Summer 2023

Modeling, Control, and Hardware Development of a Thrust-Vector Coaxial UAV

Andrew North
Embry-Riddle Aeronautical University, northa@my.erau.edu

MODELING, CONTROL, AND HARDWARE DEVELOPMENT OF A THRUST-VECTOR COAXIAL UAV

By Andrew W. North

A Thesis Submitted to the College of Engineering Department of Mechanical Engineering in Partial Fulfillment of the Requirements for the Degree of Master of Science in Mechanical Engineering

Embry-Riddle Aeronautical University
Daytona Beach, Florida
August 2023

MODELING, CONTROL, AND HARDWARE DEVELOPMENT OF A THRUST-VECTOR COAXIAL UAV

By Andrew W. North

This thesis was prepared under the direction of the candidate's Thesis Committee Chair, Dr. Christopher Hockley, Assistant Professor, Daytona Beach Campus, and Thesis Committee Members Dr. Brian Butka, Professor, Daytona Beach Campus, and Dr. Monica Garcia, Assistant Professor, Daytona Beach Campus, and has been approved by the Thesis Committee. It was submitted to the Department of Mechanical Engineering in partial fulfillment of the requirements for the degree of Master of Science in Mechanical Engineering

Thesis Review Committee:

_____
Christopher Hockley, Ph.D.
Committee Chair

_____ _____
Brian Butka, Ph.D.
Commette Member

Monica Garcia, Ph.D.
Committee Member

_____ _____
Jean-Michel Dhainaut, Ph.D.
Graduate Program Coordinator,
Mechanical Engineering

Patrick Currier, Ph.D.
Department Chair,
Mechanical Engineering

_____ _____
James Gregory, Ph.D.
Dean, College of Engineering

Christopher Grant, Ph.D.
Associate Vice President of Academics

_____
Date

Acknowledgements

I would like to thank my family and friends for their encouragement during my academic endeavors. Their continuous support has helped me tremendously.

I would also like to thank my committee members, Dr. Butka and Dr. Garcia, for their valuable feedback and counsel. Finally, I would like to thank my thesis chair, Dr. Hockley, for his technical advice, interesting ideas, and guidance during this research which has fostered my passion for robotics and controls engineering.

Abstract

Researcher: Andrew W. North
Title: Modeling, Control, and Hardware Development of a Thrust-Vector Coaxial UAV

Institution: Embry-Riddle Aeronautical University
Degree: Master of Science in Mechanical Engineering
Year: 2023

This thesis introduces a unique thrust vector coaxial unmanned aerial vehicle (UAV) configuration and presents a comprehensive investigation encompassing dynamics modeling, hardware design, and controller development. Using the Newton-Euler method, a dynamic model for the UAV is derived to gain in-depth insights into its fundamental flight characteristics. A simple thrust model is formulated and modified by comparing it with data obtained from vehicle testing. The feasibility of manufacturing such a vehicle is assessed through the development of a hardware prototype. Finally, a linear state feedback controller is designed and evaluated using the non-linear dynamics model. The results demonstrate successful validation of the hardware through flight tests. The initial thrust model is enhanced by two methods, incorporating correction factors derived from a regression line, and employing the system identification method based on the test stand data. Implementation of the linear state feedback controller effectively maintains attitude authority over a non-linear simulation of the vehicle. The limits of the controller are explored, and simulation highlights that the controller's authority fails if the operating states deviate from the linearized region of attraction. Beyond the specific thrust vector coaxial UAV configuration, this research holds implications for enhancing UAV dynamics modeling, analysis, and control in broader applications.

Table of Contents

Page

List of Tables

Page Table

List of Figures

Page Figure

Nomenclature and Acronyms

ARMAX	Autoregressive-Moving Average with Exogenous Terms
CAD	Computer-Aided Design
DC	Direct Current
DOF	Degrees of Freedom
EOM	Equations of Motion
ESC	Electronic Speed Controller
GPS	Global Positioning System
IR	Infrared
IMU	Inertial Measurement Unit
Kv	Motor Velocity Constant
LiPo	Lithium-Polymer
MIMO	Multiple-Input Multiple-Output
mAh	Milliamp Hour
PID	Proportional Integral Derivative
PLA	Polylactic Acid
PV	Process Variable
RC	Radio Control
RPM	Revolutions Per Minute
SISO	Single-Input Single-Output
SP	Setpoint
UAV	Unmanned Aerial Vehicle
UI	User Interface
USB	Universal Serial Bus
VTOL	Vertical Take-Off and Landing

Symbols

a	Acceleration $[m/s^2]$
F	Force $[Newtons]$
g	Acceleration of Gravity $[m/s^2]$
L	Lagrangian $[J]$
m	Mass $[kg]$
P	Pressure $[Pa]$
T_l	Kinetic Energy $[J]$
τ	Torque $[N-m]$
V	Velocity $[m/s]$
V_l	Potential Energy $[J]$
ρ	Density $[kg/m^3]$
ω	Angular Speed $[rpm]$

1 Introduction

1.1 Thesis Statement

The purpose of this thesis is to demonstrate the ability of a full-state feedback controller to reject disturbances on a coaxial thrust vectored unmanned aerial vehicle (UAV).

1.2 Background

Interest in UAVs has been steadily increasing in recent years due to their expanding capabilities and advances in electronics [1]. To accomplish a variety of tasks, different UAV configurations have emerged. For instance, using fixed wing UAVs flying at low altitudes and at high speeds, large scale aerial photography can be conducted rapidly and at low cost. Fixed wing aerial photography in the visible and infrared (IR) spectrum can be used for surveying hard to reach areas, non-destructive vegetation imaging, flood and forest fire tracking, and crop field monitoring [2]. Lightweight UAVs such as balloons and kite planes are commonly used in meteorological and atmospheric research [3] [4]. Vertical Take-Off and Landing (VTOL) and rotorcraft UAVs are more maneuverable and prove useful for inspecting and maintaining power lines and pipelines while keeping surveyors away from dangerous environments [5] [6]. The military has replaced many traditional piloted vehicles with UAVs due to their lower costs and complete risk reduction to pilots [7]. Most hobby and commercial UAVs today take fixed wing, coaxial helicopter, or quadcopter configurations. Despite the impressive capabilities of these UAVs, other configurations show promise of improved performance in specialized applications [8]. The benefits of different configurations are exemplified by the variety of piloted rotorcraft, including the coaxial Kamov, tandem counter-rotating Chinook, and intermeshing-rotor Kaman.

Figure 1.1: Coaxial Kamov [9], Counter-Rotating Chinook [10], and Intermeshing-Rotor Kaman [11] Helicopters

One unique configuration is that of the thrust-vector coaxial UAV, which incorporates the stability of the coaxial form without the mechanical complexity of a swashplate [12] [13] [14]. The coaxial dual rotor design offers several advantages over the traditional single main rotor design of a helicopter. Firstly, it provides greater lift while reducing the rotor diameter. Additionally, the coaxial configuration effectively cancels out the net gyroscopic forces generated by the spinning blades [15]. This cancellation and the balanced torques result in improved stability and reduced sensitivity to disturbances [12] [13] [16]. Figure 1.2 shows the forces and torques associated with a coaxial UAV and a typical helicopter.

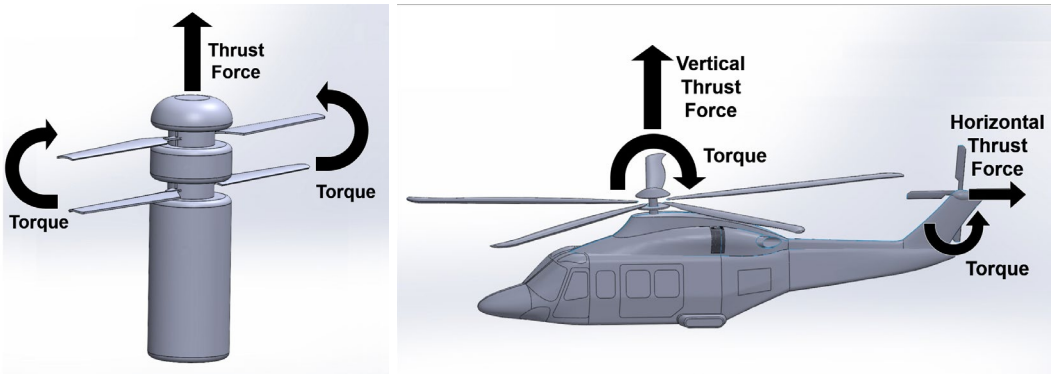


Figure 1.2: Force and Torque Diagram for a Coaxial UAV vs Helicopter

The coaxial vehicle offers several advantages over quadcopters as well, one of which is its smaller and more compact form factor [17]. This is a result of the two rotors stacked in line with the rest of the body, rather than being spread out horizontally. This compact form factor has multiple benefits including a more robust frame, easier to

transport, decreased aerodynamic drag and increased efficiency [17] [18]. A quadcopters components are splayed out between the 4 rotors, and the direction of force is always perpendicular to that area. The coaxial configuration reduces aerodynamic drag by presenting a smaller surface area in the direction of motion. Yet another interesting benefit of the compact cylindrical form is the potential to be launched from a tube cannon. This capability could send the vehicle a long distance before it starts using its internal energy, thus increasing the mission range [8].

This study is focused on a specific configuration of thrust vector coaxial UAV referred to as the CoaxUAV in this paper. The CoaxUAV uses two planar mounted motors, each with a pinion gear driving contra-rotating ring gears attached to each of the two rotors. This configuration increases output torque through motor gearing and provides a hollow center shaft through which power and communication can be run through the propulsion section. This has the advantage of allowing payloads to be mounted above or below the propulsion section, useful for global positioning system (GPS), cameras, or other sensors. This thesis develops a theoretical dynamic and thrust model for the CoaxUAV from first principals. A hardware prototype is designed, manufactured, and tested to improve upon the theoretical thrust model. Using system identification, a transfer function relating rotor speed and thrust is created. Finally, a state feedback controller is developed and tested using the non-linear dynamics model. Results demonstrate the controller's capacity to effectively maintain stability when confronted with external disturbances.

2 Review of the Relevant Literature

2.1 Relevant Literature Overview

The following section is a review of the relevant literature, covering the fundamental topics needed to understand this work. The following topics are reviewed: UAV configurations, mathematical modeling of dynamic systems, linearization, stability, linear state feedback control, disc actuator theory and system identification.

2.2 UAV Configurations

One of the most common UAV configurations in literature today is the quadcopter, featuring four equally sized propellers with constant blade pitch. The quadcopter has four rotors in a cross configuration, where the opposing blade pairs spin clockwise, while the other pair spin counterclockwise, effectively balancing the overall torque. This orientation results in a relatively simple dynamic model at the expense of controller complexity due to the coupled dynamics and underactuated configuration [19].

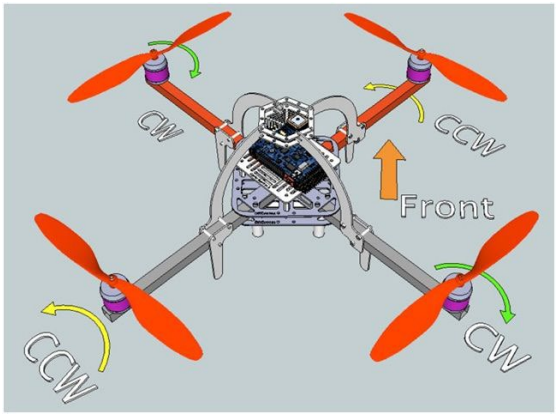


Figure 2.1: Quadcopter Configuration [20]

In contrast, a helicopter has two propellers of vastly different sizes, one with variable pitch, rotating orthogonal to each other, resulting in more complex dynamics. Typically, rotorcraft use a swashplate mechanism to turn operator inputs into collective and cyclic rotor blade control [21], allowing the vehicle to translate and pitch in all directions. This mechanism is composed of a complex arrangement of bearings, ball joints, and linkages. A simpler structure would be advantageous, and even required in specific applications due to technical limitations. One example is in UAV

miniaturization, where the mechanical swashplate cannot be achieved due to size constraints [22]. A feasible alternative method of control is thrust vectoring or center of gravity steering. Thrust vectoring is a method of control where the thrust angle can be changed relative to the vehicle, introducing a horizonal component of thrust. It is a common control technique used on missiles and rockets as well as some military aircraft, such as the F-22 and F-35, providing capabilities such as vertical takeoff and landing, and super maneuverability. [23]. The CoaxUAV vehicle employs thrust vector control, replacing the swashplate with two servo motors and a two axis gimble. Coaxial contra-rotating rotors of constant pitch are used for propulsion and to balance torque. This pair of rotors controls the vehicles vertical y-axis translational motion and yaw motion around the y-axis. See Figure 3.1. Tilting the rotors at an angle to the body results in a horizontal component of thrust force, and translational motion.

Figure 2.2: Coaxial Thrust Vector UAV [8]

2.3 Mathematical Modeling of Dynamic Systems

The motion of a dynamic system, such as a UAV, is first understood by analyzing the degrees of freedom (DOF) of the vehicle. DOF is a measure of all the independent motions a system can take, such as a linear translation or rotation about an axis. Constraints inhibit motion and reduce the DOF for the system [24]. Looking at a system's DOF is useful for analyzing how the system can and will behave given certain inputs.

Analyzing the DOF of a fixed pitch rotorcraft, the spinning blade generates a thrust vector to push or pull the vehicle, as well as generating a torque which can induce rotation around the thrust vector axis. However, it lacks the ability to perform lateral translation or rotation about any other axis, unless additional actuators are added. The vehicle configuration establishes the DOF and the constraints for the system and allows for a mathematical model to be constructed.

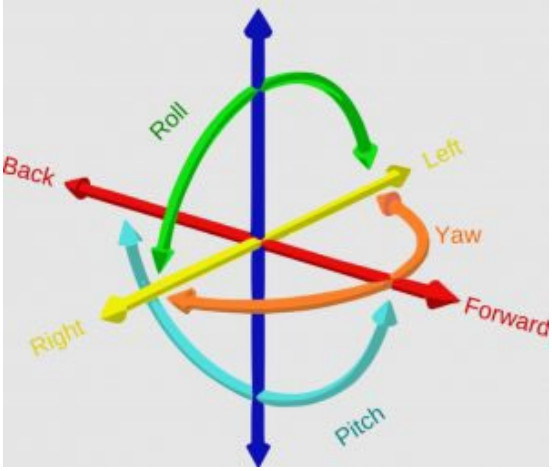


Figure 2.3: DOF Illustration [25]

There are a variety of structures that can be used for modeling the system plant, each with benefits and drawbacks. These structures include non-linear and linear differential equations, transfer functions, and state-space representations. The transfer function gives a more intuitive system representation but has a single-input single-output (SISO) structure, where state-space benefits from allowing multiple-inputs multiple-outputs (MIMO). A system will generally be represented in the form of differential equations, or block diagrams [26]. The approaches to finding this model can be divided into two general categories, first principles modeling and data driven modeling. Data driven modeling is explored in the system identification section. First principles modeling typically involves using Newtons second law $F = ma$ or the Lagrangian energy method $L = \mathrm{T}_l - \mathrm{V}_l$ to solve for the non-linear equations of motion (EOM). Using either of these methods should result in the same equations, though typically one method requires less complex computation depending on the system [24].

Two common approaches to modeling the system are developing state-space models and transfer functions. To fit the system into these models, the system must be represented in a linear form [26]. The process of taking non-linear differential equations and finding their linear approximations is called linearization [27]. This process can now be accomplished quickly using tools in MATLAB and Simulink which will numerically find the linear approximations quickly. The underlying mathematics behind this process equates the non-linear trigonometry functions into their Taylor Series representation. The leading term of this series can be used as an approximation for the entire series. Plugging in a value for the variable will result in a linear approximation at that specific point. The chosen point is decided based on where the vehicle is stable, or where the vehicle tends to stay within a given state. These points are called fixed points and are where the EOM are linearized around. Typically, it is useful to linearize the equations of motion around fixed points, such as hover, as this is where the control system tries to stabilize the system [27].

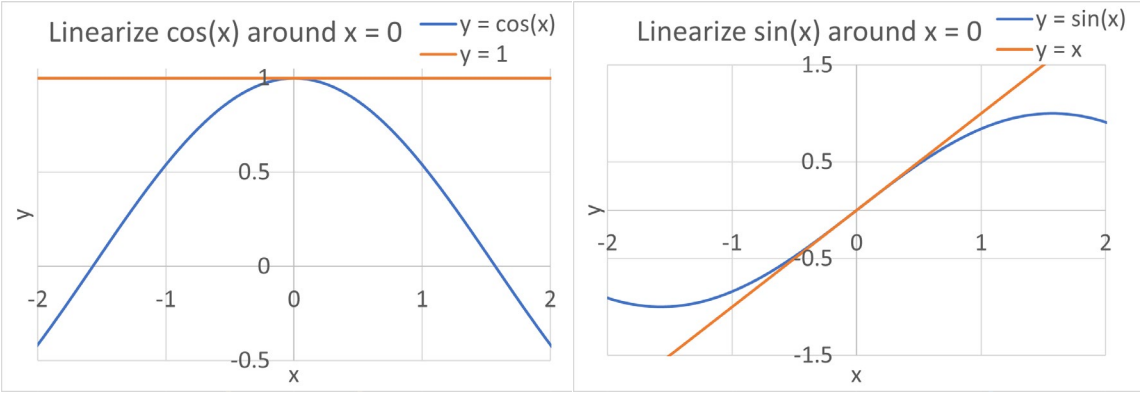


Figure 2.4: Linearization of cosine and sine functions at x=0

Figure 2.4 shows the graphical representation of linearization. Mathematically, linearization involves taking the partial derivative of each equation with respect to each of the states of the system, then plugging in the operating state values into the new matrix. The consolidated process is called the Jacobian matrix, shown in Equations 2.3 and 2.4, which is used to linearize the system about this operating state. It also represents the first term in the Taylor series expansion. This process of linearizing systems is critical for developing the modern approach to system modeling, called the state-space

representation. The state-space representation of a linear system can be written in the following form:

$$\dot{\bar{x}} = A\bar{x} + B\bar{u} \tag{2.1}$$

$$\bar{y} = C\bar{x} + D\bar{u} \tag{2.2}$$

where $\bar{x}$ is a vector of the system states, $\dot{\bar{x}}$ is the state derivatives with respect to time, $\bar{u}$ are the system inputs, and A, B, C and D are constant matrices. A and B relate the system states and inputs to the state derivative, while C and D relate the system states and inputs to the system outputs [26]. In order to find the A and B system matrices, the non-linear equations of motion must be linearized using the Jacobian matrix:

$$A = \frac{\partial f}{\partial \underline{X}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3} & \frac{\partial f_1}{\partial x_4} & \frac{\partial f_1}{\partial x_5} & \frac{\partial f_1}{\partial x_6} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \frac{\partial f_2}{\partial x_3} & \frac{\partial f_2}{\partial x_4} & \frac{\partial f_2}{\partial x_5} & \frac{\partial f_2}{\partial x_6} \\ \frac{\partial f_3}{\partial x_1} & \frac{\partial f_3}{\partial x_2} & \frac{\partial f_3}{\partial x_3} & \frac{\partial f_3}{\partial x_4} & \frac{\partial f_3}{\partial x_5} & \frac{\partial f_3}{\partial x_6} \\ \frac{\partial f_4}{\partial x_1} & \frac{\partial f_4}{\partial x_2} & \frac{\partial f_4}{\partial x_3} & \frac{\partial f_4}{\partial x_4} & \frac{\partial f_4}{\partial x_5} & \frac{\partial f_4}{\partial x_6} \\ \frac{\partial f_5}{\partial x_1} & \frac{\partial f_5}{\partial x_2} & \frac{\partial f_5}{\partial x_3} & \frac{\partial f_5}{\partial x_4} & \frac{\partial f_5}{\partial x_5} & \frac{\partial f_5}{\partial x_6} \\ \frac{\partial f_6}{\partial x_1} & \frac{\partial f_6}{\partial x_2} & \frac{\partial f_6}{\partial x_3} & \frac{\partial f_6}{\partial x_4} & \frac{\partial f_6}{\partial x_5} & \frac{\partial f_6}{\partial x_6} \end{bmatrix} \tag{2.3}$$

$$B = \frac{\partial f}{\partial \underline{u}} = \begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \frac{\partial f_1}{\partial u_2} \\ \frac{\partial f_2}{\partial u_1} & \frac{\partial f_2}{\partial u_2} \\ \frac{\partial f_3}{\partial u_1} & \frac{\partial f_3}{\partial u_2} \\ \frac{\partial f_4}{\partial u_1} & \frac{\partial f_4}{\partial u_2} \\ \frac{\partial f_5}{\partial u_1} & \frac{\partial f_5}{\partial u_2} \\ \frac{\partial f_6}{\partial u_1} & \frac{\partial f_6}{\partial u_2} \end{bmatrix} \tag{2.4}$$

State-space modeling is a MIMO method which leverages all the mathematical power of matrices and linear algebra. Real systems are typically non-linear in nature but must be linearized to perform state-space analysis.

To find the transfer function of the system, first the EOM must undergo the Laplace transform, which is a mathematical technique where the time-domain function can be converted into the complex plane. This has the benefit of turning differential equations into variables with coefficients, allowing algebraic manipulation of the equations, rather than calculus relationships [26]. The Laplace transform allows us to develop the transfer function model of a differential equation, where we have an input to the system, and we can model the output. This can also be thought of as a block model [26]. These manipulations can be done on any system modeled as a differential equation, including physical or electrical systems. The transfer function is a SISO system, which forces the designer to carefully consider which variables are of importance. If multiple variable relationships are necessary, multiple SISO transfer functions can be constructed. In Figure 2.5 the subsystems can be replaced with a transfer function.

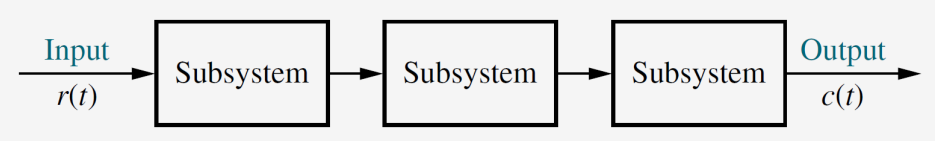


Figure 2.5: Block Diagram [26]

2.4 Control Theory

For each configuration of UAV, a controller must be developed to maintain and restore the desired vehicle attitude even while experiencing a range of external disturbances. In most cases the UAV will require an embedded system to continually collect sensor data and output commands to the control surfaces, actuators, and motors. To develop a controller, the use of a system or plant model is extremely helpful. The modeling approach allows for analysis for the system dynamics as well as optimizing controller parameters **[28]**.

