Spring 2017

Simulated Annealing-Based Optimal Proportional-Integral-Derivative (PID) Controller Design: A Case Study on Nonlinear Quadcopter Dynamics

Kristofer Kevin Nemirsky

San Jose State University

Recommended Citation

Nemirsky, Kristofer Kevin, "Simulated Annealing-Based Optimal Proportional-Integral-Derivative (PID) Controller Design: A Case Study on Nonlinear Quadcopter Dynamics" (2017). *Master's Theses*. 4816.
DOI: https://doi.org/10.31979/etd.gx5m-sq86
https://scholarworks.sjsu.edu/etd_theses/4816

SIMULATED ANNEALING-BASED OPTIMAL PROPORTIONAL-INTEGRAL-DERIVATIVE (PID) CONTROLLER DESIGN: A CASE STUDY ON NONLINEAR QUADCOPTER DYNAMICS

A Thesis

Presented to

The Faculty of the Department of Aerospace Engineering

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Kristofer Kevin Nemirsky

May 2017

The Designated Thesis Committee Approves the Thesis Titled

SIMULATED ANNEALING-BASED OPTIMAL
PROPORTIONAL-INTEGRAL-DERIVATIVE (PID) CONTROLLER DESIGN:
A CASE STUDY ON QUADCOPTER DYNAMICS

by

Kristofer Kevin Nemirsky

APPROVED FOR THE DEPARTMENT OF AEROSPACE ENGINEERING

SAN JOSÉ STATE UNIVERSITY

May 2017

Kamran Turkoglu, Ph.D.	Dept. of Aerospace Engineering
Nikos J. Mourtos, Ph.D.	Dept. of Aerospace Engineering
Jeanine Hunter, M.S.	Dept. of Aerospace Engineering
Sean Montgomery, M.S.	Dept. of Aerospace Engineering

ABSTRACT

SIMULATED ANNEALING-BASED OPTIMAL PROPORTIONAL-INTEGRAL-DERIVATIVE (PID) CONTROLLER DESIGN: A CASE STUDY ON QUADCOPTER DYNAMICS

By Kristofer Kevin Nemirsky

In this thesis, the history and evolution of rotor aircraft with simulated annealing-based PID application were reviewed and quadcopter dynamics are presented. The dynamics of a quadcopter were then modeled, analyzed, and linearized. A cascaded loop architecture with PID controllers was used to stabilize the plant dynamics, which was improved upon through the application of simulated annealing (SA). A Simulink model was developed to test the controllers and verify the functionality of the proposed control system design. In addition, the data that the Simulink model provided were compared with flight data to present the validity of derived dynamics as a proper mathematical model representing the true dynamics of the quadcopter system. Then, the SA-based global optimization procedure was applied to obtain optimized PID parameters. It was observed that the tuned gains through the SA algorithm produced a better performing PID controller than the original manually tuned one. Next, we investigated the uncertain dynamics of the quadcopter setup. After adding uncertainty to the gyroscopic effects associated with pitch-and-roll rate dynamics, the controllers were shown to be robust against the added uncertainty. A discussion follows to summarize SA-based algorithm PID controller design and performance outcomes. Lastly, future work on SA application on multi-input-multi-output (MIMO) systems is briefly discussed.

TABLE OF CONTENTS

List of Tables vi

List of Figures vii

Nomenclature ix

1 Introduction 1
- 1.1 Overview 1

2 System Dynamics, Modeling, and Analysis 7
- 2.1 Overview 7
- 2.2 Theory and Implementation 7
- 2.3 Equations of Motion 9
- 2.4 Development of State Space Model 11
- 2.5 Parameter Estimation 15
- 2.6 Open Loop Analysis of State-Space Representation 15
- 2.7 Obtaining Brushless DC Motor (BLDC) Transfer Function 16

3 Controller Design 18
- 3.1 Design Considerations 18
- 3.2 Previous Control Design 18
- 3.3 Development of Optimal Pid Controller Using Simulated Annealing (SA) 19
- 3.4 Controllability and Observability 20

4 Simulink Model 22
- 4.1 Overview 22
- 4.2 Simulink: Overall Description 23
- 4.3 Control Mixing 24
- 4.4 Motor Dynamics 25
- 4.5 Input Relations 26
- 4.6 Angular and Translational Dynamic Sub-Blocks 27
- 4.7 Feedback Controller 29

5 Simulink Findings 32
- 5.1 Simulink Results 32
- 5.2 Simulated Annealing Results 38

6 Robustness Analysis 51

7 Conclusion 55

8 Future Steps 56

9 References 57

LIST OF TABLES

Table 2-1 – Identified system parameters 15

Table 4-1 – Control mix mapping 24

Table 4-2 – PID gains 30

Table 5-1 – Specific gain values 32

Table 5-2 – SA PID gains 48

LIST OF FIGURES

Figure 2-1 – Quadcopter setup with defined Euler angle axes 8

Figure 2-2 – CIFER® results to BLDC motor transfer function 17

Figure 3-1 – Cascaded control architecture with disturbances 18

Figure 4-1 – Simulink model of non linear dynamics 22

Figure 4-2 – Control mixing sub-block 25

Figure 4-3 – Motor dynamics sub-block... 26

Figure 4-4 – Input relations... 27

Figure 4-5 – Angular dynamics sub-block... 28

Figure 4-6 – Translation dynamics sub-block 29

Figure 4-7 – Feedback control sub-block 31

Figure 5-1 – Pitch angle comparison 33

Figure 5-2 – Pitch rate comparison.. 34

Figure 5-3 – Roll angle comparison... 35

Figure 5-4 – Roll rate comparison 36

Figure 5-5 – Yaw rate comparison... 37

Figure 5-6 – Pitch angle PID comparison.. 38

Figure 5-7 – Pitch rate PID comparison 39

Figure 5-8 – Roll angle PID comparison 40

Figure 5-9 – Roll rate PID comparison.. 41

Figure 5-10 – Yaw rate PID comparison 42

Figure 5-11 – Control effort comparison for pitch angle PID 43

Figure 5-12 – Control effort comparison for pitch rate PID 44

Figure 5-13 – Control effort comparison for roll angle PID... 45

Figure 5-14 – Control effort comparison for roll rate PID 46

Figure 5-15 – Control effort comparison for yaw rate PID 47

Figure 5-16 – Roll angle output and SA cost function 49

Figure 5-17 – Rate output and SA cost function.. 49

Figure 5-18 – Yaw rate output and SA cost function 50

Figure 6-1 – Attitude uncertainty for roll and pitch... 52

Figure 6-2 – Rate uncertainty for roll and pitch 53

Figure 6-3 – Yaw rate uncertainty 53

Figure 6-4 – 20% uncertainty distribution.. 54

NOMENCLATURE

6DOF	Six degrees of freedom
b	Thrust factor of the propeller
BLDC	Brushless DC motor
CCW	Counter-clockwise
CIFER	Comprehensive identification from frequency response
CW	Clockwise
d	Drag factor of the propeller
EoM	Equation of motion
GPS	Global positioning system
h	Gyroscopic effect
I	Moment of inertia about an axis
IMU	Inertial measurement unit
ITAE	Integral time absolute error
JTP	Moment of inertia about the propeller axis
k	Gyroscopic Effect
l	Distance from motor axis to the center
LIDAR	Light detection and ranging
MIMO	Multi-input-multi-output
MoI	Moment of inertia
MRAC	Model reference adaptive control
p	Roll rate
PID	Proportional-integral-derivative
PWM	Pulse width modulation
q	Pitch rate
r	Yaw rate
RPM	Revolutions per minute
SA	Simulated annealing
u	Velocity in the x-axis direction
U1	Vertical thrust factor
U2	Rolling torque factor
U3	Pitching torque factor
U4	Yawing torque factor
UAV	Unmanned aerial vehicle
v	Velocity in the y-axis direction
VTOL	Vertical take-off and landing
w	Velocity in the z-axis direction

θ	Pitch angle
φ	Roll angle
ψ	Yaw angle
$\mathbf{\Omega}$	Total propellers’ speed
$\mathbf{\Omega}_1$	Front right propeller speed
$\mathbf{\Omega}_2$	Rear right propeller speed
$\mathbf{\Omega}_3$	Rear left propeller speed
$\mathbf{\Omega}_4$	Front left propeller speed

1 Introduction

1.1 Overview

Quadcopters are aerial vehicles that have four rotor blades attached to a rigid frame. These four rotors control lift when working together, and when used in various combinations, they are used to control roll, pitch, and yaw [1]. Quadcopters (like helicopters) are capable of vertical take-off and landing (VTOL) [1]. They can also hover during flight [1]. During these flights, the flight envelopes are of great interest to the aerospace community, since they allow the vehicle to gain access to environments inaccessible to winged aircraft. Thus, they need to be understood thoroughly. These features are unique to quadcopters and make them useful for surveillance, search and rescue operations, construction inspections, interactive gaming, and medical applications [1, 2]. However, it was only recently (within the last 5-6 years) that quadcopters have received significant attention. Early designs were overly complex and performed poorly [3, 4]. Due to advancements in microelectronics, computer science, and microprocessors, current day quadcopters have become highly maneuverable unmanned aerial vehicles (UAVs) as opposed to their earlier manned counterparts. These advancements reduced the complexity of design, cost, and weight, resulting in increased viability and popularity as a research platform.

Inherent to the quadcopter design are nonlinear dynamics that provide an excellent testbed to verify concepts of control theory and underlying dynamics. In the professional literature, it is well documented that quadcopter dynamics are highly unstable and exhibit undesired flight characteristics in the absence of a controller [1-4], [21]. Moreover, quadcopters have six degrees of freedom (6DOF): three translational and three rotational. There are also four motor inputs: U1, U2, U3, and U4, which stands for vertical thrust, rolling, pitching and yawing factors, respectively. A 6DOF system that has fewer than six inputs produces an under actuated system. In this case, the quadcopter is an under actuated system since there are more degrees of movement than there are controlling mechanisms to generate each translational and rotational movement [20]. Having an under actuated system produces an additional challenge to stabilize the system dynamics. For the development of this type of controller, it was necessary to construct a proper mathematical model of quadcopter dynamics. This methodology will yield valuable information regarding flight performance and characteristics. It also will enable engineers to determine whether the system is fully controllable and the way each input affects the system as a whole.

In order to determine which control schemes are necessary to stabilize the plant dynamics of the quadcopter system, it is important to develop a dynamic model that correctly characterizes the quadcopter vehicle. Here, plant dynamics refers to the quadcopter dynamics. To this effect, it is common to first linearize the quadcopter dynamics about an operating (equilibrium) point and then apply a desired control methodology to guarantee stability and achieve preferred performance metrics [1]. Linearization reduces the complexity of the model and allows for simpler control schemes, such as a PID controller to stabilize the system [1]. However, this reduction in complexity comes at a cost. It omits valuable information from the true nonlinear dynamic model, such as the coupling effects between pitch, roll and yaw and the gyroscopic effects introduced by the differential propeller rates whenever the quadcopter is not trimmed [1]. Other control schemes like model reference adaptive control (MRAC) and L_1 adaptive control, which are more modern and sophisticated methodologies, guarantee system stability for nonlinear systems [5-10]. The latter control schemes bypass linearization and deal with the entire plant dynamics which makes these control techniques much more robust in design.

In particular, L_1 adaptive control, a powerful tool used to guarantee system stability, has been the focus of many engineers in academia as well as research labs in the field of control theory [5-10]. This is because the L_1 adaptive control has the ability to separate fast adaptation from the control loop, which allows for arbitrarily fast adaptation without sacrificing overall robustness [5]. Apart from other adaptive control system architectures, L_1 adaptive control has a design filter, which limits the bandwidth of the control signal ensuring that the system will not reach higher frequency modes [5]. The addition of the design filter allows for the decoupling of adaptation and robustness [5]. Therefore, controls engineers can analyze the properties of the closed loop system using linear control methods, making it an extremely useful control architecture [5]. Specifically, the choice of design filter defines which value the closed loop L_1 controller will reach [5]. These values include gain margin, phase margin, and time delay margins. Thus, the L_1 adaptive control architecture enables the ability to analyze any system, including non-linear systems, using linear design methods.

The hardware associated with quadcopters usually consists of an inertial measure unit (IMU), microcontroller, global positioning system (GPS), or any visual-based hardware like LIDAR, or ultrasound [1, 2], [20]. These measuring tools are essential for control applications since they provide the information for the control schemes listed above to function. The data measured from these sensors are susceptible to sensor noise and other external disturbances [20]. In the presence of these phenomena, it is conventional to smooth the data using complementary and Kalman filtering techniques [11].

Depending on the mission requirements, and if autonomous positional control was required, it is necessary to have visual-based tracking, through either GPS or live video feed during a flight regime [12, 13]. For this thesis however, autonomous positional control is not a requirement and therefore, it is not necessary to develop such a system. In cases where robust control is required, the focus should be on how well the control scheme is resilient against external disturbances from the environment or in fast moving references [14, 15].

Aside from hardware, there are many methodologies that are used to tune controllers. One well framed and classic application is the Ziegler Nichols method [27]. This method is relatively outdated, with better tuning methodologies having been developed to yield more sophisticated controllers and better results. These controllers reduce the error between the commanded signal and the output generated by the commanded input signal [28]. The error signal is linked to the performance of the controller and is used as a metric in defining the robustness of a controller. Simulated annealing (SA) is one such tuning method and was explored and applied to the quadcopter configuration in this thesis [16].

SA is a methodology that is originated from a paper published by Metropolis et al. in 1953 [17]. Metropolis et al. introduced an algorithm that simulates the cooling of a material in a heat bath, a process known as annealing. The main idea behind annealing involves a metal being heated up to its melting point and then slowly cooled to a stable and frozen state at a controlled rate. By controlling the rate at which the metal cools, crystalline structures within the metal can reach their lowest energy state (i.e., stable or frozen state). This process inspires the simulated annealing algorithm. The algorithm simulates the cooling process by lowering the temperature of the system until the system is able to converge to a stable state. The SA algorithm utilizes cooling schedules to select the optimal parameters, which repeatedly generates, judges, and accepts/rejects the control parameters [18].

Kirkpatrick et al. in 1982 first introduced the idea of utilizing the process of annealing metals to global optimization of functions [16-19]. An algorithm was developed to solve combinatorial optimization problems by “minimizing the functions of many variables” [16]. SA’s main advantage over other optimization methods that preceded it is its ability to avoid being trapped in local minima. This was achieved through an objective function, which was weighted to accept “worse” solutions with the intent of finding better ones within a larger space. Bertsimas Dimitris and John Tsitsiklis (1993) discuss a great analogy if the reader wishes to obtain a better understanding.

The main goal was to present a simulated annealing-based optimization methodology to tune a PID controller that is capable of stabilizing the nonlinear quadcopter plant. Currently, the configuration is the quadcopter setup developed by Ankyda Ji and the associated Arduino software [20]. The quadcopter uses a cascaded loop control architecture, where the inner loop utilizes angular rate information and the outer loop utilizes angle information. Together, the two PID controllers are able to stabilize the plant, which renders the quadcopter flight ready.

2 System Dynamics, Modeling, and Analysis

2.1 Overview

2.2 Theory and Implementation

The quadcopter platform uses a crossbeam structure, in the shape of an X. At the end of each beam, a motor is mounted and is controlled by the electronics located in the center of the main frame. The four motors are the only user-defined inputs for the (6DOF) system, making it an under-actuated system. This makes it necessary to have a controller that can compensate for the under-actuated dynamics of the quadcopter platform. Since there are no mechanical linkages to change blade angle (pitch), the motors are stationary and utilize differential torque to accomplish flight maneuvers [1, 2], [20]. The four motors are capable of controlling all states (position, velocity, orientation, and angular velocities) using differential torque. Figure 2-1 visualizes this configuration with a simple diagram [20]. The image on the left shows a configuration in which each motor was separated by a distance l away from the center console, where the flight controller is located. The image in the right of Figure 2-1 represents the body frame coordinate system and the corresponding Euler angles: pitch, roll, and yaw. In this setup, propellers Ω_1 and Ω_3 rotate clockwise (CW) and propellers Ω_2 and Ω_4 rotate counter-clockwise (CCW). With the quadcopter setup discussed, it is important to define how the quadcopter commands throttle, pitch, roll, and yaw.

