aerial Vehicles

Unmanned systems are generally known as air, maritime, and ground powered vehicles that run without a human operator. These systems are operated independently or remotely and can be recoverable or expendable. Also, it can load a collection of payloads depending on their type, functionality, and mission objectives.

Unmanned Aerial Vehicles (UAVs) is a novel technology with an enormous possibility to revolutionize military or civilian applications. The use of UAVs outside the military has grown extremely with the appearance of high power density power cells, low-power and long-range micro radio apparatuses, inexpensive airframes, and powerful microprocessors and motors. They have turned to a vital portion of future urban civil and military applications.

The aptitudes of Unmanned Aerial Vehicles (UAVs) are continuously developing, and require new technics for their control. Today's UAVs generally require several operators to control UAVs path by making the proper decisions, but future UAVs will be designed to make their own decisions independently.

1.1.2 UAVs History

The first UAV was developed after World War I in 1916 by Lawrence and Sperry. It is nicknamed the aerial torpedo, see Figure 1.1. It was able to fly over a long distance. Lawrence and Sperry invented an automatic gyroscope that enabled the UAV to fly without any need to a human interference. It was the beginning of attitude control, which is utilized for the automatic steering of the first UAV.

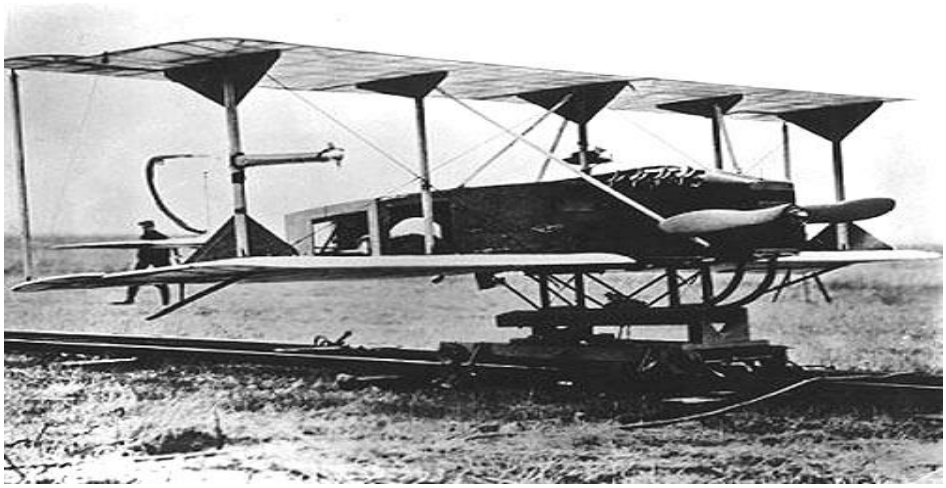

Figure 1.1 Lawrence And Sperry UAV

Shortly after, Charles F. Kettering was leading the US Army's own UAV project. Kettering wanted to improve a simpler and cheaper UAV than Lawrence and Sperry version. The first test flight did not achieve its goals well in 1918. The army then requested a large number of prototypes because the next tests were succeeded. It was known as the Kettering Bug, see Figure 1.2.

Figure 1.2 Kettering Bug UAV

Both UAVs were canceled and the funding stopped due to they have inadequate precision that is needed to serve the war exertion. The deficiency of technical advancement in the fields of steering systems, radio devices, and powerful engines was the main reason for the failure of the previous UAVs.

Etienne Oehmichen invented the first quadrotor in 1920 and was named "Oehmichen No2" [1]. It has eight propellers mounted to four rotors and operated by a single motor, see Figure 1.3. This craft implemented more than one thousand flight over a distance of '360' meters. Five propellers were used to give the lift and lateral stability because of the capability to adapt the blades angle, two were used to obtain horizontal propulsion and an extra propeller was used to direct the quadrotor.

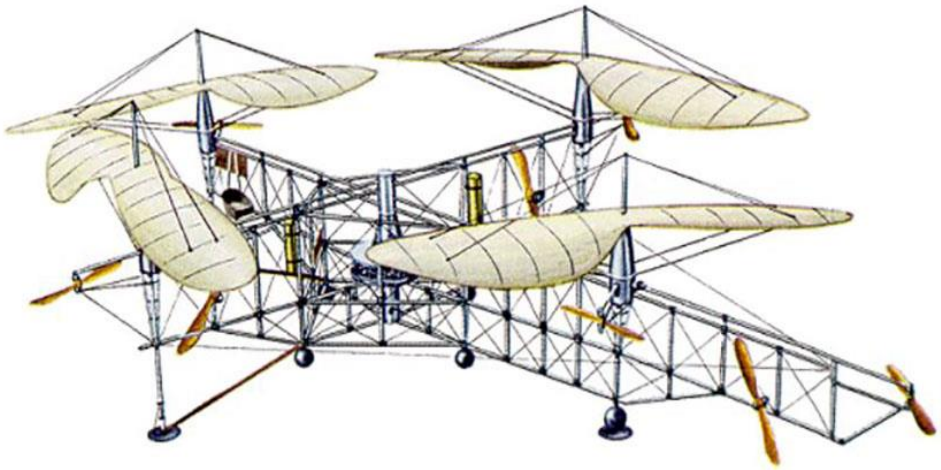

Figure 1.3 Oehmichen No2 UAV [1]

The U.S Navy started to test radio-controlled UAVs in 1930. As a result to carry on developing UAVs, the US Navy developed the Curtiss N2C-2 UAV in 1937. It has a revolutionary design at that time and was remotely controlled by other aircraft. Within the Second World War, Reginald Denny built the first model to radio-controlled UAV named the radio plane OQ-2 [2]. This UAV vastly produced in the United States in those days. These UAV drove to a new stage of UAV development.

UAVs were previously renowned to be undependable and costly aircraft, but this viewpoint altered in the 1980s. Since Israel produced the Scout and the Pioneer UAVs, It leads to change toward smaller sized and lighter UAVs. The Scout has the ability to transport a 360-degree live video of the land. These small size UAVs made them cheap to produce and difficult to overthrow. After that, the USA employed UAVs broadly in the Gulf War. The best-known model of UAV was the Predator [3], see Figure 1.4.

Figure 1.4 Predator UAV [3]

1.1.3 Applications of UAVs

UAVs are powerful and versatile industrial tools that can accomplish a vast variety of applications due to they are distinguished of a small size, low maintenance cost, capable of low risk flying for long time, high mobility, and hovering ability. UAVs uses are increasing rapidly in many civil applications like search and rescue assignments, remote sensing, goods distribution, satellite farming, security and surveillance processes, and civil infrastructure examination.

In search and rescue operations, UAVs play a pivotal role when natural or man-made catastrophes such as floods, temblors, or hurricanes occur. UAVs equipped with cameras can search for survivors, carry medical supplies to inaccessible areas, accelerate search and rescue operations and disasters management [4].

UAVs utilize sensors for collecting information about the ground objects and transfer the collected information to the ground base. Many datasets that result from UAVs remote sensing are used to help the research teams in lot of applications: crop monitoring and estimating, accurate monitoring of drought, irrigation monitoring and management, climate change monitoring, tree species, and flood mapping and monitoring.

UAVs are used to transfer different types of goods including food, packages, and other goods. They are also used to deliver medicines, blood samples, and immunizations, especially in unreachable places.

UAVs are used in precision agriculture or satellite farming for crop control, irrigation scheduling, pesticide spraying and gathering data about soil. This use leads to improve crop income, farms output, and increase gainfulness in farming methods.

UAVs has become widespread in the areas of large construction projects monitoring and management, GSM towers infrastructure inspections, power lines inspections, and gas pipelines leaks detection. UAVs let the project directors monitor and manage large construction project sites, give better vision about the project progress, and access to unreachable places in the project site. UAVs are used to reveal, check and recognize the flaws of the power line infrastructure. Besides, they can use gas detector unit that provides remote sensing to identify the places of gas leaks in gas pipelines [4].

1.1.4 Classification of UAVs

UAVs are classified as per their size and payload, range and endurance, aerodynamic configuration, and levels of autonomy.

1.1.4.1 Size And Payload Classification

UAVs are classified as per their size and payload as described below:

(a) Full-Scale UAVs: This class of UAVs has a normal size and offers the highest possible payload. These UAVs carry a pilot on board to support in case of doing complex flight tests and maneuvers.

(b) Medium-scale UAVs: They have a payload of more than ten kilograms and carry weighty and superior navigation devices on board. These class of UAVs is commonly used in security missions.

(c) Small-scale UAVs: They have a payload ranged from 2 to 10 kg and carry adequate quality navigation sensors on board. These UAVs are commonly used in disaster management thanks to their small size.

(d) Mini UAVs: They are designed for a payload under two kilograms which is enough to carry little sensors. These UAVs are used in research applications owing to their small size, low cost, and facility of maintenance.

(e) Micro UAVs: These UAVs designed to have a tiny size with a too lightweight payload. Therefore it is extremely difficult to carry sensors. The challenge is to design light-weight sensors that can be equipped with micro UAVs.

1.1.4.2 Range And Endurance Classification

UAVs are classified into numerous classes in accordance with their maximum elevation to fly and endurance [5] as described below:

(a) High-Altitude Long-Endurance (HALE): These class of UAVs can reach a maximum height of 20 km and keep flying for a period of 34 hours. Long extent intelligence, surveillance, and reconnaissance (ISR) assignments are the main use of these UAVs.

(b) Medium-Altitude Long-Endurance (MALE): These UAVs fly at an altitude from 5 to 15 km and has endurance from 5 to 24 hours. These UAVs are also utilized in intelligence, surveillance, and reconnaissance (ISR) assignments.

(c) Medium-Range or Tactical UAV (TUAV): These UAVs can fly at an altitude from 3 to 8 km and has endurance from 4 to 12 hours. This type of UAVs have smaller size and managed by more straightforward systems than the previous HALE and MALE UAVs.

(d) Small UAV: These UAVs can fly at an altitude from 300 to 1000 meters and has endurance from 2 to 3 hours. They are often applied to civil applications like power line examination, crop-spraying, and traffic observing.

(e) Mini UAV: They can fly at an altitude from 150 to 300 meters and has endurance from 1 to 2 hours.

(f) Micro UAV (MAV): These UAVs fly at low altitudes, about 250 meters. They are primarily used for spying and biological warfare.

(g) Nano UAV (NAV): They have a tiny size and very lightweight. These UAVs are often applied to explore probable threats in military applications.

1.1.4.3 Aerodynamic Configuration Classification

UAVs are classified into four essential classes according to their aerodynamic configuration [5] as follows:

(a) Fixed-wing UAVs: these classes of UAVs require an airstrip to do take-off and land operations. They have high endurance with too high fly speeds. These UAVs are basically used for reconnaissance and meteorological applications. Figure 1.5 show two example of fixed wings UAVs.

Figure 1.5 Northrop Grumman RQ-4B Global Hawk [6]

(b) Rotary-wing UAVs: These UAVs can take-off and land vertically with no need to an airstrip. They are able to fly with good controllability. These UAVs are classified into four categorizations:

(i) Single-rotor UAVs: They have a single rotor connected to a tail rotor which is used to control their heading. These UAVs are basically used in applications that request to lift heavier payloads and fly fast. A single rotor UAV is displayed in Figure 1.6(a).

(ii) Quadrotor UAVs: They have two couples of rotors and propellers that rotate in opposite directions. A quadrotor UAV is shown in Figure 1.6(b).

(iii)Multi-rotor UAVs: They consist of six or eight rotors. These UAVs provide an excellent control over position but they have limited speed and endurance. A multi rotor UAV is shown in Figure 1.6(c).

(a) Single Rotor

(b) Quadrotor

(c) Multirotor

Figure 1.6 Rotary Wings UAVs [7]

(c) Blimps UAVs: These UAVs can hold a payload around 50 tons. They look like balloons filled with helium. They also have a large endurance but can fly at low speeds, see Figure 1.7 [8].

Figure 1.7 A blimp at Ciudad Obregon Mexico [8]

(d) Flapping-wing UAVs: These UAVs are inspired by birds and flying bugs. They have small size, low payload, and low endurance. However, they are maneuverable, able to hover, and have low power consumption, see Figure 1.8.

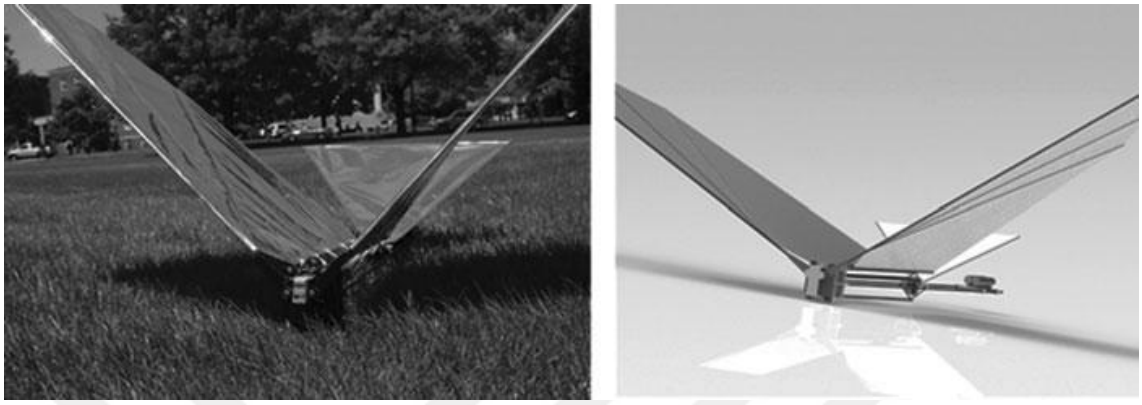

Figure 1.8 Flapping Wings UAVs [9]

1.1.4.4 Levels of Autonomy Classification

UAVs are classified as per their level of autonomy. National institute of standards and technology categorized UAVs in accordance with autonomy level [10]. The classification depends on three systems of indicators namely: human autonomy, mission complication, and environmental Complexity. The classification divided UAVs into several levels of autonomy:

(a) Level 1: These UAVs need full human interaction because they are completely remotely controlled. These UAVs are basically used for doing the simplest assignments.

(b) Level 2: These UAVs can do more complex missions than level 1 UAVs but they require a human interaction.

(c) Level 3: These UAVs require a middle level of autonomy and are used for assignments with medium intricacy.

(d) Level 4: These UAVs need a minimal human interaction with complex missions.

(e) Level 5: These UAVs do not need a human interaction and is used to do the most complex missions. This class of UAVs does not exist and they represent an objective for future researches.

1.2 Objective of the Thesis

This thesis will work on modeling and controlling quad rotor unmanned aerial vehicle. Since quad rotor is a six degrees of freedom under actuated system with nonlinearity & uncertainty properties adding to that the disturbances that affect the performance of it, many researches motivated to design linear and nonlinear controllers to stabilize and control such a system. [20].

This thesis aims to present a detailed nonlinear model of the quadrotor system taking into account aerodynamic affects, and to design and develop Linear and nonlinear controllers to stabilize and control the attitude of the quadrotor system. At the same time to assess the performance of the quad rotor system using these control algorithms by analyzing and comparing the results of the simulation applied on the presented model.

1.3 Hypothesis

Linear control methods may confirm the stability of the quadrotor but not guaranteeing that when the quad rotor performs aggressive maneuvers, and have low performance against the uncertainty and the disturbances applied or exist in the system. In the other hand the proposed nonlinear control methods give better performance of the quadrotor dealing with the nonlinearity and high aggressive maneuverability with satisfied performance against the uncertainty and the disturbances applied on the system.

1
STATE OF THE ART

Since the advancement of the Micro-Electro-Mechanical Systems (MEMS) technologies, remarkable advances in the UAVs research field were introduced. The quadrotor is the most popular unmanned aerial vehicle (UAV). Lots of researches were done on the quadrotor platform because of its simple mechanical structure, hovering ability, high maneuverability, and the ability to perform different assignments. In addition, it can vertically take-off and land (VTOL). Some literature researches concentrated on developing the related control algorithms while other researches concentrated on quadrotor models to evaluate its proposed control algorithms. This chapter introduces a discussion about the most prevalent control techniques used in quadrotor research.

2.1 Control

There are diverse control methods to control a quadrotor system which has a nonlinear model. Control methods differ from the conventional linear PD and PID controllers to more complicated nonlinear controllers like model predictive, backstepping, and sliding-mode controllers. The flight control systems are categorized into four essential groups including linear, nonlinear, learning based adaptive, and hybrid flight control systems [13]. The linear PID or PD controllers are the most popular control method to control the quadrotor system. Even though a conventional linear controller is used to control the nonlinear multivariate quadrotor system, it has been proved succeeded in many works of literature.

2.1.1 Linear Flight Control Systems

Linear flight control systems are basically predicated on PID controller and linear quadratic control. It was stated that a linear control method is used to let a full-scale aircraft accomplish autonomous navigation.

Salih et al. used the conventional PID controller to control the linearized model of the quadrotor position with the roll, pitch, and yaw angles. Research findings revealed that the PID controller was capable of control the quadrotor helicopter at low speeds [11]. Besides, Erginer and Erdinc designed four PD controllers with the purpose of stabilize the quadrotor and control the quadrotor altitude [12]. Bouabdallah et al. applied two different control methods PID and LQ to a micro quadrotor. Results turned out that both PID and LQ controllers were able to control the quadrotor's attitude in front of little perturbation [13]. Yang et al. developed an auto-tuning PID controller to ensure stabilizing for the quadrotor. The tuning process of PID parameters was implemented in accordance with the adaptive pole placement method. Results proved that the PID controller give better results after the tuning process [14].

The quadrotor’s nonlinearity may be modeled as a group of simpler linear models using the gain scheduling technique. Linear controllers like PID and LQ controllers may be utilized to control the non-linear systems using gain scheduling.

Milhim et al. utilized a gain-scheduled PID controller for controlling the fault tolerant of the quadrotor system. The proposed controller had applied in a nonlinear 6 DOF model and good tracking performance had obtained during several reference trajectories [15]. Besides, Amoozgar et al. used a fuzzy gain-scheduled PID controller for a quadrotor system in front of possible rotors defects. Results expounded the efficiency of the gain-scheduled PID controller as against the conventional PID [16].

Reyes-Valeria et al. proposed a gain-scheduled LQ control applied to the nonlinear dynamic model for quadrotor system. Results demonstrated that the LQ control using gain scheduling give better precision in the step response and trajectory tracking [17].

2.1.2 Nonlinear Flight Control Systems

The nonlinear control algorithms are necessary to be employed to flight control systems by reason of the nonlinear dynamics of the quadrotor system. Several of nonlinear control algorithms applied to the quadrotor system involving feedback

linearization, model predictive control (MPC), back-stepping control, and sliding-mode control.

Feedback Linearization: It is a control method that includes achieving a conversion of the nonlinear system into an equal linear system. Linear control approaches can be applied to stabilize the transformed linear quadrotor system. Mokhtari et al. applied a feedback linearization using a Luenberger observer to control a quadrotor system [18].

Model Predictive Control (MPC): It uses a model to do predictions about the system's future states and finds the optimal control signal using an optimization algorithm that drives the predicted output to the reference. Alexis et al. presented a switching model predictive controller which control the quadrotor's attitude in front of wind-gust disturbances. Results indicate that the considered SMPC algorithm gives high performance to control the quadrotor attitude [19].

Backstepping and Sliding-mode: Backstepping algorithm is a recursive control algorithm that may be used to control linear and nonlinear systems. Siegwart and Bouabdallah applied backstepping and sliding-mode control approaches to control a micro quadrotor called "OS4" in the presence of perturbations. Results illustrate that the suggested controller gives high performance despite the external perturbations [13]. Madani and his friend used a backstepping and sliding-mode control approaches to design controllers with control laws depends on Lyapunov stability theory for a micro quadrotor system. They separated the quadrotor model to under actuated, fully actuated, and propeller forces subsystems. Results show that their suggested controller gives good performance [20]. Fang and Gao suggested an adaptive integral backstepping control algorithm to design a robust controller. Results indicate that their suggested controller was able to solve the difficulties of exterior disorders and model uncertainties [21]. Lee et al. suggested a backstepping controller to track the desired path for a quadrotor system. Results illustrate that their proposed controller was able to control the quadrotor attitude and position in a noisy environment [22]. Zhen et al. proposed a backstepping controller tuned by an adaptive algorithm to stabilize the quadrotor system. A robust adaptive algorithm is applied to stop the effect of exterior disorders. Results

demonstrated that the proposed controller gives a good performance in front of exterior disorders and model uncertainties [23].

