Chapter 2
Dynamic Modeling and Control Techniques for a Quadrotor

Heba Elkholy
American University in Cairo, Egypt

Maki K. Habib
American University in Cairo, Egypt

ABSTRACT

This chapter presents the detailed dynamic model of a Vertical Take-Off and Landing (VTOL) type Unmanned Aerial Vehicle (UAV) known as the quadrotor. The mathematical model is derived based on Newton Euler formalism. This is followed by the development of a simulation environment on which the developed model is verified. Four control algorithms are developed to control the quadrotor's degrees of freedom: a linear PID controller, Gain Scheduling-based PID controller, nonlinear Sliding Mode, and Backstepping controllers. The performances of these controllers are compared through the developed simulation environment in terms of their dynamic performance, stability, and the effect of possible disturbances.

INTRODUCTION

In the past decade, a lot of researchers are now focusing on developing miniature flying objects due to the recent advances in technologies and the emergence of miniature sensors and actuators depending on MEMS and NEMS. The developed miniature flying objects can be used in a broad set of applications ranging from military to civil ones. The reason for choosing quadrotors over other UAVs to be the focus of our work is their advantages over their counterparts due to the presence of four separately powered propellers thus giving the quadrotors a higher payload and better maneuverability. Also, their VTOL and hovering capabilities make them good candidates for surveillance and monitoring tasks and for the use in small spaces. The quadrotors control research field is still facing a lot of challenges; this is due to the fact that the quadrotor is a highly nonlinear, multivariable and underactuated system [Hou et al. (2010)]. 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

DOI: 10.4018/978-1-5225-8365-3.ch002

the actuators and the degrees of freedom [Kim et al. (2010)]. 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 huge performance degradation whenever the quadrotor leaves the nominal conditions or performs aggressive maneuvers [Kendoul (2012)].

STATE OF THE ART

Controlling the degrees of freedom of the quadrotor can be done through various control algorithms which vary from the classical linear Proportional-Integral-Derivative (PID) or Proportional-Derivative (PD) controller to more complex nonlinear schemes such as backstepping or sliding mode controllers. Starting with the linear control algorithms; Bouabdallah et al. applied a PID and LQ controllers on an indoor micro quadrotor. The performance of the two controllers was comparable in stabilizing the attitude of the quadrotor around its hover position and under the effect of little disturbances [Bouabdallah et al. (2004)]. 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 [Li & Li (2011)]. Simulation based results showed that Yang et al. were able to control the attitude and heading of a quadrotor using a self-tuning PID controller based on adaptive pole placement [Yang et al. (2013)]. Raffo et al. used an H_∞ controller to stabilize the rotational angles and a Model Predictive Controller (MPC) to track the desired position of a quadrotor. The effect of wind and model uncertainties was added to the simulated model and it performed robustly with a zero steady-state error [Raffo et al. (2010)].

In order to employ a linear controller to control a nonlinear system like that of the quadrotor, the system's nonlinearities can be modeled as a collection of simplified linear systems and for each system a separate controller can be designed, this is the concept of gain scheduling and it is commonly used in 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 [Gillula et al. (2011)]. 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 [Ataka et al. (2013)]. Amoozgar et al. compared the performance of a conventional PID controller to that of a gain scheduled PID controller with its parameters tuned using a fuzzy logic based inference scheme. The gain scheduled PID controller outperformed the conventional PID controller when the system was tested under actuator fault conditions [Amoozgar et al. (2012)]. In load dropping applications, Sadeghzadeh et al. found that a gain scheduled PID controller was able to stabilize the system during the dropping operation [Sadeghzadeh et al. (2012)].

Moving to the nonlinear flight control algorithms, Bouabdallah and Siegwart compared the performance of a backstepping and sliding mode control algorithms to the performance of that of linear PID an LQ controllers in their prior work. They found that nonlinear controllers gave better performance in the presence of disturbances [Bouabdallah & Siegwart (2005)]. Waslander et al. compared the performance of an integral sliding mode controller to that of a reinforcement learning controller to stabilize a quadrotor in an outdoors environment. It was found that, both of the implemented control techniques were able to stabilize the quadrotor and gave a better performance over the classical control algorithms [Waslander et al. (2005)]. 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 [Madani & Benallegue (2006)]. Fang and Gao merged a backstepping controller with an adaptive controller thus resulting in an integral backstepping algorithm to overcome the problems of model uncertainties and external disturbances. The controller reduced the system's overshoot and response time and eliminated the steady state error [Fang & Gao (2012)]. 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 [Lee et al. (2013)]. 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 [Zhen et al. (2013)]. Gonzalez 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 [González et al. (2014)]. Kendoul et al. was able to control a quadrotor in several flight tests based on the concept of feedback linearization [Kendoul et al. (2010)]. Alexis et al. relied on a MPC to control the attitude of a quadrotor in the presence of atmospheric disturbances. MPC relies on predicting the future states of the system and tracking the error to give an improved performance [Kendoul (2012)]. The proposed algorithm behaved well in performing rough maneuvers in a wind induced environment as was able to accurately track the desired attitude [Alexis et al. (2011)]. 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 [Sadeghzadeh et al. (2012)].

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. Efe used a Neural Network to simplify the design of a PID controller and decrease the computational time and complexity [Efe (2011)]. Another approach is using fuzzy logic based control systems, where the experience of a skilled pilot is employed to train the controller [Kendoul (2012)].

In more recent literature, it was found that using more than one type of control algorithms together generated a better performance, especially when the quadrotor is not flying near hover. Nagaty et al. proposed the usage of a nested loop control algorithm; the outer loop consists of a PID controller responsible for generating 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 [Nagaty et al. (2013)]. 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 [Azzam & Wang (2010)].

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 and having it perform acrobatic maneuvers. Also, there is no work focused on having

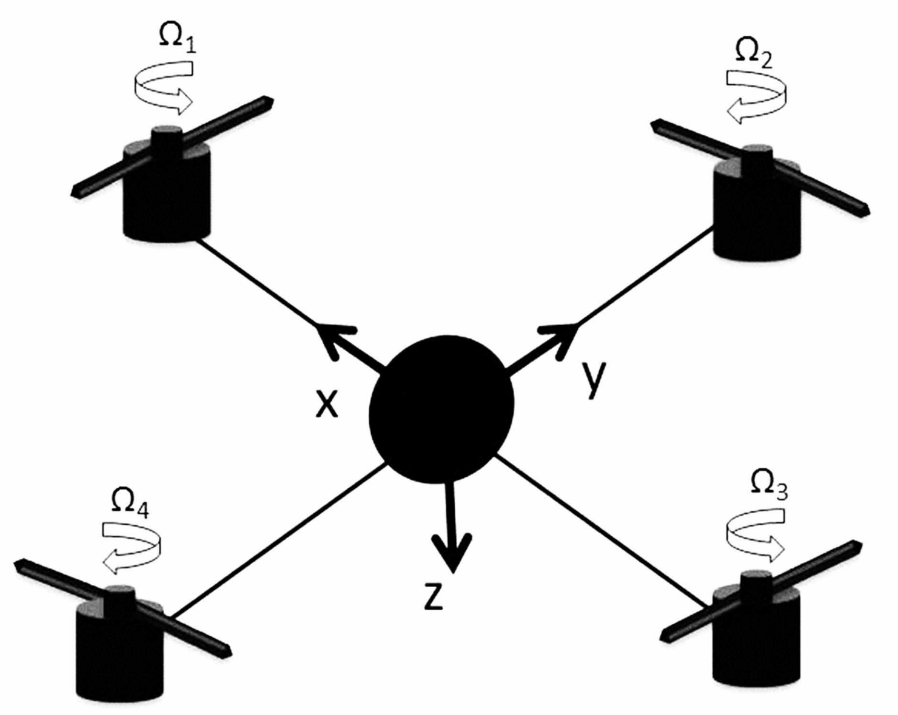
a comparative study that compares between employing different types of controllers on the quadrotor system and comparing their performances in stabilizing the quadrotor under different flight conditions and this will be the focus of this work.

THE QUADROTOR CONCEPT AND STRUCTURE

The quadrotor is a rotary wing UAV consisting of four rotors, each fitted on one end of a cross-like structure as shown in Figure 1. 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 [Bouabdallah et al. (2004), Hou et al. (2010)].

A quadrotor is a 6 DOF object, thus to express its position in space, 6 variables are used ($x, y, z, \varphi, \theta, \psi$). 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 2 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 position, 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. Quadrotor configuration



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 need to be changed.

One problem with the quadrotor configuration is that to produce yaw rotation, one needs 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 needs to be increased while the speed of the two counterclockwise turning rotors needs to be decreased [Bouabdallah et al. (2004), Mistler et al. (2001)].

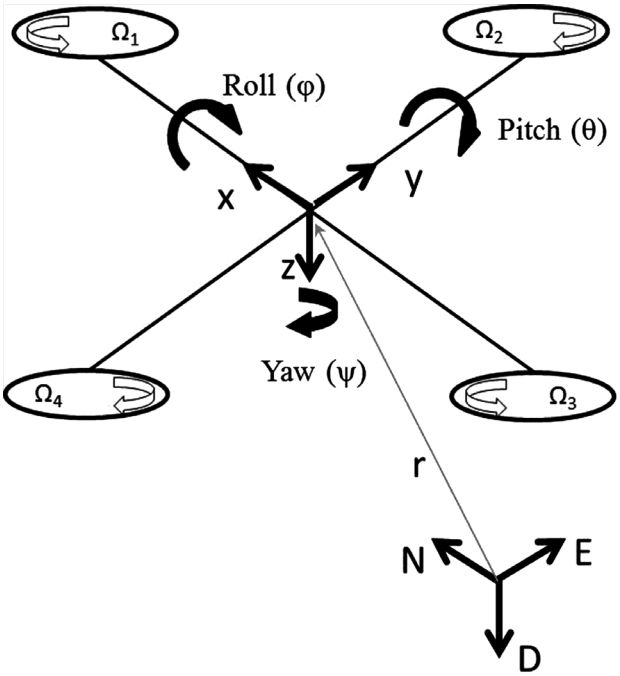
Advantages and Drawbacks of Quadrotors

Some advantages of the quadrotor over helicopters is that their 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 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 [Hou et al. (2010)].

More advantages are the vertical take-off and landing capabilities, better maneuverability and smaller size due to the absence of a tail [Li & Li (2011)], these capabilities make quadrotors useful in small area monitoring and buildings exploration [Bouabdallah & Siegwart (2005)].

Figure 2. Euler angles for a quadrotor



Moreover, quadrotors have higher payload capacities due to the presence of four motors thus providing higher thrust [Hou et al. (2010)].

On the other hand, quadrotors consume a lot of energy due to the presence of four separate propellers [Bouabdallah & Siegwart (2005)]. Also, they have a large size and heavier than some of their counterparts again to the fact that there is four separate propellers [Bouabdallah (2007), Bouabdallah & Siegwart (2005)].

SYSTEM MODELING

The kinematics and dynamics models of a quadrotor will be derived based on the 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.

Kinematic Model

Two sets of coordinate frames are defined in order to discuss the modeling of the quadrotor. Figure 2 shows the Earth reference frame, which is fixed on a specific place at ground level with its N, E and D axes positing towards North, East and Downwards respectively. The second coordinate frame is the body frame which is fixed at the center of the quadrotor body with its x, y and z axes pointing towards propeller 1, propeller 2 and to the ground respectively.

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:

$$R = \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\theta c\psi & c\phi s\theta s\psi - s\theta c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix} \tag{1.1}$$

where c and s denote cos and sin respectively.

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:

$$\omega = R_r \dot{\eta} \tag{1.2}$$

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 the identity matrix I [Nagaty et al. (2013)].

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 [Nagaty et al. (2013)].

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,

$$J\dot{\omega} + \omega \times J\omega + M_G + M_a = M_B \tag{1.3}$$

where J represents the quadrotor's diagonal inertia matrix, ω represents the angular body rates. M_G represent the gyroscopic moments due to the rotors' inertia. M_a is the aerodynamic moments acting on the quadrotor body while M_B are the moments acting on the quadrotor in the body frame.

The first two terms of Equation 1.3, $J\dot{\omega}$ and $\omega \times J\omega$, represent the rate of change of angular momentum in the body frame. 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.

$$J = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{1.4}$$

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 [Derafa et al. (2006)]. It is defined to be,

$$M_G = \omega \times [0\ 0\ J_r \Omega_r]^T \tag{1.5}$$

where J_r is the rotors' inertia and Ωr is the rotors' relative speed which is defined to be, $\Omega_r = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4$

Aerodynamic Moments

Due to the friction of the moving quadrotor body with air, a moment acts on the body of the quadrotor resisting its motion. As the angular velocity of the quadrotor increases, the drag moments in turn increase. The drag moments M_a can be approximated by,

$$M_a = K_r \dot{\eta} \tag{1.6}$$

where K_r is a constant matrix called the aerodynamic rotation coefficient matrix and $\dot{\eta}$ is the Euler rates.