To command throttle, all four propellers must rotate at the same angular rate to provide a force along the z-axis. The force that each propeller provides must be equal to at least a quarter of the weight of the quadcopter. In this case, the quadcopter is capable of providing enough force to oppose its own weight.

To execute a pitch motion, the front propellers Ω_1 and Ω_4 are either increased or decreased while the rear propellers Ω_2 and Ω_3 are given the opposite action. Commanding this action produces rotation along the y-axis.

To roll the quadcopter, simply changing the torque generated by propellers Ω_3 and Ω_4 or propellers Ω_1 and Ω_2 produces a torque along the x-axis. This creates the rolling motion.

Lastly, to command yaw, the clockwise spinning motors must increase or decrease, while the counterclockwise spinning motors must provide the opposite action. This combination produces rotation along the z-axis.

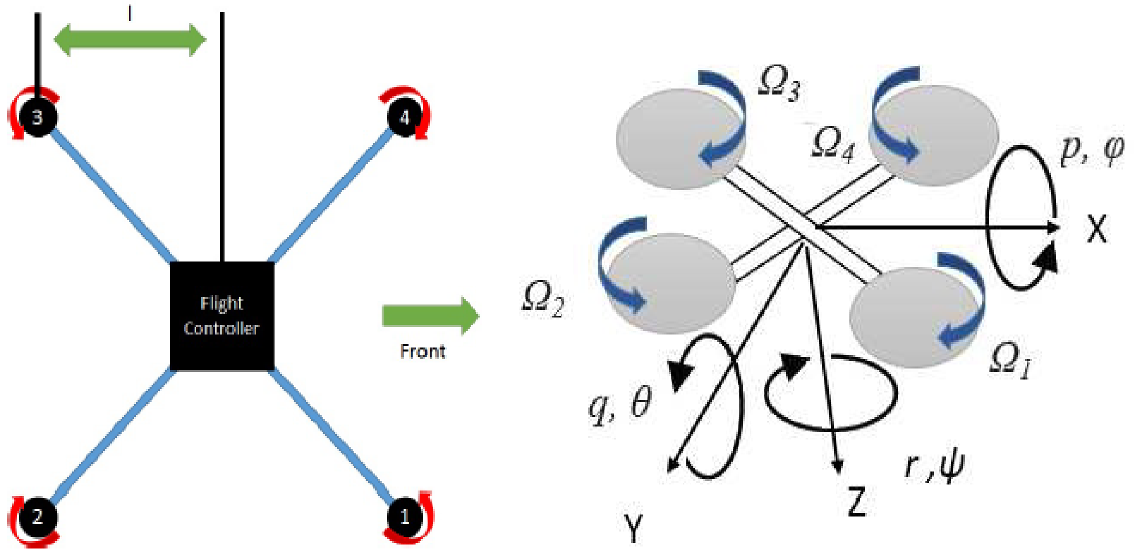

Figure 2-1 – Quadcopter setup with defined Euler angle axes

2.3 Equations of Motion

The literature shows how important it is to define an accurate model that properly characterizes the dynamics of any system [1-4], [20, 21]. For the quadcopter setup, it is no different. As such, the bulk of effort has been to develop an accurate mathematical model of the quadcopter dynamics. The equations of motion (EoMs) listed below are derived using Newton's second law of motion. The derivations of these equations uses two elementary assumptions: i) the quadcopter is a rigid body, and ii) the quadcopter's mass is distributed evenly, such that it is symmetrical along the x and y axes. The EoMs that describe the dynamics of the quadcopter found in a paper written by Bresciani, T. and are listed below.

$$\dot{u} = (vr - wq) - gS_{\theta} \tag{1}$$

$$\dot{v} = (wp - ur) + gC_{\theta}S_{\varphi} \tag{2}$$

$$\dot{w} = (uq - vp) + gC_{\theta}S_{\varphi} - \frac{U_1}{m} \tag{3}$$

$$\dot{p} = \frac{I_{YY} - I_{ZZ}}{I_{XX}}qr + \frac{J_{TP}}{I_{XX}}q\Omega + \frac{U_2}{I_{XX}} \tag{4}$$

$$\dot{q} = \frac{I_{ZZ} - I_{XX}}{I_{YY}}pr - \frac{J_{TP}}{I_{YY}}p\Omega + \frac{U_3}{I_{YY}} \tag{5}$$

$$\dot{r} = \frac{I_{XX} - I_{YY}}{I_{ZZ}}pq + \frac{U_4}{I_{ZZ}} \tag{6}$$

Eqn. 1 through Eqn. 3 govern translational motion, and Eqn. 4 through Eqn. 6 govern rotational motion of the quadcopter, which are Euler-defined. The nonlinearities described earlier are best represented in Eqn. 1 through Eqn. 6 by the coupling terms between roll, pitch, and yaw dynamics (p, q, and r, respectively). These equations, along with the controller currently used on the quadcopter platform, were implemented into MATLAB®'s Simulink software. U1, U2, U3, U4 are associated with throttle, pitch, roll and yaw, respectively, and are the inputs to the EoMs. In Eqn. 4 and Eqn. 6, the term Ω represents the summed total of the four motor angular rates. The inputs themselves are functions of the squared rotational velocities of each motor, which are multiplied by the lift and drag factor (b and d, respectively) as well as the distance from the center console, l. The lift and drag factors were calculated from blade element theory [21] and the inputs can be described as follows.

$$U_1 = b(\Omega_1^2+\Omega_2^2+\Omega_3^2+\Omega_4^2) \quad \textbf{(7)}$$

$$U_2 = lb(-\Omega_1^2-\Omega_2^2+\Omega_3^2+\Omega_4^2) \quad \textbf{(8)}$$

$$U_3 = lb(\Omega_1^2-\Omega_2^2-\Omega_3^2+\Omega_4^2) \quad \textbf{(9)}$$

$$U_4 = d(-\Omega_1^2+\Omega_2^2-\Omega_3^2+\Omega_4^2) \quad \textbf{(10)}$$

$$\Omega = (-\Omega_1+\Omega_2-\Omega_3+\Omega_4) \quad \textbf{(11)}$$

2.4 Development of State Space Model

It was necessary to develop a linearized model since it will aid in understanding of the system dynamics. Many mathematical tools apply to linearized systems that will help with developing and analyzing a control design. One such advantage that applies for a linear system is the principle of superposition, which allows each input-output relationship to be analyzed independently. Linearization of a nonlinear system allows an engineer to use tools such as bode plots to obtain valuable information regarding how the system will respond over wide frequency ranges [28]. In addition, it enables analyses through the root locus method, where poles are intelligently placed based on the desired phase and gain margins [28]. These tools inevitably affect the transient response characteristics in the time domain and frequency response in the frequency domain. By taking advantage of these traits, it is possible for rise time, steady state error, overshoot, gain margin, and phase margin to be within an acceptable margin. For this reason, the equations listed in Section 2.2 will be linearized, which will allow for easier open and close loop analyses of the system dynamics. In literature, the general form of a set of linearized equations is known as the state space representation [20]. The general form of the state space representation is in Eq. 12 [20].

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du \end{aligned} \tag{12}$$

In Eqn. 12, x represents the state variables of the system and u represents the user-defined inputs. Y is the output of the system. Eqn. 1 to Eqn. 6 are linearized about the hover equilibrium state. It was important to linearize about an equilibrium state because it guarantees stability within the vicinity of that operating point, which in this case is the hover condition.

Eqn. 13 represents the state vector as well as the operating point values.

$$x_0 = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ z_0\ 0\ 0\ 0\ ]^T \qquad x = [u\ v\ w\ p\ q\ r\ x\ y\ z\ \varphi\ \theta\ \psi]^T \tag{13}$$

To linearize the plant, the Taylor series expansion is utilized with the inclusion of the perturbation states. Eqn. 14 and Eqn. 15 represents the linearization process and the perturbation state, respectively.

$$f(x) = f(x_0) + \left(\frac{\partial f}{\partial x}\bigg|_{x=x_0}\right)(x - x_0) + higher\ order\ terms \tag{14}$$

$$\delta x = x - x_0 \tag{15}$$

If the quadcopter does not stray too far away from the operating point, the higher order terms are near zero and do not affect the dynamics. However, if the quadcopter strays too far from the operating point, then the linearized dynamics are no longer correct. For this reason, it is important to stay within the boundaries of the operating point.

Applying Eqn. 14 and Eqn. 15 together yields the following result.

$$\delta \dot{x} = a\delta x \tag{16}$$

Where the variable a is a square matrix comprised of a system of linearized equations.

The same logic to linearize Eqn. 14 can be applied to user inputs as well. Since both inputs and EoMs are linearized, the principle of superposition applies which allows them to be simply added together. Applying the Taylor series expansion to the inputs and inserting the values into Eqn. 16 yields the following equation.

$$\delta \dot{x} = \sum_{j=1}^{n} \left(\left. \frac{\partial f_i}{\partial x_j} \right|_{x_j = x_{0j}} \right) (\delta x_j) + \sum_{j=1}^{m} \left(\left. \frac{\partial f_i}{\partial u_j} \right|_{u_j = u_{0j}} \right) (\delta u_j) \qquad \textbf{(17)}$$

Applying Eqn. 17 to linearize equations, Eqn. 1 to Eqn. 6, creates the following state space representation.

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & g & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -g & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -g & 0 & 0 \\ 0 & 0 & 0 & 0 & -h & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & k & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1/m & 0 & 0 & 0 \\ 0 & 1/I_{xx} & 0 & 0 \\ 0 & 0 & 1/I_{yy} & 0 \\ 0 & 0 & 0 & 1/I_{zz} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad \textbf{(18)}$$

$$C = I_{12x12}$$

$$D = [0]$$

Where h and k are constants, and are

$$h = k = \Omega\left(\frac{J_{TP}}{I_x}\right)$$

2.5 Parameter Estimation

The parameters listed in Table 2-1 were either obtained through measurement or were inherited by authors in [20]. The weight was re-measured using a weight scale to check for discrepancies. The moments of inertia (MoI) about the x, y, and z axes were obtained using the swing test method, which is fully documented in [21]. These MoI are with respect to the center of gravity of the body.

Table 2-1 – Identified system parameters

Parameter	Value	Unit
m (Body Mass)	1.51	Kg
I_{XX}	0.02	Kg*m^2
I_{YY}	0.02	Kg*m^2
I_{ZZ}	0.04	Kg*m^2
J_{TP} (Propeller Polar MoI)	14.2*10^{-4}	Kg*m^2
b (Thrust Factor)	4.5*10^{-4}	N.m.s^2
d (Prop. Drag Factor)	0.45*10^{-5}	N.s^2
l (Moment arm)	0.20955	Meters
h (Gyroscopic Effect)	9.8237	Rad*s^{-1}
k (Gyroscopic Effect)	9.8237	Rad*s^{-1}

2.6 Open Loop Analysis of State-space Representation

Inserting the values from the Table 2-1 into Eqn. 18, yields eigenvalues that are located on the complex axis. These results show that the system is neutrally stable with two eigenvalues at $\pm$ 9.8237i and the rest at 0. For any small perturbation added to the system, the quadcopter will become unstable and the poles will be driven to the right-hand-side of the complex plane.

From this analysis, it is clear that a controller and feedback system must be designed to stabilize the plant dynamics. Before designing a controller, however, it is necessary to check and see if the system is observable and controllable. This analysis is discussed in the next chapter.

2.7 Obtaining Brushless DC Motor (BLDC) Transfer Function

To obtain the transfer function that will properly model the brushless DC motor (BLDC), a step response was used. A hall effect sensor was used to measure the rotation rate of the rotor every time it makes one full rotation. An Arduino board was used to record the measurements from the hall effect sensor. After obtaining the data, it was put into a systems identification tool known as, comprehensive identification from frequency response (CIFER®), which uses a frequency approach to obtain system dynamic properties [26]. Using CIFER®, and the data generated, a transfer function that relates voltage input (step response), and revolutions per minute (RPM) output was created. The results show that a first order transfer function with a delay 0.121 represents the dynamics well with a coherence nearly 100%. The results are shown in Figure 2-2.

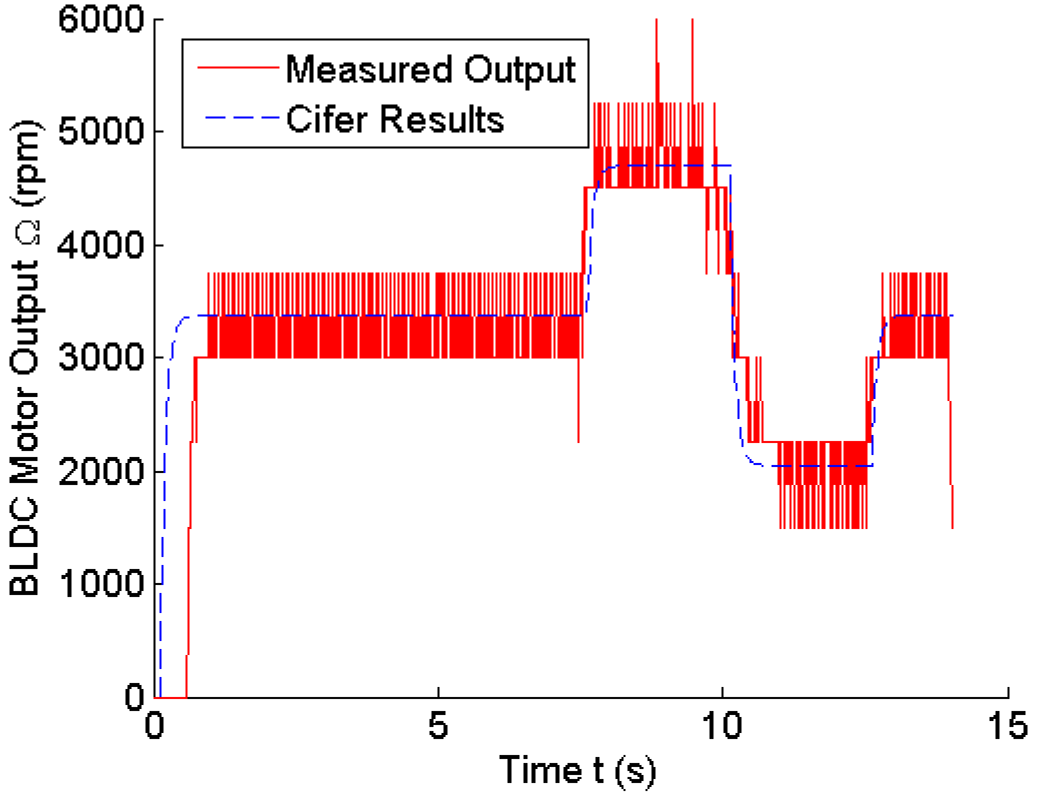

Figure 2-2 – CIFER® results to BLDC motor transfer function

Eqn. 19 represent the corresponding first order transfer function.

$$\frac{RPM(s)}{V(s)} = e^{-0.121s}\left(\frac{12.898}{s + 11.707}\right) \tag{19}$$

3 Controller Design

3.1 Design Considerations

3.2 Previous Control Design

For the control design, a PID controller was used to stabilize the plant dynamics. The conventional form of the PID controller is shown in Eqn. 20.

$$PID(s) = K_P + \frac{K_I}{s} + K_D s \tag{20}$$

The previous controller design used on the quadcopter was a cascaded control architecture that utilized an inner loop that feedbacks angular rate information and an outer loop that feedbacks attitude information. Here, the inner loop and the outer loop uses the state information produced by the quadcopter dynamics during simulation, compares it to the input, and tries to minimize the difference. This is known as the error signal that is defined as the difference between the commanded input signal and the measured output data [28]. Together the cascaded loop architecture stabilizes the quadcopter against any angular or angular rate disturbances. Figure 3-1 illustrates this cascaded control architecture [20].

