Design and Development of an Autonomous Paramotor UAV

by

Brett Jensen Fiedler

A Dissertation Presented in Partial Fulfilment
of the Requirements for the Degree
Doctor of Philosophy

Approved April 2022 by the
Graduate Supervisory Committee:

Sangram Redkar, Chair
Thomas Sugar
Amar Phatak

ARIZONA STATE UNIVERSITY

May 2022

ABSTRACT

This document is the culmination of research into small unmanned Powered Parachute aerial vehicles. This dissertation serves to provide designers of small systems with an approach to developing a Powered Parachute Unmanned Aerial Vehicle system, guiding them through the basic assumptions, dynamics, and control method. In addition, this dissertation aims to generate a reliable and generalized framework of dynamic design and control methods for autonomous Powered Parachute aircraft. The simulation methods in this paper assist in developing a consistent and robust unmanned system for applying Powered Parachutes as an alternative to multirotor or fixed-wing aircraft.

The first chapter serves as a primer on the historical applications of small Unmanned Systems and Powered Parachutes and gives an overview of the requirements for building an autonomous Powered Parachutes; the information within this chapter provides justification background for the second chapter on Powered Parachute dynamics. In the dynamics chapter, equations of motion are derived using engineering first principles. This chapter also discusses alternative methods of improving the control and robustness of the Powered Parachute airframe. The dynamics model is used in all further chapters to develop a generalized control system to operate such a model autonomously. Chapter three of this document focuses on developing simulations from the dynamics described in the previous chapter, laying the groundwork for guidance, navigation, and control algorithms ahead. Chapters four and onwards refine the autonomous control of the Powered Parachute aircraft for real-world scenarios, discussing correction factors and minimizing the errors present in current sensor systems. Chapter five covers the development of an additional adaptive controller which uses a Sigma-Pi Neural network integrated into the final control

loop. Chapter six develops advanced control methods for the Powered Parachute airframe, including simulations on a novel proposed thrust vectoring method. Finally, chapter seven discusses results accumulated from testing an experimental prototype.

ACKNOWLEDGMENTS

I would give special thanks to my faculty advisor and committee supervisor, Dr. Sangram Redkar. I'm incredibly grateful for his generosity, belief, and support in my abilities during my entire time at ASU Polytechnic. My life has taken a drastically different path than I had once imagined, and I am much better off for it with his guidance. I want to acknowledge my committee members, Dr. Thomas Sugar and Dr. Amar Phatak, for their time and assistance on my educational journey.

To my parents and family, you have given me every opportunity to grow with your unwavering support. I will be forever grateful for you both always encouraging me to follow my dreams and passions; I'm incredibly blessed that I have never been compelled to back away from any of my dreams.

TABLE OF CONTENTS

Page

CHAPTER Page

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

LIST OF ABBREVIATIONS

Abbreviation	Definition
sUAS……..	Small Unmanned Aerial System
PPC……..	Powered Parachute aircraft
COTS……..	Commercial Off the Shelf components
LCF……..	Linear Quadratic Filter
GNC…….	Guidance, Navigation, and Control
NED…….	North-East-Down Coordinate system
MAV……	Micro Aerial Vehicle
AMPC……..	Adaptive Model Predictive Control algorithm
MRAC…….	Model Reference Adaptive Control algorithm
PID……..	Proportional-Integral-Derivative Control
GPS……..	Global Positioning
6 DOF……..	Six Degrees of freedom
NN……..	Neural Network
TVC…….	Thrust Vector Control

CHAPTER 1

INTRODUCTION

Autonomous vehicles have been the fastest growing domain within robotics research in the past few decades. Commercial, humanitarian, and defense-related research have given way to exciting new technologies that can make a lasting impact on modern lives. While genuinely autonomous systems have existed since the advent of analog control schemes, digital control and tremendous advances in control theory have made the capabilities of modern autonomous vehicles into a technology that holds a permanent place in the future of our everyday lives.

Small Unmanned Aerial Systems (sUAS) have the problem of being both complex to control and optimized for power and weight. Borrowing lessons from many engineering disciplines, a system architect must consider multiple factors from electrical, mechanical, and aeronautical engineering decisions during the design process. With recent advances in computer science, algorithms for cooperative control and adaptive mission planning have changed control systems. Past systems were limited to flying simple geometric waypoints or requiring constant user input for guidance. With today's understanding of control engineering, sUAS designers have significantly more tools for completing mission goals and developing aircraft that can perform optimally and robustly. In addition, airframe systems with unique operating principles have found new ground among competing designs as our control technologies improve.

1.1 UAS Historical Background

In the last 20 years, as the capability of microprocessors and sensors has improved, unmanned aerial vehicles have become more accessible and capable. In the early years, autonomous vehicles typically consisted of fixed-wing aircraft because of the airframe's inherent stability and ease of control in simple systems. However, more recent users and their applications for unmanned systems have shifted to the multirotor platform, which is more dynamically demanding. Multirotor aircraft continue to dominate for their utility in performing a large variety of roles, with their only performance limitation being endurance. Multirotor aircraft remain the best choice for robust and agile movement through environments which is why they are the most found UAS in commercial missions. In addition, many researchers have been investigating using Powered Parachute unmanned aerial systems for other roles in surveillance and long-range payload delivery, which require maximal endurance.

Figure 1: Multirotor Package sUAS (Source: Scott Peterson via Getty Images)

A typical small Unmanned Aerial Vehicle consists of an airframe, software and hardware for a digital controller, sensors for navigation, and the mission payload. An entire

small Unmanned Aerial System may also include a ground station for human input or monitoring and possibly additional coordinated autonomous vehicles or some type of centralized controller for navigation and data processing. Alternative mission types for sUAS include but are not limited to surveillance or search and rescue, ground-mapping, payload delivery, agricultural uses like inspection and insecticide spraying, and a multitude of military uses [1].

Fixed-wing UAS has a long heritage in both civilian and military aviation. The Predator from General Atomics has been in service for nearly 30 years as a general UAV platform, carrying imaging systems, radar systems, payload, and even ordinance. Before the Predator, the Ryan AQM-91 Firefly saw service 50 years ago during the Vietnam Conflict. Many other militaries experimented with elementary radio-controlled target aircraft as far back as World War II. With advances in batteries, motors, and flight controllers, fixed-wing aerial vehicles find a new home within civilian applications. Radio Controlled fixed-wing planes are not a new idea, but for many years the technology only allowed short flights which required line of sight. The technology has improved, and prices have decreased to be affordable for the average hobbyist. Currently, Commercial Off the Shelf (COTS) fixed-wing UAS can provide control systems that are robust, versatile, and capable. A single $150 UAS can take off by itself and then fly a non-line-of-sight waypoint-guided mission to GPS coordinates with full telemetry downlink, acting entirely autonomously.

Multirotor systems are newer than fixed-wing UAS but have become a phenomenon in the UAS world for commercial applications and hobbyists alike. The same issues that plagued fixed-wing small UAS in years past also affected multirotor aircraft.

Because multirotor aircraft use motor thrust directly for lift, they must inherently have a greater than one thrust to weight ratio. This requirement proved difficult when batteries of the day could neither provide the current nor energy density to provide good flight times. Additional challenges faced included the slow response of brushed motors and expensive sensors. Today, multirotor vehicles are the utility airframe of choice for handling most missions a UAS is tasked with. For example, one of the most common hobbyist multirotor vehicles, the DJI Mavic Pro, is capable of mission times of about 30-45 minutes, carrying an HD camera on an actively stabilized gimble, and transmitting live video back to the operator. The ubiquitous Multirotor UAS is used commercially and in the military for surveillance, payload delivery, and camera/sensor platforms.

1.2 Introduction to the Powered Parachute

While powered parachute aircraft and parafoils have historically found a use for recreation, there have been a few investigations for using parafoils for space return vehicle recovery. NASA's X38 program is an example of the early development of parafoil control and guidance algorithms and provides a path for understanding the complex dynamic challenges associated with unpowered paragliders [2]. More recently, the private company SpaceX used autonomous parafoil systems to recover the main fairings on their BFR launch vehicle.

The military has also experimented with using parafoil vehicles as coordinated swarms as a proof of concept for the ability to resupply ground troops, with the capability of additional redundancy and area coverage for a delivery mission. For example, the

autonomous CQ-10 Snowgoose was a trial vehicle fielded to resupply ground troops with precision and capable of carrying large payloads to extended ranges.

Figure 2: The CQ-10 Snowgoose PPC UAV for Heavy Lift

While the PPC is not an ideal choice for specific missions requiring agile movement in tight environments, it stands in a class of its own for missions requiring persistent loitering and large payload capacity. For an airframe of equal weight as a fixed-wing vehicle, the PPC will typically be able to extend time aloft by a factor of three, significantly outclassing the fixed-wing autonomous vehicles for missions of that nature. The PPC is more efficient than multirotor aircraft for mission profiles requiring near hover or slow flight.

The unique dynamics of parafoils in glide require special algorithms for guidance and waypoint following because a parafoil in un-powered glide typically only follows a constant descent rate and forward velocity. For Powered Parachutes, this dynamic property relates to the maximum descent rate capable of being achieved. This fundamental

characteristic of PPCs is the first difference from fixed-wing aircraft realized in the following sections on dynamics within this paper.

1.3 Analyzing the Dynamic Properties of a Standard Powered Parachute sUAS

This section covers the differences in dynamic models between the various control papers surveyed. For most papers surveyed here, two key assumptions hold for the dynamics of the PPC:

1. The PPC maintains a fixed velocity relative to the air mass.
2. Directional (lateral) control for turning the aircraft is based only on the vertical (yaw) axis.

These two assumptions dictate the ability of the PPC to be controlled in flight and form the basis for the development of dynamic models. The first assumption is critical in understanding the longitudinal (pitch) axis dynamics. A PPC cannot control its forward velocity. An increase in motor thrust translates directly to a rotational moment about the pitch axis, increasing the angle of attack for the parafoil, resulting in a climb. The second assumption references the pendulum stability of the airframe's center of mass resting significantly below the parafoil's vertical lift component.

These realizations demonstrate the significant difference between PPCs and fixed-wing aircraft. The typically fixed-wing aircraft has four control inputs: thrust, pitch, yaw, and roll, whereas the PPC only has the control inputs for thrust and yaw. A typical airplane changes direction by rolling then pitching, but the only control input for lateral (yaw) movement with a PPC is to activate an airbrake on the parafoil [3], [4]. While some roll

moment will result when triggering the airbrakes on the PPC's canopy, it is mainly assumed negligible. A Fixed-wing type roll/pitch lateral control has not yet been applied for the autonomous PPC. Most papers assume the lateral control is decoupled from longitudinal control methods to simplify the dynamic models. Only Wanatabe [5] and Slegers [3] maintain the coupled dynamics between the longitudinal and lateral control axes of the documents surveyed. The yaw control system calls into question the responsiveness and controllability of the PPC. Without some form of additional control input surface or method, the PPC will be unable to counter certain windy conditions for which a fixed-wing aircraft can maneuver successfully. One of these conditions is the frequently encountered with variable or gusting crosswind environmental components.

A key challenge in accurately capturing the dynamics of a PPC model falls with uncertainties for lift and drag coefficients. With fixed-wing aircraft, there are many numerical and analytical solutions for finding these coefficients based on years of measurements and research; for PPCs and non-rigid parafoils, the airfoil can be significantly more challenging to model. However, specific fixed-wing methods may help estimate lift and drag coefficients, as seen in the dissertation by Slegers [3]. Slegers successfully demonstrates that the canopy dynamics stay rigid during the regular operation of the PPC (a completely inflated canopy).

Mihai uses a different methodology for generating lift and drag coefficients. By capturing empirical data from the testbed described in his paper, he shows it is possible to reconstruct uncertain aerodynamic coefficients with accuracy [6]. Mihai then gives methods for simulations of the estimated coefficients, comparing them to the captured flight data with success.

The attachment of the parafoil to the airframe of a powered parachute varies within the papers surveyed. The most simplistic dynamic model and one of the most simulation-detailed examples of a PPC is derived in the paper by Umenberger [7]. The chute/airframe attachment is modeled as a rigid connection, and Umenberger successfully covers decoupled longitudinal and lateral flight dynamics in simulations. Van Der Kolf [8] also shows that while this fixed-body assumption may introduce specific errors, a fixed joint model can still be used to develop a PID stable controller.

If the attachment points between the parafoil and airframe are assumed to be rotating (either freely or with some rotational stiffness coefficient), the dynamics for the PPC become increasingly more complex. Aoustin [9] and Slegers [3] develop six degrees of freedom dynamic equations for this type of connection. Ward shows how multiple attachment points may make the PPC a 9 or 10 DOF dynamic model. In most cases, certain degrees of freedom may be considered insignificant. These assumptions are derived from the time base of oscillations induced in the joints between the parafoil and the airframe, which have little effect on the vehicle's actual performance [10].

Within the papers by Ochi [11] and Umenberger [7], the dynamic models developed are linearized before developing the controller. This linearization is performed at a constant altitude and thrust setting equilibrium point for pitch axis motion. For lateral/directional movement, the linearization point is assumed as straight and level flight. Umenberger's paper shows an additional method for capturing the aerodynamic coefficients derived using results from simulation.