PID (proportional, integral, derivative) controllers are ubiquitous in many engineering applications, including UAV flight **[29]**. A PID controller works by

measuring a variable in the system called the process variable (PV) and comparing it to the desired value for that variable called the setpoint (SP). The difference between the PV and SP is the error, and the controller's goal is to minimize that error value **[29]**. The error is manipulated in three ways, proportionally, by taking its integral over time, and by taking its derivative over time. Each of these three manipulations is multiplied by a constant coefficient term, called the gain. The sum of these three terms is the output of the controller, and the input to the system. Finding these three parameters is called tuning, and can be done by trial and error, but this typically does not result in the best possible controller. By using a system model, performance-specific PID turning can be accomplished, and the optimal controller can be found **[29]**.

State feedback controllers are an effective method of control that requires the system's state values, or approximations, to be measured. After finding a linearized model for the system in state-space form, the system should be checked for controllability and observability. Controllability is a method of checking if the system can be controlled, while observability is a method to check if all the necessary states of the system can be measured. One can check the controllability by constructing the controllability matrix, P, using the A and B matrices found in Equations 2.3 and 2.4. If the controllability matrix is full rank, the system is controllable.

$$P = [B \;\; AB \;\; A^2B \; A^3B] \tag{2.5}$$

One can check observability by constructing the observability matrix, Q, and checking if it is full rank.

$$Q = \begin{bmatrix} C \\ CA \\ CA^2 \\ CA^3 \end{bmatrix} \tag{2.6}$$

The stability of the open-loop system can also be investigated by taking the eigenvalues of the system's A matrix. Eigenvalue analysis involves setting up the classic eigenvalue problem, 2.7, then solving for the eigenvalues, λ, that satisfy the equation.

$$A\bar{v} = \lambda\bar{v} \quad \rightarrow \quad (A - I\lambda)\bar{v} = 0 \quad \rightarrow \quad det(A - I\lambda) = 0 \tag{2.7}$$

were A is the linearized system, $\bar{v}$ is the eigenvector associated with the system, λ is the eigenvalue associated with the eigenvector, and I is the identity matrix. Negative eigenvalues indicate an asymptotically stable system, while positive eigenvalues indicate an unstable system. Eigenvalues of zero represent a neutrally stable, or Lyapunov stable system that will oscillate about the fixed point while never reaching it. Any positive eigenvalues will make the entire system unstable, as that one value could grow toward infinity. Due to the exponential nature of eigenvalues, the farther they are from zero, the faster the system's response will be.

To stabilize a system and manipulate how it behaves, a closed-loop controller is typically introduced. The state feedback controller is developed by setting the inputs of the system, $\bar{\boldsymbol{u}}$**,** equal to a controller gain, K, multiplied by the current states of the system, $\bar{\boldsymbol{x}}$, as follows:

$$\bar{u} = -K\bar{x} \tag{2.8}$$

Substituting Equation 2.8 into Equation 2.1 results in an equation for the closed-loop system:

$$\dot{\bar{x}} = (A - BK)\bar{x} \tag{2.9}$$

It can be seen in Equation 2.9 that the system's A matrix, which cannot be changed, has been replaced with $A - BK$, which can be influenced by choosing values for K. The method of finding a controller gain matrix, K, that provides desired closed-loop eigenvalues, is called the pole placement technique. By making all the closed-loop eigenvalues negative, an unstable system can be stabilized [27]. A controller gain, K, will next be found using the eigenvalue method, giving the closed-loop system the desired eigenvalues. First the coefficient vector, $\bar{\Phi}_i$**,** is solved for using Equation 2.10:

$$[\lambda_{d_i} I - A \quad B]\bar{\Phi}_i = \bar{0} \tag{2.10}$$

Breaking $\bar{\Phi}_i$ into its components, the eigenvector, $\bar{\psi}$, and the controller gain multiplied by the eigenvector, $K\bar{\psi}$:

$$\bar{\Phi}_l = \begin{bmatrix} \bar{\psi} \\ K\bar{\psi} \end{bmatrix} \tag{2.11}$$

The controller gain, K, can then be solved for via the following:

$$K = [K\bar{\psi}][\bar{\psi}]^{-1} \tag{2.12}$$

Finally, the new gain controller, K, is checked by finding the eigenvalues of the closed-loop system and confirming they are the desired, stable eigenvalues. This method of control will stabilize the system around the determined fixed points but could become unstable if the system states move out of the linearized region.

2.5 Disc Actuator Theory

Disc Actuator Theory is a simple rotor theory that treats rotating propellers as an infinitely thin disc that creates an instantaneous pressure delta across itself. Additional assumptions of this theory include [30]:

• The thrust is uniformly distributed over the disc

• No rotational motion is imparted into the flow by the actuator disc

• The pressure far ahead and far behind the disc are equal to ambient pressure

Figure 2.6 shows the control volume of the streamtube encompassing the actuator disc, where V_0 is the fluids freestream velocity, V_1 is the fluids velocity passing through the actuator disc, and V_2 is the exit velocity.

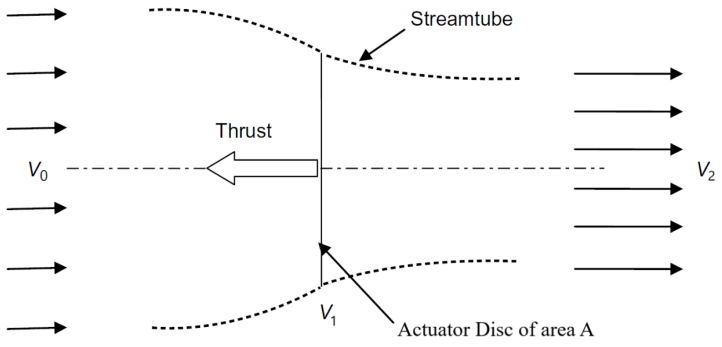


Figure 2.6: Disc Actuator Diagram

The propeller pitch, s, is a specific characteristic of a propeller design and is defined as the theoretical distance the propeller will move forward during one complete rotation in a viscous fluid. The thrust generated by the propeller F_1, can be calculated using the change in pressure across the actuator disc, ΔP, and its circular area, A_{disc}:

$$\mathrm{F}_1 = \Delta P * A_{disc} \tag{2.13}$$

Bernoulli's energy equation for fluid combines the static pressure, P, dynamic pressure, $\frac{1}{2}\rho V^2$, and potential energies, $\rho g h$ before and after an event, where ρ is fluid density, V is the fluid velocity, g is the gravitational constant, and h is the fluid height:

$$P_0 + \frac{1}{2}\rho {V_0}^2 + \rho g h_0 = P_2 + \frac{1}{2}\rho {V_2}^2 + \rho g h_2 \tag{2.14}$$

The volume before and after the disc are open to the atmosphere and therefore have equal static pressures, $P_0 = P_2$. As a simplifying assumption, the disc is assumed to have no thickness, so the potential energy terms will be ignored, $h_0 = h_2$. With these simplifications, Bernoulli's equation becomes:

$$\frac{1}{2}\rho {V_0}^2 = \frac{1}{2}\rho {V_2}^2 \tag{2.15}$$

The propeller disc will add energy to the fluid in the form of a pressure differential across the disc, ΔP. With this added energy, the Bernoulli energy equation becomes:

$$\frac{1}{2}\rho {V_0}^2 + \Delta P = \frac{1}{2}\rho {V_2}^2 \tag{2.16}$$

Solving for the disc actuator's added energy, Equation **2.16** becomes:

$$\Delta P = \frac{1}{2}\rho\left({V_2}^2 - {V_0}^2\right) \tag{2.17}$$

Substituting Equation 2.17 into Equation 2.13, we get:

$$F_1 = \frac{1}{2}A_{disc}\rho\left({V_2}^2 - {V_0}^2\right) = \frac{1}{2}A_{disc}\rho(V_2 - V_0)(V_2 + V_0) \tag{2.18}$$

The velocity of air across the actuator disc is the average of the upstream and downstream air velocities:

$$V_1 = \frac{1}{2}(V_0 + V_2) \tag{2.19}$$

Solving for V_0:

$$V_0 = (2V_1 - V_2) \tag{2.20}$$

Plugging Equations 2.19 and 2.20 into Equation 2.18, the equation becomes:

$$\mathrm{F}_1 = 2\rho A_{\mathrm{disc},1} {V_1}^2 \tag{2.21}$$

The velocity of the air passing through the disc is assumed to be equal and opposite the velocity of the disc moving through the air. The air velocity, V_1, can be calculated using the propeller pitch, s, in meters and the angular speed of the propeller, ω, in revolutions per minute (rpm)

$$V_1 = \omega \cdot \frac{1}{60} \cdot s \tag{2.22}$$

Plugging Equation 2.22 into 2.21 results in an expression for the thrust force generated by the propeller. With the assumption that the air density is constant, this equation relates the propellers rotational speed, ω, to the generated force F_1.

$$\mathrm{F}_1 = 2\rho A_{disc,1}\left(\omega \cdot \frac{1}{60} \cdot \mathrm{s}\right)^2 = \frac{2}{3600}\rho A_{disc,1}\mathrm{s}^2\omega^2 \tag{2.23}$$

2.6 System Identification

System identification is a data-driven modeling approach that utilizes collected data from a system to construct a model representing its behavior. It goes beyond simple curve fitting, which aims to find an equation that best matches a specific set of data. Instead, system identification focuses on developing a model, often in the form of a differential equation, which establishes the relationship between the system's inputs and outputs [31]. To create this system model, a suitable framework is initially chosen. This framework can take various forms, but a transfer function or a state-space model are typical. The model's parameters are then optimized by iteratively adjusting them and evaluating how closely the model output matches the actual estimation output data. This process continues until the best fit between the model and the estimation data is achieved

[31]. This procedure would be extremely tedious if done by hand yet is well suited for a computer. The optimization stage introduces the concept of a cost function, which increases as the model deviates further from the reference data. Therefore, the computer's objective is to minimize this cost function. System identification software uses numerical methods within software tools that cycle through and adjust all the parameters until the cost function is minimized [31]. The real test of the model is when you compare it with different data from the same system, this is called validation. [31]. Estimation data is used for creating the best fit model, and validation data is used for evaluating the model to see how accurate it is. Continually increasing the number of model parameters, or system order, can result in overfitting. This is where the fit to the estimation data will keep increasing, but the fit to the validation data will start getting worse, as shown in Figure 2.7. This thesis presents a transfer function created using the MATLAB System ID toolbox [31].

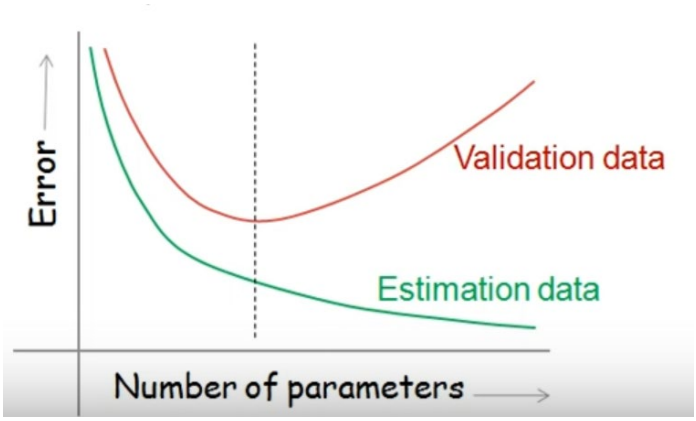


Figure 2.7: Overfitting Estimation Data

In practice, system data is collected by inputting sine sweeps, doublets, or other signals into the system, then recording the output signals. Sine sweeps cover a large range of frequencies, exciting more of the systems dynamics, and resulting in a better model. Time-domain data collected from the system should be sampled at uniformly spaced time intervals. Both the input and output signals must be recorded and used for the system identification algorithm [32].

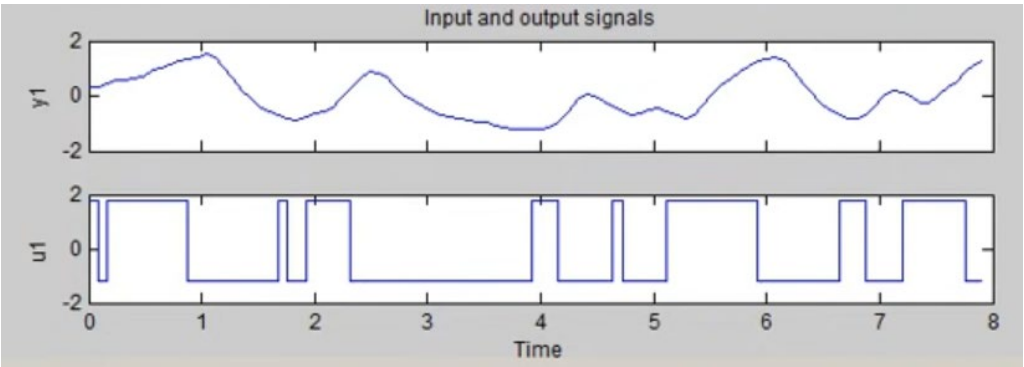


Figure 2.8: Example Input and Output Data

Once the system data is collected in the time-domain, it is transformed into the complex domain by using the Fourier transform.

$$Y(i\omega) = \int_{-\infty}^{\infty} y(t)e^{-i\omega t}dt$$
$$U(i\omega) = \int_{-\infty}^{\infty} u(t)e^{-i\omega t}dt \qquad 2.24$$

Here Y represents the system's output data, and U represents the system's input data, both of which are represented as column vectors of data. Next the model structure needs to be specified. For example, the transfer function structure can be used, with a given number of poles and zeros [32]. The transfer function structure is a ratio of polynomials and is defined as follows:

$$\mathrm{H(s)} = \frac{(\mathrm{A_0} + \mathrm{A_1 s} + \mathrm{A_2 s^2} + \cdots)}{(\mathrm{B_0} + \mathrm{B_1 s} + \mathrm{B_2 s^2} + \cdots)} \qquad 2.25$$

For example, a spring mass damper system with no zeros and two poles, the transfer function is given by:

$$\mathrm{H(s)} = \frac{1}{(\mathrm{ms^2} + \mathrm{cs} + \mathrm{k})} \qquad 2.26$$

MATLAB's System ID toolbox uses an autoregressive-moving average with exogenous (ARMAX) function to estimate model parameters. The ARMAX function is an iterative solver that continually refines model parameters until predefined criteria are

met [33]. Once the best fit model is found, a different order model is typically developed using the same process. For a transfer function, the number of zeros and poles are sequentially increased until the fit tapers off and begins to decrease. Next different order transfer functions are tested with validation data. The new validation input data is converted into the complex domain via the Laplace transform. The inputs are fed through each transfer function producing an array of output data for each function. That data is converted back into the time-domain by use of the inverse Laplace transform. All the different transfer function model's outputs are then compared to the actual system validation output. The transfer function with an output best fitting the validation data is chosen. If the model is a good predictor of how the system will respond to inputs, then both the model's output data and validation output data should be strongly correlated [32]. This process is depicted in Figure 2.9.

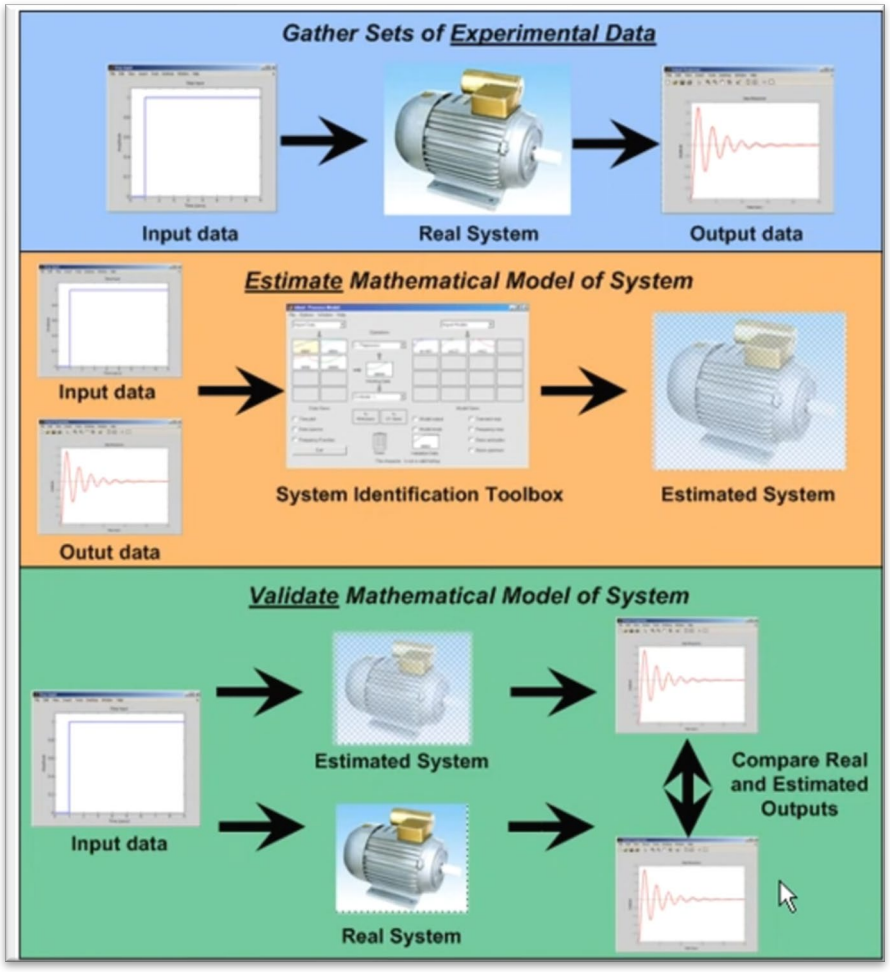


Figure 2.9: System Identification Process [32]

3 Methodology

The objective of this research is to demonstrate the ability of a full-state feedback controller to reject disturbances on a coaxial thrust vectored UAV. To achieve this a theoretical dynamics model for the vehicle is established, then used to develop a state feedback controller to study maneuverability and stability. A hardware prototype is constructed to demonstrate manufacturability and provide testing data. A simplified thrust model is established then improved using a regression line and system identification with data from the vehicle prototype. The controller results are demonstrated through simulations.

The primary research questions addressed in this study are as follows:

1. How can a theoretical thrust model be improved by gathering prototype test data?
2. Can a state feedback controller be developed to stabilize a thrust vectored coaxial UAV model when confronted with external disturbances?
3. How can a state feedback controller's gains be characterized and compared?

3.1 Dynamic System Modeling

This section discusses the simplifying assumptions made while analyzing this dynamic system. The CoaxUAV is cylindrical with approximately identical servo control in the x and z coordinates. This inherent symmetry allows for system analysis to be conducted in 2D, as the model can then be equally applied to both the x-y and z-y directions. From here forward analysis will be conducted in the x-y plane. On a coaxial vehicle the propellers are contra rotating and the sum of their angular speeds is approximately zero, so their combined gyroscopic effect is also negligible. Due to this characteristic of coaxial rotors, the gyroscopic forces will be neglected from this model.

The consolidated simplifying assumptions are:

1. System dynamics are modeled in 2D, on an x-y plane.
2. The gyroscopic forces are neglected.

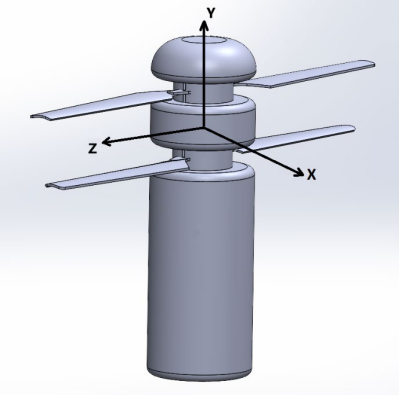


Figure 3.1: Coaxial UAV Coordinate System

Creating a model of the Coaxial UAV involves deriving the non-linear equations of motion for the vehicle. The first step is defining the reference frames. The thrust frame is attached to the top portion of the UAV, called mass 1 or M1, while the body frame is attached to the lower portion of the UAV, called mass 2 or M2. The body frame is where the flight computer inertial measurement unit (IMU) will be located, and thus where all the vehicle state information will be gathered. The inertial frame, or Newtonian frame, is attached to the ground and does not move with respect to the observer.

$$Thrust\ Frame:\ t = \begin{bmatrix} t1 \\ t2 \end{bmatrix} \tag{3.1}$$

$$Body\ Frame:\ b = \begin{bmatrix} b1 \\ b2 \end{bmatrix} \tag{3.2}$$

$$Inertial\ Frame:\ n = \begin{bmatrix} n1 \\ n2 \end{bmatrix} \tag{3.3}$$

The system has two inputs, a thrust force, F_t, and a servo angle Ψ.