Moments Acting on the Quadrotor (M_B)

For the last term of Equation 1.3, 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 1.6 and 1.7 show the aerodynamic force F_i and moment M_i produced by the i^{th} rotor [Azzam & Wang (2010)].

$$F_i = \frac{1}{2} \rho A C_T r^2 \Omega_i^2 \tag{1.7}$$

$$M_i = \frac{1}{2} \rho A C_D r^2 \Omega_i^2 \tag{1.8}$$

where ρ is the air density, A is the blade area, C_T and C_D are aerodynamic coefficients, r is the radius of the blade and Ω_i is the 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 1.6 and 1.7 can be simplified to [Nagaty et al. (2013)],

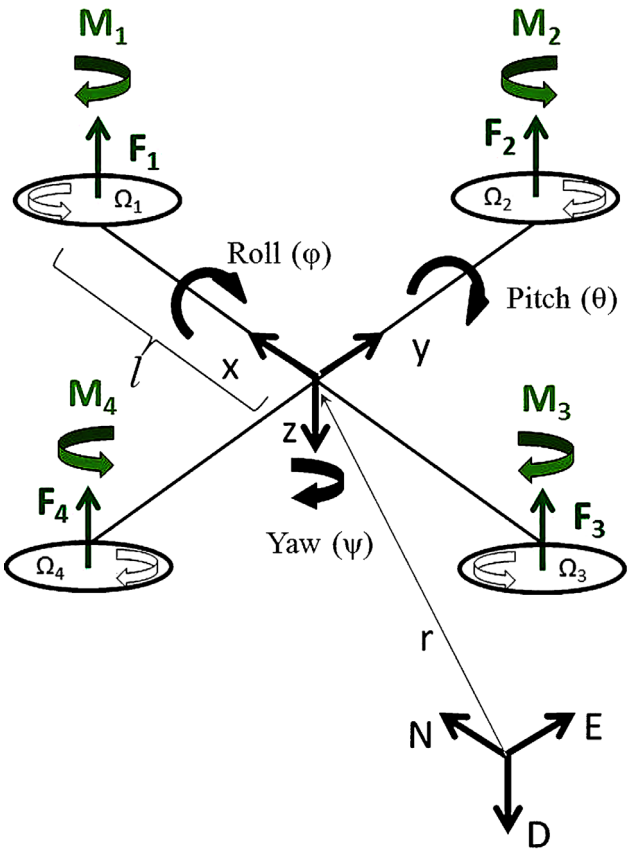
$$F_i = K_f \Omega_i^2 \tag{1.9}$$

$$M_i = K_M \Omega_i^2 \tag{1.10}$$

where K_f and K_M are the aerodynamic force and moment constants respectively. 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 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*.

Figure 3. Forces and moments acting on quadrotor



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

$$\begin{aligned} M_x &= -F_2 l + F_4 l \\ &= -(K_f \Omega_2^2) l + (K_f \Omega_4^2) l \\ &= l K_f (-\Omega_2^2 + \Omega_4^2) \end{aligned} \tag{1.11}$$

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,

$$\begin{aligned} M_y &= F_1 l - F_3 l \\ &= -(K_f \Omega_1^2) l - (K_f \Omega_3^2) l \\ &= l K_f (\Omega_1^2 - \Omega_3^2) \end{aligned} \tag{1.12}$$

For the moments about the body frame's z-axis, the thrust of the rotors does not cause a moment. On the other hand, moments are caused by the rotors' rotation as per Equation 1.9. By using the right-hand-rule, the moment about the body frame's z-axis can be expressed as,

$$\begin{aligned} M_z &= M_1 - M_2 + M_3 - M_4 \\ &= (K_M \Omega_1^2) - (K_M \Omega_2^2) + (K_M \Omega_3^2) - (K_M \Omega_4^2) \\ &= (K_M \Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2) \end{aligned} \tag{1.13}$$

Combining Equations 1.10, 1.11 and 1.12 in vector form, we get,

$$M_B = \begin{bmatrix} l K_f (-\Omega_2^2 + \Omega_4^2) \\ l K_f (\Omega_1^2 - \Omega_2^2) \\ K_M (\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2) \end{bmatrix} \tag{1.14}$$

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.

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 [Nagaty et al. (2013)],

$$m\ddot{r} + F_a = \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} + RF_B \tag{1.15}$$

where $r = [x\ y\ z]^T$ is the quadrotor's distance from the inertial frame, m is the quadrotor's mass. F_a is the aerodynamic forces. g is the gravitational acceleration and F_B is the nongravitational forces acting on the quadrotor in the body frame.

Aerodynamic Forces

Similar to the aerodynamic moments, there are aerodynamic forces acting on the quadrotor that oppose its motion. The drag forces F_a can be approximated by,

$$F_a = K_t \dot{r} \tag{1.16}$$

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.

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 force 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 1.9. Thus, the nongravitational forces acting on the quadrotor, F_B, can be expressed as,

$$F_B = \begin{bmatrix} 0 \\ 0 \\ -K_f(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \end{bmatrix} \tag{1.17}$$

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.

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. Note that the aerodynamic forces and moments (F_a and M_a) were neglected due to their minor effects in simulation.

State Vector X

Defining the state vector of the quadrotor to be,

$$X = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 & x_9 & x_{10} & x_{11} & x_{12} \end{bmatrix}^T \tag{1.18}$$

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

$$X = \begin{bmatrix} \phi & \dot{\phi} & \theta & \dot{\theta} & \psi & \dot{\psi} & z & \dot{z} & x & \dot{x} & y & \dot{y} \end{bmatrix}^T \tag{1.19}$$

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

Control Input Vector *U*

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

$$U = \begin{bmatrix} U_1 & U_2 & U_3 & U_4 \end{bmatrix} \tag{1.20}$$

where

$$\begin{aligned} U_1 &= K_f(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\ U_2 &= K_f(-\Omega_2^2 + \Omega_4^2) \\ U_3 &= K_f(\Omega_1^2 - \Omega_3^2) \\ U_4 &= K_M(\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2) \end{aligned} \tag{1.21}$$

Equations 1.21 can be arranged in a matrix form to result in,

$$\begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} K_f & K_f & K_f & K_f \\ 0 & -K_f & 0 & K_f \\ K_f & 0 & -K_f & 0 \\ K_M & -K_M & K_M & -K_M \end{bmatrix} \begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} \tag{1.22}$$

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 turn-

ing 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 1.22 to give,

$$\begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} = \begin{bmatrix} \frac{1}{4K_f} & 0 & \frac{1}{2K_f} & \frac{1}{4K_M} \\ \frac{1}{4K_f} & -\frac{1}{2K_f} & 0 & -\frac{1}{4K_M} \\ \frac{1}{4K_f} & 0 & -\frac{1}{2K_f} & \frac{1}{4K_M} \\ \frac{1}{4K_f} & \frac{1}{2K_f} & 0 & -\frac{1}{4K_M} \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} \tag{1.23}$$

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

$$\begin{aligned} \Omega_1 &= \sqrt{\frac{1}{4K_f}U_1 + \frac{1}{2K_f}U_3 + \frac{1}{4K_M}U_4} \\ \Omega_2 &= \sqrt{\frac{1}{4K_f}U_1 - \frac{1}{2K_f}U_2 - \frac{1}{4K_M}U_4} \\ \Omega_3 &= \sqrt{\frac{1}{4K_f}U_1 - \frac{1}{2K_f}U_3 + \frac{1}{4K_M}U_4} \\ \Omega_4 &= \sqrt{\frac{1}{4K_f}U_1 + \frac{1}{2K_f}U_2 - \frac{1}{4K_M}U_4} \end{aligned} \tag{1.24}$$

Rotational Equation of Motion

Substituting equations 1.21 in Equation 1.14, the equation of the total moments acting on the quadrotor becomes,

$$M_B = \begin{bmatrix} l\,U_2 \\ l\,U_3 \\ U4 \end{bmatrix} \tag{1.25}$$

Substituting Equation 1.25 into the rotational equation of motion and expanding each term with their prior definition, the following relation can be derived,

$$\begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} + \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \times \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ J_r \Omega_r \end{bmatrix} = \begin{bmatrix} l\,U_2 \\ l\,U_3 \\ U_4 \end{bmatrix} \tag{1.26}$$

Expanding that, leads to,

$$\begin{bmatrix} I_{xx}\ddot{\phi} \\ I_{yy}\ddot{\theta} \\ I_{zz}\ddot{\psi} \end{bmatrix} + \begin{bmatrix} \dot{\theta} I_{zz}\dot{\psi} - \dot{\psi} I_{yy}\dot{\theta} \\ \dot{\psi} I_{xx}\dot{\phi} - \dot{\phi} I_{zz}\dot{\psi} \\ \dot{\phi} I_{yy}\dot{\theta} - \dot{\theta} I_{xx}\dot{\phi} \end{bmatrix} + \begin{bmatrix} \dot{\theta} J_r \Omega_r \\ -\dot{\phi} J_r \Omega_r \\ 0 \end{bmatrix} = \begin{bmatrix} l\,U_2 \\ l\,U_3 \\ U_4 \end{bmatrix} \tag{1.27}$$

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

$$\begin{aligned} \ddot{\phi} &= \frac{l}{I_{xx}} U_2 - \frac{J_r}{I_{xx}} \dot{\theta} \Omega_r + \frac{I_{yy}}{I_{xx}} \dot{\psi}\dot{\theta} - \frac{I_{zz}}{I_{xx}} \dot{\theta}\dot{\psi} \\ \ddot{\theta} &= \frac{l}{I_{yy}} U_3 - \frac{J_r}{I_{yy}} \dot{\phi} \Omega_r + \frac{I_{zz}}{I_{yy}} \dot{\phi}\dot{\psi} - \frac{I_{xx}}{I_{yy}} \dot{\psi}\dot{\phi} \\ \ddot{\psi} &= \frac{1}{I_{zz}} U_4 + \frac{I_{xx}}{I_{zz}} \dot{\theta}\dot{\phi} - \frac{I_{yy}}{I_{zz}} \dot{\phi}\dot{\theta} \end{aligned} \tag{1.28}$$

To simplify, define,

$$a_1 = \frac{I_{yy} - I_{zz}}{I_{xx}}$$

$$a_2 = \frac{J_r}{I_{xx}} \qquad b_1 = \frac{l}{I_{xx}}$$

$$a_3 = \frac{I_{zz} - I_{xx}}{I_{yy}} \qquad b_2 = \frac{l}{I_{yy}}$$

$$a_4 = \frac{J_r}{I_{yy}} \qquad b_3 = \frac{l}{I_{zz}}$$

$$a_5 = \frac{I_{xx} - I_{yy}}{I_{zz}}$$

Using the above definition of $a_1 \rightarrow a_5$ and $b_1 \rightarrow b_3$, equations 1.28 can then be rewritten in a simpler form in terms of the system states,

$$\begin{aligned}\ddot{\phi} &= b_1 U_2 - a_2 x_4 \Omega_r + a_1 x_4 x_6 \\ \ddot{\theta} &= b_2 U_3 + a_4 x_2 \Omega_r + a_3 x_2 x_6 \\ \ddot{\psi} &= b_3 U_4 + a_5 x_2 x_4\end{aligned} \tag{1.29}$$

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

Translational Equation of Motion

Substituting Equation 1.21 in Equation 1.17, the equation of the total moments acting on the quadrotor becomes,

$$F_B = \begin{bmatrix} 0 \\ 0 \\ -U_1 \end{bmatrix} \tag{1.30}$$

Embedding that into the translational equation of motion and expanding the terms, we get,

$$\begin{aligned} m\begin{bmatrix}\ddot{x}\\ \ddot{y}\\ \ddot{z}\end{bmatrix} &= \begin{bmatrix}0\\0\\mg\end{bmatrix} + \begin{bmatrix} c\psi c\theta & c\psi s\phi s\theta & s\phi s\psi + c\phi c\psi s\theta \\ c\theta s\psi & c\theta c\psi + s\phi s\psi s\theta & c\phi s\psi s\theta - c\psi s\theta \\ -s\theta & c\theta s\phi & c\phi c\theta \end{bmatrix}\begin{bmatrix}0\\0\\-U_1\end{bmatrix} \\ m\begin{bmatrix}\ddot{x}\\ \ddot{y}\\ \ddot{z}\end{bmatrix} &= \begin{bmatrix}0\\0\\mg\end{bmatrix} + \begin{bmatrix}(s\phi s\psi + c\phi c\psi s\theta)(-U_1)\\ (c\phi s\psi s\theta - c\psi s\phi)(-U_1)\\ (c\phi c\theta)(-U_1)\end{bmatrix}\end{aligned} \tag{1.31}$$