1.4 Guidance Navigation and Control of PPC Aircraft Research Survey

The most straightforward control scheme implemented in longitudinal dynamics is the PID controller, as demonstrated in the linearized dynamics model by [1]. However, these controllers are not optimal and are confined to a small dynamic envelope. They may become increasingly unstable the further away from the linearization point required to operate. Chambers shows the large time scale oscillations induced by the PID controller on the non-linear model in his simulations. A step input response to the motor shows that the first assumption from the previous section holds: that the PPC will settle at a fixed forward velocity relative to the air mass. Aoustin experiences similar results for oscillations in linearized control developed for the non-fixed pendulum (6 DOF) dynamic model, recommending that further work generate robust control for a 3-dimensional dynamic model [9].

Fixed waypoint mission flying is a standard method to program compared to adaptive mission planning. The autonomous vehicle navigates a series of points in space termed "waypoints," using information from GPS and altitude sensors. The autopilot algorithm attempts to place the vehicle on an intercept with each waypoint. Several COTS solutions are available for implementation in fixed-wing and multirotor UAS.

In "A Waypoint following Control Design for Paraglider Model…" Tanaka et al. demonstrated a control design for a PPC using a six degree of freedom (6 DOF) dynamic model [12]. Their experimental model has flown a real PPC model craft and demonstrated autonomous landing.

Depending on the mission goals and type, it may not be desirable to pre-program a mission in advance. One of the limitations of waypoint navigation is that it is entirely blind

to new information and conditions. This method is dangerous to the craft and misses potential opportunities for new conditions such as wind or updrafts, which could positively affect mission parameters like loiter time. Jann (2005) is an adaptive navigation method with better guidance algorithms like the Linear Complementary Filter (LCF) Control developed for GNC. Using Monte Carlo simulations and stochastic techniques, Jann shows that a waypoint following algorithm developed for PPCs can counter environmental unknowns by continuously updating the optimal path to the waypoint [13].

1.5 Introduction to Alternative Control Methods

Ward presents a mathematical model with simulation and experimental results for autonomous control of parafoil vehicles [14]; the significant contribution of this paper is the introduction of an additional control input to modify the rigging geometry of the parafoil attachment to the airframe. As described previously, typical methods for the longitudinal Control of the PPC are only governed by the response of the airframe to thrust at a fixed velocity.

Ward's novel method of control is realized in actual flight tests experiments by changing the attachment length of certain lines, which changes the incidence angle of the parafoil above the airframe while in flight. Slegers et al. [3] described in detail within their paper the effect of angle of incidence has on the glide slope of a parafoil vehicle. However, the testing and simulations were only an analysis of incidence angle at fixed intervals. Slegers summarizes that at a fixed incidence angle, the glide slope of the parafoil vehicle and, therefore, the forward velocity of the vehicle can be modified depending on the design.

By controlling the angle of incidence, Ward can increase the control envelope of available glide slopes, allowing the vehicle to adapt to a broader range of environments and mission flight profiles. This additional degree of control breaks the previous assumption that the forward velocity remains fixed for parafoil vehicles. This assumption was consistent with all other articles reviewed up to this point.

CHAPTER 2

DYNAMIC MODEL OF A POWERED PARACHUTE UAV

This chapter aims to develop the basis for guidance navigation and control strategies for a paramotor UAV through developing the kinematic and dynamic fundamentals that govern the flight of the Parasail sUAS. As previously discussed, PPC dynamics are closely related to fixed-wing aircraft. The PPC has the equivalent of a large amount of wing dihedral, in essence acting as a pendulum keeping the airframe upright. The following figure shows a simplistic free-body diagram. This chapter provides the definitions of coordinate frames, kinematics, and dynamics deriving a generalized model for a PPC using first principles under a rigid body assumption of the parafoil to airframe junction.

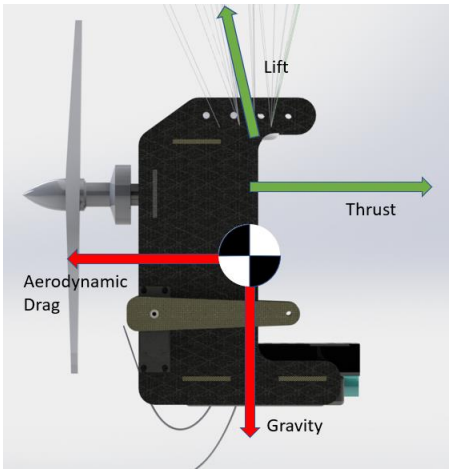

Figure 3: Simple Free Body Diagram

2.1 Coordinate Frames and Kinematic Definitions

The kinematics of the sUAS rely on 12 state variables which describe the position and orientation in various frames. The coordinate frames are used to describe the location

and orientation of the sUAS and perform calculations and measurements. Sensors that capture the position and orientation are limited to what is available on the sUAS. For example, the typical sUAS can read GPS location data and 3-axis force and gyroscopic data. Some sUAS capture magnetic compass heading, altitude, and airspeed with barometric pressure sensors. Mathematical conversion between coordinate frames requires a series of Euler angles and rotation matrices.

The rotation matrices are summarized in all three axes in the following equations:

For rotation about the z-axis:
$$R_0^1 = \begin{pmatrix} cos\theta & sin\theta & 0 \\ -sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (2.1)$$

For rotation about the y-axis:
$$R_0^1 = \begin{pmatrix} cos\theta & 0 & -sin\theta \\ 0 & 1 & 0 \\ sin\theta & 0 & cos\theta \end{pmatrix} \quad (2.2)$$

For rotation about the x-axis:
$$R_0^1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos\theta & sin\theta \\ 0 & -sin\theta & cos\theta \end{pmatrix} \quad (2.3)$$

The Inertial frame of the sUAS, $\mathcal{F}^i$, is the North-East-Down (NED) earth fixed positional coordinate of the sUAS center of gravity, defined as the cartesian distance from some arbitrarily defined home location. The following figure displays the convention North-East-Down (NED) inertial coordinate system.

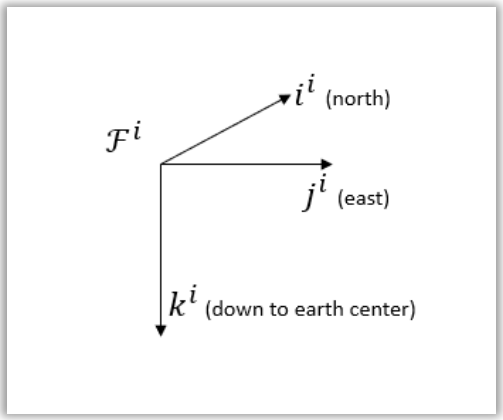

Figure 4: Inertial Coordinate Frame

The sUAS Vehicle frame, $\mathcal{F}^v$, has its axes aligned with the inertial frame NED axis, with it's origin at the CG of the sUAS. The transformation from inertial frame to vehicle frame is the translation by the NED coordinates from the Inertial frame. The vehicle frame is used to define the rotation angles to define the sUAS in space completely. After the three rotation matrices (yaw, pitch, roll) are applied to the Vehicle frame, the resulting frame is defined as the body frame. In the body frame, the CG is also the origin, but the axes are now aligned with the body of the UAS. The overall rotation from the body frame to the vehicle frame is performed using a combination of equations presented earlier. The simplified rotation matrix is given as follows:

$$\mathcal{R}_v^b(\emptyset,\theta,\varphi) = \mathcal{R}_{v2}^b(\emptyset)\,\mathcal{R}_{v1}^{v2}(\theta)\,\mathcal{R}_v^{v1}(\varphi)$$

$$= \begin{pmatrix} C_\theta C_\psi & C_\theta S_\psi & -S_\theta \\ S_\phi S_\theta C_\psi - C_\phi S_\psi & S_\phi S_\theta S_\psi + C_\phi C_\psi & S_\phi C_\theta \\ C_\phi S_\theta C_\psi + S_\phi S_\psi & C_\phi S_\theta S_\psi - S_\phi C_\psi & C_\phi C_\theta \end{pmatrix} \tag{2.4}$$

The following figure describes how the inertial positions of the sUAS relate the vehicle frame to the inertial frame. The axes of the vehicle frame are still aligned with the original inertial frame, but the center of gravity is translated by the p_n, p_e, $and\ p_d$ distances.

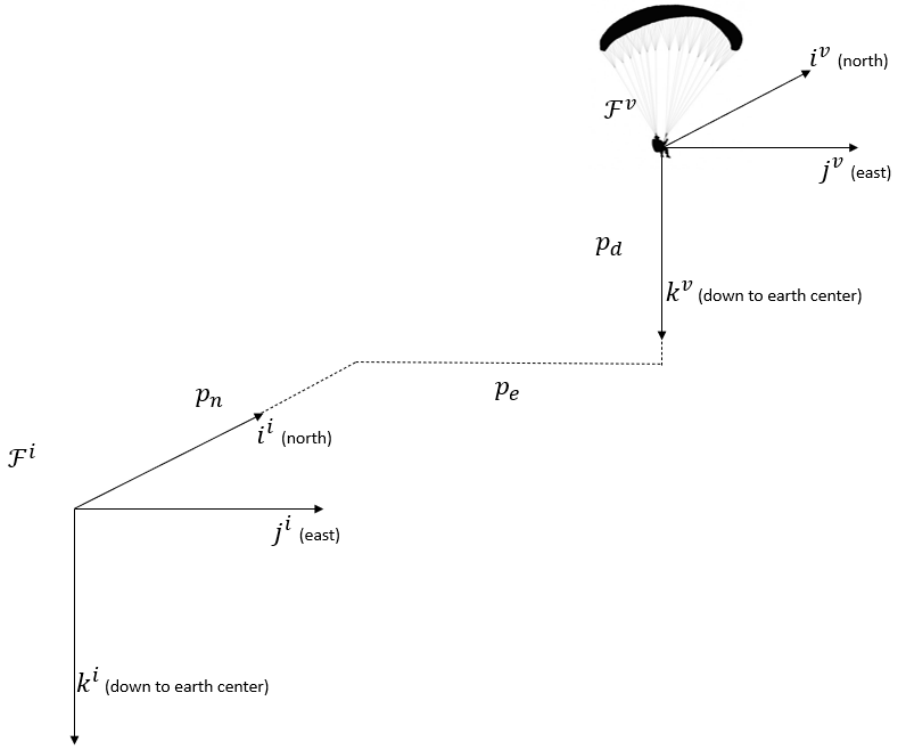

Figure 5: Inertial Coordinate Frame and Translation to the Vehicle Frame

The body frame is the orientation in which the sUAS sits in three-dimensional space. Combined with the vehicle frame coordinates, it fully constrains the sUAS about the inertial origin. The final coordinate frame, the stability frame, is a vector description of the velocity of the airframe through 3d space. Because the airframe is not moving through space directly aligned with its axes, the stability frame describes the airframe's angle of attack and sideslip.

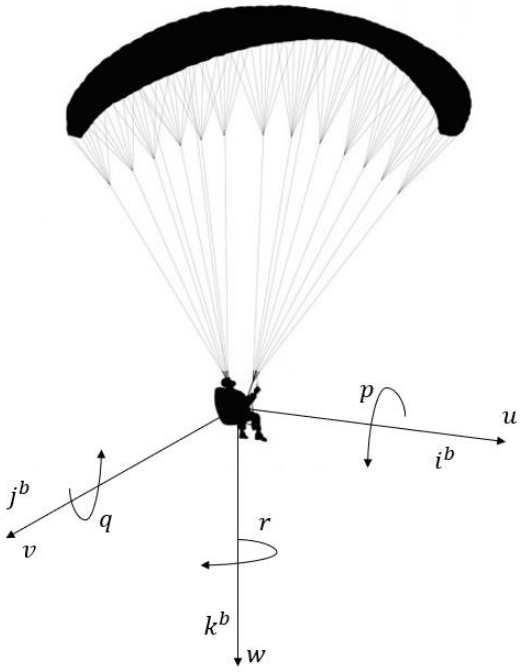

Figure 6: The Body Coordinate Frame for a Powered Parachute

The following table summarizes the 12 state variables for the sUAS.

Table 1

State and Coordinate Frame Variables for sUAS Dynamic Model

State	Coordinate Frame	Description
p_n	$\mathcal{F}^i \; to \; \mathcal{F}^v$	North position from inertial frame
p_e	$\mathcal{F}^i \; to \; \mathcal{F}^v$	East Position from inertial frame
p_d	$\mathcal{F}^i \; to \; \mathcal{F}^v$	Downward position from inertial frame
u	$\mathcal{F}^b$	Body Frame velocity along i^b
v	$\mathcal{F}^b$	Body Frame velocity along j^b
w	$\mathcal{F}^b$	Body Frame Velocity along k^b
$\emptyset$	$\mathcal{F}^{v2}$	Roll angle

θ	$\mathcal{F}^{v1}$	Pitch angle
φ	$\mathcal{F}^{v}$	Yaw angle
p	$\mathcal{F}^{b}$	Roll rate along i^b
q	$\mathcal{F}^{b}$	Roll rate along j^b
r	$\mathcal{F}^{b}$	Roll rate along k^b

2.2 Dynamics

The Rigid Body dynamics equations for the PPC and other sUAS are captured by applying Newton's Second Law in the translation and rotational degrees of freedom. The second law only applies to the fixed coordinate frame (the initial inertial coordinate frame), so all measurements taken or calculated in-vehicle referenced frames must be transformed. Several minor simplifications are made in this model, which applies to the scale of the missions that sUAS typically perform. First, the inertial frame is to stay fixed in space, although it can be arbitrarily defined. Secondly, we assume that the flat earth model applies to our calculations.

