THE AMERICAN UNIVERSITY IN CAIRO
الجامعة الأمريكية بالقاهرة

School of Sciences and Engineering

Dynamic Modeling and Control of a Quadrotor Using Linear and Nonlinear Approaches

A Thesis Submitted by

Heba talla Mohamed Nabil ElKholy

In partial fulfillment of the requirements for

The degree of Master of Science in
Robotics, Control and Smart Systems

Under the Supervision of

Prof. Maki K. Habib

Spring 2014

Dynamic Modeling and Control of a Quadrotor Using Linear and Nonlinear Approaches

by

Heba talla Mohamed Nabil ElKholy

Submitted to the School of Sciences and Engineering
on April 15, 2014, in partial fulfillment of the
requirements for the degree of
Master of Science in Robotics, Control and Smart Systems (RCSS)
Awarded from
The American University in Cairo (AUC)

Abstract

With the huge advancements in miniature sensors, actuators and processors depending mainly on the Micro and Nano-Electro-Mechanical-Systems (MEMS/NEMS), many researches are now focusing on developing miniature flying vehicles to be used in both research and commercial applications.

This thesis work presents a detailed mathematical model for a Vertical Takeoff and Landing (VTOL) type Unmanned Aerial Vehicle(UAV) known as the quadrotor. The nonlinear dynamic model of the quadrotor is formulated using the Newton-Euler method, the formulated model is detailed including aerodynamic effects and rotor dynamics that are omitted in many literature. The motion of the quadrotor can be divided into two subsystems; a rotational subsystem (attitude and heading) and a translational subsystem (altitude and x and y motion). Although the quadrotor is a 6 DOF underactuated system, the derived rotational subsystem is fully actuated, while the translational subsystem is underactuated.

The derivation of the mathematical model is followed by the development of four control approaches to control the altitude, attitude, heading and position of the quadrotor in space. The first approach is based on the linear Proportional-Derivative-Integral (PID) controller. The second control approach is based on the nonlinear Sliding Mode Controller (SMC). The third developed controller is a nonlinear Backstepping controller while the fourth is a Gain Scheduling based PID controller.

The parameters and gains of the forementioned controllers were tuned using Genetic Algorithm (GA) technique to improve the systems dynamic response. Simulation based experiments were conducted to evaluate and compare the performance of the four developed control techniques in terms of dynamic performance, stability and the effect of possible disturbances.

Keywords: Quadrotor, Quadcopter, UAV, VTOL, Nonlinear Control, PID, Sliding Mode Control (SMC), Backstepping, Gain Scheduling, Genetic Algorithm (GA).

Declaration

I hereby declare that this thesis represents my original work and all used references are properly indicated and cited.

Heba talla Mohamed Nabil ElKholy

Acknowledgments

I would like to thank my supervisor Prof. Maki K. Habib for all his help, support and guidance.

I would like to thank my mom and dad for their continuous support, for always pushing me forward and for providing me with the calm environment to work in.

This work is dedicated to my fiancé Sherif, without your encouragement, I would not have been here today. Thank you for always believing in me and pushing me forward.

My good friend Amr Nagaty, words cannot express how thankful I am for all what you have done.

Last but not least, my best friends Ingee, Marium, Mohamed Shalaby, Marwa, Sara, Mary, Omar Shehata and Tarek, thank you for always being there for me and encouraging me through the tough times.

List of Figures

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

Acronyms

DAST Drones for Aerodynamic and Structural Testing

DOF Degrees of Freedom

ERAST Environmental Research Aircraft and Sensor Technology Project

GA Genetic Algorithm

HiMAT Highly Maneuverable Aircraft Technology

IMU Inertial Measurement Unit

LQ Linear Quadratic

MEMS Micro-Electo-Mechanical Systems

MPC Model Predictive Controller

PD Proportional-Derivative

PID Proportional-Integral-Derivative

PWM Pulse Width Modulated

RC Radio Controlled

SMC Sliding Mode Controller

UAV Unmanned Aerial Vehicle

VSC Variable Structure Control

WWI World War I

List of Symbols

A blade area

C_D aerodynamic drag coefficient

C_T aerodynamic thrust coefficient

F_B nongravitational forces acting on the quadrotor

I Identity matrix

I_{xx} Area moment of inertia about the x-axis

I_{yy} Area moment of inertia about the y-axis

I_{zz} Area moment of inertia about 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 circuit inductance

M_B moments acting on the quadrotor in the body frame

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 rotors' relative speed

$\dot{\eta}$ Euler rates

ω Angular body rates

ϕ roll angle

ϕ_d Desired roll ϕ

ψ yaw angle

ψ_d Desired yaw ψ

ρ air density

θ pitch angle

θ_d Desired pitch θ

$chat$ Chattering value

e Error

g gravitational acceleration g =$9.81m/s^2$

k_d Derivative gain

k_i Integral gain

k_p Proportional gain

l moment arm

m mass of quadrotor

r Quadrotor's position in the navigation frame

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 altitude z

THIS PAGE INTENTIONALLY LEFT BLANK

Contents

Chapter 1

Introduction

This chapter will first discuss the objectives and motivation behind writing this thesis. A brief introduction about Unmanned Aerial Vehicles (UAVs), their history, types and uses will then be presented. We will then move to the quadrotor type UAVs and discuss their concept and architecture. Lastly, the structure of the thesis will be outlined.

1.1 Objectives and Motivation

This thesis work will focus on the modeling and control of a quadrotor type UAV. The reason for choosing the quadrotor is in addition to its advantages that will be addressed later, the research field is still facing some challenges in the control field because the quadrotor is a highly nonlinear, multivariable system and since it has six Degrees of Freedom (DOF) but only four actuators, it is an underactuated system [1]. Underactuated systems are those having a less number of control inputs compared to the system's degrees of freedom. They are very difficult to control due to the nonlinear coupling between the actuators and the degrees of freedom [2]. Although the most common flight control algorithms found in literature are linear flight controllers, these controllers can only perform when the quadrotor is flying around hover, they suffer from a huge performance degradation whenever the quadrotor leaves the nominal conditions or performs aggressive maneuvers [3].

The contributions of this work are: deriving an accurate and detailed mathematical model of the quadrotor UAV, developing linear and nonlinear control algorithms and applying those on the derived mathematical model in computer based simulations. The thesis will be concluded with a comparison between the developed control algorithms in terms of their dynamic performance and their ability to stabilize the system under the effect of possible disturbances.

1.2 Unmanned Aerial Vehicles

The definition for UAVs varies from one literature to the other. For our purposes, UAVs are small aircrafts that are flown without a pilot. They can either be remotely operated by a human or they can be autonomous; autonomous vehicles are controlled by an onboard computer that can be preprogrammed to perform a specific task or a broad set of tasks. While in other literatures, UAVs may refer to powered or unpowered, tethered or untethered aerial vehicles [4]. The definition used in this thesis is based on that of the American Institute of Aeronautics and Astronautics [4]:

> An aircraft which is designed or modified not to carry a human pilot and is operated through electronic input initiated by the flight controller or by an on board autonomous flight management control system that does not require flight controller intervention.

UAVs were mainly used in military application but recently they are being deployed in civil applications too [5].

1.2.1 History of UAVs

UAVs were first manufactured by Lawrence and Sperry (USA) in the year 1916. They called it the Aviation Torpedo shown in Figure 1-1 and they were able to fly it for a distance of 30 miles. It was reported that Lawrence and Sperry used a gyroscope to balance the body [5].

Figure 1-1: Lawrence and Sperry UAV [5]

1.2.1.1 Military History

A great interest was shown by the USA to develop UAVs to be used in the World War I (WWI) and two projects where funded. The first was by Elmer Sperry to develop the "Flying Bomb" UAV and the second project was the "Kettering Bug" manufactured by General Motors. Both projects were cancelled and the funding stopped as they proved unsuccessful. This is due to the fact of the absence of the required technological advances in the fields of guidance systems and engines [6]. Development of UAVs started increasing tremendously by the end of the 1950s, the USA deployed them during the Vietnam War to decrease the casualties in pilots when flying over hostile territories. After their success, the USA and Israel decided to invest more to build smaller and cheaper UAVs, they used small motors like those found in motorcycles to result in smaller sized and lighter UAVs. In addition, a video camera was added on the UAVs to transmit images to the ground operator. In 1991, the USA used UAVs extensively in the Gulf War, and the most famous model was the Predator shown in Figure 1-2. UAVs were intensively used by the USA in many conflicts and wars in the late 1990's and early 2000's and later on, UAVs were used extensively in the war against Iraq [7].

Figure 1-2: Predator Military UAV [5]

1.2.1.2 Civil History

The uses of UAVs were not only confined to military use; in 1969, NASA grew a concern to automatically control an aircraft, the first trials was the PA-30 program. The program was successful but they had a pilot onboard to take over the control of the aircraft in case anything went wrong. Other research programs followed the success of the PA-30 program like: Drones for Aerodynamic and Structural Testing (DAST) and Highly Maneuverable Aircraft Technology (HiMAT) programs [8]. Following that era, in the 1990's NASA then partnered with industrial companies to develop a nine-year long research project called the Environmental Research Aircraft and Sensor Technology Project (ERAST). They developed several UAVs models that were able to fly for altitudes up to 30 Km and endured flights up to 6 months. The resulting UAV models included the: Pathfinder, Helios, Atlus and Perseus B. The developed UAVs carried several sensors to carry out environmental measurements, the onboard sensors included a camera, a Digital Array Scanned Interferometer (DASI) and an active detect, see and avoid (DSA) system [8].

1.2.2 Applications of UAVs

In addition to the military use, UAVs can be used in many civil or commercial applications that are too dull, too dirty or too dangerous for manned aircrafts. These uses include but not limited to:

Earth Science observations from UAVs can be used side-to-side with that acquired from satellites. Such missions include [8]:

(a) Measuring deformations in the Earth's crust that may be indications to natural disasters like earthquakes, landslides or volcanos [8].

(b) Cloud and Aerosol Measurements [8].

(c) Tropospheric pollution and air quality measurements to determine the pollution sources and how plumes of pollution are transported from one place to another [8, 9]..

(d) Ice sheet thickness and surface deformation for studying global warming [8].

(e) Gravitational acceleration measurements, since the gravitational acceleration varies near Earth, UAVs are used to accurately measure gravitational acceleration at multiple places to define correct references [8].

(f) River discharge is measured from the volume of water flowing in a river at multiple points. This will help in global and regional water balance studies [8].

Search and rescue UAVs equipped with cameras are used to search for survivors after natural disasters like earthquakes and hurricanes or survivors from shipwrecks and aircraft crashes [9, 10].

Wild fire suppression UAVs equipped with infrared sensors are sent to fly over forests prone to fires in order to detect it in time and send a warning back to the ground station with the exact location of the fire before it spreads [8–10].

Law enforcement UAVs are used as a cost efficient replacement of the traditional manned police helicopters [10].

Border surveillance UAVs are used to patrol borders for any intruders, illegal immigrants or drug and weapon smuggling [5, 8, 10].

Research UAVs are also used in research conducted in universities to proof certain theories. Also, UAVs equipped with appropriate sensors are used by environmental research institutions to monitor certain environmental phenomena like pollution over large cities [10].

Industrial applications UAVs are used in various industrial applications such as pipeline inspection or surveillance and nuclear factories surveillance [9, 10].

Agriculture development UAVs also have agriculture uses such as crops spraying [5, 9, 10].

1.2.3 Classification of UAVs

There are different ways to classify UAVs, either according to their range of action, aerodynamic configuration, size and payload or according to their levels of autonomy.

1.2.3.1 Range of Action Classification

UAVs can be classified into 7 different categories based on their maximum altitude and endurance as follows [9]:

(a) **High-Altitude Long-Endurance (HALE):** they can fly over 15000 m high with an endurance of more than 24 hr. They are mainly used for long-range surveillance missions.

(b) **Medium-Altitude Long-Endurance (MALE):** they can fly between 5000-15000 m of altitude for a maximum of 24 hr. MALE UAVs are also used for surveillance missions.

(c) **Medium-Range or Tactical UAV (TUAV):** They can fly between 100 and 300 km of altitude. They are smaller and operated with simpler systems that their HALE and MALE counterparts.

(d) **Close Range UAV:** They have an operation range of 100 km. They are mainly used in the civil application such as powerline inspection, crop-spraying, traffic monitoring, homeland security, etc..

(e) **Mini UAV (MUAV):** They have a weight of about 20 kg and an operating range of about 30 km.

(f) **Micro UAV (MAV):** They have a maximum wingspan of 150 mm. They are mainly used indoors where they are required to fly slowly and hover

(g) **Nano Air Vehicles (NAV):** They have a small size of about 10 mm. they are mainly used in swarms for applications such as radar confusion. They are also used for short range surveillance if equipped with an equally small camera.

1.2.3.2 Aerodynamic Configuration Classification

UAVs can be classified into four main categories based on their aerodynamic configuration as follows [9]:

(a) **Fixed-wing UAVs:** require a run-way to take-off and land. They can fly for a long time and at high cruising speeds. They are mainly used in scientific applications such as meteorological reconnaissance and environmental monitoring, shown in Figure 1-3.

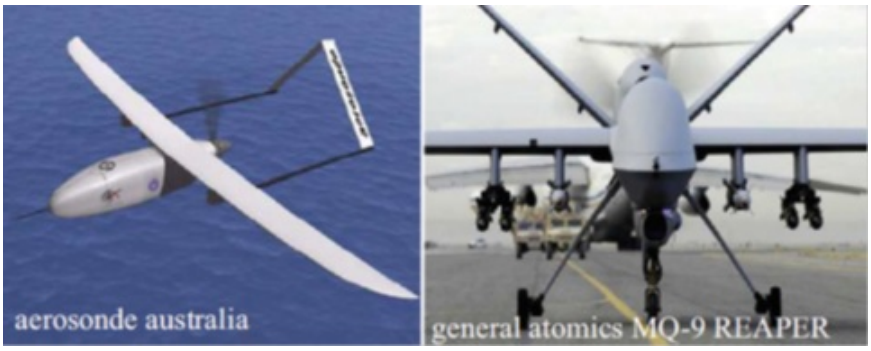


Figure 1-3: Fixed-Wing UAVs [5]

(a) Single Rotor

(b) Coaxial

(c) Quadrotor

(d) Multi-Rotor

Figure 1-4: Rotary-Wing UAVs [4]

(b) **Rotary-wing UAVs:** they can take off and land vertically. They can also hover and fly with high maneuverability. The Rotary-wing UAVs can be further classified into four groups [4]:

(i) **Single-rotor:** they have a main rotor on top and another rotor at the tail for stability, same like the helicopter configuration. Shown in Figure 1-4(a).

(ii) **Coaxial:** they have two rotors rotating in opposite directions mounted to the same shaft. Shown in Figure 1-4(b).

(iii) **Quadrotor:** they have four rotors fitted in a cross-like configuration. Shown in Figure 1-4(c).

(iv) **Multi-rotor:** UAVs with six or eight rotors. They are agile type and fly even when a motor fails, as there is redundancy due to the large number of rotors. Shown in Figure 1-4(d).

Increasing the number of rotors in turn increases the payload and maximum altitude of the UAVs but it comes at the cost of increasing the size and power consumption.

(c) **Blimps UAVs:** which may look like balloons or airships, they ensure lifting by

their helium-filled body. They are very light and have a large size. They can fly for a long time and at low speeds, shown in Figure 1-5.

(d) **Flapping-wing UAVs:** they are inspired from birds and flying insects. These UAVs have small wings and have an extremely low payload and endurance. On the other hand, they have low power consumption and can perform vertical take-off and landing. This class of UAVs is still under development, shown in Figure 1-6 [5].

Figure 1-5: Blimps UAVs [5]

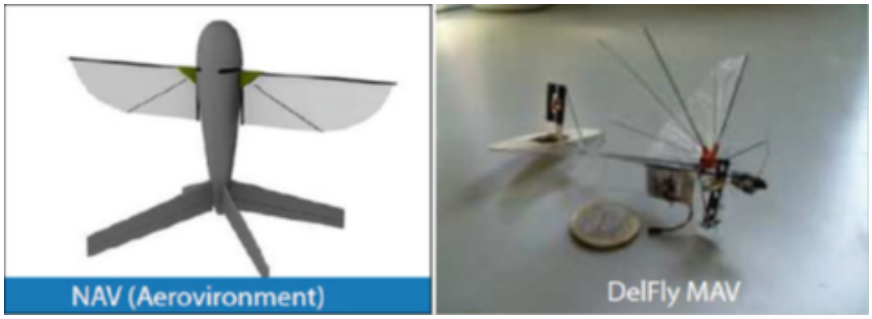


Figure 1-6: Flapping Wing UAVs [5]

1.2.3.3 Size and Payload Classification

UAVs can be classified into five main classes based on their size and payload. Figure 1-7 shows examples of these five classes as described below [3]:

(a) **Full-Scale UAVs:** these are normal sized vehicles, shown in Figure 1-7(a). Although a pilot can be present on board, the vehicle is capable of autonomous

flying. The purpose of the pilot is to backup when trying complex flight tests and maneuvers. Also, these vehicles have the highest payload and endurance.

(b) **Medium-scale UAVs:** these are the vehicles mainly used for security missions. They have a relatively high payload of 10 kg enabling having heavy and high-quality navigation sensors on board, thus achieving dependant flights. Figure 1-7(b) shows an example of the Medium Scale UAV used in the US army.

(c) **Small-scale UAVs:** these are mainly UAVs based on Radio Controlled (RC) toys, they have a lower payload of the range 2 to 10 kg, which enables them to carry adequate quality navigation sensors. An example of a small-scale UAV is shown in Figure 1-7(c).

(d) **Mini UAVs:** portable UAVs that are able to fly indoors and outdoors with a payload of less than 2 kg which is sufficient to carry small lightweight sensors. The small size, low cost and ease of maintenance of these UAVs makes them the most common test- bed UAVs in research applications. An example of a mini UAV is shown in Figure 1-7(d).

(e) **Micro UAVs:** mainly used indoors due to their very small size. They have a payload of less than 100 g which makes it very hard to be equipped with navigation and guidance sensors. The research challenge regarding these micro UAVs is to design light-weight navigation and guidance sensors. An example of a micro UAV is shown in Figure 1-7(e).

1.2.3.4 Levels of Autonomy Classification

UAVs can be also classified according to their level of autonomy. The National Institute of Standards and Technology published a framework that can be used to classify UAVs according to their autonomy level which is defined by three metrics namely: Human Independence (HI), Mission Complexity (MC) and Environmental Complexity (EC) [3, 4]. The framework proposes five levels of autonomy [4]:

(a) Full-Scale UAV

(b) Medium-Scale UAVs

(c) Small-Scale UAVs

(d) Mini UAV

(e) Micro UAV

Figure 1-7: Size Classification [3]

(a) **Level 1:** Full human interaction is needed to operate these UAVs as they are purely remotely controlled. Level 1 UAVs are mainly used for the least complex missions.

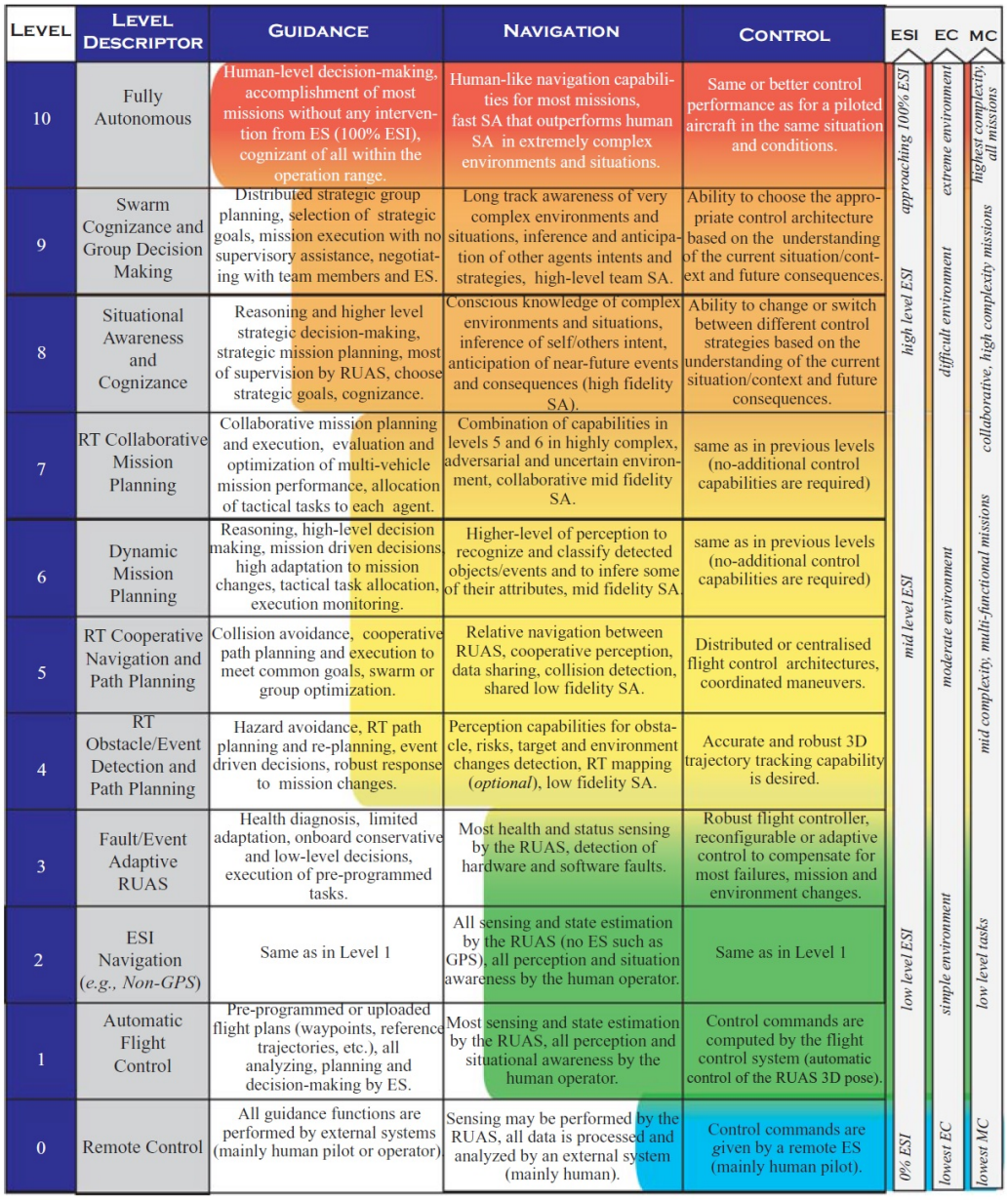
(b) **Level 2:** Still require human interaction but can perform more complex missions than level 1 UAVs.

(c) **Level 3:** Moderate level of human interaction with moderate mission complexity.

(d) **Level 4:** Minimal human interaction, used in missions where the environment is complex and dynamic and the reaction time of human operator may not be sufficient to correctly navigate the UAV.

(e) **Level 5:** Zero human interaction, used to carry out missions in the most complex environments. Literature claims that level 5 UAVs do not currently exist and they serve as a goal for future research.

Although the National Institute of Standards and Technology proposed a five levels classification framework that can be validly applied to UAVs, Kendoul finds that framework is insufficient for a proper classification of current UAV systems and proposed a new eleven-level framework called Autonomy Levels for Unmanned Rotorcraft Systems [3]. In his new framework, Kendoul used six metrics to classify UAVs which are: External System Independence (ESI), Environment Complexity (EC), Mission Complexity (MC), External System (ES), Situational Awareness (SA) and Real Time (RT) [4]. The framework is shown in Figure 1-8.