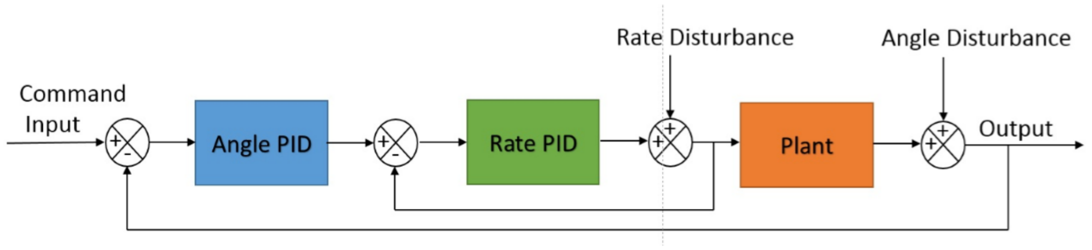

Figure 3-1 – Cascaded control architecture with disturbances

3.3 Development of Optimal PID Controller Using Simulated Annealing (SA)

The new controller that was developed is a PID controller, which was tuned using the methodology behind SA framework. The SA algorithm uses cooling schedules to select optimal parameters, and repeatedly generates, judges and accepts or rejects, the control parameters obtained [23-24], [16]. This was achieved through Eqn. 21 in which the Boltzmann probability distribution was altered to fit global optimization functions rather than annealing of metals.

$$p(taking\ the\ step\ \Delta x) = \exp\left[-\left(\frac{k}{k_{max}}\right)^{q}\left(\frac{\Delta f}{|f(x)|\varepsilon_f}\right)\right] \tag{21}$$

This method uses a cost function to define whether a solution is acceptable or not. Eqn. 22 to Eqn. 26 list the cost functions or performance indices that are available. The analysis shows that the integral time absolute error (ITAE) performance index has favorable characteristics that satisfy two goals of this paper: to eliminate steady state error and improve settling time. ITAE does this by placing more weight on the steady state error rather than the initial transient response [29]. This characteristic was more valuable to the quadcopter setup because it was best to eliminate steady state error. The other performance indices are added for convenience. Results using this methodology is described in Section 6.1.

$$ISE = \int_0^\infty e^2(t)dt \tag{22}$$

$$IAE = \int_0^\infty |e(t)|dt \tag{23}$$

$$ITAE = \int_0^\infty t|e(t)|dt \tag{24}$$

$$ITSE = \int_0^\infty t|e^2(t)|dt \tag{25}$$

$$MSE = {}^1\!/_t \int_0^\infty t|e^2(t)|dt \tag{26}$$

3.4 Controllability and Observability

Before developing a controller to stabilize the plant, it was necessary to check if all states are controllable and observable. This was done by checking if the controllability (CO) and observability (OB) matrices have full rank. They are

$$OB = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{bmatrix} \tag{27}$$

$$CO = \begin{bmatrix} B & AB & A^2B & \cdots & A^{n-1}B \end{bmatrix} \tag{28}$$

From this analysis, both OB and CO matrices have full rank. Therefore, the system was fully controllable and observable.

One novelty provided in this paper was the use of the SA algorithm to tune the gain values of a PID controller. Results show that the new PID gains show favorable response characteristics and outperform the original manually tuned PID controllers.

4 Simulink Model

4.1 Overview

Before implementing newer controllers or control architectures, it was important to develop a mathematical model that closely represents the actual dynamics of the quadcopter. This section will discuss the development of such a model and will verify the model by comparing its simulated output to flight data. Figure 4-1 shows the Simulink model in its entirety. The following sections discuss each sub-block in detail.

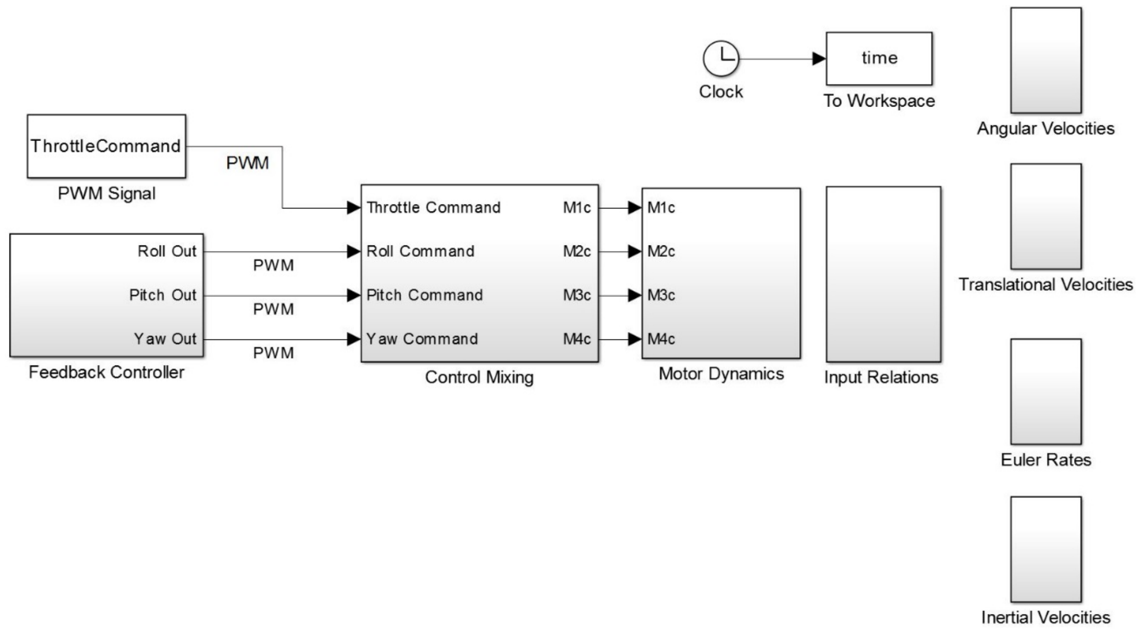

Figure 4-1 – Simulink model of non linear dynamics

4.2 Simulink: Overall Description

Figure 4-1 shows the overall Simulink model structure, which encompasses the user inputs, feedback controller, control mixing, input relations, and nonlinear dynamic functions. The inputs feed into the control-mixing sub-block as pulse width modulation (PWM) signals (0 to 255), in which the signals are properly mapped to the individual motor commands. Afterwards, the motor commands feed through the motor dynamics sub-block, which takes into account the motor offset and upper limits that the BLDC motors can output. Represented in Section 4.3, the motor dynamics maps the input (PWM) signal to propeller speed in radians per second. When this conversion is complete, the input relations sub-block converts the four individual propeller speeds to forces and torques. Finally, once the inputs are converted to thrust force and associated torques, the information is inputted into the dynamic sub-blocks, which are labeled angular velocities and translational velocities. Here, the mass properties like moments of inertia are taken into account as well as the total mass of the vehicle. This information along with the force and torque inputs are utilized to calculate the angular rates and attitude of the vehicle, which are sent to the feedback controller sub-block. The current control architecture in this model follows the cascaded architecture as described in Section 3.1.

4.3 Control Mixing

Control mixing enables Simulink to map the throttle, roll, pitch, and yaw inputs to the respective motors as motor commands. This mapping largely depends on how the body axis frame is defined. Because the body frame z-axis is positive pointing down for the given quadcopter setup, this affects how the control mixing is developed. Table 4-1 illustrates this mapping. It is important to ensure proper mapping to achieve a symmetrical stable configuration. If the roll, pitch, and yaw columns all sum to zero, this criterion is met. Otherwise, undesirable effects will occur. As can be seen in Table 4-1, the sum of the attitude inputs is zero. For throttle, it is desirable to have all inputs sum to the number of propellers mounted on the vehicle. In this case, this paper deals with a quadcopter, so there are four motors, and hence a sum of four. This ensures that the quadcopter can reach the hover condition. Figure 4-2 represents the sub-block for control mixing.

Table 4-1 – Control mix mapping

Motor #	Throttle	Roll	Pitch	Yaw
Motor 1	1	-1	1	-1
Motor 2	1	-1	-1	1
Motor 3	1	1	-1	-1
Motor 4	1	1	1	1
Total	4	0	0	0

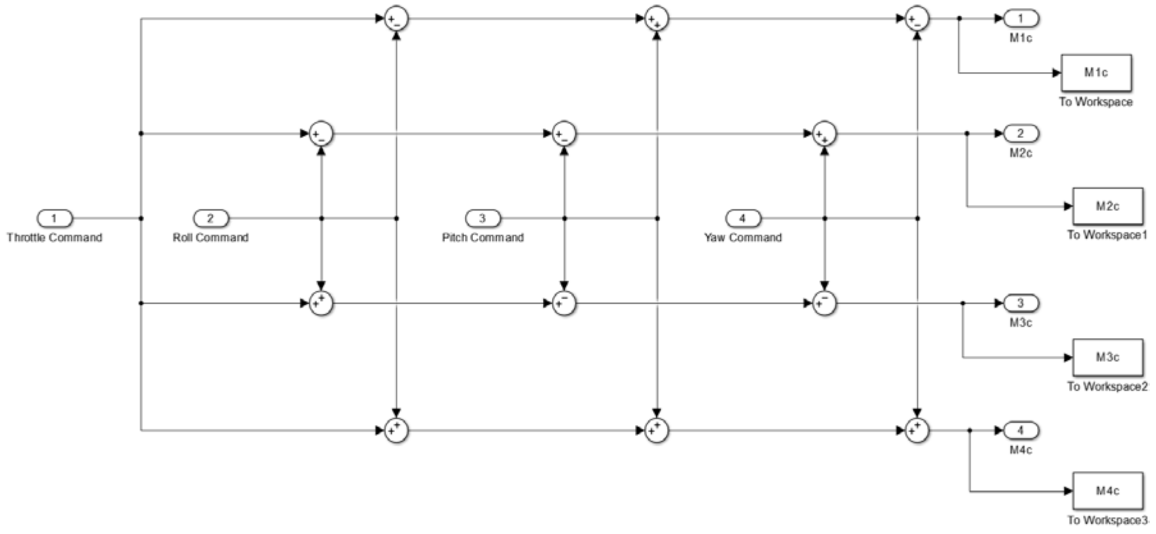

Figure 4-2 – Control mixing sub-block

4.4 Motor Dynamics

The motor dynamics represented in Figure 4-3, convert the individual motor control inputs generated by the control-mixing sub-block to output angular rates of the brushless DC motors. Before this conversion is completed, it is necessary to take into account the upper limit that the BLDC motors can output and the motor offset, which defines the minimum PWM signal necessary to produce output from the motors. After considering this, the motor dynamics sub-block converts the PWM signal to voltage through a gain, which feeds into a first order transfer function that represents the dynamics of the motor. The signal coming out of the motor dynamics feeds through a transport delay that represents the physical limitations of the motor. Using a transport delay value of 0.005 sec, the dynamics of the motor are properly modeled. Lastly, another gain converts the output of the motor dynamics from revolutions per second to radians per second, which feeds into the input relations sub-block.

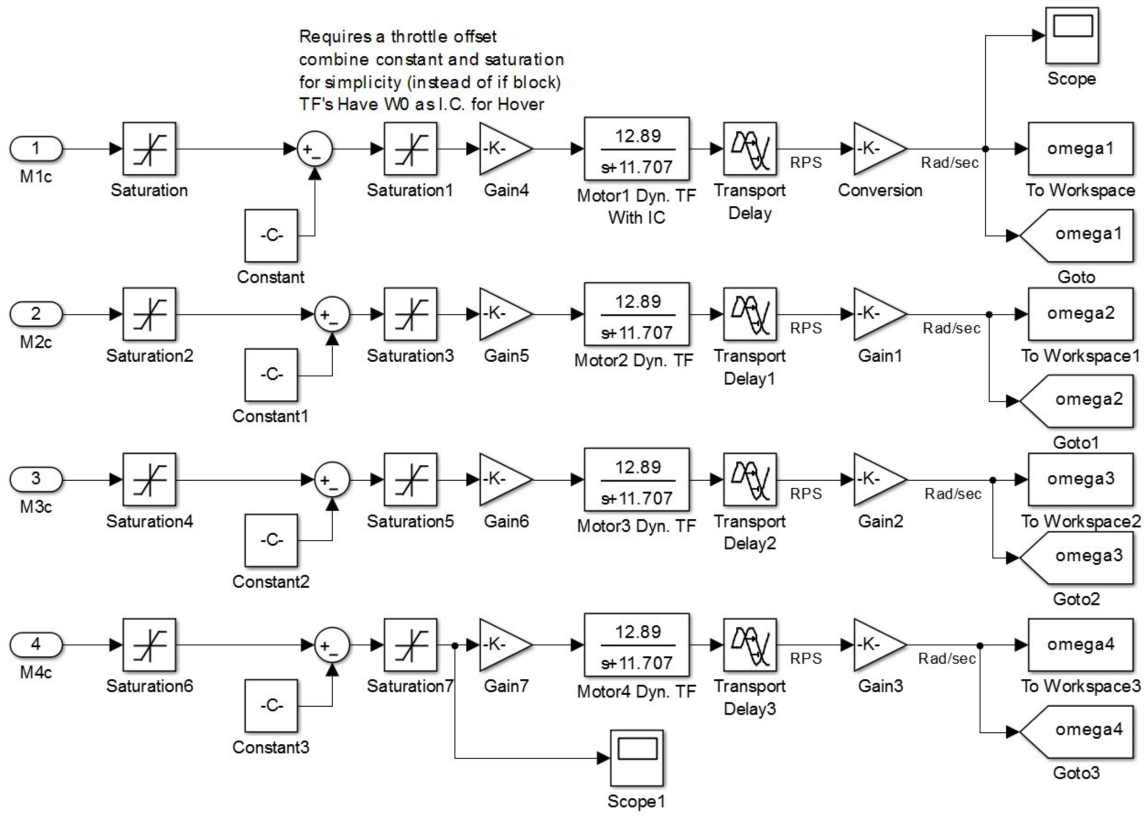

Figure 4-3 – Motor dynamics sub-block

4.5 Input Relations

This sub-block deals with the mapping between the motor dynamic outputs, which are in radians per second and the force and torques generated by the motor dynamics. Eqn. 7 through Eqn. 11 in Section 2.1 best illustrates this mapping. Derived from blade element theory [21], these equations relate the lifting forces generated by the propellers as the sum of the squared propeller angular rates. Figure 4-4 represents the sub-block as it was in the Simulink file. Here, U_1 is the thrust force, U_2 is the torque input along the rolling axis, U_3 is the torque input along the pitching axis, and U_4 is the torque input along the yawing axis. In addition, the parameters b, l, and d, are thrust factor, moment arm, and propeller drag factor, respectively.