2.1.3 Learning-based Adaptive Flight Control Systems

These systems can deal with changes and uncertainty of the controlled system. There are several approaches of learning-based adaptive flight control systems such as neural networks (NN), fuzzy logic control (FLC), linear time invariant (LTI), model reference adaptive control (MRAC), reinforcement learning control (RLC), and adaptive network-based fuzzy inference system (ANFIS).

Neural networks: Efe proposed a neural network aided FIR type controller to simplify the design of a PID controller and relieve disturbances resulting from the model mismatch, wind disturbances, and measurement noise [26].

Fuzzy Logic: It translates the experience and behaviors of skillful human beings to a group of orders that can be utilized by a machine to perform a particular assignment usually carried out by humans. Santos et al. suggested a fuzzy logic controller to stabilize the quadrotor. Results demonstrate the efficiency and robustness of the fuzzy logic controller in front of noise and uncertainty from the sensors [27].

2.1.4 Hybrid Flight Control Systems

Hybrid flight control systems depend on using two or more control methods. It is aspired to strengthen the performance of the determined system. In recent studies, it was ascertained that one flight control algorithm was not able to ensure acceptable performance, particularly when the quadrotor fly far from its nominal conditions. Wang and Azzam proposed a PD controller for controlling the altitude and yaw angle and a PID controller combined with a backstepping controller for controlling the roll and pitch angles. They used an optimization algorithm to tune the PD and PID parameters. Results displayed the efficiency of the suggested hybrid control system [24]. Nagaty et al. [25] proposed nested loops control architecture to stabilize the quadrotor. The outer loop contains a PD controller that produces the desired trajectories for the inner loop controller. While the inner loop uses a nonlinear backstepping controller so as to track the desirable attitude.

2
System Modeling

The quadrotor's kinematics and dynamics models are introduced on the basis of Newton-Euler equations and by considering the following assumptions:

- The quadrotor has a symmetrical and rigid structure.
- The quadrotor's center of mass corresponds with the body-fixed frame origin.

The quadrotor's movement is divided into translational and rotational equations that describe the dynamics model. Besides, the aerodynamic effects that may act on the quadrotor body are debated with the used motors dynamics of the quadrotor. Finally, a quadrotor state space model is presented depending on the rotational and translational equations.

3.1 Kinematic Model

The inertial and body coordinate frames are debated in order to attain the kinematic model of the quadrotor system. The quadrotor frames are the inertial reference frame which is set at a determined place on the ground and represented by (N, E, D) axes and the body frame which is set at the center of mas of the quadrotor and represented by (X, Y, Z) axes, see Figure 3.1. The used codes N, E, and D refer to the North, East, and Downwards directions while the x-axis indicates to the first rotor, the y-axis indicates to the second rotor, and the z-axis indicates to the ground.

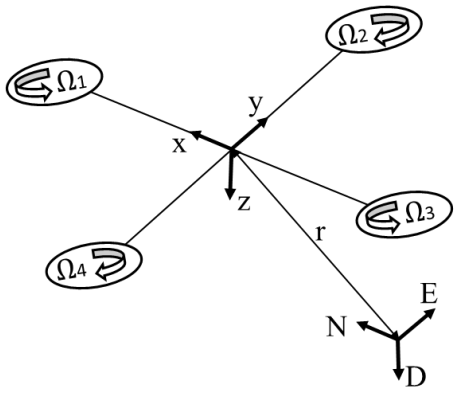

Figure 3.1 Inertial and Body Frames Of the Quadrotor

The current position of the quadrotor is calculated as the distance between the inertial and body frame. The rotation matrix R is used to move coordinates between the body frame and the inertial reference frame. It depicts the directing of the quadrotor and uses the rotations angles around the X, Y, and Z axes. The rotation matrix R is described as following:

$$R = \begin{bmatrix} C\psi C\theta & C\psi S\phi S\theta - S\psi C\phi & S\psi S\phi + C\psi C\phi S\theta \\ S\psi C\theta & C\phi C\psi + S\phi S\theta S\psi & S\psi C\phi S\theta - C\psi S\phi \\ -S\theta & S\phi C\theta & C\phi C\theta \end{bmatrix} \tag{3.1}$$

Where $s_x = \sin(x)$, *and* $c_x = \cos(x)$, It has remarkable importance to formulate the dynamics model of the quadrotor because some variables are calculated in the body frame like the thrust forces resulting from the propellers, while other variables are calculated in the inertial frame like the quadrotor's position. So, it is necessary to have a way to connect between the two frames.

The transformation matrix R_τ for angular velocities between the angular body rates ω and Euler rates $\dot{\eta}$ is described as follows.

$$\omega = R_\tau \dot{\eta}$$
$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & -s\theta \\ 0 & c\phi & s\phi c\theta \\ 0 & -s\phi & c\phi c\theta \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{3.2}$$

3.2 Dynamics Model

The quadrotor movements are divided into rotational and translational subsystems [25]. The rotational subsystem is accountable for determining the roll, pitch, and yaw angles while the translational subsystem is accountable for determining the positions x, y, and z.

3.2.1 Rotational Equations of Motion

In the body frame, the rotational movement equations are derived using the Newton-Euler equations and had the general formula:

$$J\,\dot{\omega} + \omega \text{ x } J\,\omega + M_G = M_B - M_a \tag{3.3}$$

The quadrotor inertia matrix J is a diagonal matrix and has the form:

$$J = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{3.4}$$

Where the parameters I_{xx}, I_{yy}, I_{zz} are the inertia moments around the primary axes in the body frame.

The gyroscopic moments M_G that generated from the rotor's inertia J_r have the form.

$$M_G = \omega \text{ x} \begin{bmatrix} 0 & 0 & J_r & \omega_r \end{bmatrix}^T \tag{3.5}$$

Where J_r and Ω_r are the inertia and relative speeds for rotors $\Omega_r = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4$.

The drag moments M_a that act on the quadrotor body because of the air friction have the form.

$$M_a = K_r\,\dot{\eta} \tag{3.6}$$

While K_r is the constant aerodynamic rotation coefficients matrix.

The moments that affect the quadrotor in the body frame M_B are produced from the effects of the aerodynamic forces and the resulted moments from rotors. The aerodynamic forces F_i and the moments M_i generated from i^{th} rotor are described in equation (3.7) and (3.8).

$$F_i = \frac{1}{2}\rho A C_T r_b^{\,2} \Omega_i^2 \tag{3.7}$$

$$M_i = \frac{1}{2}\rho A C_D r_b^{\,2} \Omega_i^2 \tag{3.8}$$

Where the air density ρ, the blade area A, the aerodynamic coefficients (C_D, C_T), and the blade radius r_b are constants, while Ω_i is the i^{th} rotor angular velocity. Equations (3.7) and (3.8) are modified to the following form:

$$F_i = K_f \, \Omega_i^2 \tag{3.9}$$

$$M_i = K_M \, \Omega_i^2 \tag{3.10}$$

Both the aerodynamic force K_f and moment constant K_m are defined empirically for each kind of propellers. Figure 3.2 shows the moments and forces that affect the quadrotor.

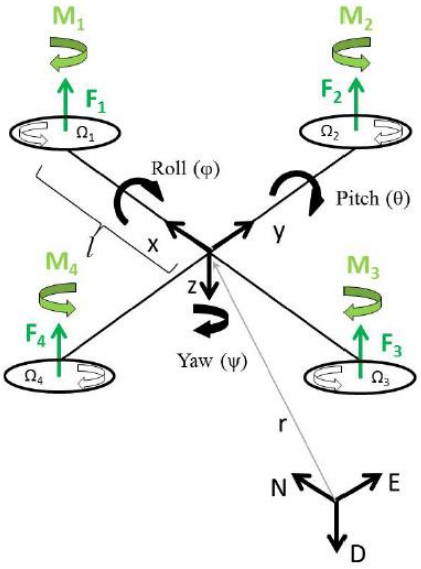

Figure 3.2 Moments and Forces Affecting On Quadrotor

The rotation of the rotors produces an upwards thrust forces F_i and generates moments M_i. The total moments that affect the quadrotor about the x-axis has the following form.

$$
\begin{aligned}
M_x &= l F_4 - l F_2 \\
&= l\left(K_f \Omega_4^2\right) - l\left(K_f \Omega_2^2\right) \\
&= l K_f \left(\Omega_4^2 - \Omega_2^2\right)
\end{aligned}
\tag{3.11}
$$

The total moments that affect the quadrotor about the y-axis has the following form.

$$
\begin{aligned}
M_y &= -F_1 l + F_3 l \\
&= -\left(K_f \Omega_1^2\right) l + \left(K_f \Omega_3^2\right) l \\
&= l K_f \left(-\Omega_1^2 + \Omega_3^2\right)
\end{aligned}
\tag{3.12}
$$

While the total moments that affect the quadrotor about the z-axis has the following form.

$$
\begin{aligned}
M_z &= M_1 - M_2 + M_3 - M_4 \\
&= K_M \Omega_1^2 - K_M \Omega_2^2 + K_M \Omega_3^2 - K_M \Omega_4^2 \\
&= K_M \left(\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2\right)
\end{aligned}
\tag{3.13}
$$

The total moments that affect the quadrotor results by combining the equations (3.11), (3.12), and (3.13) into a vector.

$$
M_B = \begin{bmatrix} l K_f \left(-\Omega_2^2 + \Omega_4^2\right) \\ l K_f \left(-\Omega_1^2 + \Omega_3^2\right) \\ K_M \left(\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2\right) \end{bmatrix}
\tag{3.14}
$$

Where *l,* refers to the distance between the rotation axes of each rotor with the quadrotor center of mass.

3.2.2 Translational Equations of Motion

In the inertial frame, the translation movement's equations for the quadrotor are derived on the basis of Newton's second law.

$$
m\,\ddot{r} = \begin{bmatrix} 0 \\ 0 \\ m g \end{bmatrix} + R F_B - F_a
\tag{3.15}
$$

Where $r=[x\ y\ z]^T$ is the Quadrotor's distance from the inertial frame, m is the quadrotor mass, g is the gravitational acceleration, F_B is the non-gravitational forces that affect the quadrotor in the body frame, and Fa is the drag forces resulting from the air friction with quadrotor body. The non-gravitational forces affect the quadrotor only in the horizontal orientation and have the form.

$$F_B = \begin{bmatrix} 0 \\ 0 \\ -K_f(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \end{bmatrix} \tag{3.16}$$

The drag forces that affect the quadrotor body because of the friction with air have the form [28].

$$F_a = K_t\ \dot{r} \tag{3.17}$$

Where K_t is the aerodynamic translation coefficients matrix.

3.3 Rotor Dynamics

The most used motors in the quadrotors are brushless DC motors owing to their high efficiency, high torque to weight ratio, reduced friction, and longer lifetime. In this study, it is presumed that the rotors are non-geared and the mechanical connection between the propellers and motors are rigid. The schematic diagram for the DC motor at a steady state, see Figure 3.3.

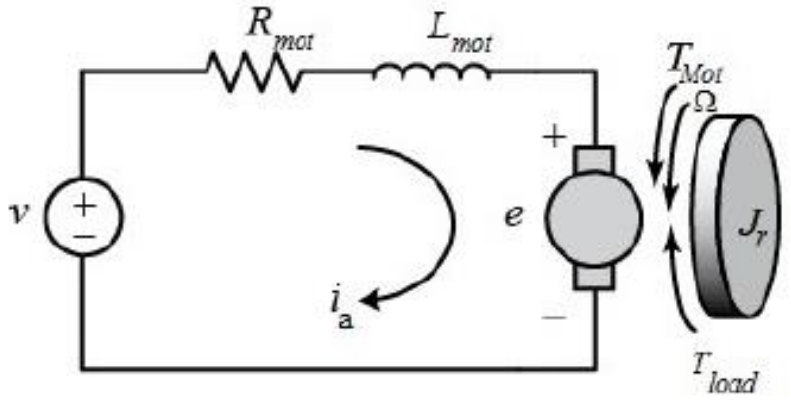

Figure 3.1 Schematic Diagram of DC Motor

The equation that define the electrical part of the motor are attained through Kirchhoff's voltage law.

$$v = R_{mot}\, i_a + L_{mot}\, \frac{di_a}{dt} + K_{mot}\, \Omega \tag{3.18}$$

Where R_{mot} is the armature resistance, L_{mot} is the armature inductance, v is the input voltage, i_a is the armature current, K_{mot} represents the *emf* constant which is equal to the torque motor constant. And since the motors of the quadrotor are normally small, their inductance is very small and that is leading to

$$v = R_{mot}\, i_a + K_{mot}\, \Omega \tag{3.19}$$

$$i_a = \frac{v - K_{mot}\, \Omega}{R_{mot}} \tag{3.20}$$

The motor equations that describe the mechanical part can be derived through Newton's law.

$$J_r\, \dot{\Omega} = T_{mot} - T_{load} \tag{3.21}$$

Where T_{mot} is the motor torque, T_{load} is the torque generated from the load or the propellers. The previous equation can be rewritten as following:

$$J_r\, \dot{\Omega}_i = K_{mot}\, i_a - K_r\, \Omega_i^2 - K_s \tag{3.22}$$

And this equation could be written:

$$J_r\, \dot{\Omega}_i = K_{mot} \left(\frac{v - K_{mot}\, \Omega_i}{R_{mot}} \right) - K_r\, \Omega_i^2 - K_s \tag{3.23}$$

After doing some modifications, the equation is written to describe the dynamic model for the used dc motors.

$$\dot{\Omega}_i = -r_0 - r_1 \Omega_i - r_2\, \Omega_i^2 + r_3\, u \tag{3.24}$$

Where u is the voltage input of the motor

$$r_0 = \frac{K_s}{J_r}, r_1 = \frac{K_{mot}^2}{R_{mot} * J_r} \qquad , r_2 = \frac{K_r}{J_r}, r_3 = \frac{K_{mot}}{R_{mot} * J_r}$$

Table 3.1 Quadrotor Constants and Parameters

Parameters	Discerption	Value	Unit
I_{xx}	MOI about body frame's x-axis	7.5e-3	$kg.m^2$
I_{yy}	MOI about body frame's y-axis	7.5e-3	$kg.m^2$
I_{zz}	MOI about body frame's z-axis	1.3e-2	$kg.m^2$
l	Moment arm	0.23	m
J_r	Rotor inertia	6e-5	$kg.m^2$
m	Quadrotor mass	0.650	kg
K_f	Aerodynamic force constant	3.13e-5	Ns^2
K_m	Aerodynamic moment constant	7.5e-7	Nms^2
R_{mot}	Motor circuit resistance	0.6	Ω
K_{mot}	Motor torque constant	5.2	$N.m/A$

3.4 State Space Model

The quadrotor state space model is introduced to facilitate the control ability. Let us suppose the state space vector to define the quadrotor's positions and velocities.

$$X = \begin{bmatrix} \phi & \dot{\phi} & \theta & \dot{\theta} & \psi & \dot{\psi} & z & \dot{z} & x & \dot{x} & y & \dot{y} \end{bmatrix}^T \tag{3.25}$$

Let consider the control input vector, composing from 4 signals, to control the altitude and the roll, pitch, and yaw angles.

$$U = \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} K_f & K_f & K_f & K_f \\ 0 & -K_f & 0 & K_f \\ K_f & 0 & K_f & 0 \\ K_M & -K_M & K_M & -K_M \end{bmatrix} \begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} \tag{3.26}$$

The control signal U_1 represents the upwards forces produced from the four rotors and control the quadrotor altitude (z). While the control signal U_2 represents the difference in thrust resulted from the second and fourth rotors and control the roll

angle ϕ that denotes the rotation around the x-axis. At the same time, the control signal U_3 represents the difference in thrust resulted from the first and third rotors and control the pitch angle θ that denotes the rotation around the y-axis. The last control signal U_4 represents the difference in torques resulted from the rotating rotors and control the yaw angle ψ that denotes the rotation around the z-axis.

The rotor velocities can be extracted as a function of the input signals by inverting the previous matrix mentioned in equation (3.26):

$$\begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} = \begin{bmatrix} \frac{1}{4K_f} & 0 & \frac{1}{2K_f} & \frac{1}{4K_M} \\ \frac{1}{4K_f} & -\frac{1}{2K_f} & 0 & -\frac{1}{4K_M} \\ \frac{1}{4K_f} & 0 & -\frac{1}{2K_f} & \frac{1}{4K_M} \\ \frac{1}{4K_f} & \frac{1}{2K_f} & 0 & -\frac{1}{4K_M} \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} \tag{3.27}$$

And the rotor's velocities take the form:

$$\begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} = \begin{bmatrix} \frac{1}{4K_f} & 0 & \frac{1}{2K_f} & \frac{1}{4K_M} \\ \frac{1}{4K_f} & -\frac{1}{2K_f} & 0 & -\frac{1}{4K_M} \\ \frac{1}{4K_f} & 0 & -\frac{1}{2K_f} & \frac{1}{4K_M} \\ \frac{1}{4K_f} & \frac{1}{2K_f} & 0 & -\frac{1}{4K_M} \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} \tag{3.28}$$

And the rotor's velocities take the final form:

$$\Omega_1 = \sqrt{\frac{U_1}{4K_f} + \frac{U_3}{2K_f} + \frac{U_4}{4K_M}} \qquad \Omega_3 = \sqrt{\frac{U_1}{4K_f} - \frac{U_3}{2K_f} + \frac{U_4}{4K_M}}$$

$$\Omega_2 = \sqrt{\frac{U_1}{4K_f} - \frac{U_2}{2K_f} - \frac{U_4}{4K_M}} \qquad \Omega_4 = \sqrt{\frac{U_1}{4K_f} + \frac{U_2}{2K_f} - \frac{U_4}{4K_M}} \tag{3.29}$$

3.4.1 Rotational Equations

If the control signals in equation (3.26) replaced in equation (3.14), the total moments that effect on the quadrotor would become as following:

$$M_B = \begin{bmatrix} U_2 l \\ U_3 l \\ U_4 \end{bmatrix} \tag{3.30}$$

The rotational equation of motion described in equation (3.3) can be extended to the next form:

$$\begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} + \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \mathrm{x} \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \mathrm{x} \begin{bmatrix} 0 \\ 0 \\ J_r \Omega_r \end{bmatrix} = \begin{bmatrix} U_2 l \\ U_3 l \\ U_4 \end{bmatrix} \tag{3.31}$$

And the final form:

$$\begin{bmatrix} I_{xx} \ddot{\phi} \\ I_{yy} \ddot{\theta} \\ I_{zz} \ddot{\psi} \end{bmatrix} + \begin{bmatrix} \dot{\theta}\dot{\psi}(I_{zz} - I_{yy}) \\ \dot{\phi}\dot{\psi}(I_{xx} - I_{zz}) \\ \dot{\theta}\dot{\phi}(I_{yy} - I_{xx}) \end{bmatrix} + \begin{bmatrix} \dot{\theta} J_r \Omega_r \\ -\dot{\phi} J_r \Omega_r \\ 0 \end{bmatrix} = \begin{bmatrix} l U_2 \\ l U_3 \\ U_4 \end{bmatrix} \tag{3.32}$$

The angular accelerations can be extracted as follows:

$$\ddot{\phi} = \frac{l}{I_{xx}} U_2 - \frac{J_r}{I_{xx}} \dot{\theta} \Omega_r + \frac{I_{yy}}{I_{xx}} \dot{\psi}\dot{\theta} - \frac{I_{zz}}{I_{xx}} \dot{\theta}\dot{\psi} \tag{3.33}$$

$$\ddot{\theta} = \frac{l}{I_{yy}} U_3 - \frac{J_r}{I_{yy}} \dot{\phi} \Omega_r + \frac{I_{zz}}{I_{yy}} \dot{\phi}\dot{\psi} - \frac{I_{xx}}{I_{yy}} \dot{\psi}\dot{\phi} \tag{3.34}$$

$$\ddot{\psi} = \frac{1}{I_{zz}} U_4 + \frac{I_{xx}}{I_{zz}} \dot{\theta}\dot{\phi} - \frac{I_{yy}}{I_{zz}} \dot{\phi}\dot{\theta} \tag{3.35}$$