The relationship between the translational velocities (u,v,w) and the inertial position requires differentiation and rotation.

$$\frac{d}{dt}\begin{pmatrix} p_n \\ p_e \\ p_d \end{pmatrix} = (R_v^b)^{\mathrm{T}} \begin{pmatrix} u \\ v \\ w \end{pmatrix} \tag{2.5}$$

$$\begin{pmatrix} \dot{p_n} \\ \dot{p_e} \\ \dot{p_d} \end{pmatrix} = \begin{pmatrix} C_\phi C_\theta & C_\phi S_\theta S_\psi - S_\phi C_\psi & C_\phi S_\theta C_\psi + S_\phi S_\psi \\ S_\phi C_\theta & S_\phi S_\theta S_\psi + C_\phi C_\psi & S_\phi S_\theta C_\psi - C_\phi S_\psi \\ -S_\theta & C_\theta S_\psi & C_\theta C_\psi \end{pmatrix} \begin{pmatrix} u \\ v \\ w \end{pmatrix} \tag{2.6}$$

Applying Newton's second law for translational dynamics yields the following equation:

$$f = m\,\frac{dV}{dt} \tag{2.7}$$

Where m is the fixed mass of the sUAS, the forces acting on the sUAS are gravitational, aerodynamic, and propulsive. The time derivative of the inertial velocity is the acceleration applied to the sUAS at any given moment. In matrix notation, eq. 2.5 may be rewritten as the following components in a rearranged form to give the state notation and then expressed according to body frame velocities. The additive components make up the ground velocity containing the wind vectors that modify the body frame velocities.

$$\begin{pmatrix}\dot{u}\\ \dot{v}\\ \dot{w}\end{pmatrix} = \begin{pmatrix}rv - qw\\ pw - ru\\ qu - pv\end{pmatrix} + \frac{1}{m}\begin{pmatrix}f_x\\ f_y\\ f_z\end{pmatrix} \tag{2.8}$$

Additionally, the three angular positional rates $(\dot{\phi}, \dot{\theta}, \dot{\psi})$ are expressed with their respective body frame rates (p, q, r).

$$\begin{pmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{pmatrix} = \begin{pmatrix}1 & S_\phi T_\theta & C_\phi T_\theta\\ 0 & C_\phi & -S_\phi\\ 0 & S_\phi / C_\theta & C_\phi / C_\theta\end{pmatrix}\begin{pmatrix}p\\ q\\ r\end{pmatrix} \tag{2.9}$$

Finally, the last three states $(\dot{p}, \dot{q}, \dot{r})$, define the rotational motion in terms of angular momentum where all force-moments are summed about the center of mass of the sUAS. The following J matrix is comprised of the moments of inertial found as a property of the sUAS airframe mass and geometries.

$$J = \begin{pmatrix} J_x & -J_{xy} & -J_{xz} \\ -J_{xy} & J_y & -J_{yz} \\ -J_{xz} & -J_{yz} & J_z \end{pmatrix} \tag{2.10}$$

The rotational motion is defined by Newton's second law and expressed in the derivative of the body frame, with the summed force-moments $m = (l, m, n)$ about the $(\check{\imath}, \check{\jmath}, \check{k})$ axes in the body frame.

$$\begin{pmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{pmatrix} = \begin{pmatrix} \Gamma_1 pq - \Gamma_2 qr \\ \Gamma_5 pr - \Gamma_6 (p^2 - r^2) \\ \Gamma_7 pq - \Gamma_1 qr \end{pmatrix} + \begin{pmatrix} \Gamma_3 l + \Gamma_4 n \\ \frac{1}{J_y} m \\ \Gamma_4 l + \Gamma_8 n \end{pmatrix} \tag{2.11}$$

Where the Γ parameters are airframe constants calculated in the following equation block:

$$\begin{aligned} \Gamma_1 &= \frac{J_{xz}(J_x - J_y + J_z)}{\Gamma} & \Gamma_2 &= \frac{J_z(J_z - J_y) + J_{xz}^2}{\Gamma} \\ \Gamma_3 &= \frac{J_z}{\Gamma} & \Gamma_4 &= \frac{J_{xz}}{\Gamma} \\ \Gamma_5 &= \frac{J_z - J_x}{\Gamma} & \Gamma_6 &= \frac{J_{xz}}{\Gamma} \\ \Gamma_7 &= \frac{(J_x - J_y)J_x + J_{xz}^2}{\Gamma} & \Gamma_8 &= \frac{J_x}{\Gamma} \end{aligned} \tag{2.12}$$

In summary, the equations described in 2.6, 2.8, 2.9, and 2.11 form the basis for the dynamic model to be used continuing forward. This generalized model will use the aerodynamic, gravitational, and propulsive forces described in a later second to calculate the overall linear forces $\begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix}$ and moments $\begin{pmatrix} l \\ m \\ n \end{pmatrix}$. All 12 states are non-linear, but the

following section will show some simplification and linearization of the state model to simplify developing a controller.

Lastly, three aerodynamic calculations will be helpful for further defining the aerodynamic model. The total airspeed, $\tilde{V}$, the angle of attack, α, and the sideslip angle, β.

$$\tilde{V} = \sqrt{u^2 + v^2 + w^2} \tag{2.13}$$

$$\alpha = \tan^{-1}(w/u) \tag{2.14}$$

$$\beta = \sin^{-1}\left(v/\tilde{V}\right) \tag{2.15}$$

2.3 Linear Design Models and PPC Forces

This section derives the forces and moments that act on the parasail UAV. There are three forces and two moments which will act on the PPC within its 6 degrees of freedom.

$$F_{Total} = F_{gravity} + F_{aero} + F_{prop} \text{ and } M_{Total} = M_{aero} + M_{prop}$$

2.4 Gravitational Forces

The force of gravity on the PPG in the inertial frame is $F = \begin{pmatrix} 0 \\ 0 \\ mg \end{pmatrix}$

Translating the inertial frame to the body frame:

$$f_g^b = \mathcal{R}_v^b \begin{pmatrix} 0 \\ 0 \\ mg \end{pmatrix} = \begin{pmatrix} -mg\sin\theta \\ mg\cos\theta\sin\phi \\ mg\cos\theta\sin\phi \end{pmatrix} \tag{2.16}$$

2.5 Aerodynamic Forces and Moments

Up to this point, the kinematics and dynamics have been generalized among most fixed-wing sUAS. There are only two control degrees of freedom for the paramotor sUAS compared to the four on a fixed-wing aircraft. The standard configuration Paramotor sUAS can only apply a yaw moment and a thrust in its control. Applying a yaw moment to the wingtip, a small amount of adverse roll is induced to the PPC. However, the adverse roll is considered much smaller than the stabilizing 'pendulum' effect of the airframe hanging far below the parafoil. This mechanism is like a fixed-wing aircraft with a significant self-stabilizing dihedral design, except to a greater extent. This is not to say that all parafoil aircraft cannot use roll mode flight; instead, most aerobatic parachutes can use roll techniques and turn similar to a standard fixed-wing aircraft. However, this type of chute and dynamic design is not desired for autonomous control.

Building on the previous assumption, the general control schemes of Paramotor sUAS assume decoupled dynamics between the lateral and longitudinal axis. Similarly, the aerodynamic forces present on the body of the paramotor are present but not as significant as the aerodynamic forces on the parafoil. In general terms, we can define the force of lift, the force of drag, and the aerodynamic moment within the following basic equations:

$$F_{Lift} = \frac{1}{2}\rho V_a^2 S C_L \tag{2.17}$$

$$F_{Drag} = \frac{1}{2}\rho V_a^2 S C_D \tag{2.18}$$

$$m = \frac{1}{2}\rho V_a^2 S c C_m \tag{2.19}$$

Where C_L, C_D, C_M are aerodynamic coefficients, S is the planform wing area of the airfoil, and c is the wing chord length (distance to front and back of the airfoil). While the typical aerodynamic coefficients of airfoils are most significantly affected by the airfoil shape, angle of attack of the wing, and the Reynolds and Mach Numbers. The small size of most parafoil UAVs and the velocity they travel in the air means that an approximation for the Reynolds and Mach as constants is made, and the angle of attack will stay roughly constant for a specific wing loading the parafoil. The following two sections will further deconstruct the principles here into the longitudinal and lateral dynamics and cover the effects of the angle of attack, sideslip angle, control surface deflections on the aerodynamic coefficients, and how forces are applied to the vehicle through the parafoil.

2.6 Longitudinal Forces and Moments

The longitudinal vehicle dynamics affect motion in the vehicle frame pitch plane. The non-linear equations can be simplified by performing a first-order Taylor series approximation of the coefficient of lift at the point where the angle of attack, pitch rate, and control surface deflection is zero.

$$F_{Lift} = \frac{1}{2}\rho V_a^2 S\left[C_{L_o} + \frac{\partial C_L}{\partial \alpha}\alpha + \frac{\partial C_L}{\partial q}q\right] \tag{2.20}$$

Equation 2.20 is then non-dimensionalized to remove the partial derivatives for the lift, drag, and moment equations.

$$F_{Lift} = \frac{1}{2}\rho V_a^2 S\left[C_{L_o} + C_{L_\alpha}\alpha + C_{L_q}\frac{c}{2V_a}q\right] \tag{2.21}$$

$$F_{Drag} = \frac{1}{2}\rho V_a^2 S\left[C_{D_o} + C_{D_\alpha}\alpha + C_{D_q}\frac{c}{2V_a}q\right] \tag{2.22}$$

These new, non-dimensional coefficients C_{L_o}, C_{L_α}, C_{L_q} and the respective terms for drag and moment are referred to as stability derivatives.

The forward motor thrust in the x-axis is given in the following equation:

$$F_{x_p} = \frac{1}{2}\rho S_{prop} C_{prop}[(k_{motor}\delta_t)^2 - V_a^2] \tag{2.23}$$

Where S_{prop} and C_{prop} are the physical dimensions of the propellor used in length and pitch within the simplistic motor model. δ_t and k_{motor} are the commanded thrust setting and the motor angular velocity, which have a linear relationship.

Additionally, the longitudinal pitching moment is derived in a similar method as equations 2.21 and 2.22:

$$m = \frac{1}{2}\rho V_a^2 S_{chute}\left[c(C_{m_o} + C_{m_\alpha}\alpha + C_{m_q}\frac{c}{2V_a}q) + F_{x_{body}}H_{chute}\right] \quad (2.24)$$

Unlike fixed-wing aircraft, which may be capable of stalling the airfoil in modes of flight where the angle of attack exceeds a critical angle, the PPC airfoil is considered incredibly difficult to stall because of its high lift nature. For this reason, the Taylor series equations given in () are a good approximation of the longitudinal aerodynamics for the PPC wing. In contrast, a lift model which incorporated stall effects would include additional terms. Therefore, for the PPC operating within a critical angle of attack, the flat plate model for the lift coefficient may be used.

$$C_{L,flat\ plate} = 2sign(\alpha)sin^2\alpha\cos\alpha \quad (2.25)$$

The longitudinal dynamics in this section will be further experimented with in later chapters but are overall good approximations for the coefficient of lift. These dynamic equations will not account for aircraft stability within longitudinal and lateral aerodynamics change. The stability will decrease in high rates of climb. For this reason, it is

recommended to restrict the rates of climb with the maximum throttle available and restrict the rate that the throttle is applied.

2.7 Lateral Forces and Moments

Paramotor aircraft apply a steering input by deflecting a drag surface on the outside of the wing. In an ideal model, the drag surface will cause the PPC to yaw about the body frame Z-axis of rotation. However, similar to how a fixed-wing airplane will experience adverse yaw from using aileron control, the PPC will have some amount of adverse roll condition in the direction opposite to the desired control input. This adverse roll is transitory, and the roll stability coefficients of the airframe will determine how a specific airframe will respond to a control input.

The force is applied at the edge of the wing, off from the center of gravity, and will cause a moment yawing the entire airframe of the PPC. The airbrake surface area exposed to the wind stream is a function of the deflection of the control surface. The simplified mechanism is pictured in the following image:

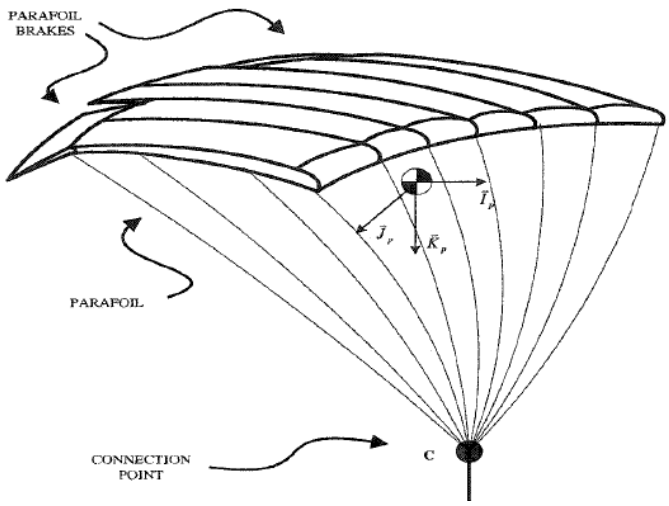

Figure 7: Parafoil Steering Configuration [10]

The total aerodynamic forces expressed within the body frame are calculated as an addition between the forces acting on the canopy and the forces acting on the body.