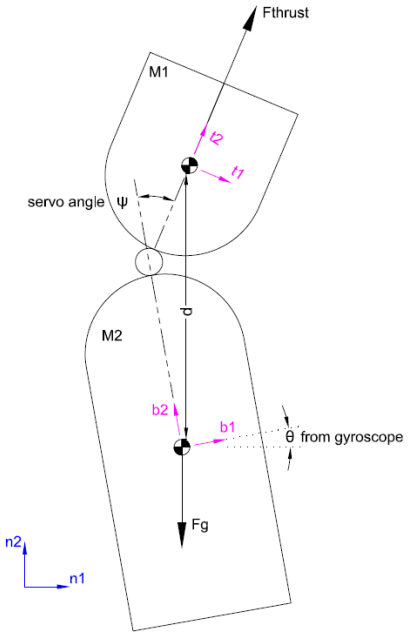


Figure 3.2: Coaxial UAV free body diagram

Analyzing the system, it can be seen that the thrust force only acts in the t2 direction. The thrust force, F_t, in the thrust frame can be written:

$$\bar{F}_t^{\,t} = \begin{bmatrix} 0 \cdot t1 \\ F_{thrust} \cdot t2 \end{bmatrix} = \begin{bmatrix} 0 \\ F_t \end{bmatrix} \tag{3.4}$$

Rotation matrices will be used to relate reference frame orientations based on their angle difference. The thrust force can be written in terms of the body frame where the servo angle relates the two frame orientations.

$$\bar{F}_t^{\,b} = \begin{bmatrix} {F_{t_1}}^b \\ {F_{t_2}}^b \end{bmatrix} = \begin{bmatrix} \cos(\Psi) & \sin(\Psi) \\ -\sin(\Psi) & \cos(\Psi) \end{bmatrix} \begin{bmatrix} 0 \\ F_t \end{bmatrix} = \begin{bmatrix} F_t \sin(\Psi) \\ F_t \cos(\Psi) \end{bmatrix} \tag{3.5}$$

The body frame can then be related to the inertial frame by again using a rotation matrix and the body frames inertial rotation angle θ, measured by the IMU. Here the thrust force is written in the inertial frame:

$$\bar{F}_t^{\,n} = \begin{bmatrix} {F_{t_1}}^n \\ {F_{t_2}}^n \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} F_t \sin(\Psi) \\ F_t \cos(\Psi) \end{bmatrix} = \begin{bmatrix} F_t \sin(\Psi)\cos(\theta) + F_t \cos(\Psi)\sin(\theta) \\ -F_t \sin(\Psi)\sin(\theta) + F_t \cos(\Psi)\cos(\theta) \end{bmatrix} \tag{3.6}$$

Additional forces acting on the vehicle are drag, F_D, and gravity, g. The drag force is a result of air resistance and is modeled as:

$$F_D = \frac{1}{2}\rho v^2 C_D A \tag{3.7}$$

The following definitions and assumptions are used: Density of air: $\rho = 1.2\ kg/m^3$, coefficient of drag: $C_D = 1$, Cross sectional area: $A = 0.018m^3$. From Equation 3.17

$$F_D \approx 0.01v^2 \tag{3.8}$$

The gravitational force, $F_{gravity}$, only acts in the inertial i2 direction and is defined as:

$$F_{gravity} = mg = m9.81$$

Newtons second law, $\sum F = ma$, can now be applied to derive the translational equations of motion in the inertial frame.

$$\sum F = m\begin{bmatrix}\ddot{x}\\ \ddot{y}\end{bmatrix} = \begin{bmatrix} F_t \sin(\Psi)\cos(\theta) + F_t\cos(\Psi)\sin(\theta) \\ -F_t\sin(\Psi)\sin(\theta) + F_t\cos(\Psi)\cos(\theta)\end{bmatrix} - \begin{bmatrix}0.01\dot{x}^2\\ 0.01\dot{y}^2\end{bmatrix} - \begin{bmatrix}0\\ mg\end{bmatrix}$$

A similar process can be applied to solving the rotational equation of motion in the inertial frame. The thrust force acts along the t2 axis, and as a result, torque is not generated in that reference frame. Rotation matrices are used to relate the thrust force in perpendicular force in the body frame:

$$\begin{bmatrix}\cos(\Psi) & \sin(\Psi)\\ -\sin(\Psi) & \cos(\Psi)\end{bmatrix}\begin{bmatrix}0\\ F_t\end{bmatrix}$$

Using Newtons law for rotational motion and applied torque, τ:

$$\sum \tau = I\ddot{\theta} = [d \quad 0]\begin{bmatrix}\cos(\Psi) & \sin(\Psi)\\ -\sin(\Psi) & \cos(\Psi)\end{bmatrix}\begin{bmatrix}0\\ F_t\end{bmatrix} = [d \quad 0]\begin{bmatrix}F_t\sin(\Psi)\\ F_t\cos(\Psi)\end{bmatrix} = F_t\sin(\Psi)d$$

$$I\ddot{\theta} = F_t\sin(\Psi)d$$

Where $d = 0.15m$ and is the distance from the perpendicular force to the center of mass. Combining the translational and rotational equations, the complete equations of motion are

$$m\ddot{x} = F_t \sin(\Psi)\cos(\theta) + F_t\cos(\Psi)\sin(\theta) - 0.01\dot{x}^2$$

$$m\ddot{y} = -F_t \sin(\Psi)\sin(\theta) + F_t \cos(\Psi)\cos(\theta) - 0.01\dot{y}^2 - mg \quad (3.9)$$

$$I\ddot{\theta} = F_t\sin(\Psi)d$$

Defining the states and inputs:

$$States = \underline{X} = \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{Bmatrix} = \begin{Bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \\ \theta \\ \dot{\theta} \end{Bmatrix} \quad (3.10)$$

$$Inputs = \underline{u} = \begin{Bmatrix} u_1 \\ u_2 \end{Bmatrix} = \begin{Bmatrix} F_t \\ \Psi \end{Bmatrix}$$

Where F_t is the thrust input in Newtons, and Ψ is the servo angle in degrees. Converting Equation 3.9 into state-space form:

$$eqn_1 = \dot{x_1} = x_2$$

$$eqn_2 = \dot{x_2} = \frac{1}{m}(F_t \sin(\Psi)\cos(x_5) + F_t\cos(\Psi)\sin(x_5) - 0.01{x_2}^2)$$

$$eqn_3 = \dot{x_3} = x_4$$

$$eqn_4 = \dot{x_4} = -\frac{1}{m}(F_t \sin(\Psi)\sin(x_5) + F_t \cos(\Psi)\cos(x_5) - 0.01{x_4}^2 - mg) \quad (3.11)$$

$$eqn_5 = \dot{x_5} = x_6$$

$$eqn_6 = \dot{x_6} = \frac{1}{I}(F_t\sin(\Psi)\, d)$$

These non-linear equations of motion can be solved using a numerical solver such as Simulink or MATLAB ode45. Figure 3.3 shows these equations Simulink.

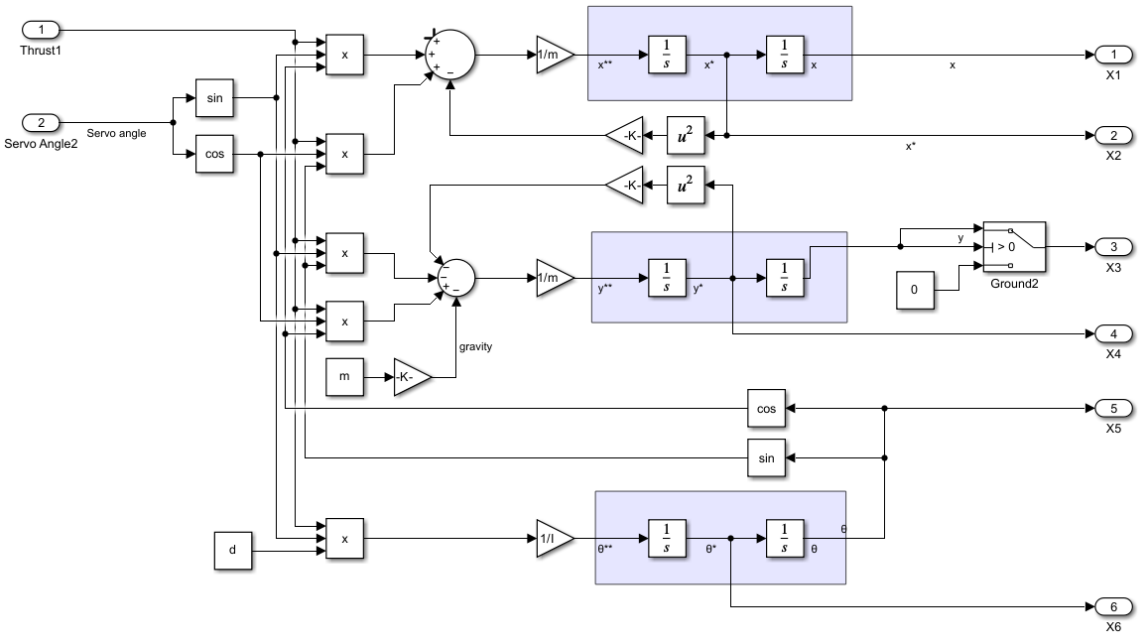


Figure 3.3: Coaxial UAV Simulink Non-Linear EOM

There are many benefits to working with linear differential equations, including the ability to use the state-space representation and create a linear state feedback controller. The following section outlines the steps to linearize the non-linear Equations 3.11. The systems equilibrium points are found by setting the state derivatives equal to zero, $f_i = 0$. The equilibrium points are found to be $x_2 = x_4 = x_5 = x_6 = \Psi = 0,\ T = mg$. The free variables are x_1, and x_3. This operating state corresponds to the coaxial UAV in hover at any x-y-z location with zero pitch angle, and zero translational and rotational velocities. The Jacobian matrix, Equations 2.3 and 2.4, are used to linearize Equations 3.11 at the equilibrium points:

$$A = \frac{Df}{d\underline{X}} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{F_t}{m_1 + m_2} & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{F_t(\pi - \Psi)}{m_1 + m_2} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad 3.12$$

$$B = \frac{Df}{d\underline{u}} = \begin{bmatrix} 0 & 0 \\ \dfrac{(\pi - \Psi + x_5)}{m_1 + m_2} & -\dfrac{F_t}{m_1 + m_2} \\ 0 & 0 \\ \dfrac{(-\pi x_5 + \Psi x_5 + 1)}{m_1 + m_2} & \dfrac{F_t x_5}{m_1 + m_2} \\ 0 & 0 \\ \dfrac{\pi - \Psi}{I} d & -\dfrac{F_t}{I} d \end{bmatrix}$$

Plugging in the equilibrium points, the A and B matrices are found. As noted above, it is assumed that all state variables can be measured, resulting in an identity C matrix and zero D matrix. The C matrix used here is more complex than the identity, showing that not all states need to be measured directly. The coaxial UAV can now be represented by the linear state-space form 2.1 and 2.2.

$$\dot{\bar{X}} = \begin{bmatrix} \dot{x_1} \\ \dot{x_2} \\ \dot{x_3} \\ \dot{x_4} \\ \dot{x_5} \\ \dot{x_6} \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{y} \\ \ddot{y} \\ \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & g & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & -g \\ 0 & 0 \\ \dfrac{1}{m} & 0 \\ 0 & 0 \\ 0 & -\dfrac{mgd}{I} \end{bmatrix} \begin{bmatrix} F_t \\ \Psi \end{bmatrix} \qquad 3.13$$

$$\bar{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} F_t \\ \Psi \end{bmatrix} \qquad 3.14$$

An approximation for the moment of inertia was calculated for initial system modeling. Using a uniform density cylinder with and an axis of rotation crossing the end diameter, Figure 3.4.

$$I = \frac{1}{4} m r^2 + \frac{1}{3} m h^2 \qquad 3.15$$

Where the mass = 1 kg, r = 10 cm, h = 150 cm, the moment of inertia is approximately $I = 0.0075\ kg \cdot m^2$. The hardware prototype's moment of inertia will be measured and compared to this theoretical value in the results section.

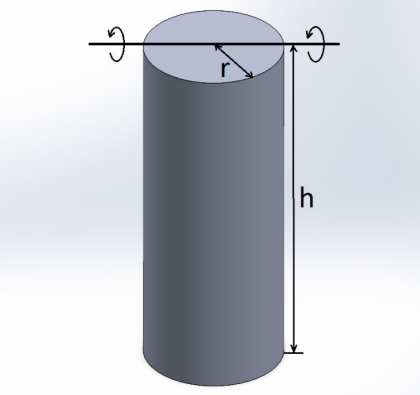


Figure 3.4: Moment of Inertia for a Cylinder

Initial testing of this new dynamic model involved using a universal serial bus (USB) game controller to manually fly the model through the simulation. This was done by mapping the controller inputs to the vehicles thrust and servo angle, allowing these values to be changed in real time and observing the vehicle response. The next steps involve removing the human in the loop and developing an automatic feedback controller.

3.2 Controller Development

This section covers the development of a state feedback controller and state visualization simulation based on the systems dynamic model. By introducing a controller gain, K, the closed-loop system eigenvalues can be moved into the negative real plane, creating an asymptotically stable system. Using the pole placement technique, described in section 2.4, a controller gain K can be designed to give the system specific eigenvalues, and thus desired stability characteristics. In situations where all the system states cannot be accurately measured, it is necessary to develop an observer to estimate the system states. The objective of the observer gain is to drive the error between the actual and estimated states to zero. Similar pole placement techniques are used for designing the observer gain L.

The first steps in developing a state feedback controller are checking if the system is controllable and observable. The open loop stability will then be analyzed, then the state controller will be developed.

One can check the controllability by constructing the controllability matrix P, using Equation 2.5. The controllability matrix P is full rank, $rank(P) = 6$, so the system is controllable.

$$P = \begin{bmatrix} 0 & 0 & 0 & -9.8 & 0 & 0 & 0 & -7.7 \\ 0 & 0 & 0 & 0 & 0 & -7.7 & 0 & 0 \\ 0 & 0 & 1.25 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & -785 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad 3.16$$

One can check the observability by constructing the observability matrix Q, shown by Equation 3.17. The observability matrix Q is full rank, $rank(Q) = 6$, so the system is observable.

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & g & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & g \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad 3.17$$

Eigenvalue analysis of the A matrix can be conducted to determine stability of the open loop system. Using Equation 2.7, all eigenvalues of the open loop system are found to be zero, meaning the system is neutrally stable. $\lambda_{1,2,3,4,5,6} = 0$. The system will oscillate around the equilibrium point indefinitely, not converging, because the damping effects of air resistance have been removed during linearization. A state feedback controller is then developed using the pole placement technique, to move the systems closed-loop eigenvalues into the stable left half plane. The desired eigenvalues for the system are first chosen to be negative and close to zero. Negative values are stable, and close to zero will make the controller less aggressive. The controller is developed, then the desired

eigenvalues are slowly moved more negatively, until the desired controller response is achieved. The process was followed until the following eigenvalues were found:

$\lambda_{d1} = -1.5, \lambda_{d2} = -1.6, \lambda_{d3} = -1.7, \lambda_{d4} = -1.8, \lambda_{d5} = -1.9, \lambda_{d6} = -2.0$

Using the analytical pole placement method and Equations 2.10 and 2.11, $\bar{\psi}$ and $K\bar{\psi}$ are found. The controller gain K is then found using Equation 2.12:

$$K = \begin{bmatrix} -2.11 & -3.71 & 2.593 & 3.061 & -21.088 & -3.851 \\ -0.007 & -0.016 & 0 & 0 & -0.13 & -0.048 \end{bmatrix} \quad 3.18$$

Checking the new gain controller K by finding the eigenvalues of the closed-loop system: $eig(A - BK)$. This analytical pole placement method can be rapidly achieved by using the MATLAB place() command, by which a variety controller gains can be easily tested. The closed-loop controller is then implemented into Simulink by multiplying the state vector by the controller gain and feeding it back to the system inputs. See Figure 3.5.

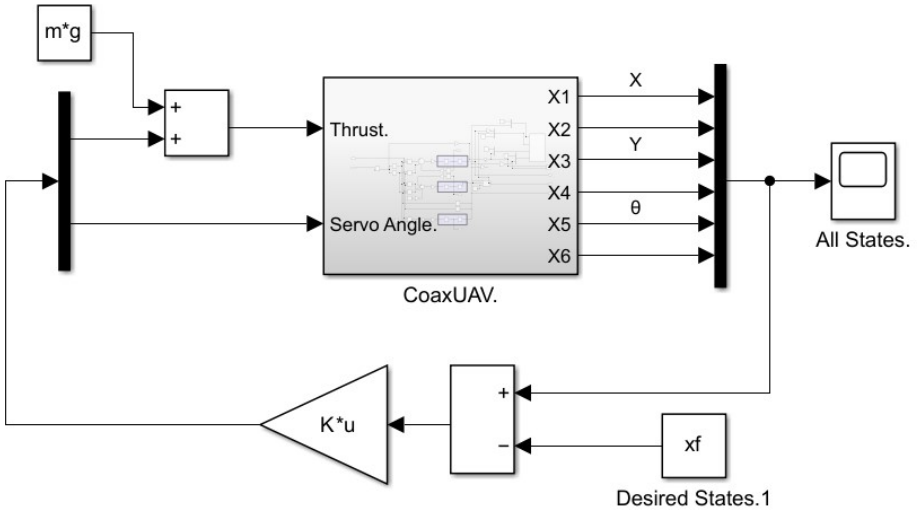


Figure 3.5: State Feedback Controller

Further constraints were then added to the controller model to better represent the CoaxUAV system. Based on testing data from the physical prototype, the thrust input was restricted to a maximum value of 18 Newtons, Table 4.3, while the gimble angle was restricted to 14 degrees, Table 3.1.

The controller was tested by giving it setpoints and observing the vehicle's response characteristics such as the settling time, overshoot, and ability to reach the desired states. External disturbances, such as simulated wind, were also introduced to the system to push the controller to its limits. These external forces included constant and impulse forces in both the x and y directions and applied rotational torque.

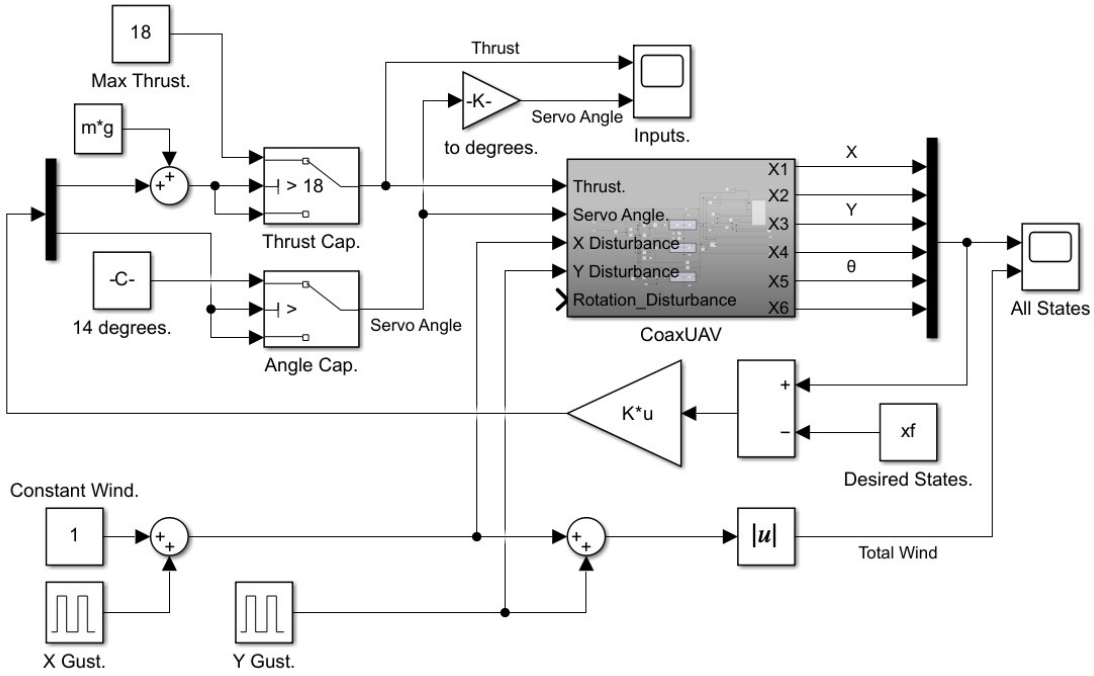


Figure 3.6: State Feedback Controller with Input Limitations and Wind Disturbances

The simulated vehicle and controller were subjected to a variety of simulated disturbance forces in the x-direction to test the limits of the controller's ability to recover the vehicle. For these tests, if the vehicles Y altitude stayed above 0m, it was considered a successful recovery. The applied disturbance can be defined by two variables, the magnitude of applied force in Newtons and the amount of time it was applied. These parameters were varied systematically and the limit values where the controller could recover stability were recorded and plotted.

The vehicle was also subjected to torsional disturbance forces, and the responses were analyzed. As before, the applied torque can be defined by two variables, the

magnitude in Nm and the amount of time it was applied. Again, these parameters were varied, and the limit of successful recoveries were recorded and plotted.

To better visualize the vehicles response to the controller, an animation was developed. Simulink's V-Realm Builder 2.0 was used to create a simulated representation of the CoaxUAV vehicle out of primitive shapes. The Simulink EOM model states were then mapped to the simulations associated with DOF. A USB gaming controller was then used to manually change system inputs and observe the simulations response over time.

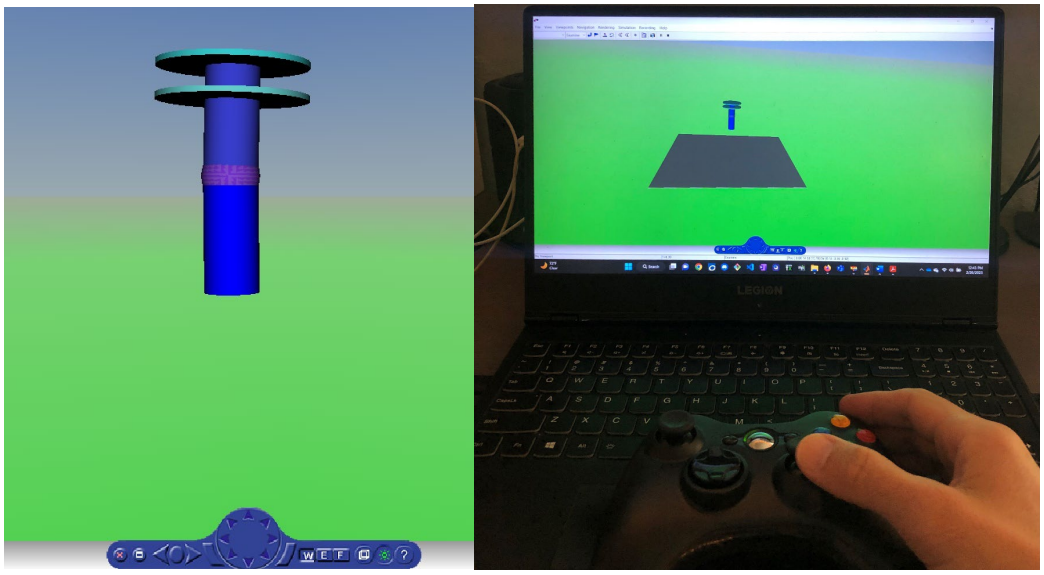


Figure 3.7: CoaxUAV Simulation

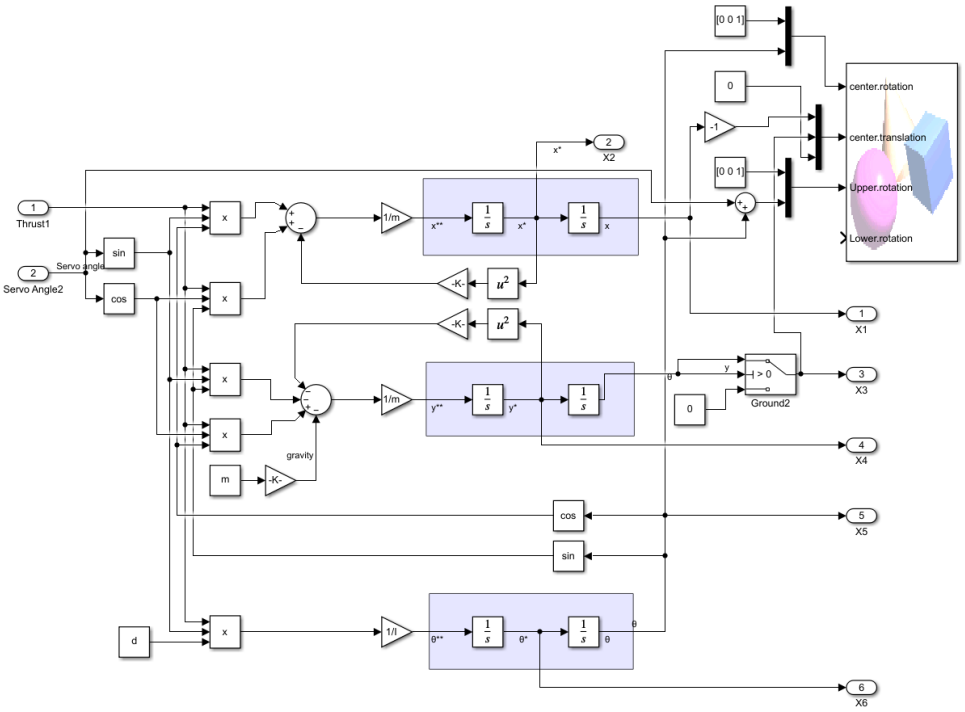


Figure 3.8: State Feedback Controller Model with V-Realm Model

3.3 Thrust Model

A simple theoretical thrust model was first developed as a starting point, then improved upon by gathering testing data. A transfer function model is also developed using the same thrust test data.