3.4.2 Translational Equation

If the control signals in equation (3.26) replaced in equation (3.16), the total moments that effect on the quadrotor would become as following:

$$F_B = \begin{bmatrix} 0 \\ 0 \\ -U_1 \end{bmatrix} \tag{3.36}$$

The translational movement equation described in equation (3.15) can be extended to the next form:

$$m\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} + \begin{bmatrix} c\psi c\theta & s\phi s\theta c\psi & s\psi s\phi + c\phi c\psi s\theta \\ c\theta s\psi & s\psi c\theta + s\phi s\psi s\theta & c\phi s\psi s\theta - c\psi s\theta \\ -s\theta & c\theta s\phi & c\phi c\theta \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ -U_1 \end{bmatrix}$$

$$m\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} + \begin{bmatrix} U_1(-s\phi s\psi - c\phi c\psi s\theta) \\ U_1(c\psi s\theta - s\psi s\theta c\phi) \\ U_1\left(-c\theta c\phi\right) \end{bmatrix} \tag{3.37}$$

The accelerations as a function of other parameters can be extracted as follows:

$$\ddot{x} = -\frac{U_1}{m}(\sin(\theta)\cos(\psi)\cos(\phi) + \sin(\psi)\sin(\phi)) \tag{3.38}$$

$$\ddot{y} = \frac{U_1}{m}(\sin(\theta)\cos(\psi) - \cos(\phi)\sin(\theta)\sin(\psi)) \tag{3.39}$$

$$\ddot{z} = +g - \frac{U_1}{m}\left(\cos(\phi)\cos(\theta)\right) \tag{3.40}$$

The accelerations are rewritten as a function of the state variables:

$$\ddot{x} = -\frac{U_1}{m}(\sin(x_3)\cos(x_5)\cos(x_1) + \sin(x_5)\sin(x_1))$$

(3.41)

$$\ddot{y} = \frac{U_1}{m}(\sin(x_1)\cos(x_5) - \cos(x_1)\sin(x_3)\sin(x_5)) \tag{3.42}$$

$$\ddot{z} = +g - \frac{U_1}{m}\left(\cos(x_3)\cos(x_1)\right) \tag{3.43}$$

3.4.3 State Space Representation

A state space representation of the quadrotor mathematical model is described depending on the rotational and translational equations:

$$
\begin{aligned}
\dot{x}_1 &= \dot{\phi} = x_2 \\
\dot{x}_2 &= \ddot{\phi} = b_1 U_2 + x_6 x_4 a_1 - x_4 \Omega_r a_2 \\
\dot{x}_3 &= \dot{\theta} = x_4 \\
\dot{x}_4 &= \ddot{\theta} = x_2 x_6 a_1 - x_2 \Omega_r a_4 + b_2 U_3 \\
\dot{x}_5 &= \dot{\psi} = x_6 \\
\dot{x}_6 &= \ddot{\psi} = x_2 x_4 a_5 + b_3 U_4 \\
\dot{x}_7 &= \dot{z} = x_8 \\
\dot{x}_8 &= \ddot{z} = +g - \frac{U_1}{m}(\cos(\mathrm{x}_3)\cos(\mathrm{x}_1)) \\
\dot{x}_9 &= \dot{x} = x_{10} \\
\dot{x}_{10} &= \ddot{x} = -\frac{U_1}{m}(\cos(\mathrm{x}_1)\sin(\mathrm{x}_3)\cos(\mathrm{x}_5) + \sin(\mathrm{x}_1)\sin(\mathrm{x}_5)) \\
\dot{x}_{11} &= \dot{y} = x_{12} \\
\dot{x}_{12} &= \ddot{y} = \frac{U_1}{m}(\cos(\mathrm{x}_1)\sin(\mathrm{x}_3)\sin(\mathrm{x}_5) + \sin(\mathrm{x}_1)\cos(\mathrm{x}_5))
\end{aligned}
\tag{3.44}
$$

Where

$$
\begin{aligned}
a_1 &= \frac{I_{yy} - I_{zz}}{I_{xx}} & b_1 &= \frac{l}{I_{xx}} \\
a_2 &= \frac{J_r}{I_{xx}} & b_2 &= \frac{l}{I_{yy}} \\
a_3 &= \frac{I_{zz} - I_{xx}}{I_{yy}} & b_3 &= \frac{l}{I_{zz}} \\
a_4 &= \frac{J_r}{I_{yy}} & & \\
a_5 &= \frac{I_{xx} - I_{yy}}{I_{zz}} & &
\end{aligned}
\tag{3.45}
$$

3
SYSTEM CONTROL

The resulted quadrotor model is employed in controller's development. Three control methods are designed to control the quadrotor which are a self-tuning PID control, sliding mode control, and backstepping control. Adaptive PSO algorithm is applied to tune the variables and gains of these controllers. Simulations are carried out using MATLAB/Simulink environment and employed to assess the designed controllers' performance.

4.1 Particle Swarm Optimization

4.1.1 Introduction to Particle Swarm Optimization

Particle swarm optimization algorithm (PSO) is an evolutional and computational strategy that searches for optimal solutions to a problem by continuously improving solutions inspired by the birds flocking and schooling of fish. Eberhart and Kennedy proposed the basic idea of the PSO algorithm [29]. Particle swarm optimization algorithm is computationally efficient, easy to accomplish and does not require much memory compared to other evolutionary algorithms. Besides, PSO provides better performance than other search algorithms due to it uses fewer parameters, can reach an optimal solution region with rapid convergence speed and has a very effective global search algorithm. PSO algorithm is applied successfully in many optimization problems like tuning PID parameters, training of artificial neural networks, optimization for functions, fuzzy logic systems, scheduling, etc.

PSO is actually one of the swarm intelligence algorithms. It depends on the participation of social information among particles in the swarm. Every single particle in the swarm refers to a possible solution to the problem being optimized. Particles benefit from their past experiences and the best particle experience throughout the search process.

PSO algorithm begins with a set of random particles distributed over the search area. It searches for the optimal solution by updating its particles to track the best particle in every iteration. Every single particle in the swarm denotes to a possible solution, so the more particles used in the swarm, the more possible solutions produced in the PSO algorithm. Each particle keeps its current and previous positions in its memory to determine its best position during several iterations. The best position of the same particle is named pbest whereas the best position among particles is named gbest. If the algorithm stopping criteria has achieved, the gbest particle in PSO algorithm will be selected as the optimal solution.

4.1.2 Particle Swarm Optimization Algorithm

The PSO algorithm begins with a set of randomly generated particles then seeks for the best particle each iteration. In each iteration in the PSO algorithm, the particles move toward the best particle in the search area at different speeds. Every single particle is assessed in accordance with its fitness function. The fitness function is an evaluation function used to determine how good the solution is. Each particle in the search area is updated in accordance with its best position (pbest) and the best position (gbest) of all particles. The particles speed and position are updated depending on pbest and gbest values according to the following equations.

$$V_i^{(k+1)} = w * V_i^{(k)} + c_1 * r_1 * (x_{i,best}^{(k)} - x_i^{(k)}) + c_2 * r_2 * (x_{gbest}^{(k)} - x_i^{(k)}) \qquad (4.1)$$

$$x_i^{(k+1)} = x_i^{(k)} + V_i^{(k+1)} \qquad (4.2)$$

The acceleration coefficients c_1 and c_2 described in equation (4.1) are generally chosen as $c_1 = c_2 = 2$. The coefficients r_1 and r_2 are set as random numbers vary from 0 to 1 and renewed in every iteration. The variable w represent the inertia weight generally ranges from 0.1 to 1. The pbest and gbest values can be expressed as follows.

$$Pbest_i^{(k)} = x_{i,best}^{(k)} = \begin{cases} x_{i,best}^{(k-1)} & if \quad f(x_i^{(k)}) \geq f(x_{i,best}^{(k-1)}) \\ x_i^{(k)} & if \quad f(x_i^{(k)}) < f(x_{i,best}^{(k-1)}) \end{cases} \qquad (4.3)$$

$$gbest^{(k)} = x_{gbest}^{(k)} = \min\left\{ f(x_{1,best}^{(k)}), f(x_{2,best}^{(k)}), ..., f(x_{n,best}^{(k)}) \right\} \qquad (4.4)$$

The particles in PSO algorithm change their positions in the search area in each iteration. The motion mechanism of particles in the search area is shown in figure 4.1.

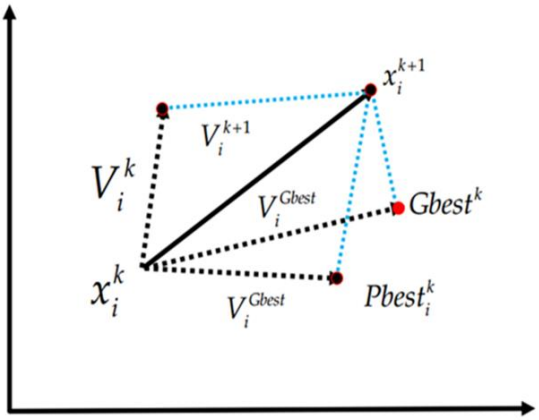

Figure 4.1 The Motion Mechanism Of Particles In The Search Area

Here; $x_i^{(k)}$ is the particle's position, $x_i^{(k+1)}$ is the particle's position at next iteration, $V_i^{(k)}$ is the particle's velocity, $V_i^{(k+1)}$ is the particle's velocity at next iteration, $V_i^{(pbest)}$ is the pbest particle velocity, and $V_i^{(Gbest)}$ is the gbest particle velocity.

The particle swarm optimization algorithm may be split into the following steps.

Step 1: The initialized positions (x_i) of the particles in the first iteration are defined randomly in the search area.

Step 2: The fitness values of all particles in the swarm are calculated each iteration in order to evaluate the particles.

Step 3: If the particle has a fitness value better than the fitness value of the pbest (best local position of a particle through several iterations), the present particle position is updated as the new local position pbest.

Step 4: The best particle among all local best particles is defined as the new global best particle (gbest) that have the best fitness value at all.

Step 5: The positions and velocities of the particles are renewed depending on the pbest and gbest values.

Step 6: The algorithm is repeated from the third step until the stop criterion is achieved.

The Pseudo code that describes the PSO algorithm is given below:

For every particle

Specify a random position

End

Do

For every particle

Compute the fitness value

If the particle fitness value is lower than the pbest particle fitness value;

Specify this particle as a new pbest particle

End

Specify the best particle among pbest particles as a gbest particle

For every particle

Compute the particle's velocity as stated in the equation (4.1)

Compute the particle's position as stated in the equation (4.2)

End

While the stopping condition is not reached

In light of these statements, the basic flow chart of the PSO algorithm is presented in Figure 4.2.

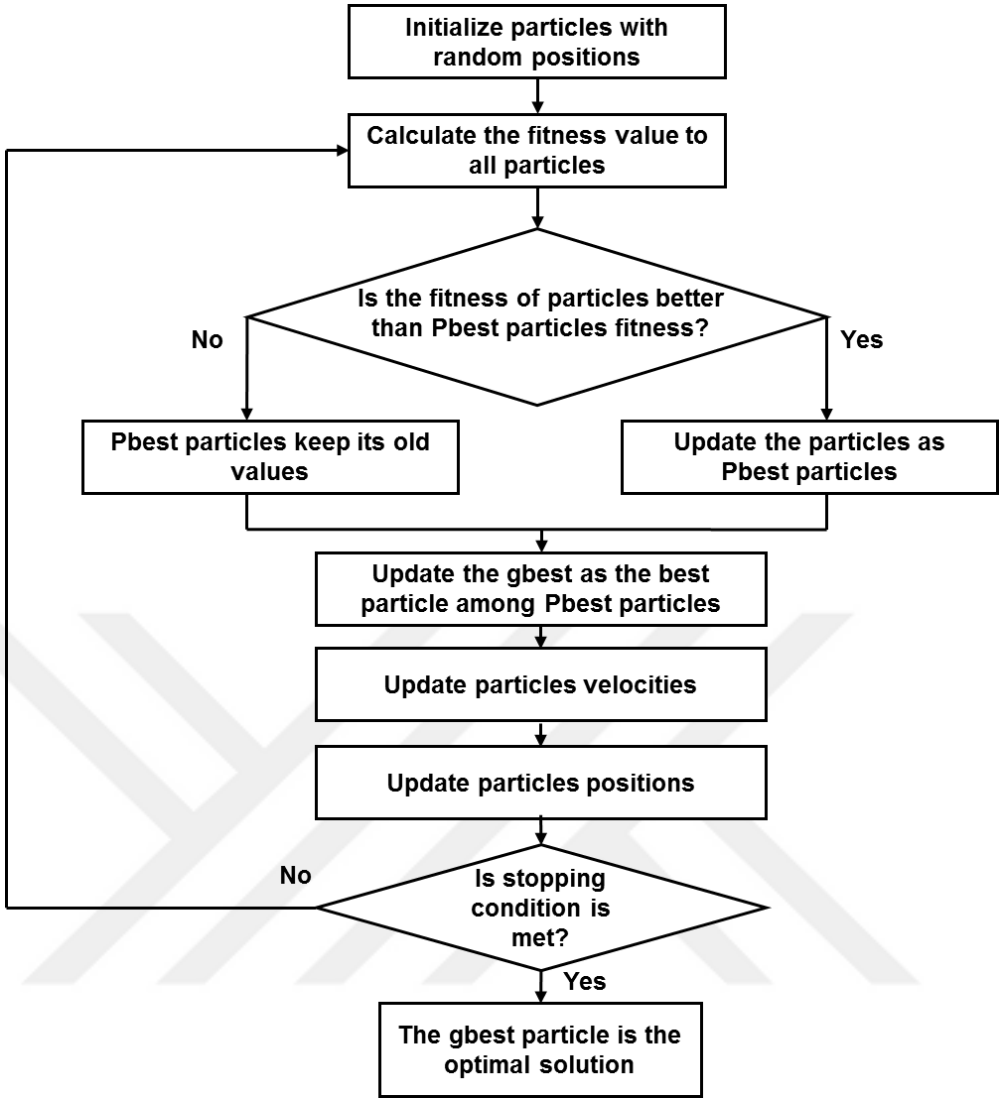

Figure 4.2 PSO Algorithm Flow Chart

The fitness function is applied for assessing the particles in every iteration in PSO algorithm. As a rule, the fitness function demonstrates how good the particle is. The stopping criterion may be a determined number of iterations or a defined fitness value of the gbest particle. The gbest position produced at the final iteration represents the best solution to solve the optimized problem.

4.1.3. The Main Parameters Used In Particle Swarm Optimization

PSO consists of many control parameters such as problem size, size of the swarm, inertia weight, acceleration coefficients, and random values that determine cognitive and social contributions.

The swarm size: the swarm size represent the particles number existed in the swarm. It determines the number of available solutions in each iteration. Moreover, a large number of particles causes an increment in the computation time in each iteration but makes it possible to achieve an optimal solution in less number of iterations. The most common swarm size usually varies between 20 and 40 particles. In fact, in many problems, 10 particles are adequate to obtain acceptable solutions. For some specific problems, more than 40 particles may be required.

The number of iterations: it can be selected depending on the problem being optimized. The low number of iterations in the PSO algorithm may not be sufficient to give a good solution. Although the high number of iterations increases the calculation difficulty, it increases the possibility to reach a better optimal solution.

Acceleration coefficients c_1 and c_2: These coefficients affect the cognitive and social components of particles. The cognitive component affects the local search process and lets the particles to move depending on their local best positions. While the social component affects the global search process and lets the particles to move depending on the global best position. If the acceleration coefficients are set to zero, the particles continue moving at their current speed in the search area. The low values of acceleration coefficients let the particles to move slowly toward the target zone. However, selecting the acceleration coefficients at high values may cause unexpected movements and exceeding the target area. It has been reported that selecting the acceleration coefficient as $c_1 = c_2 = 2$ gives the best results after some trials on the PSO algorithm performed by researchers [30].

Inertia weight w: PSO algorithm uses inertia weight to retain the balance during the global and local search processes. Selecting inertia weight at high value improve the global search abilities but low inertia weight improve the local search abilities. Therefore, inertia weight is responsible for keeping a balance during local and global search processes which cause to reach better solutions in a lower number of iterations.

The inertia weight is the parameter that controls the particles speed depending on the effect of the old speed information. It simply determines how the particles old

speed information affects the new speed. If the inertia weight is more than one, the particles velocity will accelerate gradually and the swarm particles will diverge. If the inertia weight is less than zero, the particles velocity will slow down until reaching zero.

4.1.4. Adaptive Particle Swarm Optimization Algorithm

Adaptive PSO has better characteristics than classical PSO. The APSO can perform a global and local search processes with fast convergence speed. The APSO can do automatic control of its parameters in particular inertia weight and acceleration coefficients to get better search efficiency and convergence speed. Several strategies to adjust inertia weight were proposed since the beginning of the PSO algorithm [31-32-33].

4.1.4.1 The Suggested Adaptive Particle Swarm Optimization Algorithm

In this thesis, a new strategy suggested to adjust inertia weight depending on a comparison among the fitness values of the particles. Many researchers recommend that inertia weight must be large in the global search process and small in the local search process. The correct adaptation of the inertia weight increases the efficiency of the PSO algorithm, causes to reach better solutions in lower number of iterations.

We suggest an adaptive PSO algorithm to adjust the inertia weight (w) adaptively by this new equation:

$$\alpha_{1,i}^{K} = \frac{F_{pbest,i}^{K}}{2F_{p,i}^{K}}$$
$$\alpha_{2,i}^{K} = \frac{F_{gbest}^{K}}{2F_{pbest,i}^{K}} \qquad (4.5)$$
$$w_{i}^{K} = w_{\max} - (w_{\max} - w_{\min}).\left(\alpha_{1,i}^{K} + \alpha_{2,i}^{K}\right)$$

The inertia weight (w) values range from the upper limit 'w_{max}=0.9' to the lower limits 'w_{min}=0.3'. Every particle in the swarm has a different inertia weight value in the same iteration. The values of (α_1, α_2) determine the evolutionary states of the particles in the swarm. If a particle has an unsuccessful solution ($F_{p,i} >> F_{pbest,i} >> F_{gbest}$), then the values of ($\alpha_1$, α_2) will be too close to zero and lead to

large inertia weight in order to improve the global search process. If a particle has a successful or a good solution ($F_{p,i} \approx F_{pbest,i} \approx F_{gbest}$), then the values of ($\alpha_1$, α_2) will be close or equal to (0.5) and lead to low inertia weight in order to improve the local search process.

4.2 PID Controller

PID controller is the most widespread strategy to control the quadcopter system [34-35]. The PID controller is a type of feedback controllers which is used to stabilize the quadcopter system. The general control law of the PID controller is described as follows:

$$u = K_p\, e(t) + K_i \int_0^t e(t)dt + K_d \frac{de(t)}{dt} \tag{4.6}$$

The parameters K_p, K_i, and K_d indicate to the proportional, integral, and derivative gains respectively. The control signal aims to diminish the error signal (e) between the reference value and the measured output.

The tuning process of the PID parameters determines the controller performance. In this thesis, a new APSO algorithm is recommended to tune the PID parameters in order to find its optimal values. Besides, the tuning process is also done by standard particle swarm optimization and trial and error method.

Four PID controllers were developed to control altitude (z), attitude (Φ, θ), and heading (ψ) of the quadrotor. Below the general structure of the PID controller as a block diagram is displayed in Figure 4.3.

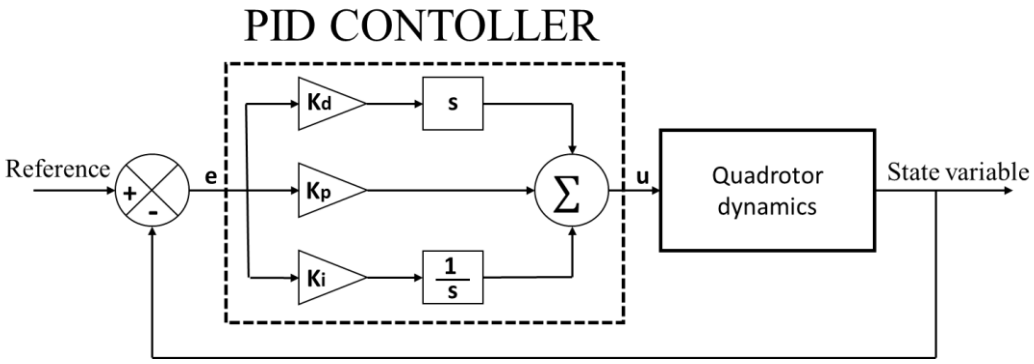

Figure 4.3 The Block Diagram Of The PID Controller

4.2.1 Altitude Control

A PID controller is designed for controlling the quadrotor altitude (z). It produces the proper control signal U_1 to minimize the difference between the reference and real altitude. The following equation (4.7) describes the PID control law:

$$U_1 = K_p(\mathrm{z}-\mathrm{z}_r) + K_i\int_0^t(\mathrm{z}-\mathrm{z}_r)dt + K_d(\dot{\mathrm{z}}-\dot{\mathrm{z}}_r) \tag{4.7}$$

Where z_r is the reference value or the desired value of altitude.