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix}_{Chute} = \frac{1}{2} p \tilde{V}^2 S_{chute} \begin{bmatrix} c\alpha & 0 & -s\alpha \\ 0 & 1 & 0 \\ s\alpha & 0 & c\alpha \end{bmatrix} \begin{pmatrix} -Cd \\ C_{Y\beta}\beta \\ -C_L \end{pmatrix} \tag{2.26}$$

β is defined as the sideslip air velocity vector (Eqn. 2.15). Additionally, the aerodynamic forces on the airframe:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix}_{Airframe} = \frac{1}{2} p \tilde{V}^2 S_{body} C_{D,f} \begin{pmatrix} u \\ v \\ w \end{pmatrix} \tag{2.27}$$

The aerodynamic moments on the airframe are a function of air brake surfaces applied at the distance between parafoil chord wingspan, c, and parafoil height over the airframe, b.

$$\begin{pmatrix} l \\ m \\ n \end{pmatrix} = \frac{1}{2} p\tilde{V}^2 S_C J \begin{pmatrix} b \\ c \\ b \end{pmatrix} \tag{2.28}$$

The propulsion forces are derived as a simplified model where the force of thrust produced by the motor and propellor can be mapped directly to the throttle control. The torque on the airframe is proportional to the amount of thrust.

$$F_{prop} = \begin{pmatrix} T\delta_t - V_a \\ 0 \\ 0 \end{pmatrix} \tag{2.29}$$

The moment is similarly related to the fixed-wing aircraft, except a pitching moment related to the thrust amount and the vertical distance, d, that the propellor is mounted to the airframe from the center of gravity. The constant R is the proportionality coefficient.

$$m_{prop} = \begin{pmatrix} TR\delta_t \\ (T\delta_t - V_a)d \\ 0 \end{pmatrix} \tag{2.30}$$

2.8 Computational Fluid Dynamics Analysis of Parafoil

A Computational Fluid Dynamics (CFD) analysis may be performed on the geometry of the parafoil to determine the aerodynamic coefficients C_L, C_D, and C_M. The simulation creates a surface mesh of an imported 3D geometry for the parafoil model, with defined velocities and angles of attack for a specific simulation run. To develop a result table applicable to all angles of attack, forward velocities, multiple simulations are necessary to interpolate an accurate representation of aerodynamic coefficients. This simulation modeled the parafoils as a rigid body for simplicity, a method examined within [15] to have acceptable accuracy.

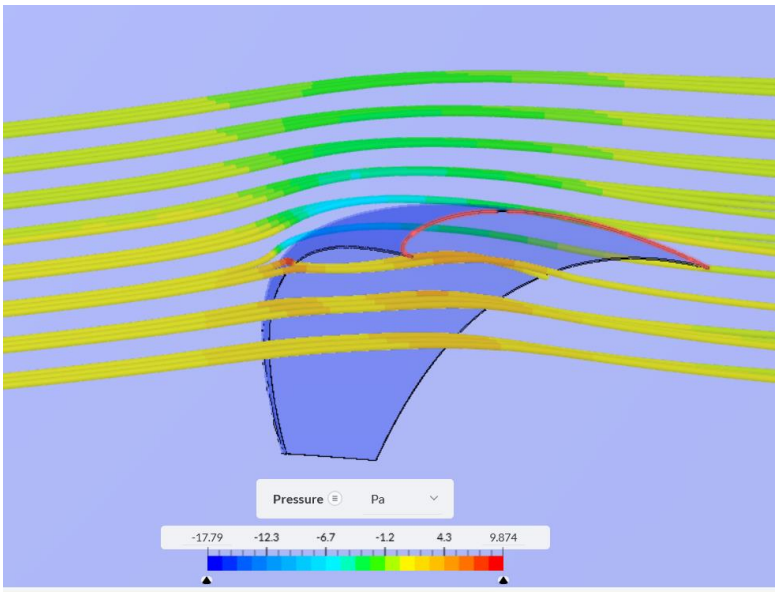

Figure 8: Vector Field Rendering of Pressure Gradient

CFD software can calculate the lift force and drag force, which may be processed to calculate lift and drag coefficients for each angle of attack. This process will help to accurately model the aerodynamic forces that the paramotor sees during flight. By using a slightly simplified geometric model of the test parafoil, simulations at various angles of

attack were performed to capture the coefficients of lift, drag, and moments. These results are then used to fit a 3rd order polynomial expression for use within the simulation. It should be noted that the rigging of parafoil to the airframe was excluded from this simulation, and instead, an estimate was included in the airframe coefficient of drag. The results are shown in the following table, using the actual parafoil geometry and parameters used in the testing in chapter polynomial fit equations for the coefficient of lift and drag at the end of this chapter and the simulations performed in Chapter 3 of this document.

Table 2

CFD Results for Coefficients of Lift and Drag (Opale H1.5 Airfoil)

AoA (deg)	Lift Force	CL	Drag Force	CD
0	5.90	0.411	0.98	0.43
2	8.72	0.605	1.14	0.54
4	11.88	0.83	1.52	0.64
6	14.52	1.012	1.52	0.55
8	17.10	1.191	1.83	0.59
10	19.26	1.343	2.21	0.63
12	21.00	1.47	2.62	0.68
14	22.66	1.57	3.04	0.72
16	23.18	1.62	3.88	0.83
18	21.10	1.49	4.94	0.98
20	20.76	1.45	5.00	0.92
22	24.20	1.41	6.40	1.10
24	26.40	1.766	6.94	1.11
26	20.20	1.47	7.60	1.14
28	19.60	1.28	8.80	1.25

The main benefit from running this simulation is it can help rigidly define a range for desirable air frame mass. By selecting the mass to be equal to the lift force at the point where the ratio of C_L/C_D is the greatest, it will be insured the parafoil is operating with its

most efficient angle of attack in straight and level flight. The results of this simulation are validated against the closed form flat plate solution for lift and drag of a plate with a similar area [16]. It should be noted that the flat plate solution does not consider stall regimes; therefore, it is only useful for comparisons below a certain angle of attack. Small scale PPC aircraft typically have a high angle of attack while flying (approximately 8˚) so this comparison is only used for validation of the CFD simulation. The flat plate model used is given in the following equation:

$$C_L(\alpha) = \big(1 - \sigma(\alpha)\big)\big[C_{L_0} + C_{L_\alpha}\alpha\big) + \sigma(\alpha)[2 * sign(\alpha)\sin^2\alpha\cos\alpha] \quad (2.31)$$

Where the C_{L_0}, C_{L_α} terms are constants describing the zero AoA (Angle of Attack) lift coefficients, and α is the angle of attack of the airfoil. By using the measurements from the prototype airfoil and a comparison of the flat plate model, the results from the CFD analysis yield the following results. The following results are the cumulative results of the many CFD simulations for the Opale H1.5 PPC wing. The simulations were performed at varying angles of attack of 0˚ - 28˚ at every 2˚ to capture results accurately. The CFD simulation results are first verified against the flat plate model resulting in the following plot:

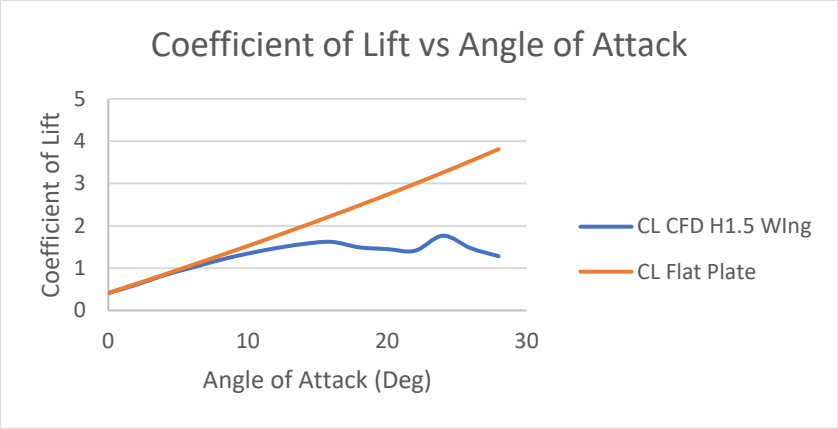

Figure 9: Coefficient of Lift vs Angle of Attack (Theoretical and Simulated)

A divergence of the two results is seen within an angle of attack at approximately 10˚. As mentioned previously, the divergence results from the closed-form flat plate model's inability to model stall and near stall lift characteristics.

The following results graph includes the CFD calculated coefficient of drag compared to the coefficient of lift at the same angle of attack. Of note, the most efficient design AoA for the parafoil will be equal to the highest ratio for C_L/C_D. In the case of this parafoil, that point varies from approximately 6 ˚ to 10˚. In contrast to a fixed-wing airplane, this directly correlates to the lift force available for the wing, which will equal the gross airframe weight in straight and level flight.

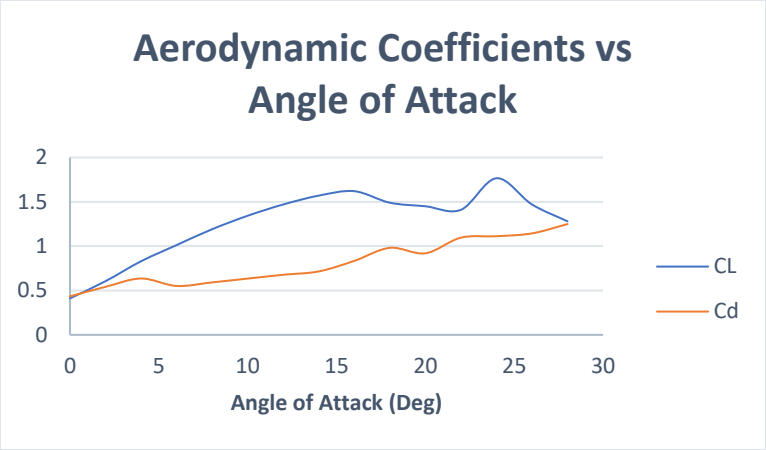

Figure 10: Coefficient of Lift and Drag CFD Simulated Results

A visible delineation between a stalled and unstalled airfoil was captured within the simulation with a cross-sectional view of the pressure magnitude. The following two images show both stalled and unstalled parafoils. A low-pressure area is attached to the wing in the first image of the unstalled lifting surface. The second image from the simulation shows the stalled airfoil, where the flow has become separated from the wing.

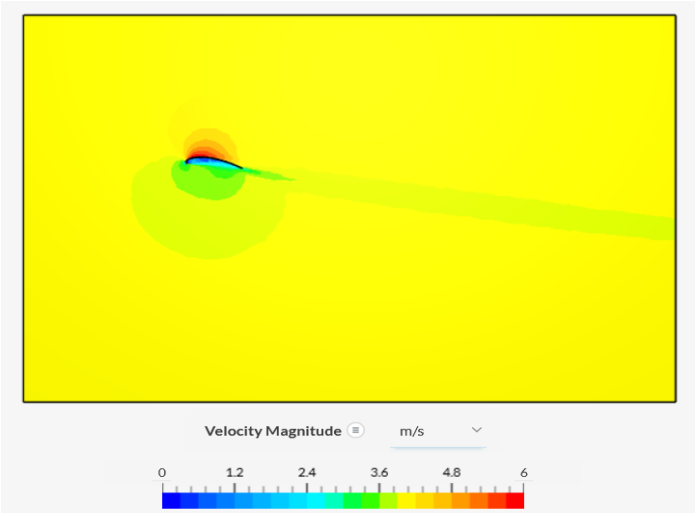

Figure 11: Wing root, 8 degree Angle of attack, Attached Flow (Unstalled)

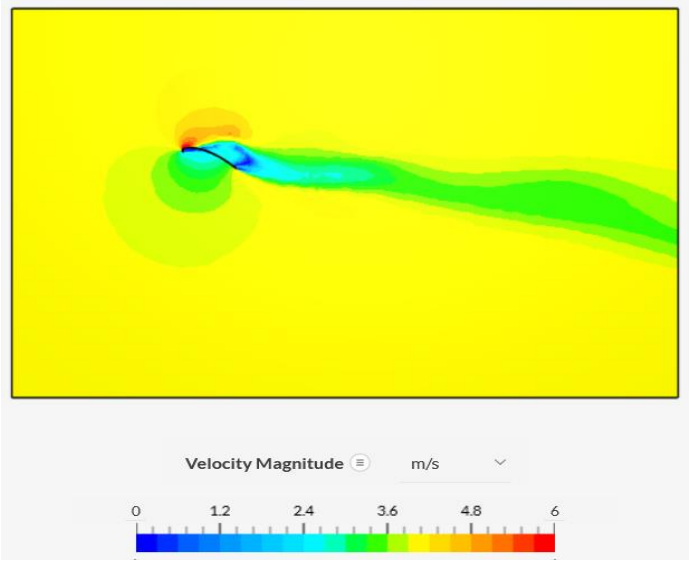

Figure 12: Wing root, 20 degree Angle of Attack, Unattached Airflow (Stalled)

The color gradient within the previous images represents the calculated pressure magnitude, where red areas are of higher pressure-magnitude, and blue areas are of lower pressure-magnitude. The PPC designer should develop a payload not to exceed critical angles of attack and to maintain the best ratio of C_L/C_D for efficiency.