Rewriting Equation 1.31 to have the accelerations in terms of the other variables, we get,

$$\begin{aligned}\ddot{x} &= \frac{-U_1}{m}(\sin\phi\sin\psi + \cos\phi\cos\psi\sin\theta) \\ \ddot{y} &= \frac{-U_1}{m}(\cos\phi\sin\psi\sin\theta - \cos\psi\sin\phi) \\ \ddot{z} &= g - \frac{U_1}{m}(\cos\phi\cos\theta)\end{aligned} \tag{1.32}$$

Rewriting in terms of the state variable *X*,

$$\begin{aligned}
\ddot{x} &= \frac{-U_1}{m}(\sin x_1 \sin x_5 + \cos x_1 \cos x_5 \sin x_3) \\
\ddot{y} &= \frac{-U_1}{m}(\cos x_1 \sin x_5 \sin x_3 - \cos x_5 \sin x_1) \\
\ddot{z} &= g - \frac{U_1}{m}(\cos x_1 \cos x_3)
\end{aligned} \tag{1.33}$$

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

State Space Representation

Using the equations of the rotational angular acceleration and those of translation, the complete mathematical model of the quadrotor can be written in a state space representation as follows,

$$\begin{aligned}
\dot{x}_1 &= \dot{\phi} = x_2 \\
\dot{x}_2 &= \ddot{\phi} = x_4 x_6 a_1 - x_4 \Omega_r a_2 + b_1 U_2 \\
\dot{x}_3 &= \dot{\theta} = x_4 \\
\dot{x}_4 &= \ddot{\theta} = x_2 x_6 a_3 + x_2 \Omega_r a_4 + b_2 U_3 \\
\dot{x}_5 &= \dot{\psi} = x_6 \\
\dot{x}_6 &= \ddot{\psi} = x_2 x_4 a_5 + b_3 U_4 \\
\dot{x}_7 &= \dot{z} = x_8 \\
\dot{x}_8 &= \ddot{z} = g - \frac{U_1}{m}(\cos x_1 \cos x_3) \\
\dot{x}_9 &= \dot{x} = x_{10} \\
\dot{x}_{10} &= \ddot{x} = \frac{-U_1}{m}(\sin x_1 \sin x_5 + \cos x_1 \sin x_3 \cos x_5) \\
\dot{x}_{11} &= \dot{y} = x_{12} \\
\dot{x}_{12} &= \ddot{y} = \frac{U_1}{m}(\sin x_1 \cos x_5 - \cos x_1 \sin x_3 \sin x_5)
\end{aligned} \tag{1.34}$$

or

$$f(X,U) = \begin{bmatrix} x_2 \\ x_4 x_6 a_1 - x_4 \Omega_r a_2 + b_1 U_2 \\ x_4 \\ x_2 x_6 a_3 + x_2 \Omega_r a_4 + b_2 U_3 \\ x_6 \\ x_2 x_4 a_5 + b_3 U_4 \\ x_8 \\ g - \frac{U_1}{m}(\cos x_1 \cos x_3) \\ x_{10} \\ \frac{-U_1}{m}(\sin x_1 \sin x_5 + \cos x_1 \sin x_3 \cos x_5) \\ x_{12} \\ \frac{U_1}{m}(\sin x_1 \cos x_5 - \cos x_1 \sin x_3 \sin x_5) \end{bmatrix} \tag{1.35}$$

SYSTEM CONTROL

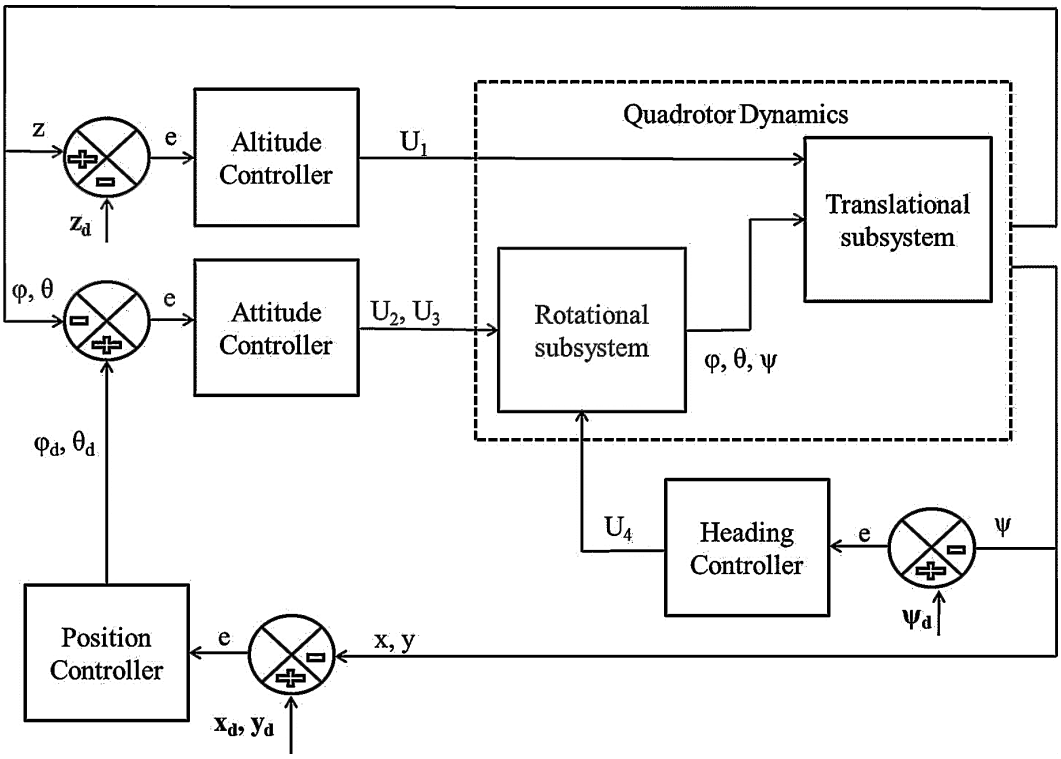
The mathematical model of the quadrotor implies a nested structure controller, where the outer controller will control the position of the quadrotor in space by generating the required attitude that in turn will be fed to the inner controller that will directly control the attitude by generating the required control inputs to be fed to the quadrotor dynamics as shown in Figure 4.

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 from the translational equations of motion as follows,

$$\begin{aligned} \ddot{x} &= \frac{-U_1}{m}(\sin\phi_d \sin\psi + \cos\phi_d \sin\theta_d \cos\psi) \\ \ddot{y} &= \frac{-U_1}{m}(\cos\phi_d \sin\theta_d \sin\psi - \sin\phi_d \cos\psi) \end{aligned} \tag{1.36}$$

Since the quadrotor is operating around hover, which means small values for the roll and pitch angles, 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,

Figure 4. Closed loop control system of a quadrotor



$$\begin{aligned} \ddot{x} &= \frac{-U_1}{m}(\phi_d \sin\psi + \theta_d \cos\psi) \\ \ddot{y} &= \frac{-U_1}{m}(\theta_d \sin\psi - \phi_d \cos\psi) \end{aligned} \tag{1.37}$$

which can be written in a matrix form as,

$$\begin{bmatrix} -\sin\psi & -\cos\psi \\ \cos\psi & -sin\psi \end{bmatrix} \begin{bmatrix} \phi_d \\ \theta_d \end{bmatrix} = \frac{m}{U_1} \begin{bmatrix} \ddot{x}_d \\ \ddot{y}_d \end{bmatrix} \tag{1.38}$$

which can be inverted to get

$$\begin{aligned} \begin{bmatrix} \phi_d \\ \theta_d \end{bmatrix} &= \begin{bmatrix} -\sin\psi & -\cos\psi \\ \cos\psi & -sin\psi \end{bmatrix}^{-1} \frac{m}{U_1} \begin{bmatrix} \ddot{x}_d \\ \ddot{y}_d \end{bmatrix} \\ &= \frac{m}{U_1} \begin{bmatrix} -\sin\psi & \cos\psi \\ -\cos\psi & -sin\psi \end{bmatrix} \begin{bmatrix} \ddot{x}_d \\ \ddot{y}_d \end{bmatrix} \\ &= \frac{m}{U_1} \begin{bmatrix} -\ddot{x}_d \sin\psi + \ddot{y}_d \cos\psi \\ -\ddot{x}_d \cos\psi - \ddot{y}_d sin\psi \end{bmatrix} \end{aligned} \tag{1.39}$$

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 controller blocks in the block diagram in Figure 4 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.

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 that 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 5.

PID Controller

After the mathematical model of the quadrotor along with its open loop simulation is 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 6.

Figure 5. Controller tuning using genetic algorithm

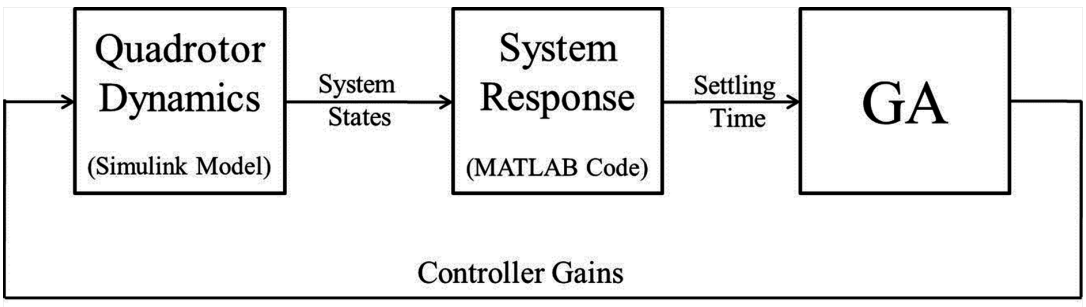
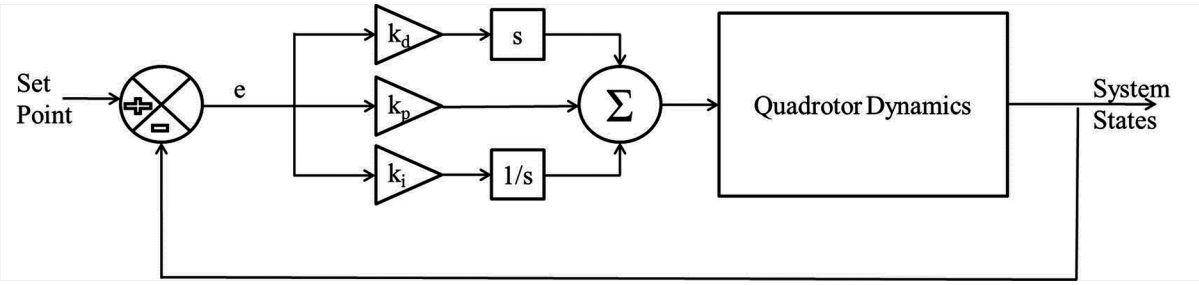


Figure 6. PID controller block diagram



A PID controller is used to track user defined trajectories for the altitude, attitude and heading of the quadrotor. The PID controllers generated the control inputs U_1 through U_4 based on the following control laws,

$$\begin{aligned} U_1 &= k_p(z - z_d) + k_d(\dot{z} - \dot{z_d}) + k_i\int(z - z_d)dt \\ U_2 &= k_p(\phi_d - \phi) + k_d(\dot{\phi}_d - \dot{\phi}) + k_i\int(\phi_d - \phi)dt \\ U_3 &= k_p(\theta_d - \theta) + k_d(\dot{\theta}_d - \dot{\theta}) + k_i\int(\theta_d - \theta)dt \\ U_4 &= k_p(\psi_d - \psi) + k_d(\dot{\psi}_d - \dot{\psi}) + k_i\int(\psi_d - \psi)dt \end{aligned} \tag{1.40}$$

where k_p, k_d and k_i are the proportional, derivative and integral gains respectively and z_d, ϕ_d, θ_d and ψ_d are the desired altitude, roll, pitch and heading respectively and $\dot{z}_d$, $\dot{\phi}_d$, $\dot{\theta}_d$ and $\dot{\psi}_d$ are their desired rate of change.

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 $\ddot{y}_d$,

$$\begin{aligned} \ddot{x}_d &= k_p(x_d - x) + k_d(\dot{x}_d - \dot{x}) + k_i\int(x_d - x)dt \\ \ddot{y}_d &= k_p(y_d - y) + k_d(\dot{y}_d - \dot{y}) + k_i\int(y_d - y)dt \end{aligned} \tag{1.41}$$

where again, k_p, k_d and k_i are the proportional, derivative and integral gains respectively and x_d and y_d are the desired *x* and *y* position and $\dot{x}_d$ and $\dot{y}_d$ are their desired rate of change.