The simple model uses the disc actuator theory, where the propeller is replaced by an infinitely thin disc. This disc adds energy to the fluid in the form of a pressure differential. Using Equation **2.23**, the output force, F, can be calculated as a function of the propellers rotational speed.

$$F_1 = 2(1.204)(0.1078)\left(\omega \cdot \frac{1}{60} \cdot 0.1397\right)^2 = (1.407E-6)\omega^2 \qquad 3.19$$

For the CoaxUAV the disc area is an annulus, found by subtracting the smaller circular area from the larger circular area, where $A_1 = A_{annulus} = 0.1078\ m^2$. The air density is assumed to be a constant $\rho_{air} = 1.2\ kg/m^3$, and blade pitch s = 0.14 m. The overall

thrust of the coaxial UAV can be obtained by summing the thrust generated by each propeller:

$$F_{total} = F_{propeller\,1} + F_{propeller\,2} \tag{3.20}$$

Combining 3.19 and 3.20, assuming both propellers generate the same thrust:

$$F_{total} = (2.814E-6)\omega^2 \tag{3.21}$$

The model was then compared and plotted with thrust data from the hardware prototype, shown in Figure 3.9.

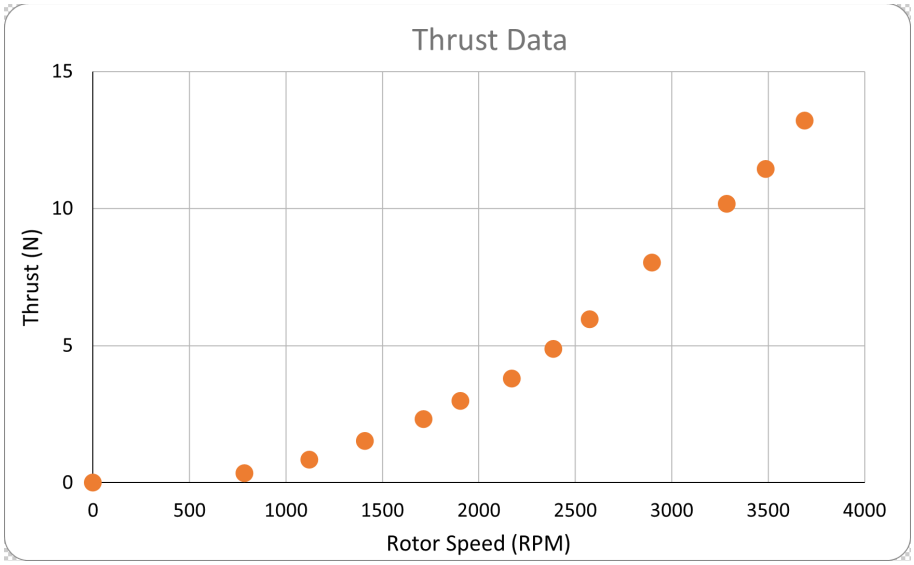


Figure 3.9: Rotor Speed vs. Thrust Data

Using Microsoft Excel's regression tool, a quadratic regression The majority of analysis done line is found for the thrust data, representing the actual thrust versus rotor speed relationship for this vehicle.

$$F_{regression} = (1E-6)\omega^2 \tag{3.22}$$

In order to adjust the model to fit this curve, a correction factor C and additional term $D\omega$ are added to the thrust Equation 2.23.

$$\mathrm{F_1} = C\frac{2}{3600}\rho A_1 \mathrm{s}^2\omega^2 + D\omega \tag{3.23}$$

The correction values are solved so the theoretical model, 3.21 matches the regression line, 3.22.

$$C(2.814E-6)\omega^2 = (1E-6)\omega^2$$

$$C = \frac{1E-6}{2.814E-6} = 0.3554 \tag{3.24}$$

The updated model with the correction factor

$$\mathrm{F}_1 = (0.35)\frac{2}{3600}\rho A_1 \mathrm{s}^2\omega^2 - 0.0002\omega \tag{3.25}$$

A transfer function thrust model is next developed using MATLABs System Identification toolbox and testing data from the CoaxUAV. First thrust testing is conducted on the CoaxUAV hardware system, where rotor speed and total thrust generated are both recorded. Multiple tests are conducted to gather model estimation and validation datasets. This input and output data is then uploaded into the MATLAB workspace. Both the model estimation data and validation data are uploaded into the system ID toolbox, shown in Figure 3.10.

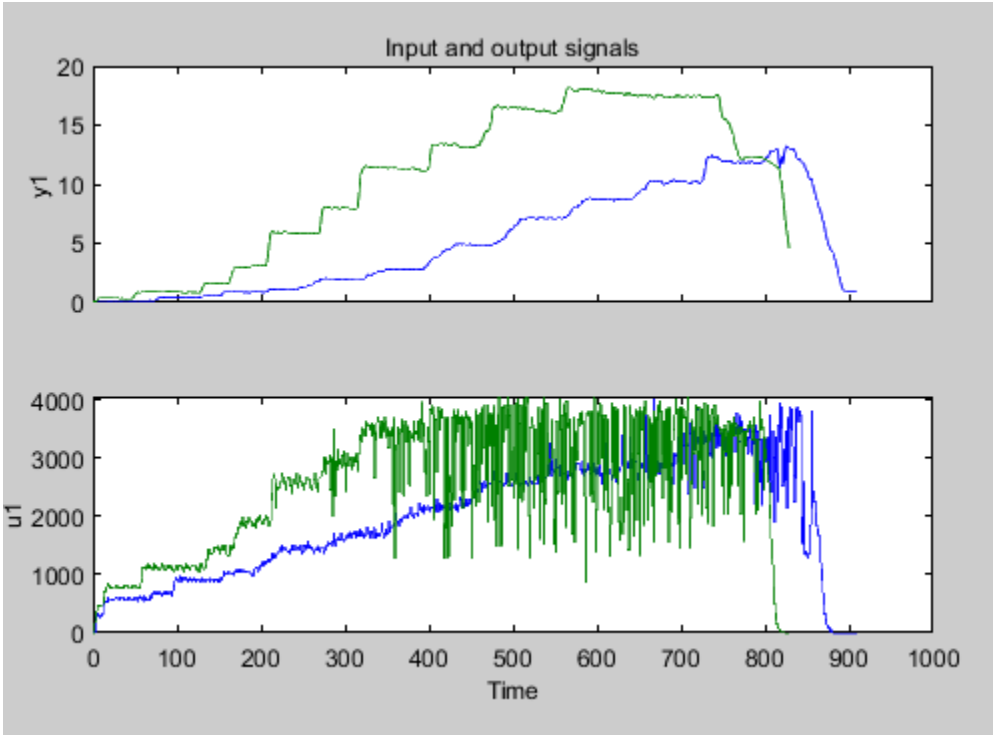


Figure 3.10: System ID I/O Estimation and Validation Data

The system inputs, u1 are rotor speed values in RPM, where the system outputs, y1 is the total thrust generated in Newtons. The model estimation data is in blue and the validation data in green.

The transfer function model is selected, Equation 2.25, and starting with one pole and no zeros, transfer function models are generated repeatedly while increasing the number of poles and zeros. This process continues until the model fit levels off and begins to decrease. The validation data is now selected, and the validation rotor speed input data is fed through each of the developed transfer function models. Each model's output is then compared to the actual validation thrust output data. The model providing the best fit is considered the most accurate model. Figure 3.11 shows the system ID toolbox interface.

Figure 3.11: System ID Toolbox

3.4 Hardware Prototype Development

Hardware development and testing was an iterative process where the design informed the test, and the test informed subsequent re-designs. Due to this feedback cycle, the report will follow a chronological approach, rather than attempt to isolate hardware design and data collection. There were two purposes of developing a hardware

prototype, the first was to prove that the design concept was feasible and provide a path for developing such a vehicle. The second was to gather testing data from the system for improving the thrust model. The design requirements for the hardware prototype are as follows:

1. Hardware configuration is a constant pitch coaxial thrust vector UAV.
2. Ability to provide enough thrust to lift twice the vehicles weight to provide control and flight authority.
3. Can be manufactured using a low-cost hobby level 3D printer.
4. 3D printed thrust vector and propulsion components able to withstand flight conditions.
5. Use of low-cost off-the-shelf components.

3.4.1 Test Stand

The test stand was developed for measuring the test vehicles thrust and torque. The stand encompassed two load cells measuring vertical thrust and rotational torque. Data was recorded using an Arduino Uno. The structure was designed in CAD (Computer-Aided Design) and then 3D printed in polylactic acid (PLA)+ thermoplastic.

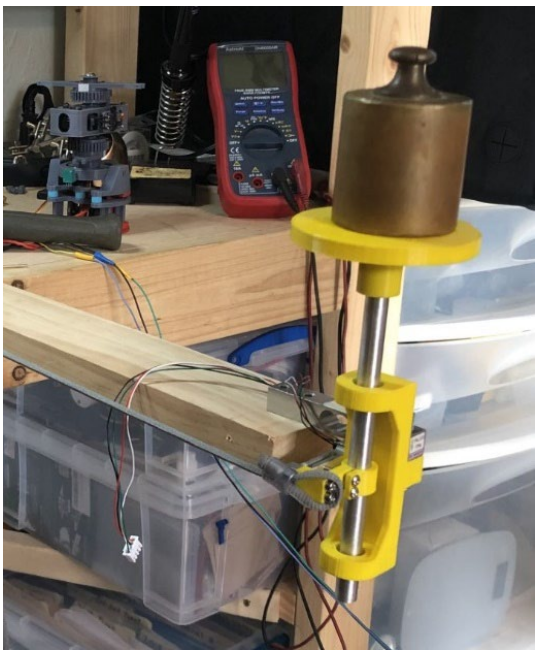


Figure 3.12: Test Stand with 1 kg Calibration Weight

The stand was calibrated using a one kilogram weight for vertical thrust, and a 200 gram weight for torque. Lift force was measured as a positive value, so the calibration weight shows a negative force.

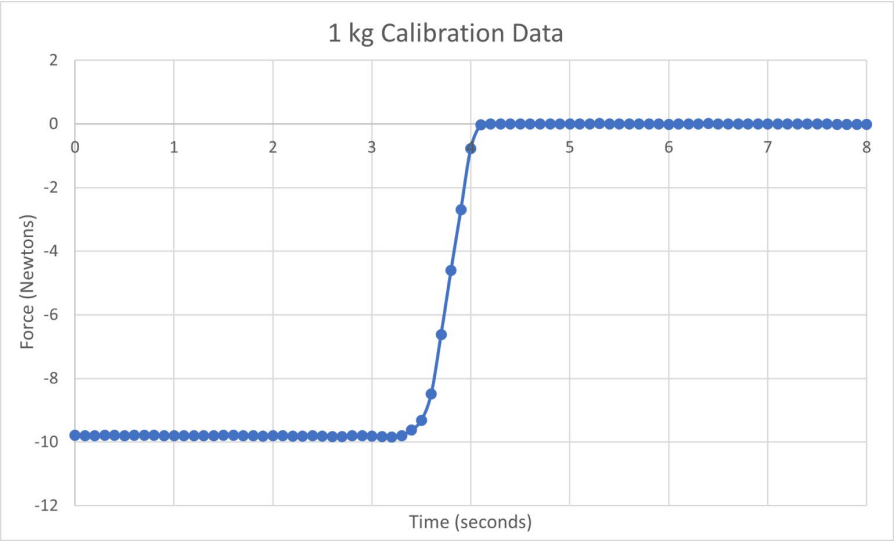


Figure 3.13: 1 kg Calibration Weight Being Removed from Stand

3.4.2 Hardware Design and Testing

Multiple design iterations were conducted before arriving at a functional prototype capable of meeting all design goals. SolidWorks was used as the CAD software, Ultimaker Cura was used as the slicer software, and Creality's Ender 3 3D printer used for rapid prototyping. The CAD software contains the digital model, and the slicer converts the digital model into G-code printing instructions that can be read by the 3D printer [34]. All the printer settings are established in the slicer software, including layer thickness, infill density, extrusion speed, temperature and much more. The propulsion section's design was initially inspired by the opensource Tdrone design [35]. This incorporated an enclosure with two planer aligned brushless direct current (DC) motors driving two coaxial propellers located above and below the motor enclosure. This configuration has a hollow center tube, allowing for power and communication wires to be run above and below the propulsion section.

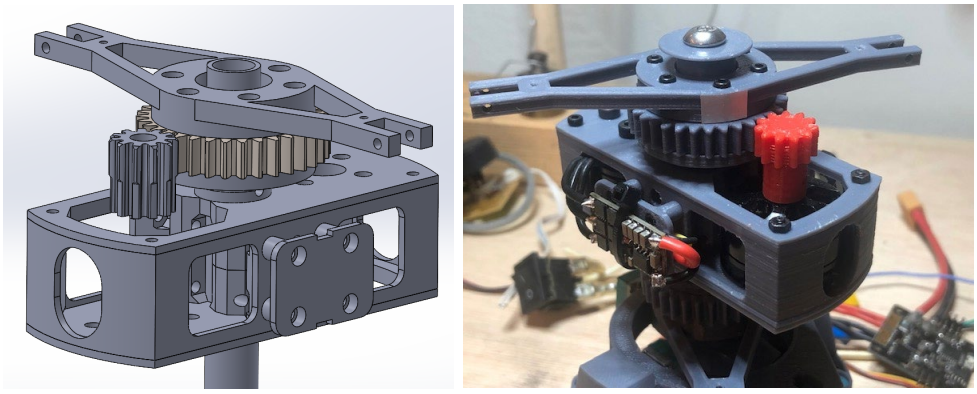


Figure 3.14: CoaxUAV Version 1 Propulsion Section

Testing was conducted by first attaching the vehicle to the top of the test stand. Next calibration weights were used to calibrate the load cells and remove the vehicles weight from the measurements. A plexiglass safety shield was installed between the vehicle and operators, and an ethernet data cable was run around the shield to the operation computer. The method of thrust control evolved from using a simple potentiometer, to using both a radio control (RC) controller and the ArduPilot user interface (UI).

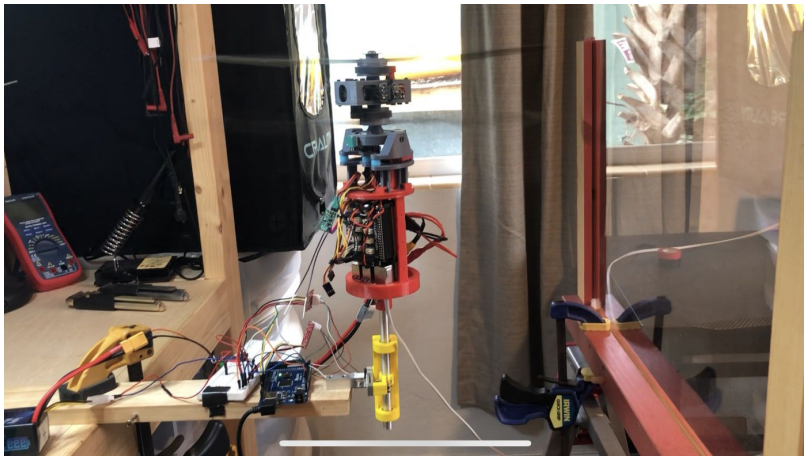


Figure 3.15: CoaxUAV Version 1 Thrust Test

Initially both the pinion and spur gears were designed in SolidWorks then 3D printed. Testing resulted in catastrophic gear failure due to friction and overheating. This failure can be seen in the thrust and torque data as sudden drops in thrust and an

increased torque. Friction between the gears created excessive heat, melting the pinion gear. The spur gear stayed intact due to its larger circumference allowing more time for cooling.

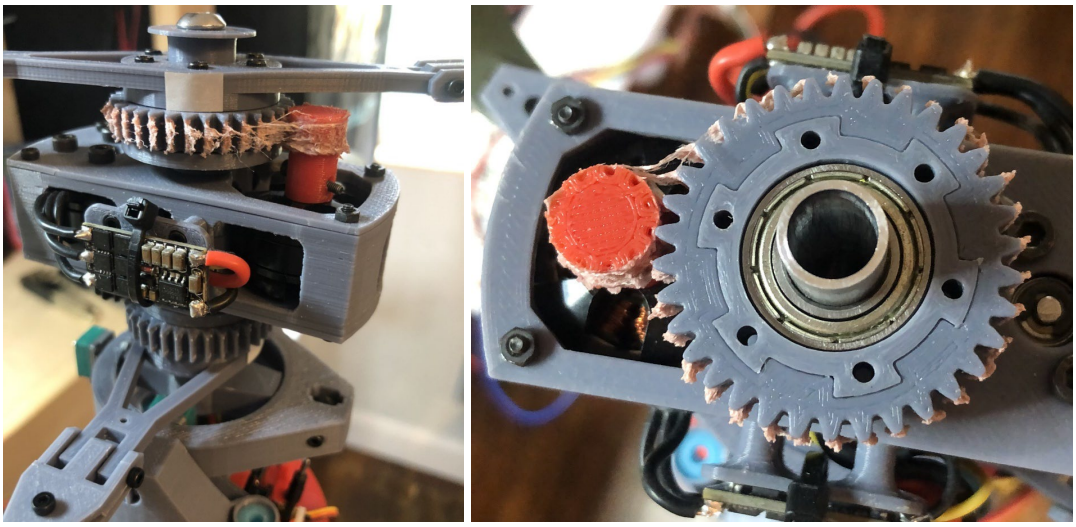


Figure 3.16: CoaxUAV Version 1 Gear Melt

Initial testing resulted in a peak thrust of 8.1 Newtons before the gear melted.

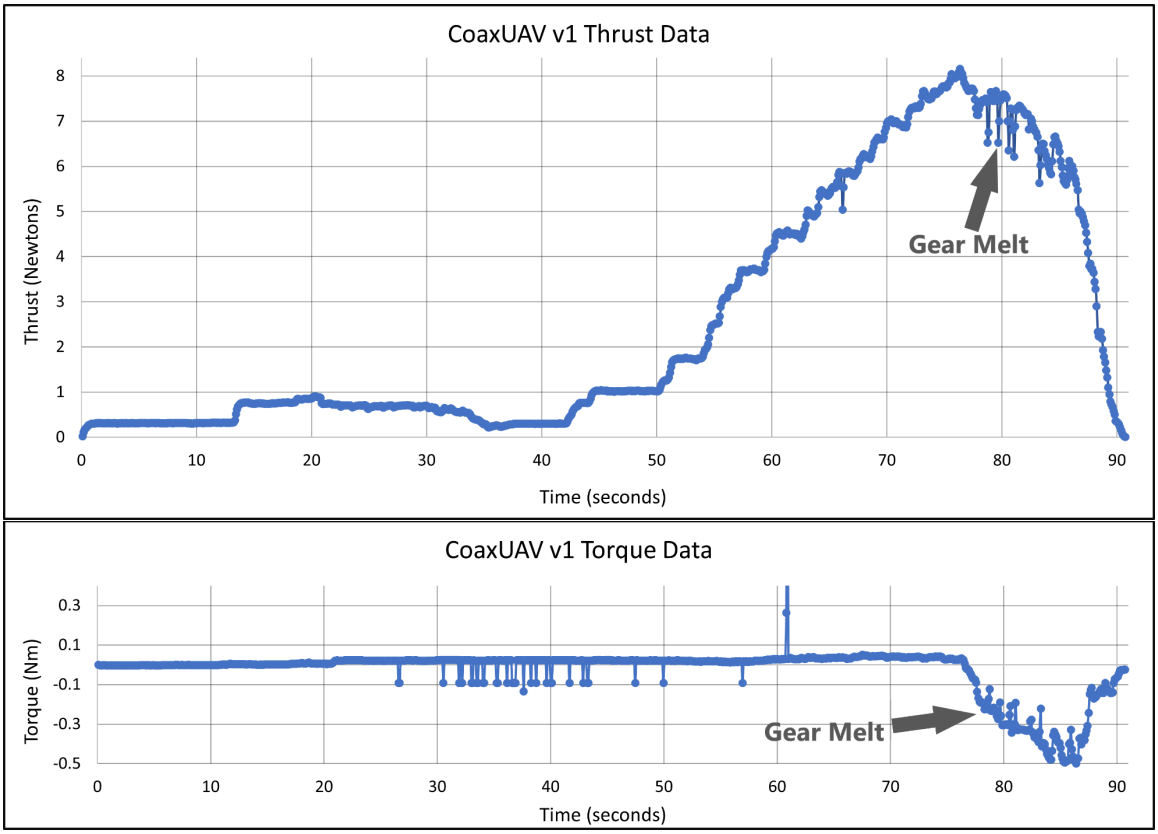


Figure 3.17: CoaxUAV Version 1 Gear Melt Data

The pinion was replaced with a metal gear, and the meshing of metal and PLA plastic gears was tested. The vehicle experienced a high degree of vibration, observed visually and audibly, and the plastic spur gear experienced deformation. Both results were mostly likely due to imperfect gear meshing, which points to inefficiencies and a loss of potential thrust.