Level	Level Descriptor	Guidance	Navigation	Control	ESI	EC	MC
10	Fully Autonomous	Human-level decision-making, accomplishment of most missions without any intervention from ES (100% ESI), cognizant of all within the operation range.	Human-like navigation capabilities for most missions, fast SA that outperforms human SA in extremely complex environments and situations.	Same or better control performance as for a piloted aircraft in the same situation and conditions.	*approaching 100% ESI*	*extreme environment*	*highest complexity, all missions*
9	Swarm Cognizance and Group Decision Making	Distributed strategic group planning, selection of strategic goals, mission execution with no supervisory assistance, negotiating with team members and ES.	Long track awareness of very complex environments and situations, inference and anticipation of other agents intents and strategies, high-level team SA.	Ability to choose the appropriate control architecture based on the understanding of the current situation/context and future consequences.	*high level ESI*	*difficult environment*	*collaborative, high complexity missions*
8	Situational Awareness and Cognizance	Reasoning and higher level strategic decision-making, strategic mission planning, most of supervision by RUAS, choose strategic goals, cognizance.	Conscious knowledge of complex environments and situations, inference of self/others intent, anticipation of near-future events and consequences (high fidelity SA).	Ability to change or switch between different control strategies based on the understanding of the current situation/context and future consequences.			
7	RT Collaborative Mission Planning	Collaborative mission planning and execution, evaluation and optimization of multi-vehicle mission performance, allocation of tactical tasks to each agent.	Combination of capabilities in levels 5 and 6 in highly complex, adversarial and uncertain environment, collaborative mid fidelity SA.	same as in previous levels (no-additional control capabilities are required)			
6	Dynamic Mission Planning	Reasoning, high-level decision making, mission driven decisions, high adaptation to mission changes, tactical task allocation, execution monitoring.	Higher-level of perception to recognize and classify detected objects/events and to infere some of their attributes, mid fidelity SA	same as in previous levels (no-additional control capabilities are required)	*mid level ESI*	*moderate environment*	*mid complexity, multi-functional missions*
5	RT Cooperative Navigation and Path Planning	Collision avoidance, cooperative path planning and execution to meet common goals, swarm or group optimization.	Relative navigation between RUAS, cooperative perception, data sharing, collision detection, shared low fidelity SA.	Distributed or centralised flight control architectures, coordinated maneuvers.			
4	RT Obstacle/Event Detection and Path Planning	Hazard avoidance, RT path planning and re-planning, event driven decisions, robust response to mission changes.	Perception capabilities for obstacle, risks, target and environment changes detection, RT mapping (*optional*), low fidelity SA.	Accurate and robust 3D trajectory tracking capability is desired.			
3	Fault/Event Adaptive RUAS	Health diagnosis, limited adaptation, onboard conservative and low-level decisions, execution of pre-programmed tasks.	Most health and status sensing by the RUAS, detection of hardware and software faults.	Robust flight controller, reconfigurable or adaptive control to compensate for most failures, mission and environment changes.			
2	ESI Navigation (*e.g., Non-GPS*)	Same as in Level 1	All sensing and state estimation by the RUAS (no ES such as GPS), all perception and situation awareness by the human operator.	Same as in Level 1	*low level ESI*	*simple environment*	*low level tasks*
1	Automatic Flight Control	Pre-programmed or uploaded flight plans (waypoints, reference trajectories, etc.), all analyzing, planning and decision-making by ES.	Most sensing and state estimation by the RUAS, all perception and situational awareness by the human operator.	Control commands are computed by the flight control system (automatic control of the RUAS 3D pose).			
0	Remote Control	All guidance functions are performed by external systems (mainly human pilot or operator)	Sensing may be performed by the RUAS, all data is processed and analyzed by an external system (mainly human).	Control commands are given by a remote ES (mainly human pilot).	*0% ESI*	*lowest EC*	*lowest MC*

Figure 1-8: Kendoul's 11-level Framework [3]

1.3 Quadrotors

The quadrotor concept for aerial vehicles was developed a long time ago. It was reported that the Breguet-Richet quadrotor built in 1907 had actually flown. A quadrotor is considered to be a rotary-wing UAV due to its configuration that will be discussed later.

1.3.1 The Quadrotor Concept

A quadrotor consists of four rotors, each fitted in one end of a cross-like structure as shown in Figure 1-9. Each rotor consists of a propeller fitted to a separately powered DC motor. Propellers 1 and 3 rotate in the same direction while propellers 2 and 4 rotate in an opposite direction leading to balancing the total system torque and cancelling the gyroscopic and aerodynamics torques in stationary flights [1, 11].

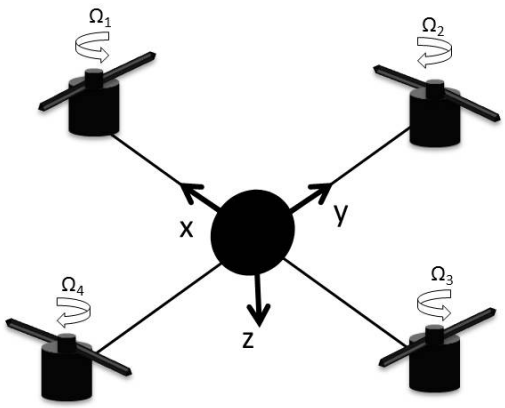


Figure 1-9: Quadrotor Configuration

The quadrotor is a 6 DOF object, thus 6 variables are used to express its position in space (x, y, z, ϕ, θ and ψ). x, y and z represent the distances of the quadrotor's center of mass along the x,y and z axes respectively from an Earth fixed inertial frame. ϕ, θ and ψ are the three Euler angles representing the orientation of the quadrotor. ϕ is called the roll angle which is the angle about the x-axis, θ is the pitch angle about the y-axis, while ψ is the yaw angle about the z-axis. Figure 1-10 clearly explains the Euler Angles. The roll and pitch angles are usually called the attitude of the quadrotor, while the yaw angle is referred to as the heading of the quadrotor. For

the linear motion, the distance from the ground is referred to as the altitude and the x and y position in space is often called the position of the quadrotor.

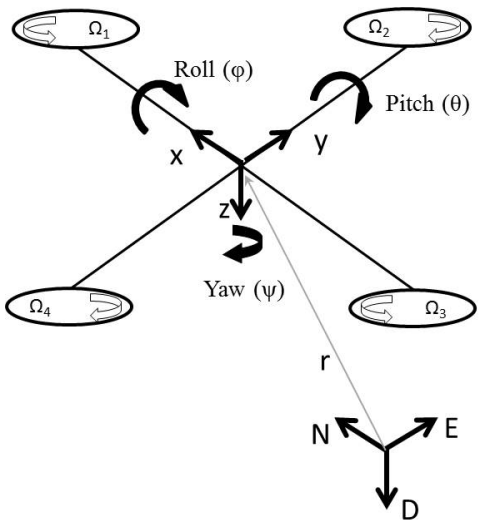


Figure 1-10: Euler Angles for a Quadrotor UAV

To generate vertical upwards motion, the speed of the four propellers is increased together whereas the speed is decreased to generate vertical downwards motion. To produce roll rotation coupled with motion along the y-axis, the second and fourth propellers speeds are changed while for the pitch rotation coupled with motion along the x-axis, it is the first and third propellers speeds that need to be changed.

One problem with the quadrotor configuration is that to produce yaw rotation, one need to have a difference in the opposite torque produced by each propeller pair. For instance, for a positive yaw rotation, the speed of the two clockwise turning rotors need to be increased while the speed of the two counterclockwise turning rotors need to be decreased [11, 12]. Figure 1-11 shows how different movements can be produced, note that a thicker arrow means a higher propeller speed.

1.3.2 Advantages and Drawbacks of Quadrotors

Some advantages of the quadrotor over helicopters is that the rotor mechanics are simplified as it depends on four fixed pitch rotors unlike the variable pitch rotor in the helicopter, thus leading to easier manufacturing and maintenance. Moreover, due

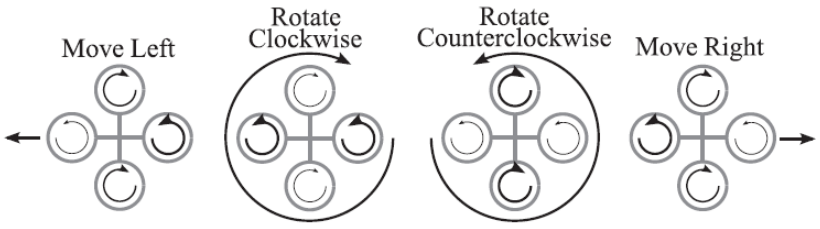


Figure 1-11: Generated Motion of the Quadrotor [13]

to the symmetry in the configuration, the gyroscopic effects are reduced leading to simpler control.

Stationary hovering can be more stable in quadrotors than in helicopters due to the presence of four propellers providing four thrust forces shifted a fixed distance from the center of gravity instead of only one propeller centered in the middle as in the helicopters structure [1].

More advantages are the vertical take-off and landing capabilities, better maneuverability and smaller size due to the absence of a tail [14], these capabilities make quadrotors useful in small area monitoring and buildings exploration [15].

Moreover, quadrotors have higher payload capacities due to the presence of four motors thus providing higher thrust [1].

On the other hand, quadrotors consume a lot of energy due to the presence of four separate propellers [15]. Also, they have a large size and heavier than some of their counterparts again to the fact that there is four separate propellers [15, 16].

1.4 Thesis Structure

This thesis is organized as follows, Chapter 2 presents a deep literature review of the commonly used control algorithms in the quadrotors field together with some of the famous hardware platforms. Chapter 3 presents the mathematical modeling of a quadrotor UAV based on the Newton-Euler formalism in full details including the rotor dynamics and aerodynamic effects acting on the quadrotor body. Chapter 4 shows four developed control techniques to control and stabilize the attitude, heading,

altitude and position of the quadrotor in space. The controllers are verified using computer simulations and the results of these simulations are shown. Chapter 5 presents the discussion of the results acquired in the previous chapter and finally Chapter 6 shows the conclusion and future work for this thesis.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 2

State of the Art

Since the advances in technologies and the ability to manufacture miniature sensors and controllers using the Micro-Electo-Mechanical Systems (MEMS) technologies, there have been a lot of advances in the UAVs area. A lot of the research conducted focused on the quadrotor due to its previously mentioned advantages of easier manufacturing, compactness and maneuverability among others. Some literature focused only on developing a control algorithm to be applied in a simulation environment while others developed quadrotors models to test their proposed flight algorithm on. This chapter discusses some of the most commonly used control techniques and some of the hardware platforms used in research.

2.1 Control

There are several control techniques that can be used to control a quadrotor varying between the classical linear Proportional-Integral-Derivative (PID) or Proportional-Derivative (PD) controller to more complex nonlinear schemes as backstepping or sliding-mode controllers. The flight control systems can be classified into four main categories which are: linear fight control systems, non-linear flight control systems, hybrid and learning-based flight control systems [3]. It was found that the most common control technique used is the PID or PD controller. Although it is a linear controller used for the nonlinear multivariable quadrotor system, it was proven

successful in many literature [17].

2.1.1 Linear Flight Control Systems

They are the most common and conventional flight control systems, typically based on PID, Linear Quadratic (LQ) or H_∞ algorithms. It was reported that in the late 1960's, a full scale helicopter achieved autonomous waypoint navigation using a classical linear control technique [3].

PID and LQ Bouabdallah et al. proposed the usage of PID and LQ control techniques to be applied on an indoor micro quadrotor, it was found out that these two types of controllers performed comparably and were able to stabilize the quadrotor's attitude around its hover position when it undergoes little disturbances [11, 17]. Li and Li used the classical PID to control the position and orientation of a quadrotor and it was able to stabilize in a low speed wind environment [14]. Yang et al. used a self tuning PID controller based on adaptive pole placement to control the attitude and heading of a quadrotor. Simulation showed that the proposed controller performed well with online tuning of the parameters [18].

H_∞ Raffo et al. used an H_∞ controller to stabilize the rotational angles together with a Model Predictive Controller (MPC) to track the desired position [19]. The effect of wind and model uncertainties was added to the simulated model and it performed robustly with a zero steady-state error. H_∞ is a linear robust controller; robust controllers are those taking into account parametric uncertainty and unmodeled dynamics. It is reported that it is used for control of full-scaled helicopters [3].

Switched Dynamics and Gain Scheduling To use a linear controller to control a nonlinear system like a quadrotor, the nonlinearity of the system can be modelled as a collection of simplified linear models. This is the concept of gain scheduling and it is commonly used to design flight controllers. Gillula et al. divided the state space

model of a STARMAC quadrotor to a set of simple hybrid modes and this approach enabled the quadrotor to carry out aerobatic maneuvers [3, 20]. Also, Ataka et al. used gain scheduling on a linearized model of the quadrotor around some equilibrium points and tested the controllability and observability of the resulting system [21]. Amoozgar et al. used a gain scheduled PID controller with the later's parameters tuned using a fuzzy logic inference scheme to control a quadrotor. The system was tested under actuator fault conditions and compared with the performance of the conventional PID controller. The results showed a better performance for the gain scheduled PID controller [22]. Sadeghzadeh et al. also use a gain scheduled PID controller applied to a quadrotor dropping a carried payload at a designated time. The algorithm was able to stabilize the system during the dropping operation [23].

2.1.2 Nonlinear Flight Control Systems

Due to the fact that the dynamics of the quadrotor is of a nonlinear nature, developing nonlinear control algorithms to be used as flight controllers was necessary. There is a variety of nonlinear control algorithms applied to quadrotors including: feedback linearization, model predictive control, backstepping and sliding-mode.

Backstepping and Sliding-mode Backstepping is a recursive control algorithm that can be applied to both linear and nonlinear systems [3]. In a more recent paper, Bouabdallah and Siegwart proposed the use of backstepping and sliding-mode nonlinear control methods to control the quadrotor which gave better performance in the presence of disturbances. [15]. Waslander et al. proposed developing controllers that can stabilize the quadrotor in an outdoors environment, they compared the performance of an integral sliding-mode controller vs. a reinforcement learning controller. They reached a conclusion that both controllers were able to stabilize the quadrotor outdoors with an improved performance over classical control techniques [24]. Madani and Benallegue used a backstepping controller based on Lyapunov stability theory to track desired values for the quadrotor's position and orientation. They divided the quadrotor model into 3 subsystems: underactuated, fully-actuated and propeller

subsystems. Their proposed algorithm was able to stabilize the system under no disturbances [25]. Fang and Gao proposed merging a backstepping controller with an adaptive controller to overcome the problems of model uncertainties and external disturbances. The proposed adaptive integral backstepping algorithm was able to reduce the system's overshoot and response time and eliminate steady state error [26]. Lee et al. used a backstepping controller to control the position and attitude of a quadrotor, the proposed controller was tested in a noisy environment and gave a satisfactory performance [27]. Zhen et al. combined a backstepping controller with an adaptive algorithm to control the attitude of a quadrotor. A robust adaptive function is used to approximate the external disturbances and modeling errors of the system. Simulations showed the success of the proposed controller in overcoming disturbances and uncertainties [28]. González et al. proposed using a chattering free sliding mode controller to control the altitude of a quadrotor. The proposed controller performed well in both simulations and on a real system in the presence of disturbances [29].

Feedback Linearization Feedback linearization is a control techniques that uses a nonlinear transformation between the system's nonlinear state variables to linear ones. Linear algorithms can be then used to stabilize the transformed linear system which will then be inversely transformed back into the original state variables. Kendoul et al. was able to control a quadrotor in several flight tests based on the concept of feedback linearization [3, 30].

Model Predictive Control (MPC) Model predictive control relies on predicting the future states of the system and tracking the error to give an improved performance [3]. Alexis et al. relied on a MPC to control the attitude of a quadrotor in the presence of atmospheric disturbances. The proposed algorithm behaved well in performing rough maneuvers in a wind induced environment as was able to accurately track the desired attitude [31]. Sadeghzadeh et al. also used a MPC applied to a quadrotor in dropping a carried payload, the MPC was able to stabilize the system with a promising performance [23].

2.1.3 Learning-Based Flight Control Systems

Opposing the previous control techniques, learning based flight control systems do not need a precise and accurate dynamic model of the system to be controlled. On the other hand, several trials are carried out and flight data are used to "train" the system. There are many types of learning-based flight control systems, the most widely used are: neural networks, fuzzy logic and human-based learning.

Neural Network Efe used a Neural Network to simplify the design of a PID controller and decrease the computational time and complexity [32].

Fuzzy Logic The idea behind fuzzy logic is to translate the knowledge and actions of skilled human beings to a set of rules that can be used by a machine to execute a certain task usually executed by humans. So for flight control systems, a skilled pilot is usually the one doing the training for a fuzzy logic system [3].

2.1.4 Hybrid Flight Control Systems

In recent literature, it was found out that using only one type of flight control algorithms was not sufficient to guarantee a good performance, specially when the quadrotor is not flying near its nominal condition, so researchers are now proposing using more than one type of flight control algorithms. Azzam and Wang used a PD controller for altitude and yaw rotation and a PID controller integrated with a backstepping controller for the pitch and roll control. An optimization algorithm was used instead of the pole placement technique to overcome the difficulty of pole placement in a nonlinear time variant system. The system was divided into rotational and translation subsystems where the translation subsystem stabilizes the quadrotor position in flight and generates the needed roll and pitch angles to be fed to the rotational subsystem [5]. Nagaty et al. proposed the usage of a nested loop control algorithm; the outer loop consists of a PID controller responsible for the generation of the desired attitude angles that would achieve the desired position. These attitude

angles are then fed to the inner loop. The inner loop stabilization controller relies on the backstepping algorithm to track the desired altitude, attitude and heading [33].

2.2 Available Hardware Platforms

A lot of universities are using quadrotors for research, whether it is a control oriented research or autonomous systems and navigation oriented research. We will show here a brief overview of some of the famous quadrotor hardware used by universities and research groups.

Bouabdallah et al. used a platform called the OS4 for testing their proposed control algorithms, the OS4 was manufactured and assembled in their laboratory [11]. Hoffmann et al. used the STARMAC II UAV to study the aerodynamic effects on the quadrotor when operating far from the hovering position [34]. The X4-Flyer is used in the Australian National University [35]. Also, the Mesicopter was developed by Stanford university using off-the-shelf components [35].

Other universities and research groups would rather save the time, cost and effort of building a quadrotor model from scratch and use a commercially available platform. One of the most widely used platform is the one produced by AscTec [36] company, which is a company that sells quadrotors specially designed to be used in research related tasks. Their models were used by a lot of universities and research groups such as: the Aerospace Controls Lab, MIT [37] and the Robust Robotics Group, MIT [38] and in GRASP Lab, University of Pennsylvania [39]. Another widely used platform is the Draganflyer from Draganfly Innovations [40] also used by the Aerospace Controls Lab, MIT [37] [3].

There is also the German company Microdrones [41] that develops UAVs to be used in many applications such as aerial surveillance by police, power lines inspection, aerial terrain mapping, bridge inspections among others. Another company is the French company Parrot [42], their model AR.Drone is mainly designed for entertainment purposes and it can be remotely controlled by an iPhone.

2.3 Research Challenges

After an intensive literature review of the control algorithms applied on quadrotors, it was found out that one of the research challenges facing researchers in this field is flying the quadrotor outside the linear or hovering region. Also, there no work focused on having a comparative study that compares between employing different types of controllers on the quadrotor system and comparing their performances in stabilizing the quadrotor under different flight conditions and this will be the focus of this work.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 3

System Modeling

In this chapter, the kinematics and dynamics models of a quadrotor will be derived based on a Newton-Euler formalism with the following assumptions:

- The structure is rigid and symmetrical.
- The center of gravity of the quadrotor coincides with the body fixed frame origin.
- The propellers are rigid.
- Thrust and drag are proportional to the square of propeller's speed.

After deriving the kinematics and dynamics models of the quadrotor, the aerodynamic effects acting on the quadrotor body will be discussed together with the rotor dynamics of the actuators of the quadrotor. The chapter will be ended with formulating a state space model for the quadrotor system that will be used in the subsequent modeling chapter.

3.1 Kinematic Model

In order to discuss the modeling of the quadrotor, we first need to define the coordinate frames that will be used. Figure 3-1 shows the Earth reference frame with N, E and D axes and the body frame with x, y and z axes. The Earth frame is an inertial

frame fixed on a specific place at ground level as its name implies, it uses the N-E-D notation where the axes point to the North, East and Downwards respectively. On the other hand, the body frame is at the center of the quadrotor body, with its x-axis pointing towards propeller 1, y-axis pointing towards propeller 2 and the z-axis is pointing to the ground.

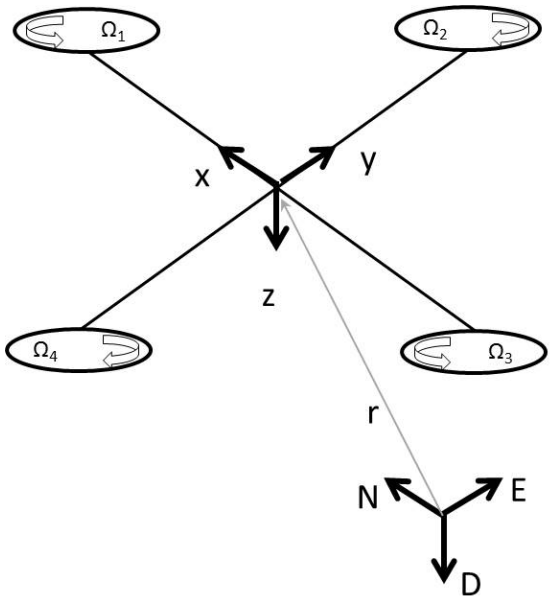


Figure 3-1: Quadrotor Reference Frames

The distance between the Earth frame and the body frame describes the absolute position of the center of mass of the quadrotor $r = [x\ y\ z]^T$. The rotation R from the body frame to the inertial frame describes the orientation of the quadrotor. The orientation of the quadrotor is described using roll, pitch and yaw angles (ϕ, θ and ψ) representing rotations about the X, Y and Z-axes respectively. Assuming the order of rotation to be roll (ϕ), pitch (θ) then yaw (ψ), the rotation matrix R which is derived based on the sequence of principle rotations is:

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

where c and s denote cos and sin respectively. The derivation of the rotation

matrix R is shown in details in Appendix A.1.1.

The rotation matrix R will be used in formulating the dynamics model of the quadrotor, its significance is due to the fact that some states are measured in the body frame (e.g. the thrust forces produced by the propellers) while some others are measured in the inertial frame (e.g. the gravitational forces and the quadrotor's position). Thus, to have a relation between both types of states, a transformation from one frame to the other is needed.

To acquire information about the angular velocity of the quadrotor, typically an on-board Inertial Measurement Unit (IMU) is used which will in turn give the velocity in the body coordinate frame. To relate the Euler rates $\dot{\eta} = [\dot{\phi}\ \dot{\theta}\ \dot{\psi}]^T$ that are measured in the inertial frame and angular body rates $\omega = [p\ q\ r]^T$, a transformation is needed as follows:

$$\omega = R_r \dot{\eta} \tag{3.2}$$

where

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

Around the hover position, small angle assumption is made where $\cos\phi \equiv 1$, $\cos\theta \equiv 1$ and $\sin\phi = \sin\theta = 0$. Thus R_r can be simplified to an identity matrix I [33]. The derivation for the previous transformation is shown in Appendix A.1.2.

3.2 Dynamics Model

The motion of the quadrotor can be divided into two subsystems; rotational subsystem (roll, pitch and yaw) and translational subsystem (altitude and x and y position). The rotational subsystem is fully actuated while the translational subsystem is underactuated [33].

3.2.1 Rotational Equations of Motion

The rotational equations of motion are derived in the body frame using the Newton-Euler method with the following general formalism,

$$J\dot{\omega} + \omega \times J\omega + M_G = M_B \tag{3.3}$$

Where:

J	Quadrotor's diagonal inertia Matrix
ω	Angular body rates
M_G	Gyroscopic moments due to rotors' inertia
M_B	Moments acting on the quadrotor in the body frame

The first two terms of Equation 3.3, $J\dot{\omega}$ and $\omega \times J\omega$, represent the rate of change of angular momentum in the body frame. While M_G represent the gyroscopic moments due to the rotors' inertia J_r. The Gyroscopic moments are defined to be $\omega \times [0\ 0\ J_r\Omega_r]^T$, thus the rotational equation of the quadrotor's motion can be written as [33],

$$J\dot{\omega} + \omega \times J\omega + \omega \times [0\ 0\ J_r\Omega_r]^T = M_B \tag{3.4}$$

Where:

J_r	rotors' inertia
Ω_r	rotors' relative speed $\Omega_r = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4$

The reason behind deriving the rotational equations of motion in the body frame and not in the inertial frame, is to have the inertia matrix independent on time.

Inertia Matrix

The inertia matrix for the quadrotor is a diagonal matrix, the off-diagonal elements, which are the product of inertia, are zero due to the symmetry of the quadrotor.

$$J = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{3.5}$$

Where I_{xx}, I_{yy} and I_{zz} are the area moments of inertia about the principle axes in the body frame.

Gyroscopic Moment

The gyroscopic moment of a rotor is a physical effect in which gyroscopic torques or moments attempt to align the spin axis of the rotor along the inertial z-axis [43].

Moments Acting on the Quadrotor (M_B)

For the last term of equation (3.4), there is a need to define two physical effects which are the aerodynamic forces and moments produced by a rotor. As an effect of rotation, there is a generated force called the aerodynamic force or the lift force and there is a generated moment called the aerodynamic moment. Equations (3.6) and (3.7) show the aerodynamic force F_i and moment M_i produced by the i^{th} rotor [5].

$$F_i = \frac{1}{2}\rho A C_T r^2 \Omega_i^2 \tag{3.6}$$

$$M_i = \frac{1}{2}\rho A C_D r^2 \Omega_i^2 \tag{3.7}$$

where

ρ	air density
A	blade area
C_T, C_D	aerodynamic coefficients
r_b	radius of blade
Ω_i	angular velocity of rotor i

Clearly, the aerodynamic forces and moments depend on the geometry of the propeller and the air density. Since for the case of quadrotors, the maximum altitude is usually limited, thus the air density can be considered constant, Equations (3.6) and (3.7) can be simplified to [33],

$$F_i = K_f \Omega_i^2 \tag{3.8}$$

$$M_i = K_M \Omega_i^2 \tag{3.9}$$

Where K_f and K_M are the aerodynamic force and moment constants respectively and Ω_i is the angular velocity of rotor i. The aerodynamic force and moment constants can be determined experimentally for each propeller type.