Plugging the values of the desired accelerations $\ddot{x}_d$ and $\ddot{y}_d$ into Equation 1.39, 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 Equation 1.40.

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 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 7. 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 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 8.

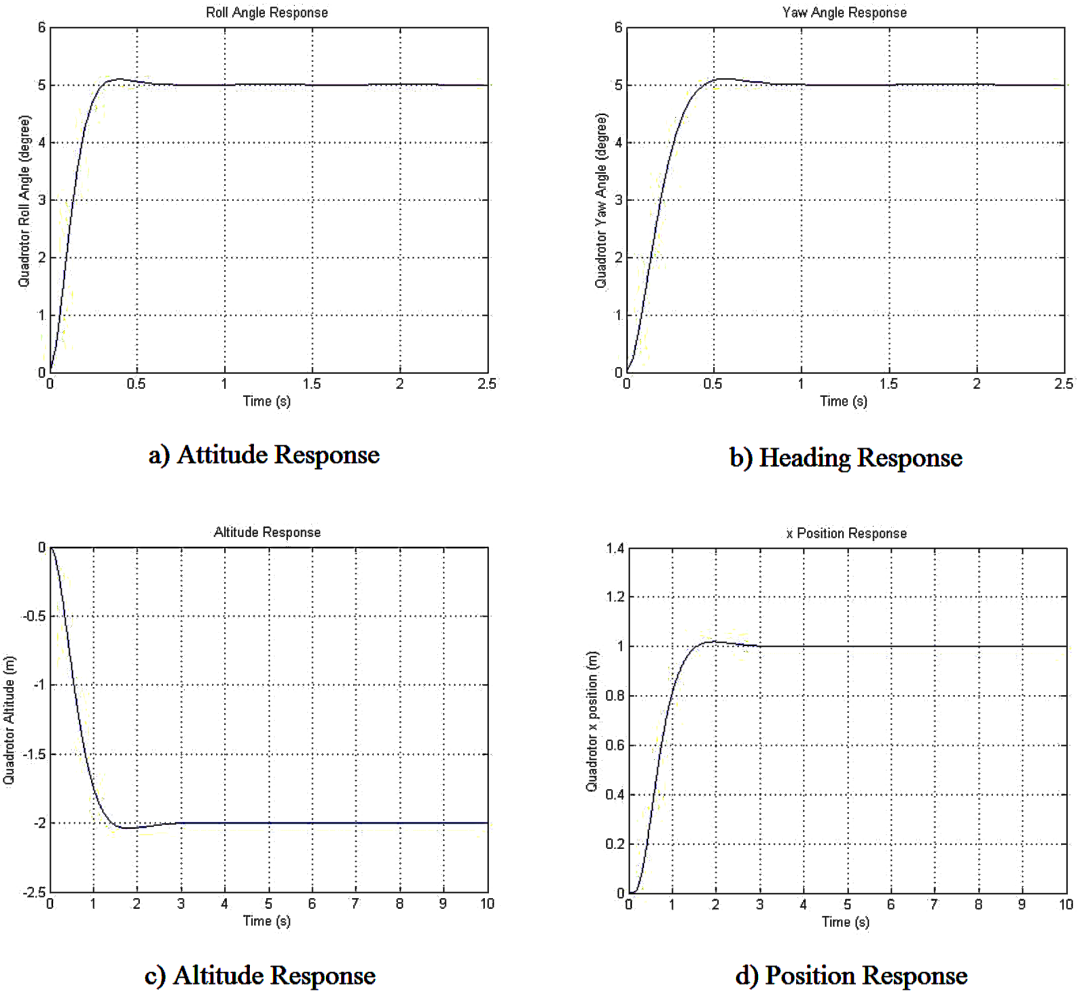
Gain Scheduling Based PD Controller

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 [McNichols & Fadali (2003)].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

Table 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%

Figure 7. PD controller simulation response



at different operating points. The acquired gains were used in a look up table fashion in the developed MATLAB/Simulink shown in Figure 9.

Gain Scheduling Based PD Controller Simulation Results

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 2 together with the resulting settling time for the system.

In order to show the strength of the developed gain 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 10 compares the performance of the classical PD to the gain scheduled PD.

Figure 8. Trajectory response under PD

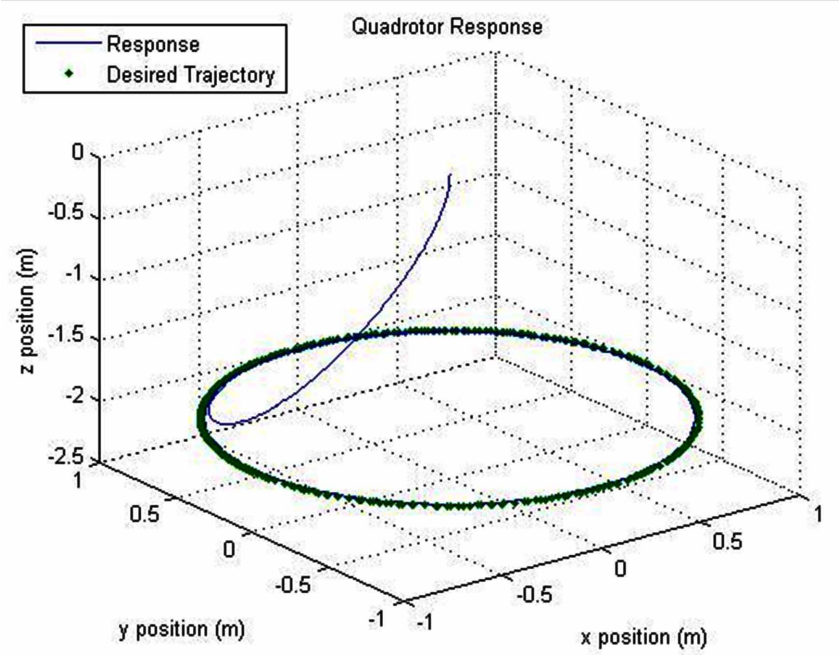
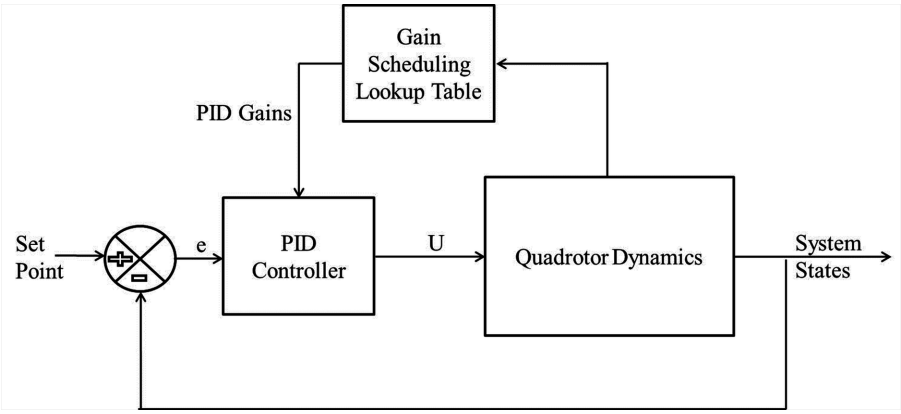


Figure 9. Gain scheduling block diagram



Attitude Controller

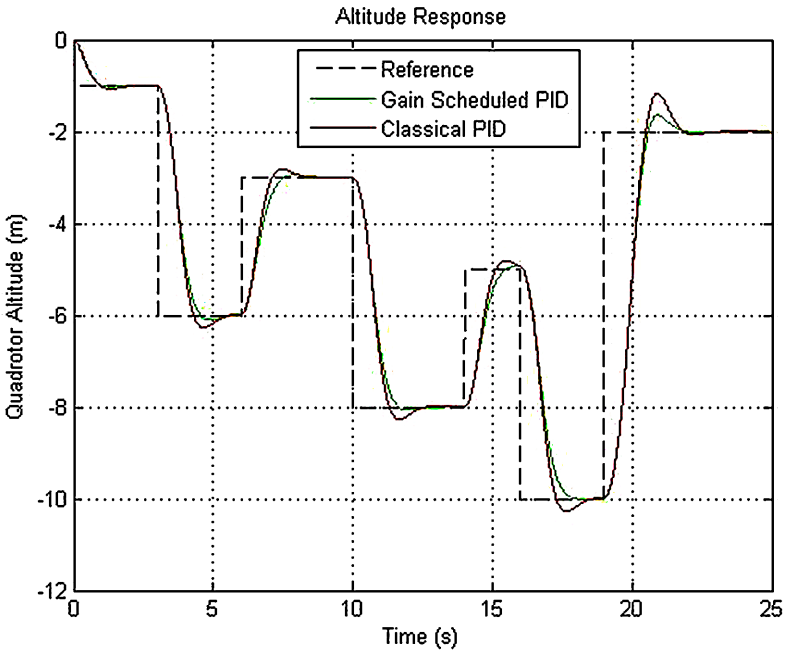
For the roll and pitch control, GA was also used to find the controller gains for a set of operating points. Table 3 shows the operating points together with their controller gains and performance.

Figure 11 shows a comparison between the performance of the gain scheduled PD controller and the classical PD controller in following a varying trajectory.

Table 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

Figure 10. Altitude response



Heading Controller

Similar to the attitude controller, a look up table was synthesized for the heading controller. The controller gains and their respective performances at multiple operating points are shown in Table 4 and the response is shown in Figure 12.

Table 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 11. Attitude response

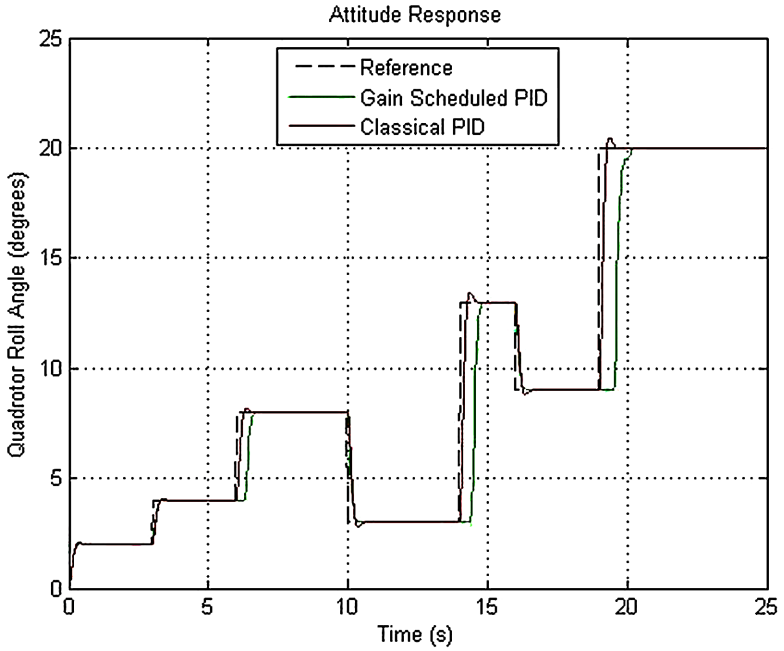


Table 4. Heading gain scheduling based PD controller gains and results

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

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.

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 state space and to maintain the state trajectories on this surface [Liu & Wang (2012)].The control law for a SMC consists of two parts as per Equation 1.42; a corrective control part and an equivalent control part. The corrective control function is to compensate any variations in 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.

$$U(t) = U_c(t) + U_{eq}(t) \tag{1.42}$$

where $U(t)$ is the control law, $U_c(t)$ is the corrective control and $U_{eq}(t)$ is the equivalent control.

A block diagram showing the SMC is shown in Figure 13.