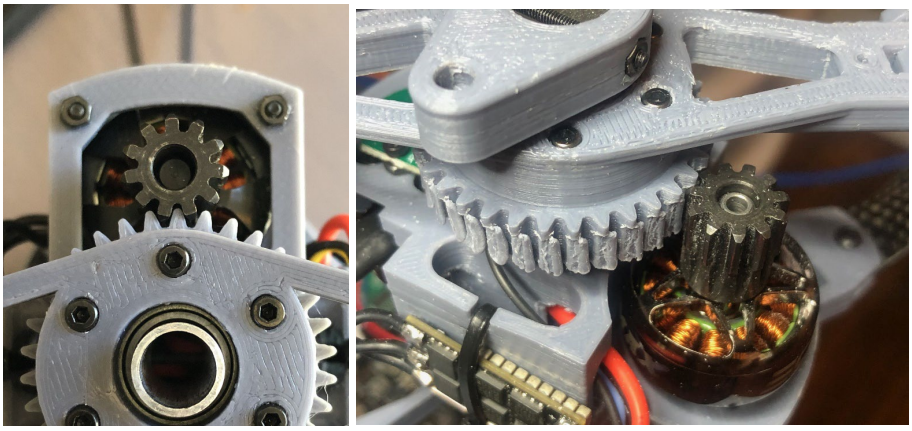


Figure 3.18: CoaxUAV Version 1 Gear Meshing

Due to the high vibrations during testing one of the rotor blades failed, the motor housing began to fail, and the motor housing and gimbal bolts started to loosen. A stress analysis was conducted and found the rotor failure point had experience approximately 0.45 Mpa of static stress. The material datasheet for PLA gives an ultimate tensile strength of around 26 Mpa, but due to the nature of being 3D printed, this value is unreliable. It is concluded that the vibrations drastically increased the momentary stress on the failure point. For calculations, see Appendix B. Loctite was used on future assemblies to stop bolts from loosening.

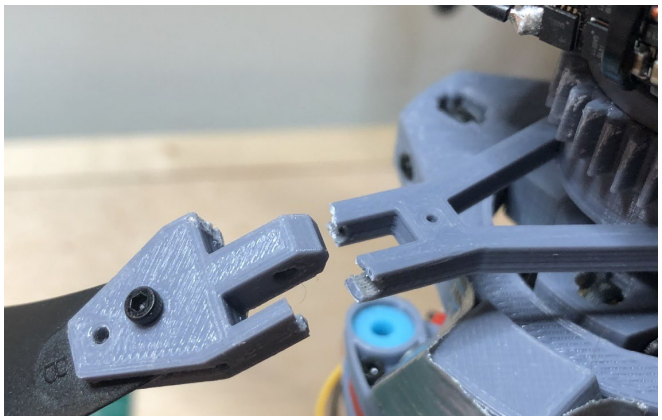


Figure 3.19: Rotor Failure Point

Following results from initial testing, an updated and improved version of the vehicle was developed, called CoaxUAV Version 2. The metal pinion gears were kept, but the spur gears were changed to off-the-shelf acetal plastic gears, designed with the same gear pitch as the metal pinion gears. These spur gears did require some machining to allow the bearings and center shaft to fit. The motor housing was re-designed to add rigidity, eliminating the more fragile components. Ease of installing the motors was also considered. The new motor housing design reduced the number of parts from 18 to 12, as well as increasing the internal conduit diameter from 8mm to 13mm to allow for easier wire installation. The motors and electronic speed controllers (ESCs) were changed for more modern and higher performance models. The maximum motor amperage rating was upgraded from 11.5A to 43A and the ESC current rating was changed from 20A to 35A.

The propellers were also changed from flexible plastic to carbon fiber with a slightly larger pitch. A magnetic encoder was also introduced to the motor housing to record rotor speed onboard the vehicle.

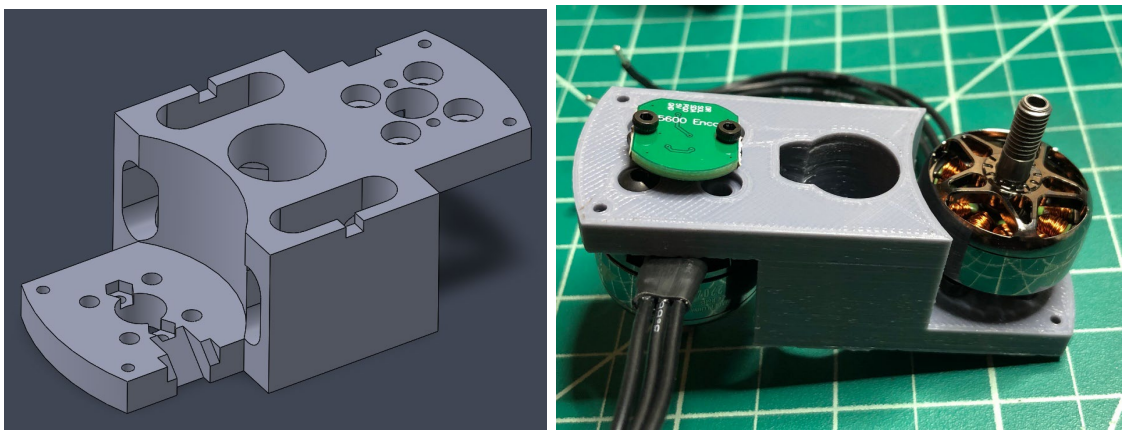


Figure 3.20: Updated Motor Mount with Rotor Speed Encoder

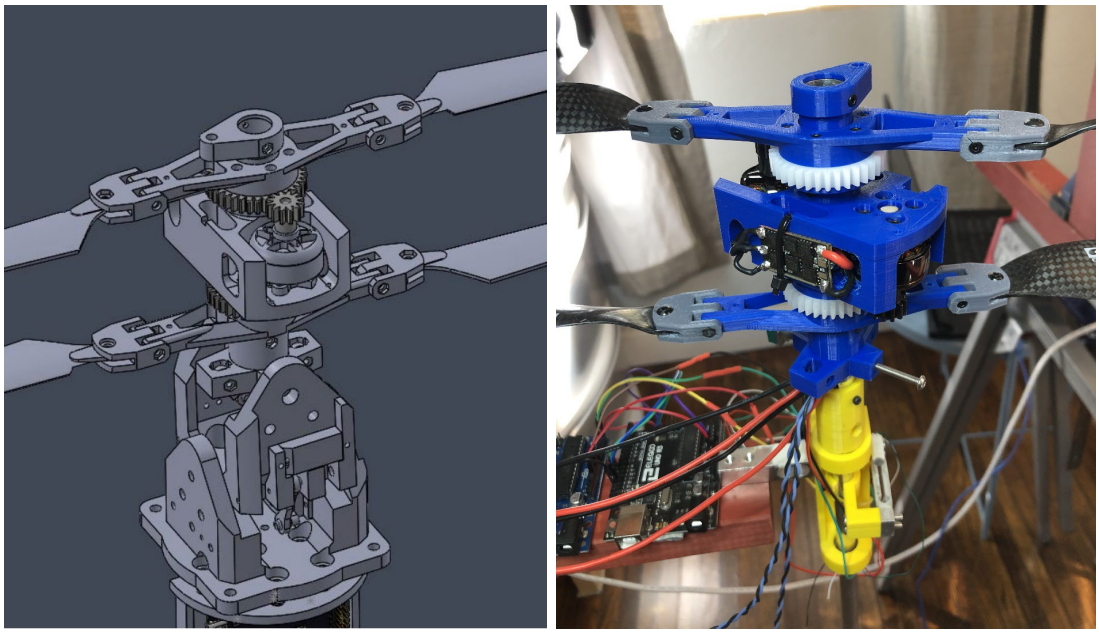


Figure 3.21: CoaxUAV Version 2 CAD and on Test Stand

The CoaxUAV Version 2 testing included gathering total thrust, torque and rotor speed data using an Arduino Uno. The RC input signals, power consumption, amperage

draw, and voltage of the battery were all recorded using the onboard Navigator flight controller running ArduPilot. This data was then used to develop accurate transfer functions relating these system variables.

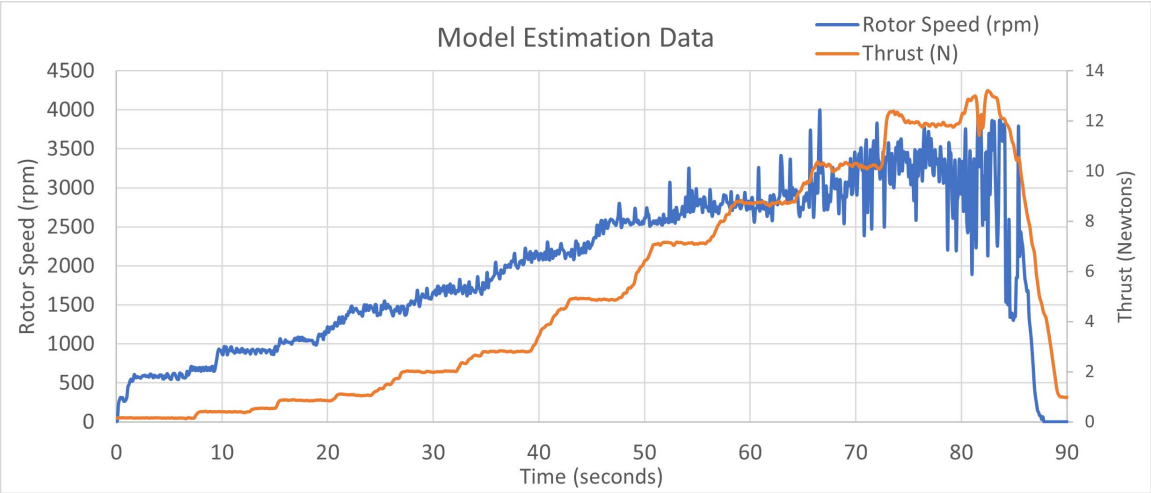


Figure 3.22: Thrust and Rotor Speed Model Estimation Data

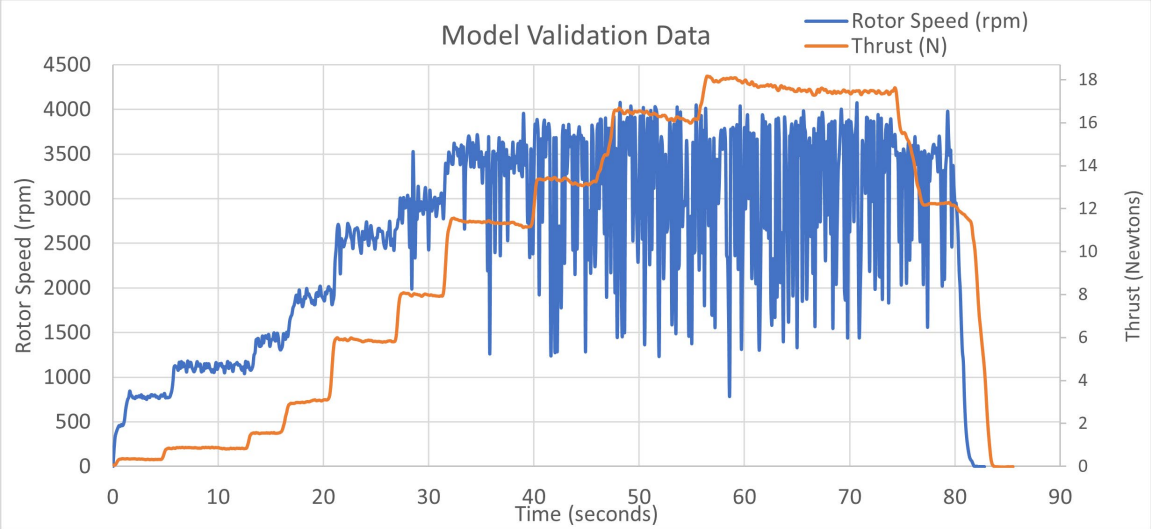


Figure 3.23: Thrust and Rotor Speed Model Validation Data

The physical measurements were taken from the final CoaxUAV prototype for input into the mathematical dynamics model. Measurements are summed up in Table 3.1.

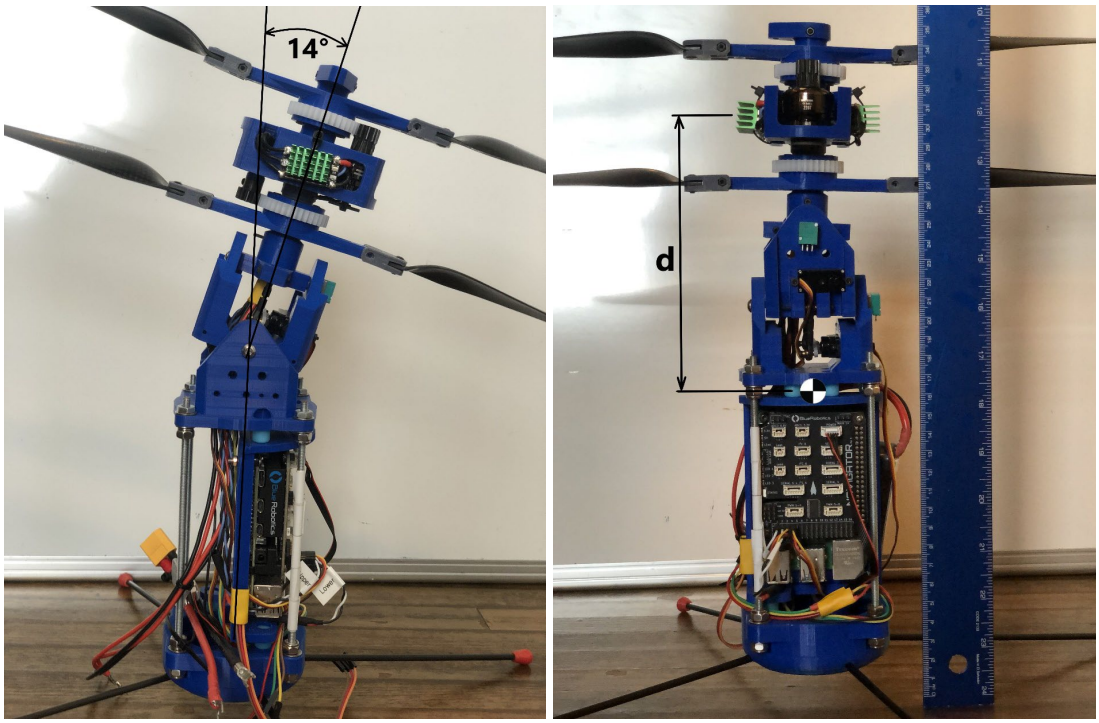


Figure 3.24: CoaxUAV Version 2 Servo Angle and Center of Mass Distance, d

Table 3.1

CoaxUAV v2 Physical Measurements

Description	Value
Total Mass	0.902 kg
Mass Moment of Inertia	0.00856 kg m^2
Physical Dimensions	Height = 0.355m d = 0.14m
Max Servo Angle	14 degrees
Propellers	Diameter: 15.5” Pitch: 5.5”

The mass moment of inertia was measured experimentally using the pendulum method. This involves suspending the vehicle from two lines and inducing a gentle oscillation. By measuring the oscillation period, the moment of inertia can be found. [36]

Flight testing was conducted, confirming the vehicle’s ability to provide the required thrust for flight. These tests also verified the hardware was robust enough to survive ascent, descent, and hovering conditions. During testing the vehicle was constrained to a single DOF along the z axis. This was accomplished by attaching a 40lb

rated nylon line to the table, running it through the center of the vehicle, and securing it to the ceiling. The line constrained translational motion, while the RJ45 data cable provided enough resistance to constrain minimal rotational motion around the z-axis.

Figure 3.25: CoaxUAV v2 Flight Test

Altitude data was recorded using the HC-SR04 ultrasonic range sensor connected to an Arduino.

3.4.3 Data Collection Devices

Two instruments were used for measuring rotor speeds. The AS5600 magnetic rotary position sensor and a handheld laser tachometer. The laser tachometer was used initially as a calibration reference. The AS5600 was mounted to the vehicle, so was used more extensively.

Figure 3.26: AS5600 Rotor Encoder and Laser Tachometer

The constant signal error in the AS5600 sensor was analyzed and compared to the laser tachometer reading. During constant rotor speed, the encoder produced a range of values between 654 and 775 RPM, with an average speed of 713 RPM. The laser tachometer measured a constant 685 RPM. This is a 28 RPM discrepancy, or approximately 4% error. At higher rotor speeds, the discrepancy between the two sensors was similar, which resulted in lowering the error to around 1% at 2500 RPM.

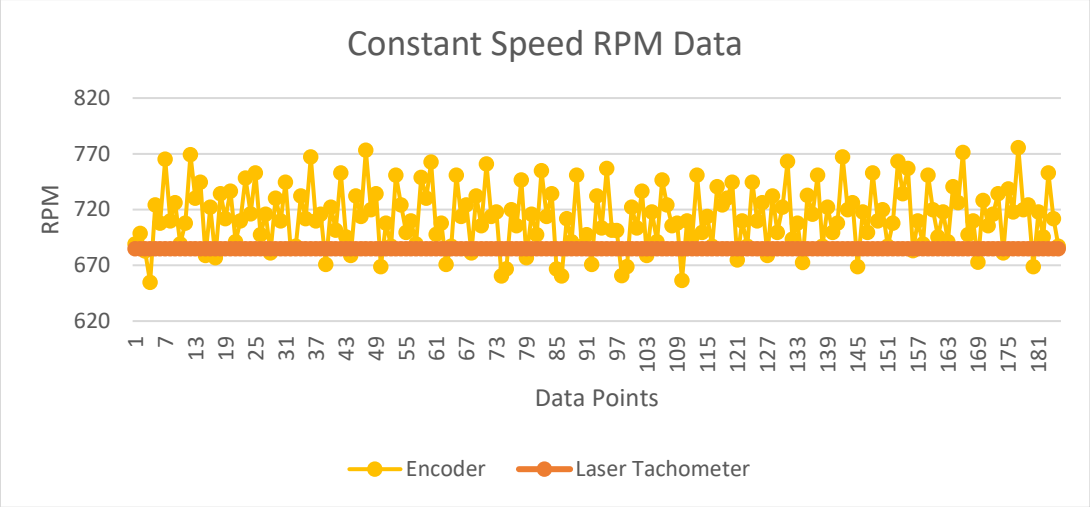


Figure 3.27: Rotor Speed Encoder and Laser Tachometer Data

Unlike the laser tachometer, the AS5600 allowed for data logging over time. While logging rotor speed, aliasing occurs in the data at speeds above 3000 RPM. This is a result of the sampling speed reaching the Nyquist frequency limit. At 3000 RPM, there

is approximately 20 ms per revolution. The Arduino's serial output was at 420Hz (2.4 ms/data point). Because the 2.4 ms serial output is well below 20 ms revolution time, the aliasing mostly likely was occurring between the sensor and the Arduino.

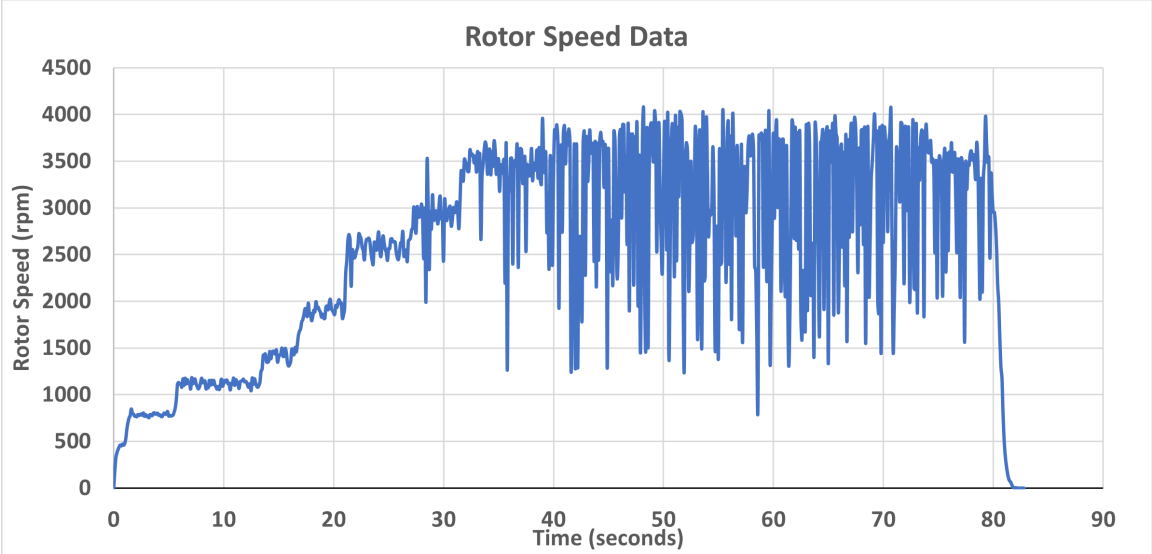


Figure 3.28: RPM Data Aliasing

The thrust test stand was composed of two load cells, attached to conversion modules. One measuring thrust and one measuring torque, with max weight capacities of 10kg and 500g respectively.