By identifying the forces and moments generated by the propellers, we can study the moments M_B acting on the quadrotor. Figure 3-2 shows the forces and moments acting on the quadrotor. Each rotor causes an upwards thrust force F_i and generates a moment M_i with direction opposite to the direction of rotation of the corresponding rotor i.

Starting with the moments about the body frame's x-axis, by using the right-hand-rule in association with the axes of the body frame, F_2 multiplied by the moment arm l generates a negative moment about the y-axis, while in the same manner, F_4 generates

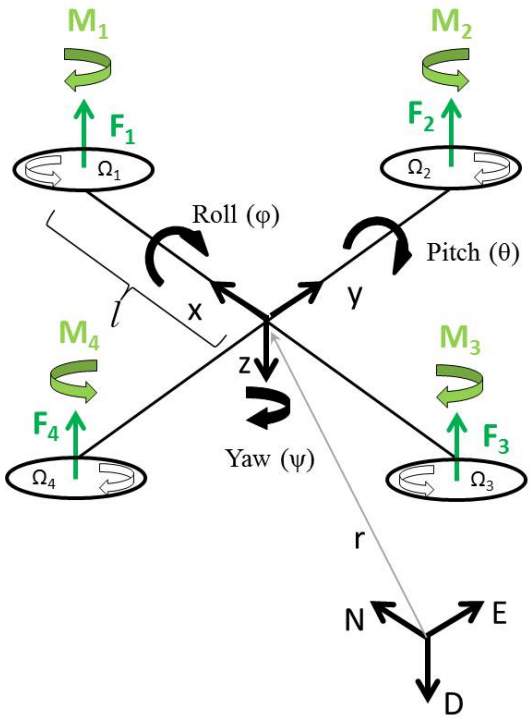


Figure 3-2: Forces and Moments acting on Quadrotor

a positive moment. Thus the total moment about the x-axis can be expressed as

$$\begin{aligned} M_x &= -F_2 l + F_4 l \\ &= -(K_f \Omega_2^2) l + (K_f \Omega_4^2) l \\ &= l K_f (-\Omega_2^2 + \Omega_4^2) \end{aligned} \tag{3.10}$$

For the moments about the body frame's y-axis, also using the right-hand-rule, the thrust of rotor 1 generates a positive moment, while the thrust of rotor 3 generates a negative moment about the y-axis. The total moment can be expressed as,

$$\begin{aligned} M_y &= F_1 l - F_3 l \\ &= (K_f \Omega_1^2) l - (K_f \Omega_3^2) l \\ &= l K_f (\Omega_1^2 - \Omega_3^2) \end{aligned} \tag{3.11}$$

For the moments about the body frame's z-axis, the thrust of the rotors does

not cause a moment. On the other hand, moment caused by the rotors' rotation as per Equation 3.7. By using the right-hand-rule, the moment about the body frame's z-axis can be expressed as,

$$\begin{aligned} M_z &= M_1 - M_2 + M_3 - M_4 \\ &= (K_M\Omega_1^2) - (K_M\Omega_2^2) + (K_M\Omega_3^2) - (K_M\Omega_4^2) \\ &= K_M(\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2)) \end{aligned} \tag{3.12}$$

Combining equations (3.10), (3.11) and (3.12) in vector form, we get,

$$M_B = \begin{bmatrix} lK_f(-\Omega_2^2 + \Omega_4^2) \\ lK_f(\Omega_1^2 - \Omega_2^2) \\ K_M(\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2) \end{bmatrix} \tag{3.13}$$

where l is the moment arm, which is the distance between the axis of rotation of each rotor to the origin of the body reference frame which should coincide with the center of the quadrotor.

3.2.2 Translational Equations of Motion

The translation equations of motion for the quadrotor are based on Newton's second law and they are derived in the Earth inertial frame [33],

$$m\ddot{r} = \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} + RF_B \tag{3.14}$$

Where

$r = [x\ y\ z]^T$	Quadrotor's distance from the inertial frame
m	Quadrotor's mass
g	gravitational acceleration g =$9.81m/s^2$
F_B	nongravitational forces acting on the quadrotor in the body frame

Nongravitational Forces Acting on the Quadrotor

When the quadrotor is in a horizontal orientation (i.e. it is not rolling or pitching), the only nongravitational forces acting on it is the thrust produced by the rotation of the propellers which is proportional to the square of the angular velocity of the propeller as per Equation (3.8). Thus, the nogravitational forces acting on the quadrotor, F_B, can be expressed as,

$$F_B = \begin{bmatrix} 0 \\ 0 \\ -K_f(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \end{bmatrix} \tag{3.15}$$

The first two rows of the force vector are zeros as there is no forces in the X and Y directions, the last row is simply an addition of the thrust forces produced by the four propellers. The negative sign is due to the fact that the thrust is upwards while the positive z-axis in the body framed is pointing downwards.

F_B is multiplied by the rotation matrix R to transform the thrust forces of the rotors from the body frame to the inertial frame, so that the equation can be applied in any orientation of the quadrotor.

3.3 Aerodynamic Effects

In the previous dynamics formulation, the aerodynamic effects acting on the quadrotor body were neglected. However, in order to have an accurate and realistic model to be used in simulations, aerodynamic effects should be included. There are namely two types of aerodynamic effects, drag forces and drag moments [44].

3.3.1 Drag Forces

Due to the friction of the moving quadrotor body with air, a force acts on the body of the quadrotor resisting the motion. As the velocity of travel of the quadrotor increases, the drag forces in turn increase. The drag forces F_a can be approximated

by,

$$F_a = K_t \dot{r} \tag{3.16}$$

where K_t is a constant matrix called the aerodynamic translation coefficient matrix and $\dot{r}$ is the time derivative of the position vector r. This indicates that there is an extra force acting on the quadrotor body, the translational equation of motion Equation (3.14) should be rewritten to be,

$$m\ddot{r} = \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} + RF_B - F_a \tag{3.17}$$

3.3.2 Drag Moments

The same as the drag force, due to the air friction, there is a drag moment M_a acting on the quadrotor body which can be approximated by,

$$M_a = K_r \dot{\eta} \tag{3.18}$$

where K_r is a constant matrix called the aerodynamic rotation coefficient matrix and $\dot{\eta}$ is the Euler rates. Accordingly, the rotational equation of motion expressed by Equation (3.4) can be rewritten to as,

$$J\dot{\omega} + \omega \times J\omega + \omega \times [0\ 0\ J_r\Omega_r]^T = M_B - M_a \tag{3.19}$$

3.4 Rotor Dynamics

The motors typically used in quadrotors are brushless DC motors that provide high torque and little friction. In the following derivation, it is assumed that the rotors are nongeared with rigid mechanical coupling between the motors and the propellers. The dynamics of a brushless DC motor at steady state is the same as a conventional

DC motor. The schematic for a brushless DC motor at steady state is shown in Figure 3-3.

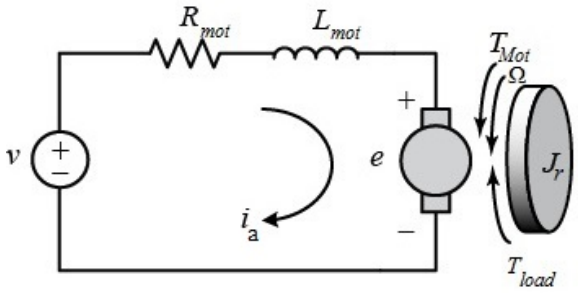


Figure 3-3: DC Motor Schematic Diagram

Using Kirchhoff's voltage law, the following equation can be derived

$$v = R_{mot} i_a + L_{mot} \frac{di_a}{dt} + K_{mot} \Omega \tag{3.20}$$

where R_{mot} and L_{mot} are the i_{th} motor's resistance and inductance respectively, i_a is the armature current, v is the input voltage and the term $K_{mot}\Omega$ represents the generated emf, e, with K_{mot} as the motor torque constant. Since the quadrotor relies on small motors, their inductance is very small and thus can be neglected leading to

$$v = R_{mot} i_a + K_{mot} \Omega_i \tag{3.21}$$

or

$$i_a = \frac{v - K_{mot}\Omega_i}{R_{mot}} \tag{3.22}$$

Moving to the mechanical derivation

$$J_r \dot{\Omega}_i = T_{mot} - T_{load} \tag{3.23}$$

where T_{mot} is the torque produced by the motor which is equal to $K_e i_a$, where K_e is the motor's electric constant and for small motors it is approximately equal to K_{mot}. T_{load} is the load torque which is the torque generated from the propeller system which

is $K_M\Omega^2$ as per Equation (3.7). Substituting the equations of T_{mot} and T_{load} together with the current equation from Equation (3.22) yields,

$$J_r\dot{\Omega}_i = K_{mot}\frac{v - K_{mot}\Omega_i}{R_{mot}} - K_M\Omega_i^2 \tag{3.24}$$

After simplification the voltage can be written as a function of the rotor's velocity as follows

$$v = \frac{R_{mot}}{K_{mot}}J_r\dot{\Omega}_i + K_{mot}\Omega_i + K_M R_{mot}\Omega_i^2 \tag{3.25}$$

The rotor dynamics can be approximated to a first order lag transfer function with its parameters (gain and time constant) identified experimentally by using a system identification tool (e.g. MATLAB's System Identification Toolbox). The transfer function maps the desired propeller's speed to the actual speed [16]. Depending on the [OS4] quadrotor hardware in [16], the first order lag transfer function was identified to be,

$$G(s) = \frac{\text{Actual rotor speed}}{\text{Commanded rotor speed}} = \frac{0.936}{0.178s + 1} \tag{3.26}$$

Brushless DC motor typically have their own embedded controllers and they work with a Pulse Width Modulated (PWM) signal. The voltage supplied to the brushless DC motor is directly proportional to the RPM of their rotation, the relationship can be found by a black box identification process for the motor and propeller pair. The constant of proportionality of this linear relationship appears as a gain in the transfer function in (3.26). The application of the rotor dynamics transfer function will be further elaborated in Section 4.1.

3.5 State Space Model

Formulating the acquired mathematical model for the quadrotor into a state space model will help make the control problem easier to tackle.

3.5.1 State Vector X

Defining the state vector of the quadrotor to be,

$$X = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 & x_9 & x_{10} & x_{11} & x_{12} \end{bmatrix}^T \tag{3.27}$$

which is mapped to the degrees of freedom of the quadrotor in the following manner,

$$X = \begin{bmatrix} \phi & \dot{\phi} & \theta & \dot{\theta} & \psi & \dot{\psi} & z & \dot{z} & x & \dot{x} & y & \dot{y} \end{bmatrix}^T \tag{3.28}$$

The state vector defines the position of the quadrotor in space and its angular and linear velocities.

3.5.2 Control Input Vector U

A control input vector, U, consisting of four inputs; U_1 through U_4 is defined as,

$$U = \begin{bmatrix} U_1 & U_2 & U_3 & U_4 \end{bmatrix} \tag{3.29}$$

Where

$$U_1 = K_f(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \tag{3.30}$$

$$U_2 = K_f(-\Omega_2^2 + \Omega_4^2) \tag{3.31}$$

$$U_3 = K_f(\Omega_1^2 - \Omega_3^2) \tag{3.32}$$

$$U_4 = K_M(\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2) \tag{3.33}$$

Equations (3.30) through (3.33) can be arranged in a matrix form to result in,

$$\begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} K_f & K_f & K_f & K_f \\ 0 & -K_f & 0 & K_f \\ K_f & 0 & -K_f & 0 \\ K_M & -K_M & K_M & -K_M \end{bmatrix} \begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} \tag{3.34}$$

U_1 is the resulting upwards force of the four rotors which is responsible for the altitude of the quadrotor and its rate of change $(z, \dot{z})$. U_2 is the difference in thrust between rotors 2 and 4 which is responsible for the roll rotation and its rate of change $(\phi, \dot{\phi})$. U_3 on the other hand represents the difference in thrust between rotors 1 and 3 thus generating the pitch rotation and its rate of change $(\theta, \dot{\theta})$. Finally U_4 is the difference in torque between the two clockwise turning rotors and the two counterclockwise turning rotors generating the yaw rotation and ultimately its rate of change $(\psi, \dot{\psi})$. This choice of the control vector U decouples the rotational system, where U_1 will generate the desired altitude of the quadrotor, U_2 will generate the desired roll angle, the desired pitch angle will be generated by U_3 whereas U_4 will generate the desired heading.

If the rotor velocities are needed to be calculated from the control inputs, an inverse relationship between the control inputs and the rotors' velocities is needed, which can be acquired by inverting the matrix in Equation (3.34) to give,

$$\begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} = \begin{bmatrix} \frac{1}{4K_f} & 0 & \frac{1}{2K_f} & \frac{1}{4K_M} \\ \frac{1}{4K_f} & -\frac{1}{2K_f} & 0 & -\frac{1}{4K_M} \\ \frac{1}{4K_f} & 0 & -\frac{1}{2K_f} & \frac{1}{4K_M} \\ \frac{1}{4K_f} & \frac{1}{2K_f} & 0 & -\frac{1}{4K_M} \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} \tag{3.35}$$

Taking the square root of that, the rotors' velocities can be calculated from the control inputs as follows,

$$\begin{aligned} \Omega_1 &= \sqrt{\frac{1}{4K_f}U_1 + \frac{1}{2K_f}U_3 + \frac{1}{4K_M}U_4} \\ \Omega_2 &= \sqrt{\frac{1}{4K_f}U_1 - \frac{1}{2K_f}U_2 - \frac{1}{4K_M}U_4} \\ \Omega_3 &= \sqrt{\frac{1}{4K_f}U_1 - \frac{1}{2K_f}U_3 + \frac{1}{4K_M}U_4} \\ \Omega_4 &= \sqrt{\frac{1}{4K_f}U_1 + \frac{1}{2K_f}U_2 - \frac{1}{4K_M}U_4} \end{aligned} \tag{3.36}$$

3.5.3 Rotational Equation of Motion

Substituting equations (3.30) through (3.33) in equation (3.13), the equation of the total moments acting on the quadrotor becomes

$$M_B = \begin{bmatrix} l\,U_2 \\ l\,U_3 \\ U4 \end{bmatrix} \tag{3.37}$$

Substituting (3.37) into the rotational equation of motion (3.4) and expanding each term with their prior definition from Chapter 3, the following relation can be derived,

$$\begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} + \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \times \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ J_r\Omega_r \end{bmatrix} = \begin{bmatrix} l\,U_2 \\ l\,U_3 \\ U4 \end{bmatrix}$$

Expanding that, leads to,

$$\begin{bmatrix} I_{xx}\ddot{\phi} \\ I_{yy}\ddot{\theta} \\ I_{zz}\ddot{\psi} \end{bmatrix} + \begin{bmatrix} \dot{\theta}I_{zz}\dot{\psi} - \dot{\psi}I_{yy}\dot{\theta} \\ \dot{\psi}I_{xx}\dot{\phi} - \dot{\phi}I_{zz}\dot{\psi} \\ \dot{\phi}I_{yy}\dot{\theta} - \dot{\theta}I_{xx}\dot{\phi} \end{bmatrix} + \begin{bmatrix} \dot{\theta}J_r\Omega_r \\ -\dot{\phi}J_r\Omega_r \\ 0 \end{bmatrix} = \begin{bmatrix} l\,U_2 \\ l\,U_3 \\ U4 \end{bmatrix} \tag{3.38}$$

Rewriting the last equation to have the angular accelerations in terms of the other variables,

$$\ddot{\phi} = \frac{l}{I_{xx}}U_2 - \frac{J_r}{I_{xx}}\dot{\theta}\Omega_r + \frac{I_{yy}}{I_{xx}}\dot{\psi}\dot{\theta} - \frac{I_{zz}}{I_{xx}}\dot{\theta}\dot{\psi} \tag{3.39}$$

$$\ddot{\theta} = \frac{l}{I_{yy}}U_3 - \frac{J_r}{I_{yy}}\dot{\phi}\Omega_r + \frac{I_{zz}}{I_{yy}}\dot{\phi}\dot{\psi} - \frac{I_{xx}}{I_{yy}}\dot{\psi}\dot{\phi} \tag{3.40}$$

$$\ddot{\psi} = \frac{1}{I_{zz}}U_4 + \frac{I_{xx}}{I_{zz}}\dot{\theta}\dot{\phi} - \frac{I_{yy}}{I_{zz}}\dot{\phi}\dot{\theta} \tag{3.41}$$

To simplify, define,

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

Using the above definition of $a_1 \to a_5$ and $b_1 \to b_3$, equations (3.39) through (3.41) can then be rewritten in a simpler form in terms of the system states,

$$\ddot{\phi} = b_1 U_2 - a_2 x_4 \Omega_r + a_1 x_4 x_6 \tag{3.42}$$

$$\ddot{\theta} = b_2 U_3 + a_4 x_2 \Omega_r + a_3 x_2 x_6 \tag{3.43}$$

$$\ddot{\psi} = b_3 U_4 + a_5 x_2 x_4 \tag{3.44}$$

With the choice of the control input vector U, it is clear that the rotational subsystem is fully-actuated, it is only dependant on the rotational state variables $x_1 \to x_6$ that correspond to ϕ, $\dot{\phi}$, θ, $\dot{\theta}$, ψ, $\dot{\psi}$ respectively.

3.5.4 Translational Equation of Motion

Substituting equations (3.30) through (3.33) in equation (3.15), the equation of the total moments acting on the quadrotor becomes,

$$F_B = \begin{bmatrix} 0 \\ 0 \\ -U_1 \end{bmatrix} \tag{3.45}$$

Embedding that into the translational equation of motion (3.14) and expanding the

terms, we get

$$
\begin{aligned}
m\begin{bmatrix}\ddot{x}\\ \ddot{y}\\ \ddot{z}\end{bmatrix} &= \begin{bmatrix}0\\0\\mg\end{bmatrix} + \begin{bmatrix}c\psi c\theta & c\psi s\phi s\theta & s\phi s\psi + c\phi c\psi s\theta\\ c\theta s\psi & c\theta c\psi + s\phi s\psi s\theta & c\phi s\psi s\theta - c\psi s\theta\\ -s\theta & c\theta s\phi & c\phi c\theta\end{bmatrix}\begin{bmatrix}0\\0\\-U_1\end{bmatrix}\\
m\begin{bmatrix}\ddot{x}\\ \ddot{y}\\ \ddot{z}\end{bmatrix} &= \begin{bmatrix}0\\0\\mg\end{bmatrix} + \begin{bmatrix}(s\phi s\psi + c\phi c\psi s\theta)(-U_1)\\ (c\phi s\psi s\theta - c\psi s\phi)(-U_1)\\ (c\phi c\theta)(-U_1)\end{bmatrix}
\end{aligned}
\tag{3.46}
$$

Rewriting Equation (3.46) to have the accelerations in terms of the other variables, we get

$$\ddot{x} = \frac{-U_1}{m}(\sin\phi\sin\psi + \cos\phi\cos\psi\sin\theta) \tag{3.47}$$

$$\ddot{y} = \frac{-U_1}{m}(\cos\phi\sin\psi\sin\theta - \cos\psi\sin\phi) \tag{3.48}$$

$$\ddot{z} = g - \frac{U_1}{m}(\cos\phi\cos\theta) \tag{3.49}$$

Rewriting in terms of the state variable X,

$$\ddot{x} = \frac{-U_1}{m}(\sin x_1\sin x_5 + \cos x_1\cos x_5\sin x_3) \tag{3.50}$$

$$\ddot{y} = \frac{-U_1}{m}(\cos x_1\sin x_5\sin x_3 - \cos x_5\sin x_1) \tag{3.51}$$

$$\ddot{z} = g - \frac{U_1}{m}(\cos x_1\cos x_3) \tag{3.52}$$

It is clear here that the translational subsystem is underactuated as it dependant on both the translational state variables and the rotational ones.

3.5.5 State Space Representation

Using the equations of the rotational angular acceleration. Equations (3.42) to (3.44), and those of translation, Equations (3.50) to (3.52), the complete mathematical model

of the quadrotor can be written in a state space representation as follows,

$$
\begin{aligned}
\dot{x}_1 &= \dot{\phi} = x_2 \\
\dot{x}_2 &= \ddot{\phi} = x_4 x_6 a_1 - x_4 \Omega_r a_2 + b_1 U_2 \\
\dot{x}_3 &= \dot{\theta} = x_4 \\
\dot{x}_4 &= \ddot{\theta} = x_2 x_6 a_3 + x_2 \Omega_r a_4 + b_2 U_3 \\
\dot{x}_5 &= \dot{\psi} = x_6 \\
\dot{x}_6 &= \ddot{\psi} = x_2 x_4 a_5 + b_3 U_4 \\
\dot{x}_7 &= \dot{z} = x_8 \\
\dot{x}_8 &= \ddot{z} = g - \frac{U_1}{m}(\cos x_1 \cos x_3) \\
\dot{x}_9 &= \dot{x} = x_{10} \\
\dot{x}_{10} &= \ddot{x} = \frac{-U_1}{m}(\sin x_1 \sin x_5 + \cos x_1 \sin x_3 \cos x_5) \\
\dot{x}_{11} &= \dot{y} = x_{12} \\
\dot{x}_{12} &= \ddot{y} = \frac{U_1}{m}(\sin x_1 \cos x_5 - \cos x_1 \sin x_3 \sin x_5)
\end{aligned}
$$

$$
f(X,U) = \begin{bmatrix}
x_2 \\
x_4 x_6 a_1 - x_4 \Omega_r a_2 + b_1 U_2 \\
x_4 \\
x_2 x_6 a_3 + x_2 \Omega_r a_4 + b_2 U_3 \\
x_6 \\
x_2 x_4 a_5 + b_3 U_4 \\
x_8 \\
g - \frac{U_1}{m}(\cos x_1 \cos x_3) \\
x_{10} \\
\frac{-U_1}{m}(\sin x_1 \sin x_5 + \cos x_1 \sin x_3 \cos x_5) \\
x_{12} \\
\frac{U_1}{m}(\sin x_1 \cos x_5 - \cos x_1 \sin x_3 \sin x_5)
\end{bmatrix} \tag{3.53}
$$

Chapter 4

System Control

In this chapter, the formulated quadrotor model will be used in open-loop simulations and controller design. Four controllers will be developed, a linear PID controller, non-linear Sliding Mode, Backstepping and Gain Scheduling controllers. The parameters and gains of these controllers will be tuned using Genetic Algorithm (GA). Computer based simulations will then be implemented on MATLAB/Simulink and will be used to asses the performance of the developed controllers.

4.1 Open Loop Simulation

To verify the mathematical model, an open loop simulation was carried out using MATLAB/Simulink. The quadrotor's parameters were taken from Bouabdallah's PhD thesis which is based on the OS4 hardware [16]. The block diagram for the simulation is shown in Figure 4-1.

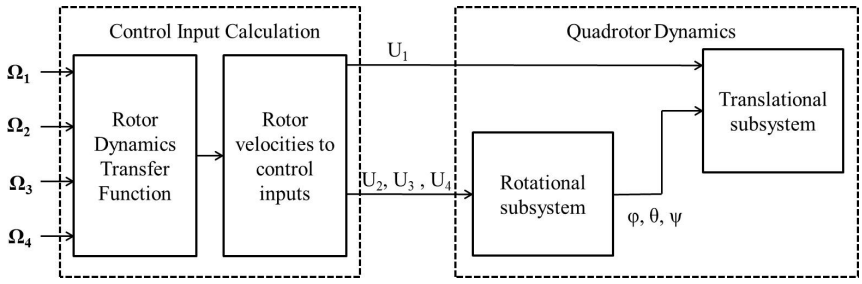


Figure 4-1: Open Loop Block Diagram

The rotors speed required for the quadrotor to hover was calculated using the following equation

$$\begin{aligned} mg &= 4F_i \\ mg &= 4(K_f \Omega_{ih}^2) \end{aligned} \tag{4.1}$$

where Ω_{ih} is the hover angular velocity of rotor i.

It is assumed that the only forces acting on the quadrotor are the four upwards thrust forces of the four rotors and the downwards gravitational acceleration. Equation (3.34) is used to calculate the control input values for different rotor angular velocities. When Ω_{ih} was fed to the open loop simulation model shown in Figure 4-1 in place of the angular velocities Ω_1 through Ω_4, it was observed that all the state variables $(x, y, z, \phi, \theta, \psi)$ and their derivatives were kept at a zero value. By increasing the rotors' speed above their hover value with the same value of increase, the only change in the state variables was in the altitude of the quadrotor. Also, varying the angular velocity of the 4 rotors, produced the respective roll, pitch or yaw motion. These tests helped verify the correctness of the derived mathematical model and the integrity of the open loop simulation.

Rotor dynamics are included in the "Control Input Calculation" block as the first order lag transfer function shown in Equation (3.26). Similarly, in all the proceeding block diagrams, the rotor velocities calculation based on the control inputs is followed by the lag transfer function to map the desired rotor velocities to the actual ones.

4.2 Closed Loop Simulation

After the derived mathematical model of the quadrotor was verified using the open loop simulation, the simulation environment is then extended to include an altitude, attitude, heading, and position controllers. The frequency of the simulation environment is set to 250 Hz with a fixed step size which is the average frequency of a controller in an actual quadrotor system platform.