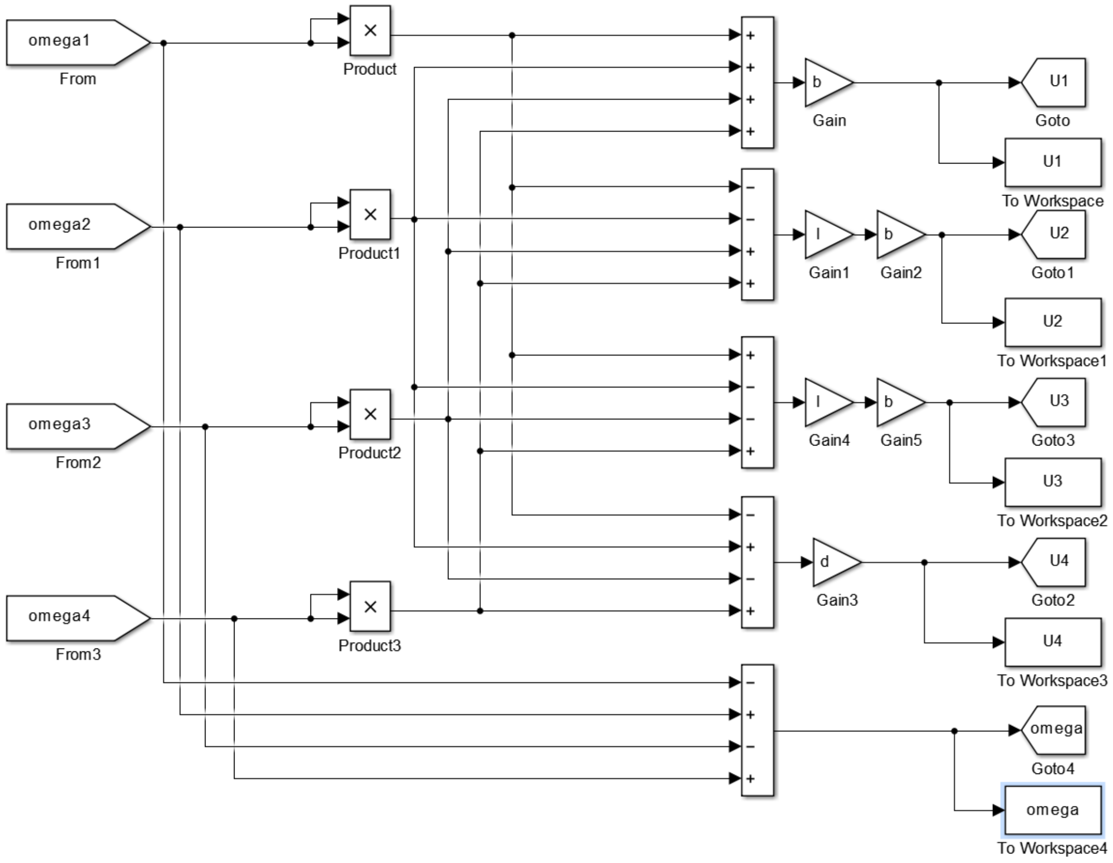

Figure 4-4 – Input relations

4.6 Angular and Translational Dynamic Sub-Blocks

The dynamics discussed in Section 2.2 are formulated here in the dynamics sub-blocks. Figure 4-5 and Figure 4-6 represent the angular and translational dynamics, respectively. These dynamics are with respect to the body fixed frame. Taken into account are the MoI along each axis, the total mass, and the coupling effects between roll, pitch, and yaw. It may be beneficial to represent the translation dynamics with respect to the inertial frame if it is desirable to control the z-axis positon of a quadcopter. When doing so, it is easier to develop controllers to control z-axis position. For the purposes of this paper, however, this was unnecessary but may be useful to add in future work. Therefore, 6DOF rigid body equations were sufficient.

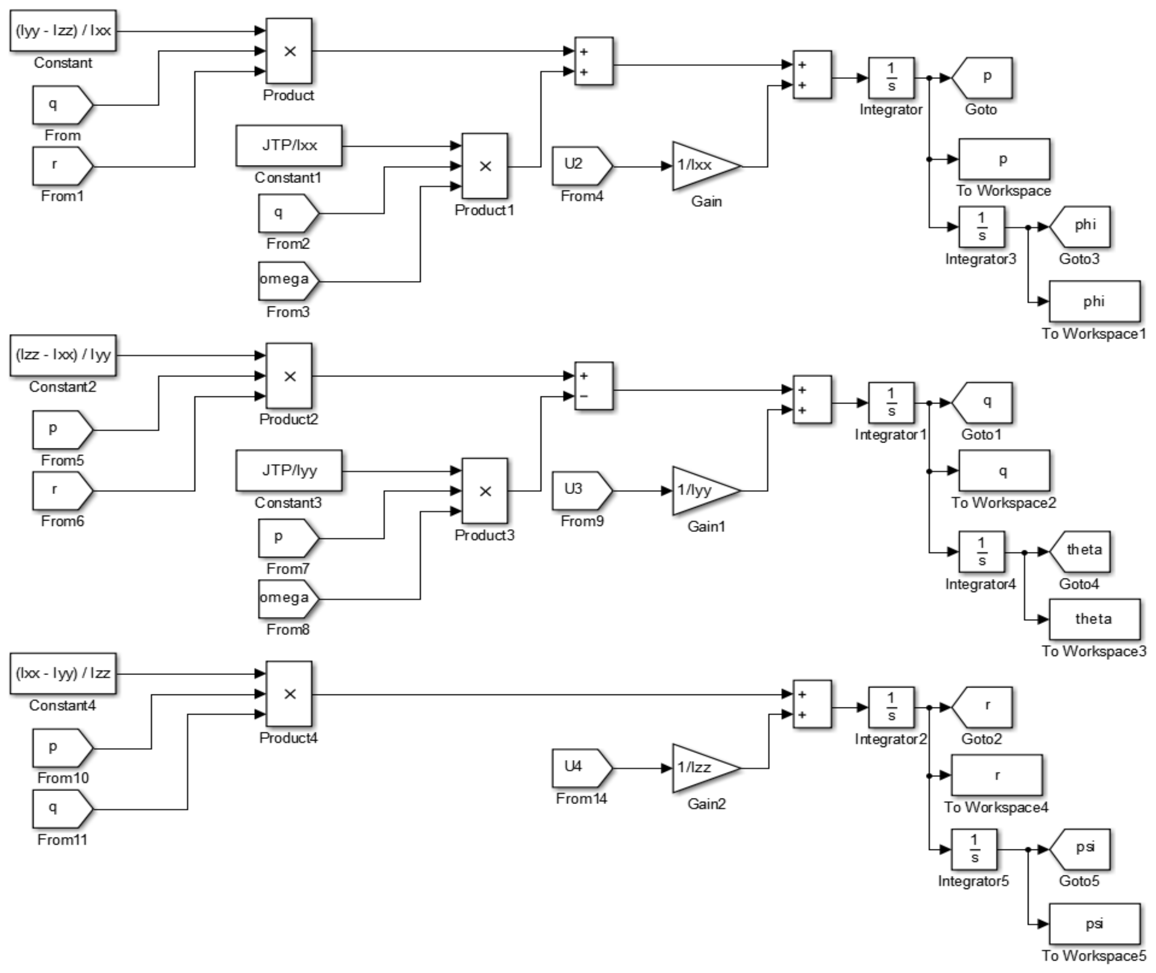

Figure 4-5 – Angular dynamics sub-block

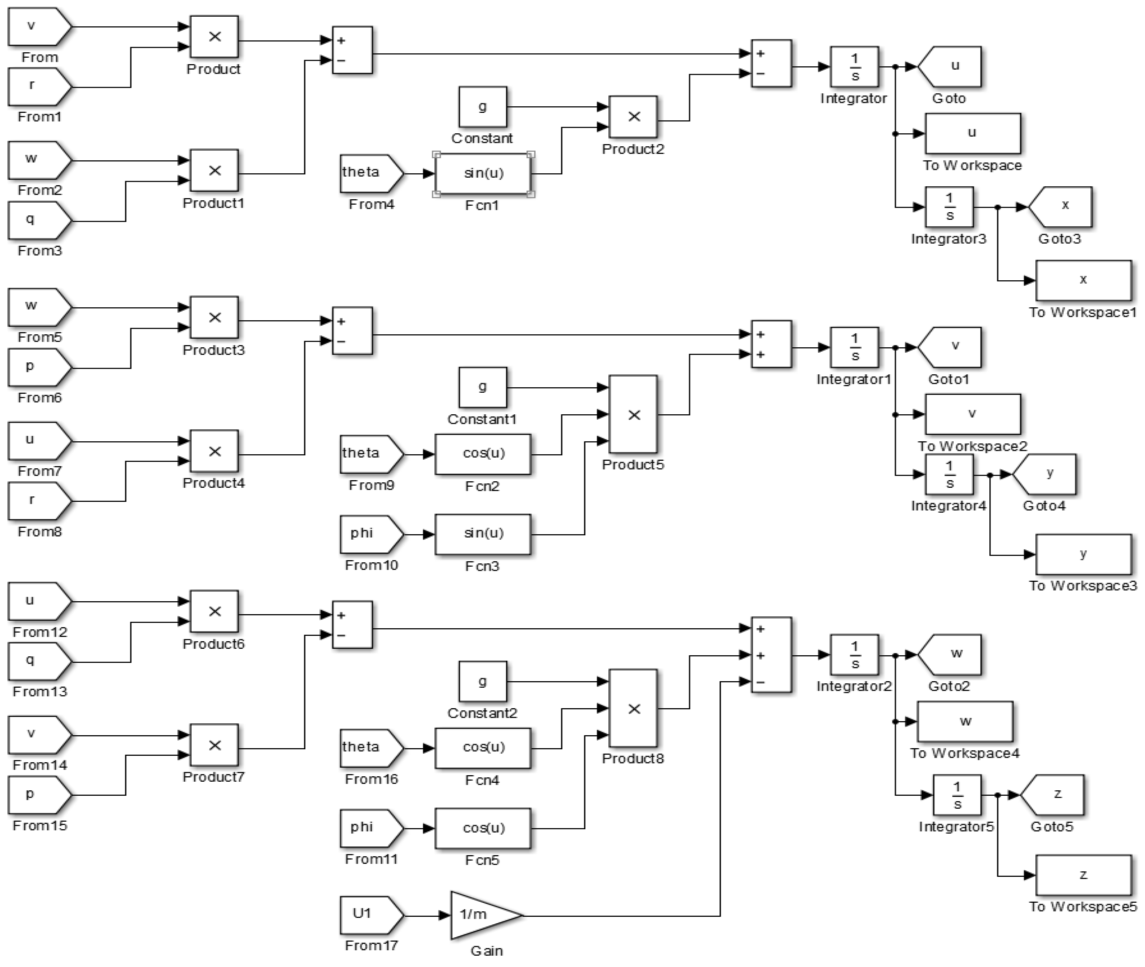

Figure 4-6 – Translation dynamics sub-block

4.7 Feedback Controller

The feedback controller sub-block consists of the cascaded loop control architecture discussed in Section 3.1. Here the pitch, and roll attitude and yaw rate inputs are used that run through two PID controllers. Referring to Figure 4-7, these inputs are Pitch_in, Roll_in, and Yaw_in. The outer loop minimizes the angle information error while the inner loop minimizes the rate information error. Yaw, however, consists only of one PID controller and minimizes only yaw rate error. The rate and attitude information generated by the dynamics sub-blocks feed into the controller along with the user inputs. In this case, the user inputs are the inputs recorded from flight data.

Along with the controllers was a saturation block, which limited the maximum control effort that the controller could provide. This allowed for simulation that was more accurate since it was not realistic for the system to provide and achieve the large control effort effectively. Table 4-2 shows the gains from each PID controller used in the Simulink model.

Table 4-2 – PID gains

PID Controller	Kp	Ki	Kd
Roll Angle	1.60	0.00	0.00
Roll Rate	0.12	0.11	0.01
Pitch Angle	1.60	0.00	0.00
Pitch Rate	0.12	0.11	0.01
Yaw Rate	3.00	1.50	0.03

The gains listed are used on the PID controller, which is implemented on the quadcopter using the Arduino microcontroller and the associated software. A series of three values, Kp, Ki, and Kd constitute a singular controller. In this case, there are five controllers. During flight, they are the PID controllers put into the Simulink file. It was important to match every element of the Simulink file to the flight regime. By implementing these values to the controllers built into the Simulink diagram, the Simulink model is properly capturing the quadcopter dynamics. As a result, the mathematical model better accounts for the flight data.

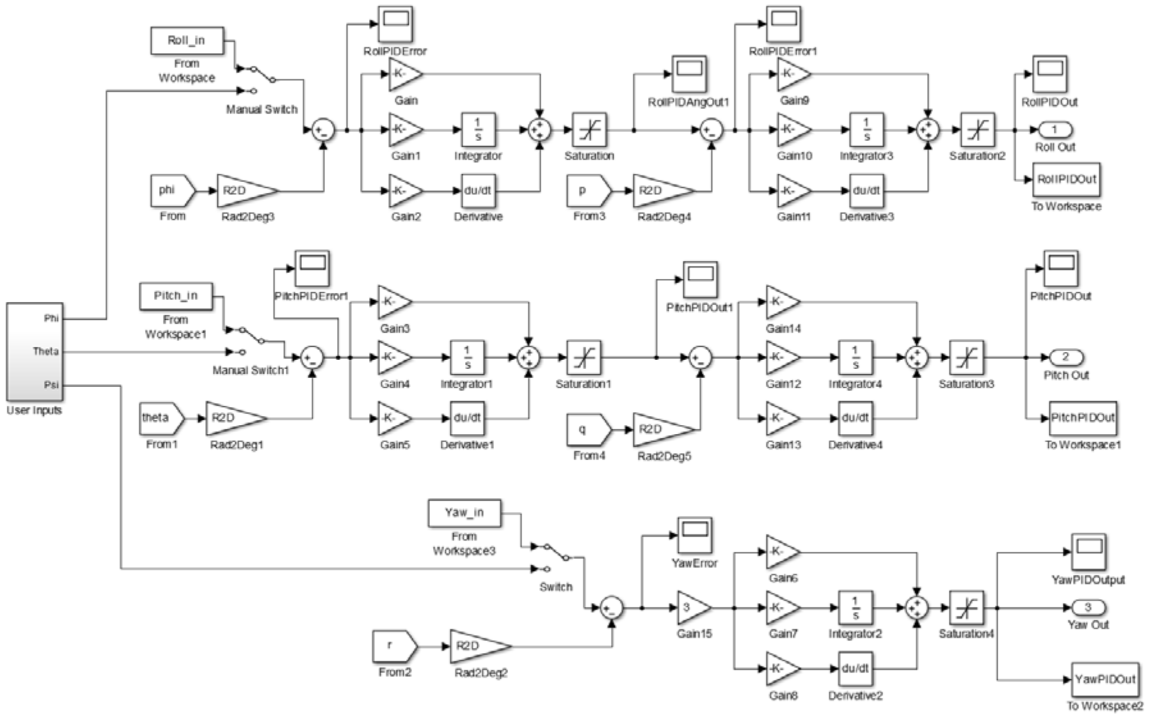

Figure 4-7 – Feedback control sub-block

5 Simulink Findings

5.1 Simulink Results

Figure 5-1 through Figure 5-5 show the results of the Simulink output and compare the results to the flight data. These data use the parameters listed in Section 2.6 and as can be seen, the simulation fits the flight data within reason. These results can be improved upon by manipulating the thrust and drag factors or the delay. After some manipulation of the these factors, the Simulink model achieved a better correlation to the flight data except when large yaw inputs were commanded. Whenever a yaw command was given, the roll rate and pitch rate violently oscilated causing more error in attitude output. The gains that would cause this oscillitory behavior are the drag factor and the motor delay. By carefully picking the right values for these gains, the Simulink model was more capable of following the measured output. Specific gains were chosen to avoid this; these gains are listed in Table 5-1.

Table 5-1 – Specific gain values

Parameter	Value	Unit
b (Thrust Factor)	$4.5*10^{-4}$	Nms^2
d (Prop. Drag Factor)	$0.5*10^{-5}$	Ns^2
c (Motor Delay)	$0.5*10^{-2}$	Seconds

Since the Simulink model does not take into account sensor noise and external disturbances despite the choice in gains, the Simulink output and the flight data do not perfectly match. The combination of these two factors really comes into play when dealing with angular rate comparisons. This was a result of the rate information runing at higher frequency ranges and therefore being more susceptible to noise and external disturbances like a gust of wind, for example. Overall, the results show that the mathematical model accurately captures the dynamics of the quadcopter.