The developed sliding mode controller is based on the approach used by [Bouabdallah & Siegwart (2005)]. The SMC is used to track a reference trajectory for the roll angle. The error in the roll is defined as,

Figure 12. Heading response

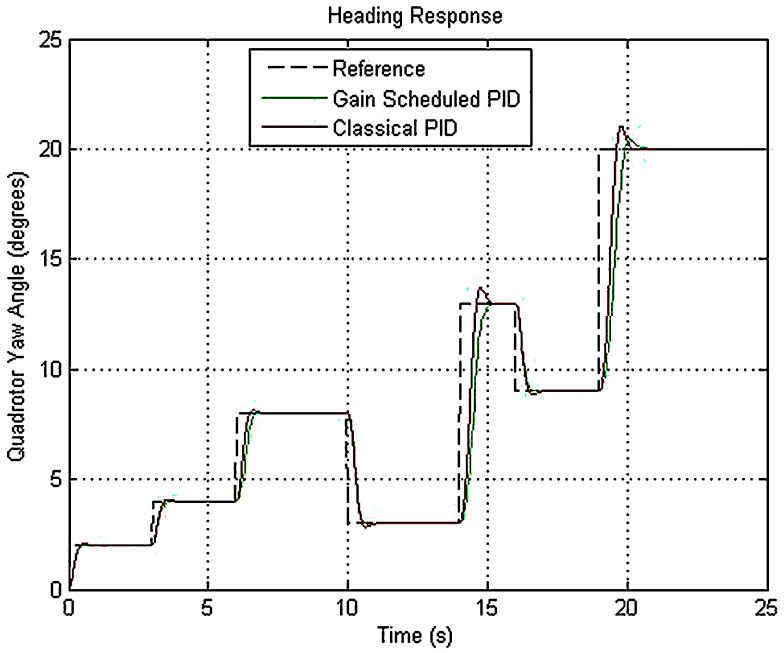
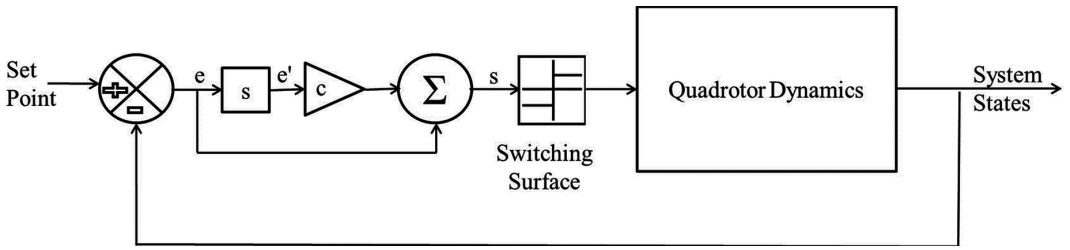


Figure 13. SMC block diagram



$$e = \phi_d - \phi \tag{1.43}$$

The sliding surface is defined as,

$$s = ce + \dot{e} \tag{1.44}$$

where c 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 1.44 with the substitution of Equation 1.43 is formulated as the following,

$$\begin{aligned} \dot{s} &= c_1\dot{e} + \ddot{e} \\ &= c_1(\dot{\phi}_d - \dot{\phi}) + \ddot{\phi}_d - \ddot{\phi} \end{aligned} \tag{1.45}$$

A Lyapunov function is then defined to be,

$$V(e,s) = \frac{1}{2}(e^2 + s^2) \tag{1.46}$$

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

$$\dot{s} = -k_1 sgn(s) - k_2 s \tag{1.47}$$

where

$$sgn(s) = \begin{cases} -1 & \text{if } s < 0; \\ 1 & \text{if } s > 0. \end{cases} \tag{1.48}$$

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 1.47 to the derivative of the sliding surface in Equation 1.45 and substituting $\ddot{\phi}$ by its definition from the rotational equations of motion, the control input U_2 is calculated to be,

$$U_2 = \frac{1}{b_1}[k_1 sgn(s) + k_2 s + c_1(\dot{\phi}_d - \dot{\phi}) + \ddot{\phi}_d + a_2\dot{\theta}\Omega_r - a_1\dot{\theta}\dot{\psi}] \tag{1.49}$$

Similarly, if the same steps are repeated for altitude, pitch and heading; the control inputs U_1, U_3 and U_4 are calculated to be,

$$\begin{aligned} U_1 &= \frac{m}{\cos\phi\cos\theta}[k_1 sgn(s) + k_2 s + c_1(\dot{z} - \dot{z}_d) + g - \ddot{z}_d] \\ U_3 &= \frac{1}{b_2}[k_1 sgn(s) + k_2 s + c_1(\dot{\theta}_d - \dot{\theta}) + \ddot{\theta}_d - a_4\dot{\phi}\Omega_r - a_3\dot{\phi}\dot{\psi}] \\ U_4 &= \frac{1}{b_3}[k_1 sgn(s) + k_2 s + c_1(\dot{\psi}_d - \dot{\psi}) + \ddot{\psi}_d - a_5\dot{\phi}\dot{\theta}] \end{aligned} \tag{1.50}$$

SMC Simulation Results

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 5 shows the GA generated design parameters and the resulting settling time and overshoot of the system. The response is

shown in Figure 14. The problem of chattering is clear in the response plots and the problem to overcome it will be addressed in the next section.

Due to the symmetry of the quadrotor, the results for the pitch controller were exactly the same as that of the roll and accordingly the GA produced the same control gains.

SMC Chattering Reduction

As shown in Figure 14, 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 15 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 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 [González et al. (2014)].

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.

$$chat = \int_{T_s}^{T_{end}} \left\| \phi - \phi_d \right\| dt \tag{1.51}$$

where T_s is the settling time for the system response and T_{end} is the simulation end time.

By visual inspection, Equation 1.51 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,

$$chat = \max_{T_s \to T_{end}} \dot{s} \tag{1.52}$$

Table 5. SMC results

	Desired Value	c	k_1	k_2	Setting 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%

Figure 14. SMC controller simulation response

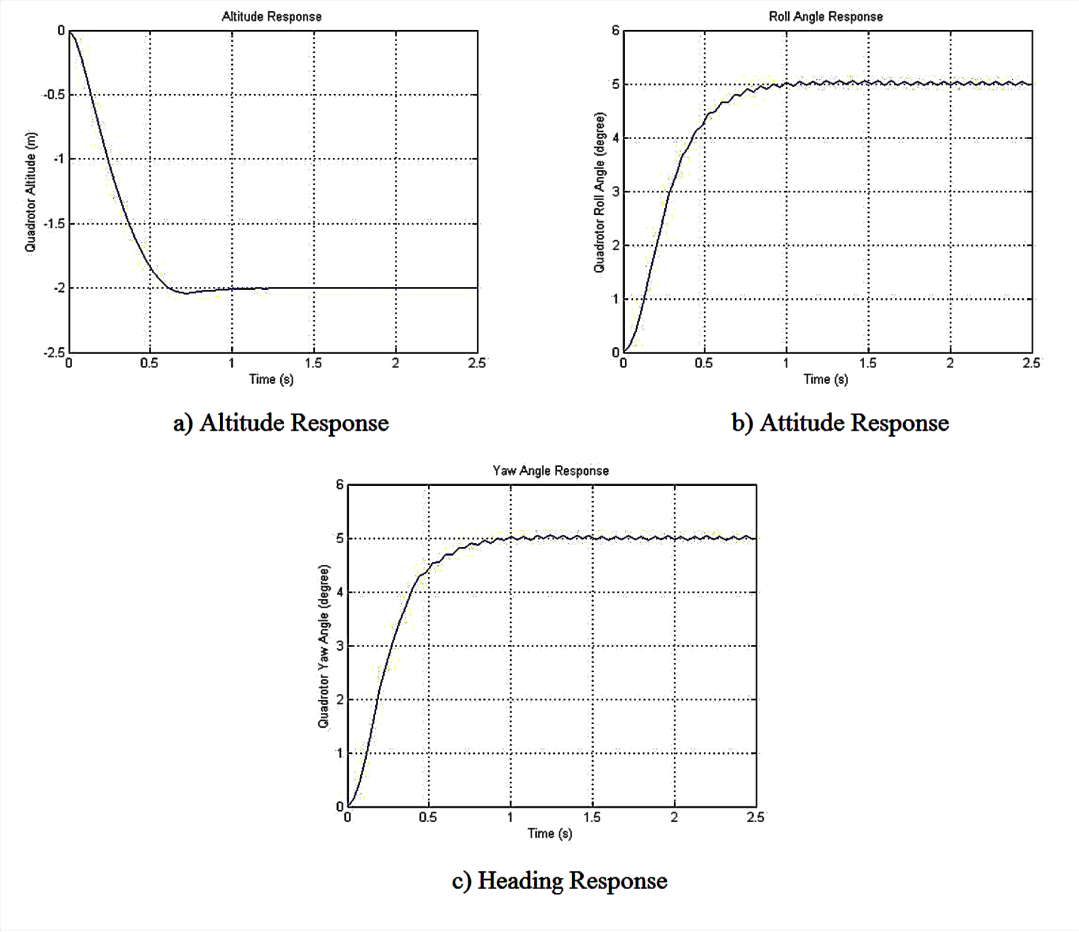
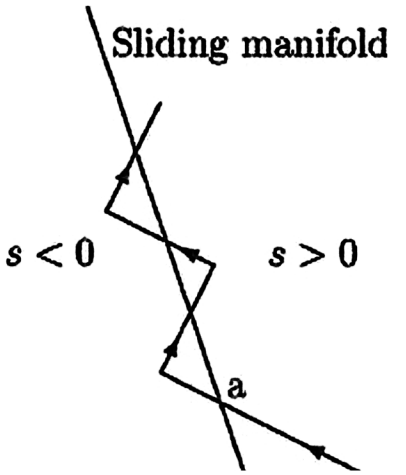


Figure 15. The chattering effect
[González et al. (2014)].



Simulations

The result of Equation 1.52 was used as an objective function for the GA to calculate the design gains. Table 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 16.

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 affected. It resulted in an increase in the altitude's settling time from 0.57 sec to 3.11 sec. 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.

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 17 shows graphs of both functions. The *sat* function is defined by,

$$sat(s) = \begin{cases} s & \text{if } \mid s \mid \leq 1; \\ sgn(s) & \text{if } \mid s \mid > 1. \end{cases} \tag{1.53}$$

Thus, to implement this modification on the SMCs for our system, the *sgn(s)* terms in the control laws should be replaced by $sat(s / \epsilon)$. 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 [Fuh et al. (2008)].

Simulations

Using the obtained control parameters in Table 5 with the modified control laws (replacing *sgn(s)* by $sat(s / \epsilon)$ in the control laws), chattering was eliminated as shown in Figure 18, the response graphs for altitude, attitude and heading respectively. Moreover, the quadrotor was commanded to follow a circular trajectory and its response graph is shown in Figure 19.

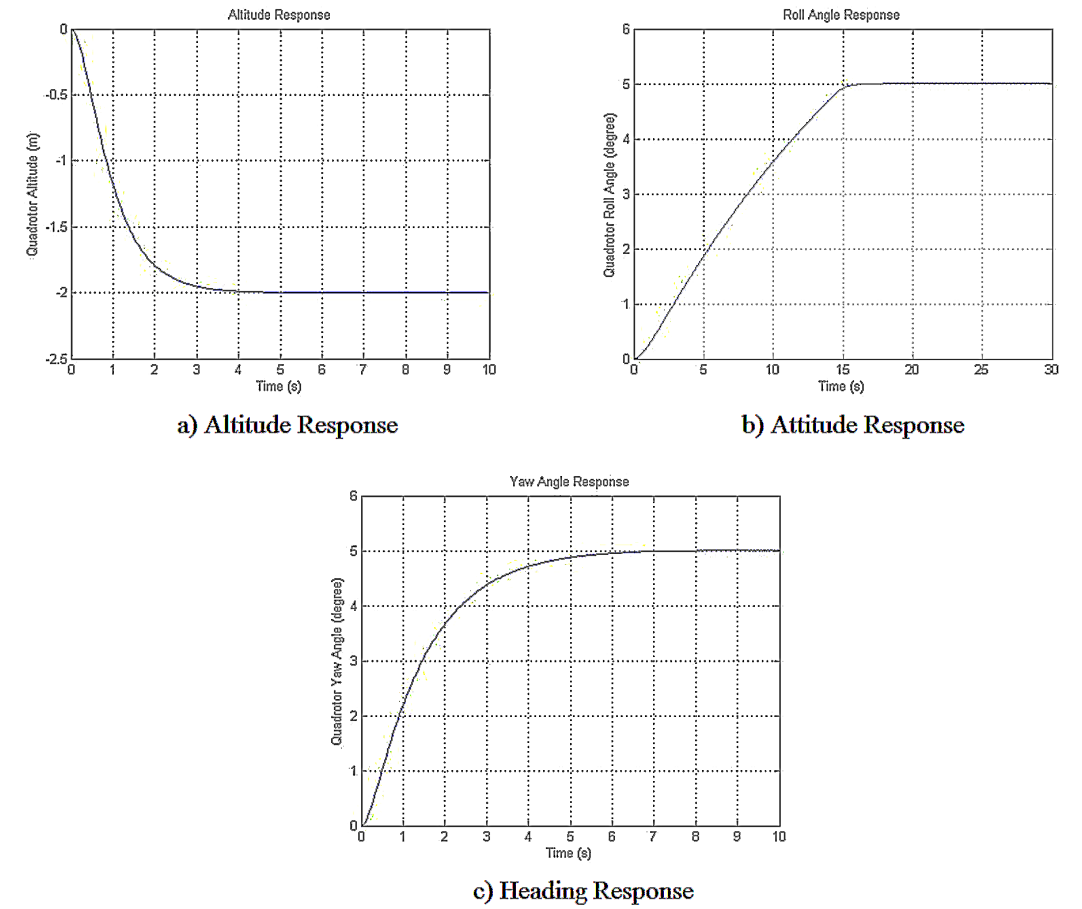
Backstepping Controller

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 previously derived.