4.2.1 Altitude Controller

The open loop simulation previously developed was expanded to include an altitude controller as shown in the block diagram in Figure 4-2. The altitude controller takes

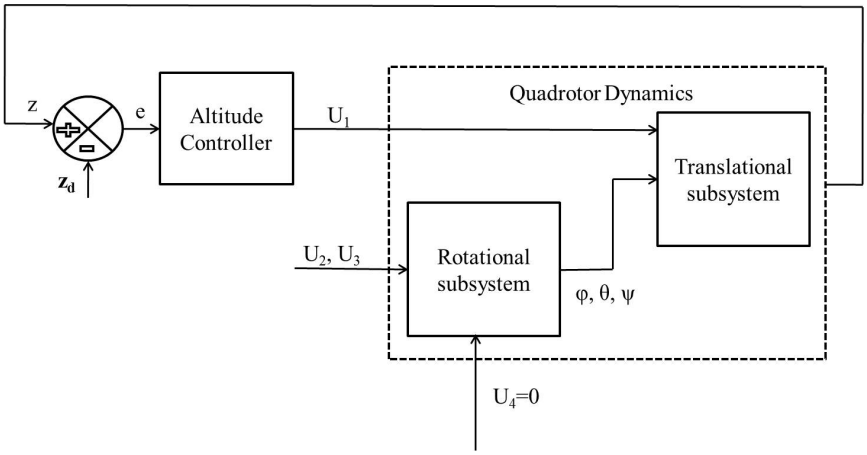


Figure 4-2: Block Diagram for Altitude Controller

an error signal e as an input which is the difference between the desired altitude z_d and the actual altitude z and produces a control signal U_1.

4.2.2 Attitude and Heading Controller

To control the attitude and heading of the quadrotor, the open loop simulation was modified to include the attitude and heading controllers as shown in Figure 4-3. Similar to the attitude controller block, the attitude and heading controller take as an input an error signal e which is the difference between the desired roll ϕ_d, pitch θ_d and yaw ψ_d and their actual values ϕ, θ and ψ. The attitude and heading controller produces the output signals U_2, U_3 and U_4.

4.2.3 Position Controller

Unlike the altitude and orientation of the quadrotor, its x and y position is not decoupled and cannot be directly controlled using one of the four control laws U_1 through U_4. On the other hand, the x and y position can be controlled through the roll and pitch angles. The desired roll and pitch angles ϕ_d and θ_d can be calculated

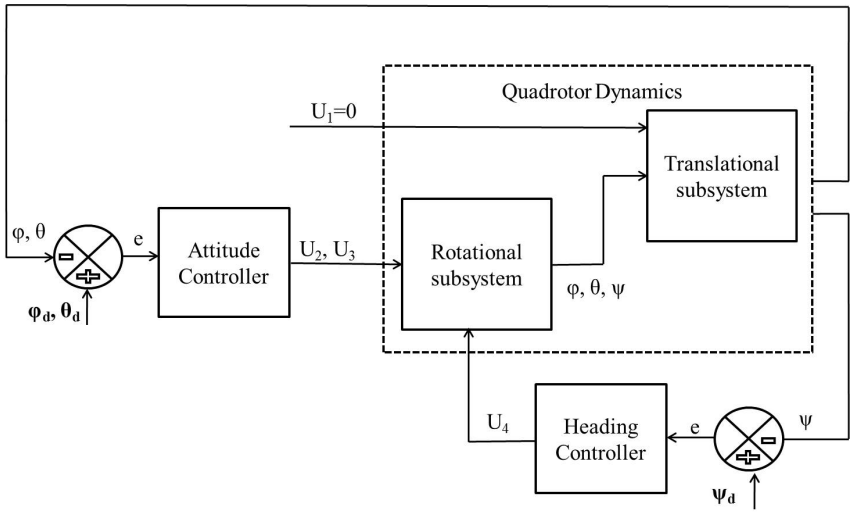


Figure 4-3: Block Diagram for Attitude and Heading Controller

from the translational equations of motion, Equations (3.47) and (3.48) as follows:

$$\ddot{x} = \frac{-U_1}{m}(\sin\phi_d \sin\psi + \cos\phi_d \sin\theta_d \cos\psi)$$
$$\ddot{y} = \frac{-U_1}{m}(\cos\phi_d \sin\theta_d \sin\psi - \sin\phi_d \cos\psi)$$

Since the quadrotor is operating around hover, which means small values for the roll and pitch angles ϕ and θ, we can use the small angle assumption ($\sin\phi_d \equiv \phi_d, \sin\theta_d \equiv \theta_d$ and $\cos\phi_d = \cos\theta_d = 1$) to simplify the above equations,

$$\ddot{x} = \frac{-U_1}{m}(\phi_d \sin\psi + \theta_d \cos\psi) \tag{4.2}$$
$$\ddot{y} = \frac{-U_1}{m}(\theta_d \sin\psi - \phi_d \cos\psi) \tag{4.3}$$

which can be written in a matrix form as,

$$\begin{bmatrix} -\sin\psi & -\cos\psi \\ \cos\psi & -sin\psi \end{bmatrix} \begin{bmatrix} \phi_d \\ \theta_d \end{bmatrix} = \frac{m}{U_1} \begin{bmatrix} \ddot{x}_d \\ \ddot{y}_d \end{bmatrix} \tag{4.4}$$

which can be inverted to get

$$\begin{aligned}\begin{bmatrix}\phi_d \\ \theta_d\end{bmatrix} &= \begin{bmatrix}-\sin\psi & -\cos\psi \\ \cos\psi & -sin\psi\end{bmatrix}^{-1} \frac{m}{U_1}\begin{bmatrix}\ddot{x}_d \\ \ddot{y}_d\end{bmatrix} \\ &= \frac{m}{U_1}\begin{bmatrix}-\sin\psi & \cos\psi \\ -\cos\psi & -sin\psi\end{bmatrix}\begin{bmatrix}\ddot{x}_d \\ \ddot{y}_d\end{bmatrix} \\ &= \frac{m}{U_1}\begin{bmatrix}-\ddot{x}_d\sin\psi + \ddot{y}_d\cos\psi \\ -\ddot{x}_d\cos\psi - \ddot{y}_d sin\psi\end{bmatrix}\end{aligned} \tag{4.5}$$

The calculated ϕ_d and θ_d have to be limited to the range between $-20°$ and $20°$ to fulfill the small angle assumption used in the derivation and this can be done via a saturation function in the simulation.

The closed loop simulation for the altitude and attitude controllers is further enhanced to include the position controller as shown in the block diagram in Figure 4-4.

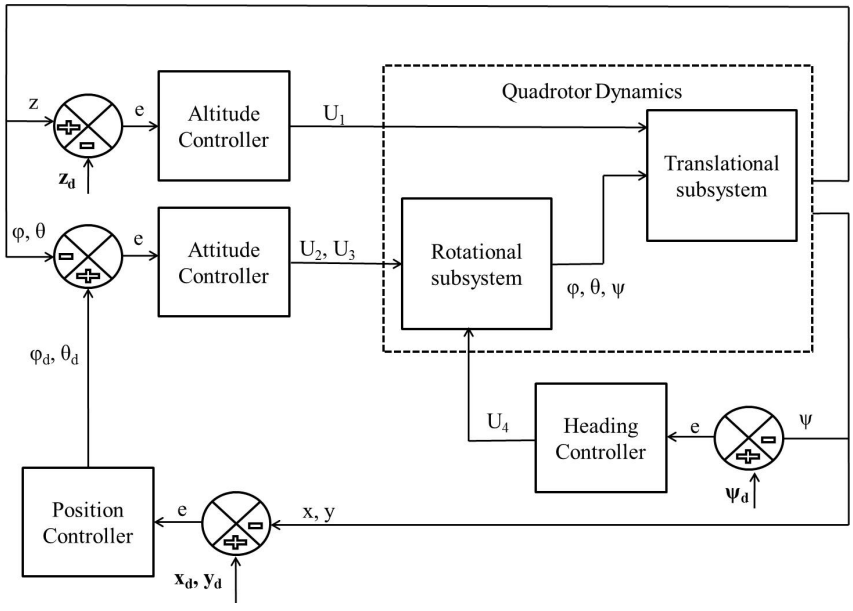


Figure 4-4: Position Controller Block Diagram (Complete System)

The controller blocks in the previous block diagrams can contain any type of control algorithm, whether linear or nonlinear. All controllers input(s) are the error related to some of the quadrotor's states and produce an output which is either one

or several control inputs U_1 through U_4 or ϕ_d and θ_d if it is the position controller.

4.3 Parameters Tuning Using GA

For the proceeding control algorithms, tuning the controller constants (gains and different parameters) was done using GA. The objective function of the GA was set to be the settling time of the response of the system. The GA is an iterative optimization algorithm works in the following way; first it generates a random "population" consisting of many individuals, which in our case will be a vector of values for the controller gains. The fitness of the individuals of the population is evaluated using an objective function, which is the settling time of the response of the system with these values set as the control gains. Another population is then generated from the current one using genetic operations like evolution, mutation and crossover and their fitness is also evaluated. The "elite" of the two populations are then selected to form a third population. The term "elite" indicates those individuals having the best fitness or the least value of the objective function (settling time in our case). The algorithm keeps on iterating until it reaches a population where all (or most of) its individuals are elite individuals and returns the individual (the value for the control gains) that has the least possible fitness (produces the least possible settling time for the system). In this work, we have not gone through the process of implementing a GA from scratch as this is out of our scope. Instead, the optimization toolbox in MATLAB was used and it includes a built-in command for GA optimization. The Block Diagram for the GA is shown in Figure 4-5.

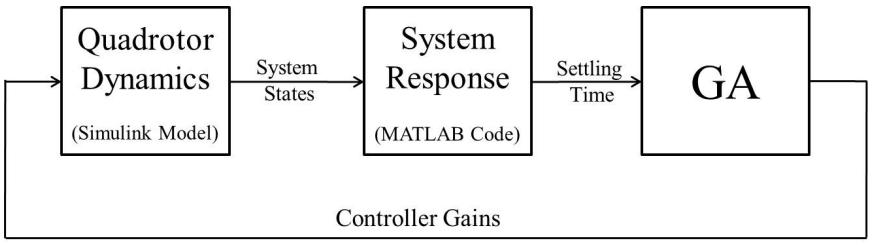


Figure 4-5: Controller Tuning using Genetic Algorithm

4.4 PID Controller

After the mathematical model of the quadrotor along with its open loop simulation are verified, a PID controller was developed. The PID controller generates the desired control inputs for the quadrotor. The block diagram for a PID controller is shown in Figure 4-6.

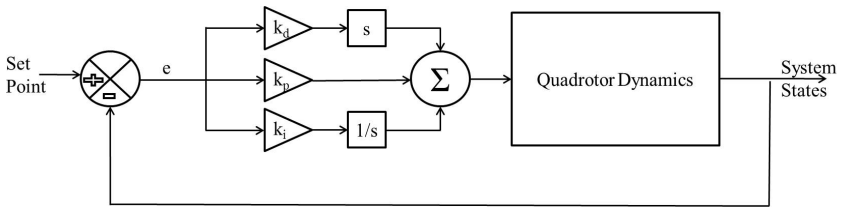


Figure 4-6: PID Controller Block Diagram

4.4.1 Altitude Control

A PID controller is developed to control the altitude of the quadrotor. It generates the control input U_1 which is responsible for the altitude for the quadrotor as per Equation (3.30). The derived control law is as follows

$$U_1 = k_p(z - z_d) + k_d(\dot{z} - \dot{z_d}) + k_i \int (z - z_d)dt \tag{4.6}$$

where

k_p Proportional gain

z_d Desired altitude

k_d Derivative gain

$\dot{z_d}$ Desired altitude rate of change

k_i Integral gain

4.4.2 Attitude and Heading Control

4.4.2.1 Roll Controller

Another PID controller is developed to control the roll angle ϕ of the quadrotor. The derived control law generates the input U_2 that controls the roll angle as follows

$$U_2 = k_p(\phi_d - \phi) + k_d(\dot{\phi}_d - \dot{\phi}) + k_i \int (\phi_d - \phi)dt \tag{4.7}$$

where

k_p Proportional gain

ϕ_d Desired roll angle

k_d Derivative gain

$\dot{\phi}_d$ Desired roll angle rate of change

k_i Integral gain

4.4.2.2 Pitch Controller

A PID controller is developed to control the pitch angle θ of the quadrotor. The derived control law generates the input U_3 that controls the pitch angle as follows,

$$U_3 = k_p(\theta_d - \theta) + k_d(\dot{\theta}_d - \dot{\theta}) + k_i \int (\theta_d - \theta)dt \tag{4.8}$$

where

k_p Proportional gain

θ_d Desired pitch angle

k_d Derivative gain

$\dot{\theta}_d$ Desired pitch angle rate of change

k_i Integral gain

4.4.2.3 Yaw Controller

Similar to the pitch and roll controllers, a yaw controller was developed to generate the control input U_4 based on the following control law,

$$U_4 = k_p(\psi_d - \psi) + k_d(\dot{\psi}_d - \dot{\psi}) + k_i \int (\psi_d - \psi)dt \tag{4.9}$$

where

k_p Proportional gain

ϕ_d Desired yaw angle

k_d Derivative gain

$\dot{\phi}_d$ Desired yaw angle rate of change

k_i Integral gain

4.4.3 Position Controller

After acquiring stable controllers for the altitude and the attitude of the quadrotor, a complete position controller is developed. PID controllers are used to calculate the desired accelerations $\ddot{x}_d$ and $\ddot{y}_d$

$$\ddot{x}_d = k_p(x_d - x) + k_d(\dot{x}_d - \dot{x}) + k_i \int (x_d - x)dt \tag{4.10}$$

$$\ddot{y}_d = k_p(y_d - y) + k_d(\dot{y}_d - \dot{y}) + k_i \int (y_d - y)dt \tag{4.11}$$

where

k_p Proportional gain

x_d Desired x position

k_d Derivative gain

$\dot{x}_d$ Desired x position rate of change

y_d Desired y position

$\dot{y}_d$ Desired y position rate of change

k_i Integral gain

Plugging the values of the desired accelerations $\ddot{x}_d$ and $\ddot{y}_d$ into Equation (4.5), the desired roll and pitch angles ϕ_d and θ_d can be calculated which are in turn fed to the attitude controller previously expressed in Equations (4.7) and (4.8).

4.5 PID Controller Simulation

For the altitude controller, GA was used to choose the control gains for the PID controller with a desired altitude z_d of 2 m. No steady state error was observed, so the PID controller was simplified to a PD controller by settling the integral gain k_i to zero. The control gains produced by the GA were $k_p = 5.2$ and $k_d = 1.3$. The objective function used to evaluate the GA was the settling time of the system. GA works to find control gains that would result in the least possible settling time for the altitude of the quadrotor. Running the closed loop simulation with the acquired gains resulted in a settling time of 1.3 sec and an overshoot of 1.4%.

Similarly, attitude, heading and position controllers gains were optimized using GA, Table 4.1 shows a summary of the optimized control gains and the performance of the system in terms of its settling time and overshoot. The response of the system is shown in Figure 4-7 and the respective control inputs are shown in Figure 4-8. Note that the reason the altitude response is in the negative z-axis is our previously assigned N-E-D axes for the quadrotor, that the z-axis points downwards.

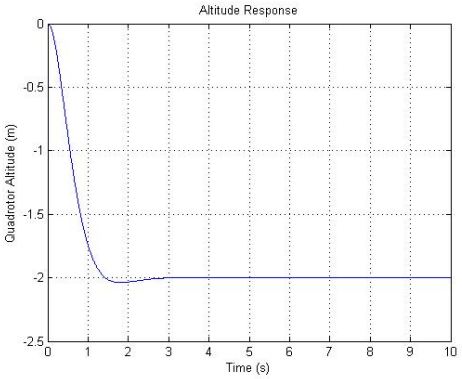
Due to the symmetry of the quadrotor, the controller for the pitch rotation is

equivalent to that of the roll rotation. This theory was verified and proved using the closed loop simulation and their results is shown in the attitude row in Table 4.1. As with the roll and pitch, the performance of the y position controller was exactly the same as the performance of the x position controller due to the symmetry of the quadrotor.

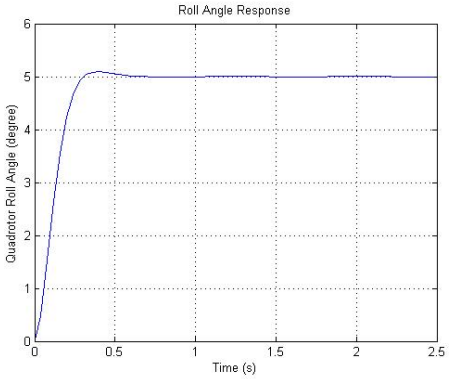
Thus, a complete position and altitude PD controller was developed for the quadrotor. This controller is able to perform well near hovering. The controller was also tested in commanding the quadrotor to follow a circular trajectory as shown in Figure 4-9.

Table 4.1: PD Controller Results

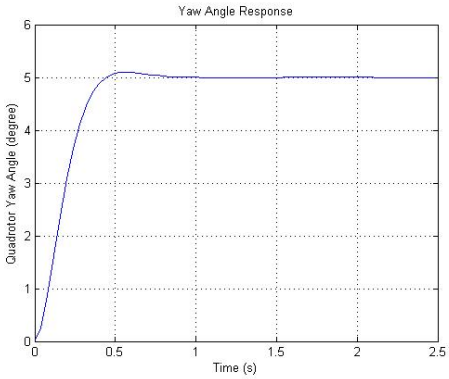
	Desired Value	k_p	k_d	Settling Time	Overshoot
Altitude (z)	2 m	5.2	1.3	1.3 sec	1.4 %
Attitude (ϕ and θ)	5°	4.5	0.5	0.3 sec	2%
Heading (ψ)	5°	3.9	0.7	0.42 sec	1.9%
Position (x and y)	1 m	7.5	4.2	1.4 sec	1.9%



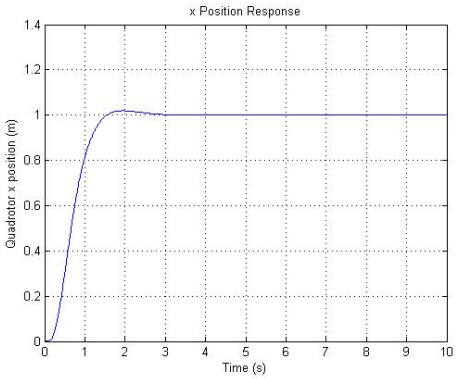
(a) Altitude Response



(b) Attitude Response

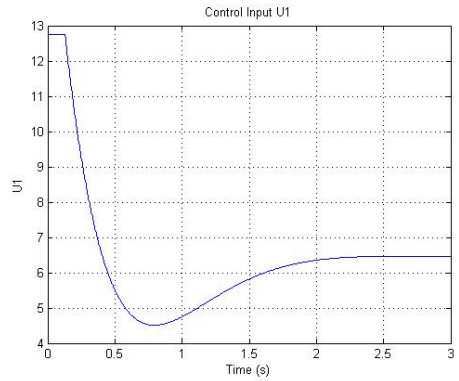


(c) Heading Response

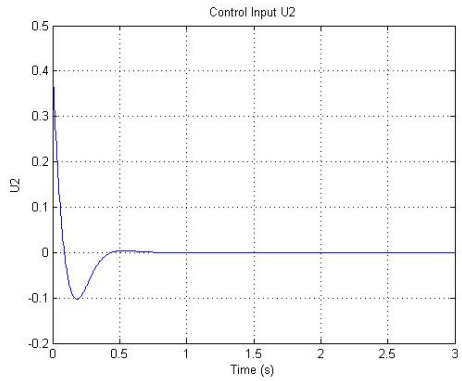


(d) Position Response

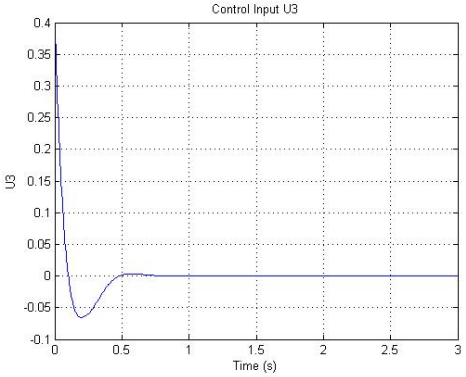
Figure 4-7: PD Controller Simulation Response



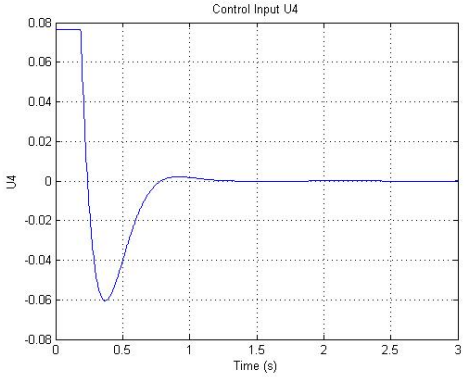
(a) Control Input U_1



(b) Control Input U_2



(c) Control Input U_3



(d) Control Input U_4

Figure 4-8: PD Control Inputs

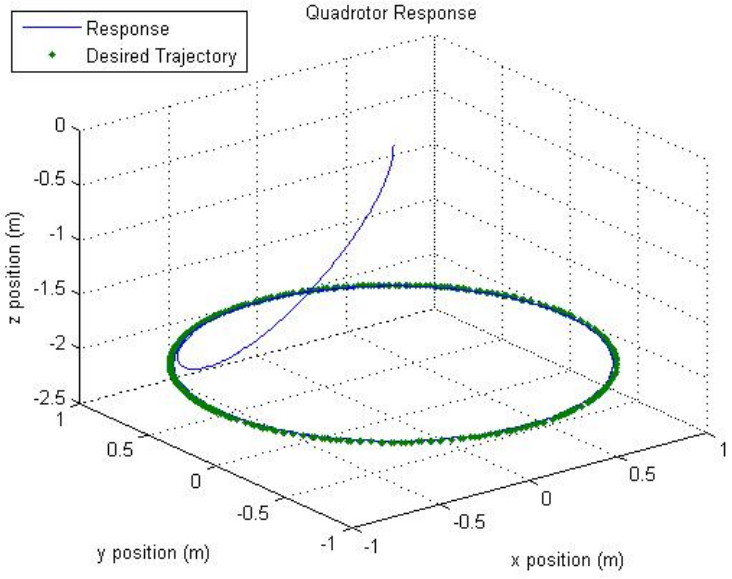


Figure 4-9: Trajectory Response under PD

4.6 Gain Scheduling Based PD Controller

To overcome the shortcomings of the linear PD controller in its ability to only operate in the linear near hover region, a gain scheduling based PD controller is proposed. The theory behind gain scheduling is developing a set of controllers for different operating points and switching between these controllers depending on the operating point of the system [45]. In this work, a family of PD controllers will be developed, each PD controller having different controller gains and will be able to stabilize the quadrotor system in a certain range of operation. Gain scheduling will then be used to choose an appropriate controller from the family of developed PD controllers. This approach renders the classical PD controller an adaptive controller since the controller's parameters are adapting to different operating conditions. Similar to the previously implemented controllers, GA was used to acquire the control gains, that would result in the least possible settling time, for the family of PD controllers are different operating points. The acquired gains were used in a look up table fashion in the developed MATLAB/Simulink model. The Block Diagram for the developed Gain Scheduling based PD controller is shown in Figure 4-10.

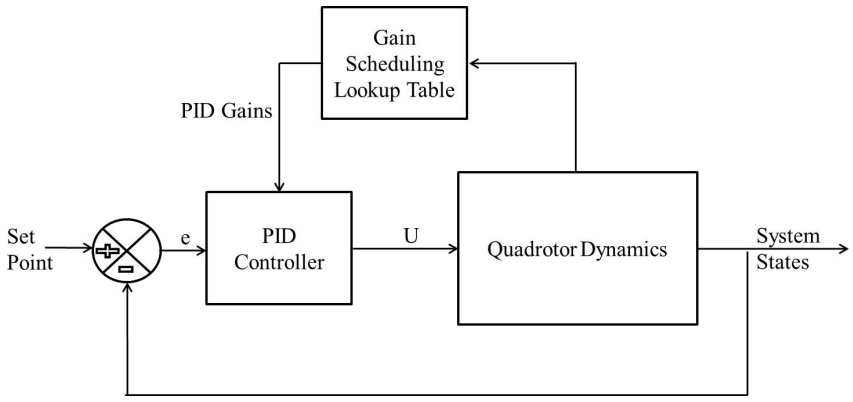


Figure 4-10: Gain Scheduling Block Diagram

4.7 Gain Scheduling Based PD Controller Simulation Results

4.7.1 Altitude Controller

GA was used to tune the parameters of the PD controller for the system, the parameters for different desired altitudes are shown in Table 4.2 together with the resulted settling time for the system. In order to show the strength of the developed gain

Table 4.2: Altitude Gain Scheduling Based PD Controller Gains and Results

Desired Altitude	k_p	k_d	Settling Time
1 m	8.91	3.75	0.98 sec
2 m	5.97	3.07	1.24 sec
3 m	4.67	2.71	1.42 sec
4 m	8.77	3.64	1.25 sec
5 m	5.06	2.79	1.51 sec
6 m	6.10	3.04	1.51 sec
7 m	5.14	2.79	1.64 sec
8 m	6.24	3.21	1.69 sec
9 m	4.64	2.67	1.82 sec
10 m	5.69	3.13	1.82 sec

scheduling based PD controller, it was tested to follow a wide range trajectory unlike the step input that was used in the classical PD. The response shown in Figure 4-11 compares the performance of the classical PD to the gain scheduled PD.