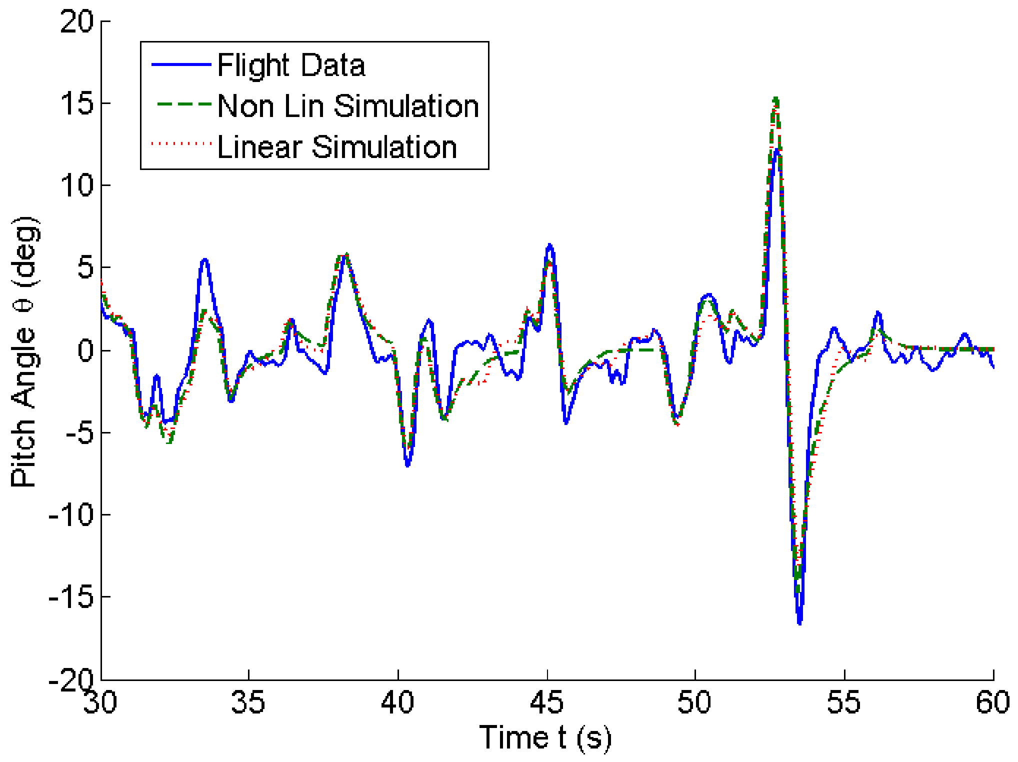

Figure 5-1 – Pitch angle comparison

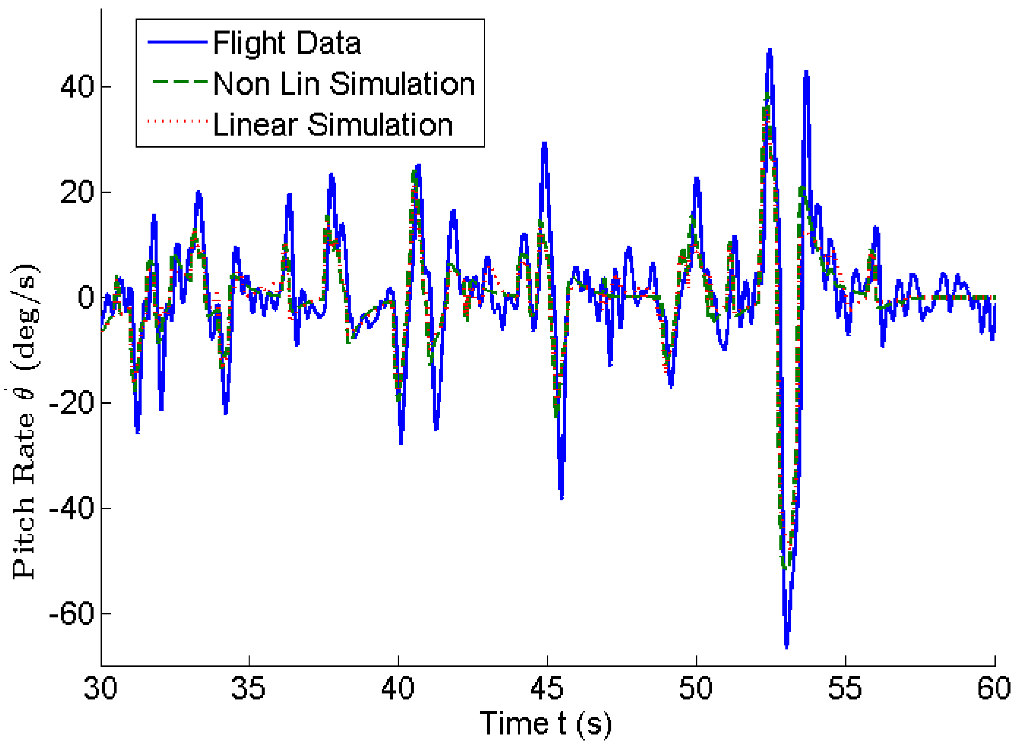

Figure 5-2 – Pitch rate comparison

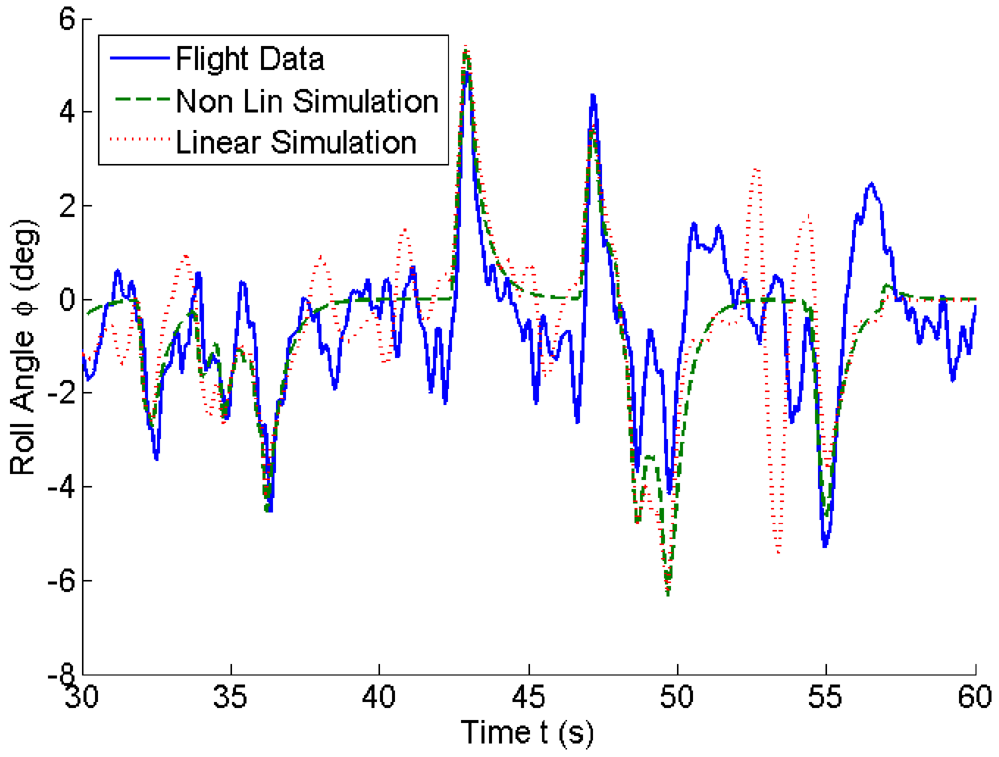

Figure 5-3 – Roll angle comparison

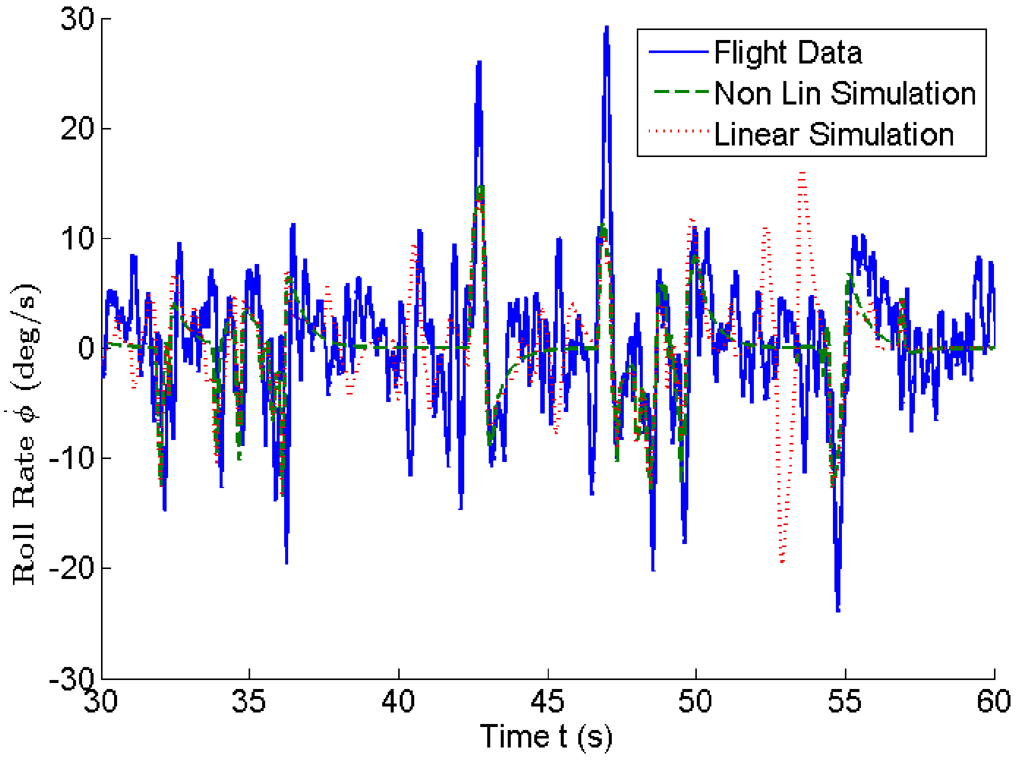

Figure 5-4 – Roll rate comparison

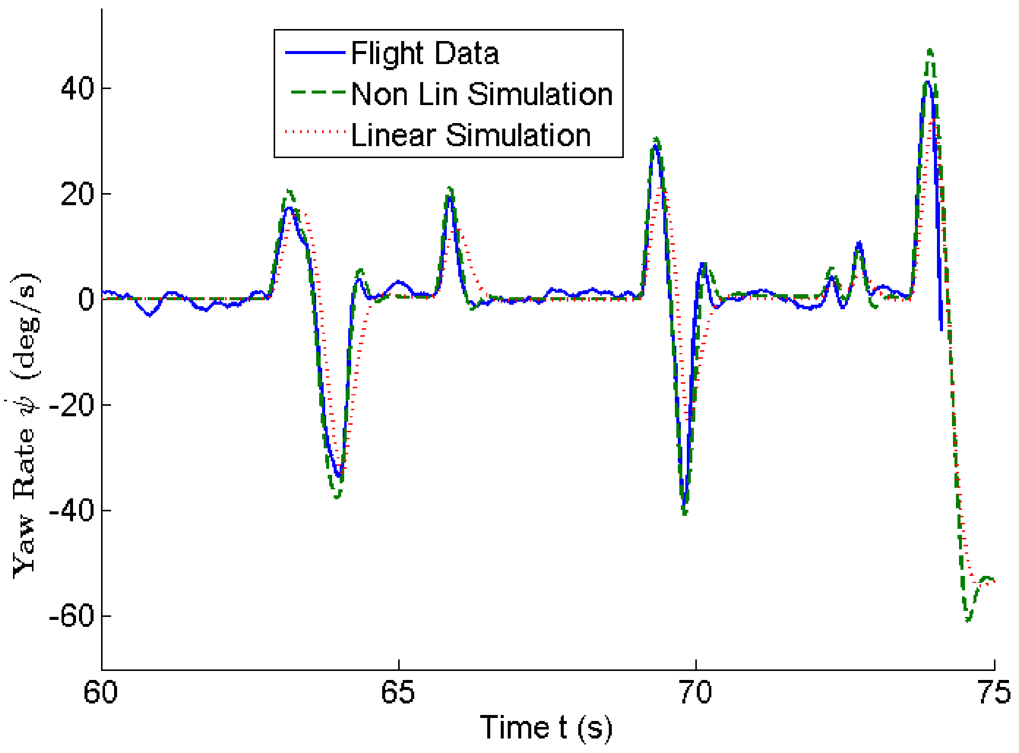

Figure 5-5 – Yaw rate comparison

5.2 Simulated Annealing Results

For the SA results shown below, the quenching factor was set to four and the maximum number of iterations was 1450. As can be seen from Figure 5-6 to Figure 5-10, the SA algorithm established PID controllers with faster rise times and much less steady state error. Overall, the SA-based PID tuning method has improved transient response characteristics.