This next image is another example of verifying simulation flow by capturing the airfoil's particle trace during simulation. Again, as is typical of all lifting surfaces, a wing vortex is visible on the outside edge of the wing.

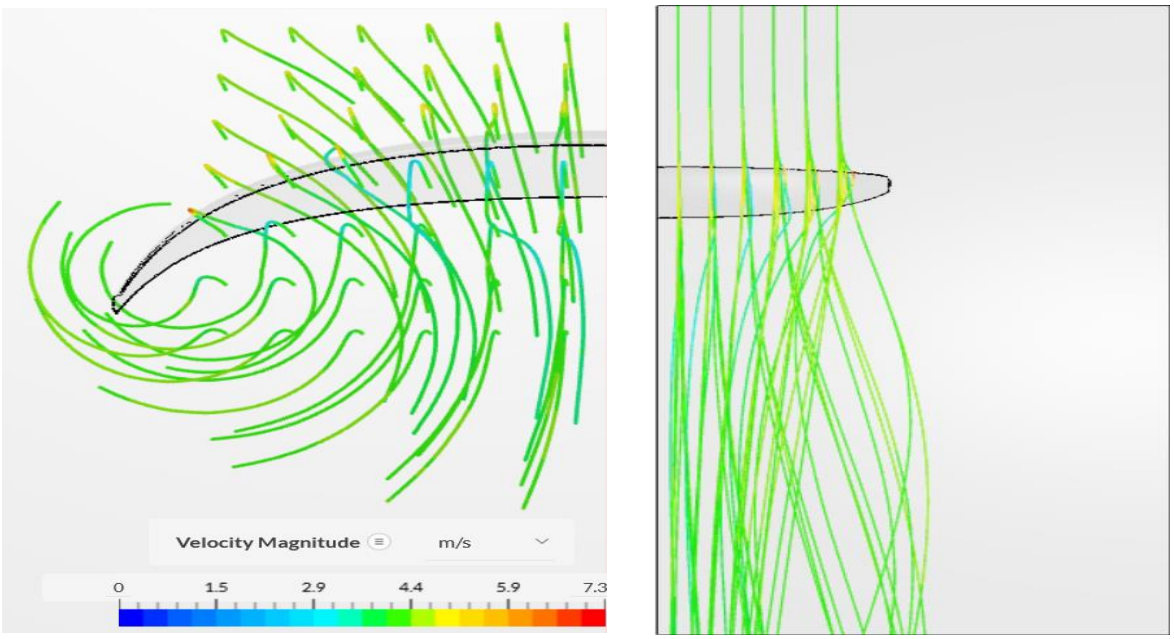

Figure 13: Particle Trace of CFD on Parafoil

Results from the CFD simulations should not be considered conclusive until verified against real-world testing; however, the aerodynamic behavior is similar to the results of the prototype. CFD Model verification will be performed on drop tests to determine glide ratio and Coefficient of Lift and Drag ratios from experimental data.

2.9 Atmospheric Disturbances

This section will cover the application of wind to our dynamic model. An earlier section discussed that wind is a statistically applied process wherein it is assumed that it is componentized into two parts: a steady constant velocity and a gusting wind component. The wind vector is three-dimensional and can initially be defined in the NED coordinate frame. The ‘gust’ component is expressed in the body frame dynamics because that process is higher frequency nature. This ensures that the gust component is figured into shorter

period control loops. The Dryden transfer functions are the basis for the gust model used in this section. Where L_u, L_v, and L_w are the turbulence intensities given by the Dryden model parameter table as follows:

Table 3

Dryden Gust Parameters for various altitudes

Wind Gust description	*Altitude (m)*	$L_u = L_v$	L_w	$\sigma_u = \sigma_v$	σ_w
Low altitude, light turbulence	50	200	50	1.06	.07
Low altitude, moderate turbulence	50	200	50	2.12	1.4
medium altitude, light turbulence	600	533	533	1.5	1.5
medium altitude, moderate turbulence	600	533	533	3.0	3.0

The equations for the model in the vehicle body frame are given below:

$$H_u(s) = \sigma_u \sqrt{\frac{2V_a}{L_u}} \frac{1}{s + \frac{V_a}{L_u}} \tag{2.32}$$

$$H_v(s) = \sigma_v \sqrt{\frac{2V_a}{L_v}} \frac{\left(s + \frac{V_a}{\sqrt{3}L_v}\right)}{\left(s + \frac{V_a}{L_v}\right)^2} \tag{2.33}$$

$$H_w(s) = \sigma_w \sqrt{\frac{2V_a}{L_w}} \frac{\left(s + \frac{V_a}{\sqrt{3}L_w}\right)}{\left(s + \frac{V_a}{L_w}\right)^2} \tag{2.34}$$

The body frame airspeed vector is given as:

$$V_a^b = \begin{pmatrix} u_r \\ v_r \\ w_r \end{pmatrix} = \begin{pmatrix} u - u_w \\ v - v_w \\ w - w_w \end{pmatrix} \tag{2.35}$$

From the body frame vector, the airspeed, angle of attack, and sideslip angle are modified to include the new wind airspeed vector:

$$V_a = \sqrt{u_r^2 + v_r^2 + w_r^2} \tag{2.36}$$

$$\alpha = \tan^{-1}\left(\frac{w_r}{u_r}\right) \tag{2.37}$$

$$\beta = \sin^{-1}\left(\frac{v_r}{\sqrt{u_r^2 + v_r^2 + w_r^2}}\right) \tag{2.38}$$

These expressions are crucial for calculating the aerodynamic forces on the PPC model. Airspeed, angle of attack, and sideslip are contained within the calculations for aerodynamic lift, drag, and moment.

2.10 Aerodynamic Stability Coefficients

The aerodynamic stability coefficients of the PPC are defined as follows and describe the dynamics of a specific airframe.

Table 4

Aerodynamic Stability Coefficients descriptions

Coefficient	Description	Stability Value Sign
C_{m_α}	Airfoil moment pitch stability	-
C_{l_β}	Roll stability correlates to dihedral effect of a fixed wing aircraft	-
C_{n_β}	Yaw stability (weathervane tendency in wind)	+
C_{m_q}	Pitch damping, decreases angular rate of pitch	-
C_{l_p}	Roll damping	-
C_{n_r}	Yaw damping	-
$C_{n_{\delta A}}$	Airbrake primary control derivative (acting in yaw axis)	N/A
$C_{l_{\delta A}}$	Airbrake cross control derivative (adverse roll affect)	N/A

These parameters are usually defined as constants for typical autopilot controller development. However, they will become critical within future sections on adaptive control development. It is possible to tune these parameters better to match the dynamic model with actual prototype dynamic responses.

2.11 Chapter Summary

The kinematics and dynamics can be accumulated in the following equations within this section in an order that is typical for use within a simulation loop. An initial calculation

of the data describing airspeed, angle of attack (α), and sideslip angle (β) are required using the wind vector velocity components.

$$V_a = \sqrt{u_r^2 + v_r^2 + w_r^2} \tag{2.36}$$

$$\alpha = \tan^{-1}\left(\frac{w_r}{u_r}\right) \tag{2.37}$$

$$\beta = \sin^{-1}\left(\frac{v_r}{\sqrt{u_r^2 + v_r^2 + w_r^2}}\right) \tag{2.38}$$

An additional calculation for the lift and drag coefficients is made before continuing. These coefficients are functions of the angle of attack and are rotated to the body reference frame. The following two equations are second-order polynomial fit equations and use coefficients derived from this chapter's fluid analysis.

$$C_L(\alpha) = C_{L_1}\alpha^2 + C_{L_2}\alpha + C_{L_3} \tag{2.39}$$

$$C_D(\alpha) = C_{D_1}\alpha^2 + C_{D_2}\alpha + C_{D_3} \tag{2.40}$$

The following equations are the rotations for the lift and drag coefficients to a body reference frame:

$$C_x(\alpha) = -C_D(\alpha)\cos\alpha + C_L(\alpha)\sin\alpha \tag{2.41}$$

$$C_{x_q}(\alpha) = -C_{D_q}\cos\alpha + C_{L_q}\sin\alpha \tag{2.42}$$

$$C_z(\alpha) = -C_D(\alpha)\sin\alpha - C_L(\alpha)\cos\alpha \tag{2.43}$$

$$C_{z_q}(\alpha) = -C_{D_q}\sin\alpha - C_{L_q}\cos\alpha \tag{2.44}$$

The forces and moments that act on the airframe are calculated using updates from the equations (2.36), (2.37) and (2.38), and include the commanded input controls for airbrake and throttle. The generalized force equation is given as follows.

$$\begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix} = \begin{pmatrix} -mg\sin\theta \\ mg\cos\theta\sin\phi \\ mg\cos\theta\sin\phi \end{pmatrix} + \frac{1}{2}p\tilde{V}^2 S_{chute} \begin{bmatrix} c\alpha & 0 & -s\alpha \\ 0 & 1 & 0 \\ s\alpha & 0 & c\alpha \end{bmatrix} + \frac{1}{2}p\tilde{V}^2 S_{body} C_{D,f} \begin{pmatrix} u \\ v \\ w \end{pmatrix} + \frac{1}{2}\rho S_{prop} C_{prop} \begin{pmatrix} (k_{motor}\delta_t)^2 - V_a^2 \\ 0 \\ 0 \end{pmatrix} \tag{2.45}$$

This yields the following result upon simplification of the force components to include the rotation to the body reference frame:

$$f_x = -mg\sin\theta + \frac{1}{2}p\tilde{V}^2 S_{chute}\left(C_x(\alpha) + C_{x_q}(\alpha)q/(2V_a)\right) + \frac{1}{2}\rho S_{prop} C_{prop}\left((k_{motor}\delta_t)^2 - V_a^2\right) \tag{2.46}$$

$$f_y = mg\cos\theta\sin\phi + \frac{1}{2}p\tilde{V}^2 S_{chute}\left(C_{Y\beta}(\alpha)\beta + C_{Y\delta A}\delta A\right) \tag{2.47}$$

$$f_z = mg\cos\theta\cos\phi + \frac{1}{2}p\tilde{V}^2 S_{chute}\left(C_z(\alpha) + C_{chute}C_{z_q}(\alpha)q/(2V_a)\right) \tag{2.48}$$

Moment summary:

$$l = \frac{1}{2}p\tilde{V}^2 S_C\left(b\left(C_{l_0} + C_{l_\beta}\beta + C_{l_p}\frac{b}{2V_a}p + C_{l_r}\frac{2}{2V_a}r + C_{l_{\delta A}}\delta A\right)\right) \tag{2.49}$$

$$m = \frac{1}{2} p \tilde{V}^2 S_C \left(c \left(C_{m_0} + C_{m_\alpha} \alpha + C_{m_q} \frac{c}{2V_a} q + C_{m_{\delta A}} \delta A \right) \right. \tag{2.50}$$
$$\left. + \left(C_x(\alpha) + C_{x_q}(\alpha) q/(2V_a) \right) * H_{chute} \right)$$

$$n = \frac{1}{2} p \tilde{V}^2 S_C \left(b \left(C_{n_0} + C_{n_\beta} \beta + C_{n_p} \frac{b}{2V_a} p + C_{n_r} \frac{2}{2V_a} r + C_{n_{\delta A}} \delta A \right) \right) \tag{2.51}$$

The state derivatives are then calculated with the forces, moments, and previous states:

Position derivatives:

$$\dot{p}_N = (\cos\theta \cos\psi) u + (\sin\phi \sin\theta \cos\psi - \cos\phi \sin\psi) v \tag{2.52}$$
$$+ (\cos\phi \sin\theta \cos\psi + \sin\phi \sin\psi) w$$

$$\dot{p}_E = (\cos\theta \sin\psi) u + (\sin\phi \sin\theta \sin\psi + \cos\phi \cos\psi) v \tag{2.53}$$
$$+ (\cos\phi \sin\theta \sin\psi - \sin\phi \cos\psi) w$$

$$\dot{p}_D = -u \sin\theta + v \sin\phi \cos\theta + w \cos\phi \cos\theta \tag{2.54}$$

Velocity Derivatives:

$$\dot{u} = rv - qw + \frac{1}{M} f_x \tag{2.55}$$

$$\dot{v} = pw - ru + \frac{1}{M} f_y \tag{2.56}$$

$$\dot{w} = qu - pv + \frac{1}{M} f_z \tag{2.57}$$

Angular Position Derivatives:

$$\dot{\phi} = p + q \sin\phi \tan\theta + r \cos\phi \tan\theta \tag{2.58}$$

$$\dot{\theta} = q \cos\phi - r \sin\phi \tag{2.59}$$

$$\dot{\psi} = q \sin\phi \sec\theta + r \cos\phi \sec\theta \tag{2.60}$$

Angular Rate Derivatives:

$$\dot{p} = \Gamma_1 pq - \Gamma_2 qr + \Gamma_3 l + \Gamma_4 n \tag{2.61}$$

$$\dot{q} = \Gamma_5 pr - \Gamma_6 (p^2 - r^2) + \frac{1}{J_y} m \tag{2.62}$$

$$\dot{r} = \Gamma_7 pq - \Gamma_1 qr + \Gamma_4 l + \Gamma_8 n \tag{2.63}$$

These twelve state equations and the forces and moments calculations become the basis for the simulation of dynamics and future controller development within the next chapters. The following Chapter will validate this non-linear model and move towards linearization with the goal of controller development for autonomous control.