4.7.2 Attitude Controller

For the roll and pitch control, GA was also used to find the controller gains for a set of operating points. Table 4.3 shows the operating points together with their controller gains and performance. Figure 4-12 shows a comparison between the performance of

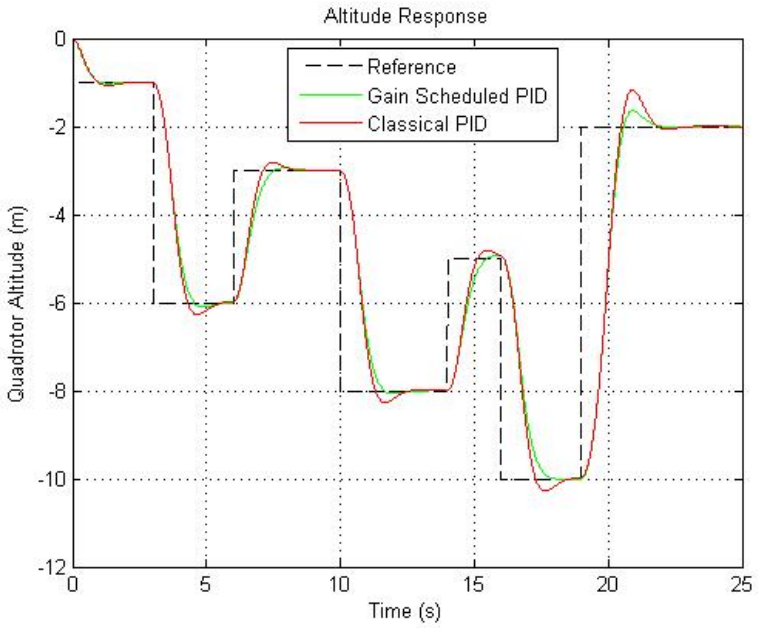


Figure 4-11: Altitude Response

the gain scheduled PD controller and the classical PD controller in following a varying trajectory.

4.7.3 Heading Controller

Similar to the attitude controller, a look up table was synthesised for the heading controller. The controller gains and their respective performances at multiple operating points are shown in Table 4.4 and the response is shown in Figure 4-13.

The control inputs for the previous trajectories are shown in Figure 4-14.

Table 4.3: Attitude Gain Scheduling Based PD Controller Gains and Results

Desired Attitude	k_p	k_d	Settling Time
2°	6.29	0.694	0.26 sec
4°	5.89	0.675	0.27 sec
6°	7.10	0.737	0.24 sec
8°	7.04	0.742	0.25 sec
10°	4.25	0.573	0.31 sec
12°	5.69	0.661	0.27 sec
14°	5.90	0.678	0.27 sec
16°	5.24	0.637	0.28 sec
18°	3.05	0.486	0.37 sec
20°	5.40	0.657	0.29 sec

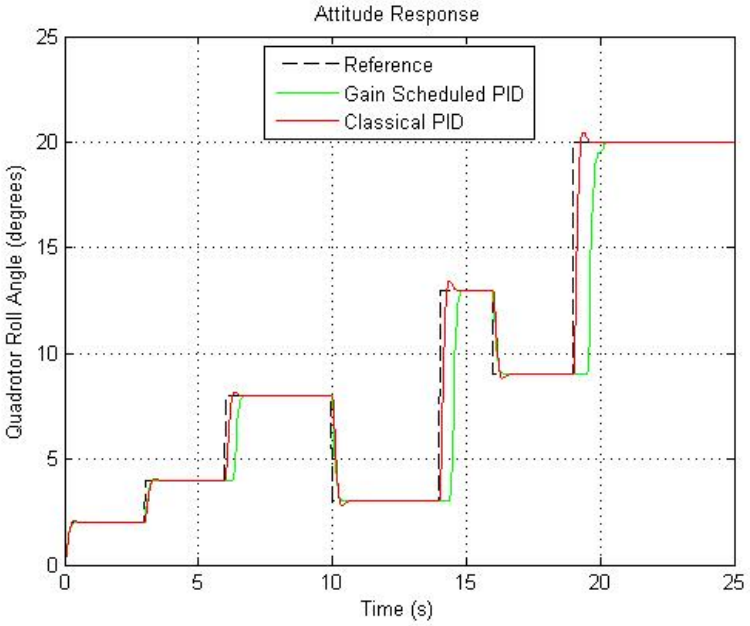


Figure 4-12: Attitude Response

Table 4.4: Heading Gain Scheduling Based PD Controller Gains and Results

Desired Heading	k_p	k_d	Settling Time
2°	6.25	0.921	0.38 sec
4°	3.28	0.669	0.53 sec
6°	4.96	0.809	0.52 sec
8°	3.75	0.697	0.60 sec
10°	4.00	0.775	0.64 sec
12°	3.94	0.816	0.69 sec
14°	4.51	0.991	0.75 sec
16°	2.27	0.570	0.825 sec
18°	3.31	0.821	0.84 sec
20°	4.70	1.20	0.88 sec

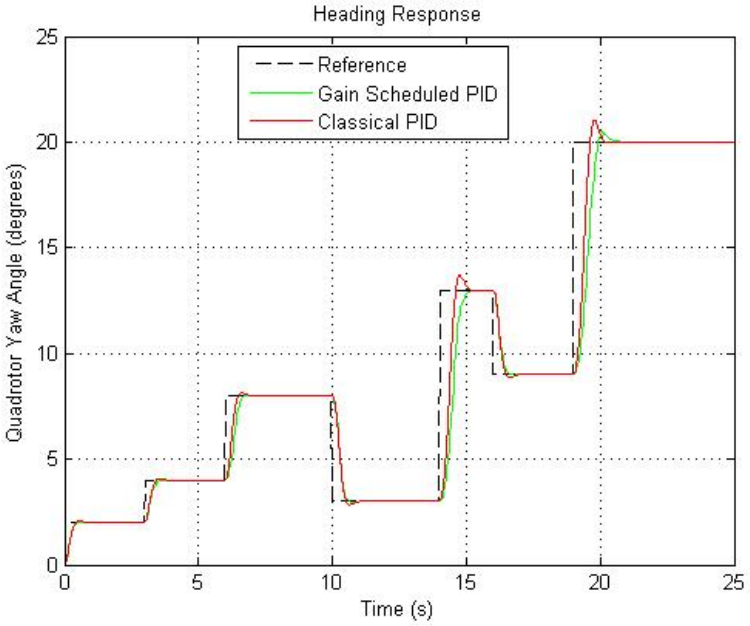
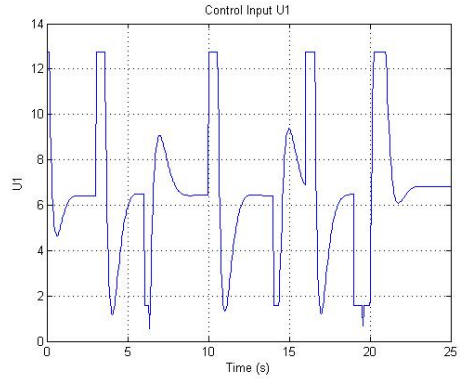
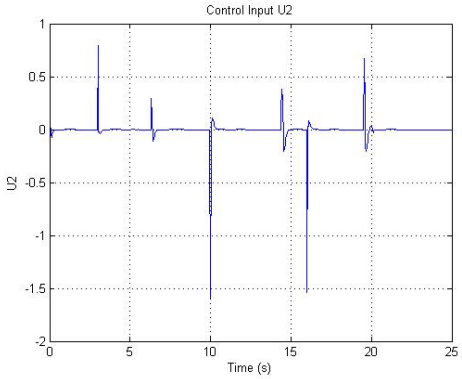


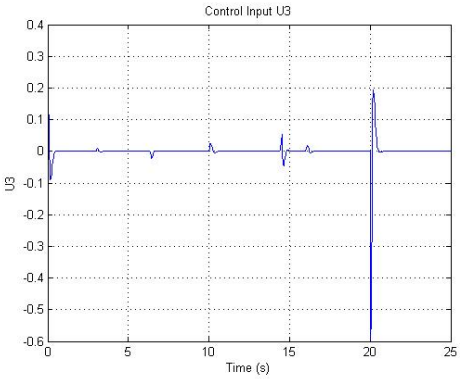
Figure 4-13: Heading Response



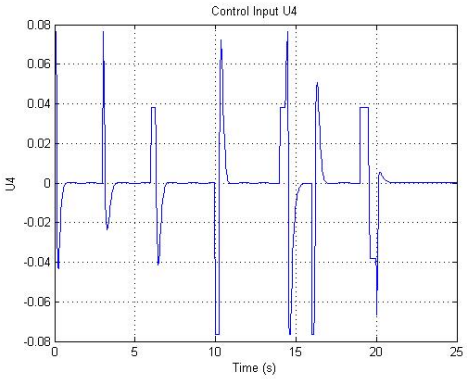
(a) Control Input U_1



(b) Control Input U_2



(c) Control Input U_3



(d) Control Input U_4

Figure 4-14: Gain Sheduling based PD Control Inputs

4.8 Sliding Mode Controller

Since the quadrotor system is a nonlinear type system, we proposed using a nonlinear Sliding Mode Controller (SMC) to control the states of the quadrotor.

4.8.1 Introduction to SMC

A SMC is a type of Variable Structure Control (VSC). It uses a high speed switching control law to force the state trajectories to follow a specified, user defined surface in the states space and to maintain the state trajectories on this surface [46]. The control law for a SMC consists of two parts as per Equation (4.12); a corrective control part and an equivalent control part. The corrective control function is to compensate any variations of the state trajectories from the sliding surface in order to reach it. The equivalent control on the other hand, makes sure the time derivative of the surface is maintained to zero, so that the state trajectories would stay on the sliding surface.

$$U(t) = U_c(t) + U_{eq}(t) \tag{4.12}$$

Where

$U(t)$	Control Law
$U_c(t)$	Corrective Control
$U_{eq}(t)$	Equivalent Control

A block diagram showing the SMC is shown in Figure 4-15.

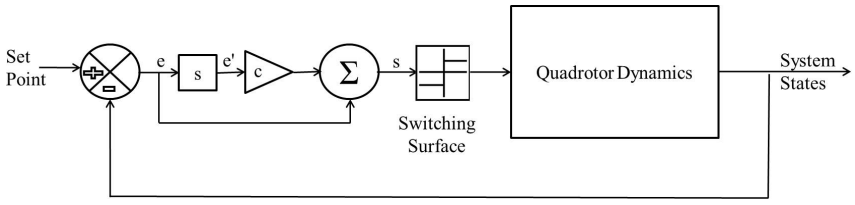


Figure 4-15: SMC Block Diagram

4.8.2 Attitude Control

The sliding mode attitude controller is based on the approach used by Bouabdallah and Siegwart [15].

4.8.2.1 Roll Controller

The SMC is used to track a reference trajectory for the roll angle. The error in the roll is defined as,

$$e = \phi_d - \phi \tag{4.13}$$

The sliding surface is defined as,

$$s = c_1 e + \dot{e} \tag{4.14}$$

where c_1 is a constant that has to be greater than zero. This format is a common format for the sliding surface in tracking problems. The derivative of the sliding surface defined in Equation (4.14) with the substitution of Equation (4.13) is formulated as the following,

$$\begin{aligned} \dot{s} &= c_1 \dot{e} + \ddot{e} \\ &= c_1(\dot{\phi}_d - \dot{\phi}) + \ddot{\phi}_d - \ddot{\phi} \end{aligned} \tag{4.15}$$

A Lyapunov function is then defined to be,

$$V(e, s) = \frac{1}{2}(e^2 + s^2) \tag{4.16}$$

Based on the Lyapunov function, an exponential reaching law is proposed for the sliding mode controller as follows

$$\dot{s} = -k_1 sgn(s) - k_2 s \tag{4.17}$$

where

$$sgn(s) = \begin{cases} -1 & \text{if } s < 0; \\ 1 & \text{if } s > 0. \end{cases}$$

and k_1 and k_2 are design constants. To satisfy the sliding mode condition $s\dot{s} < 0$, limits has to be set on k_1 and k_2 such as $k_1 > 0$ and $k_2 > 0$. By equating the proposed reaching law (4.17) to the derivative of the sliding surface in Equation (4.15) and substituting $\ddot{\phi}$ by its definition from Equation (3.42), the control input U_2 is calculated to be,

$$U_2 = \frac{1}{b_1}[k_1 sgn(s) + k_2 s + c_1(\dot{\phi}_d - \dot{\phi}) + \ddot{\phi}_d + a_2\dot{\theta}\Omega_r - a_1\dot{\theta}\dot{\psi}] \tag{4.18}$$

4.8.2.2 Pitch Controller

Following exactly the same steps as the roll controller, the control input U_3 responsible of generating the pitch rotation θ is calculated to be,

$$U_3 = \frac{1}{b_2}[k_1 sgn(s) + k_2 s + c_1(\dot{\theta}_d - \dot{\theta}) + \ddot{\theta}_d - a_4\dot{\phi}\Omega_r - a_3\dot{\phi}\dot{\psi}] \tag{4.19}$$

4.8.2.3 Yaw Controller

Following the same steps as the roll and pitch controller, the control input U_4 responsible of producing the yaw rotation ψ is calculated to be,

$$U_4 = \frac{1}{b_3}[k_1 sgn(s) + k_2 s + c_1(\dot{\psi}_d - \dot{\psi}) + \ddot{\psi}_d - a_5\dot{\phi}\dot{\theta}] \tag{4.20}$$

4.8.3 Altitude Control

As what has been done with the development of the attitude controllers, an altitude SMC is implemented. The error is defined as,

$$e = z - z_d \tag{4.21}$$

where z_d is the desired altitude of the quadrotor. The sliding surface is defined as previously

$$s = c_1 e + \dot{e} \tag{4.22}$$

Also, c_1 has to be a positive constant. The derivative of the sliding surface is then equated to the exponential reaching law as follows

$$-k_1 sgn(s) - k_2 s = c_1(\dot{z} - \dot{z}_d) + \ddot{z} - \ddot{z}_d \tag{4.23}$$

Substituting $\ddot{z}$ by its definition from equation (3.52), the control input U_1 is calculated to be

$$U_1 = \frac{m}{\cos\phi\cos\theta}[k_1 sgn(s) + k_2 s + c_1(\dot{z} - \dot{z}_d) + g - \ddot{z}_d] \tag{4.24}$$

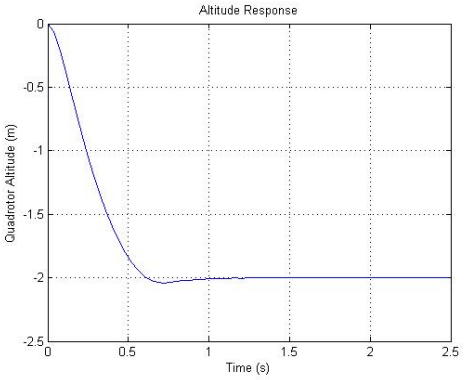
4.9 SMC Simulation

As for the previously implemented PD controller, GA was used to find the design parameters (c_1, k_1 and k_2) for the implemented SMCs to achieve the least settling time for the system. Table 4.5 show the GA generated design parameters and the resulting settling time and overshoot of the system. The response is shown in Figure 4-16. The problem of chattering is clear in the response plots and the problem to overcome it will be addressed in Section 4.10.

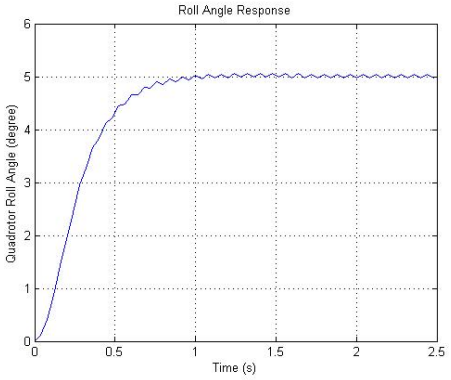
Table 4.5: SMC Results

	Desired Value	c	k_1	k_2	Settling Time	Overshoot
Altitude (z)	2 m	6.98	2.66	6.64	0.57 sec	2%
Attitude (ϕ and θ)	5°	4.68	1.99	1.80	0.8 sec	1.9%
Heading (ψ)	5°	5.24	1.72	4.33	0.74 sec	1.7%

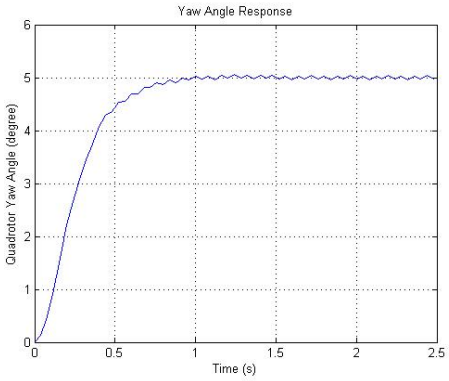
Due to the symmetry of the quadrotor, the results for the pitch controller were



(a) Altitude Response



(b) Attitude Response



(c) Heading Response

Figure 4-16: SMC Controller Simulation Response

exactly the same as that of the roll and accordingly the GA produced the same control gains.

4.10 SMC Chattering Reduction

As shown in Figure 4-16, there is a clear chattering effect which is a common outcome of a SMC due to its switching nature. The presence of the *sgn* term in the SMC's control law makes it a discontinuous controller. Figure 4-17 shows that whenever the value of the surface s is positive, the control law works to decrease the trajectory to reach the sliding surface ($s = 0$) at point a. Ideally it should continue sliding on the surface once hitting it, but due to the delay between the change of sign of s and the the change in the control action, the trajectory passes the surface to the side

$(s < 0)$. Accordingly, the control law works to drive the trajectory again to $(s = 0)$, yet it passes it and this causes the famous chattering effect. The main drawbacks of chattering are that it causes the excitation of unmodeled system dynamics that yields a possible instability of the system. In addition to that it is associated with a high power consumption and possible actuator damage. These drawbacks make the SMC hard to be implemented on real systems [29].

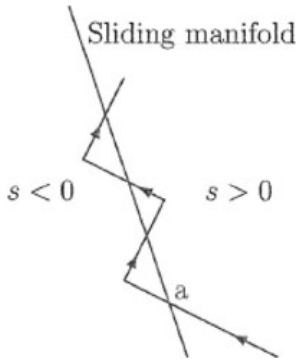


Figure 4-17: The Chattering Effect [29]

The chattering can be observed more clearly in the plot of the derivative of the sliding surface $\dot{s}$ for the altitude shown in Figure 4-18.

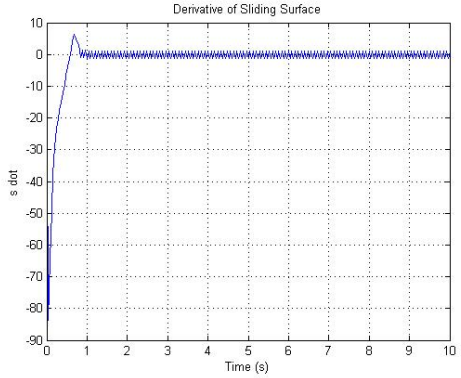


Figure 4-18: Derivative of Sliding Surface for $c_1 = 6.98$, $k_1 = 2.66$ and $k_2 = 6.64$

4.10.1 Re-tuning using GA

In a trial to eliminate chattering effect, we proposed a mathematical formula to have a numerical value for the chattering based on calculating the difference in areas under

the desired and actual response graphs starting from the settling time until the end of the simulation time.

$$chat = \int_{T_s}^{T_{end}} \|\phi - \phi_d\| \, \mathrm{d}t \tag{4.25}$$

where

T_s Settling Time for Roll Response

T_{end} Simulation End Time

By visual inspection, Equation (4.25) did not provide accurate means of numerically measuring the chattering of the SMC. Hence, another method was proposed and verified for measuring the chattering which depends on the time derivative of the sliding surface s. The lower $\dot{s}$ is, the lower the chattering.

$$chat = \max_{T_s \to T_{end}} \dot{s} \tag{4.26}$$

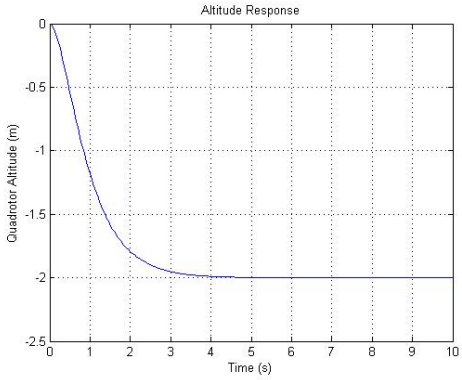
4.10.1.1 Simulations

The result of Equation (4.26) was used as an objective function for the GA to calculate the design gains. Table 4.6 shows the GA generated design parameters that result in the least chattering and based on these parameters the resulting settling time and overshoot of the system. The response is shown in Figure 4-19.

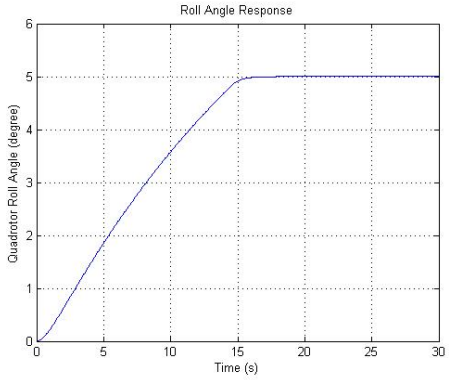
Table 4.6: SMC Results with Minimal Chattering

	Desired Value	c	k_1	k_2	Settling Time
Altitude (z)	2 m	2.84	0.0011	1.49	3.11 sec
Attitude (ϕ and θ)	5°	1.49	0.0062	0.0474	15 sec
Heading (ψ)	5°	4.94	0.0016	0.7334	5.25 sec

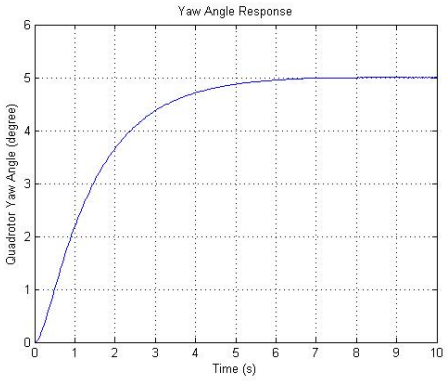
Using the chattering as the objective function of the GA led to the desired reduction of chattering but as a counter effect, the settling time of the system was largely



(a) Altitude Response



(b) Attitude Response



(c) Heading Response

Figure 4-19: SMC Controller Simulation Response with Chattering Reduction

affected. It resulted in an increase in the altitude's settling time from 0.57 sec to 3.11 sec. Figure 4-20 shows the derivative of the surface clearly indicating the absence of chattering. The attitude's settling time increased from 0.8 sec to 15 sec. While that of the heading increased from 0.74 to 5.25 sec.

For a final trial to optimize the traditional SMC, compromising between its performance in terms of the settling time and chattering, the objective function given to the GA was both the settling time and chattering of the roll angle. The control gains were found to be $c_1 = 1.98$, $k_1 = 0.0010$ and $k_2 = 1.33$ and they resulted in a settling time of 3.57 sec and a minimal chattering effect as shown in Figure 4-21.

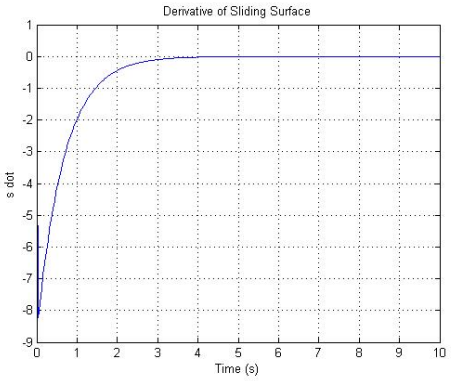


Figure 4-20: Derivative of Altitude's Sliding Surface

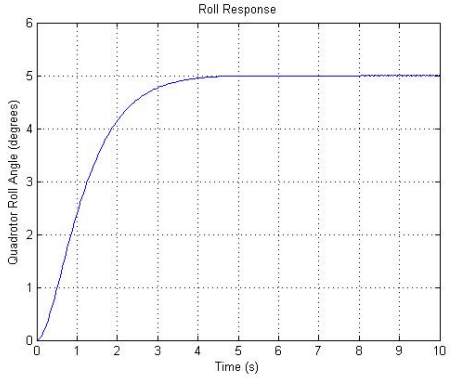


Figure 4-21: Roll Response for $c_1 = 1.98$, $k_1 = 0.0010$ and $k_2 = 1.33$

4.10.2 Discontinuous to Continuous Control Law

Another approach was proposed to get rid of the chattering effect. As stated earlier, the reason behind chattering is the discontinuity of the control law due to the presence of *sgn* function which is a discontinuous function. Replacing *sgn* with the saturation function *sat* will change the control law to be a continuous one instead of discontinuous. Figure 4-22 shows graphs of both functions. The *sat* function is defined by,

$$sat(s) = \begin{cases} s & \text{if } |s| \leq 1; \\ sgn(s) & \text{if } |s| > 1. \end{cases}$$

Thus, to implement this modification on the SMCs for our system, the $sgn(s)$ terms in Equations (4.18), (4.19), (4.20) and (4.24) should be replaced by $sat(s/\epsilon)$.