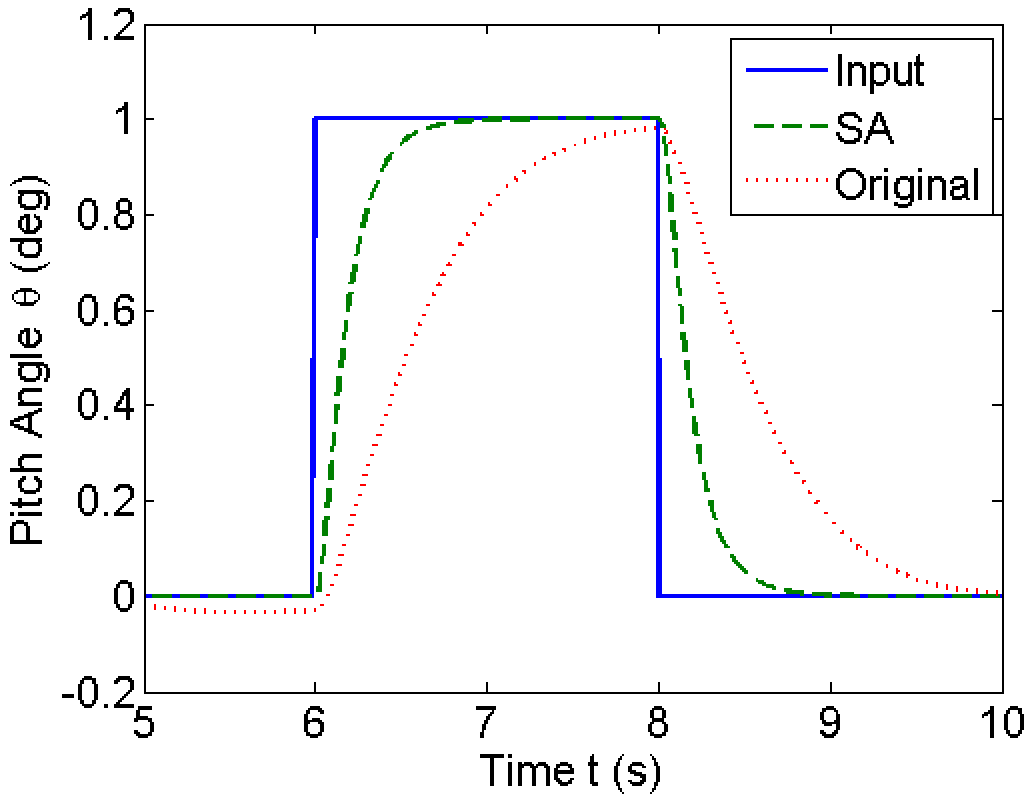

Figure 5-6 – Pitch angle PID comparison

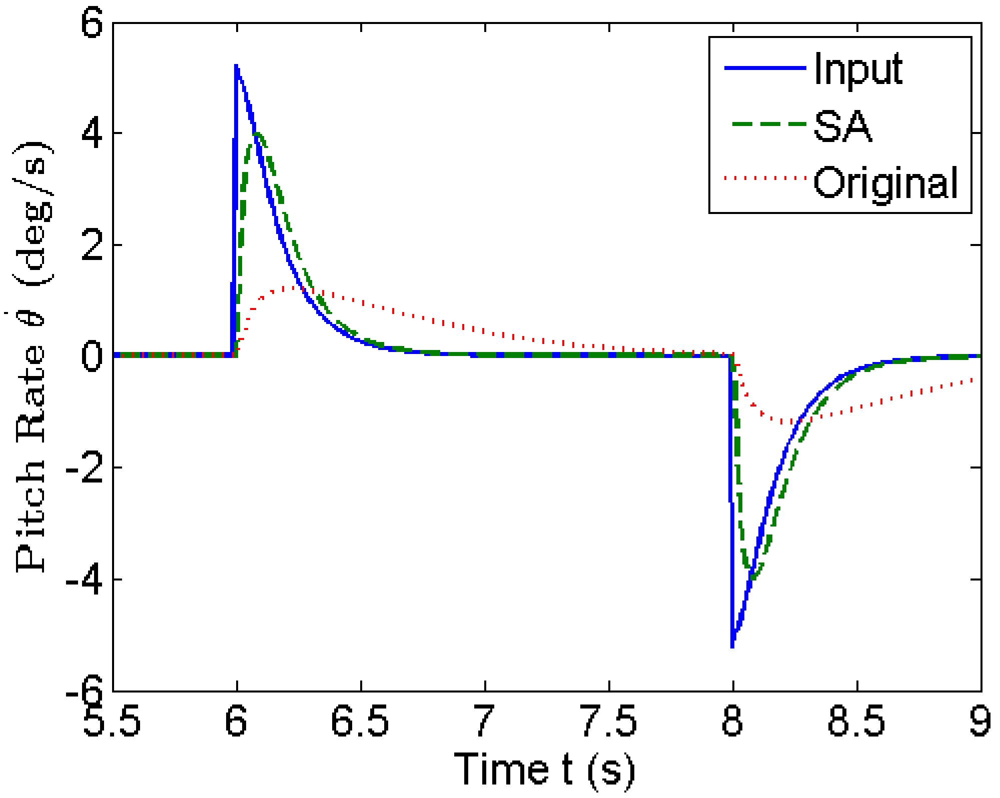

Figure 5-7 – Pitch rate PID comparison

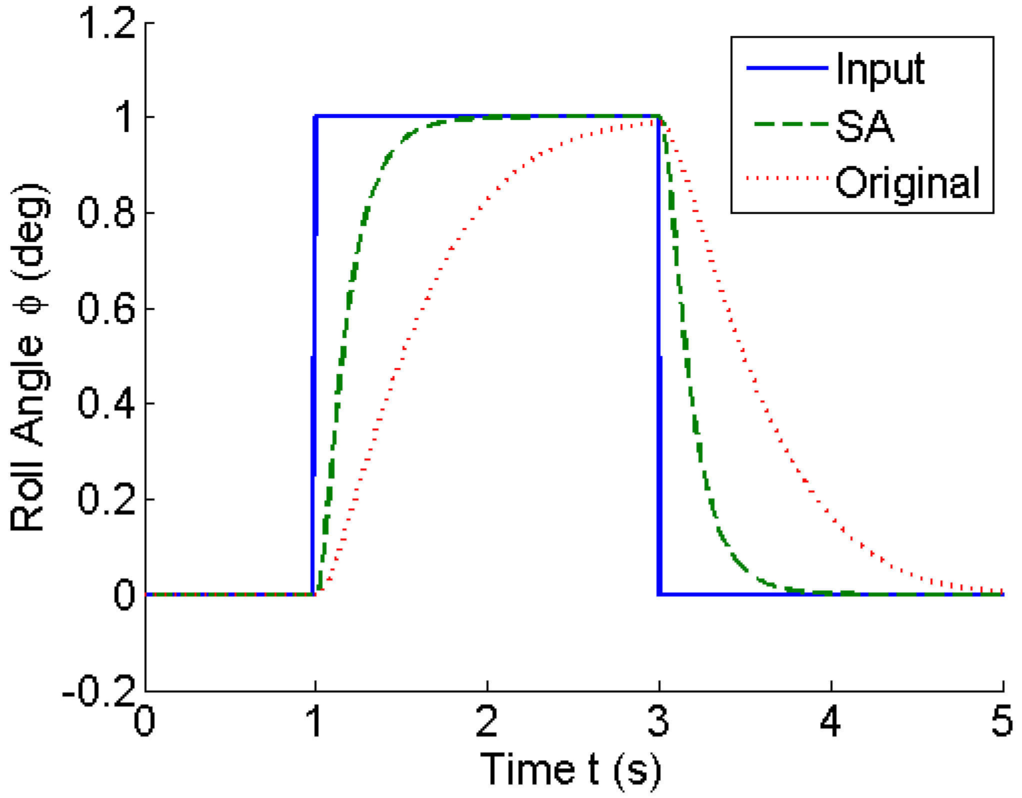

Figure 5-8 – Roll angle PID comparison

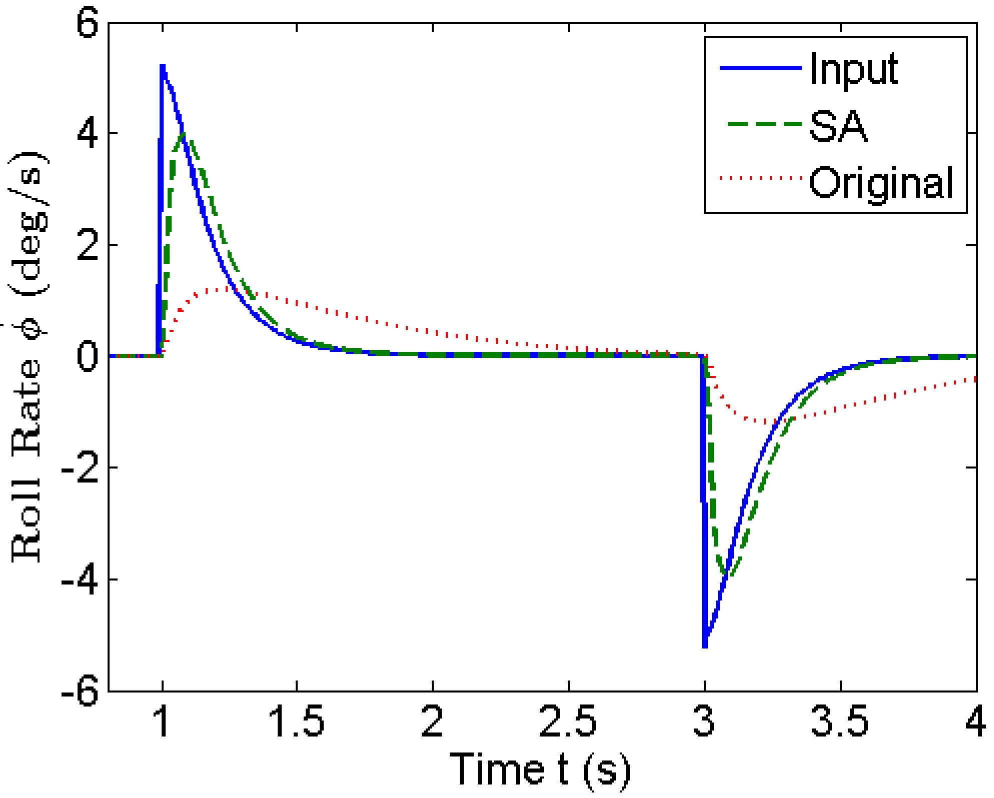

Figure 5-9 – Roll rate PID comparison

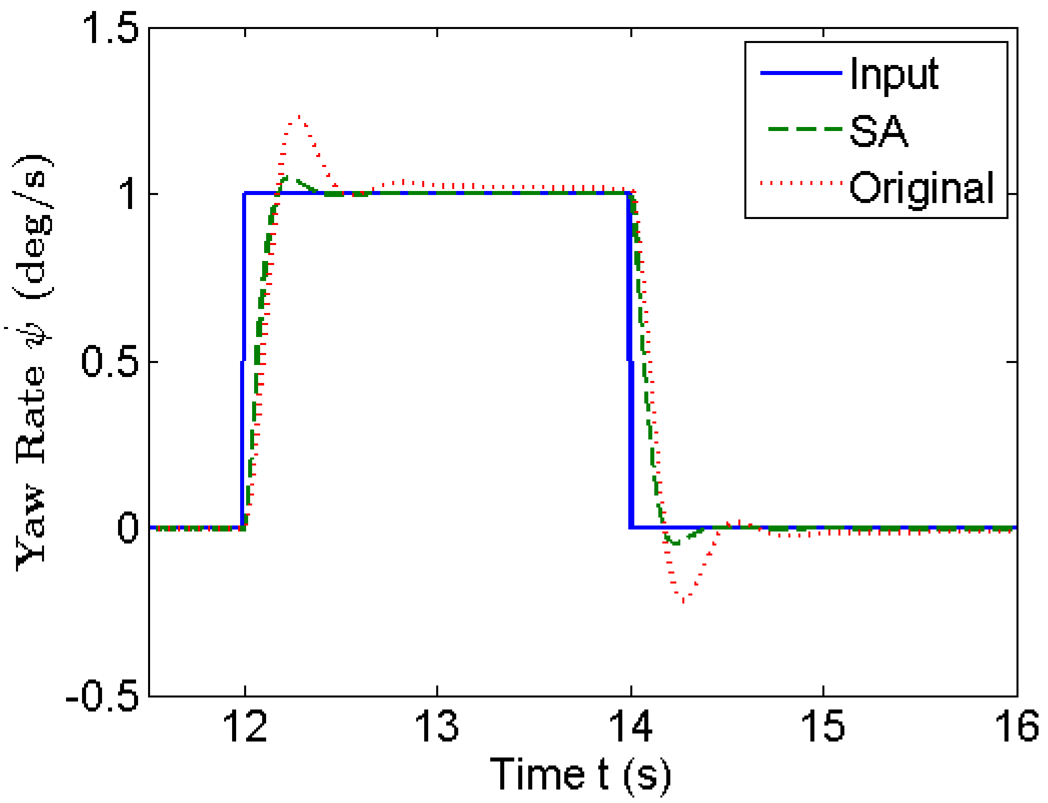

Figure 5-10 – Yaw rate PID comparison

Figure 5-11 to Figure 5-15 show the control effort that resulted from the user commands for both the SA and original PID controllers. These plots correspond to the outputs represented in Figure 5-6 to Figure 5-10. The results show that the control efforts from the SA-based PID controllers are within the available bandwidth. The saturation was ± 50° for the angle PID controllers and ± 50° per second for the rate PID controllers. The tuned SA controllers do not exceed these saturation limits.