Table 6. SMC results with minimal chattering

	Desired Value	c	k_1	k_2	Setting 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

Figure 16. SMC controller simulation response with chattering reduction



a) Altitude Response

b) Attitude Response

c) Heading Response

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 [Krstic et al. (1995)]. 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 [Marquez (2003), Krstic et al. (1995)].

Figure 17. Sgn vs. sat functions
[González et al. (2014)].

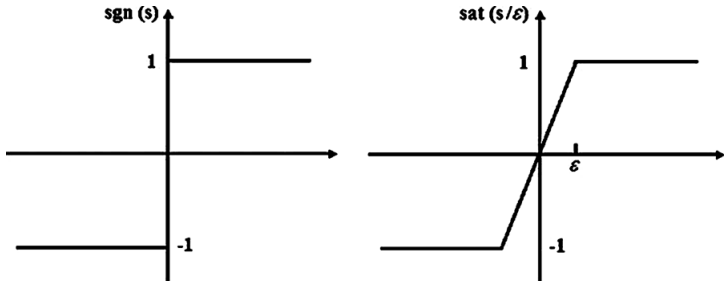
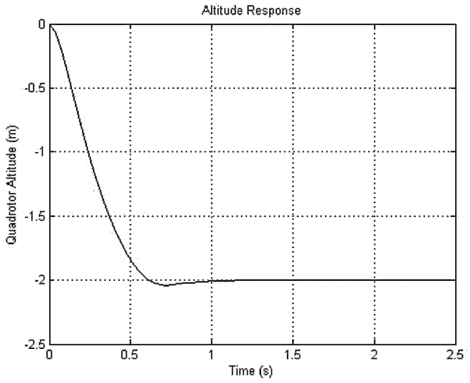
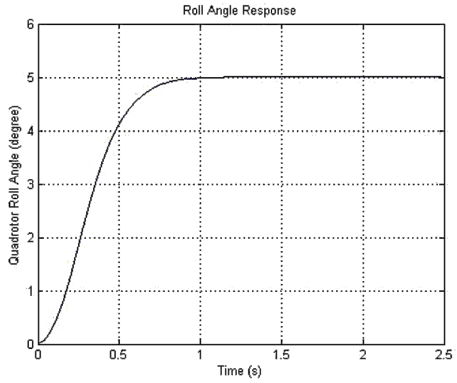


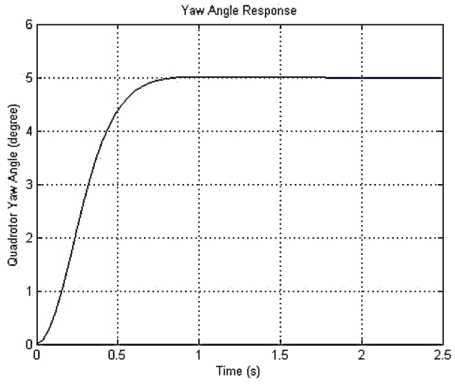
Figure 18. Modified SMC controller simulation



a) Altitude Response

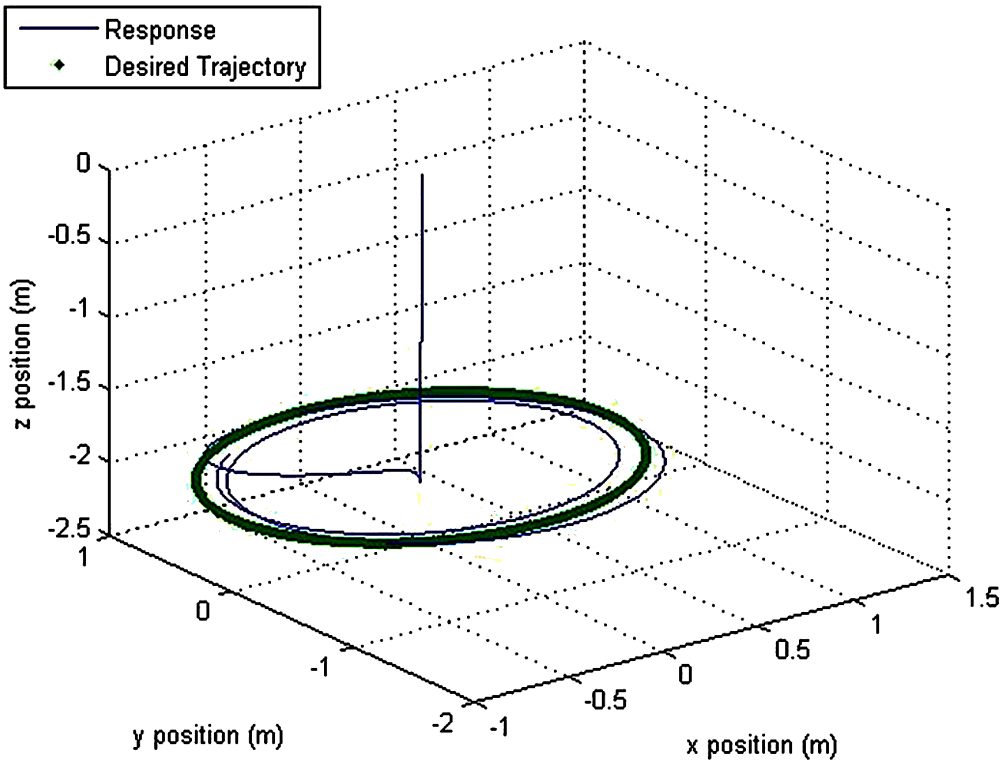


b) Attitude Response



c) Heading Response

Figure 19. Trajectory response under SMC



Attitude and Heading Control

The backstepping controller implemented to control the quadrotor's orientation is based on the control approaches proposed in [Nagaty et al. (2013)] and [Bouabdallah & Siegwart (2005)]. For the roll controller, the first two states of the state space model are used which are the roll angle and its rate of change. Extracting those we get,

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= x_4 x_6 a_1 - x_4 \Omega_r a_2 + b_1 U_2 \end{aligned} \tag{1.54}$$

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,

$$V_1 = \frac{1}{2} z_1^2 \tag{1.55}$$

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

$$z_1 = x_{1d} - x_1 \tag{1.56}$$

The time derivative of the Lyapunov function defined in Equation1.52 is derived to be,

$$\begin{aligned} \dot{V}_1 &= z_1 \dot{z}_1 \\ &= z_1(\dot{x}_{1d} - \dot{x}_1) \end{aligned} \tag{1.57}$$

and from Equation 1.54 this can be rewritten as,

$$\dot{V}_1 = z_1(\dot{x}_{1d} - x_2) \tag{1.58}$$

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 [Krstic et al. (1995)]. To achieve this, we choose a positive definite bounding function $W_1(z) = c_1 z_1^2$ to bound $\dot{V}_1$ as in Equation 1.59. This choice of $W_1(z)$ is also a common choice for a bounding function for strict feedback systems [Krstic et al. (1995)].

$$\dot{V}_1 = z_1(\dot{x}_{1d} - x_2) \leq -c_1 z_1^2 \tag{1.59}$$

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

$$(x_2)_{desired} = \dot{x}_{1d} + c_1 z_1 \tag{1.60}$$

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

$$z_2 = x_2 - \dot{x}_{1d} - c_1 z_1 \tag{1.61}$$

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

$$\begin{aligned} \dot{V}_1 &= z_1 \dot{z}_1 \\ &= z_1(\dot{x}_{1d} - x_2) \\ &= z_1(\dot{x}_{1d} - (z_2 + \dot{x}_{1d} + c_1 z_1)) \\ &= -z_1 z_2 - c_1 z_1^2 \end{aligned} \tag{1.62}$$

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,

$$V_2 = V_1 + \frac{1}{2} z_2^2 \tag{1.63}$$

with time derivative,

$$\begin{aligned} \dot{V}_2 &= \dot{V}_1 + z_2 \dot{z}_2 \\ &= -z_1 z_2 - c_1 z_1^2 + z_2 (\dot{x}_2 - \ddot{x}_{1d} - c_1 \dot{z}_1) \end{aligned} \tag{1.64}$$

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 1.54 leads to the following inequality,

$$\dot{V}_2 = -z_1 z_2 - c_1 z_1^2 + z_2 (x_4 x_6 a_1 - x_4 \Omega_r a_2 + b_1 U_2 - \ddot{x}_{1d} - c_1 \dot{z}_1) \leq -c_1 z_1^2 - c_2 z_2^2 \tag{1.65}$$

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

$$U_2 = \frac{1}{b_1} (-c_2 z_2 + z_1 - x_4 x_6 a_1 + x_4 \Omega_r a_2 + \ddot{x}_{1d} + c_1 \dot{x}_{1d} - c_1 x_2) \tag{1.66}$$

Repeating exactly the same steps for the pitch, heading and altitude, control laws are found to be,

$$\begin{aligned} U_1 &= \frac{m}{\cos x_1 \cos x_3} (-z_7 + g - \ddot{x}_{7d} - c_7 \dot{x}_{7d} + c_7 x_8 + c_8 z_8) \\ U_3 &= \frac{1}{b_2} (-c_4 z_4 + z_3 - x_2 x_6 a_3 - x_2 \Omega_r a_4 + \ddot{x}_{3d} + c_3 \dot{x}_3 d - c_3 x_4) \\ U_4 &= \frac{1}{b_3} (-c_6 z_6 + z_5 - x_2 x_4 a_5 + \ddot{x}_{5d} + c_5 \dot{x}_{5d} - c_5 x_6) \end{aligned} \tag{1.67}$$

where

$$\begin{aligned} z_3 &= x_{3d} - x_3 \\ z_4 &= x_4 - \dot{x}_{3d} - c_3 z_3 \\ z_5 &= x_{5d} - x_5 \\ z_6 &= x_6 - \dot{x}_{5d} - c_5 z_5 \\ z_7 &= x_{7d} - x_7 \\ z_8 &= x_8 - \dot{x}_{7d} - c_7 z_7 \end{aligned} \tag{1.68}$$

Backstepping Controller Simulation Results

The control inputs U_1 through U_4 derived in the previous section 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 7 shows the optimized parameters acquired from the GA and the resulting settling time for the attitude, heading and altitude of the quadrotor. Figure 20 shows the response when running the simulation with the constants in Table 7.

RESULTS DISCUSSION

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 21 shows the altitude response while Figure 22 and Figure 23 show the attitude and heading responses respectively.

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 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 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 of wind is shown in Figure 24 for PD, SMC and Backstepping.

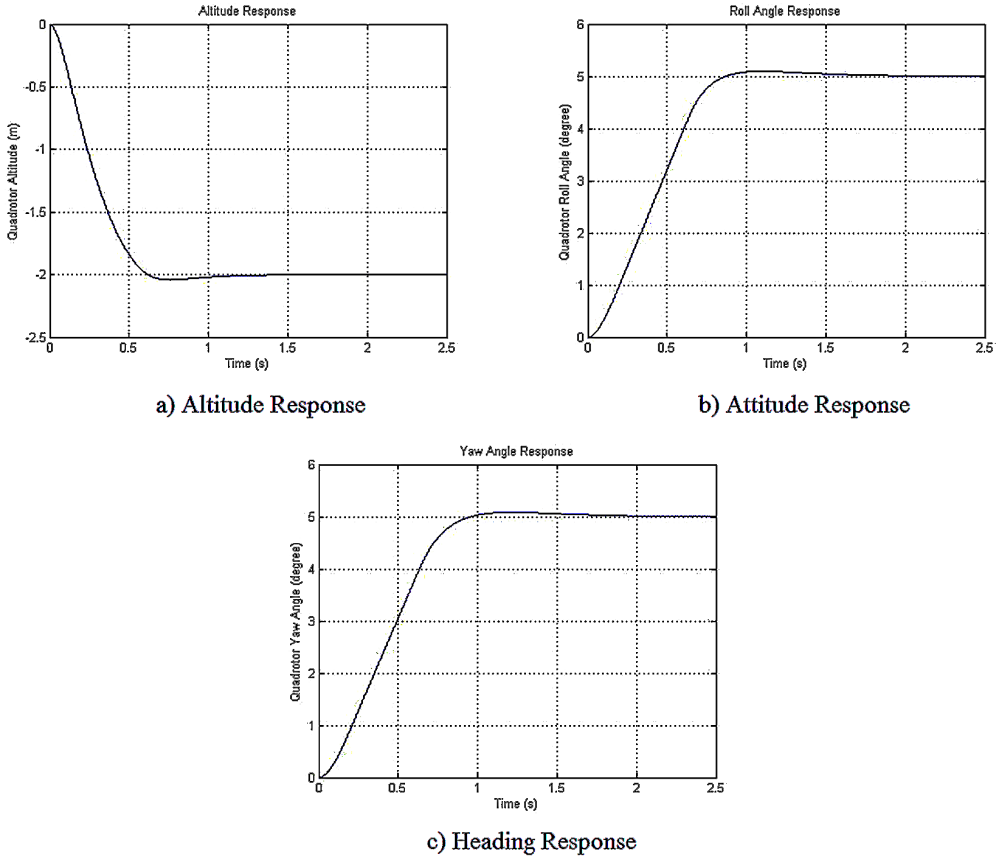
Nonhovering Operation

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 25. 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 Table 5 and Table 7 were used.