4.2.2 Roll Control

Another PID controller is designed for controlling the roll angle (ϕ) of the quadrotor system. It produces the control signal U_2 which is responsible for reducing the error of the roll angle. The control law of the PID for controlling the roll angle as follow:

$$U_2 = K_p(\phi-\phi_r) + K_i\int_0^t(\phi-\phi_r)dt + K_d(\dot{\phi}-\dot{\phi}_r) \tag{4.8}$$

Where ϕ_r is the reference value or the desired value of the roll angle.

4.2.3 Pitch Control

The third PID controller is designed for controlling the pitch angle (θ). It produces the control signal U_3 that reduces the error signal between the reference and real pitch angle. The control law of the PID for controlling the pitch angle as follow:

$$U_3 = K_p(\theta-\theta_r) + K_i\int_0^t(\theta-\theta_r)dt + K_d(\dot{\theta}-\dot{\theta}_r) \tag{4.9}$$

Where θ_r is the reference value or the desired value of the pitch angle.

4.2.4 Yaw Control

The last PID controller is designed for controlling the yaw angle (ψ). It produces the control signal U_4 that reduces the error signal between the reference and real yaw angle. The control law of the PID for controlling the yaw angle as follow:

$$U_4 = K_p(\psi - \psi_r) + K_i \int_0^t (\psi - \psi_r)dt + K_d(\psi - \dot{\psi}_r) \qquad (4.10)$$

Where ψ_r is the reference value or the desired value of the yaw angle.

4.3 Backstepping Controller

4.3.1 Introduction to Backstepping

The backstepping is a feedback control method that can be applied to nonlinear systems [36]. Such a recursive algorithm which is designed to stabilize the system progressively by creating simple inner control laws that stabilize each outer one, and gradually by attaining the final outer control law the algorithm will ensure the stabilization of the controlled system. It is basically used to control a particular type of nonlinear dynamical systems by means of Lyapunov stability theory. Unlike linear control algorithms that linearize the nonlinear systems, backstepping does not take the nonlinearities of the studied system in consideration and can extract a control law to control the nonlinear system.

4.3.2 Controller design

In this section, we will develop a new control law able to produce the proper input signals $u = [u_1, u_2, u_3, u_4]^T$ for DC motors. It aims to tracks asymptotically the desired trajectory $\{\phi_d, \theta_d, \psi_d, z_d\}$.

The dynamic model for a quadrotor are rewritten using the new form of state vectors ignoring gyroscopic effects and friction forces:

$$\begin{aligned}
\ddot{\phi} &= b_1 U_2 \\
\ddot{\theta} &= b_2 U_3 \\
\ddot{\psi} &= b_3 U_4 \\
\ddot{z} &= \frac{U_1}{m}(\cos(\phi)\cos(\theta)) - g \\
\ddot{x} &= \frac{U_1}{m}(\cos(\phi)\sin(\theta)\cos(\psi) + \sin(\phi)\sin(\psi)) \\
\ddot{y} &= \frac{U_1}{m}(\cos(\phi)\sin(\theta)\sin(\psi) + \sin(\phi)\cos(\psi))
\end{aligned} \qquad (4.11)$$

$$x_1 = \begin{bmatrix} \phi \\ \theta \end{bmatrix} \quad , \; x_2 = \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \end{bmatrix} \quad , \quad x_3 = \begin{bmatrix} \psi \\ z \end{bmatrix} \quad , \; x_4 = \begin{bmatrix} \dot{\psi} \\ \dot{z} \end{bmatrix} \quad , \; x_5 = \begin{bmatrix} \Omega_1 \\ \Omega_2 \\ \Omega_3 \\ \Omega_4 \end{bmatrix} \tag{4.12}$$

Let

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= g_0 \varphi_0 \\ \dot{x}_3 &= x_4 \\ \dot{x}_4 &= g_1 \varphi_1 + g_2 \\ \dot{x}_5 &= f_0 + g_3 u \end{aligned} \tag{4.13}$$

Where

$$g_0 = \begin{bmatrix} b_1 & 0 \\ 0 & b_2 \end{bmatrix}, \quad g_1 = \begin{bmatrix} b_3 & 0 \\ 0 & \dfrac{c\phi c\theta}{m} \end{bmatrix}, \quad g_2 = \begin{bmatrix} 0 \\ -g \end{bmatrix}, \quad g_3 = \begin{bmatrix} b \\ b \\ b \\ b \end{bmatrix}$$

$$\varphi_0 = \begin{bmatrix} k_f(\Omega_4^2 - \Omega_2^2) \\ k_f(\Omega_1^2 - \Omega_3^2) \end{bmatrix}, \quad \varphi_1 = \begin{bmatrix} k_m \sum_{i=1}^{4} (-1)^{i+1} \Omega_i^2 \\ k_f \sum_{i=1}^{4} \Omega_i^2 \end{bmatrix}$$

$$f_0 = \begin{bmatrix} -r_0 - r_1\Omega_1 - r_2\Omega_1^2 \\ -r_0 - r_1\Omega_2 - r_2\Omega_2^2 \\ -r_0 - r_1\Omega_3 - r_2\Omega_3^2 \\ -r_0 - r_1\Omega_4 - r_2\Omega_4^2 \end{bmatrix}$$

Using the backstepping approach, one can extract the control law forcing the quadrotor to follow the desired trajectories. It aims to construct a control law in five steps.

Step1:

First, we suppose a virtual system as following.

$$v_1 = \dot{x}_1 \tag{4.14}$$

Suppose that the first tracking error as the error in roll and pitch angles.

$$z_1 = x_{1d} - x_1 \tag{4.15}$$

Hence $\dot{z}_1 = \dot{x}_{1d} - \dot{x}_1 = \dot{x}_{1d} - x_2$.

Considering that the Lyapunov function as a positive definite function.

$$V_1 = \frac{1}{2} z_1^T z_1 \tag{4.16}$$

Its time derivative is

$$\dot{V}_1 = z_1^T \dot{z}_1 = z_1^T \left(\dot{x}_{1d} - x_2 \right) = z_1^T \left(\dot{x}_{1d} - v_1 \right) \tag{4.17}$$

The stabilization of z_1 can be obtained by presenting a first virtual control input v_1:

$$v_1 = \dot{x}_{1d} + A_1 z_1 \tag{4.18}$$

With $A_1 \in R^{2\times 2}$ is a positive definite matrix.

Step2:

For the second step we consider the following new virtual system v_2:

$$\dot{x}_2 = g_0 v_2 \tag{4.19}$$

Where v_2 is a second virtual control input. Let us proceed to suppose the second tracking error as following.

$$z_2 = v_1 - x_2 = \dot{x}_{1d} + A_1 z_1 - \dot{x}_1 = +A_1 z_1 + \dot{z}_1 \tag{4.20}$$

Hence $\dot{z}_1 = z_2 - A_1 z_1$.

The second augmented Lyapunov function are supposed to be:

$$V_2 = \frac{1}{2} \sum_{i=1}^{2} z_i^T z_i \tag{4.21}$$

The time derivative is:

$$\begin{aligned} \dot{V}_1 &= z_1^T \dot{z}_1 + z_2^T \dot{z}_2 \\ &= z_1^T \left(z_2 - A_1 z_1 \right) + z_2^T \left(\dot{v}_1 - \dot{x}_2 \right) \\ &= -z_1^T A_1 z_1 + z_2^T \left(z_1 + \dot{v}_1 - g_0 v_2 \right) \end{aligned} \tag{4.22}$$

The stabilization of z_2 are obtained by presenting the following virtual control.

$$v_2 = g_0^{-1}\left(z_1 + A_2 z_2 + \dot{v}_1\right) \tag{4.23}$$

With $A_2 \in R^{2\times 2}$ is a positive definite matrix.

Step3:

In this step we suppose the virtual system.

$$v_3 = \dot{x}_3 \tag{4.24}$$

Suppose that the third tracking error as the error in yaw angle and the altitude.

$$z_3 = x_{3d} - x_3 \tag{4.25}$$

Hence $\dot{z}_3 = \dot{x}_{3d} - \dot{x}_3 = \dot{x}_{3d} - x_4$

Considering that the Lyapunov function as a positive definite function.

$$V_3 = \frac{1}{2} z_3{}^T z_3 \tag{4.26}$$

The time derivative

$$\dot{V}_3 = z_3{}^T \dot{z}_3 = z_3{}^T \left(\dot{x}_{3d} - x_4\right) = z_3{}^T \left(\dot{x}_{3d} - v_3\right) \tag{4.27}$$

The stabilization of z_3 are achieved by presenting a virtual control:

$$v_3 = \dot{x}_{3d} + A_3 z_3 \tag{4.28}$$

With $A_3 \in R^{2\times 2}$ is a positive definite matrix.

Step 4:

For this step we suppose the following new virtual system v_4:

$$\dot{x}_4 = g_2 + g_1 v_4 \tag{4.29}$$

Let

$$z_4 = v_3 - x_4 = \dot{x}_{3d} + A_3 z_3 - \dot{x}_3 = +A_3 z_3 + \dot{z}_3 \tag{4.30}$$

Hence $\dot{z}_3 = z_4 - A_3 z_3$

The augmented Lyapunov function is:

$$V_4 = \frac{1}{2}\sum_{i=1}^{2} z_i^T z_i \qquad \text{i= 3 to 4} \qquad (4.31)$$

Its time derivative is

$$\begin{aligned} \dot{V}_4 &= z_3^T \dot{z}_3 + z_4^T \dot{z}_4 \\ &= z_3^T \left(z_4 - A_3 z_3\right) + z_4^T \left(\dot{v}_3 - \dot{x}_4\right) \\ &= -z_3^T A_3 z_3 + z_4^T \left(z_3 + \dot{v}_3 - g_2 - g_1 v_4\right) \end{aligned} \qquad (4.32)$$

The stabilization of z_4 are achieved by presenting a following virtual control:

$$v_4 = g_1^{-1}\left(z_3 + A_4 z_4 + \dot{v}_3 - g_2\right) \qquad (4.33)$$

With $A_4 \in R^{2\times 2}$ is a positive definite matrix.

Step 5:

For the last step we consider the motor dynamics.

$$\dot{x}_5 = f_0 + g_3 u \qquad (4.34)$$

Let

$$z_5 = \begin{bmatrix} v_2 - \varphi_0 \\ v_2 - \varphi_1 \end{bmatrix} = \begin{bmatrix} g_0^{-1}\left(z_1 + A_2 z_2 + \dot{v}_1 - g_0\varphi_0\right) \\ g_1^{-1}\left(z_3 + A_4 z_4 + \dot{v}_3 - g_1\varphi_1 - g_2\right) \end{bmatrix} = \begin{bmatrix} g_0^{-1}\left(z_1 + A_2 z_2 + \dot{z}_2\right) \\ g_1^{-1}\left(z_3 + A_4 z_4 + \dot{z}_4\right) \end{bmatrix} \qquad (4.35)$$

Hence

$$\dot{z}_2 = g_0^* z_5 - z_1 - A_2 z_2$$
$$g_0^* = \left[g_0, 0_{2x2}\right]$$

$$\dot{z}_4 = g_1^* z_5 - z_3 - A_4 z_4$$
$$g_1^* = \left[0_{2x2}, g_1\right]$$

The Lyapunov function of the whole system is supposed to be.

$$V_5 = \frac{1}{2}\sum_{i=1}^{5} z_i^T z_i \qquad (4.36)$$

Its time derivative is given

$$
\begin{aligned}
\dot{V}_5 &= z_1^T \dot{z}_1 + z_2^T \dot{z}_2 + z_3^T \dot{z}_3 + z_4^T \dot{z}_4 + z_5^T \dot{z}_5 \\
&= z_1^T \left(z_2 - A_1 z_1 \right) + z_2^T \left(g_0^* z_5 - z_1 - A_2 z_2 \right) \\
&+ z_3^T \left(z_4 - A_3 z_3 \right) + z_4^T \left(g_1^* z_5 - z_3 - A_4 z_4 \right) + z_5^T \left(\begin{bmatrix} \dot{v}_2 \\ \dot{v}_4 \end{bmatrix} - \begin{bmatrix} \dot{\varphi}_0 \\ \dot{\varphi}_1 \end{bmatrix} \right) \\
&= -\sum_{i=1}^{5} z_i^T A_i z_i + z_5^T \left(\begin{bmatrix} g_0 & 0_{2x2} \\ 0_{2x2} & g_1 \end{bmatrix}^T \begin{bmatrix} z_2 \\ z_4 \end{bmatrix} + \begin{bmatrix} \dot{v}_2 \\ \dot{v}_4 \end{bmatrix} - \begin{bmatrix} J_1 \\ J_2 \end{bmatrix} (f_0 + \mathrm{g}_3 \mathrm{u}) \right)
\end{aligned}
\tag{4.37}
$$

Where

$$
J_1 = \dot{\varphi}_0 = \frac{\partial \varphi_0}{\partial \Omega_i} \dot{\Omega}_i = \begin{bmatrix} 0 & -2k_f\Omega_2 & 0 & 2k_f\Omega_4 \\ 2k_f\Omega_1 & 0 & -2k_f\Omega_3 & 0 \end{bmatrix} \dot{\Omega}_i \tag{4.38}
$$

$$
J_2 = \dot{\varphi}_1 = \frac{\partial \varphi_1}{\partial \Omega_i} \dot{\Omega}_i = \begin{bmatrix} 2k_m\Omega_1 & -2k_m\Omega_2 & 2k_m\Omega_3 & -2k_m\Omega_4 \\ 2k_f\Omega_1 & 2k_f\Omega_2 & 2k_f\Omega_3 & 2k_f\Omega_4 \end{bmatrix} \dot{\Omega}_i \tag{4.39}
$$

Therefore, the stabilization of the whole system are achieved by presenting a following control law:

$$
u = \mathrm{g}_3^{-1} \left\{ \begin{bmatrix} J_1 \\ J_2 \end{bmatrix}^{-1} \left(\begin{bmatrix} g_0 & 0_{2x2} \\ 0_{2x2} & g_1 \end{bmatrix}^T \begin{bmatrix} z_2 \\ z_4 \end{bmatrix} + \begin{bmatrix} \dot{v}_2 \\ \dot{v}_4 \end{bmatrix} + A_5 z_5 \right) - f_0 \right\} \tag{4.40}
$$

Where

$$
\begin{bmatrix} J_1 \\ J_2 \end{bmatrix}^{-1} = \begin{bmatrix} 0 & \frac{1}{4k_f\Omega_1} & \frac{1}{8k_m\Omega_1} & \frac{1}{8k_f\Omega_1} \\ \frac{-1}{4k_f\Omega_2} & 0 & \frac{-1}{8k_m\Omega_2} & \frac{1}{8k_f\Omega_2} \\ 0 & \frac{-1}{4k_f\Omega_3} & \frac{1}{8k_m\Omega_3} & \frac{1}{8k_f\Omega_3} \\ \frac{1}{4k_f\Omega_4} & 0 & \frac{-1}{8k_m\Omega_4} & \frac{1}{8k_f\Omega_4} \end{bmatrix}
$$

$$
z_5 = \begin{bmatrix} g_0^{-1} \left(z_1 + A_2 z_2 + \dot{z}_2 \right) \\ g_1^{-1} \left(z_3 + A_4 z_4 + \dot{z}_4 \right) \end{bmatrix}
$$

4.4 Sliding Mode Controller

Sliding Mode Controller (SMC) is suggested to control the quadrotor model because of its nonlinearity properties.

4.4.1 Introduction to SMC

Sliding mode control is one kind of Variable Structure Control (VSC). SMC control law compels the state paths to keep track of user-defined state paths on a defined sliding surface. It is composed of a corrective and an equivalent control parts. The corrective control part goals to redress any changes of the state paths from the determined sliding surface. At the same time, the equivalent control part keeps the time derivative of the determined surface close to zero in order to retain the state paths on the determined sliding surface.

$$U(\mathrm{t}) = U_c(\mathrm{t}) + U_{eq}(\mathrm{t}) \tag{4.41}$$

The SMC control signal U is composed of U_c that represents the corrective control part and U_{eq} that represents the equivalent control part.

The sliding mode controller block diagram is displayed in Figure 4.4.

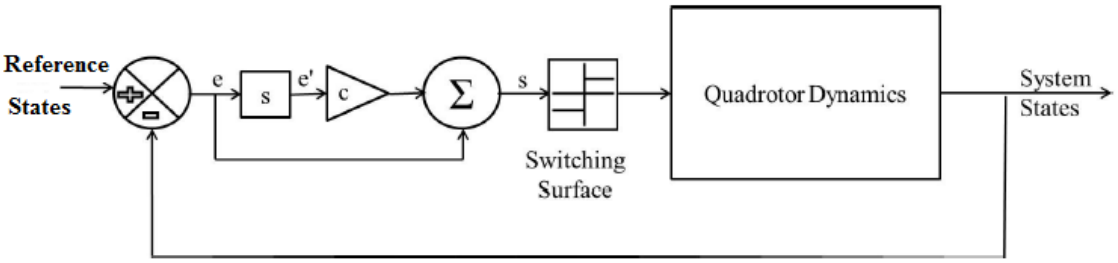

Figure 4.4 Sliding Mode Control Block Diagram

4.4.2 Roll Control

The SMC generates control law to pursue a determined reference path of the roll angle. The difference between the real and reference value of the roll angle determined as:

$$e = \phi_d - \phi \tag{4.42}$$

The determined sliding surface is identified as following

$$s = c_1 e + \dot{e} \tag{4.43}$$

The parameter c1 is a positive constant. The derivative of the previous equation takes the next formula:

$$\begin{aligned} \dot{s} &= c_1 \dot{e} + \ddot{e} \\ &= c_1(\dot{\phi}_d - \dot{\phi}) + (\ddot{\phi}_d - \ddot{\phi}) \end{aligned} \tag{4.44}$$

The next equation is resulted by defining the following Lyapunov function:

$$V = \frac{1}{2}(\mathrm{e}^2 + \mathrm{s}^2) \tag{4.45}$$

An exponential law is suggested depending on the Lyapunov function as in the next equation:

$$\dot{s} = -k_1 \,\mathrm{sign(s)} - k_2 \,\mathrm{s} \tag{4.46}$$

By considering that k_1 and k_2 are positive constants and defining the variable sign(s) as

$$\mathrm{sign(s)} = \begin{cases} -1 & if\ s < 0 \\ +1 & if\ s > 0 \end{cases}$$

(4.47)

The suggested equations (4.46) and (4.44) are equal to each other. The control signal U_2 is computed by replacing $\ddot{\phi}$ by its value from equation (3.30).

$$U_2 = \frac{1}{b_1}\left[k_1 \,\mathrm{sign(s)} + \mathrm{k}_2 \, s + c_1(\dot{\phi}_d - \dot{\phi}) + \ddot{\phi}_d + a_2 \dot{\theta} \Omega_r - a_1 \dot{\theta} \dot{\psi} \right]$$

(4.48)

4.4.3 Pitch Control

The control law for the pitch angle is extracted as the same method as the roll control. The extracted control law U_3 is described as follows:

$$U_3 = \frac{1}{b_2}\left[k_1 \,\mathrm{sign(s)} + \mathrm{k}_2 \, s + c_1(\dot{\theta}_d - \dot{\theta}) + \ddot{\theta}_d - a_4 \dot{\phi} \Omega_r - a_3 \dot{\phi} \dot{\psi} \right] \tag{4.49}$$

4.4.4 Yaw Control

The control law for the yaw angle is extracted as the same method as the previous control laws. The extracted control law U_4 is described as follows:

$$U_4 = \frac{1}{b_3}\left[k_1 \,\mathrm{sgn(s)} + \mathrm{k}_2\, s + c_1(\dot{\psi}_d - \dot{\psi}) + \ddot{\psi}_d - a_5 \dot{\phi}\,\dot{\theta} \right] \tag{4.50}$$

4.4.5 Altitude Control

The difference between the real and desired value of the altitude is determined as an error signal:

$$e = z - z_d \tag{4.51}$$

The determined sliding surface is identified as following:

$$s = c_1 e + \dot{e} \tag{4.52}$$

By considering that c_1 as a positive constant. The sliding surface derivative is equal to the exponential law and it is described in next equation.

$$-k_1 \,\mathrm{sign(s)} - k_2\, \mathrm{s} = \mathrm{c}_1(\dot{\mathrm{z}} - \dot{\mathrm{z}}_d) + \ddot{\mathrm{z}} - \ddot{\mathrm{z}}_d \tag{4.53}$$

The control signal U_1 is computed by replacing $\ddot{z}$ by its value in equation (3.40).

$$U_1 = \frac{m}{\cos(\phi)\cos(\theta)}\left[k_1 \,\mathrm{sgn(s)} + \mathrm{k}_2\, s + c_1(\dot{z} - \dot{z}_d) + g - \ddot{z}_d \right] \tag{4.54}$$

4
RESULTS & DISCUSSION

5.1 PID Controller Simulation

In this section, a comparison between three techniques to tune the PID parameters are introduced. These techniques are trial and error, particle swarm optimization (PSO), and adaptive particle swarm optimization (APSO). The tuned PID controllers are used for controlling the altitude and yaw, pitch, and roll angles of the quadrotor.