CHAPTER 3

MODEL SIMULATION

The PPC aerodynamics from the previous section are 6 DOF and 12 states with many non-linearities in both the model itself and the calculation of coefficients of lift and drag, which change depending on airspeed. This section aims to discuss results of the non-linear dynamic simulation with model verification and validation steps and develop an autonomous controller implemented within the MATLAB environment and capable of being deployed on existing flight controller hardware. To simplify work, linearizing the previous model will also be performed in this chapter.

3.1 Non-Linear Simulation

This section focuses on the 6 DOF simulation developed and visualized in MATLAB Simulink using the dynamics from Chapter 2. The simulation detail within this section will be used for developing linearization and trim routines. Furthermore, this simulation will use the plant from chapter two to test controller development and as a model reference for desired flight characteristics in subsequent chapters.

The simulation comprises of three parts: input definition, physics simulation, and output visualization. The controllable inputs for the PPC are airbrake deflection and throttle setting; the wind model is the Dryden Gust model random component.

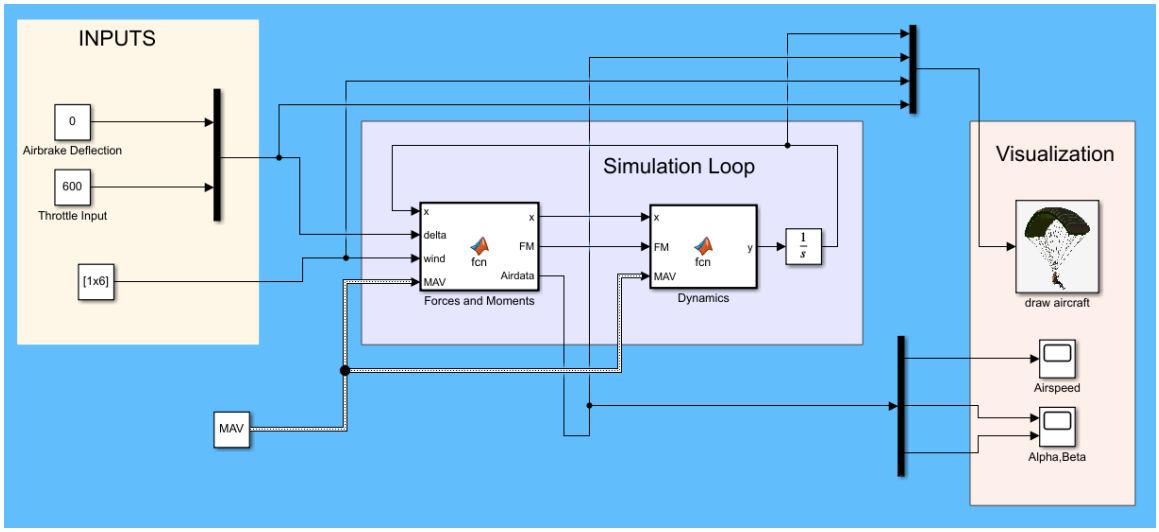

Figure 14: PPC Dynamic Simulation Block Diagram

The visualization block animates a powered parachute model using updates on the position and rotational states. Simulink has native blocks to handle visualization and the physics portion; however, for consistency, these functions were defined directly from the equations summarized at the end of Chapter 2 of this document.

The following image displays simulation data from an uncontrolled run with a constant throttle setting and a slight airbrake deflection to initiate a turn to the right. The PPC flies in a circle because of the airbrake deflection, and it settles into a constant airspeed and angle of attack. The sideslip angle graph results from a combination of the yaw component of the airbrakes and the steady-state wind.

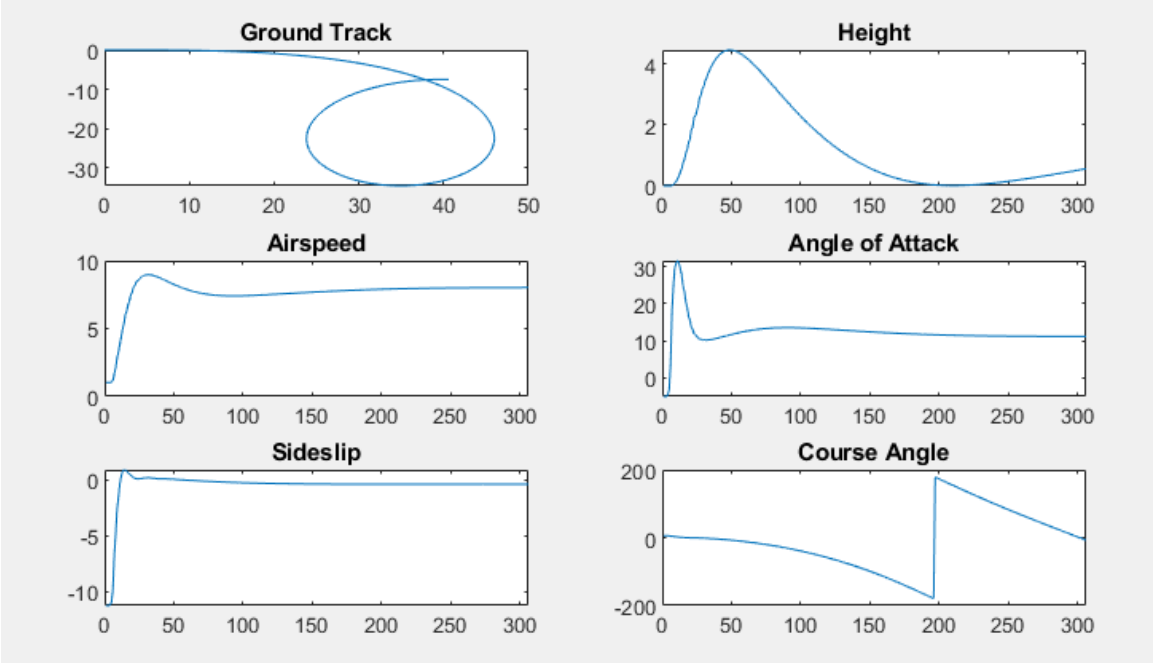

Figure 15: Basic Simulation Result Plot

The simulation validates the assumptions on PPC flight from Chapter 2. The airspeed and angle of attack should settle into a steady-state condition, and additionally, when an airspeed over the steady-state line was present, a climb was initiated. Trim and linearization routines can now be performed with the dynamics and parameters simulated and validated.

3.2 Discussion on Dynamic Simplification and Linearization

The lateral (horizontal) and longitudinal (vertical) dynamics of the PPC are coupled loosely in chapter 2 non-linear model. The coupling occurs via two mechanisms. First, the induced adverse roll results from the application of airbrakes. Second, a roll component is

present in the opposite direction from the torque generated at the propellor. Both couplings of the lateral and longitudinal dynamics are within the flight roll-axis. However, it is essential to note that the roll axis is not directly controllable in a standard configuration PPC vehicle. In other words, there are no control inputs to change the paramotor's roll angle directly. It is important to note that all of the induced roll mechanisms in a standard PPC are unwanted secondary effect. Additionally, these roll mechanisms can be strongly mitigated with the addition of attention to specific design characteristics. For example, the amount of torque roll can be lessened with careful application of the thrust input so that the torque effect is spread over more time or eliminated with a second motor or counter-rotating propellor. Adverse roll from the canopy spoilers can also be mitigated by adding Finally, a crosswind induced roll can be reduced by matching the lateral aerodynamic forces of the canopy with the lateral forces that would be expected on the airframe, likely with that same addition of a vertical stabilizer surface.

In a simplified model, the lateral movement of the paramotor is dependent on the deflection of the wingtip air brakes and is also stable, i.e., the craft will not continue a turn so long the vehicle has been trimmed and the control inputs are removed. In the longitudinal model, the climb rate is prescribed as linearly proportional to the amount of thrust applied. This is due to the overarching assumption seen in multiple papers and experiments that the parafoil will correct its nominal angle of attack, α, for the parafoil, dependent on the wing loading of the vehicle.

It should be further noted that the aerodynamic turning mechanism for lateral control of the paramotor is significantly different than that of a typical fixed wing aircraft, as the paramotor is operating in a skid-steer manner and not using a coordinated turning

system. A coordinated turn is desired to produce maximum stability against stall and spin dynamics in a fixed-wing aircraft. For a PPC, however, the ability for the airframe to be caught in a significant stall or spin condition is directly correlated to the amount of control input applied to the wing tip brakes. These distinctions are important for robust controller design. The linearization point must be selected far away from the stall point on a fixed-wing aircraft to avoid non-linear dynamics. In the PPC system, the linearization point and actual flight envelope depend significantly on the overall weight of the airframe and wing loading of the parafoil.

3.3 Trim Conditions

The PPC is assumed to be flying at a constant altitude, level flight, and with constant throttle input so that all states are unchanged except for the positional vectors. The two control vectors are also in steady-state such that the throttle control is constant, and the steering control is only applied to counteract an unknown wind component. The three desired trim points for maximal control of the PPC are dictated within the following flight characteristics:

1. Flight along the desired vector
2. Climb and descent at a constant rate
3. Constant turn of the desired radius about a fixed point.

For a PPC aircraft, the 12 states are given by:

$$x \triangleq (p_n, p_e, p_d, u, v, w, \phi, \theta, \psi, p, q, r)$$

With the following inputs:

$$u \triangleq (d_T, d_A)$$

The states P_N, P_E, P_D are independent of the trim conditions.

These simplified longitudinal equations are presented:

$$\sum F_x = \ddot{x} = \frac{1}{m}\left[T_x - D_{chute_x} - L_x - D_{fuse_x}\right] \tag{3.1}$$

$$\sum F_z = \ddot{z} = \frac{1}{m}\left[W + D_{chute_z} + D_{fuse_x} - L_z - T_z\right] \tag{3.2}$$

$$\begin{aligned}\sum M_{CG} = \ddot{\theta} \\ = \frac{1}{I}\big[C_x D_{chute_z} + C_z D_{chute_x} + C_z L_x - C_x L_z \\ - M_x T_z - M_z T_x\big]\end{aligned} \tag{3.3}$$

With the following definitions:

$$\begin{aligned} T_x &= T\cos(\theta - \beta) & T_z &= T\sin(\theta - \beta) \\ L_x &= L\sin\gamma & L_z &= L\cos\gamma \\ D_{fuse_x} &= D_{fuse}\cos\gamma & D_{fuse_z} &= D_{fuse}\sin\gamma \\ D_{chute_x} &= D_{chute}\cos\gamma & D_{chute_z} &= D_{chute}\sin\gamma \end{aligned} \tag{3.4}$$

Lastly, the following distances are defined for moment calculations:

$$C_x = (l + h)\sin\theta \quad C_z = (l + h)\cos\theta \tag{3.5}$$
$$M_x = h\sin\theta \quad M_z = h\cos\theta$$

These dynamic state equations and forces calculations provide the basis for simulations and controller design in all subsequent chapters.

CHAPTER 4

APPLICATION OF MODEL REFERENCE CONTROL

One of the challenges with implementing the dynamics from Chapter 2 is that many dynamic state updates received within the control loop are inexact. Additionally, deviations between an actual prototype and the dynamic model developed can result in discrepancies within a linear feedback controller. Therefore, the preliminary flight testing was conducted on the PPC prototype by tuning individual flight characteristics with trial and error. One strategy for improving these methods is to use an adaptive controller that continuously updates control parameters. Adaptive Model Predictive Control (AMPC) algorithms are used because they can increase the overall accuracy of dynamic controllers, increase robustness and stability of controllers, or in some instances, estimate parameters that could be difficult to calculate. One example is how the lift coefficient of a wing surface may change depending on altitude, control inputs, or experimental error.