Table 7. Backstepping controller constants and results

	Desired Value	$c_1/c_5/c_7$	$c_2/c_6/c_8$	Setting Time	Overshoot
Attitude (φ and θ)	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%

Figure 20. Backstepping controller simulation response



a) Altitude Response

b) Attitude Response

c) Heading Response

Figure 21. Altitude response

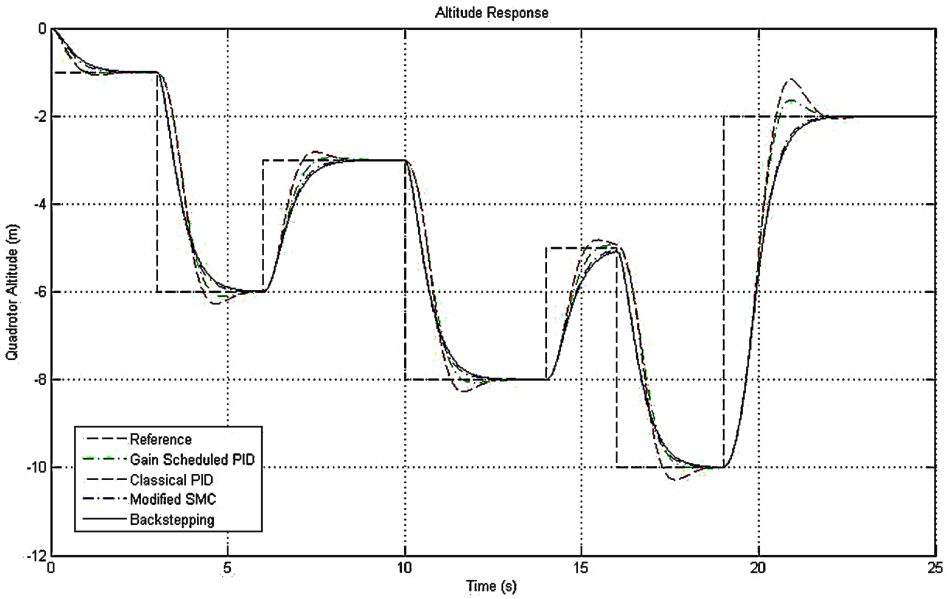


Figure 22. Attitude response

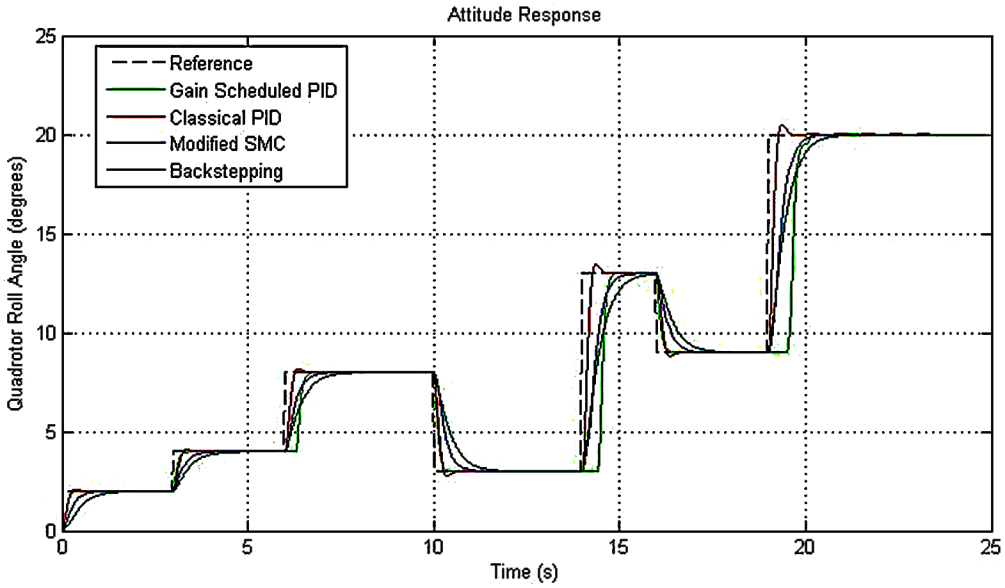


Figure 23. Heading response

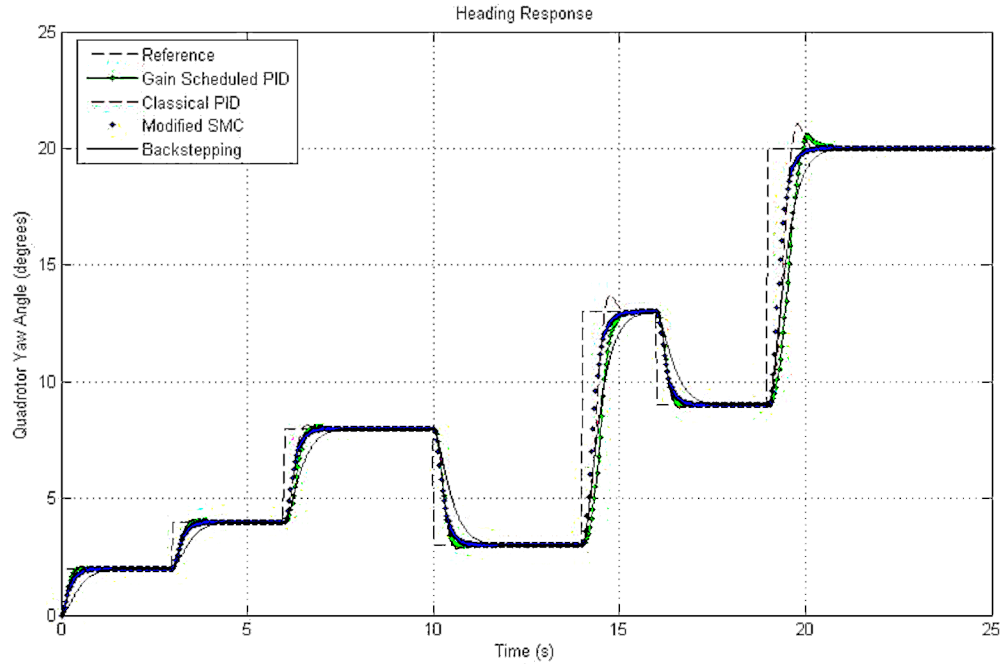
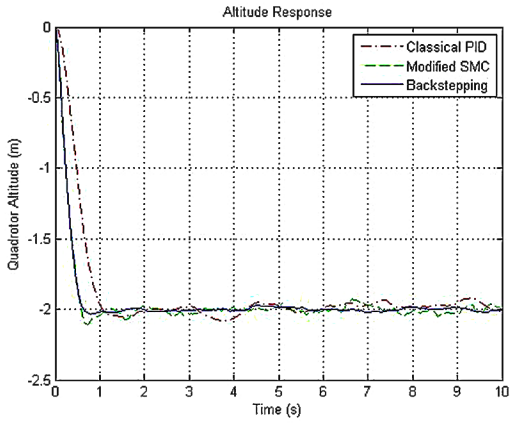
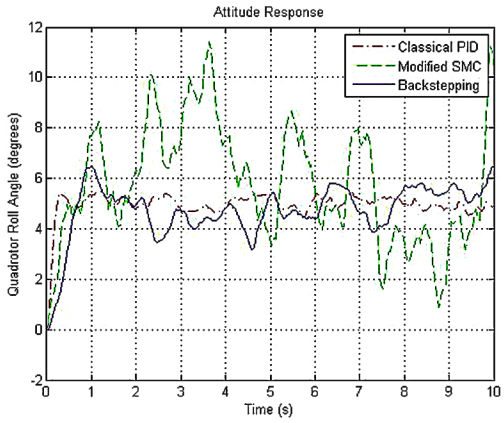


Figure 24. System response in a windy environment



a) Altitude Response with a desired altitude of 2m



b) Attitude Response with a desired attitude of 5 degrees

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

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.

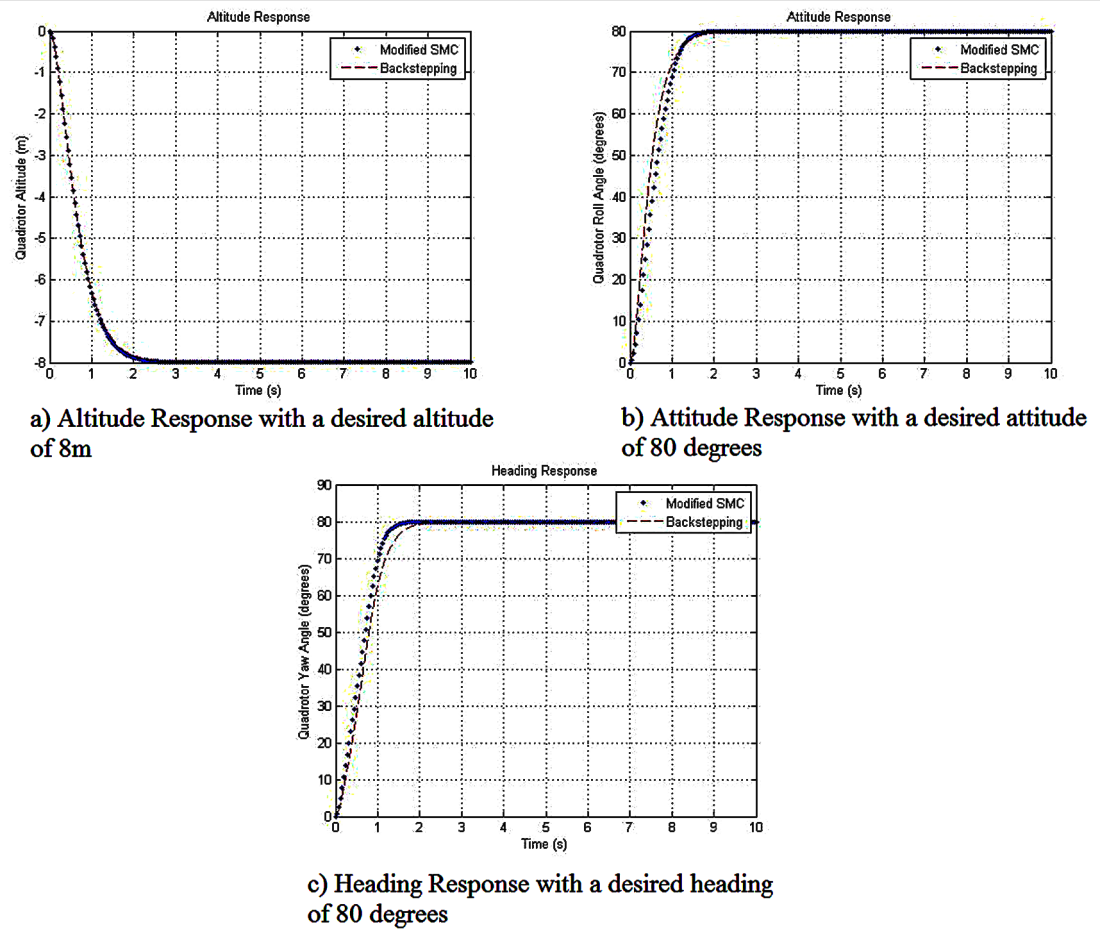
Nonlinear Operation

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 from 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. 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

Figure 25. System response when operated outside the linear region



a) Altitude Response with a desired altitude of 8m

b) Attitude Response with a desired attitude of 80 degrees

c) Heading Response with a desired heading of 80 degrees

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.