The first technique to tune the PID controllers is a trial and error. In this technique, the PID parameters are tuned experimentally until reaching a good performance. The second technique to tune the PID controllers is the standard particle swarm optimization algorithm. The acceleration coefficients are equal to '2' and constant inertia weight equals '1'. The third technique to tune the PID controllers is the adaptive particle swarm optimization algorithm. The acceleration coefficients are constants and equal to '2' and inertia weight is changed in accordance with equation (4.5).

PSO algorithm depends on the fitness function to determine the competence of particles. The used fitness function is sum square error (SSE) and is defined as follows:

$$F = 0.25(\phi_r - \phi)^2 + 0.25(\theta_r - \theta)^2 + 0.25(\psi_r - \psi)^2 + 0.25(z_r - z)^2 \qquad (5.1)$$

Where Φ_r is the desired roll angle, θ_r is the desired pitch angle, ψ_r is the desired yaw angle, and z_r is the desired altitude. According to equation (5.1), the fitness function measures the particles competence depending on the error signals. The particles or solutions with less fitness value have better performance than particles with higher fitness value.

Two different simulations are obtained to test the performance of the backstepping controller. In the first simulation, the roll ϕ, pitch θ, yaw ψ angles are set with low initial values of 15 degrees and desired values of 0 degree, while the quadrotor has an initial altitude of 3 meters and the desired altitude of 4 meters. Figures 5.1, 5.2, and 5.3 demonstrate the roll (ϕ), pitch (θ), and yaw (ψ) angles response using the PID controllers which are tuned by three techniques. Altitude (z) response using the tuned PID controllers is displayed in the Figure 5.4.

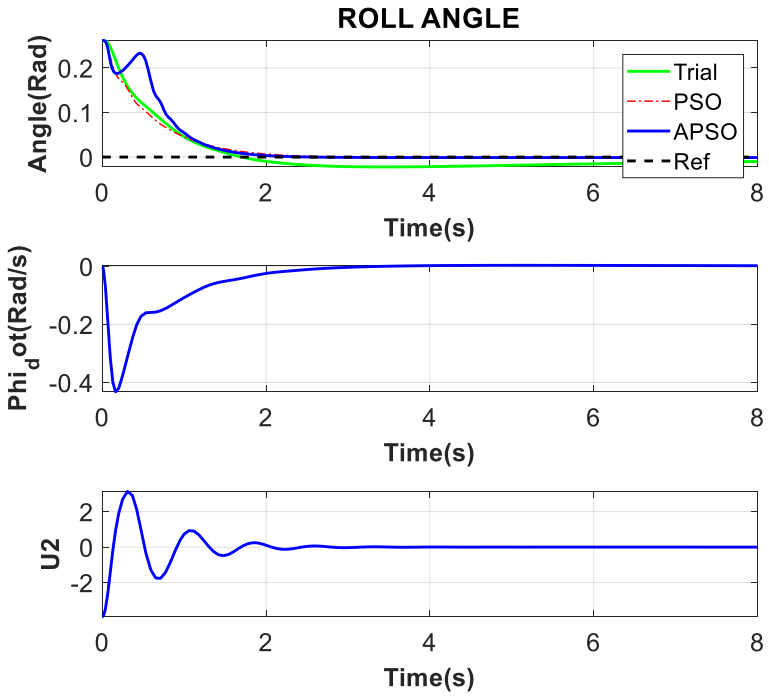

Figure 5.1 Roll Angle Responses Using The Tuned PID Controllers With Low Initial Values Of 15 Degrees

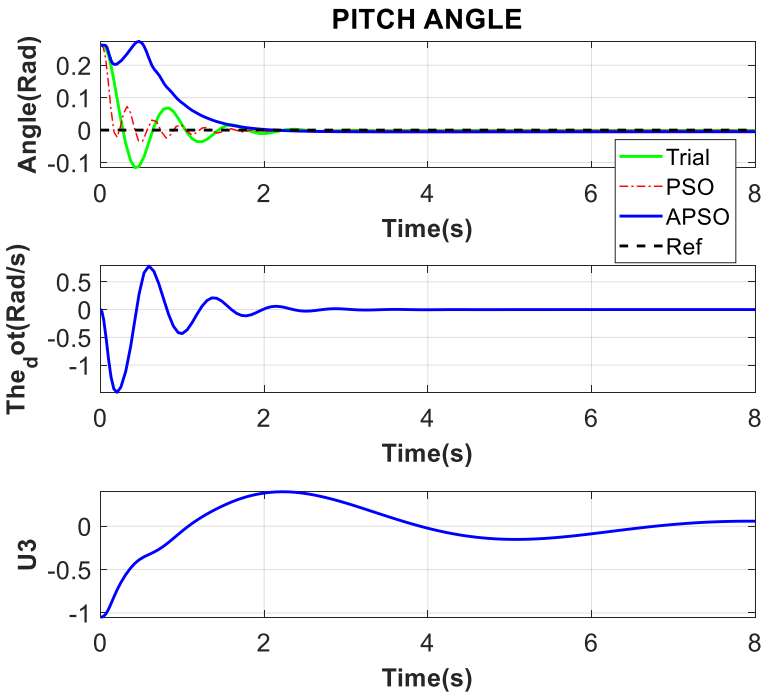

Figure 5.2 Pitch Angle Responses Using The Tuned PID Controllers With Low Initial Values Of 15 Degrees

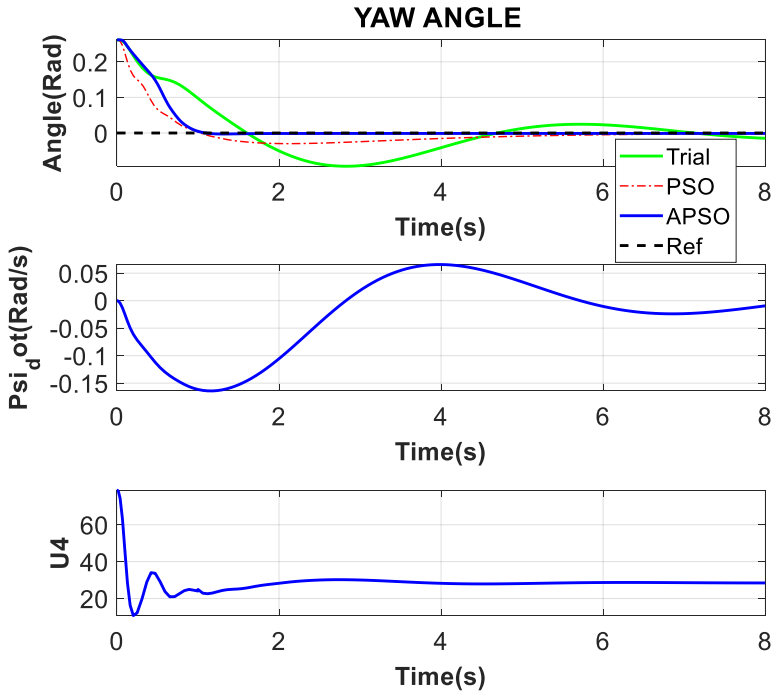

Figure 5.3 Yaw Angle Responses Using The Tuned PID Controllers With Low Initial Values Of 15 Degrees

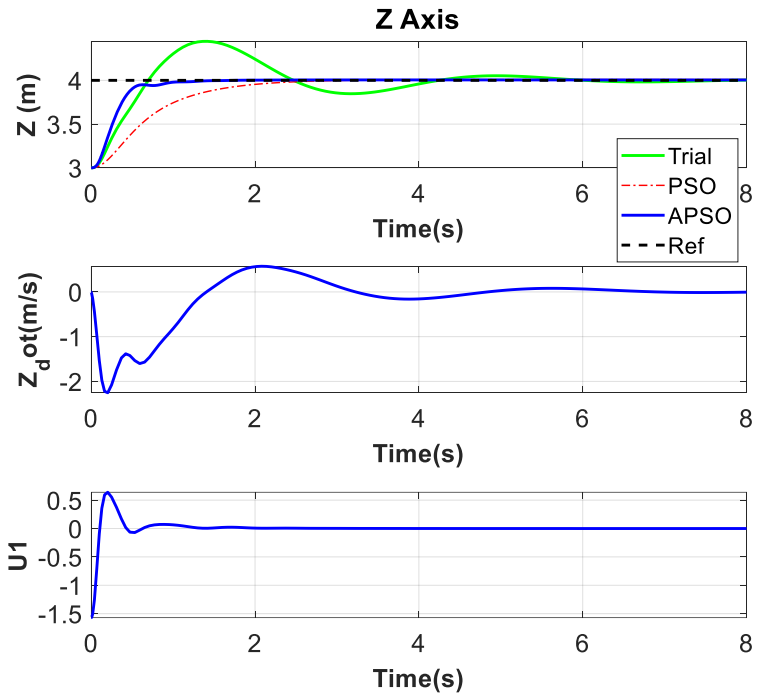

Figure 5.4 Altitude Responses Using The Tuned PID Controllers

In the second simulation made in case of model uncertainty, the roll ϕ, pitch θ, yaw ψ angles have initial values of 15 degrees and desired values of 0 degree, while the quadrotor has an initial altitude of 3 meters and the desired altitude of 4 meters the mass of the model had (650g, 600g, 550g). Figures 5.5, 5.6, and 5.7 demonstrate the roll (ϕ), pitch (θ), and yaw (ψ) angles response using the PID controllers which are tuned by three techniques. Altitude (z) response using the tuned PID controllers is displayed in the Figure 5.8. Simulation results demonstrate that the Optimized PID controllers with APSO algorithm confirmed the stability of the system with uncertainty of the drone model.

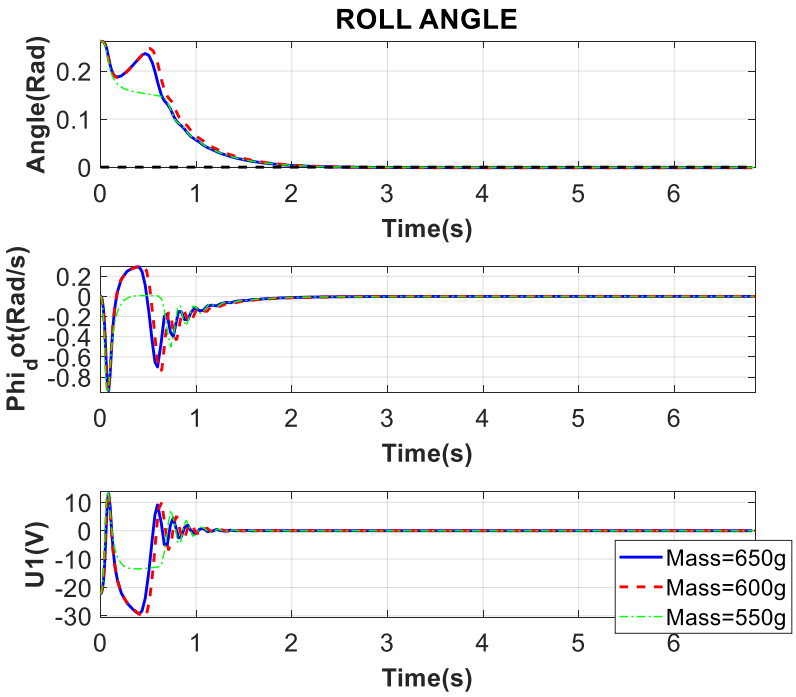

Figure 5.5 Roll Angle Responses Using The Tuned PID Controllers With Model Uncertainty

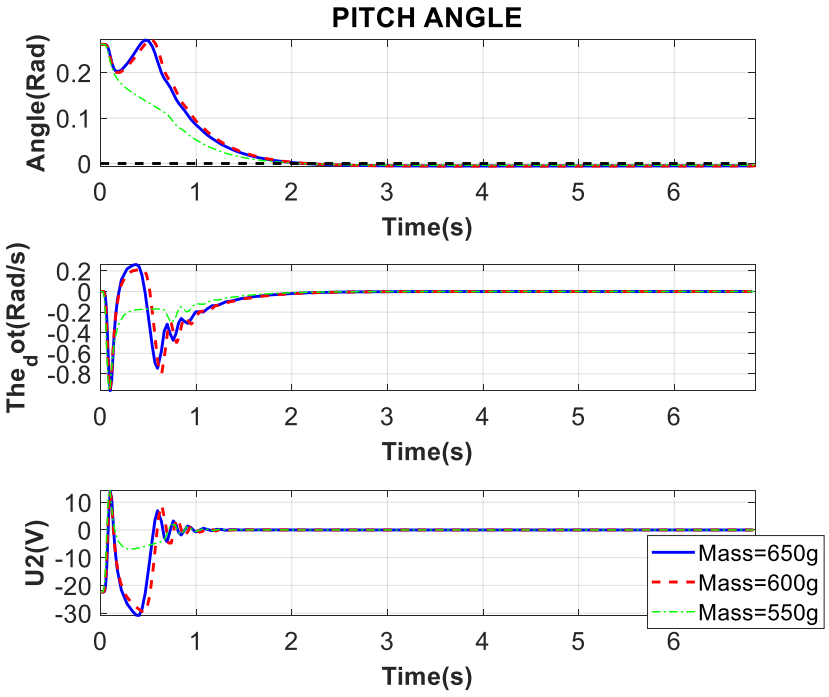

Figure 5.6 Pitch Angle Responses Using The Tuned PID Controllers With Model Uncertainty

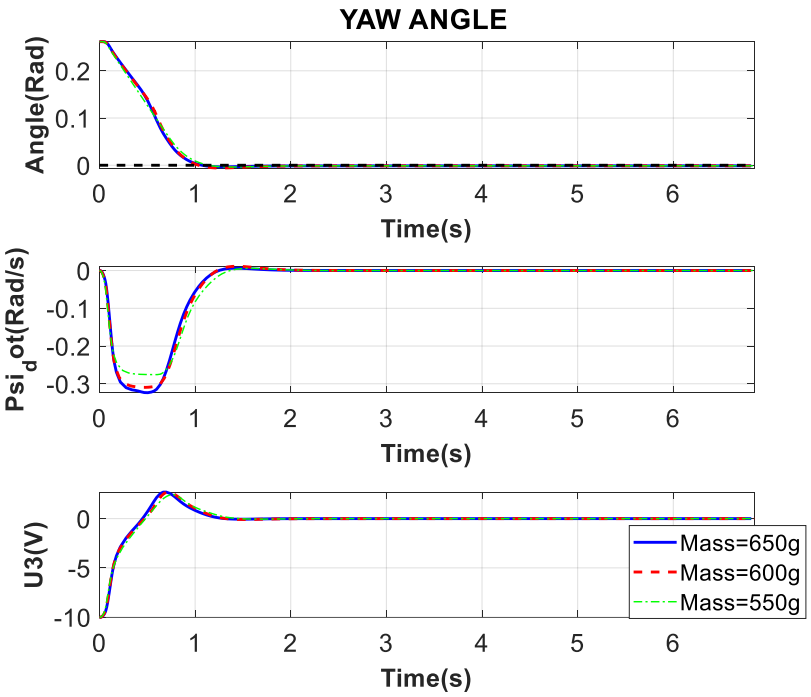

Figure 5.7 Yaw Angle Responses Using The Tuned PID Controllers With Model Uncertainty

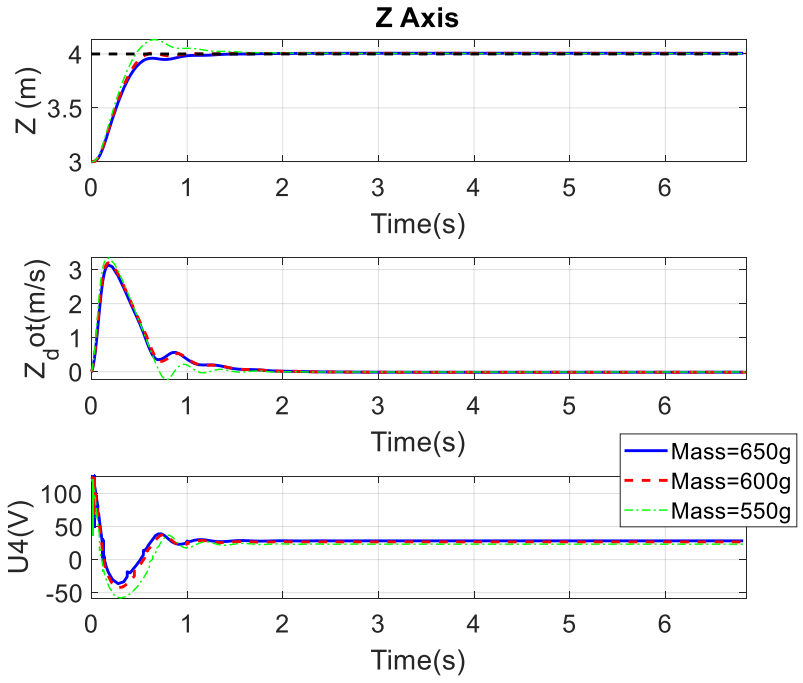

Figure 5.8 Altitude Responses Using The Tuned PID Controllers With Model Uncertainty

The time response specifications of quadrotor system using PID controllers tuned by trial and error, standard PSO, and the suggested adaptive PSO algorithm are presented in tables 5.1, 5.2, 5.3, and 5.4.

Table 5.1 Time Response Specifications for the Roll Angle Using The Tuned PID Controllers

Roll Angle (Φ)	PID Controller Trial and Error	PID Controller Standard PSO	PID Controller Adaptive PSO
Settling time(s)	11.1	2.15	1.42
Rise time(s)	1.14	1.24	0.7
Overshoot (%)	8.68	0.64	0.23
Steady state error	Approx. 0	Approx. 0	Approx. 0

Table 5.2 Time Response Specifications for the Pitch Angle Using The Tuned PID Controllers

Pitch Angle (θ)	PID Controller Trial and Error	PID Controller Standard PSO	PID Controller Adaptive PSO
Settling time(s)	1.86	1.55	1.53
Rise time(s)	0.15	0.09	0.88
Overshoot (%)	36.85	12.87	1.21
Steady state error	Approx. 0	Approx. 0	Approx. 0

Table 5.3 Time Response Specification for the Yaw Angle Using The Tuned PID Controllers

Yaw Angle (ψ)	PID Controller Trial and Error	PID Controller Standard PSO	PID Controller Adaptive PSO
Settling time(s)	9.27	6.14	1.37
Rise time(s)	1.16	0.75	0.65
Overshoot (%)	33.43	12.19	0.48
Steady state error	Approx. 0	Approx. 0	Approx. 0

Table 5.4 Time Response Specification for the Altitude Using the Tuned PID Controllers

Altitude Position (Z)	PID Controller Trial and Error	PID Controller Standard PSO	PID Controller Adaptive PSO
Settling time(s)	6.82	2.22	1.39
Rise time(s)	0.47	1.24	0.72
Overshoot (%)	11.78	0.23	0.16
Steady state error	Approx. 0	Approx. 0	Approx. 0

The resulted PID parameters from the tuning process using the three tuning techniques are listed in table 5.5.