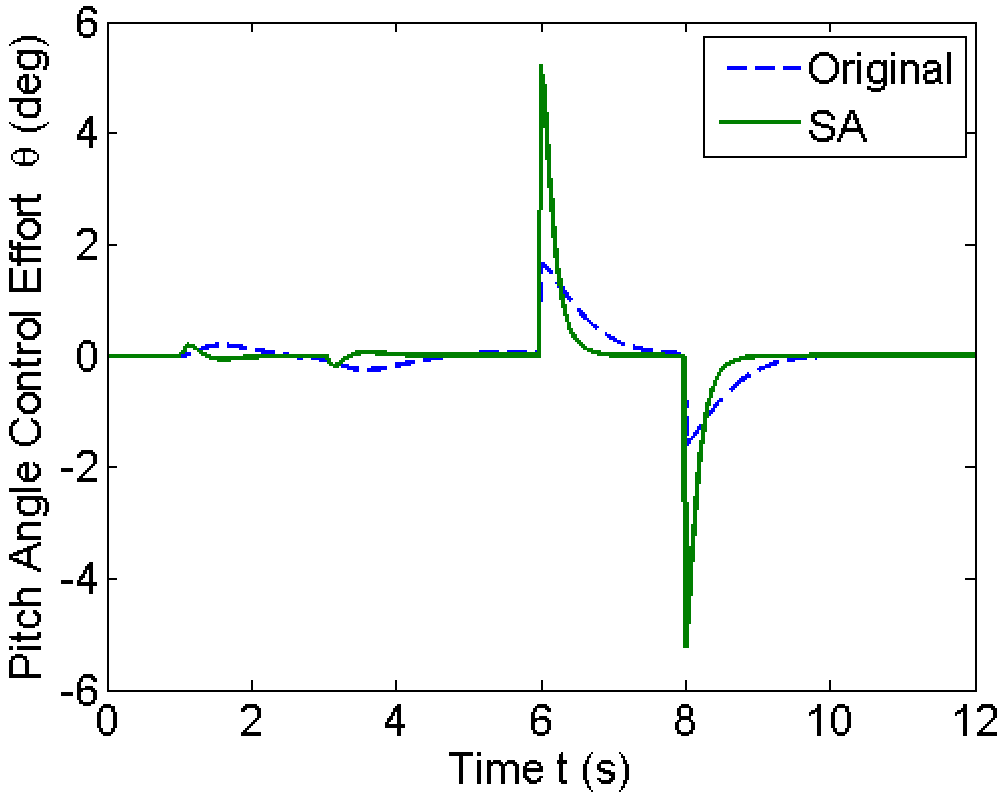

Figure 5-11 – Control effort comparison for pitch angle PID

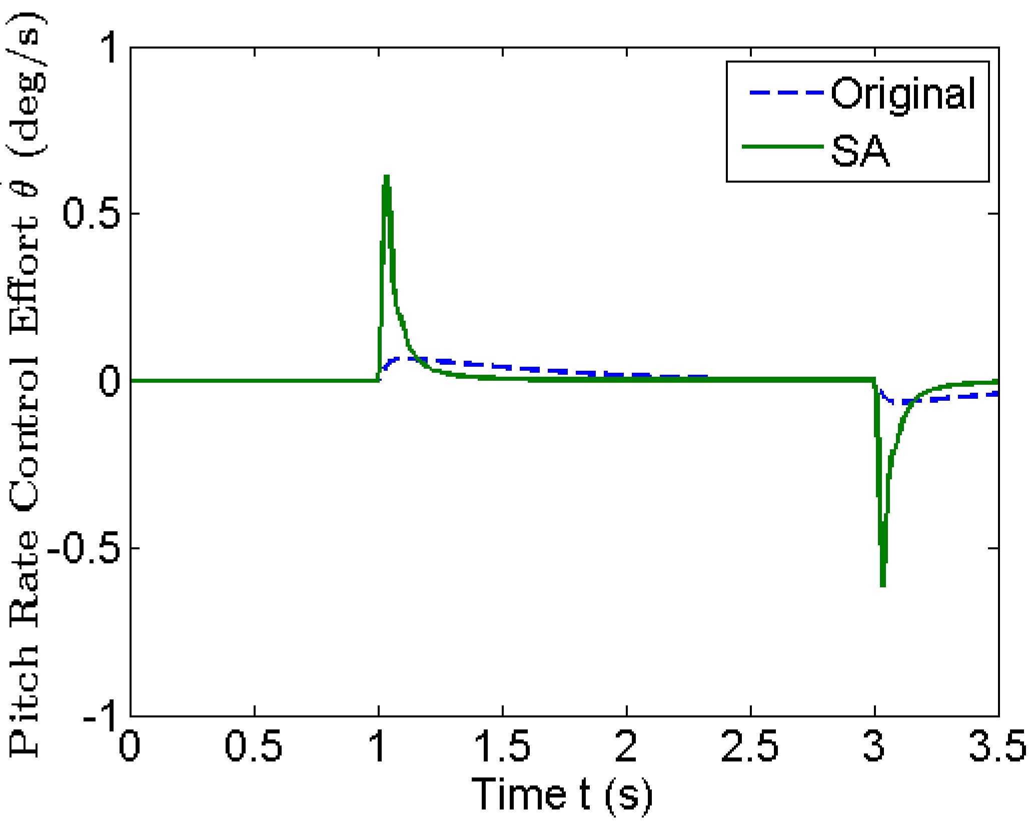

Figure 5-12 – Control effort comparison for pitch rate PID

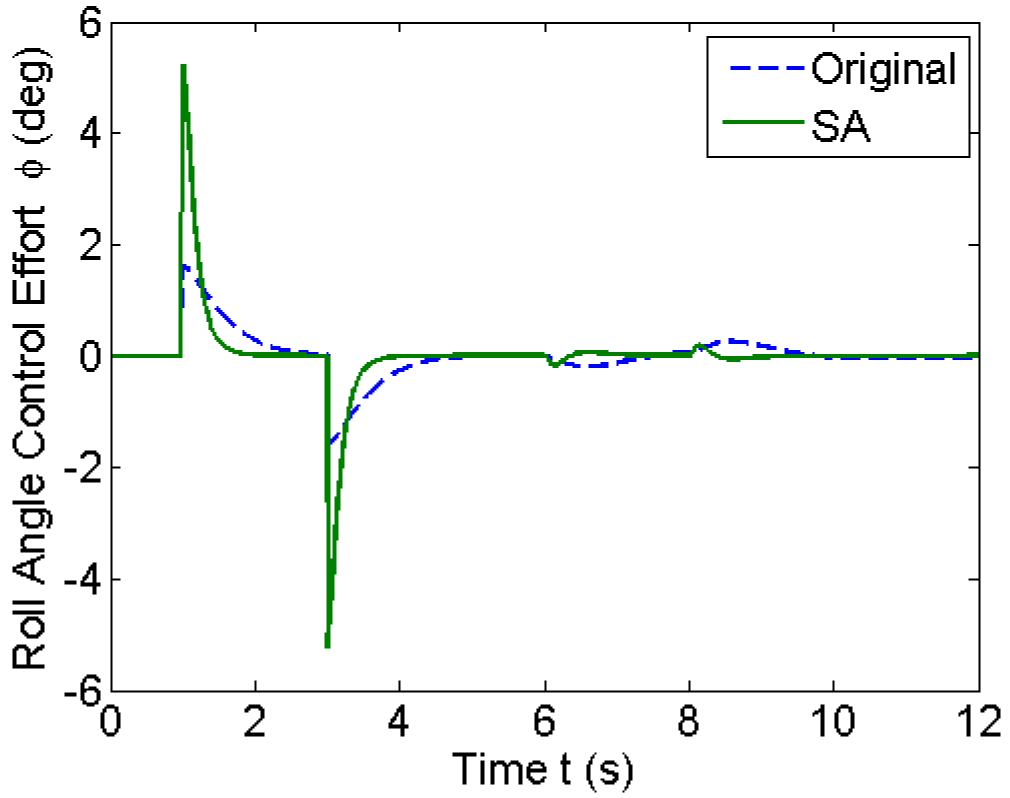

Figure 5-13 – Control effort comparison for roll angle PID

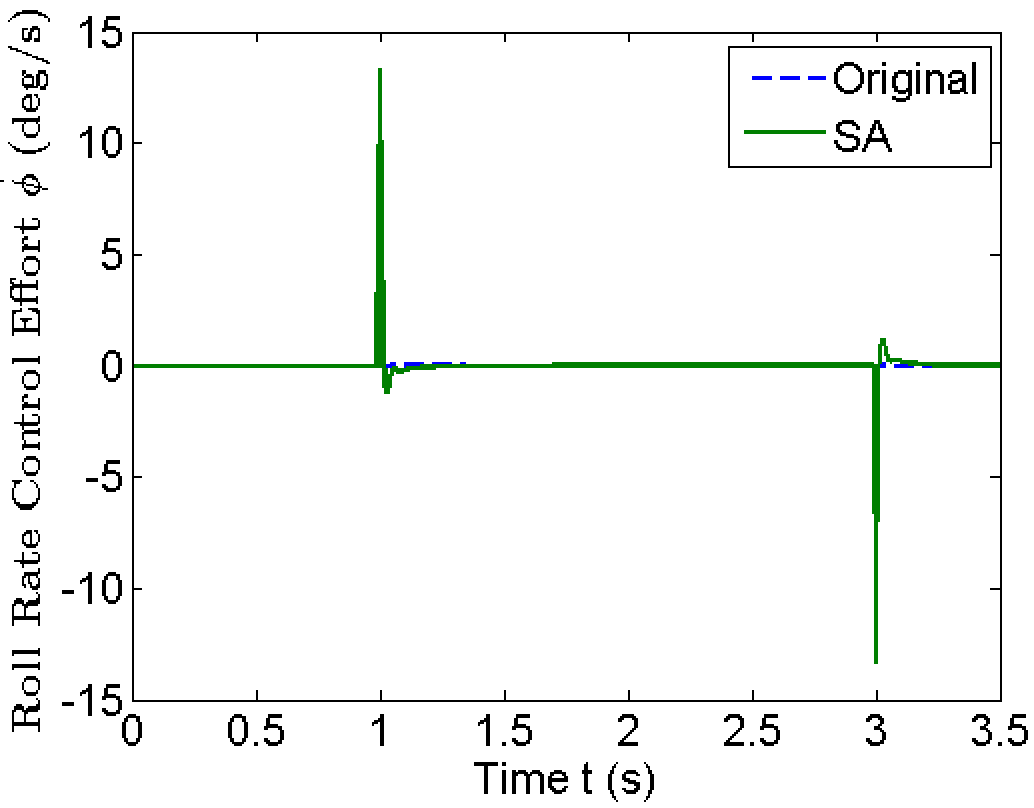

Figure 5-14 – Control effort comparison for roll rate PID

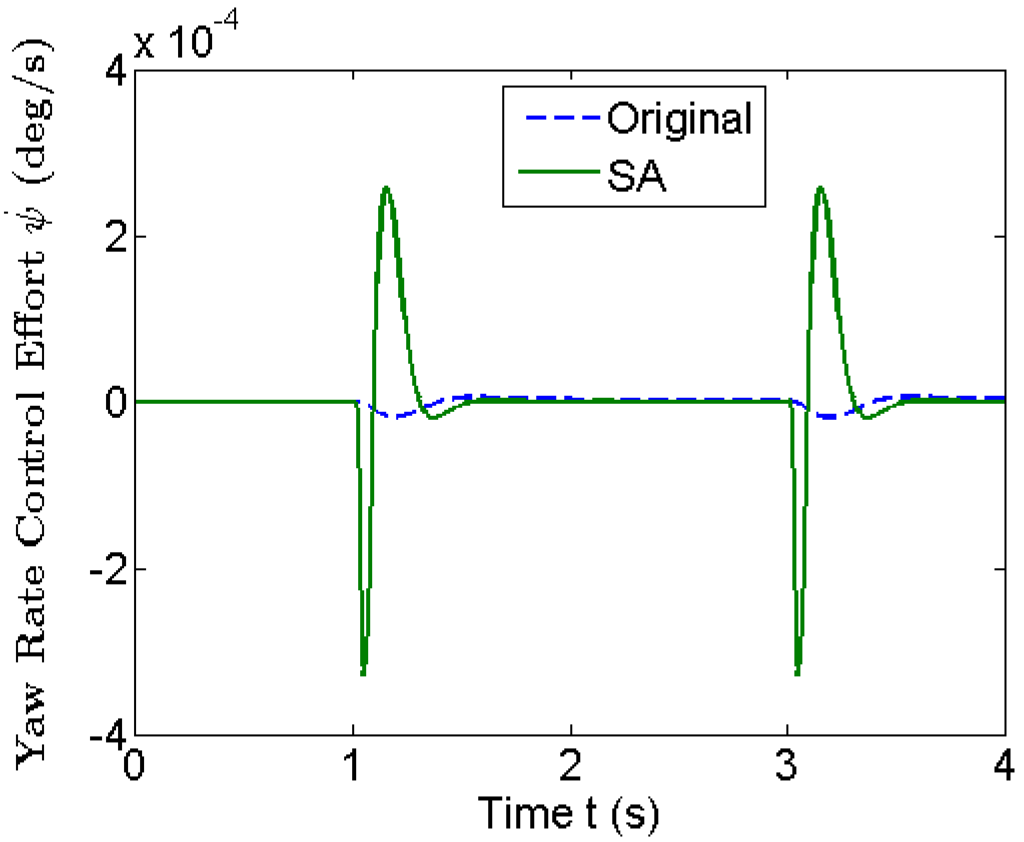

Figure 5-15 – Control effort comparison for yaw rate PID

Figure 5-16 through Figure 5-18 show the results of the simulated annealing algorithm. The image to the left of these figures represents how the output was tracking the input signal, and the image to the right shows cost plotted against each iteration. Although the output in Figure 5-17 has a lot of steady state error, it was necessary to allow for this error in order to prevent the controller from being saturated. The simulated annealing program finds global optimum solutions, which would cause it to drive up the gain values for the rate PID controller. This produced undesirable effects when applying the SA algorithm to the outer loop with large inner loop PID gains. It is found that limiting the upper bounds that the simulated algorithm searched through for the rate PID gains prevented the control effort from becoming saturated. In addition, pitch information was omitted in the case of the cost function since it represents the same dynamics as the roll output, and hence, the outputs are identical. This is represented in Figure 5-7 through Figure 5-9 where the output signals for the corresponding PID controllers are identical. Listed in Table 5-2 are the PID gain values generated by the SA optimization algorithm.

Table 5-2 – SA PID gains

PID Controller	Kp	Ki	Kd
Roll Angle	5.00000	0.03583	0.00230
Roll Rate	0.48216	2.00000	0.02071
Pitch Angle	5.00000	0.03583	0.00230
Pitch Rate	0.48216	2.00000	0.02071
Yaw Rate	5.00000	0.00500	0.21547

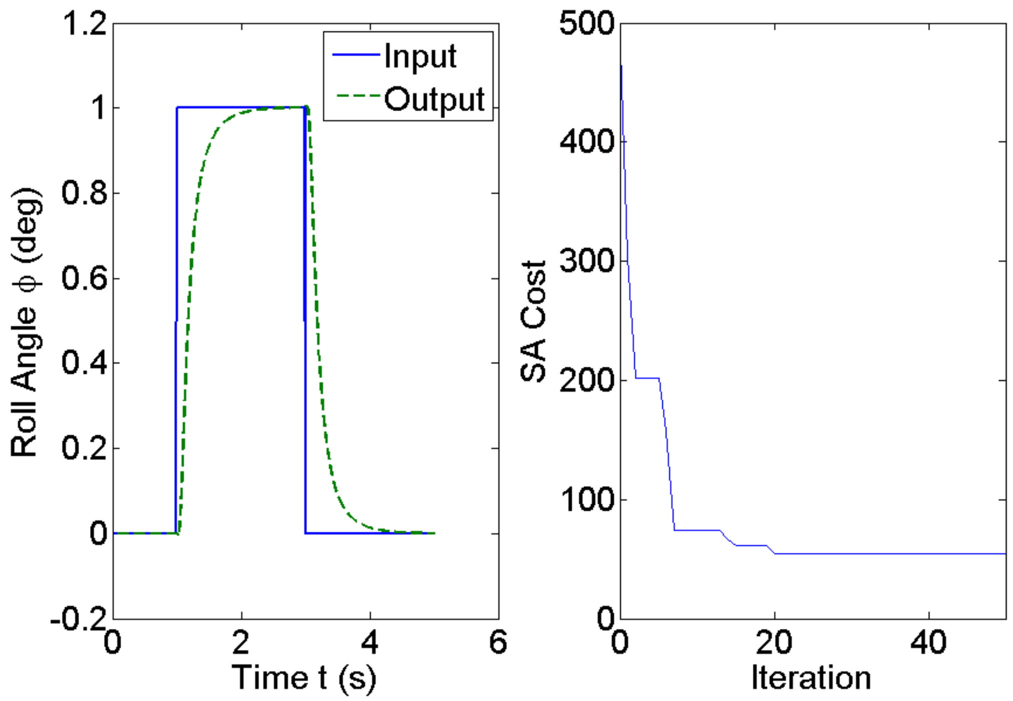

Figure 5-16 – Roll angle output and SA cost function

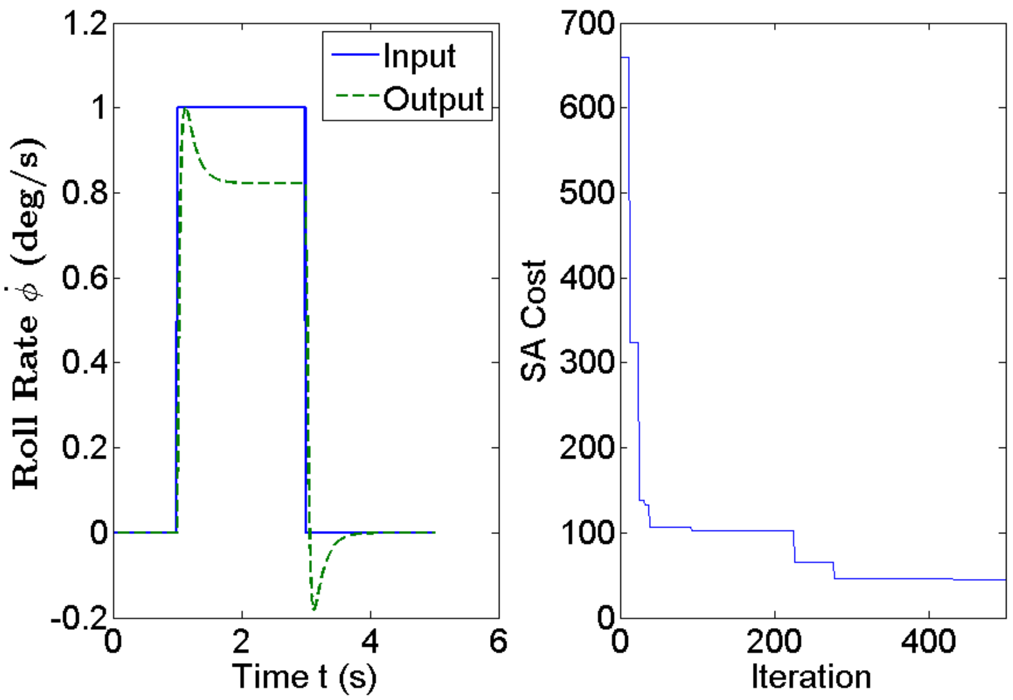

Figure 5-17 – Rate output and SA cost function

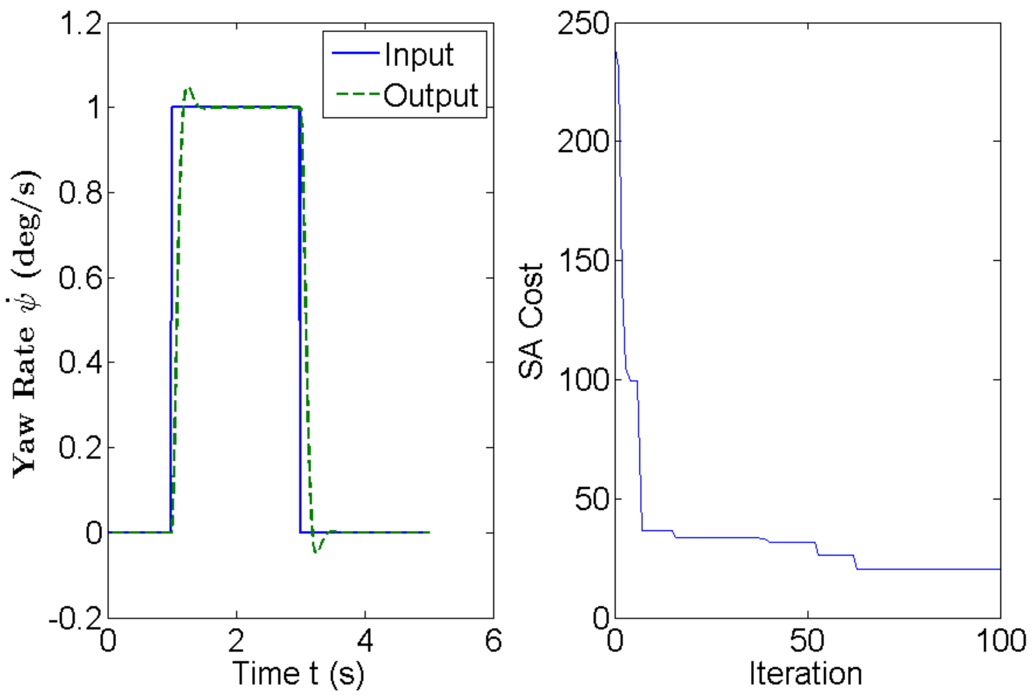

Figure 5-18 – Yaw rate output and SA cost function

6 Robustness Analysis

To analyze the new SA controllers, 1,000 Monte Carlo simulations tested the new PID controllers’ performance under a range of uncertainties. Here, the controllers were tested against an added 20% uncertainty to a parameter in the state matrix. Specifically, the gyroscopic effect was chosen as the basis to perform this analysis. Gyroscopic effects occur when the angular rates of the propellers do not sum to zero. This adds more nonlinearities to the dynamics that specifically occur when a system exhibits transient response. From this analysis, varying the gyroscopic effects affected pitch and roll outputs but had no effect on the yaw rate. This was because the gyroscopic effects will always sum to zero if the only command was yaw. Eqn. 6 illustrates the dynamics for yaw that are not influenced by gyroscopic effects. Listed in Figure 6-1 through Figure 6-3 are the results of adding this uncertainty to the gyroscopic term. Despite the uncertainties, both controllers are capable of maintaining stability. However, the SA-based PID controller outperformed the originally manually tuned PID controller. The SA-based PID controller better tracked the user input in spite of the uncertainties. Figure 6-4 shows the 20% uncertainty distribution.