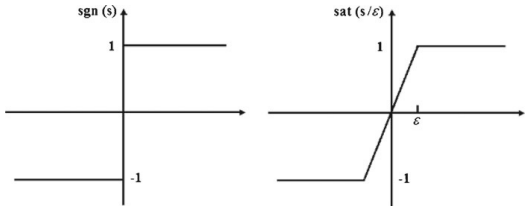
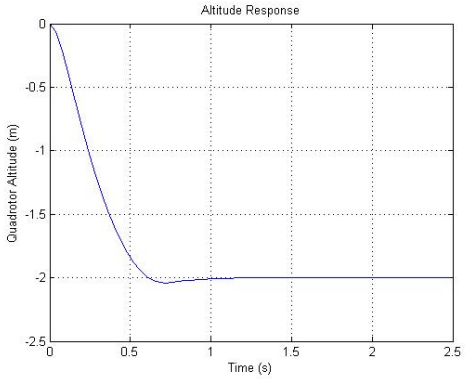


Figure 4-22: *sgn* vs. *sat* Functions [29]

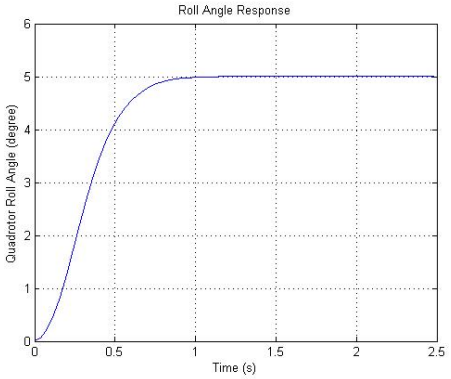
Where ϵ is a constant that determines the slope of the line between 1 and -1, this region is called the boundary region or boundary layer. Higher values ϵ means a thicker boundary layer and thus an increase in the error [47].

4.10.2.1 Simulations

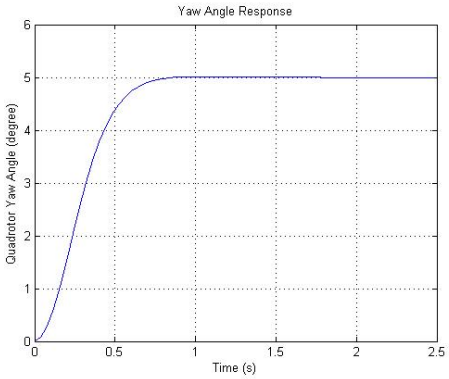
Using the obtained control parameters in Table 4.5 with the modified control laws (replacing $sgn(s)$ by $sat(s/\epsilon)$ in Equations (4.18), (4.19), (4.20) and (4.24)), chattering was eliminated as shown in Figure 4-23, the response graphs for altitude, attitude and heading respectively. Also, the control inputs are shown in Figure 4-24. Moreover, the quadrotor was commanded to follow a circular trajectory and its response graph is shown in Figure 4-25



(a) Altitude Response

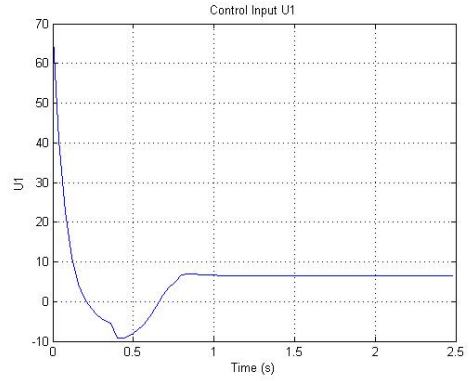


(b) Attitude Response

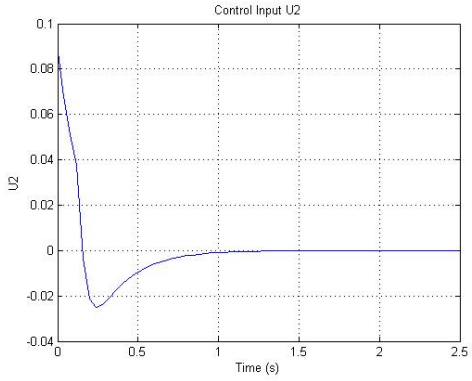


(c) Heading Response

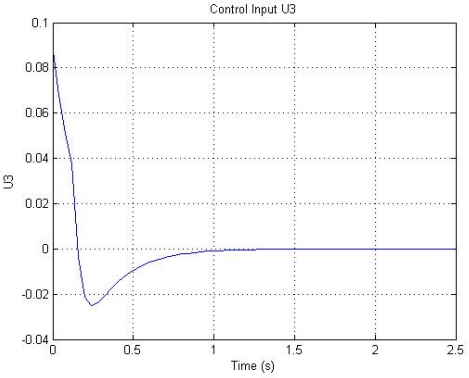
Figure 4-23: Modified SMC Controller Simulation Response



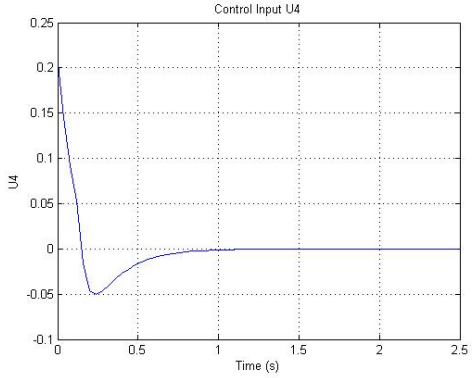
(a) Control Input U_1



(b) Control Input U_2



(c) Control Input U_3



(d) Control Input U_4

Figure 4-24: SMC Control Inputs

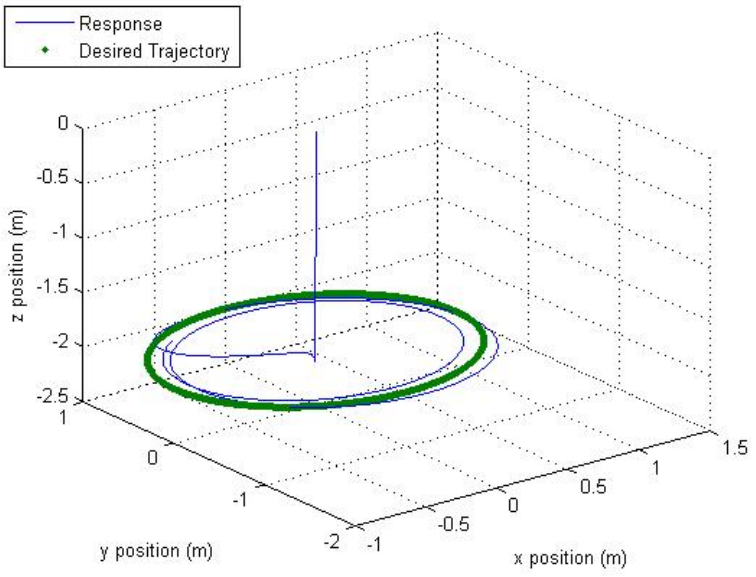


Figure 4-25: Trajectory Response under SMC

4.11 Backstepping Controller

In this section, a Backstepping controller is used to control the attitude, heading and altitude of the quadrotor. The Backstepping controller is based on the state space model derived in (3.53).

4.11.1 Introduction to Backstepping

Backstepping is a recursive control algorithm that works by designing intermediate control laws for some of the state variables. These state variables are called "virtual controls" for the system [48]. Unlike other control algorithms that tend to linearize nonlinear systems such as the feedback linearization algorithm, backstepping does not work to cancel the nonlinearities in the system. This leads to more flexible designs since some of the nonlinear terms can contribute to the stability of the system. An example of such terms that add to the stability of the system are state variables taking the form of negative terms with odd powers (e.g. $-x^3$), they provide damping for large values of x [48, 49].

4.11.2 Attitude and Heading Control

The backstepping controller implemented to control the quadrotor's orientation is based on the control approaches proposed in [33] and [15].

4.11.2.1 Roll Controller

The first two states of the state space model in Equation (3.53) are the roll angle and its rate of change. Extracting those we get:

$$\dot{x}_1 = x_2 \tag{4.27}$$

$$\dot{x}_2 = x_4 x_6 a_1 - x_4 \Omega_r a_2 + b_1 U_2 \tag{4.28}$$

The roll angle subsystem is in the strict feedback form (only the last state is a function of the control input U_2) which makes it easy to pick a positive definite Lyapunov

function for it,

$$V_1 = \frac{1}{2} z_1^2 \tag{4.29}$$

where z_1 is the error between the desired and actual roll angle defined as follows,

$$z_1 = x_{1d} - x_1 \tag{4.30}$$

The time derivative of the Lyapunov function defined in Equation (4.29) is derived to be,

$$\dot{V}_1 = z_1 \dot{z}_1 \tag{4.31}$$

$$= z_1(\dot{x}_{1d} - \dot{x}_1) \tag{4.32}$$

and from Equation (4.27) this can be rewritten as,

$$\dot{V}_1 = z_1(\dot{x}_{1d} - x_2) \tag{4.33}$$

According to Krasovskii–LaSalle principle, the system is guaranteed to be a stable system if the time derivative of a positive definite Lyapunov function is negative semi-definite [48]. To achieve that, we choose a positive definite bounding function $W_1(z) = c_1 z_1^2$ to bound $\dot{V}_1$ as in Equation (4.34). This choice of $W_1(z)$ is also a common choice for a bounding function for strict feedback systems [48].

$$\dot{V}_1 = z_1(\dot{x}_{1d} - x_2) \leq -c_1 z_1^2 \tag{4.34}$$

where c_1 is a positive constant. To satisfy this inequality the virtual control input can be chosen to be,

$$(x_2)_{desired} = \dot{x}_{1d} + c_1 z_1 \tag{4.35}$$

Defining a new error variable z_2 to be the deviation of the state x_2 from its desired

value,

$$z_2 = x_2 - \dot{x}_{1d} - c_1 z_1 \tag{4.36}$$

Rewriting Lyapunov's function time derivative $\dot{V}_1$ in the new coordinate (z_1, z_2) we get,

$$\begin{aligned}
\dot{V}_1 &= z_1 \dot{z}_1 \\
&= z_1(\dot{x}_{1d} - x_2) \\
&= z_1(\dot{x}_{1d} - (z_2 + \dot{x}_{1d} + c_1 z_1)) \\
&= -z_1 z_2 - c_1 z_1^2
\end{aligned} \tag{4.37}$$

Note that the presence of the term $z_1 z_2$ in $\dot{V}_1$ may not lead to a negative semi-definite time derivative but this will be taken care of in the next iteration of the backstepping algorithm. The next step is to augment the first Lyapunov function V_1 with a quadratic term in the second error variable z_2 to get a positive definite V_2,

$$V_2 = V_1 + \frac{1}{2} z_2^2 \tag{4.38}$$

with time derivative,

$$\begin{aligned}
\dot{V}_2 &= \dot{V}_1 + z_2 \dot{z}_2 \\
&= -z_1 z_2 - c_1 z_1^2 + z_2(\dot{x}_2 - \ddot{x}_{1d} - c_1 \dot{z}_1)
\end{aligned} \tag{4.39}$$

Choosing the positive definite bounding function to be $W_2(z) = -c_1 z_1^2 - c_2 z_2^2$ where c_2 is a positive definite and substituting by the value of $\dot{x}_2$ from equation (4.28) leads to the following inequality,

$$\dot{V}_2 = -z_1 z_2 - c_1 z_1^2 + z_2(x_4 x_6 a_1 - x_4 \Omega_r a_2 + b_1 U_2 - \ddot{x}_{1d} - c_1 \dot{z}_1) \leq -c_1 z_1^2 - c_2 z_2^2 \tag{4.40}$$

Solving the last inequality, the control input U_2 can be written as,

$$U_2 = \frac{1}{b_1}(-c_2 z_2 + z_1 - x_4 x_6 a_1 + x_4 \Omega_r a_2 + \ddot{x}_{1d} + c_1 \dot{x}_{1d} - c_1 x_2) \tag{4.41}$$

4.11.2.2 Pitch Controller

The pitch controller is derived in the same manner as the roll controller. The states used are,

$$\dot{x}_3 = x_4 \tag{4.42}$$

$$\dot{x}_4 = x_2 x_6 a_3 + x_2 \Omega_r a_4 + b_2 U_3 \tag{4.43}$$

And the error in pitch is defined as $z_3 = x_{3d} - x_3$ leading to a positive definite Lyapunov function,

$$V_3 = \frac{1}{2} z_3^2 \tag{4.44}$$

with time derivative,

$$\begin{aligned} \dot{V}_3 &= z_3 \dot{z}_3 \\ &= z_3(\dot{x}_{3d} - x_4) \end{aligned} \tag{4.45}$$

Choosing the bounding function to be $W_3(z) = -c_3 z_3^2$ with c_3 a positive constant, the desired x_4 state is,

$$(x_4)_{desired} = \dot{x}_{3d} + c_3 z_3 \tag{4.46}$$

and the error in state x_4 is,

$$z_4 = x_4 - \dot{x}_{3d} - c_3 z_3 \tag{4.47}$$

Rewriting the Lyapunov function's time derivative

$$\begin{aligned}\dot{V}_3 &= z_3\dot{z}_3 \\ &= z_3(\dot{x}_{3d} - (z_4 + \dot{x}_3 d + c_3 z_3)) \\ &= -z_3 z_4 - c_3 z_3^2 \end{aligned} \tag{4.48}$$

Augmenting the previous Lyapunov function with a quadratic term in the error variable z_4,

$$V_4 = V_3 + \frac{1}{2}z_4^2 \tag{4.49}$$

Defining a new bounding function to be $W_4(z) = -c_3 z_3^2 - c_4 z_4^2$ with c_4 a positive constant, the following inequality can be reached,

$$\dot{V}_4 = -z_3 z_4 - c_3 z_3^2 + z_4(\dot{x}_4 - \ddot{x}_{3d} - c_3 z_3) \leq -c_3 z_3^2 - c_4 z_4^2 \tag{4.50}$$

replacing x_4 with its definition from equation (4.43) and solving for U_3. The roll angle control input is found to be,

$$U_3 = \frac{1}{b_2}(-c_4 z_4 + z_3 - x_2 x_6 a_3 - x_2 \Omega_r a_4 + \ddot{x}_{3d} + c_3 \dot{x}_3 d - c_3 x_4) \tag{4.51}$$

4.11.2.3 Yaw Controller

Following exactly the same steps as the roll and pitch controllers, the control input for the yaw angle is derived to be,

$$U_4 = \frac{1}{b_3}(-c_6 z_6 + z_5 - x_2 x_4 a_5 + \ddot{x}_{5d} + c_5 \dot{x}_{5d} - c_5 x_6) \tag{4.52}$$

with

$$z_5 = x_{5d} - x_5 \tag{4.53}$$

$$z_6 = x_6 - \dot{x}_{5d} - c_5 z_5 \tag{4.54}$$

and c_5 and c_6 are positive constants.

4.11.3 Altitude Control

For the altitude controller, the control input U_1 is derived in the same manner as U_2, U_3, U_4 to be

$$U_1 = \frac{m}{\cos x_1 \cos x_3}(-z_7 + g - \ddot{x}_{7d} - c_7\dot{x}_{7d} + c_7 x_8 + c_8 z_8) \tag{4.55}$$

with

$$z_7 = x_{7d} - x_7 \tag{4.56}$$

$$z_8 = x_8 - \dot{x}_{7d} - c_7 z_7 \tag{4.57}$$

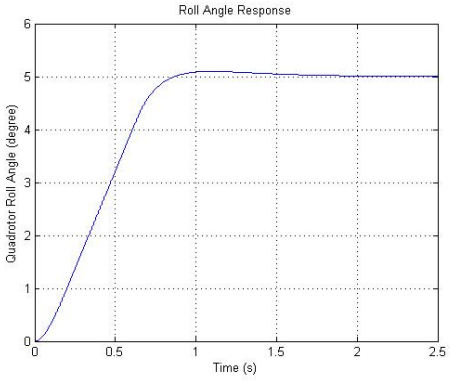
and c_7 and c_8 are positive constants.

4.12 Backstepping Controller Simulation

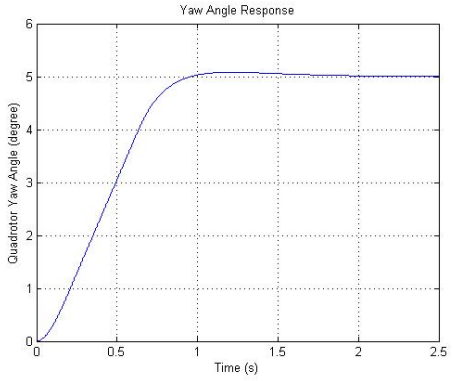
The control inputs U_1 through U_4 derived in sections 4.11.2 and 4.11.3 were added to the previously implemented simulation model and similar to the PD and the SMC controllers, GA was used to tune the parameters of the Backstepping controllers. The parameters to be tuned are c_1 through c_8. The objective function used for the GA was the settling time of the system. Table 4.7 shows the optimized parameters acquired from the GA and the resulting settling time for the attitude, heading and altitude of the quadrotor. Figure 4-26 shows the response when running the simulation with the constants in Table 4.7 and Figure 4-27 shows the control inputs.

Table 4.7: Backstepping Controller Constants and Results

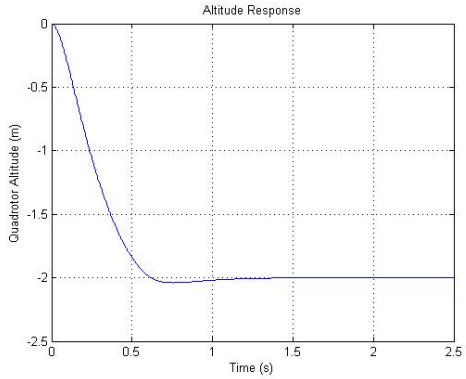
	Desired Value	$c_1/c_5/c_7$	$c_2/c_6/c_8$	Settling Time	Overshoot
Attitude (ϕ and ψ)	5°	5.52	3.40	0.80 sec	1.9%
Heading (ψ)	5°	3.07	4.71	0.86 sec	1.6%
Altitude (z)	2 m	6.11	7.96	0.58 sec	2 %



(a) Attitude Response

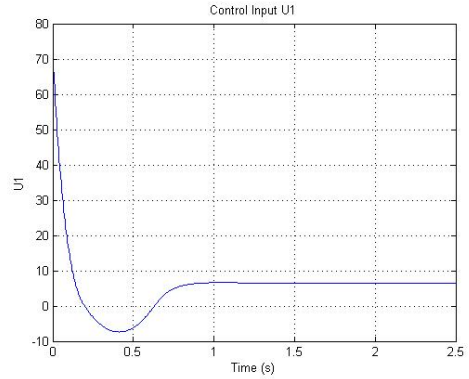


(b) Heading

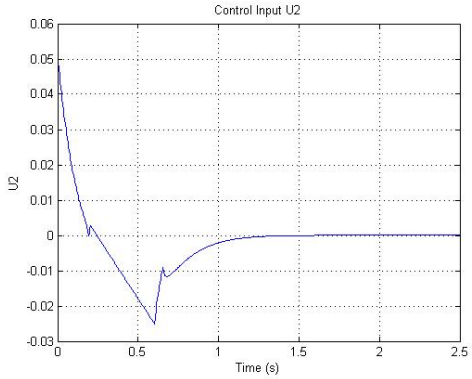


(c) Altitude Response

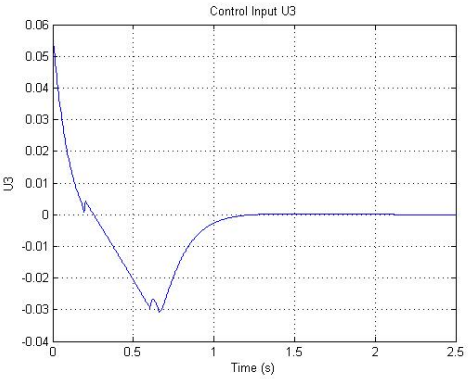
Figure 4-26: Backstepping Controller Simulation Response



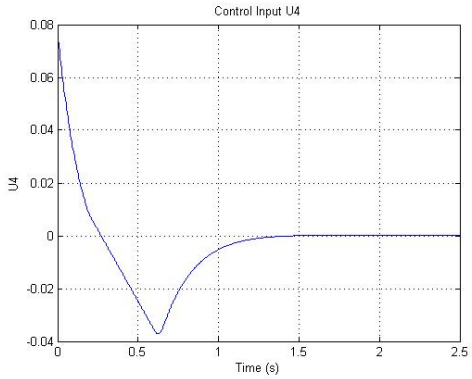
(a) Control Input U_1



(b) Control Input U_2



(c) Control Input U_3



(d) Control Input U_4

Figure 4-27: Backstepping Control Inputs

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 5

Results Discussion

5.1 Varying Trajectory

To be able to compare fairly between the four implemented control techniques, the response graph of the system under the effect of each the four controllers was plotted superimposed on one another. Figure 5-1 shows the altitude response while Figures 5-2 and 5-3 show the attitude and heading responses respectively.

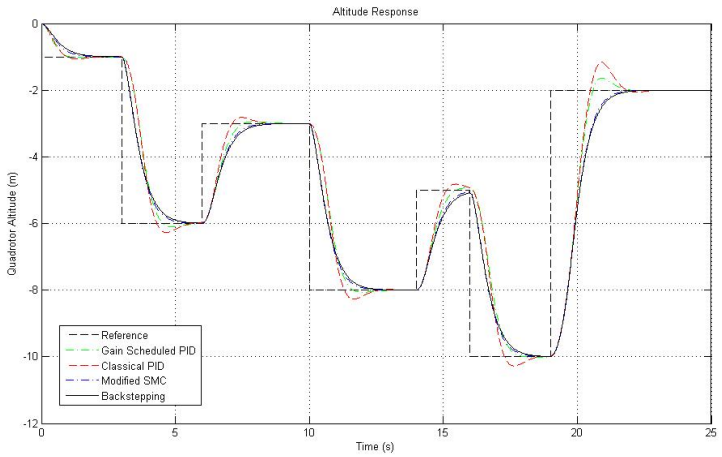


Figure 5-1: Altitude Response

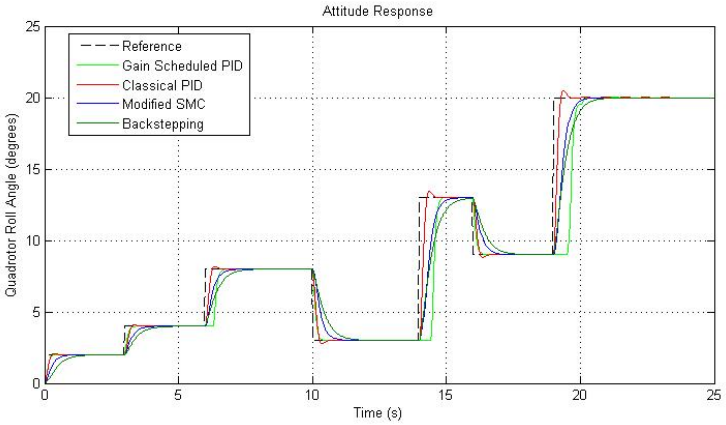


Figure 5-2: Attitude Response

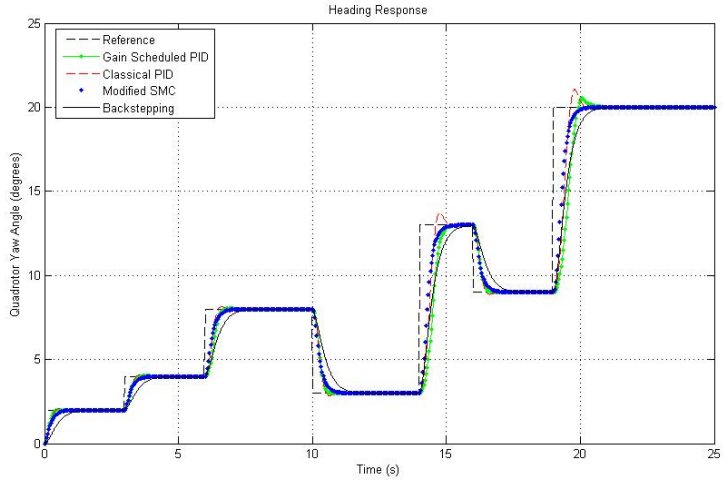
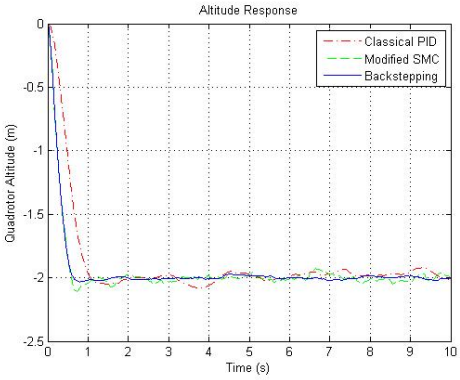


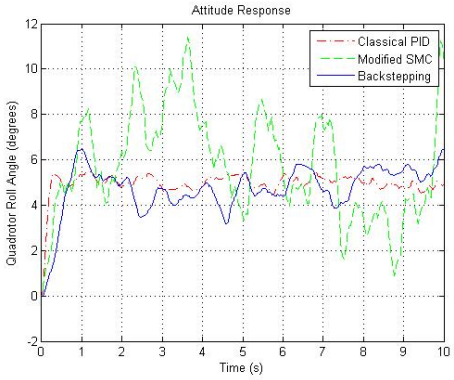
Figure 5-3: Heading Response

5.2 Performance in a Windy Environment

Disturbance was then added to the quadrotor model in the form of additional forces and moments to give the effect of operating the quadrotor in a windy environment. The forces were added to the right hand side of the system's translational equation of motion (Equation (3.14)) as Gaussian noise with zero mean and with a maximum value of 1 N. The added moments were also added to the right hand side of the system's rotational equation of motion (Equation (3.4)) as Gaussian noise with zero mean and a maximum value of 0.5 Nm. The system was commanded to follow a certain desired altitude and attitude. The performance of the system under the effect



(a) Attitude Response with a desired attitude of 2m



(b) Attitude Response with a desired attitude of 5°

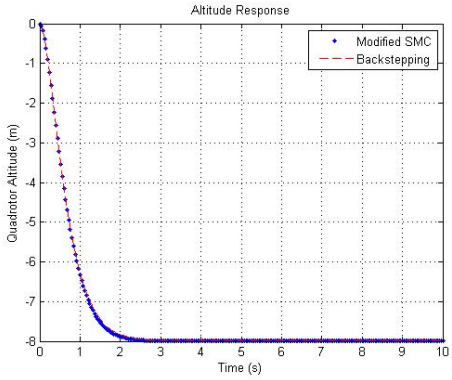
Figure 5-4: System Response in a Windy Environment

of wind is shown in Figure 5-4 for PD, SMC and Backstepping.