Table 5.5 Resulted PID Controllers Parameters from the Tuning Processes

PID Controller Parameters	Trial and Error			Standard PSO			Adaptive PSO		
	K_p	K_i	K_d	K_p	K_i	K_d	K_p	K_i	K_d
Roll Angle	6	1.1	4.5	25.75	0.3	15.6	84.6	0.49	34.53
Pitch angle	15	0.9	2.3	85.55	3.87	9.64	85.39	2.15	35.45
Yaw angle	4	0.5	2.6	40.91	16.41	25.63	38.3	0.38	18.56
Altitude	50	133	40	12.2	0.14	10.3	97.8	1.6	42.3

Results illustrate that the suggested adaptive PSO technique to tune the PID controller give the best response to control the roll, pitch, yaw, and altitude with the lowest overshoot, settling time and has no steady-state error. The tuning process of the PID controller tuned by trial and error technique has the worst response specifications. The standard PSO strategy to tune the PID controllers gives satisfactory responses specifications. Besides, there is near to zero steady-state error for all PID controllers depending on the three tuning techniques.

5.2 Backstepping Controller Simulation

The backstepping controller is designed depending on the control inputs u described in equation (4.40). This controller contains four positive definite matrices A_1 to A_4 that need to be tuned.

Three different simulations are obtained to test the performance of the backstepping controller. In the first simulation, the roll ϕ, pitch θ, yaw ψ angles have low initial values of 15 degrees and desired values of 0 degree, while the quadrotor has an initial altitude of 3 meters and the desired altitude of 4 meters. Figures 5.9, 5.10, and 5.11 demonstrate the roll (ϕ), pitch (θ), and yaw (ψ) angles responses using the backstepping controller. Altitude (z) response using the tuned backstepping controller is presented in Figure 5.12 showing how it responses to the disturbances at time 7sec.

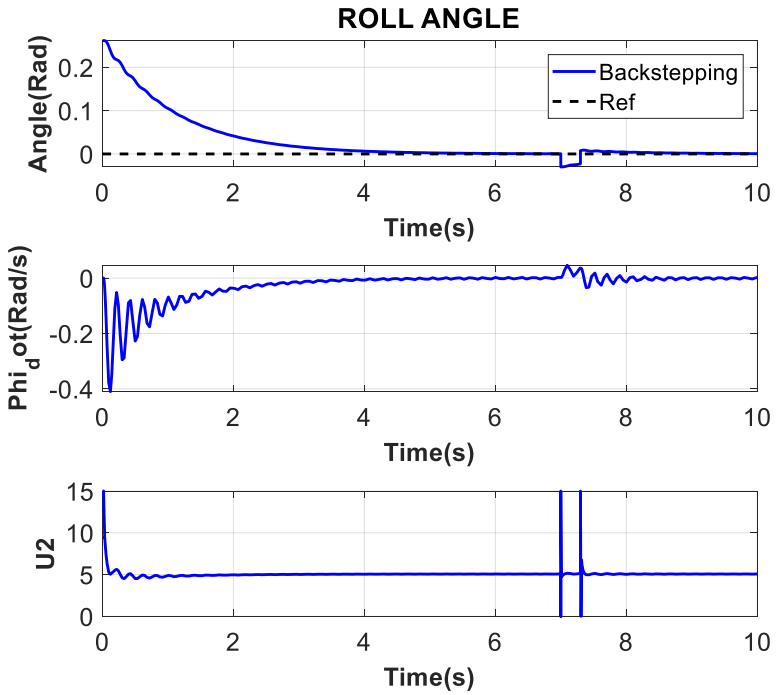

Figure 5.9 Roll Angle Responses Using The Tuned Backstepping Controllers With Low Initial Value Of 15 Degrees

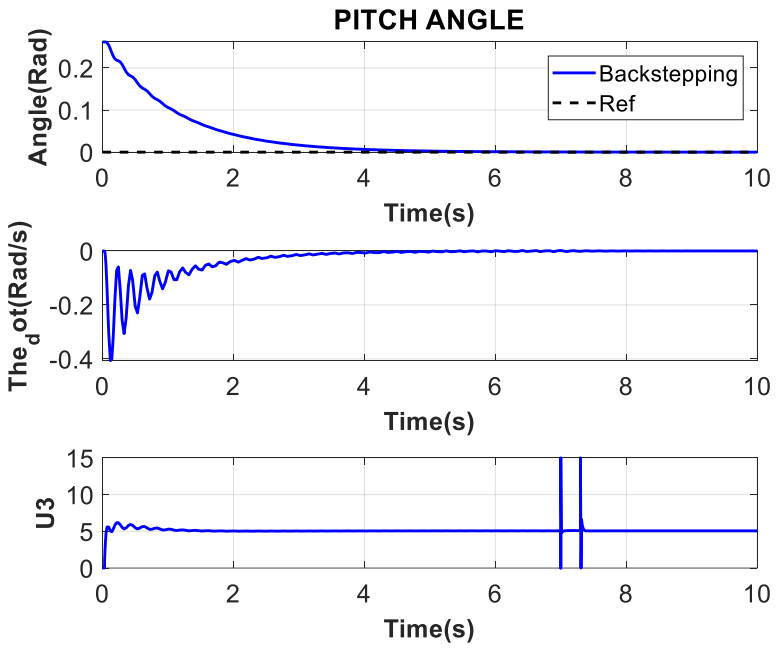

Figure 5.10 Pitch Angle Responses Using the Tuned Backstepping Controllers With Low Initial Value Of 15 Degrees

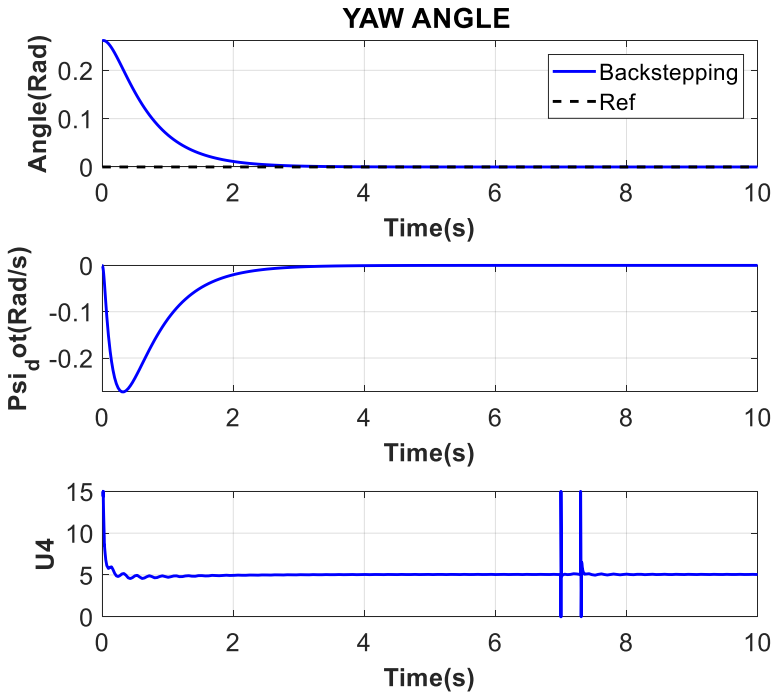

Figure 5.11 Yaw Angle Responses Using the Tuned Backstepping Controllers With Low Initial Value Of 15 Degrees

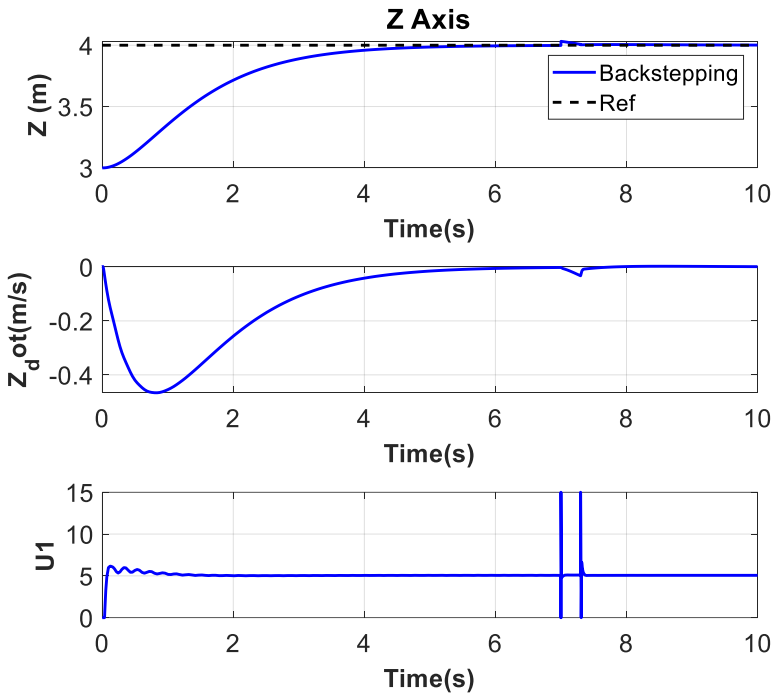

Figure 5.12 Altitude Responses Using the Tuned Backstepping Controllers With Low Initial Angle Values

In the second simulation, the roll ϕ, pitch θ, yaw ψ angles have high initial values of 55 degrees and desired values of 0 degree, while the quadrotor has an initial altitude of 3 meters and the desired altitude of 4 meters. Figures 5.13, 5.14, and 5.15 demonstrate the roll (ϕ), pitch (θ), and yaw (ψ) angles responses using the backstepping controller. Altitude (z) response using the tuned backstepping controller is presented in Figure 5.16.

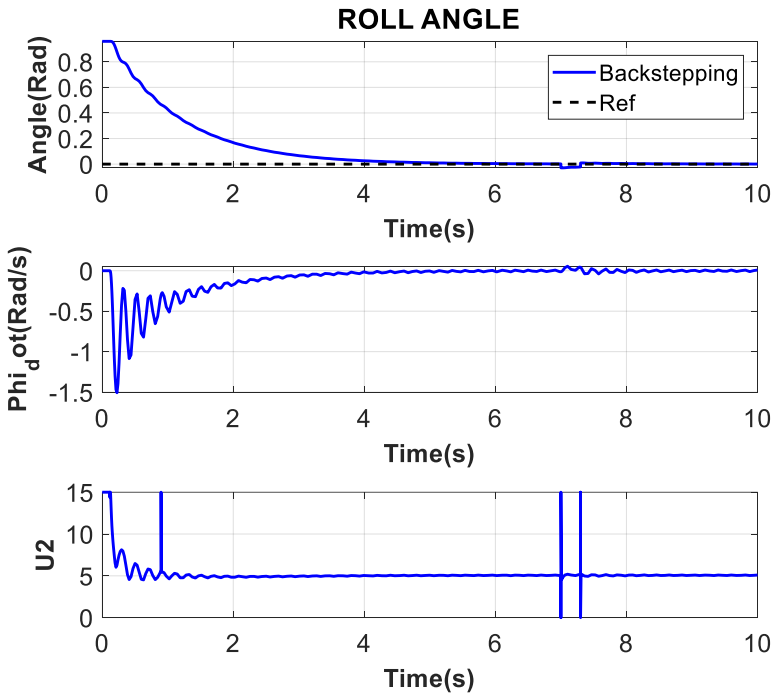

Figure 5.13 Roll Angle Responses Using The Tuned Backstepping Controllers With High Initial Value Of 55 Degrees And Without Disturbances

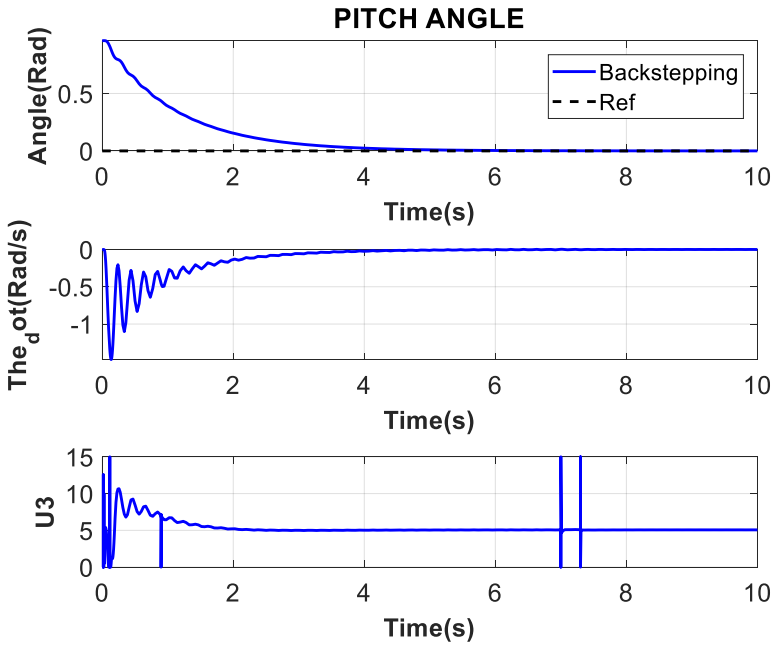

Figure 5.14 Pitch Angle Responses Using the Tuned Backstepping Controllers With High Initial Value Of 55 Degrees And Without Disturbances

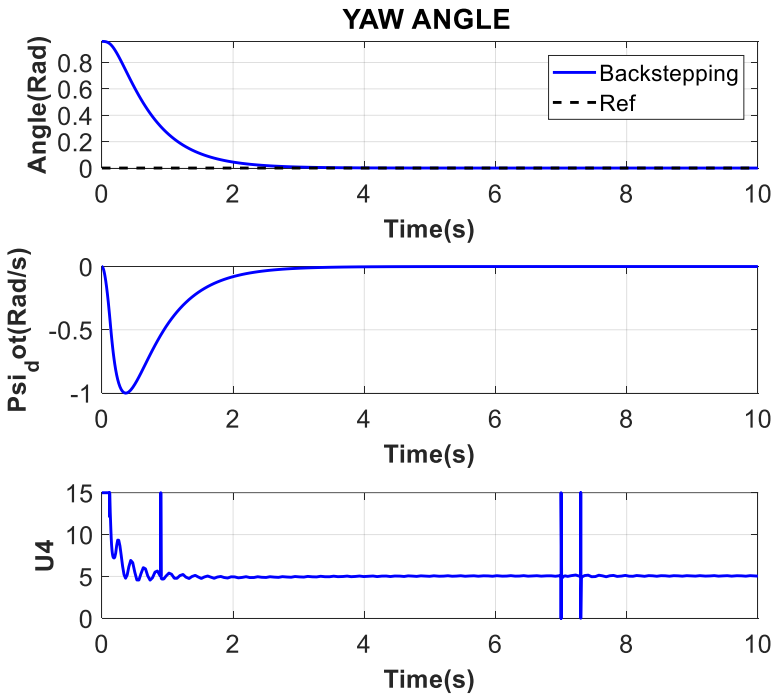

Figure 5.15 Yaw Angle Responses Using the Tuned Backstepping Controllers With High Initial Value Of 55 Degrees And Without Disturbances

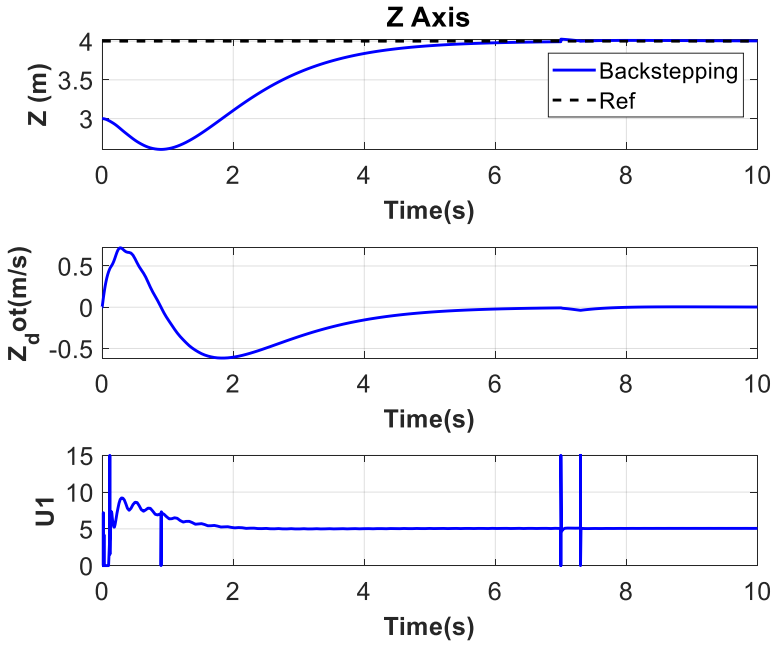

Figure 5.16 Altitude Responses Using the Tuned Backstepping Controllers With High Initial Angle Values And Without Disturbances

The time response specification of quadrotor system using Backstepping controller are given in tables 5.6, 5.7, 5.8, and 5.9.

Table 5.6 Time Response Specifications for the Roll Angle Using The Tuned Backstepping Controllers

Roll Angle (Φ)	Backstepping Controller
Settling time(s)	3.88
Rise time(s)	2.19
Overshoot (%)	0
Steady state error	Approx. 0

Table 5.7 Time Response Specifications for the Pitch Angle Using The Tuned Backstepping Controllers

Pitch Angle (θ)	Backstepping Controller
Settling time(s)	3.91
Rise time(s)	2.19
Overshoot (%)	0
Steady state error	Approx. 0

Table 5.8 Time Response Specifications for the Yaw Angle Using The Tuned Backstepping Controllers

Yaw Angle (ψ)	Backstepping Controller
Settling time(s)	7.81
Rise time(s)	4.39
Overshoot (%)	0
Steady state error	Approx. 0

Table 5.9 Time Response Specifications for the Altitude Using The Tuned Backstepping Controllers

Altitude Position (Z)	Backstepping Controller
Settling time(s)	7.42
Rise time(s)	3.98
Overshoot (%)	0
Steady state error	Approx. 0

The resulted matrices of backstepping controllers from the tuning techniques are listed in table 5.10.

Table 5.10 Resulted matrices of Backstepping Controllers from the Tuning Processes

Backstepping Contrqoller	A_1	A_2	A_3	A_4
Controller Parameters	diag([1,1])	diag([2,2])	diag([0.5,0.5])	diag([5,5])

In the third simulation, the mass (m) of the quadrotor considered to be uncertain and changed between (650 and 550 g) and the roll ϕ, pitch θ, yaw ψ angles have high initial values of 55 degrees and desired values of 0 degree, while the quadrotor has an initial altitude of 3 meters and the desired altitude of 4 meters. Figures 5.17, 5.18, and 5.19 demonstrate the roll (ϕ), pitch (θ), and yaw (ψ) angles responses using the backstepping controller. Altitude (z) response using the tuned backstepping controller is presented in Figure 5.20.