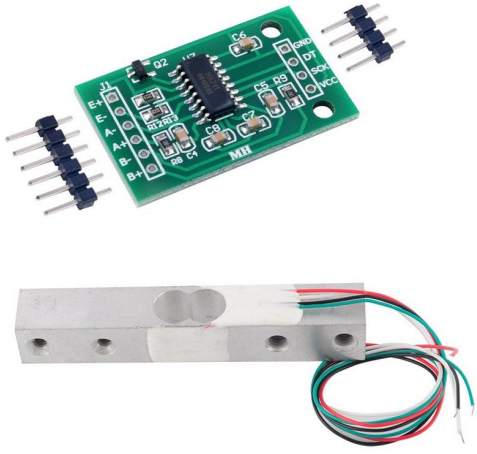


Figure 3.29: Strain Gauge Load Cell

4 Results and Analysis

4.1 Controller Results

The CoaxUAV dynamics model was first tested by manually flying it in a simulation using a USB gaming controller. This involved mapping the input signals to the correct outputs. Next the state feedback controller was implemented. The following controller is used throughout these results:

$$K = \begin{bmatrix} -2.11 & -3.71 & 2.593 & 3.061 & -21.088 & -3.851 \\ -0.007 & -0.016 & 0 & 0 & -0.13 & -0.048 \end{bmatrix} \quad 4.1$$

with system eigenvalues of:

$$\bar{\lambda} = [-1.5 \quad -1.6 \quad -1.7 \quad -1.8 \quad -1.9 \quad -2.0] \quad 4.2$$

The final states of the vehicle were set to 8 meters to the right, 7 meters up.

$$\bar{X}_{final} = \begin{Bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \\ \theta \\ \dot{\theta} \end{Bmatrix} = \begin{Bmatrix} 8 \\ 0 \\ 7 \\ 0 \\ 0 \\ 0 \end{Bmatrix} \quad 4.3$$

The results show the vehicle adjusting the thrust and vector angle and moving to the desired final states:

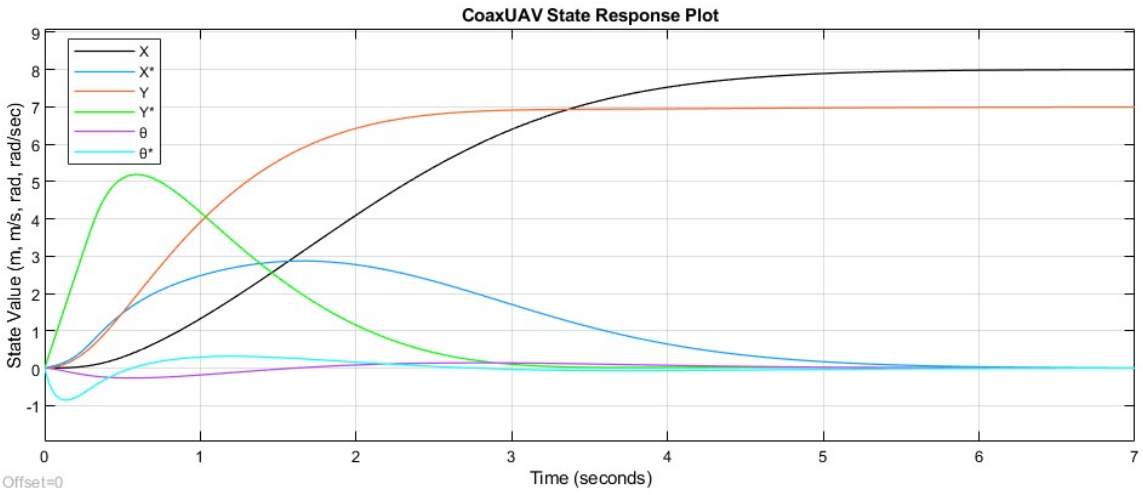


Figure 4.1: State Response of Vehicle in Simulation

Visualizing the physical vehicle over time:

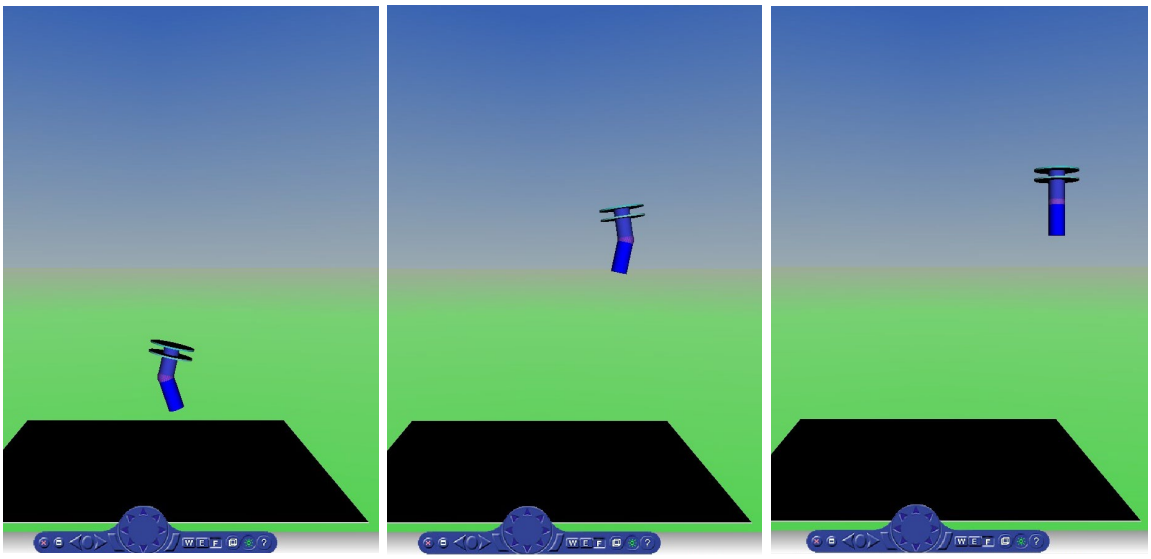


Figure 4.2: Vehicle Simulation at 1sec, 3sec, and 7sec

Translational and rotational disturbance forces were used to test the controller's response. Figure 4.3 shows the vehicle subjected to a constant wind force of 0.9 Newtons and two wind gusts of 14 Newtons. The first gust is angled 45 degrees up and to the left, the second gust is angled 45 degrees down and to the left. The final desired states are x = 8m and y = 7m.

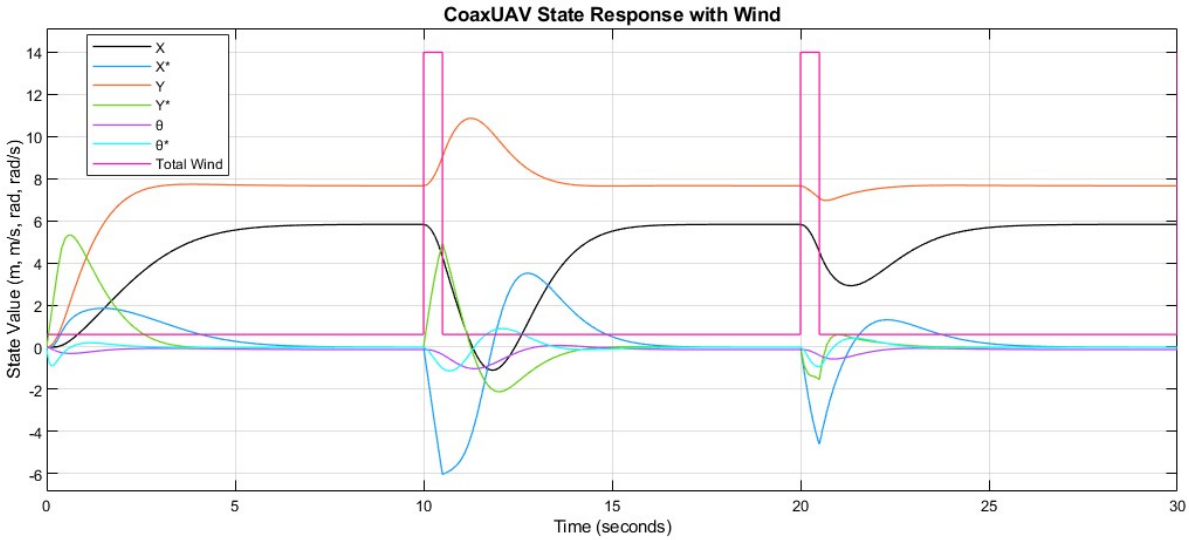


Figure 4.3: State Response of Vehicle with Wind Force

Visualizing the vehicle responding to wind gusts:

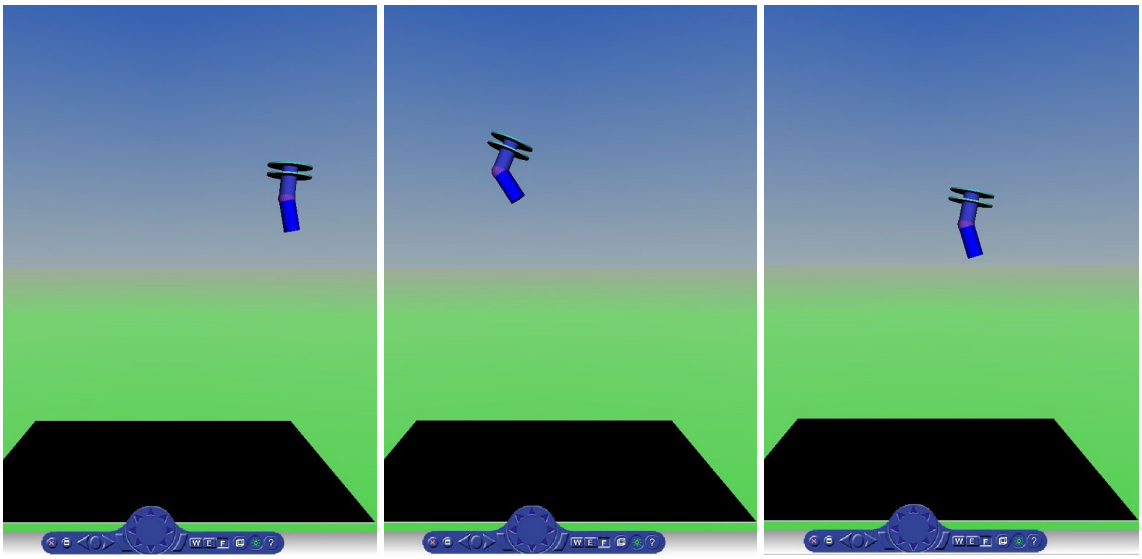


Figure 4.4: Vehicle Simulation at 10sec, 13sec and 23sec

The limits of the controller was determined by applying an x-direction force of varying magnitude and duration. The edge of the controller's ability to stabilize the vehicle was recorded and the parameters were plotted. Figure 4.7 can be repeated for different controller gains and used to qualify the best controller for certain applications. The curve resembles a reciprocal function, where the allowable applied force is very large for very sort times periods. Some examples of this force are gusts of wind or applied energy from physical impacts.

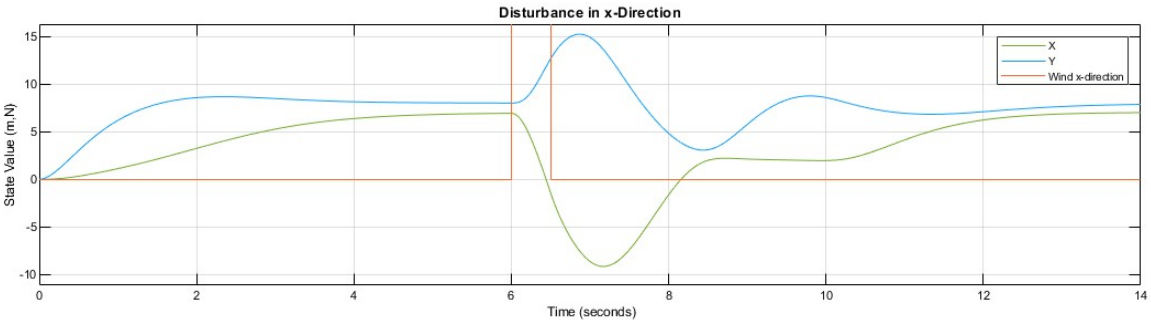


Figure 4.5: Vehicle Successful Recovery from 74 N x-direction Force for 0.5 seconds

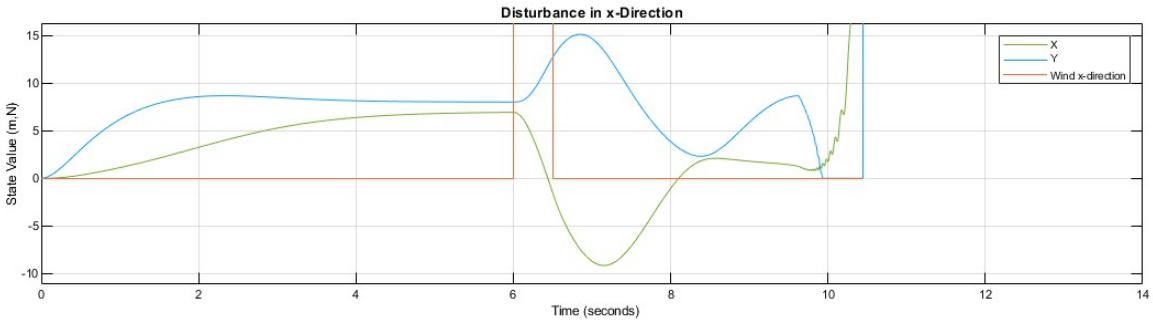


Figure 4.6: Vehicle Unsuccessful Recovery from 75 N x-direction Force for 0.5 seconds

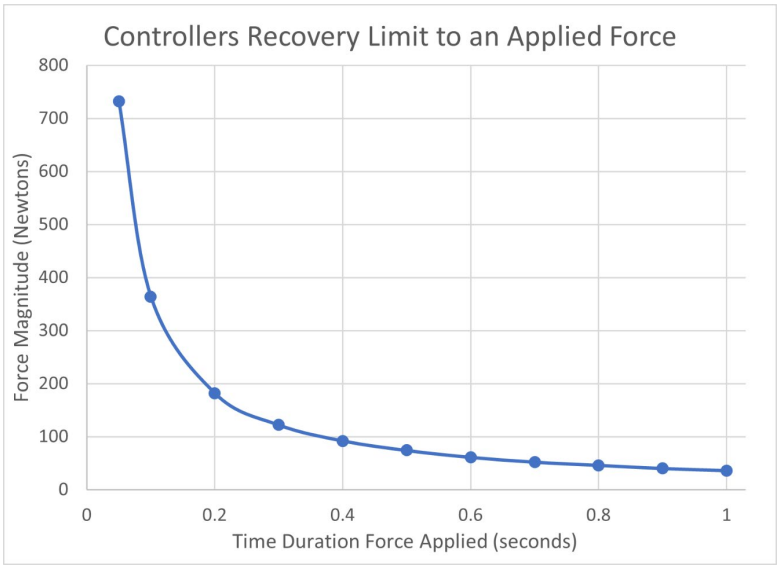


Figure 4.7: Controllers Recovery Limit to an Applied Force in the x-direction

The vehicle simulation was also subjected to torsional disturbance forces and the results were analyzed. The parameters in this case were the torque magnitude in Nm and the amount of time it was applied. The limits of the controller's ability to stabilize the vehicle were plotted, Figure 4.10. This curve also resembles a reciprocal function and for short time periods the controller can handle a large, applied torque. A real application of this force could include circular wind gusts, or the reaction to onboard servos such as gimbaling a camera.

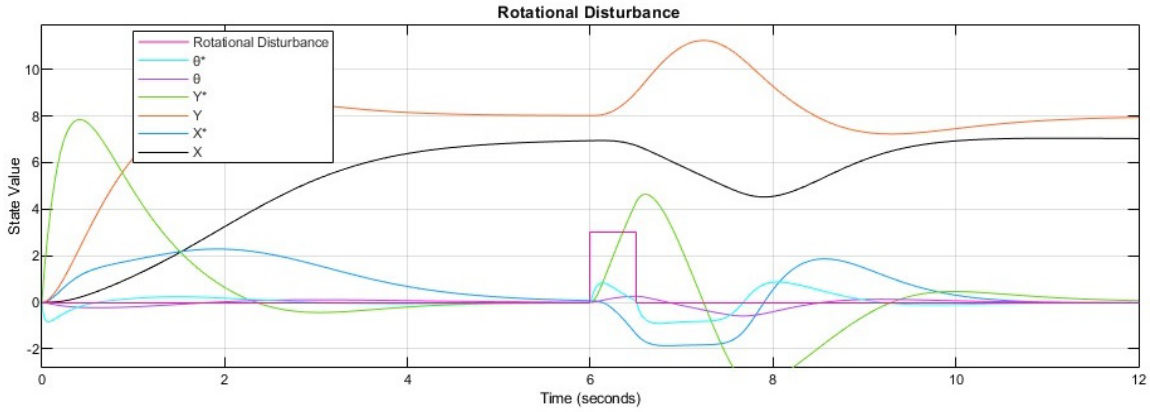


Figure 4.8: Vehicle Successful Recovery from 0.015Nm Torque for 0.3seconds

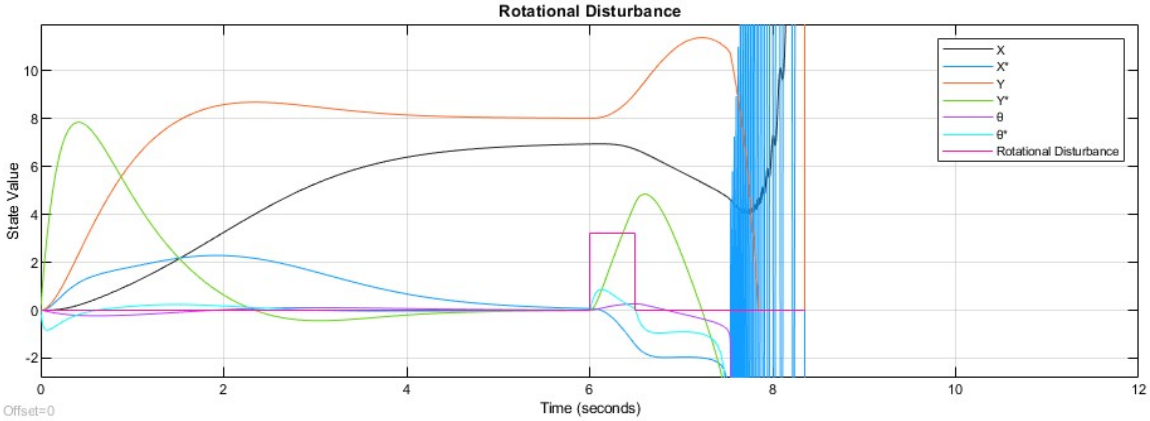


Figure 4.9: Vehicle Unsuccessful Recovery from 0.016Nm Torque for 0.3seconds

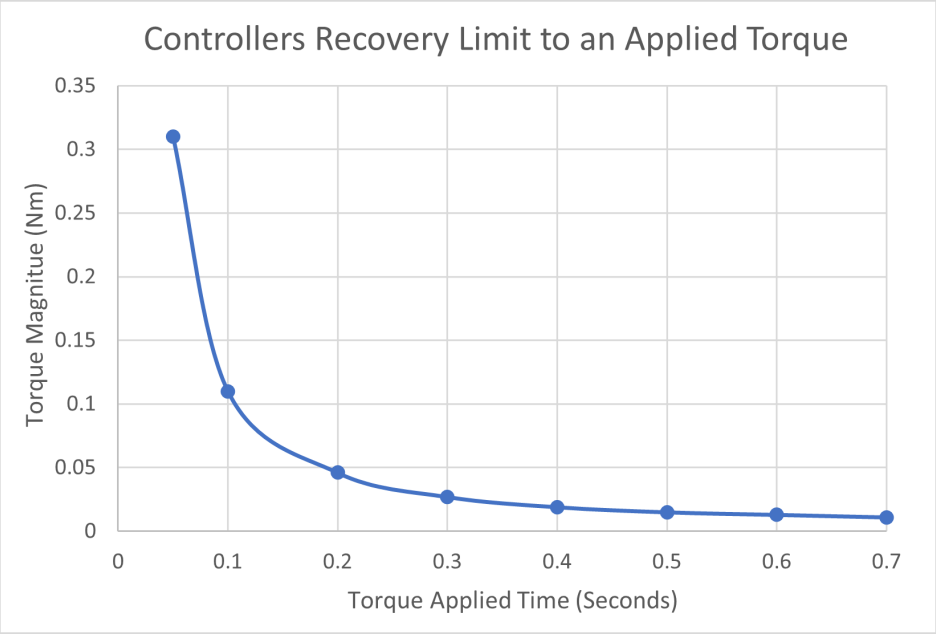


Figure 4.10: Controllers Recovery Limit to an Applied Torque

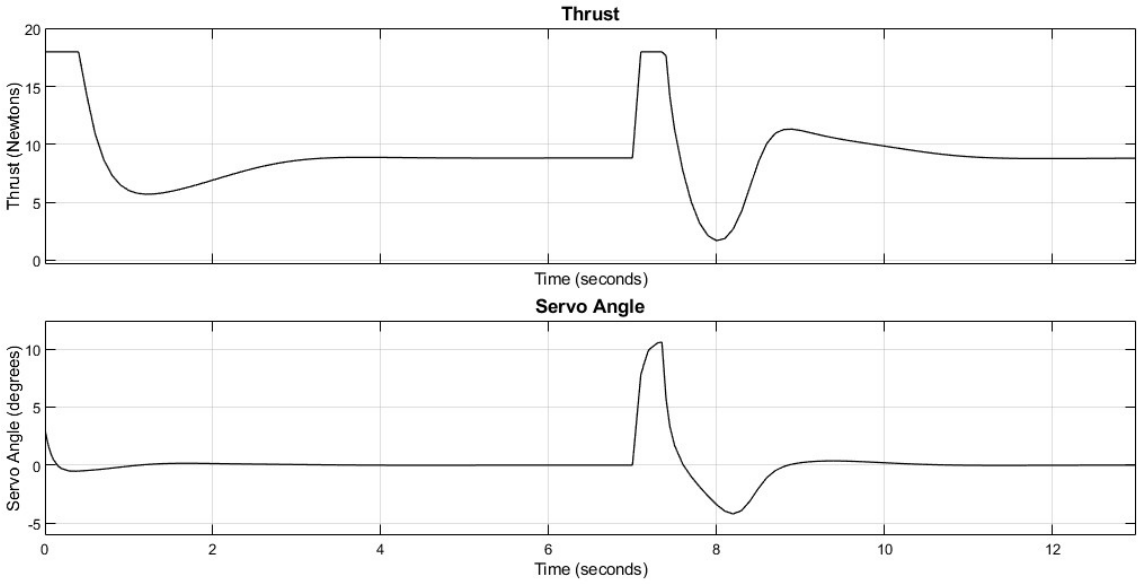


Figure 4.11: System Thrust and Servo Inputs during Successful Recovery from Torque Disturbance

The majority of analysis done in this research was conducted using a single feedback controller gain matrix, Equation 4.1. The matrix had reasonable response characteristics but could be further optimized, depending on the UAV's use case. If

smooth flight is required for optical sensors, or videography, the translational acceleration would need to be minimized, while maximizing the controller's ability to stabilize from disturbances. The hardware design could be incorporated into this optimization cycle as well, where increased thrust capability, adding extra weight, would be considered while designing the controller to achieve rejection of expected disturbances. Further work could use the above methodology to measure the responses to external forces and torques for many different controllers. Plotting these controller gains on the same plot, such as in Figure 4.7 and Figure 4.10, would allow for a specific controller to be selected based on desirable response traits such as stability with high amplitude vs. long duration disturbance.