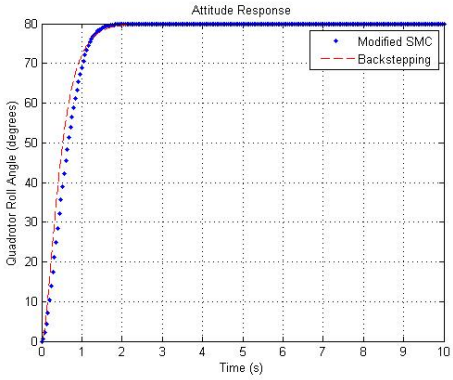
5.3 Nonhovering Operation

In order to operate the system outside its linear region (the hovering condition) a nonlinear controller has to be used. Applying the SMC and the Backstepping controller, the system response for the altitude, attitude and heading is shown in Figure 5-5. Since the SMC and the Backstepping controller are nonlinear controllers, their tuning is not affected by the operating region. Thus, the control gains acquired in Tables 4.5 and 4.7 were used.

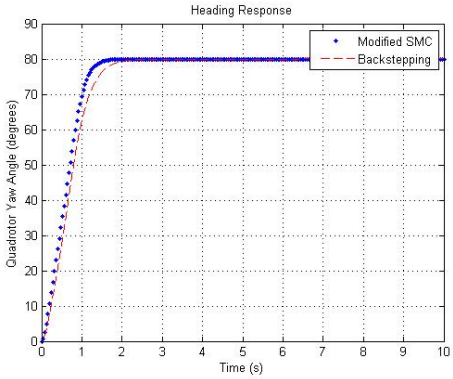
When the PD controller was used to operate the system outside its linear region (more than 20°) of orientation, the system went unstable.



(a) Attitude Response with a desired attitude of 8 m



(b) Attitude Response with a desired attitude of 80°



(c) Heading Response with a desired heading of 80°

Figure 5-5: System Response When Operated Outside the Linear Region

5.4 Summary of Findings

Linear Operation The four employed controllers developed to control the quadrotor model under consideration gave comparable dynamic performances in terms of settling time and overshoot when they were deployed in near hover stabilization of the quadrotor. When first implemented, the SMC resulted in an undesirable chattering effect which was very notable in the attitude response unlike the altitude's. This chattering effect was then eliminated by using a modified version for the control law rendering the response chattering free. Tables 5.1 and 5.2 show a quantitative comparison between the performance of the PD, SMC and Backstepping controllers in terms of the settling time and overshoot of the system's response respectively.

Nonlinear Operation When the controllers were used outside of the linear region (away from hover), the PD controller failed to stabilize the system due to the fact that PD comes out of a family of linear controllers. On the other hand, the SMC and the Backstepping controller were able to stabilize the system with a good dynamic performance as shown in Tables 5.3 and 5.4. The developed gain scheduled PD controller was also not able to stabilize the system as it is also based on the linear PD controller.

PD One notable advantage of the PD controller over the other implemented controllers is that its control law is not a function of the system parameters, it is only a function of the state error and its derivative making the control law less computationally intensive and easier to implement. Also, the controller is less prone to slight variations or uncertainties in system parameters.

Gain Scheduled PD A gain scheduling based PD controller is essentially a PD controller with its gains tuned for a different set of operating conditions, thus its performance at following a fixed value trajectory is the exactly the same as the performance of the classical PD. On the other hand, the gain scheduled PD controller performs better than the traditional PD controller when following a varying trajectory, which is a more practical or realistic application for a quadrotor UAV. A quadrotor is more likely to be commanded follow a changing trajectory rather than flying to fixed place in space and hovering or maintaining its position there.

A worth mentioning drawback of the Gain Scheduling algorithm is the criticality of the switching time, the switching from a set of controller gains to the other has to be done in infinitesimally small time to guarantee a good performance. This is a critical issue in some quadrotor applications that mainly rely on Gain Scheduling such as the load drop applications, if the switching is not done once the load is dropped, the quadrotor might overshoot and go unstable.

Control Effort Comparison Comparing the four developed controllers in terms of their generated control signals, U_1 through U_4, PD comes out to be the most energy efficient (as shown in Figure 4-8) followed by Backstepping (Figure 4-27) and then SMC (Figure 4-24). Gain scheduling based PD controllers (with their control signal shown in Figure 4-14) suffer from spikes in the control signals due to the sudden transition between one set of controller gains to the other, these sudden transitions will probably lead to system instability if the controller was implemented on a real system. To overcome this effect, a smoothing filter needs to be used to result in a smooth transition before sending the control signals to the actuators. Also, changing the switching variable to be the error of the state instead of the desired set point might lead to decreasing the presence of the spikes in the control signals.

Windy Environment When operating in a windy environment, which was simulated by Gaussian noise of zero mean, the performance of the modified SMC suffered a huge degradation. This is due to two reasons; the first of them is that the presence of noise excites the chattering phenomenon and cancels the effect of the boundary layer. The second reason for the degradation of the performance of the SMC is that the system model is changed by the added forces and moments that simulate the windy environment. As a consequence to that change, the time derivative of the Lyapunov functions is not guaranteed to be negative semi-definite anymore thus causing the system to be unstable. While the performance of the PD and the Backstepping controllers was comparable in controlling the quadrotor's altitude, yet Backstepping also suffered a slight performance degradation in stabilizing the quadrotor's attitude. This is due to the fact that the Backstepping's reaching law design also depends on a Lyapunov function and the added forces and moments cause its time derivative not to be guaranteed negative semi-definite. Figure 5-6 shows the time derivative of the Lyapunov function under the effect of wind, while Figure 5-7 shows it in the no wind condition.

Choice of Controller The choice of the controller to be used will depend mainly on the application, if the quadrotor is to be operating near a hovering condition, a PD controller will be sufficient to stabilize it. On the other hand, if it will be performing tough acrobatic maneuvers thus operating outside its linear region, a SMC or a Backstepping controller should be employed. The environment too will help make the choice, for example simulations showed that PD and Backstepping controllers were more robust to disturbances which might come in the form of a windy environment.

Table 5.1: System Settling Time Response Under Different Controllers in the Linear Region

Controller	Altitude 2 m	Attitude 5°	Heading 5°
PD	1.3 sec	**0.3 sec**	**0.42 sec**
SMC	**0.57 sec**	0.8 sec	0.74 sec
Backstepping	0.58 sec	0.77 sec	0.86 sec

Table 5.2: System Overshoot Response Under Different Controllers in the Linear Region

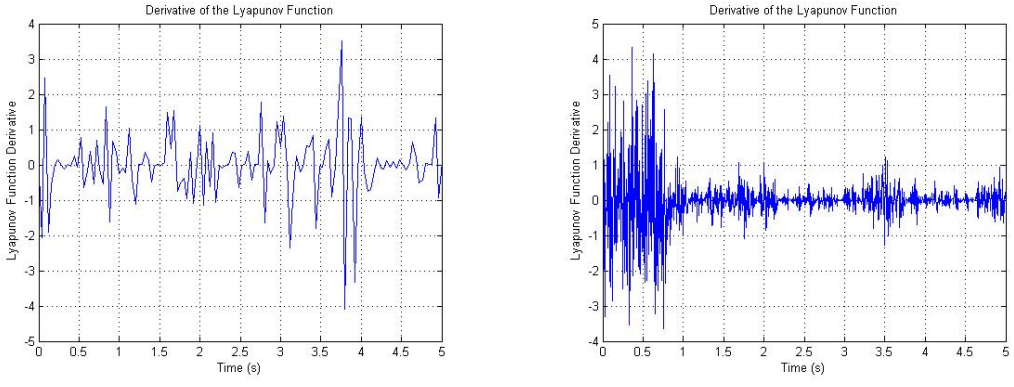
Controller	Altitude 2 m	Attitude 5°	Heading 5°
PD	**1.4%**	2%	1.9%
SMC	2%	1.9%	**1.7%**
Backstepping	2%	**1.8%**	2%

Table 5.3: System Settling Time Response Under Different Controllers in the Nonlinear Region

Controller	Altitude 8 m	Attitude 80°	Heading 80°
SMC	1.90 sec	1.42 sec	1.37 sec
Backstepping	2.02 sec	1.54 sec	1.75 sec

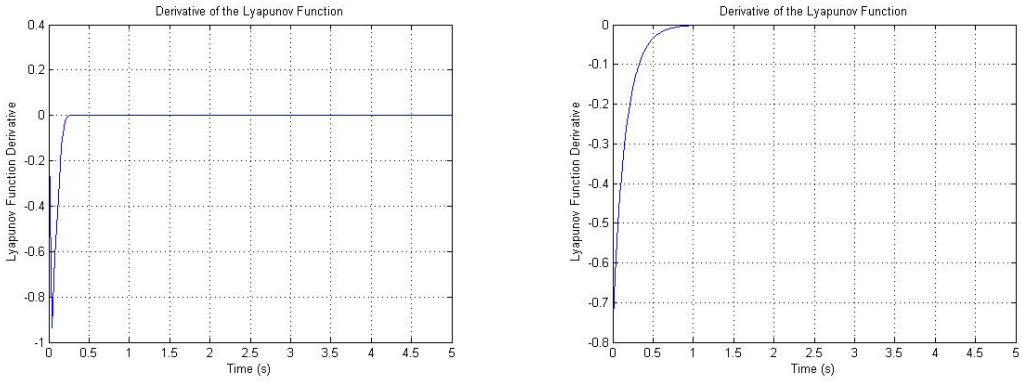
Table 5.4: System Overshoot Response Under Different Controllers in the Nonlinear Region

Controller	Altitude 8 m	Attitude 80°	Heading 80°
SMC	0 %	0 %	0 %
Backstepping	0 %	0 %	0 %



(a) SMC (b) Backstepping

Figure 5-6: Lyapunov Function Derivative Under the Effect of Wind



(a) SMC (b) Backstepping

Figure 5-7: Lyapunov Function Derivative in no Wind Condition

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 6

Conclusion

The goal of this work was to derive a mathematical model for the quadrotor Unmanned Aerial Vehicle (UAV) and develop linear and nonlinear control algorithms to stabilize the states of the quadrotor, which include its altitude, attitude, heading and position in space and to verify the performance of these controllers with comparisons via computer simulations.

The mathematical model of a quadrotor UAV was developed in details including its aerodynamic effects and rotor dynamics which we found lacking in many literatures. Four control techniques were then developed and synthesized; a linear Proportional-Integral-Derivative Proportional-Derivative (PD) controller, a Gain Scheduling based PD controller, a nonlinear Sliding Mode Controller (SMC) and a nonlinear Backstepping controller. A complete simulation was then implemented on MATLAB/Simulink relying on the derived mathematical model of the quadrotor. The simulation environment was used to evaluate the mentioned controllers and compare their dynamic performances under different types of input conditions.

Tuning the parameters and constants of the four used controllers was done using Genetic Algorithm (GA) where the objective function was the dynamic response of the system in terms of its settling time and/or overshoot. The four controllers performed comparably in near hovering operation of the quadrotor in the range of $0 \sim 20^{\circ}$ of attitude and heading. The Gain Scheduling based PD controller gave a better

performance than the traditional PD controller when the quadrotor was commanded to follow a varying trajectory. The SMC and Backstepping controllers gave better performance outside the linear hovering region due to their nonlinear nature. The PD and Backstepping controllers gave better performance than all the other controllers when the effect of wind was added to the system. The wind effect was modeled as extra forces and moments on the quadrotor body.

For future work, we recommend testing the Gain Scheduled PD controller in load drop applications and actuators failure conditions and compare it to the three other controllers. Also, changing the Gain Scheduled PD to have the switching to be a function of the error and its derivative instead of the desired final value. This might enhance its performance in following a varying trajectory. One valuable addition would be the robustification of the developed control techniques against wind as this is a common problem with quadrotors control and our simulation results showed a huge degradation of the performance of the controllers when the system was exposed to wind. Moreover, in our work it was assumed that all the model parameters are known accurately without any uncertainties, which is not the case in reality, thus, developing adaptive control algorithms to count for the system uncertainties would enhance the performance of the quadrotor when operating in a real environment. Adding an integral action to the developed Backstepping controller will lead to the formulation of an adaptive control algorithm robust to system uncertainties. Moreover, sensors were assumed to be perfect which is not the case in reality, sensors modeling and noise need to be taken into consideration and checking the effect on system stability under the effect of the developed controllers. Last but not least, implementing the developed control techniques on a real quadrotor hardware to give a more fair comparison between their performances.

Appendix A

System Modeling Derivations

A.1 Kinematics Model

A.1.1 Rotation Matrix R

To describe the orientation of the quadrotor in space, 2 intermediate coordinate systems need to be defined; the vehicle-1 frame and the vehicle-2 frame together with the previously defined inertial frame and body frame [50].

The inertial frame is rotated about its y-axis by the yaw angle ψ to get the vehicle-1 frame. The transformation from the inertial frame to the vehicle-1 frame is given by

$$R_i^{v1} = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{A.1}$$

The notation R_i^{v1} indicates a rotation from frame F_i which is the inertial frame to frame F_v1 which is the vehicle-1 frame.

The resulting frame $v1$ is then rotated by the pitch angle θ around its y-axis to result in the vehicle-2 frame. The transformation from the vehicle-1 frame to the

vehicle-2 frame is given by

$$R_{v1}^{v2} = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \tag{A.2}$$

The last rotation is the rotation of the vehicle-2 frame about its x-axis to result in the body frame. The transformation from the vehicle-2 frame to the body frame is given by

$$R_{v2}^{b} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \tag{A.3}$$

Finally the rotation matrix or transformation from the inertial frame to the body frame is given by

$$\begin{aligned} R_i^b &= R_{v2}^b R_{v1}^{v2} R_i^{v1} \\ &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} c\theta c\psi & c\theta s\psi & -s\theta \\ s\psi s\theta c\psi - c\phi s\psi & s\phi s\theta s\psi + c\phi s\psi & s\phi c\theta \\ c\phi s\theta c\psi + s\phi s\psi & c\phi s\theta s\psi - s\phi c\psi & c\phi c\theta \end{bmatrix} \end{aligned} \tag{A.4}$$

where c and s denote cos and sin respectively. Note that due to the premultiplication rule of rotation matrices, the order of rotation (ψ followed by θ followed by ϕ) is opposite to that of the order of multiplication (ϕ followed by θ followed by ψ).

In order to get the rotation matrix that transforms the body frame to the inertial frame, the previous rotation matrix R_i^b is transposed, yielding

$$R = (R_i^b)^T = R_b^i$$

$$= \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\theta c\psi & c\phi s\theta s\psi - s\theta c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix} \tag{A.5}$$

A.1.2 Euler Rates

The transformation between the Euler rates $\dot{\eta} = [\dot{\phi}\ \dot{\theta}\ \dot{\psi}]^T$ and angular body rates $\omega = [p\ q\ r]^T$ shown in Equation (3.2) is derived as follows [50]

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = R(\dot{\phi}) \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + R(\phi)R(\dot{\theta}) \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + R(\phi)R(\theta)R(\dot{\psi}) \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} \tag{A.6}$$

Note that $\dot{\phi}$, $\dot{\theta}$ and $\dot{\psi}$ are small thus $R(\dot{\phi}) = R(\dot{\theta}) = R(\dot{\psi}) = I$, then

$$\begin{aligned} \begin{bmatrix} p \\ q \\ r \end{bmatrix} &= \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} \\ &= \begin{bmatrix} \dot{\phi} - \sin\theta\dot{\psi} \\ \cos\phi\dot{\theta} + \sin\phi\cos\theta\dot{\psi} \\ -\sin\phi\dot{\theta} + \cos\phi\cos\theta\dot{\psi} \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & -\sin\phi & \cos\phi\cos\theta \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \end{aligned} \tag{A.7}$$

THIS PAGE INTENTIONALLY LEFT BLANK

Appendix B

Quadrotor Parameters

This appendix contains the quadrotor parameters used in the simulations. These parameters are adopted from Bouabdallah's thesis [16].

Table B.1: Quadrotor Parameters and Constants

Parameter	Description	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	N s^2
K_M	Aerodynamic moment constant	7.5e-7	Nm s^2
R_{mot}	Motor circuit resistance	0.6	Ω
K_{mot}	Motor torque constant	5.2	mNm/A
K_t	Aerodynamic translation coefficient	diag(0.1, 0.1, 0.15)	Ns/m
K_t	Aerodynamic rotation coefficient	diag(0.1, 0.1, 0.15)	Nm s

THIS PAGE INTENTIONALLY LEFT BLANK

Appendix C

Listings

C.1 Quadrotor Parameters Initialization

```
clear
clc
close all
%% Ineria
Ixx=7.5e-3;
Iyy=7.5e-3;
Izz=1.3e-2;
% axel length
l=0.23;
%% rotor inertia
Jr=6e-5;
%% mass
m=0.650;
g=9.81;
%% aerodynamic force and moments constant
kf=3.13e-5;
km=7.5e-7;
%% aerodynamic coefficients
Krx=0;%0.1;
Kry=0;%;0.1;
```

```
21 Krz=0;%0.15;
22 Ktx=0;%0.1;
23 Kty=0;%0.1;
24 Ktz=0;%0.15;
25 %% constants calculations
26 a1=(Iyy-Izz)/Ixx;
27 a2=Jr/Ixx;
28 a3=(Izz-Ixx)/Iyy;
29 a4=Jr/Iyy;
30 a5=(Ixx-Iyy)/Izz;
31 b1=l/Ixx;
32 b2=l/Iyy;
33 b3=l/Izz;
34 %%  Rotor Dynamics
35 R_mot=0.6;  %Motor Circuit Resistance
36 K_mot=5.2; %Motor Torque Constant
37 %% Disturbances
38 noise_rot=0;
39 noise_trans=0;
40 %% PID Constants
41 ki_Z=0;
42 ki_Theta=0;
43 ki_Phi=ki_Theta;
44 ki_Psi=0;
45 %% desired velocities and accelerations
46 z_dot_d=0;
47 phi_dot_d=0;
48 theta_dot_d=0;
49 psi_dot_d=0;
50 phi_dot_dot_d=0;
51 theta_dot_dot_d=0;
52 psi_dot_dot_d=0;
53 z_dot_dot_d=0;
54 x_dot_d=0;
55 y_dot_d=0;
56 %% hover omega
```

```
omega_hover=sqrt( (m*g)/(4*kf) );
deltaU1_max=kf*4*omega_hover^2;
deltaU2_max=kf*omega_hover^2;
deltaU4_max=km*2*omega_hover^2;
omega_max=(90*100)*9000*(2*pi/(60));  %90% of 9000 rpm to rad/s
U1_max=kf*4*omega_max^2;
U2_max=kf*omega_max^2;
U4_max=km*2*omega_max^2;
```

C.2 GA Parameters Tuning

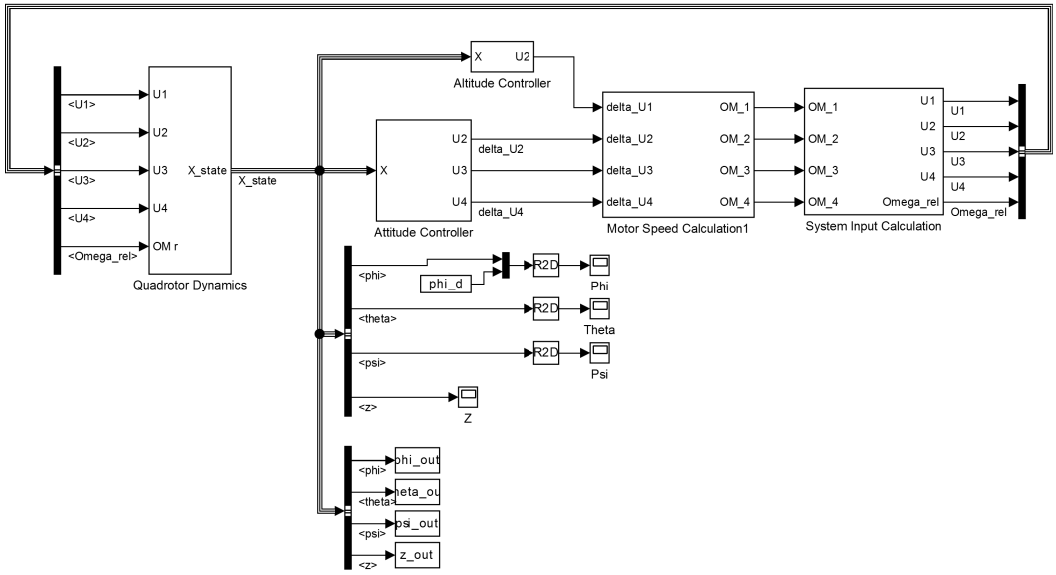
```
clear
clc
close all
initialize

%[k,Perf]=ga(@performancePID, 4)
%[k,Perf]=ga(@performancePIDGS, 2)
%[k,Perf]=ga(@performanceBackstepping, 2)
[k,Perf]=ga(@performanceSMC,3,[],[],[],[],[0.001;0.001;0.001],[])
```