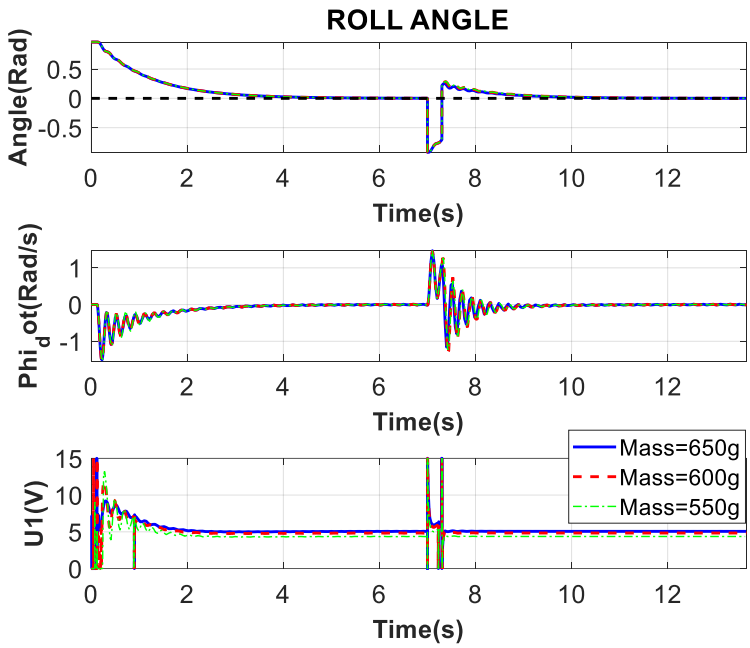

Figure 5.17 Roll Angle Responses Using the Tuned Backstepping Controllers with High Initial Value and The Case of Uncertainty & Disturbances

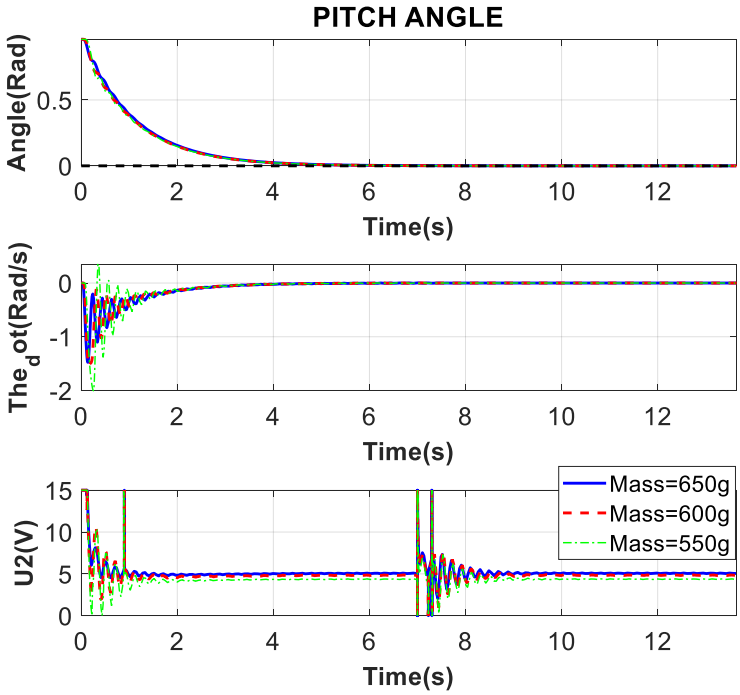

Figure 5.18 Pitch Angle Responses Using the Tuned Backstepping Controllers With High Initial Value and The Case of Uncertainty & Disturbances

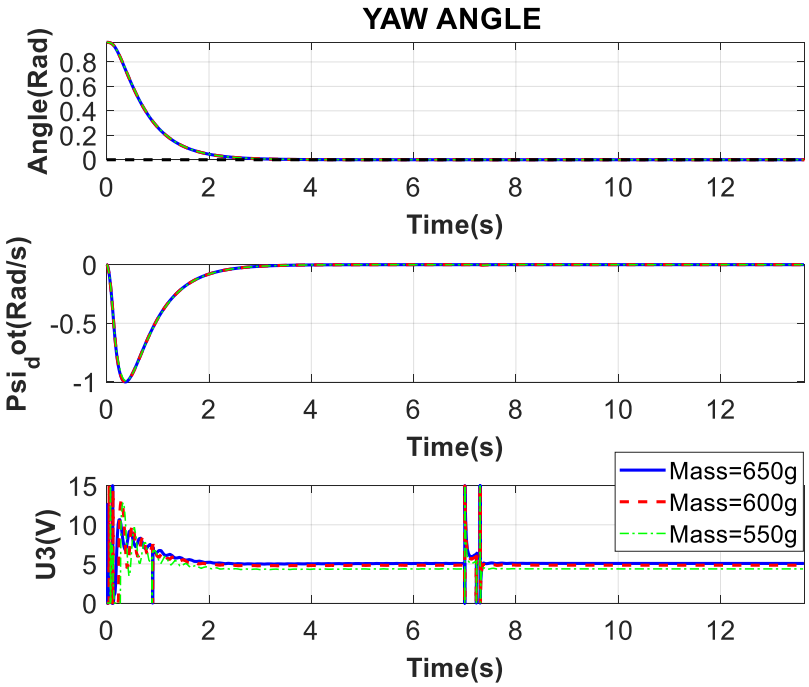

Figure 5.19 Yaw Angle Responses Using the Tuned Backstepping Controllers With High Initial Value and The Case of Uncertainty & Disturbances

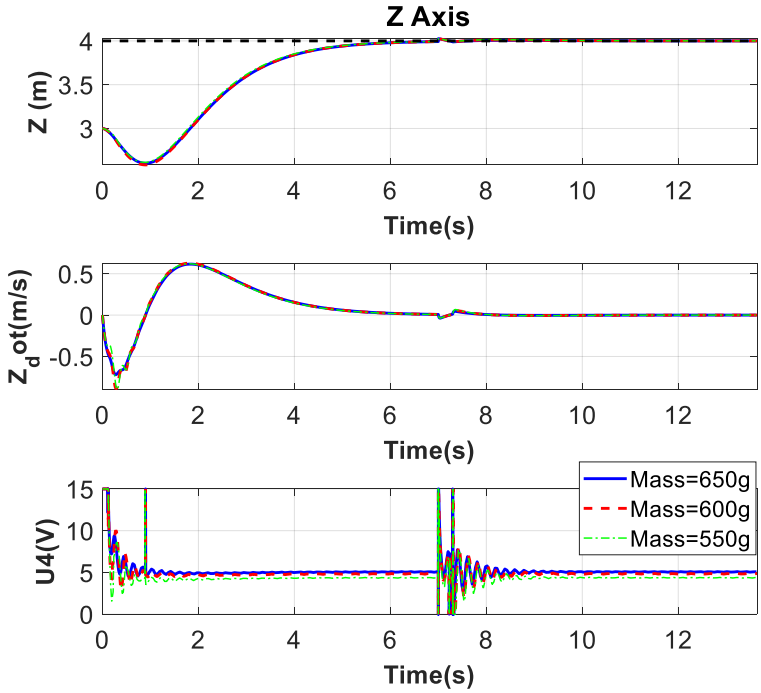

Figure 5.20 Altitude Responses Using the Tuned Backstepping Controllers With High Initial Value and The Case of Uncertainty & Disturbances

Results illustrate that backstepping control method gives good response specifications with almost no overshoot and no steady-state error. The suggested backstepping controller gives better response to control the roll, yaw, pitch, and altitude with a lower settling time, with the ability to handle the disturbances and the nonlinearity and the uncertainty of the system.

5.3 Sliding Mode Controller Simulation

The sliding mode controller is designed depending on the control inputs U_1 to U_4 described in equations 4.48, 4.49, 4.50, and 4.54. Each equation contains three design parameters c_1, k_1, and k_2 that need to be tuned.

Two different simulations are obtained to test the performance of the backstepping controller. In the first simulation, the roll ϕ, pitch θ, yaw ψ angles have initial values of 15 degrees and desired values of 0 degree, while the quadrotor has an initial altitude of 3 meters and the desired altitude of 4 meters.

At the same time external disturbances were applied on the altitude (z) at the time (5 sec) of the simulation and on the roll angel (ϕ) at the time of (7 sec). Figures 5.21, 5.22, and 5.23 demonstrate the roll (ϕ), pitch (θ), and yaw (ψ) angles responses using the sliding mode controller. Altitude (z) response using the tuned sliding mode controller is presented in Figure 5.24.

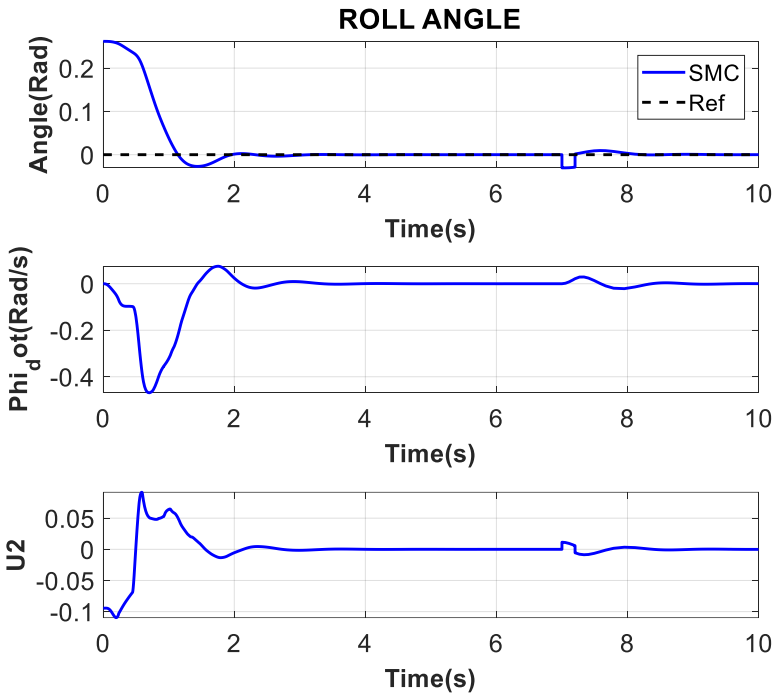

Figure 5.21 Roll Angle Responses Using the Tuned Sliding Mode Controllers With Low Initial Value Of 15 Degrees

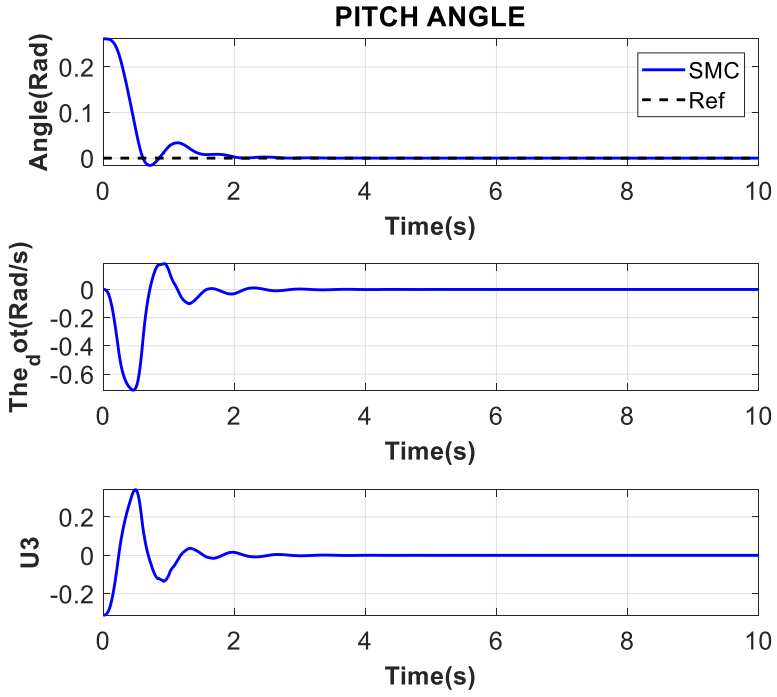

Figure 5.22 Pitch Angle Responses Using the Tuned Sliding Mode Controllers With Low Initial Value Of 15 Degrees

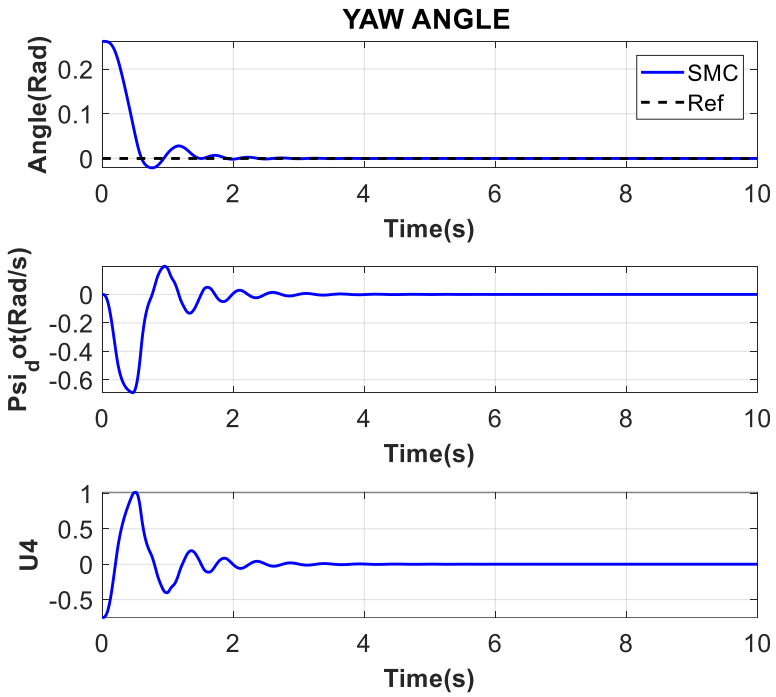

Figure 5.23 Yaw Angle Responses Using the Tuned Sliding Mode Controllers With Low Initial Value Of 15 Degrees

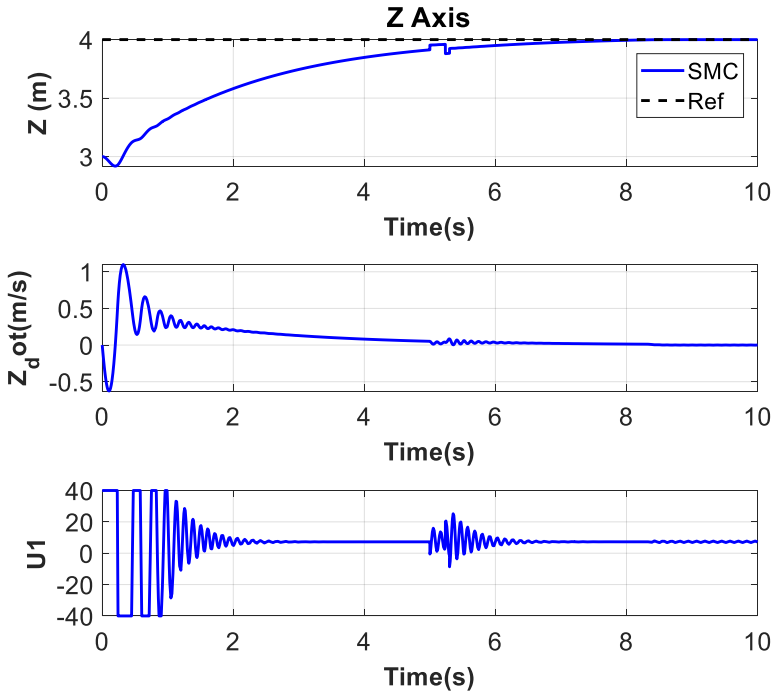

Figure 5.24 Altitude Responses Using the Tuned Sliding Mode Controllers With Low Initial Angle Values

In the second simulation, the roll ϕ, pitch θ, yaw ψ angles have initial values of 55 degrees and desired values of 0 degree, while the quadrotor has an initial altitude of 3 meters and the desired altitude of 4 meters. Figures 5.25, 5.26, and 5.27 demonstrate the roll (ϕ), pitch (θ), and yaw (ψ) angles responses using the sliding mode controller which is tuned by trial and error and APSO algorithm. Altitude (z) response using the tuned sliding mode controller is presented in Figure 5.28.

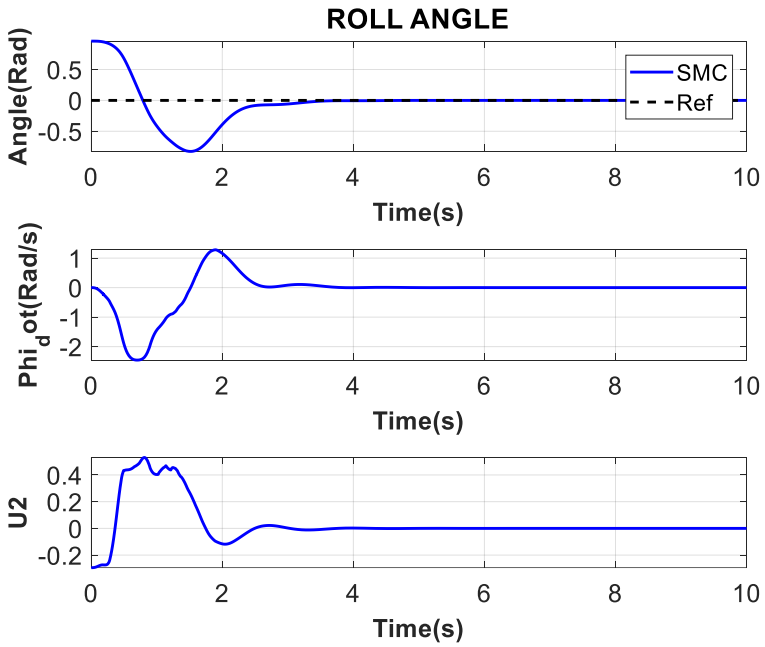

Figure 5.25 Roll Angle Responses Using the Tuned Sliding Mode Controllers With High Initial Value Of 55 Degrees

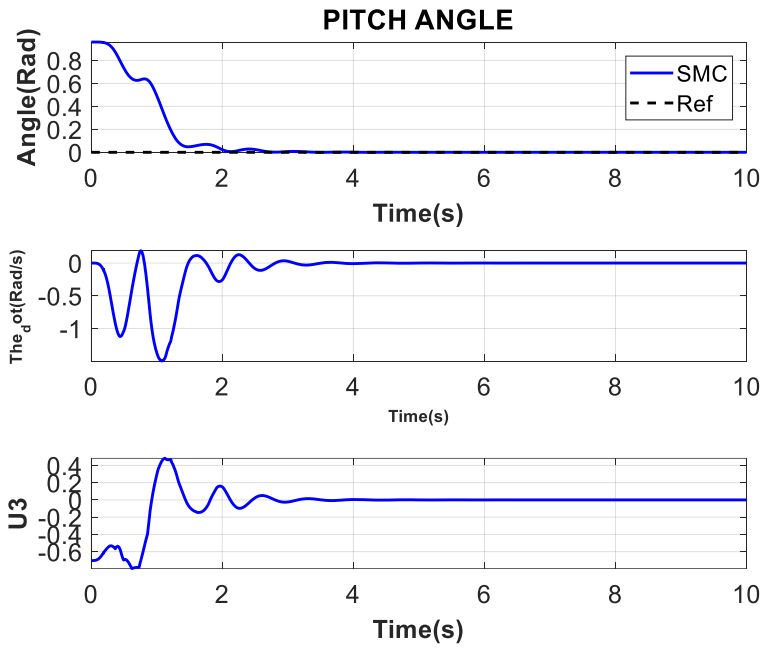

Figure 5.26 Pitch Angle Responses Using the Tuned Sliding Mode Controllers With High Initial Value Of 55 Degrees

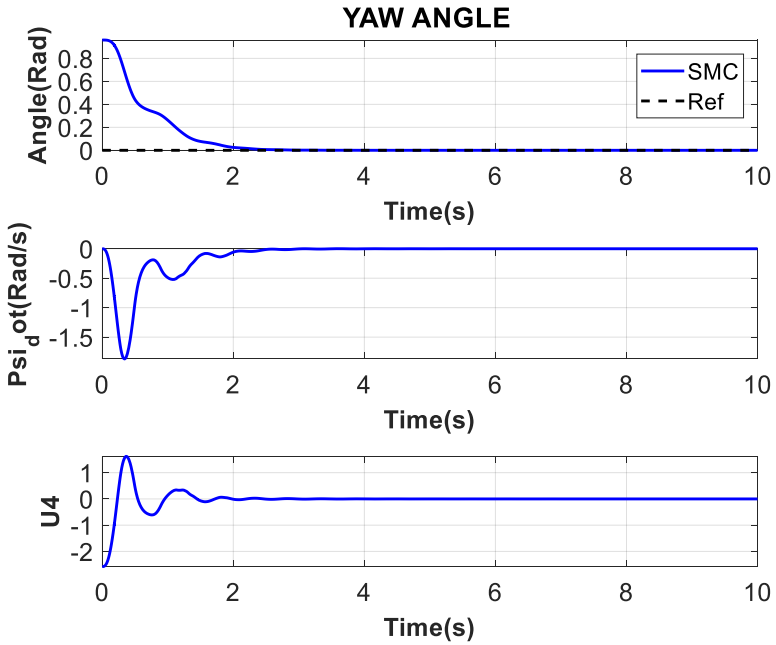

Figure 5.27 Yaw Angle Responses Using the Tuned Sliding Mode Controllers With High Initial Value Of 55 Degrees

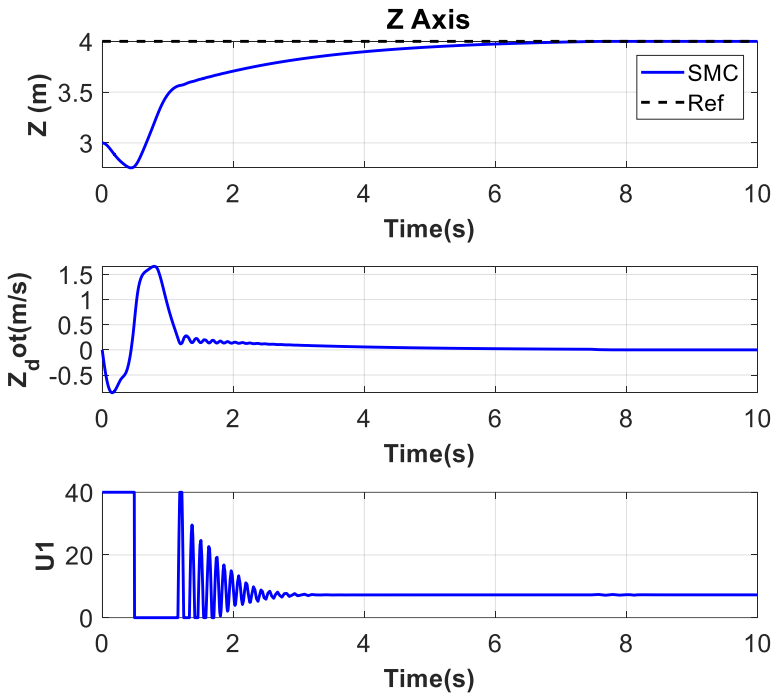

Figure 5.28 Altitude Angle Responses Using the Tuned Sliding Mode Controllers At High Initial Angle Values

The time response specification of quadrotor system using sliding mode controllers are given in tables 5.11, 5.12, 5.13, and 5.14.