4.2 Thrust Model and Data

The results from comparing the theoretical disc actuator thrust model and data from hardware testing are shown in Figure 4.12. The single and coaxial propeller models come from Equations 3.19 and 3.21.

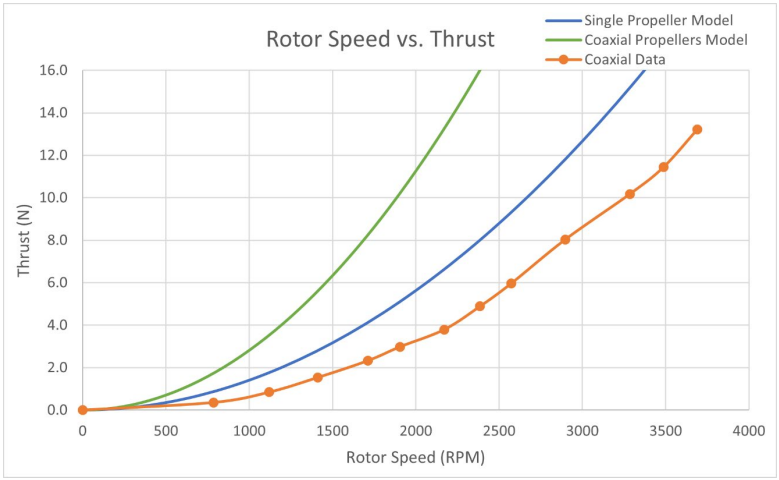


Figure 4.12: Theoretical and Actual Thrust Curves

With the new thrust model, Equation 3.25, the average residual dropped to 0.02 Newtons. Figure 4.13 shows the updated model plotted with the actual thrust data.

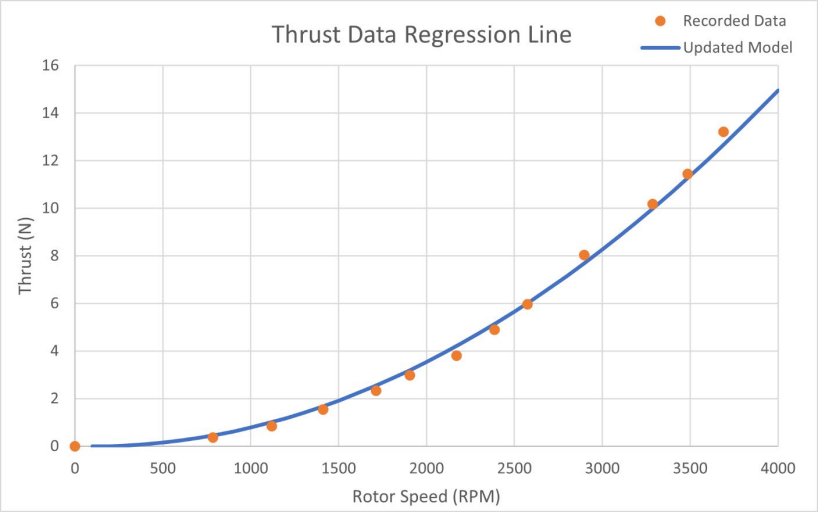


Figure 4.13: Updated Thrust Model and Test Data

The system identification approach is used to find a transfer function that best describes the relationship between rotor speed and thrust, as given by testing data. Both the model estimate data and the validation data are uploaded into the MATLAB System ID toolbox, shown in Figure 3.10.

Figure 4.14 compares the transfer functions outputs, in color, to the model estimate data output in black.

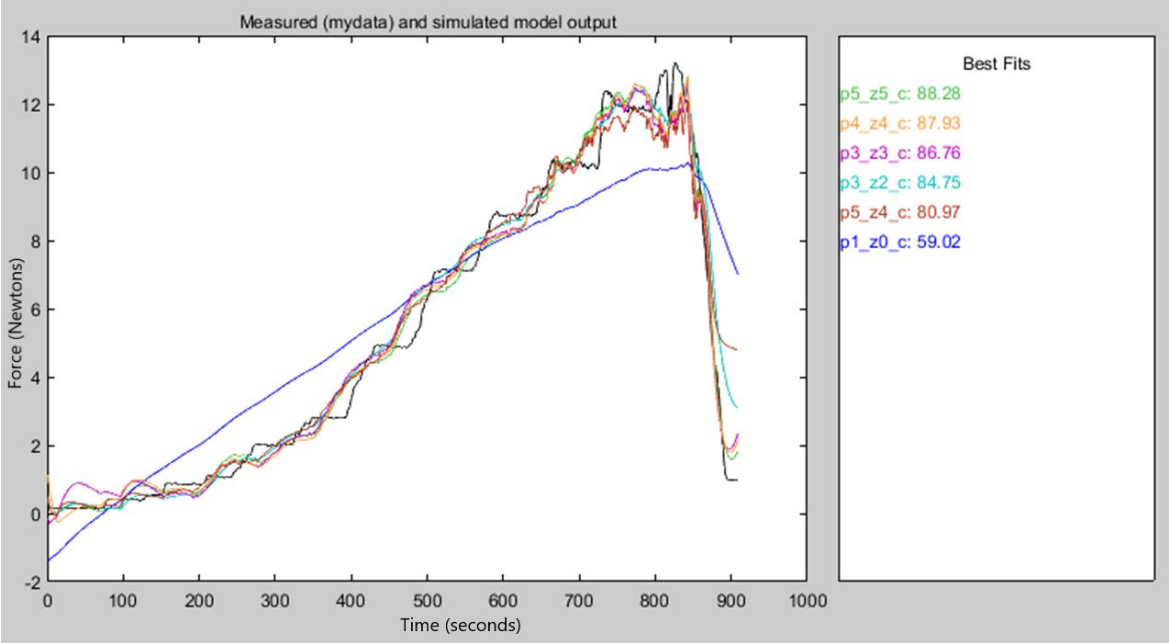


Figure 4.14: Transfer Functions Fit Compared to Model Estimate Data

Comparing these same transfer function outputs to the validation data:

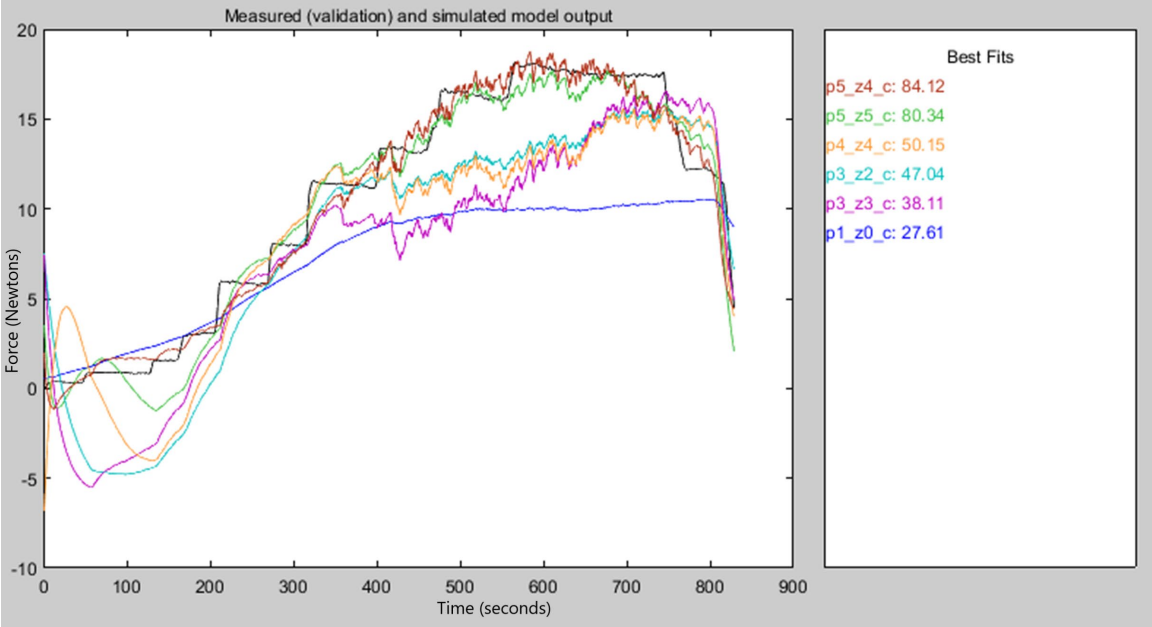


Figure 4.15: Transfer Functions Fit Compared to Validation Data

The transfer function models fits are shown in Table 4.1.

Table 4.1

System Identification Transfer Function Fit

Transfer Function	Estimate Fit	Validation Fit
1 pole 0 zero	59.0 %	27.6 %
1 pole 1 zero	38.2 %	21.0 %
2 pole 2 zero	68.7 %	55.7 %
3 pole 2 zero	84.8 %	47.0 %
3 pole 3 zero	86.8 %	38.1 %
4 pole 4 zero	87.9 %	50.2 %
5 pole 4 zero	81.0 %	84.1 %
5 pole 5 zero	88.3 %	80.3 %
6 pole 6 zero	86.4 %	58.2 %
7 pole 7 zero	62.3 %	68.5 %

The best fit transfer function has 5 poles and 4 zeros:

$$H(s) = \frac{3.668E-4s^4 + 6.495E-6s^3 + 4.147E-8s^2 + 4.3E-10s + 1.7E-12}{s^5 + 0.2031s^4 + 0.0034s^3 + 2.753E-5s^2 + 1.176E-7s + 7.15E-10} \quad 4.4$$

The poles and zeros plot for the best fit transfer function:

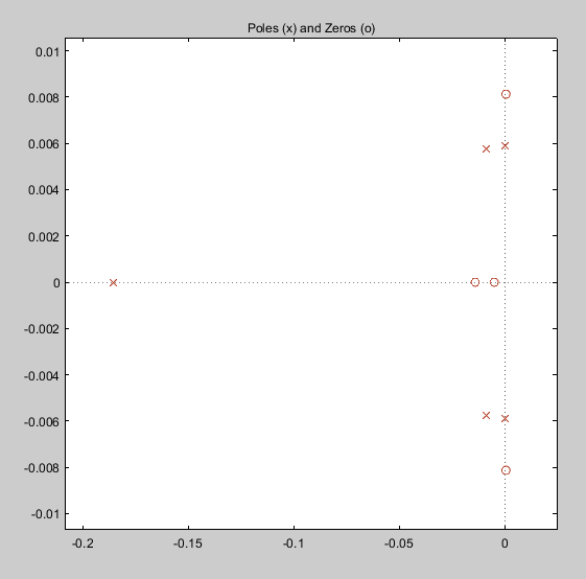


Figure 4.16: Best Fit Transfer Function Poles and Zeros Plot

Isolating the best fit transfer function response on validation data:

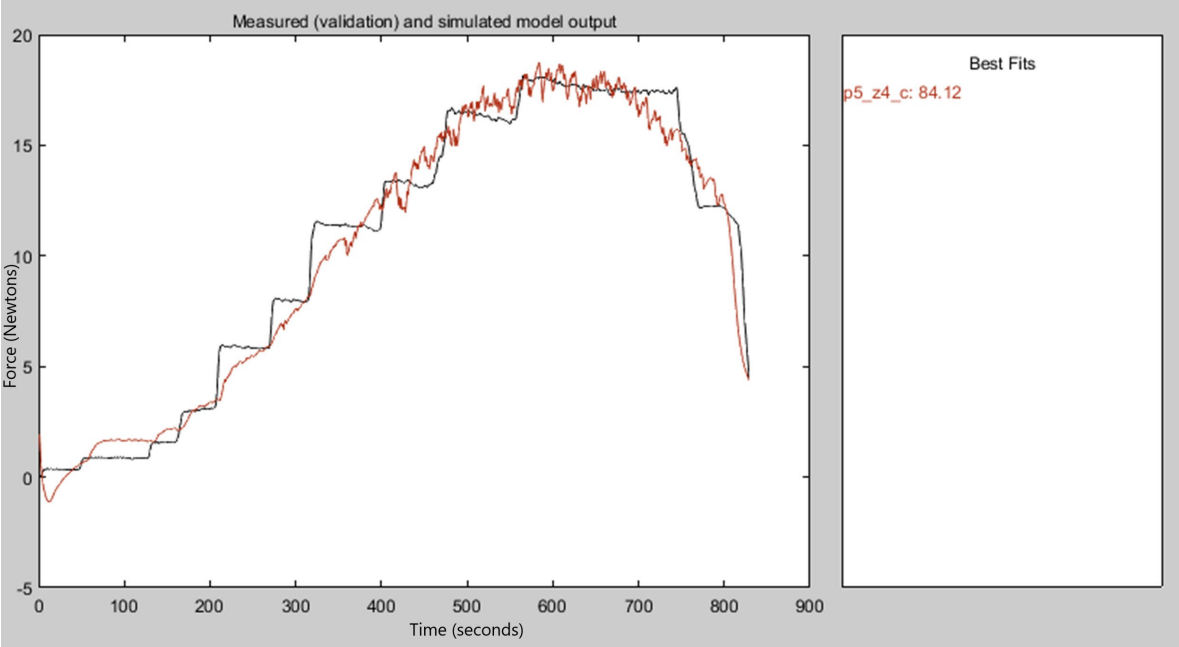


Figure 4.17: Best Fit Transfer Function on Validation Data

4.3 Hardware Design and Testing

A hardware prototype was designed and manufactured using rapid prototyping technology. Fused deposition modeling (FDM) 3D printing was chosen due to its low cost and ease of accessibility. The total cost for the final prototype's hardware components, not including development cost, was $958. This figure could be drastically decreased if moving toward a production unit. Two versions of the CoaxUAV were designed and constructed. Version 2 resulted in a 221% increase in thrust and decreases in vibrations and sound. During testing the vehicle produced a peak of 18.2 Newtons, slightly over twice the force of gravity acting on the vehicle. Comparisons between the CoaxUAV Version 1 and Version 2 are made in Table 4.2 and Table 4.3

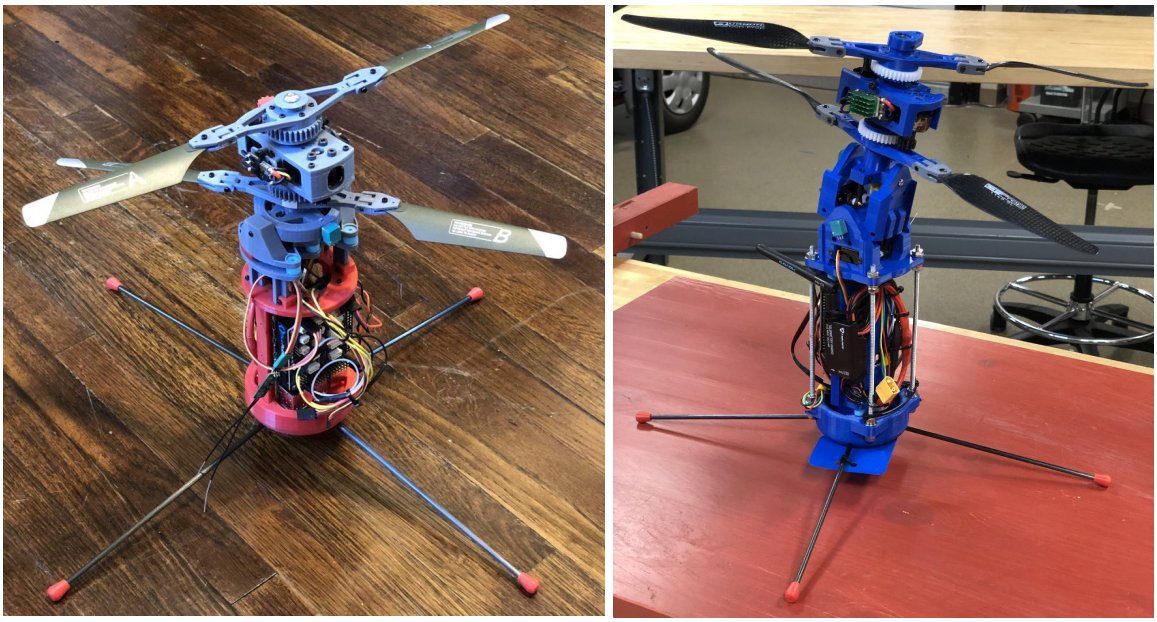


Figure 4.18: CoaxUAV Version 1 and Version 2

Table 4.2

CoaxUAV Version 1 and Version 2 Hardware

Description	Version 1	Version 2
(2) Brushless DC Motors	Hobbymate 2204-2300KV	Emax ECOII 2207-2400KV
(2) ESCs	Favourite Little Bee 20A	AIKON AK32 35A
(1) Flight Controller	Navigator by Blue Robotics	Navigator by Blue Robotics
(1) Battery	Tattu 1300 mAh 3S	Tattu 1300 mAh 3S
(2) Servos	SG90 Micro Servos	EMAX ES08MA II
(1) Radio Receiver	Frsky X4R	FrSky XM+
(4) Propellers	Plastic, unknown	Carbon Fiber, 5.5 in pitch

Table 4.3

CoaxUAV Version 1 and Version 2 Performance

Category	Version 1	Version 2
Max Thrust	8.2 Newtons	18.1 Newtons
Weight	770 grams	902 grams
Thrust to Weight	1.08	2.04
Flight Test	No	Yes

Using the simple cylinder model, the moment of inertia was estimated to be $I_{model} = 0.0075\ kg \cdot m^2$. The measured moment of inertia for the vehicle was $I_{actual} = 0.00856\ kg \cdot m^2$. The accuracy of the model was surprising, considering its simplicity. Flight test altitude data was recorded using an ultrasonic range sensor. Error is introduced into the data due to the sensors wide field of view, encompassing the bottom of the vehicle, the propellers. and the ceiling. A clear trajectory can be seen in the data, representing the bottom of the vehicle's altitude over time.

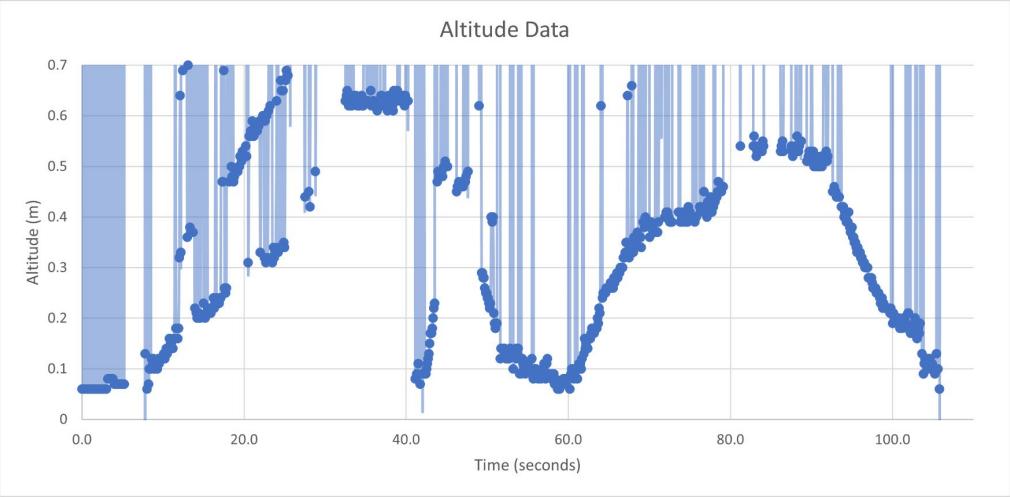


Figure 4.19: Flight Test Altitude Data

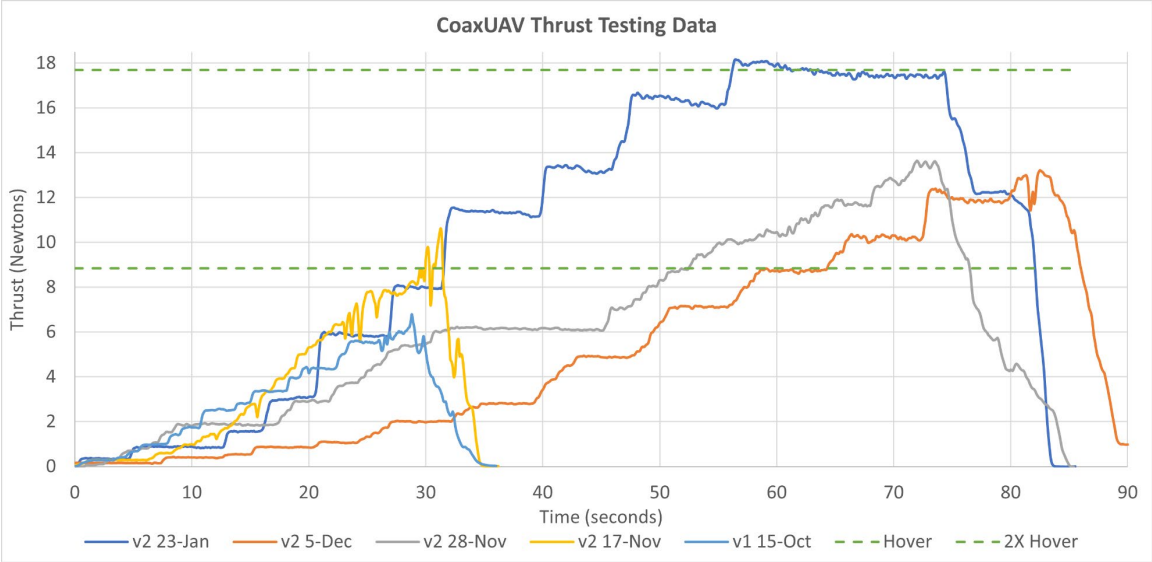


Figure 4.20: CoaxUAV Thrust Testing Data

At peak thrust, the vehicle was consuming 471 watts of energy, 44 Amps at approximately 10.7 Volts.