There are various methods for applying adaptive control to dynamic systems, the algorithm presented in this paper is a variation of AMPC. An AMPC controller generates predictions of current model conditions to adapt to specific desired parameters. A method that falls under this category of AMPC is called Model Reference Adaptive Control (MRAC). MRAC compares a simulated dynamic model with constantly updating parameters against an idealized dataset, a commanded flight path, and updates the model coefficients by optimizing for the least error between the output and reference [17]. A successful application of MRAC will show that this calculated error is minimized as the controller adapts and improves the control coefficients. Ideally, this error should eventually reach zero, and typically some minimum threshold is set to stop the optimization routine

and hold the model coefficient values. The principle of the AMPC algorithm is described in the following figure:

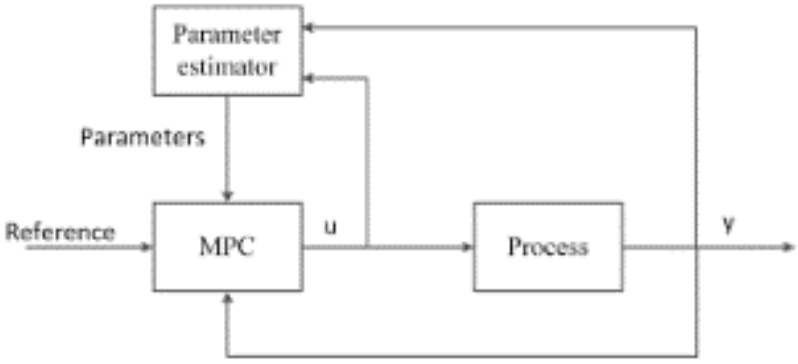

Figure 16: Adaptive Model Predictive Control Block Diagram Overview

When an uncontrolled and trimmed simulation model is compared to recorded flight data, this method can be used to 'search' for the unknown aerodynamic coefficients against an analytical model. This method also becomes increasingly essential for time-varying systems, such as internal combustion engine vehicles, which lose fuel mass during operation, potentially altering control responsiveness and stability. In a typically fixed-parameter system like an electric-powered paramotor, the parameters could be set to 'fix' overall control rates after controller convergence or alternatively left to run indefinitely to account for varying environmental conditions, which could have minor effects on the aerodynamic properties of the aircraft. The MRAC model utilized within this section updates a PID controller's coefficients, and the model reference is a smoothed ramp function to drive the PPC to change altitudes over time. An updated figure of the Simulink model diagram showing an altitude controller implementation within the previous simulation is provided below:

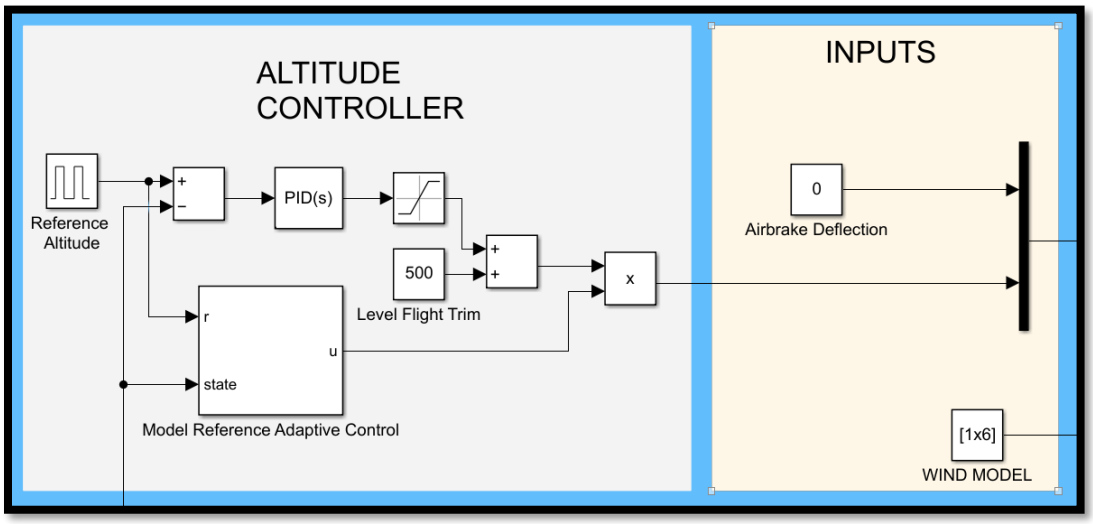

Figure 17: MRAC Block Diagram with PID Control

The MRAC controller operates in parallel with the standard PID controller but optimizes the internal weighting functions as time goes on. The cost function reduces the error between the plant output (PPC flight path) and the internal reference model. The desired input is summed into a transfer function describing a paramotor with ideal flight characteristics and compared against the non-linear plant with the dynamics from Figure 17. This error output is multiplied by some learning parameter gamma, which is used to calculate, θ, the PID control weight.

The MRAC subsystem block is shown in the following figure:

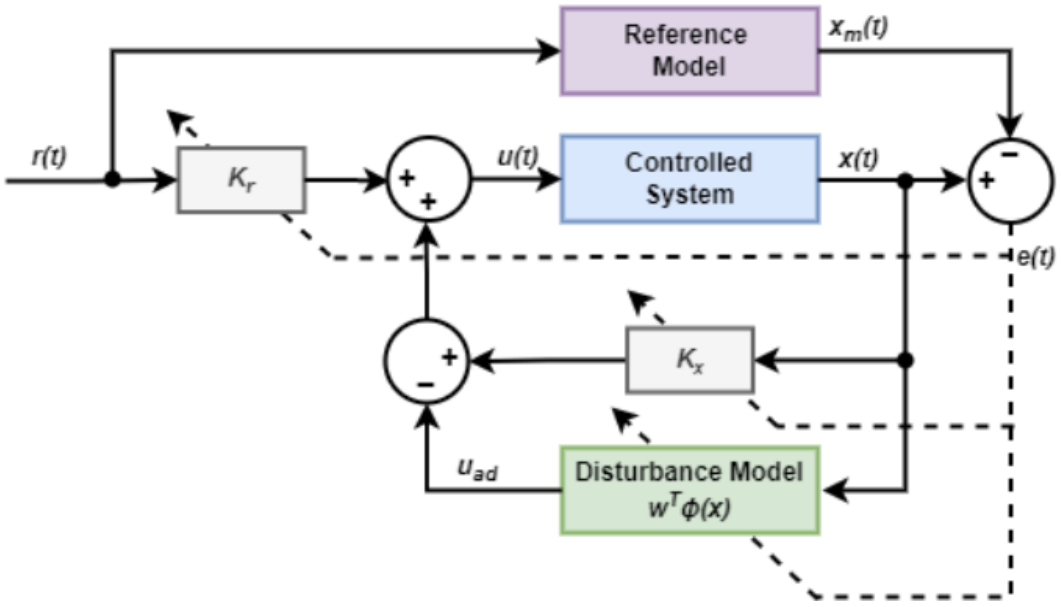

Figure 18: MATLAB Diagram of MRAC Control (https://www.mathworks.com/help/slcontrol/ug/model-reference-adaptive-control.html)

Within the MRAC subsystem block, the controller computes the error between the simulated reference model and the controlled system plant, where the MRAC controller computes the control signal as follows:

$$u(t) = k_x x(t) + k_r r(t) - u_{ad} \tag{4.1}$$

$$u_{ad} = w^T \varphi(x) \tag{4.2}$$

The nominal reference model is given as the state-space linear system with desired control matrices A and B:

$$\dot{x}(t) = Ax(t) + B(u(t) + f(x)) \tag{4.3}$$

The controller weights are adjusted and updated by the following equations:

$$\dot{k_x} = \Gamma_x x(t) e^T(t) PB \tag{4.4}$$

$$\dot{k_r} = \Gamma_r r(t) e^T(t) PB \tag{4.5}$$

$$\dot{w_x} = \Gamma_w \Phi(x) e^T(t) PB \tag{4.6}$$

The adaptive process utilizes a parameter, w_x, for the learning rate or the rate at which the parameter is allowed to update. A lower gamma will take longer for convergence as the step size is reduced. The optimization routine selects a new theta per iteration based on the LMS algorithm for simplicity. The adaptive portion of this algorithm adds a small amount of control loop calculation time equivalent to the error calculation LMS search. The adaptive controller will fail to converge with a step size too large and instead 'bounce' around the trim point. In real-world testing, too slow of convergence could immediately result in aircraft instability; this requires the ability to determine a good guess of initial conditions.

This simplified adaptive controller does not consider other errors in the modeling of aerodynamic coefficients or controller inputs. This method forces the PID controller to fit with the plant to achieve the desired dynamics. The result is that the controller weight is fit against an imperfect dynamic model to match the desired characteristics. In addition, the rate at which controller weights update will dictate overall system stability.

4.1 Simulating the Adaptive Controller to Estimate Optimal PID Values

The model of the paramotor is validated by commanding a step input to the commanded altitude to test the longitudinal control of the paramotor. Large oscillations and typical controller-plant behavior for a non-optimally tuned system are displayed. The given parameters for the commanded altitude are $T = 10,\ altitude = square(T, .5)$ where the setpoint altitude is a square function computed within MATLAB.

The simulation of the algorithm presented in the previous section initializes a periodic step in altitude to demonstrate longitudinal control, alternating between a high and low commanded altitude in the controller. With a fixed controller, the error stays constant and similar amount of flight path deviation would be seen for every command altitude change. An additional amount of band-limited noise summed into the system simulating random deviations that could be seen in the environment and measurement-based errors. The simulated flight path of the paramotor is shown compared against the linearized reference model. The simulation of the MRAC altitude control was initiated by generating a source of reference to fly the PPC to repeatedly, in this case alternating the controller altitude between 0 and 5 meters for a period of 20 seconds. The following Figure 19 compares the driven altitude (with noise addition), how the reference model responds to that control signal, and finally, the simulated altitude of the PPC when the MRAC algorithm is applied.

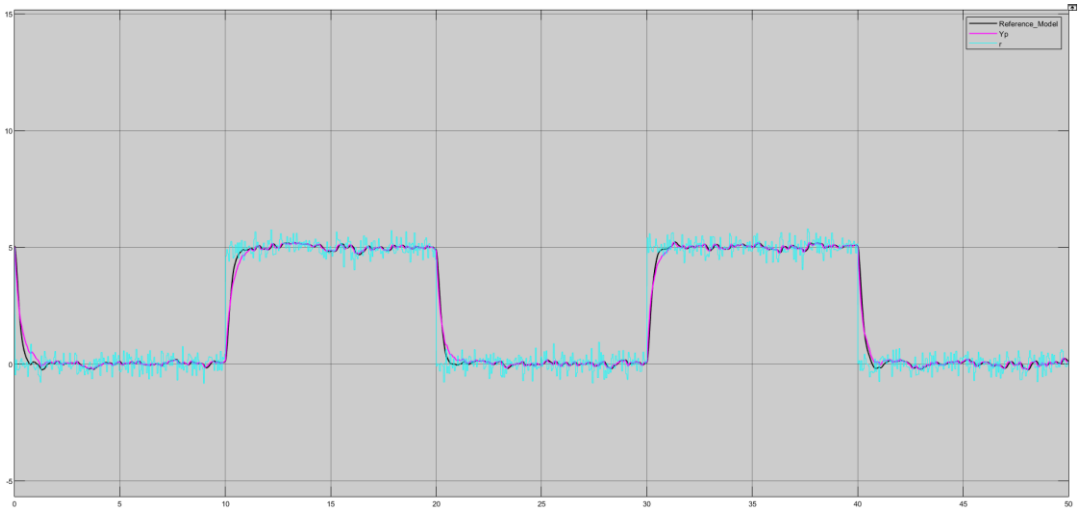

Figure 19: Reference Signal, Plant Output, Error

The difference error between the reference model and the plant shows the ability of the system to adapt and optimize the control coefficient. This error is converging to a lower level seen in the image below.

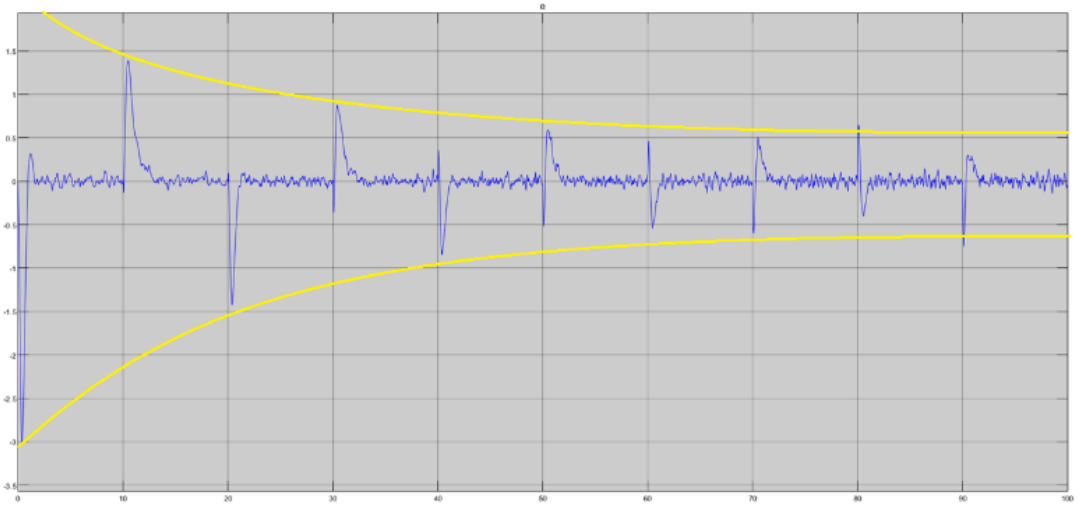

Figure 20: Convergence of PID Flight Control Parameters (Error Convergence)

The simulation reaches a steady-state error over 100 seconds, reducing the error between commanded flight and the desired flight path. The adaptive model controller shows it can tune the weights of the PID controller for the paramotor model to improve the path following error between the simulated paramotor and the ideal flight path. Furthermore, the error converged as the paramotor as the adaptive controller retrieved more system information over time, improving the initial estimations for PID coefficients. Future work on this project could include a method of storing retrieving previously converged solutions, potentially increasing the speed of the algorithm control loop processing time up and parameter convergence.

CHAPTER 5

NEURAL NETWORK AUTOPILOT CONTROLLER DESIGN

This chapter will focus on developing additional autopilot control algorithms for the PPC design. In the last chapter, a simplified dynamics model was presented using an MRAC controller to provide longitudinal control of the PPC aircraft. This chapter will further extend the control to capture 6 DOF simulation of model reference adaptive control and present a control method using a neural network as an adaptive autopilot.