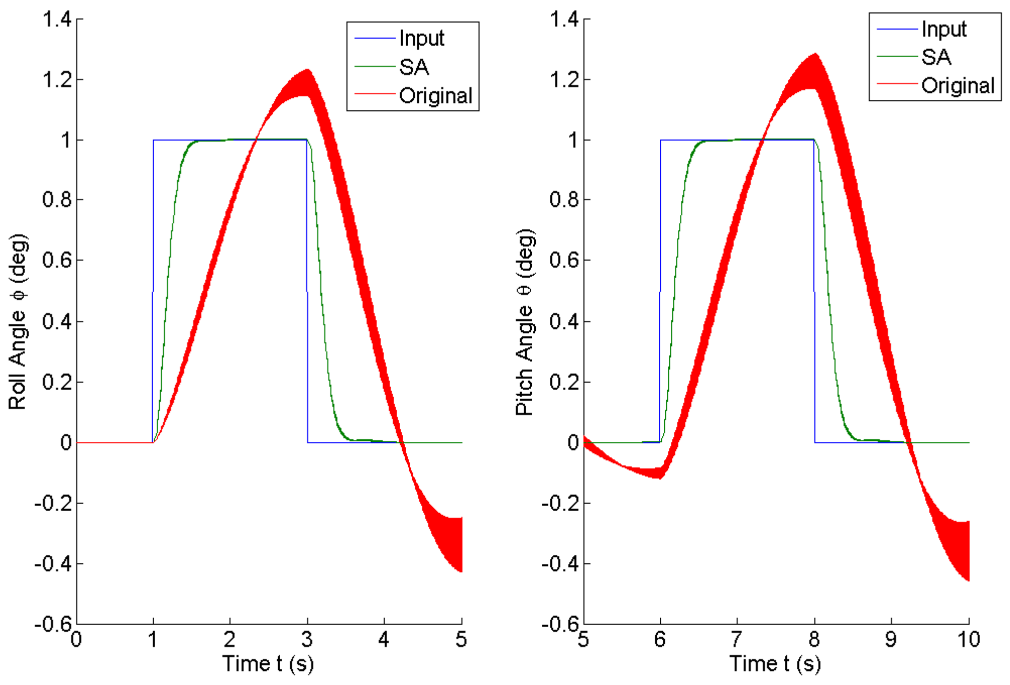

Figure 6-1 – Attitude uncertainty for roll and pitch

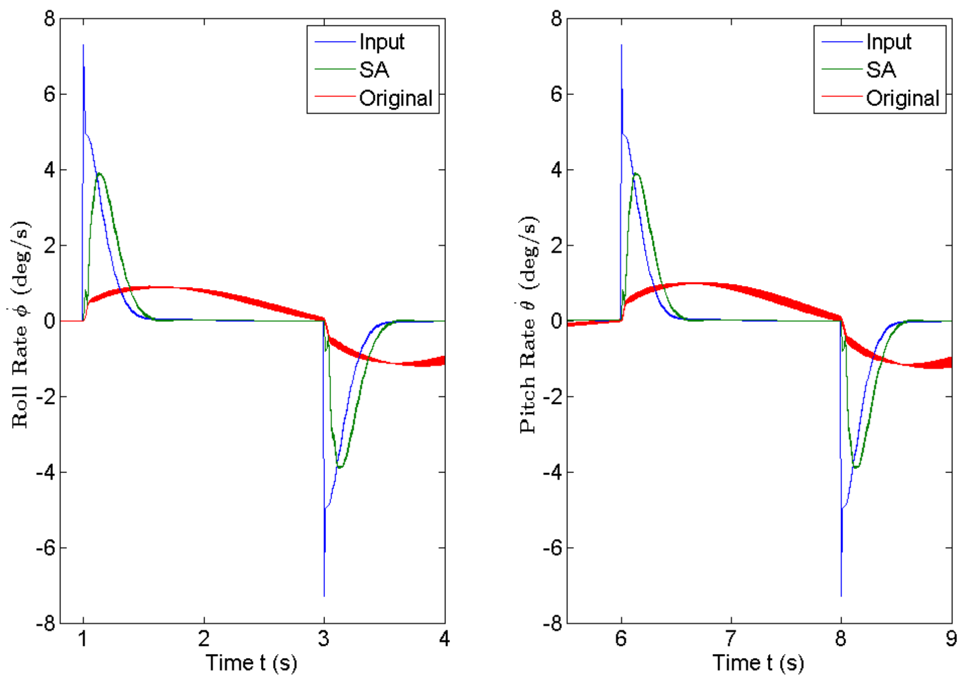

Figure 6-2 – Rate uncertainty for roll and pitch

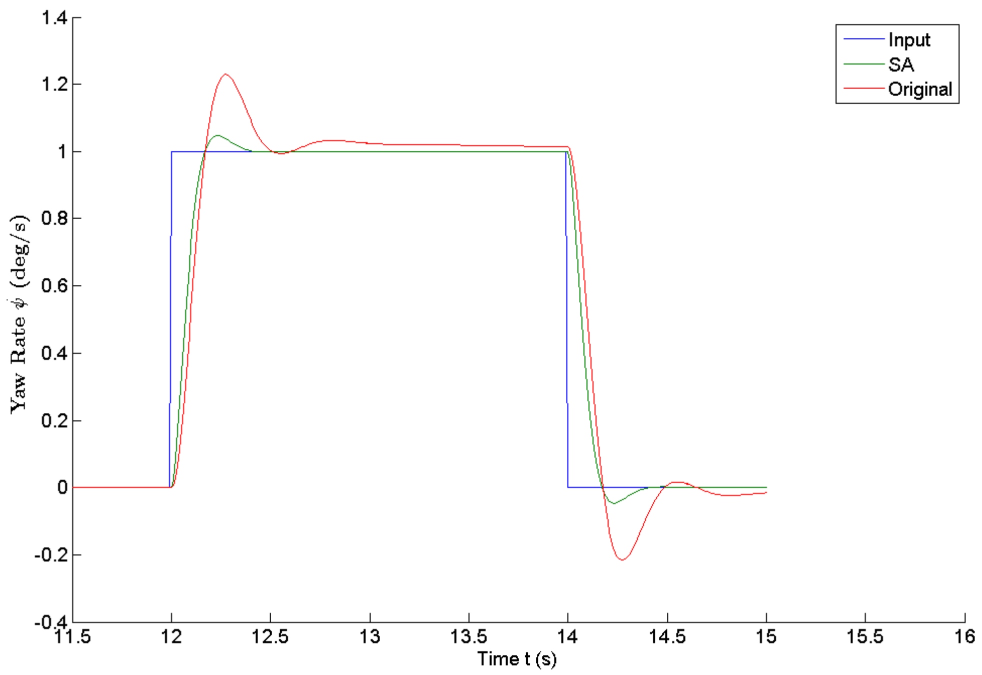

Figure 6-3 – Yaw rate uncertainty

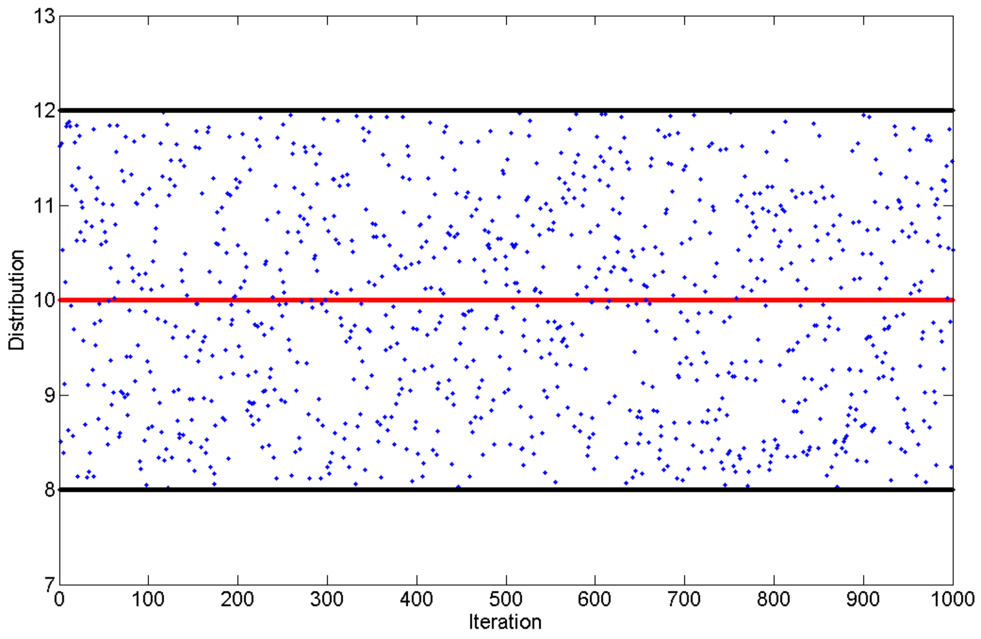

Figure 6-4 – 20% uncertainty distribution

7 Conclusion

In this study, a linear and nonlinear mathematical model was developed to analyze PID and SA-based PID controllers. This Simulink model was verified by comparing it to flight data, and validated results showed an exact match between data and simulation. The comparison showed that although the Simulink and the flight data outputs were not statistically correlated, the trends were correct. It was concluded that the Simulink model was satisfactory as a proper mathematical model of quadcopter dynamics. A simulated annealing algorithm was used to tune the PID controllers using the developed Simulink model. The new SA-based PID controllers were tested against the previous manually tuned PID controllers. The main goal was to eliminate steady state error, which was why the ITAE index was used. Based on the results, the application of the SA algorithm proved to be an excellent tuner for designing better performance PID controllers. Even when adding 20% uncertainty to the gyroscopic effects, the SA-based PID outperformed the original controller. Steady state error, rise time, settling time, and overshoot were improved.

8 Future Steps

One suggestion for future studies is to adapt the simulated algorithm to a multi-input-multi-output (MIMO) systems. Based on the literature in the field of dynamics and control, there has not been adaptation of the simulated annealing algorithm to a MIMO system like the quadcopter setup. This research could extend the knowledge and breadth that this paper provides, and may provide better solutions to tuning PID parameters. In this scenario, since all controllers are simultaneously tuned, the SA algorithm could find a global minimum energy state that best optimizes the system as a whole rather than in parts.

9 References

[1] T. Luukkonen, “Modelling and control of quadcopter,” School of Science, Aalto Univ., Espoo, Finland, Rep. Mat-2.4108, Aug. 22, 2011.

[2] I. Sa, and P. Corke, “Estimation and control for an open-source quadcopter,” in *Proceedings Australasian Conference Robotics and Automation, ACRA 2011*, Monash University, Melbourne, Australia, pp. 1-7.

[3] *History of Quadcopters and Multirotors* [Online]. Available: http://www.krossblade.com/history-of-quadcopters-and-multirotors/. [Accessed 6 Apr. 2017].

[4] *World’s first helicopter - Today in history: September 14* [Online]. Available: https://connecticuthistory.org/worlds-first-helicopter-today-in-history/. [Accessed 6 Apr. 2017].

[5] N. Hovakimyan, (2012, May 3). *Unsubstantiated and wrong claims about L_1 adaptive control advanced controls research laboratory* [Online]. Available: http://naira.mechse.illinois.edu/clarifications-on-l1-adaptive-control/. [Accessed 6 Apr. 2017].

[6] N. Hovakimyan, and C. Cao, “Introduction” in *L_1 Adaptive Control Theory*. Philadelphia, PA, SIAM, 2010, ch. 1, sec. 1.1, pp. 1-4.

[7] N. Hovakimyan, *Theory advanced controls research laboratory* [Online]. Available: http://naira.mechse.illinois.edu/theory/. [Accessed 6 Apr. 2017].

[8] C. Cao, and N. Hovakimyan, “Design and analysis of a novel L_1 adaptive controller, part I: Control signal and asymptotic stability,” in *American Control Conference,* Minneapolis, MN, Jun. 14-16, 2006, pp. 3397-3402.

[9] I. M. Gregory et al., “L_1 adaptive control design for NASA airstar flight test vehicle,” *AIAA Journal of Guidance, Control, and Dynamics,* pp. 1-5, Aug. 2009.

[10] J. Wang et al., “Novel L_1 adaptive control methodology for aerial refueling with guaranteed transient performance,” *AIAA Journal of Guidance, Control, and Dynamics,* vol. 31, pp. 182-193, Jan. 2008.

[11] N. Abbas, A. Legowo, and R. Akmeliawati, “Parameter identification of an autonomous quadrotor,” in *4th International Conference,* Kuala Lumpur, Malaysia, 2011, pp. 1-8.

[12] D. Saakes et al., ”A teleoperating interface for ground vehicles using autonomous flying cameras,” in *23rd International Conference,* Tokyo, Japan, 2013. doi: 10.1109/ICAT.2013.6728900.

[13] M. B. Hurd, “Control of a quadcopter aerial robot using optic flow sensing,” M.S. thesis, Mech. Eng. Dept., Univ. of Reno, Reno, NV, 2013.

[14] S. Siebert, and J. Teizer, “Mobile 3D mapping for surveying earthwork projects using an unmanned aerial vehicle (UAV) system,” *Automation in Construction* vol. 41, pp. 1-14, May 2014.

[15] A. F. Sorensen, ”Autonomous control of a miniature quadrotor following fast trajectories,” M.S. thesis, Control Eng. Dept., Aalborg Univ., Aalbord, Denmark, 2010.

[16] M. Kishnani et al., “Optimal tuning of DC motor via simulated annealing,” In *Advances in Engineering and Technology Research, ICAETR 2014,* 2014 © IEEE. doi: 10.1109/ICAETR.2014.7012928.

[17] N. Metropolis et al., "Equation of state calculations by fast computing machines," *Journal of Chemical Physics*, vol. 21, pp. 1087-1092, Mar. 1953.

[18] R.W. Eglese, "Simulated annealing: A tool for operational research," *European Journal of Operational Research*, vol. 46, pp. 271-281, Jun. 1990.

[19] D. Bertsimas and J. Tsitsiklis, "Simulated annealing," *Statistical Sci.*, vol. 8, pp. 10-15, Feb. 1993.

[20] A. Ji, “Development of a low-cost experimental quadcopter testbed using an Arduino controller for video surveillance,” M.S. thesis, Dept. of Aerospace Eng., San Jose State Univ., San Jose, CA, 2015.

[21] T. Bresciani, “Modeling, identification and control of a quadrotor helicopter,” M.S. thesis, Dept. of Automatic Control, Lund Univ., Lund, Sweden, 2008.

[22] A. Kotikalpudi, et al., “Swing tests for estimation of moments of inertia,” unpublished.

[23] Yachen, Z., and H. Yueming. "On PID controllers based on simulated annealing algorithm,” in *27th Chinese Control Conference*. Kunming, China, 2008, pp. 225-228.

[24] W. Yang, W. Cao, and T. Chŏng. “Applied numerical methods using matlab,” 1st ed. Hoboken, NJ: Wiley, 2005.

[25] CIFER, student version, available at http://uarc.ucsc.edu/flight-control/cifer/. [Accessed 6 Jul. 2017].

[26] MATLAB 8.0 and Simulink Toolbox 8.1, The MathWorks, Inc. Natick, MA.

[27] J. G. Ziegler and N. B. Nichols. “Optimum settings for automatic controllers,” *The American society of mechanical engineers*, vol. *64*, pp. 759-768, Dec. 1941.

[28] K. Ogata, “Introduction to control systems,” in *Modern Control Engineering*, 3rd ed. Upper Saddle River, NJ: Prentice Hall, 1997, ch. 1, sec. 1-3, pp. 6-7.

[29] S. M. Shinners, “Performance Criteria,” in *Modern Control System Theory and Design*, 2nd ed., Hoboken, NJ: Wiley, 1998, ch. 5, sec 5-7, pp. 290-292.