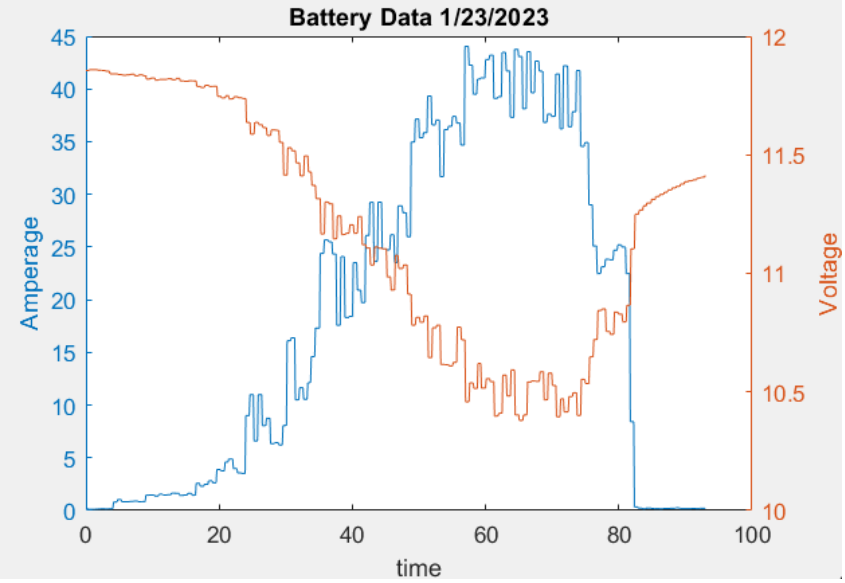


Figure 4.21: CoaxUAV Version 2 Power Consumption

At peak power, the ESC temperatures were in danger of melting the thermoplastic motor housing. To help alleviate this problem, large aluminum heat sinks were attached to the top of each ESC.

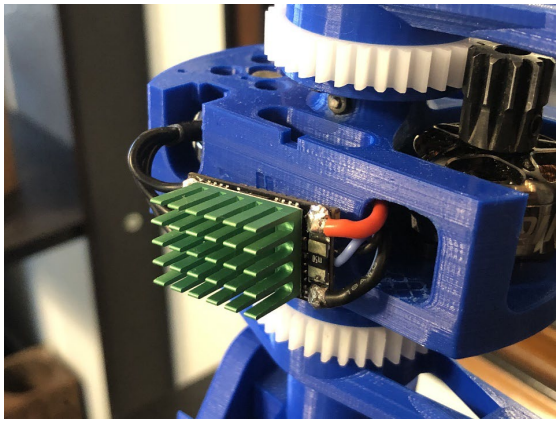


Figure 4.22: ESC Heat Sink

5 Conclusions and Recommendations

5.1 Dynamics Model and Controller

The initial dynamics modeling revealed that the application of a servo torque induces distinct rotation angles in both the upper and lower bodies. To establish their relationship, a rotation matrix was required. This difference in angles arises from the vertical alignment of the two mass centers due to gravity, and the difference in physical geometry of the two bodies. The sign of related angels also had to be considered carefully, especially for controller implementation, as a clockwise servo angle resulted in a counterclockwise gyroscope angle. The non-linear EOM were solved using both MATLAB and Simulink, but Simulink was ultimately chosen so the controllers block diagrams could be visualized.

Initial testing of the CoaxUAV dynamics model was done by manually controlling the simulation with a USB gaming controller. This was extremely helpful in troubleshooting issues with the model, as the system inputs were then known, and the simulation physics could be observed, as well as giving a better understanding of the system as a whole. This process revealed the lack of a ground normal force in the original model, as gravity would pull the vehicle below the inertial reference plane. It also clarified the need to limit the input values to the simulated system. The CoaxUAV has a max thrust of 18N and servo tilt angle of 14 degrees, so the model was constrained to fit these parameters. Future work should adjust these constraints to match the hardware.

Testing of the state feedback controller first involved providing desired setpoints in 2D space and observing the vehicle response. Later the limits of the controller were tested by adding translational and rotational disturbance forces. With no disturbances the controller responded exceptionally well, adjusting the thrust and servo angle to guide the vehicle to the desired final states. With the forces introduced, limits to the controller's authority were found. In hover, a purely horizontal or rotational force was imparted on the vehicle and the state response was recorded. In this study, successful controller authority meant the vehicle didn't touch the ground. This result was dependent on multiple factors, including the vehicles parameters, controller gain K, the magnitude and duration of the force, and the hover altitude. By holding most of these parameters

constant, the magnitude and duration of the forces were plotted for a single controller gain, shown in Figure 4.7 and Figure 4.10. These plots are a valuable result, as they show the maximum disturbance forces that the current controller gain can compensate for. This research could be further expanded upon by running the same tests on other controller gains and comparing. This could be taken even further by relating the system's response to changing specific eigenvalues of the system.

This research could hold implications in stabilizing drones in windy environments, or during physical bombardments, such as rain. By mapping out responses for different controller gains, an environment specific controller can be quickly selected and implemented.

5.2 Thrust Model

Reviewing the thrust model and data, the actual vehicle thrust values fall below both the single and coaxial theoretical model values. This was an expected outcome as the model had idealized performance assumptions that fall short of reality. Bernoulli's energy Equation, 2.4, doesn't account for the energy loss due to turbulence, and assumes there is no interaction across the streamline boundary. While in fact the coaxial system generates substantial fluid turbulence, especially regarding rotor interaction and wingtip vortices. Equation 3.21 assumes both propellers generate the same amount of thrust, which would require more testing to validate. Finally, Equation 2.5 assumes the fluids free stream and exit densities are equal. With these assumptions stacking up, it's not surprising that the actual thrust performance is substantially lower than the idealized model.

The revised model accurately predicts thrust within the rotor speed range of 0 to 4000 rpm. According to the model's predictions, the vehicle's maximum thrust of 18 Newtons aligns with a rotor speed of nearly 4400 rpm, although due to encoder data aliasing the actual rotor speed could not be accurately recorded above 3000 rpm. This indicates that approximately 90% of the model's operational range has been validated, including hover scenarios at approximately 3000 rpm. Future analysis of this updated model could include testing higher rotor speeds, as well as realizing the effects of varying

model parameters. Comparisons should be made between the updated model and hardware testing data while changing the disc area, blade pitch, and fluid density values.

While using the system identification approach, Figure 4.14 and Figure 4.15 show the importance of using validation data in finding the most accurate system representation. Figure 4.14 shows that five of the transfer function outputs fit the model estimate data within 80%. When comparing their outputs to the validation data in Figure 4.15, only two of these models were above an 80% fit. This result is due to overfitting of certain models to the estimate data.

The best transfer function resulted in 84.12% fit. This is an impressive result, considering the aliasing noise in the rotor speed data. At rotor speeds above 3000 rpm, the data recording rate fell below the Nyquist limit, resulting in inaccurate and sporadic data, see Figure 3.23: . Future improvements in this model would involve taking new thrust data at a sampling frequency that will avoid aliasing.

Figure 4.16 shows the poles and zeros plot for this transfer function. All the poles are negative, meaning the system is stable. The makes sense, because there are no input rotor speeds which will result in the thrust values increasing unbounded.

Using this method of system identification, many relationships within this system can be modeled using a transfer function. Other potentially useful models that can be developed using this technique include RC input signal to thrust output, and rotor speed to power draw. With a flying prototype, flight acceleration data could be recorded, and system identification could be used to develop transfer functions relating controller signals to vehicle motion in each DOF. This approach reduces the complexity of motion analysis as each DOF is considered an independent SISO system. These models could be used to increase the accuracy of the theoretical EOM that were previously developed. With better models, the controller could also be improved.

5.3 Hardware Prototype

The hardware prototype was successfully flown, while constraining it to just vertical motion. This result proved that this coaxial UAV design could be manufactured using mostly 3D printed material and provide the required thrust for flight. The next large milestone for the hardware prototype would be unconstrained, controlled flight. Steps to

this goal would involve using the ArduCopter coaxial flight software to attempt RC and waypoint flight. The state controller developed in this paper could also be validated by writing it in C++ and implementing directly onto the flight controller hardware for autonomous flight testing.

References

[1] S. Jayanthi, H. Shaheen, U. Balashivudu and M. S. Rani, "Evolution and significance of unmanned aerial vehicles," *Unmanned Aerial Vehicle Cellular Communications,* pp. 287-311, 2022.

[2] M. C. Quilter and V. J. Anderson, "Low Altitude/Large Scale Aerial Photographs: A Tool For Range And Resource Managers," *Rangelands Archives 22.2,* pp. 13-17, 2000.

[3] J. R. Soddell, K. McGuffie and G. J. Holland, "Intercomparison of atmospheric soundings from the aerosonde and radiosonde," *Journal of Applied Meteorology 43.9,* pp. 1260-1269, 2004.

[4] T. Watai, . T. Machida, N. Ishizaki and G. Inoue, "A lightweight observation system for atmospheric carbon dioxide concentration using a small unmanned aerial vehicle," *Journal of Atmospheric and Oceanic Technology,* vol. 23, no. 5, pp. 700-710, 2006.

[5] M. Williams, D. I. Jones and G. K. Earp, "Obstacle avoidance during aerial inspection of power lines," *Aircraft Engineering and Aerospace Technology 73.5,* pp. 472-479, 2001.

[6] D. Hausamann, W. Zirnig, G. Schreier and P. Strobl, "Monitoring of gas pipelines–a civil UAV application," *Aircraft Engineering and Aerospace Technology 77.5,* pp. 352-360, 2005.

[7] K. C. Aleksander, "Military use of unmanned aerial vehicles–a historical study," *Safety & Defense 4,* pp. 17-21, 2018.

[8] H. Denton, M. Benedict and H. Kang, "Design, development, and flight testing of a tube-launched coaxial-rotor based micro air vehicle," *International Journal of Micro Air Vehicles, 14,* 2022.

[9] E. Kessler, "Military Factory," [Online]. Available: https://www.militaryfactory.com/aircraft/detail.php?aircraft_id=403. [Accessed 26 July 2023].

[10] G. Anderson, "dvidshub," [Online]. Available: https://www.dvidshub.net/image/1219093/us-army-ch-47-chinook-helicopter-flyby#.V0jiN_9f2M8. [Accessed 26 July 2023].

[11] G. Goebel, "flickr," [Online]. Available: https://www.flickr.com/photos/37467370@N08/albums/72157669185119053. [Accessed 26 July 2023].

[12] C. P. Coleman, "A Survey of Theoretical and Experimental Coaxial Rotor Aerodynamic Research," No. NASA-TP-3675, 1997.

[13] K. e. Volkov, Flight Physics: Models, Techniques and Technologies, Rijeka, Croatia: InTech, 2018.

[14] V. H. Dominguez, O. Garcia-Salazar, L. Amezquita-Brooks, L. A. Reyes-Osorio, C. Santana-Delgado and E. G. Rojo-Rodriguez, "Micro Coaxial Drone: Flight Dynamics, Simulation and Ground Testing," *Aerospace,* vol. 9.5, p. 245, 2022.

[15] M. R. Mokhtari, B. Cherki and A. C. Braham, "Disturbance observer based hierarchical control of coaxial-rotor UAV," *ISA Transactions,* vol. 67, pp. 466-475, 2017.

[16] N. C. Joel, H. Djalo and K. J. Aurelien , "Robust control of UAV coaxial rotor by using exact feedback linearization and PI-observer," *International Journal of Dynamics and Control 7.1,* pp. 201-208, 2019.

[17] T. Khamvilai, J. B. Mains, M. Z. Miller and E. M. Feron, "Trajectory Control of a Swashplate-less Coaxial Helicopter Using Nonlinear Techniques," in *2019 IEEE Aerospace Conference*, 2019.

[18] H. W. Kim and R. E. Brown, "A comparison of coaxial and conventional rotor performance," *Journal of the American Helicopter Society 55.1,* 2010.

[19] A. Zulu and S. John, "A Review of Control Algorithms for Autonomous Quadrotors," *Open Journal of Applied Sciences,* vol. 4, pp. 547-556, 2014.

[20] O. Tatale, N. Anekar, S. Phatak and S. Sarkale , "Quadcopter: Design, Construction and Testing," *International Journal for Research in Engineering Application & Management (IJREAM),* vol. 4, no. 2454-9150, 2018.

[21] C. Venkatesan, Fundamentals of helicopter dynamics, Boca Raton, FL: CRC Press, 2014.

[22] C. Bermes, S. Leutenegger, S. Bouabdallah, D. Schafroth and R. Siegwart, "New design of the steering mechanism for a mini coaxial helicopter," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008.

[23] O. Garcia, A. Sanchez, K. C. Wong and R. Lozano, "Modeling and control of a vectored-thrust coaxial UAV," in *2009 European Control Conference (ECC)*, 2009.

[24] H. Schaub and J. L. Junkins, Analytical mechanics of space systems, AIAA, 2018.

[25] A. Kumar, "Degree of Freedom in Statistics: Meaning & Examples," 14 Janurary 2022. [Online]. Available: https://vitalflux.com/degree-of-freedom-in-statistics-meaning-examples/.

[26] N. S. Nise, Control systems engineering, John Wiley & Sons, 2020.

[27] S. L. Brunton and J. N. Kutz, Data-driven science and engineering: Machine learning, dynamical systems, and control, Cambridge University Press, 2019.

[28] Z.-S. Hou and Z. Wang, "From model-based control to data-driven control: Survey, classification and perspective," *Information Sciences, 235,* pp. 3-35, 2013.

[29] R. P. Borase, . D. K. Maghade, S. Y. Sondkar and S. N. Pawar, "A review of PID control, tuning methods and applications," *International Journal of Dynamics and Control, 9,* pp. 818-827, 2021.

[30] P. M. Sforza, Theory of aerospace propulsion (Second edition), Butterworth-Heinemann, 2017.

[31] L. Ljung , "System Identification," in *Signal Analysis and Prediction*, Boston, MA, Birkhäuser Boston, 1998, pp. 163-173.

[32] L. Ljung, "System Identification Toolbox: For use with MATLAB," 2015.

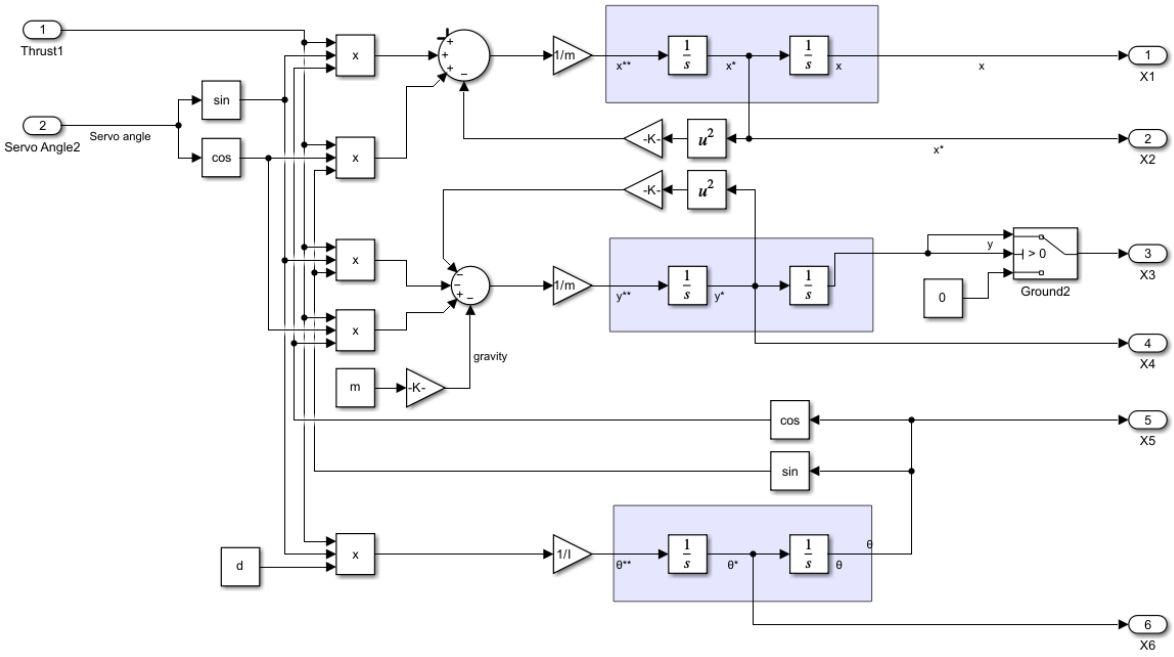
[33] MathWorks, "armax: Estimate parameters of ARMAX, ARIMAX, ARMA, or ARIMA model using time-domain data," [Online]. Available: https://www.mathworks.com/help/ident/ref/armax.html.

[34] Ultimaker, "What is g-code?," 8 November 2022. [Online]. Available: https://support.makerbot.com/s/article/1667337572861. [Accessed 4 April 2023].

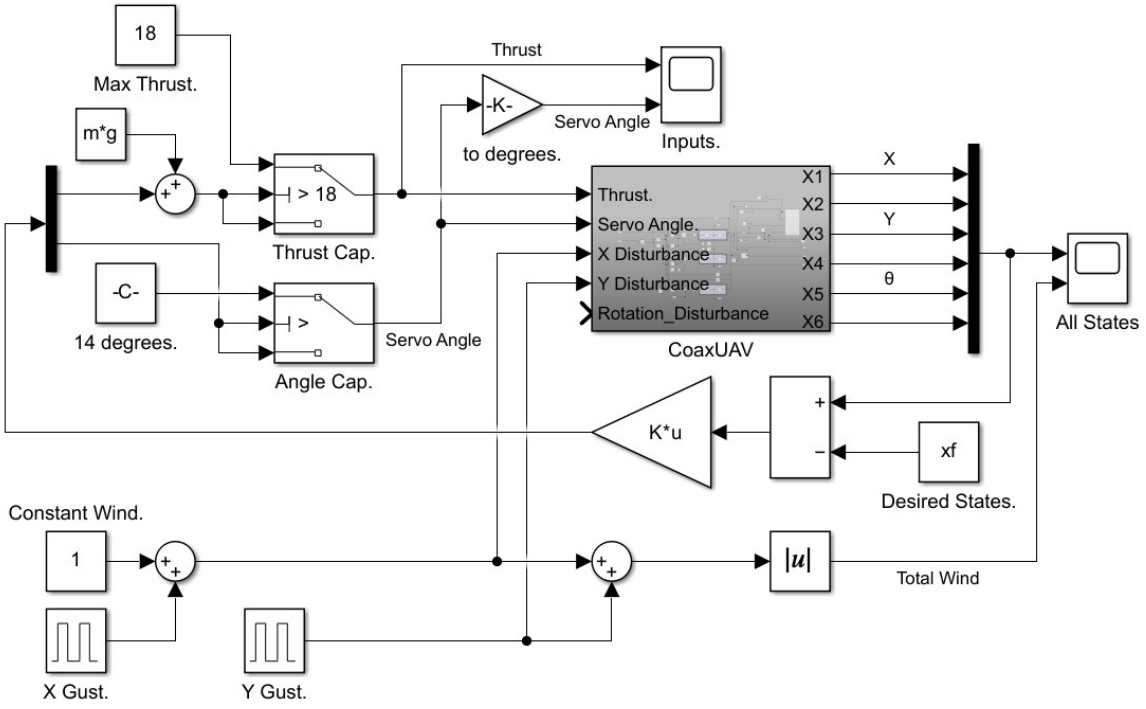
[35] J. Hu, "DIY Drones," 29 November 2018. [Online]. Available: https://diydrones.com/profiles/blogs/tdrone-open-source-coaxial-drone. [Accessed 14 June 2022].

[36] H. A. Soule and M. P. Miller, "The experimental determination of the moments of inertia of airplanes," *No. NACA-TR-467,* 1934.

Appendix A
Simulink Models



Coaxial UAV Simulink Non-Linear EOM



State Feedback Controller with Input Limitations and Wind Disturbances

Appendix B
Rotor Failure Stress Analysis

The rotor conditions at failure:
Rotor speed = 2,500 rpm, velocity = 12 m/s
Mass of the tip plus blade = 4 grams (0.004 kg)
Calculating the centripetal force:

$$F_{centripital} = \frac{mv^2}{r} = \frac{(0.004\ kg)(12\ m/s)^2}{0.046\ m} = 12.5\ N$$

The cross-sectional area that failed:

$$A_{cross-section} = 2.8\text{E} - 5\ m^2$$

The failure stress is then calculated:

$$\sigma_{failure} = \frac{F}{A} = \frac{12.5\ N}{2.8x10^{-5}\ m^2} = 446\ kPa = 0.45\ MPa$$

Appendix C
MATLAB Script

Simulink Model MATLAB Script

```
m1 = 0.3; % upper mass (kg)
m2 = 0.6; % lower mass (kg)
L1 = 0.05; % (m)
L3 = 0.1; % (m)
g = 9.81; % gravitational constant
m = m1 + m2;
d = 0.14; % Torque distance. Distance from center of mass to thrust torque.
radius = 0.03;
height = 0.32; % drone height in meters
d1 = (m2/m)*(L1+L3); % distance from m1 to drone center of mass
d2 = (m1/m)*(L1+L3); % distance from m2 to drone center of mass
di = radius*2; % drone diameter in meters
I = 0.00856; % Mass Moment of Inertia - Experimental Data
Fg1 = m1*g; % upper mass gravitational force (N)
Fg2 = m2*g; % lower mass gravitational force (N)

A = [0 1 0 0 0 0;
    0 0 0 0 g 0;
    0 0 0 1 0 0;
    0 0 0 0 0 0;
    0 0 0 0 0 1;
    0 0 0 0 0 0];
B = [0 0;
    0 -g;
    0 0;
    1/m 0;
    0 0;
    0 -(g*m*d)/I];
C = eye(6);
D = [0 0;
    0 0;
    0 0;
    0 0;
    0 0;
    0 0];

eig(A);
rank(ctrb(A,B))

eigVals = [-1.5 -1.6 -1.7 -1.8 -1.9 -2.0];
K_place = place(A,B,eigVals);
xf = [-8; 0; 7; 0; 0; 0]; % desired end conditions
```