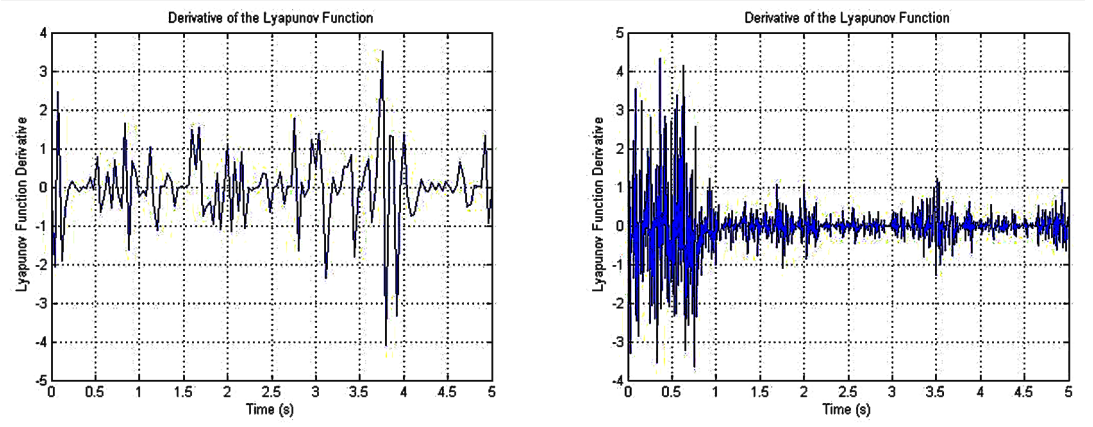
Windy Environment

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 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 26 shows the time derivative of the Lyapunov function under the effect of wind, while Figure 27 shows it in the no wind condition.

Choice of Controller

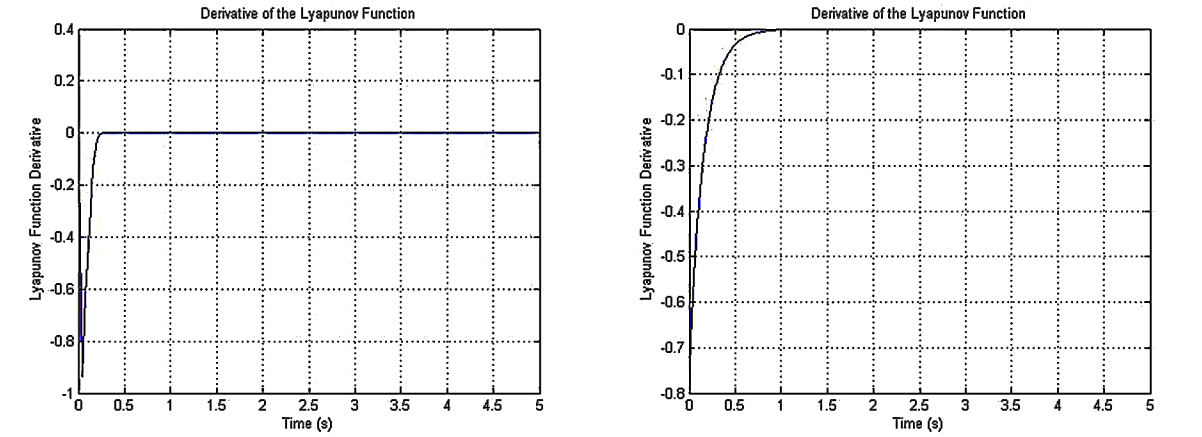
The choice of the controller to be used will depend mainly on the application, if the quadrotor is to be operating near a hovering condition, a PD controller will be sufficient to stabilize it. On the other hand, if it will be performing tough acrobatic maneuvers thus operating outside its linear region, 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.

Figure 26. Lyapunov function derivative under the effect of wind



a) SMC b) Backstepping

Figure 27. Lyapunov function derivative in no wind condition



a) SMC b) Backstepping

CONCLUSION AND FUTURE WORK

The goal of this work was to derive a mathematical model for the quadrotor 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. Four control techniques were then developed and synthesized; a linear Proportional-Integral-Derivative controller, a Gain Scheduling based PD controller, a nonlinear 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 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. Also, it is a good idea to study and implement global linearization methods including differential geometric approaches and differential flatness theory. Approximate linearization methods should also be tested, an example to these methods is the nonlinear H∞ control and the Takagi-Sugeno fuzzy control. Last but not least, implementing the developed control techniques on a real quadrotor hardware to give a more fair comparison between their performances.

REFERENCES

Alexis, K., Nikolakopoulos, G., & Tzes, A. (2011). Switching model predictive attitude control for a quadrotor helicopter subject to atmospheric disturbances. *Control Engineering Practice*, *19*(10), 1195–1207. doi:10.1016/j.conengprac.2011.06.010

Amoozgar, M. H., Chamseddine, A., & Zhang, Y. (2012). Fault-tolerant fuzzy gain scheduled pid for a quadrotor helicopter testbed in the presence of actuator faults. In *Proceedings of IFAC Conference on Advances in PID Control*. Academic Press.

Ataka, A., Tnunay, H., Inovan, R., Abdurrohman, M., Preastianto, H., Cahyadi, A., & Yamamoto, Y. (2013). Controllability and observability analysis of the gain scheduling based linearization for uav quadrotor. In *Proceedings of Robotics, Biomimetics, and Intelligent Computational Systems* (ROBIONETICS) (pp. 212–218). IEEE. 10.1109/ROBIONETICS.2013.6743606

Azzam, A., & Wang, X. (2010), Quad rotor arial robot dynamic modeling and configuration stabilization. In *Proceedings of Informatics in Control, Automation and Robotics* (CAR) (Vol. 1, pp. 438–444). Academic Press. 10.1109/CAR.2010.5456804

Bouabdallah, S. (2007). *Design and control of quadrotors with application to autonomous flying*. (Phd. thesis). Ecole Polytechnique Federale de Lausanne.

Bouabdallah, S., Noth, A., & Siegwart, R. (2004). PID vs LQ control techniques applied to an indoor micro quadrotor. In *Proceedings of Intelligent Robots and Systems, 2004* (IROS 2004) (Vol. 3, pp. 2451–2456). Academic Press. 10.1109/IROS.2004.1389776

Bouabdallah, S., & Siegwart, R. (2005). Backstepping and sliding-mode techniques applied to an indoor micro quadrotor. In *Proceedings of Robotics and Automation* (pp. 2247–2252). Academic Press. 10.1109/ROBOT.2005.1570447

Derafa, L., Madani, T., & Benallegue, A. (2006). Dynamic modelling and experimental identification of four rotors helicopter parameters. In *Proceedings of Industrial Technology* (pp. 1834–1839). Academic Press. 10.1109/ICIT.2006.372515

Efe, M. (2011). Neural network assisted computationally simple PID control of a quadrotor UAV. *IEEE Transactions on Industrial Informatics*, *7*(2), 354–361.

Fang, Z. & Gao, W. (2012). Adaptive backstepping control of an indoor micro-quadrotor. *Research Journal of Applied Sciences, 4*.

Fuh, C.-C., & ... . (2008). Variable-thickness boundary layers for sliding mode control. *Journal of Marine Science and Technology*, *16*(4), 286–292.

Gillula, J. H., Hoffmann, G. M., Huang, H., Vitus, M. P., & Tomlin, C. J. (2011). Applications of hybrid reachability analysis to robotic aerial vehicles. *The International Journal of Robotics Research*, *30*(3), 335–354. doi:10.1177/0278364910387173

González, I., Salazar, S., & Lozano, R. (2014). Chattering-free sliding mode altitude control for a quad-rotor aircraft: Real-time application. *Journal of Intelligent & Robotic Systems*, *73*(1-4), 137–155. doi:10.100710846-013-9913-8

Hou, H., Zhuang, J., Xia, H., Wang, G., & Yu, D. (2010). A simple controller of minisize quad-rotor vehicle. In *Proceedings of Mechatronics and Automation* (ICMA) (pp. 1701–1706). Academic Press. 10.1109/ICMA.2010.5588802

Kendoul, F. (2012). Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems. *Journal of Field Robotics*, *29*(2), 315–378. doi:10.1002/rob.20414

Kendoul, F., Yu, Z., & Nonami, K. (2010). Guidance and nonlinear control system for autonomous flight of minirotorcraft unmanned aerial vehicles. *Journal of Field Robotics*, *27*(3), 311–334.

Kim, J., Kang, M.-S., & Park, S. (2010). Accurate modeling and robust hovering control for a quadâ€“rotor vtol aircraft. *Journal of Intelligent & Robotic Systems*, *57*(1-4), 9–26. doi:10.100710846-009-9369-z

Krstic, M., Kokotovic, P. V., & Kanellakopoulos, I. (1995). *Nonlinear and adaptive control design*. John Wiley & Sons, Inc.

Lee, H., Kim, S., Ryan, T., & Kim, H. J. (2013), Backstepping control on se (3) of a micro quadrotor for stable trajectory tracking. In *Proceedings of Systems, Man, and Cybernetics* (SMC) (pp. 4522–4527). IEEE.

Li, J., & Li, Y. (2011). Dynamic analysis and PID control for a quadrotor. In *Proceedings of Mechatronics and Automation* (ICMA) (pp. 573–578). Academic Press. 10.1109/ICMA.2011.5985724

Liu, J., & Wang, X. (2012). *Advanced sliding mode control for mechanical systems: Design, analysis and MATLAB simulation*. Springer.

Madani, T., & Benallegue, A. (2006). Backstepping control for a quadrotor helicopter. In *Proceedings of Intelligent Robots and Systems* (pp. 3255–3260). Academic Press. 10.1109/IROS.2006.282433

Marquez, H. J. (2003). *Nonlinear control systems: Analysis and design*. John Wiley.

McNichols, K. H., & Fadali, M. S. (2003). Selecting operating points for discrete-time gain scheduling. *Computers & Electrical Engineering*, *29*(2), 289–301. doi:10.1016/S0045-7906(01)00031-3

Mistler, V., Benallegue, A., & M'Sirdi, N. (2001). Exact linearization and noninteracting control of a 4 rotors helicopter via dynamic feedback. In *Proceedings of Robot and Human Interactive Communication* (pp. 586–593). Academic Press. 10.1109/ROMAN.2001.981968

Nagaty, A., Saeedi, S., Thibault, C., Seto, M., & Li, H. (2013). Control and navigation framework for quadrotor helicopters. *Journal of Intelligent & Robotic Systems*, *70*(1-4), 1–12. doi:10.100710846-012-9789-z

Raffo, G. V., Ortega, M. G., & Rubio, F. R. (2010). An integral predictive/nonlinear hâ^ž control structure for a quadrotor helicopter. *Automatica*, *46*(1), 29–39. doi:10.1016/j.automatica.2009.10.018

Sadeghzadeh, I., Abdolhosseini, M., & Zhang, Y. M. (2012). Payload drop application of unmanned quadrotor helicopter using gain-scheduled PID and model predictive control techniques. In *Intelligent robotics and applications* (pp. 386–395). Springer. doi:10.1007/978-3-642-33509-9_38

Waslander, S., Hoffmann, G., Jang, J. S., & Tomlin, C. (2005). Multi-agent quadrotor testbed control design: integral sliding mode vs. reinforcement learning. In *Proceedings of Intelligent Robots and Systems* (IROS 2005). IEEE. 10.1109/IROS.2005.1545025

Yang, J., Cai, Z., Lin, Q., & Wang, Y. (2013), Self-tuning PID control design for quadrotor UAV based on adaptive pole placement control. In *Proceedings of Chinese Automation Congress (CAC)*. IEEE. 10.1109/CAC.2013.6775734

Zhen, H., Qi, X. & Dong, H. (2013). An adaptive block backstepping controller for attitude stabilization of a quadrotor helicopter. *WSEAS Transactions on Systems & Control, 8*(2).

KEY TERMS AND DEFINITIONS

Backstepping Controller: A recursive nonlinear controller that can be used to control a dynamic system given its model.

Gain Scheduling Control: An adaptive control approach that is used to control nonlinear systems by deploying a set of linear controllers for different linear operating regions.

Genetic Algorithm: An iterative optimization algorithm that works to minimize a given objective function by generating a random population and performing genetic operations to generate a new population.

Proportional Derivative (PD) Controller: A simplification of a PID controller, consisting only of a proportional and a derivative term.

Proportional Integral Derivative (PID) Controller: A linear controller that can be used to control any dynamic system given its model, it consists of a proportional, an integral and a derivative term.

Quadrotor: A class of UAV that flies with the aid of four separately powered rotors.

Sliding Mode Controller (SMC): A nonlinear controller of a switching nature that can be used to control any dynamic system given its model.

Unmanned Aerial Vehicle (UAV): A flying vehicle that is flown without a pilot which can be controlled autonomously or remotely.

This research was previously published in the Handbook of Research on Advancements in Robotics and Mechatronics edited by Maki K. Habib, pages 408-454, copyright year 2015 by Engineering Science Reference (an imprint of IGI Global).