C.3 GA Objective Function

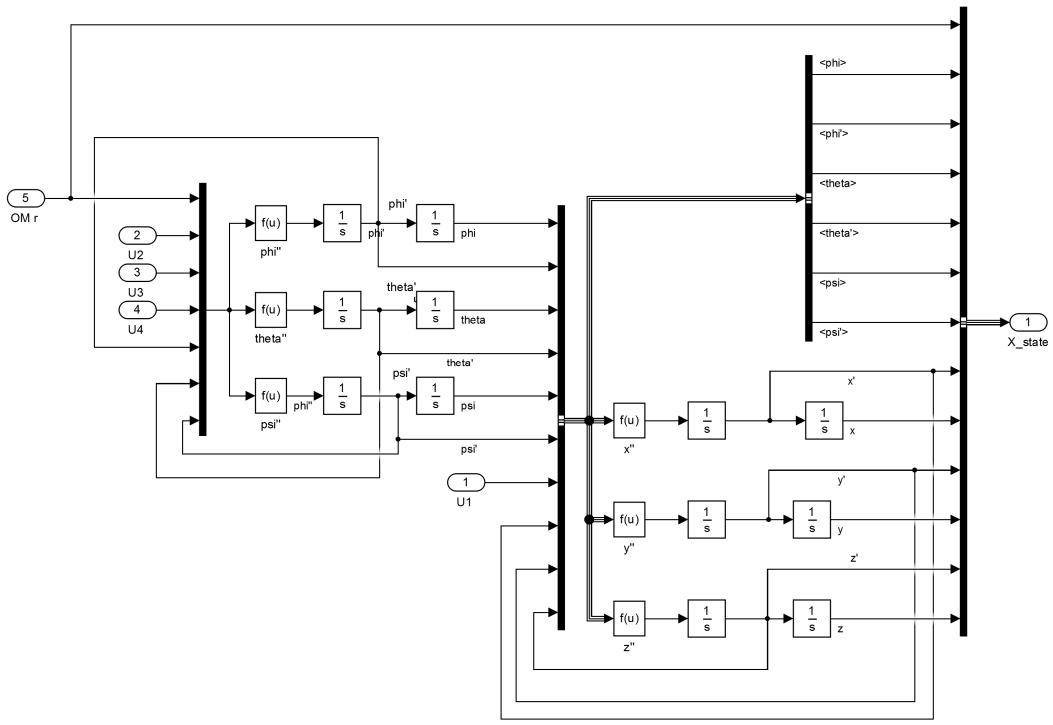
```
function perf=performanceSMC(k)

assignin('base', 'c_phi', 1)
assignin('base', 'k1_phi', 1)
assignin('base', 'k2_phi', 1)

assignin('base', 'c_theta', 1)
```

```
8  assignin('base', 'k1_theta', 1)
9  assignin('base', 'k2_theta', 1)
10
11 assignin('base', 'c_psi', 1)
12 assignin('base', 'k1_psi', 1)
13 assignin('base', 'k2_psi', 1)
14
15 assignin('base', 'c_z', k(1))
16 assignin('base', 'k1_z', k(2))
17 assignin('base', 'k2_z', k(3))
18
19 assignin('base','epsilon',0.1)
20
21 assignin('base', 'z_d', -12)
22 assignin('base', 'phi_d', 0) %(5*2*pi)/360
23 assignin('base', 'theta_d', 0)
24 assignin('base', 'psi_d',0)% (5*2*pi)/360)
25
26 sim modelSMC_ChatRed_altitude_orientation
27
28 S=stepinfo(z_out.signals.values,z_out.time);
29 settlingTz=S.SettlingTime;
30
31 perf=settlingTz
```

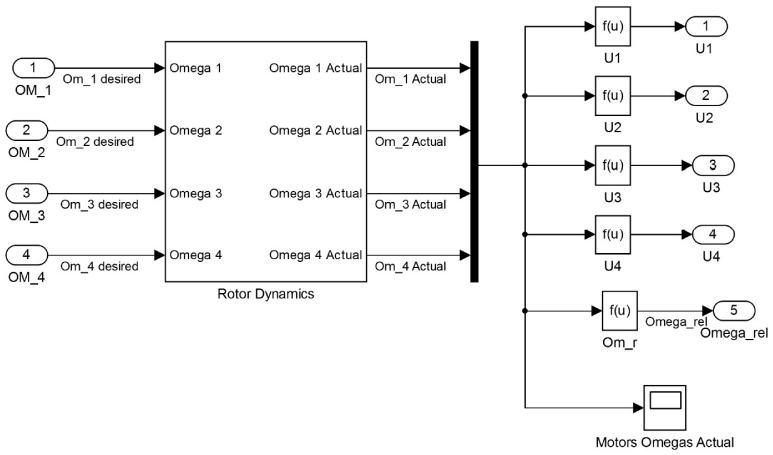
C.4 Quadrotor Model



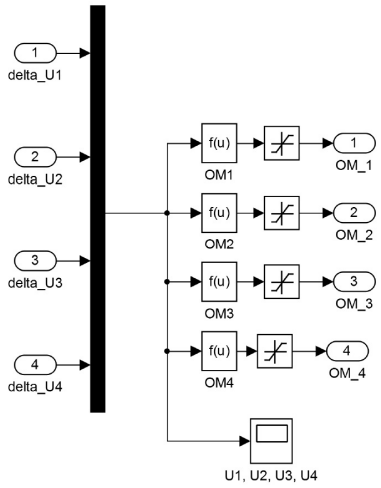
C.4.1 Quadrotor Dynamics



C.4.2 System Input Calculation

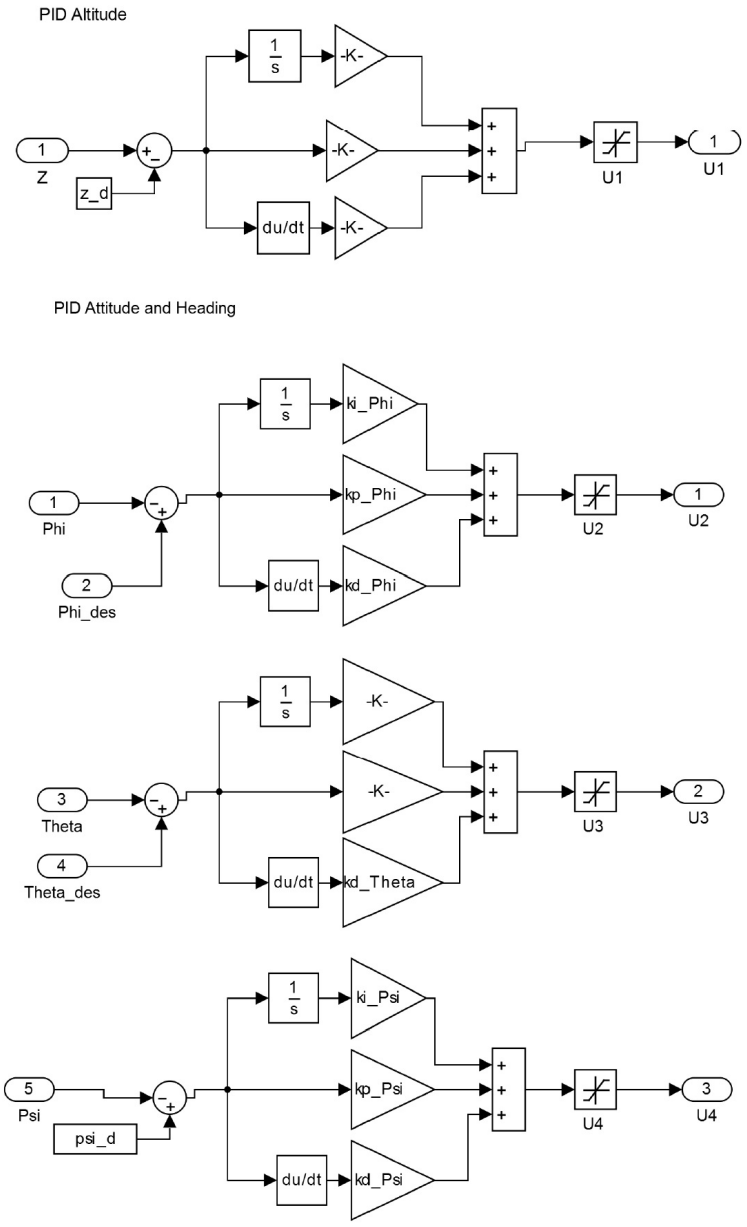


C.4.3 Motor Speed Calculation

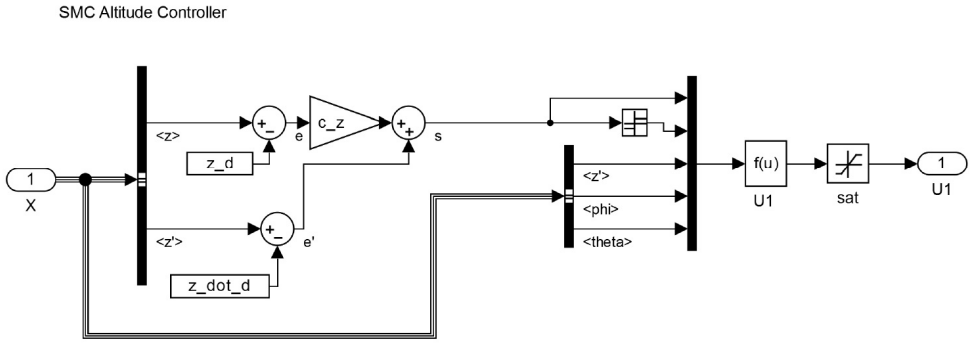


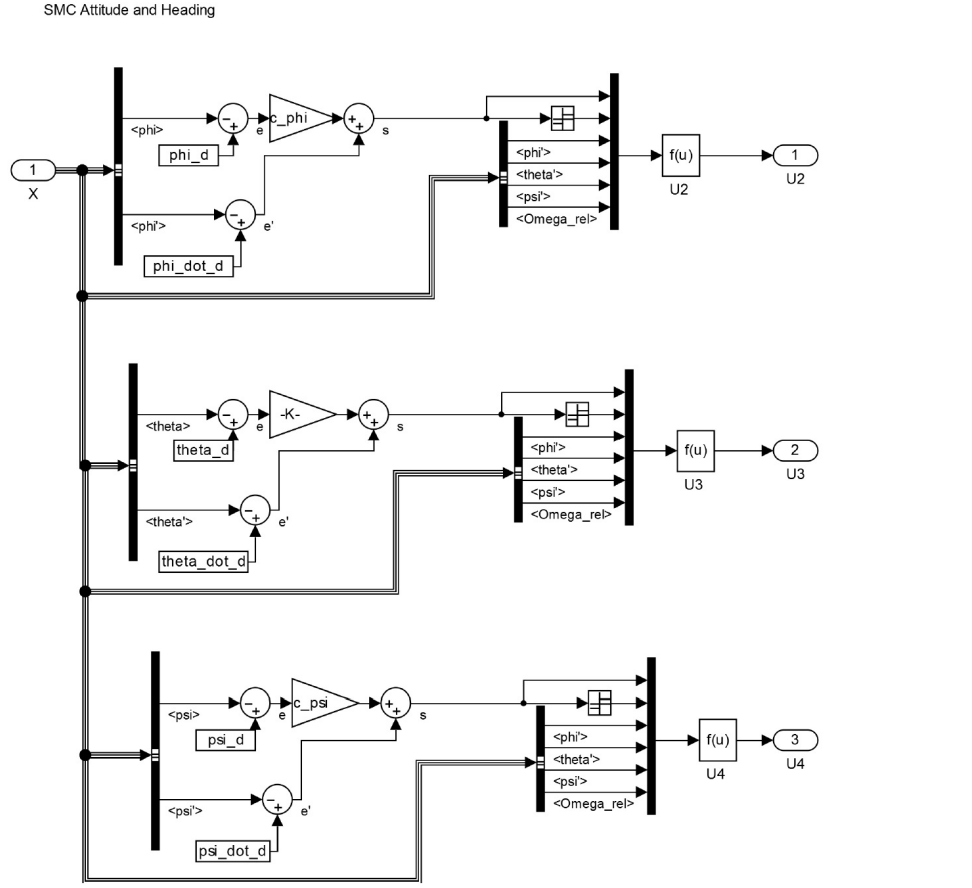
C.4.4 Controller Blocks

C.4.4.1 PID

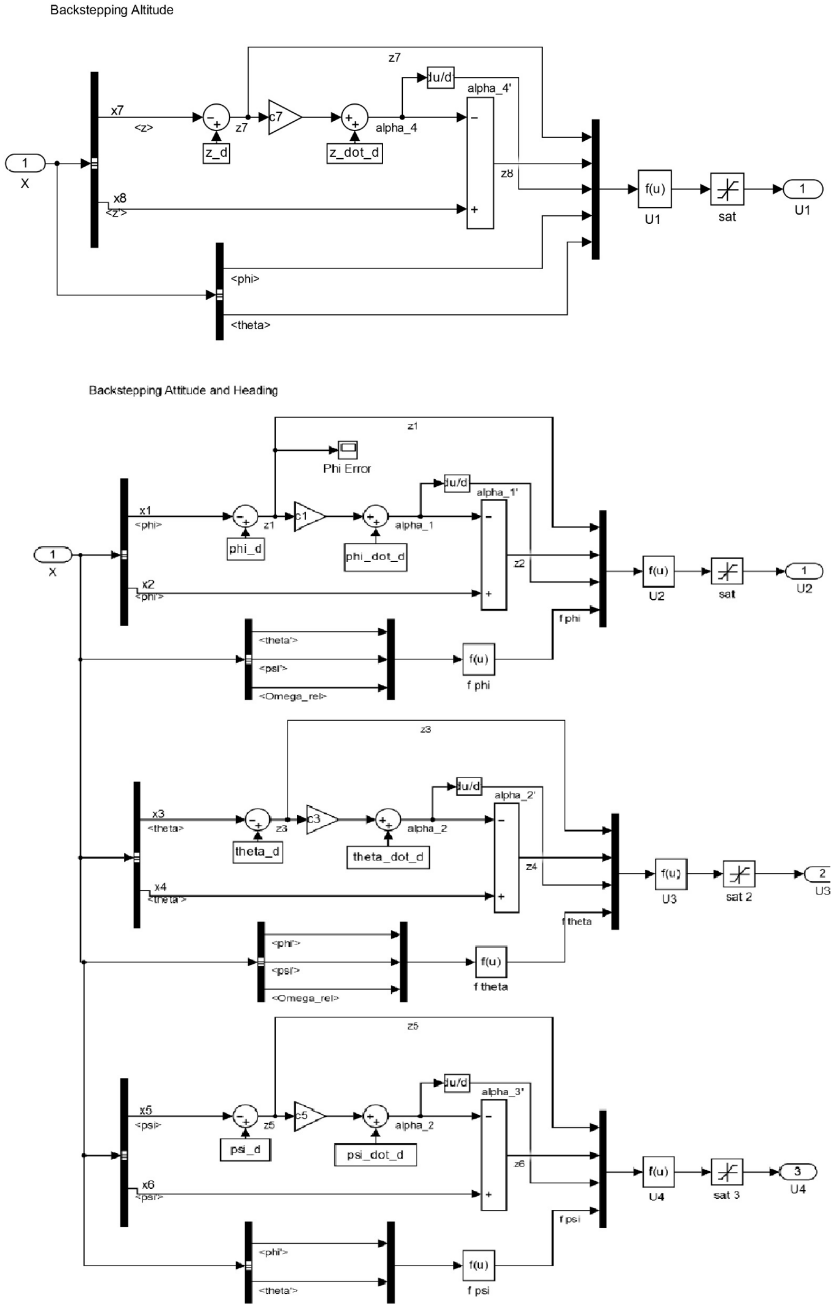


C.4.4.2 SMC


SMC Altitude Controller
1
X
<z>
z_d
e
c_z
s
<z'>
z_dot_d
e'
<z'>
<phi>
<theta>
f(u)
U1
sat
1
U1


SMC Attitude and Heading
1
X
<phi>
phi_d
c_phi
<phi'>
phi_dot_d
<phi'>
<theta'>
<psi'>
<Omega_rel>
f(u)
U2
1
U2
<theta>
theta_d
-K-
<theta'>
theta_dot_d
<phi'>
<theta'>
<psi'>
<Omega_rel>
f(u)
U3
2
U3
<psi>
psi_d
c_psi
<psi'>
psi_dot_d
<phi'>
<theta'>
<psi'>
<Omega_rel>
f(u)
U4
3
U4

C.4.4.3 Backstepping



Bibliography

[1] Hongning Hou, Jian Zhuang, Hu Xia, Guanwei Wang, and Dehong Yu. A simple controller of minisize quad-rotor vehicle. In *Mechatronics and Automation (ICMA), 2010 International Conference on*, pages 1701–1706, 2010. doi: 10.1109/ICMA.2010.5588802.

[2] Jinhyun Kim, Min-Sung Kang, and Sangdeok Park. Accurate modeling and robust hovering control for a quadrotor vtol aircraft. *Journal of Intelligent and Robotic Systems*, 57(1-4):9–26, 2010. ISSN 0921-0296. doi: 10.1007/s10846-009-9369-z. URL `http://dx.doi.org/10.1007/s10846-009-9369-z`.

[3] Farid Kendoul. Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems. *Journal of Field Robotics*, 29(2):315–378, 2012.

[4] Mark Willis Bailey. Unmanned aerial vehicle path planning and image processing for orthoimagery and digital surface model generation. 2012.

[5] A. Azzam and Xinhua Wang. Quad rotor arial robot dynamic modeling and configuration stabilization. In *Informatics in Control, Automation and Robotics (CAR), 2010 2nd International Asia Conference on*, volume 1, pages 438–444, 2010. doi: 10.1109/CAR.2010.5456804.

[6] Richard P Schwing. Unmanned aerial vehicles-revolutionary tools in war and peace. Technical report, DTIC Document, 2007.

[7] JR Wilson. UAV worldwide roundup 2009. *Aerospace America*, 47(4), 2009.

[8] Timothy H Cox, Christopher J Nagy, Mark A Skoog, Ivan A Somers, and R Warner. Civil UAV capability assessment. *NASA, Tech. Rep., draft Version*, 2004.

[9] Luis Rodolfo García Carrillo, Alejandro Enrique Dzul López, Rogelio Lozano, and Claude Pégard. Quad rotorcraft control, 2012.

[10] Zak Sarris and STN ATLAS. Survey of UAV applications in civil markets (june 2001). In *The 9 th IEEE Mediterranean Conference on Control and Automation (MED'01)*, 2001.

[11] S. Bouabdallah, A. Noth, and R. Siegwart. PID vs LQ control techniques applied to an indoor micro quadrotor. In *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 3, pages 2451–2456 vol.3, 2004. doi: 10.1109/IROS.2004.1389776.

[12] V. Mistler, A. Benallegue, and N.K. M'Sirdi. Exact linearization and noninteracting control of a 4 rotors helicopter via dynamic feedback. In *Robot and Human Interactive Communication, 2001. Proceedings. 10th IEEE International Workshop on*, pages 586–593, 2001. doi: 10.1109/ROMAN.2001.981968.

[13] C. Nicol, C.J.B. Macnab, and A. Ramirez-Serrano. Robust adaptive control of a quadrotor helicopter. *Mechatronics*, 21(6):927 – 938, 2011. ISSN 0957-4158. doi: http://dx.doi.org/10.1016/j.mechatronics.2011.02.007. URL `http://www.sciencedirect.com/science/article/pii/S0957415811000316`.

[14] Jun Li and Yuntang Li. Dynamic analysis and PID control for a quadrotor. In *Mechatronics and Automation (ICMA), 2011 International Conference on*, pages 573–578, 2011. doi: 10.1109/ICMA.2011.5985724.

[15] S. Bouabdallah and R. Siegwart. Backstepping and sliding-mode techniques applied to an indoor micro quadrotor. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 2247–2252, 2005. doi: 10.1109/ROBOT.2005.1570447.

[16] Samir Bouabdallah. Design and control of quadrotors with application to autonomous flying. Phd. thesis, Ecole Polytechnique Federale de Lausanne, 2007.

[17] Paul Edward Ian Pounds. Design, construction and control of a large quadrotor micro air vehicle. Phd. thesis, Australian National University, 2007.

[18] Jinpeng Yang, Zhihao Cai, Qing Lin, and Yingxun Wang. Self-tuning pid control design for quadrotor uav based on adaptive pole placement control. In *Chinese Automation Congress (CAC), 2013*, pages 233–237. IEEE, 2013.

[19] Guilherme V Raffo, Manuel G Ortega, and Francisco R Rubio. An integral predictive/nonlinear h control structure for a quadrotor helicopter. *Automatica*, 46(1):29–39, 2010.

[20] Jeremy H Gillula, Gabriel M Hoffmann, Haomiao Huang, Michael P Vitus, and Claire J Tomlin. Applications of hybrid reachability analysis to robotic aerial vehicles. *The International Journal of Robotics Research*, 30(3):335–354, 2011.

[21] A. Ataka, H. Tnunay, R. Inovan, M. Abdurrohman, H. Preastianto, A.I. Cahyadi, and Y. Yamamoto. Controllability and observability analysis of the gain scheduling based linearization for uav quadrotor. In *Robotics, Biomimetics, and Intelligent Computational Systems (ROBIONETICS), 2013 IEEE International Conference on*, pages 212–218, Nov 2013. doi: 10.1109/ROBIONETICS.2013.6743606.

[22] Mohammad Hadi Amoozgar, Abbas Chamseddine, and YM Zhang. Fault-tolerant fuzzy gain scheduled pid for a quadrotor helicopter testbed in the presence of actuator faults. In *IFAC Conference on Advances in PID Control, Brescia, Italy (March 2012)*, 2012.

[23] Iman Sadeghzadeh, Mahyar Abdolhosseini, and Youmin M Zhang. Payload drop application of unmanned quadrotor helicopter using gain-scheduled pid and model predictive control techniques. In *Intelligent Robotics and Applications*, pages 386–395. Springer, 2012.

[24] S.L. Waslander, G.M. Hoffmann, Jung Soon Jang, and C.J. Tomlin. Multi-agent quadrotor testbed control design: integral sliding mode vs. reinforcement learning. In *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 3712–3717, 2005. doi: 10.1109/IROS.2005.1545025.

[25] T. Madani and A. Benallegue. Backstepping control for a quadrotor helicopter. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 3255–3260, 2006. doi: 10.1109/IROS.2006.282433.

[26] Zheng Fang and Weinan Gao. Adaptive backstepping control of an indoor micro-quadrotor. *Research Journal of Applied Sciences*, 4, 2012.

[27] Hyeonbeom Lee, Suseong Kim, Tyler Ryan, and H Jin Kim. Backstepping control on se (3) of a micro quadrotor for stable trajectory tracking. In *Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on*, pages 4522–4527. IEEE, 2013.

[28] Hongtao Zhen, Xiaohui Qi, and Hairui Dong. An adaptive block backstepping controller for attitude stabilization of a quadrotor helicopter. *WSEAS Transactions on Systems & Control*, 8(2), 2013.

[29] Iván González, Sergio Salazar, and Rogelio Lozano. Chattering-free sliding mode altitude control for a quad-rotor aircraft: Real-time application. *Journal of Intelligent & Robotic Systems*, 73(1-4):137–155, 2014.

[30] Farid Kendoul, Zhenyu Yu, and Kenzo Nonami. Guidance and nonlinear control system for autonomous flight of minirotorcraft unmanned aerial vehicles. *Journal of Field Robotics*, 27(3):311–334, 2010.

[31] Kostas Alexis, George Nikolakopoulos, and Anthony Tzes. Switching model predictive attitude control for a quadrotor helicopter subject to atmospheric disturbances. *Control Engineering Practice*, 19(10):1195–1207, 2011.

[32] M.O. Efe. Neural network assisted computationally simple pid control of a quadrotor uav. *Industrial Informatics, IEEE Transactions on*, 7(2):354–361, 2011. ISSN 1551-3203. doi: 10.1109/TII.2011.2123906.

[33] Amr Nagaty, Sajad Saeedi, Carl Thibault, Mae Seto, and Howard Li. Control and navigation framework for quadrotor helicopters. *Journal of Intelligent and Robotic Systems*, 70(1-4):1–12, 2013. ISSN 0921-0296. doi: 10.1007/s10846-012-9789-z. URL http://dx.doi.org/10.1007/s10846-012-9789-z.

[34] Gabriel M Hoffmann, Haomiao Huang, Steven L Waslander, and Claire J Tomlin. Quadrotor helicopter flight dynamics and control: Theory and experiment. In *Proc. of the AIAA Guidance, Navigation, and Control Conference*, pages 1–20, 2007.

[35] Jun Jiang, Juntong Qi, Dalei Song, and Jianda Han. Control platform design and experiment of a quadrotor. In *Control Conference (CCC), 2013 32nd Chinese*, pages 2974–2979. IEEE, 2013.

[36] Ascending Technologies. Accessed 15 April 2014. URL http://www.asctec.de/uav-applications/research/products/.

[37] MIT Aerospace Controls Lab. Accessed 15 April 2014. URL http://acl.mit.edu/.

[38] MIT Robust Robotics Group. Accessed 15 April 2014. URL http://groups.csail.mit.edu/rrg/.

[39] University of Pennsylvania GRASP Lab. Accessed 15 April 2014. URL https://www.grasp.upenn.edu/.

[40] Draganfly Innovations. Accessed 15 April 2014. URL http://www.draganfly.com/.

[41] Microdrones GmbH. Accessed 15 April 2014. URL http://www.microdrones.com.

[42] Parrot AR Drone. Accessed 15 April 2014. URL http://ardrone2.parrot.com/.

[43] L Derafa, T Madani, and A Benallegue. Dynamic modelling and experimental identification of four rotors helicopter parameters. In *Industrial Technology, 2006. ICIT 2006. IEEE International Conference on*, pages 1834–1839. IEEE, 2006.

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

[45] Kenneth H McNichols and M Sami Fadali. Selecting operating points for discrete-time gain scheduling. *Computers & Electrical Engineering*, 29(2):289–301, 2003.

[46] Jinkun Liu and Xinhua Wang. *Advanced Sliding Mode Control for Mechanical Systems: Design, Analysis and MATLAB Simulation*. Springer, 2012.

[47] Chyun-Chau Fuh et al. Variable-thickness boundary layers for sliding mode control. *Journal of Marine Science and Technology*, 16(4):286–292, 2008.

[48] Miroslav Krstic, Petar V Kokotovic, and Ioannis Kanellakopoulos. *Nonlinear and adaptive control design*. John Wiley & Sons, Inc., 1995.

[49] Horacio J Marquez. *Nonlinear control systems: analysis and design*. John Wiley, 2003.

[50] Randal W Beard. Quadrotor dynamics and control. *Brigham Young University*, 2008.