Table 5.11 Time Response Specifications for the Roll Angle Using The Tuned Sliding Mode Controllers

Roll Angle (Φ)	Sliding Mode Controller
Settling time(s)	2.69
Rise time(s)	1.05
Overshoot (%)	0
Steady state error	Approx. 0

Table 5.12 Time Response Specifications for the Pitch Angle Using The Tuned Sliding Mode Controllers

Pitch Angle (θ)	Sliding Mode Controller
Settling time(s)	2.25
Rise time(s)	1.19
Overshoot (%)	0
Steady state error	Approx. 0

Table 5.13 Time Response Specifications for the Yaw Angle Using The Tuned Sliding Mode Controllers

Yaw Angle (ψ)	Sliding Mode Controller
Settling time(s)	1.93
Rise time(s)	1
Overshoot (%)	0
Steady state error	Approx. 0

Table 5.14 Time Response Specifications for the Altitude Using The Tuned Sliding Mode Controllers

Altitude Position (Z)	Sliding Mode Controller
Settling time(s)	5.61
Rise time(s)	3.37
Overshoot (%)	0
Steady state error	Approx. 0

The resulted sliding mode parameters from the tuning process using the previous tuning techniques are listed in table 5.15.

Table 5.15 Resulted Sliding Mode Controllers Parameters from the Tuning Processes

SMC Controller Parameters	Controllers parameters		
	c	K_1	K_2
Roll Angle	4.5	0.6	1.95
Pitch angle	2.2	12.09	4.5
Yaw angle	30.8	1.25	1.5
Altitude	600	8.5	0.5

The Sliding Mode Controller had been tested with system uncertainty and this is the result of the simulation:

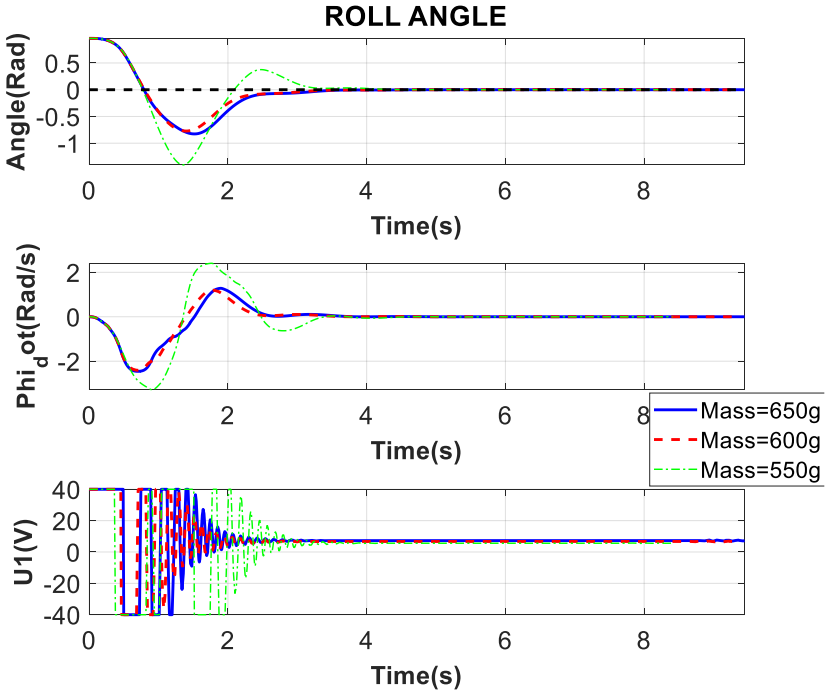

Figure 5.29 Roll Angle Responses Using the Tuned Sliding Mode Controllers With High Initial Value and The Case of Uncertainty

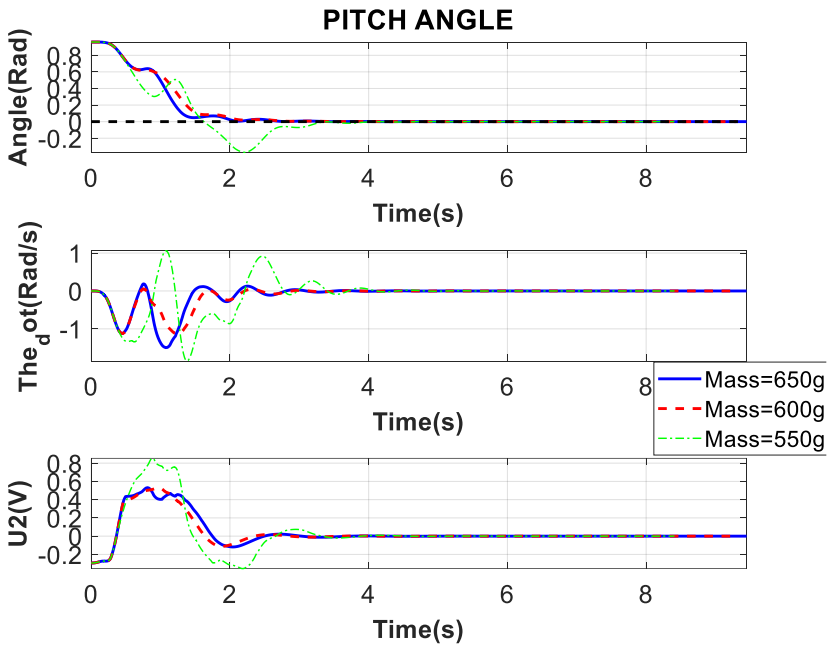

Figure 5.30 Pitch Angle Responses Using the Tuned Sliding Mode Controllers With High Initial Value and The Case of Uncertainty

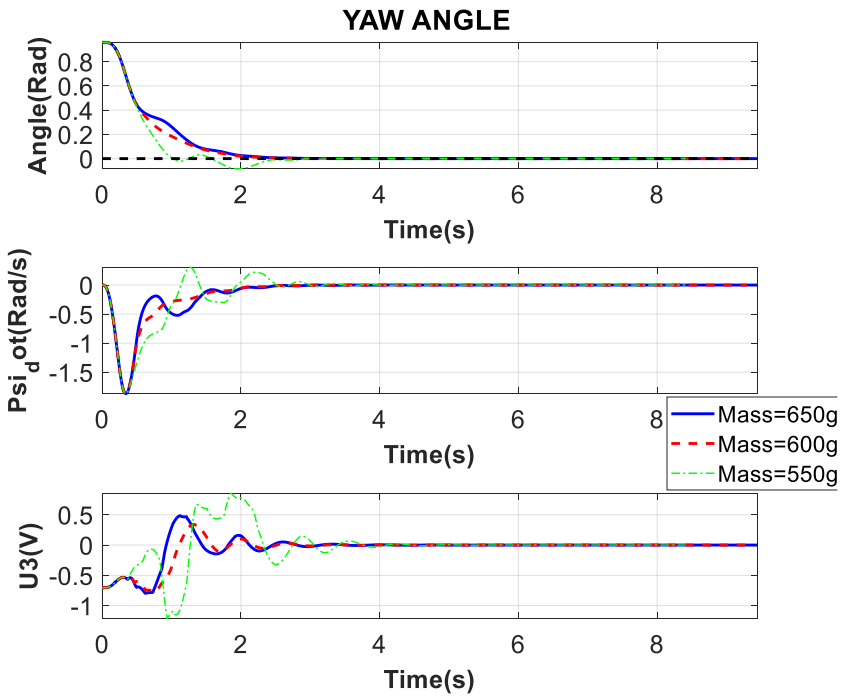

Figure 5.31 Yaw Angle Responses Using the Tuned Sliding Mode Controllers With High Initial Value and The Case of Uncertainty

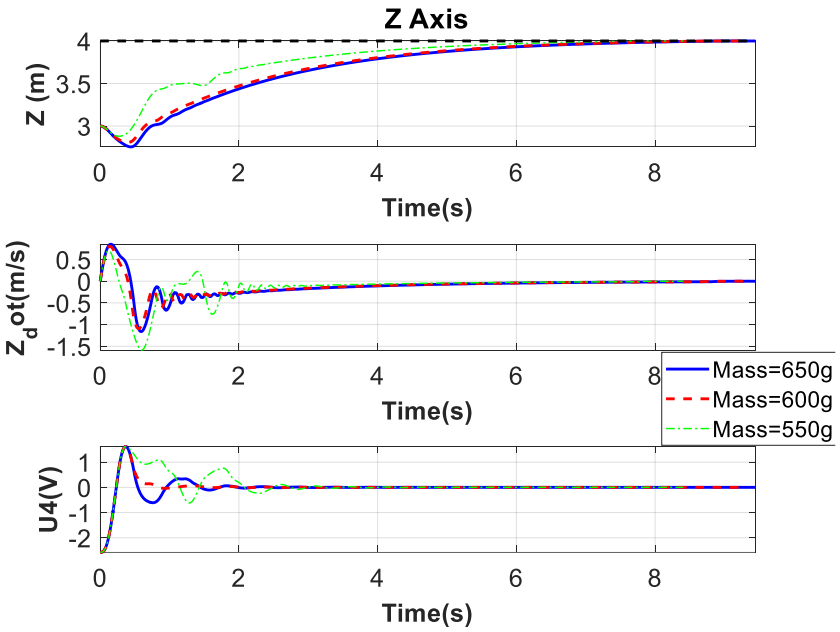

Figure 5.32 Altitude Angle Responses Using the Tuned Sliding Mode Controllers At High Initial Angle Values and The Case of Uncertainty

Results illustrate that Slide Mode control (SMC) method gives good response specifications with no overshoot and very small steady-state error. The suggested SMC controller gives better response to control the roll, yaw, pitch, and altitude with a lower settling time than the optimal PID controller, with the ability to handle the disturbances and the nonlinearity of the system.

5.4 Conclusion

In this thesis, kinematic and dynamic models for the quadrotor UAV are extracted and linear and nonlinear controllers are designed to balance and control the obtained quadrotor model. Results are used to compare and verify the performance of these controllers through the quadrotor's attitude, heading, and positions.

The mathematical model of the quadrotor UAV was extracted in detail based on Newton-Euler equations including the aerodynamic effects and rotor dynamics. A linear PID controller, which is designed and optimized using the suggested adaptive PSO algorithm, compared to a nonlinear Backstepping controller, and a nonlinear Sliding Mode Controller. The entire simulation is carried out on

MATLAB/Simulink environment depending on the extracted mathematical model of the quadrotor.

Results illustrate that the suggested adaptive PSO technique to tune the parameters of the linear controller give better response to control the roll, yaw, pitch, and altitude with lower settling time, overshoot, and has no steady-state error. The PID controller are designed to stabilize the quadrotor near the linear region that works at small roll, yaw, and pitch angles. The PID controller failed to stabilize the system outside the linear region. At the same time, the backstepping and sliding mode controllers were able to stabilize the quadrotor out the linear area because of its nonlinear properties and could handle the system uncertainty and external disturbances.

For future work, we advise to apply the backstepping and sliding mode controllers to the quadrotor system in the case of reality. In this thesis, it has presumed that all the extracted model parameters are determined precisely with no uncertainties opposite to the case in reality. So, it is required to develop an adaptive control algorithm to calculate the system uncertainties in order to enhance the performance of the quadrotor in a real operating environment. Besides, sensors have presumed to be ideal opposite to the case in reality. Thus, sensors modeling and disturbance effect must be considered.

Furthermore, these controllers needed to be developed and tested on physical platform considering all the indoor and outdoor conditions to figure out how much these controllers are robust against environment disturbance and effects, and to see how much it is practical to develop such controllers on embedded systems and to know what is the energy would be consumed and the capacity needed in the microcontroller that would be used for each control method.

References

[1] Krossblade Aerospace Systems "History of quadcopter and other multirotor." krossblade.com [online] available:http://krossblade.com/history-of-quadcopters-and-multirotors/ (accessed Oct. 1, 2019).

[2] Marshall, D. M., Barnhart, R. K., Hottman, S. B., Shappee, E., & Most, M. T. "*Introduction to unmanned aircraft systems*." Crc Press, 2016.

[3] Nonami, Kenzo. "Prospect and recent research & development for civil use autonomous unmanned aircraft as UAV and MAV." *Journal of system Design and Dynamics* 1, no. 2 (2007): 120-128.

[4] Shakhatreh, H., Sawalmeh, A. H., Al-Fuqaha, A., Dou, Z., Almaita, E., Khalil, I., ... & Guizani, M. "Unmanned aerial vehicles (UAVs): A survey on civil applications and key research challenges." *IEEE Access 7* (2019): 48572-48634.

[5] Gupta, Suraj G., Mangesh M. Ghonge, and P. M. Jawandhiya. "Review of unmanned aircraft system (UAS)." *International journal of advanced research in computer engineering & technology (IJARCET)* 2.4 (2013): 1646-1658.

[6] Gundlach, Jay. *Designing unmanned aircraft systems: a comprehensive approach*. American Institute of Aeronautics and Astronautics, 2012.

[7] Drone Omega "The Different Types of Drones Explained." Droneomega.com. [online] available: https://www.droneomega.com/types-of-drones/ (accessed Oct. 1, 2019).

[8] Krishna, Kowligi R. *Unmanned Aerial Vehicle Systems in Crop Production: A Compendium*. CRC Press, 2019.

[9] Gerdes, John, et al. "Robo Raven: a flapping-wing air vehicle with highly compliant and independently controlled wings." *Soft Robotics* 1.4 (2014): 275-288.

[10] Huang, Hui-Min. *Terminology for Specifying the Autonomy Levels for Unmanned Systems: Version 1.0*. No. Special Publication (NIST SP)-1011, 2004.

[11] Salih, Atheer L., et al.. "Flight PID controller design for a UAV quadrotor." *Scientific research and essays 5*, no. 23 (2010): 3660-3667.

[12] Erginer, Bora, and Erdinc Altug. "Modeling and PD control of a quadrotor VTOL vehicle." 2007 IEEE *Intelligent Vehicles Symposium*. IEEE, 2007.

[13] Bouabdallah, Samir, Andre Noth, and Roland Siegwart. "PID vs LQ control techniques applied to an indoor micro quadrotor." 2004 *IEEE/RSJ International Conference on Intelligent Robots and Systems* (IROS)(IEEE Cat. No. 04CH37566). Vol. 3. IEEE, 2004.

[14] Yang, Jinpeng, et al. "Self-tuning pid control design for quadrotor uav based on adaptive pole placement control." 2013 Chinese Automation Congress. IEEE, 2013.

[15] Milhim, Alaeddin, Youmin Zhang, and Camille-Alain Rabbath. "Gain scheduling based pid controller for fault tolerant control of quad-rotor uav." AIAA infotech@ aerospace 2010. p. 3530. 2010..

[16] Amoozgar, Mohammad Hadi, Abbas Chamseddine, and Youmin Zhang. "Fault-tolerant fuzzy gain-scheduled PID for a quadrotor helicopter testbed *in the presence of actuator faults*." IFAC Proceedings Volumes 45.3 (2012): 282-287.

[17] Reyes-Valeria, Elias, et al. "LQR control for a quadrotor using unit quaternions: Modeling and simulation." CONIELECOMP 2013, 23rd International Conference on Electronics, Communications and Computing. IEEE, 2013.

[18] Mokhtari, Abdellah, et al. "Feedback linearization and linear observer for a quadrotor unmanned aerial vehicle." *Advanced Robotics* 20.1 (2006): 71-91.

[19] Alexis, Kostas, George Nikolakopoulos, and Anthony Tzes. "Switching model predictive attitude control for a quadrotor helicopter subject to atmospheric disturbances." *Control Engineering Practice* 19.10 (2011): 1195-1207.

[20] Madani, Tarek, and Abdelaziz Benallegue. "Backstepping sliding mode control applied to a miniature quadrotor flying robot." IECON 2006-32nd Annual Conference on IEEE Industrial Electronics. IEEE, 2006.

[21] Fang, Zheng, and Weinan Gao. "Adaptive backstepping control of an indoor micro-quadrotor." *Research Journal of Applied Sciences, Engineering and Technology* 4.21 (2012): 4216-4226.

[22] Lee, Hyeonbeom, et al. "Backstepping control on se (3) of a micro quadrotor for stable trajectory tracking." 2013 IEEE International Conference on Systems, Man, and Cybernetics. IEEE, 2013.

[23] Zhen, Hongtao, Xiaohui Qi, and Hairui Dong. "An adaptive block backstepping controller for attitude stabilization of a quadrotor helicopter." *Trans. Syst. Control* 8.2 (2013): 46-55.

[24] Azzam, A., and Xinhua Wang. "Quad rotor arial robot dynamic modeling and configuration stabilization." 2010 2nd International Asia Conference on Informatics in Control, Automation and Robotics (CAR 2010). Vol. 1. IEEE, 2010.

[25] Nagaty, Amr, et al. "Control and navigation framework for quadrotor helicopters." *Journal of intelligent & robotic systems*70.1-4 (2013): 1-12.

[26] Efe, Mehmet Önder. "Neural Network Assisted Computationally Simple PID Control of a Quadrotor UAV." IEEE Transactions on Industrial Informatics7.2 (2011): 354-361.

[27] Santos, Matilde, Victoria Lopez, and Franciso Morata. "Intelligent fuzzy controller of a quadrotor." 2010 IEEE International Conference on Intelligent Systems and Knowledge Engineering. IEEE, 2010.

[28] Regula, Gergely. “Formation control of autonomous aerial vehicles.” Phd thesis, Budapest University of Technology and Economics, 2013.

[29] Eberhart, Russell, and James Kennedy. "A new optimizer using particle swarm theory." MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science. Ieee, 1995.

[30] Shi, Yuhui, and Russell Eberhart. "A modified particle swarm optimizer." 1998 IEEE international conference on evolutionary computation proceedings. IEEE world congress on computational intelligence (Cat. No. 98TH8360). IEEE, 1998.

[31] Bansal, Jagdish Chand, et al. "Inertia weight strategies in particle swarm optimization." 2011 Third world congress on nature and biologically inspired computing. IEEE, 2011.

[32] Berber, Ömer, et al. "Parçacık Sürü Optimizasyonu ve PID ile Mobil Robotun Optimum Yörünge Kontrolü." Kahramanmaras Sutcu Imam University Journal of Engineering Sciences 19.3 (2016): 165-169.

[33] ALHASAN, Hussein ALRUIM, and Mahit GÜNEŞ. "A New Adaptive Particle Swarm Optimization Based on Self-Tuning of PID Controller for DC Motor System." Çukurova University Journal of the Faculty of Engineering and Architecture 32.3 (2017): 243-249.

[34] Bolandi, Hossein, et al. "Attitude control of a quadrotor with optimized PID controller." *Intelligent Control and Automation* 4.03 (2013): 335.

[35] Salih, Atheer L., et al. "Modelling and PID controller design for a quadrotor unmanned air vehicle." 2010 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR). Vol. 1. IEEE, 2010.

[36] Madani, Tarek, and Abdelaziz Benallegue. "Backstepping control for a quadrotor helicopter." 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 2006.

Publications from the thesis

Contact Information: sayd.fahed@gmail.com

Conference Papers

1. Sayed, F. and Turker, T., (2019). “Optimal Tuning of PID Controller for Quadrotor System Using a new Adaptive Particle Swarm Optimization”. *International Symposium for Environmental Science and Engineering Research* ISESER2019, 25-27 May 2019, Konya.