5.1 Trim Conditions

During the development of control methods for this chapter, it is desired that a state of controls is selected for which all forces are balanced or 'trimmed'. The trim state of a PPC is predicated on several factors including weight, battery voltage, and environmental conditions (discounting wind). For this chapter, trim will be defined as the combination of inputs for throttle and airbrake to control the PPC in straight and level flight. The trim conditions of the paramotor should be given for desired navigational maneuvers of the aircraft.

1. The PPC is climbing, descending, or level at a constant rate $\frac{dH}{dt} = Constant$
2. The PPC is flying in an orbit of constant radius

In the last chapter, it was shown that small control inputs could be assumed linear operating about the trimmed state of the PPC. In typical operation, PPC can navigate using only a basic tuned PID controller to steer course angle and adjust altitude for flight. Sudden upsets by wind challenge this linear control assumption. Linear controller design can cause

the PPC, in some instances, to exceed the margins for stable flight. One method for improving the robustness of a control system past the linear controller is to use a Neural Network Adaptive controller design.

5.2 Introduction to Neural Network Adaptive Controllers

This section will apply a Neural Network, NN, to adaptively control a PPC dynamic system. This method involves using the NN to capture unknown dynamic properties through a training method, modifying coefficient weights within the NN. The NN output is then routed to the plant control inputs. This process generates an adaptive control rate modifier for the system, learning and adapting to provide a more optimal control over time.

The following image displays the architecture of a dynamic control loop containing a traditional PID controller with an output that is modified by a trained neural network. When this method is applied, the CNN captures information from current and past vehicle states and modifies the PID controller output. An adaptive learning control using a neural network could help the performance of sUAS in-flight maneuvers and, in return, the performance of the vehicle within its mission. The following figure is a block diagram for the Neural Network dynamic controller proposed in this section.

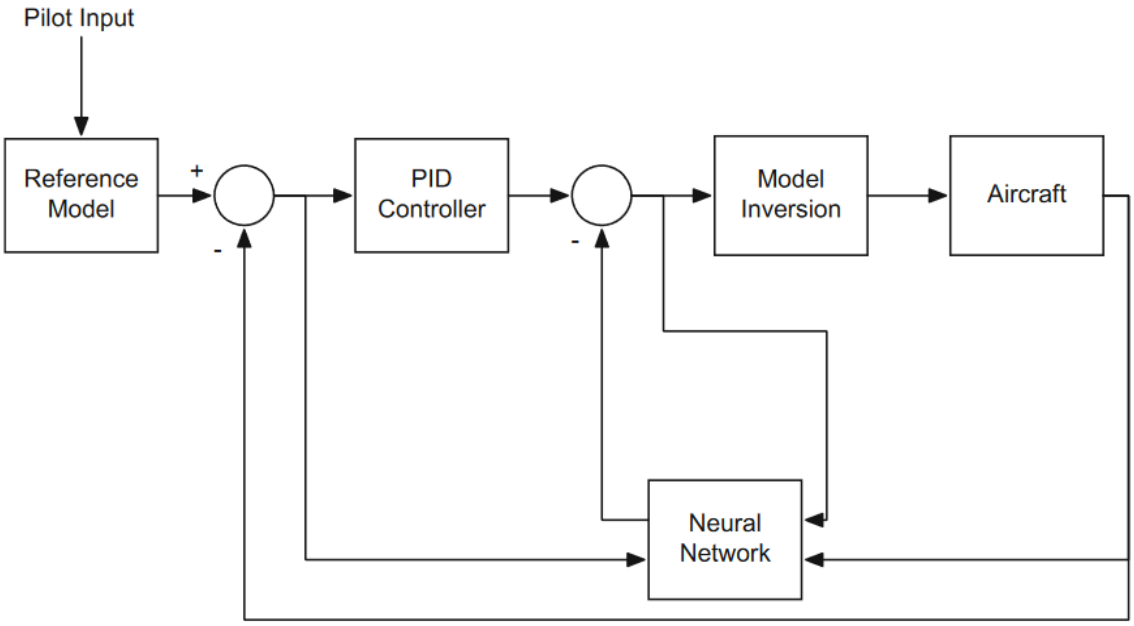

Figure 21: Neural Network Adaptive Control Block Diagram

In comparison with the MRAC control from the previous chapter, inconsistencies in the reference model and the inability of the MRAC to effectively capture a dynamic response history could result in poorer performance when compared to a neural network adaptive controller. Previously, the reference model that is selected assumes the plant will be able to operate within the constraints of that reference model, so knowledge of the system dynamics is necessary. Additionally, the use of a neural network controller could further improve the ability of the controller to respond to adverse conditions over other types of adaptive control like MRAC, for example: loss of a motor, abrupt changes in environmental conditions, or partial parafoil stall or collapse.

5.3 Activation Function

The neural network uses an 'activation function' mimicking a neuron or similar process. The activation function for the simulations within this chapter were performed by defining a sigmoid function to act as the neuron. The sigmoid is among the common functions for this purpose.

$$s(x) = \frac{1 - e^{-kx}}{1 + e^{-kx}} \tag{5.1}$$

In this function, k, is applied as a constant scaling unit modifier.

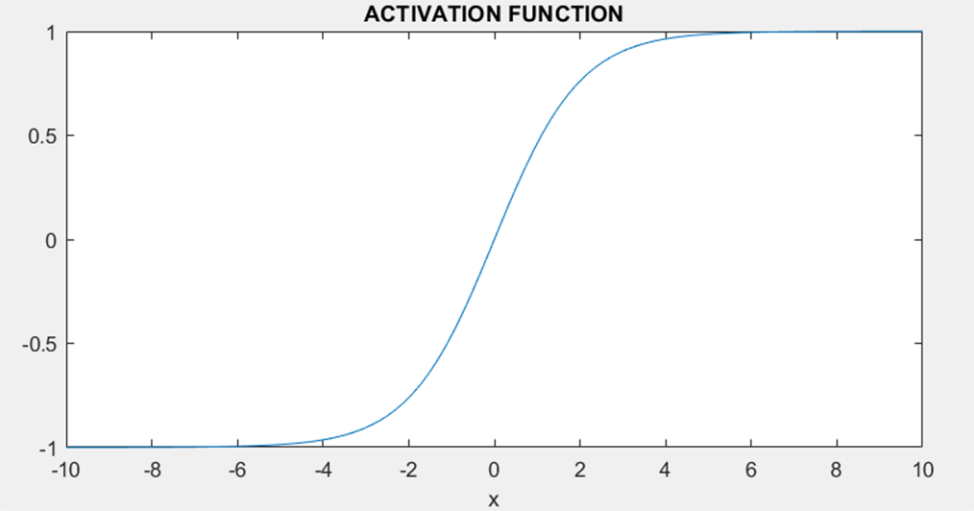

Figure 22: Sigmoid Activation Function Example Plot

The activation function bounds and scales the neuron output. Various activation functions are used within neural networks, both sigmoid and alternative functions, but this was not further explored within the scope of this chapter.

5.4 Sigma-Pi Function

This section will introduce the Sigma-Pi function application into the dynamic control loop for the PPC. As shown in Figure 21, the sigma-Pi neural net is inserted into the control loop for the PPC after the PID control and provides weighting functions to modify the control input into the plant. The neural network input is designed to combat model uncertainties and additional non-linear functions. Specifically, the Sigma-Pi neural network used for this chapter has a mechanism for weighting the input states individually and in combination. This is further detailed in the following Figure 22.

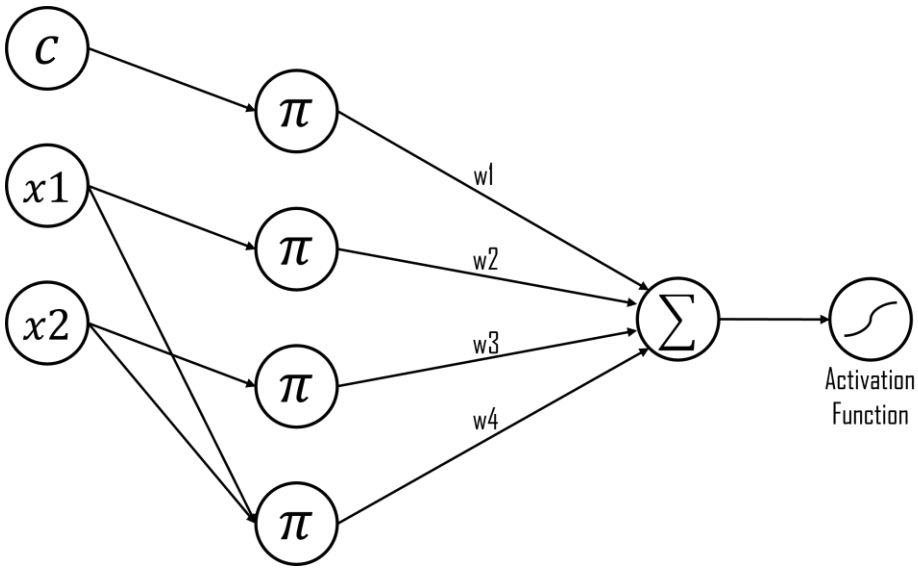

Figure 23: Sigma-Pi Neural Network Diagram

For implementation to the PPC, we want to design our neural network to modify thrust application, specifically during climb or descent maneuvers during strong wind or similar environmental conditions. The neural network's output will be a weight applied to

the thrust controller, and the inputs are defined as the external acceleration applied by the environment. The weight function from the previous figure are shown in equation form:

$$y = w_1 c + w_2 x_1 + w_3 x_2 + w_4 x_1 x_2 \tag{5.2}$$

In the case of the PPC control design, it would be useful to feed the CNN controller with information on thrust setting, airspeed (for calculation of dynamic pressure), angle of attack, and pitch angle. The number of inputs will grow the CNN weights by the following equation:

$$J = \frac{n!}{(n-k)!\,k!} \tag{5.3}$$

For a given n inputs number of inputs, the number of weights grow significantly, which is a practical memory limitation for this type of network.

5.5 Integration and Simulation of Neural Net to PID Controller

This section is dedicated to integrating the Sigma Pi Neural Net algorithm with the PPC dynamic system. During a recursive learning routine, the weights of the system are updated by the previously described equations until they are converged. For example, the following figure shows the result of recursively training the Sigma-Pi NN.

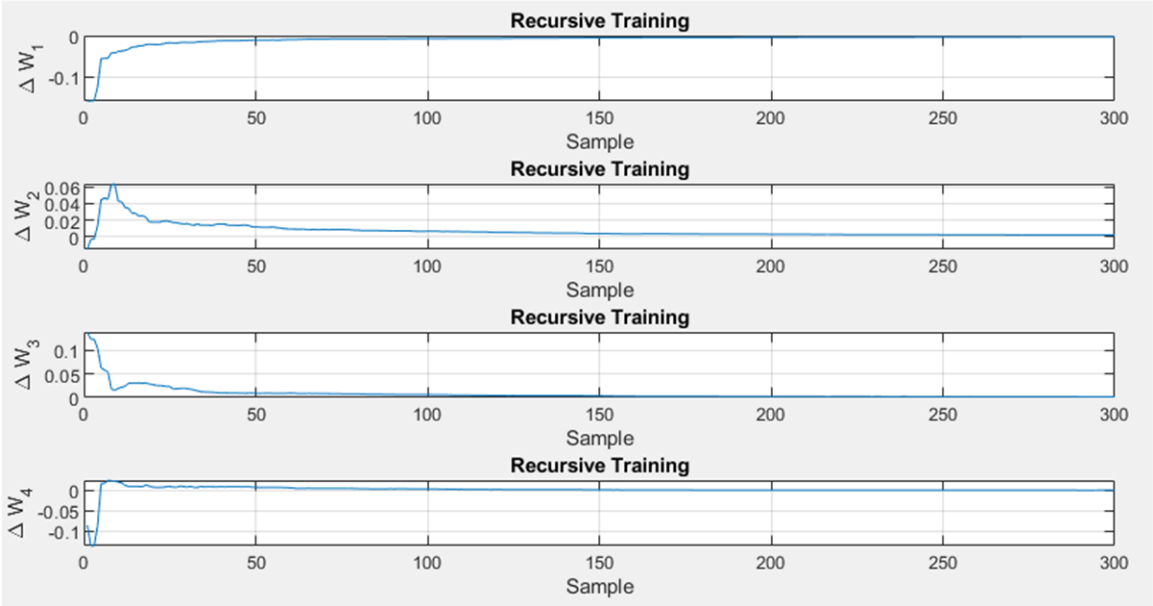

Figure 24: Example of NN Weight Convergence

The simulation within this chapter was run using only two inputs for the NN. The thrust force applied by the PPC's motor/propeller directly drives the altitude setpoint that the controller is designed to drive the system to. The second input that was simulated for the NN was the pitch angular rate of the PPC, $\dot{\omega}$. This was selected as determined from [5][8], where there exists non-linearities in the pitch dynamics where the PPC is flying is rapidly approaching larger angles of attack. It is also necessary to anticipate and 'catch' a pitching PPC aircraft to drive to the desired altitude from flight testing. A NN applied in this way could potentially assist the PID controller even if the PID coefficients were improperly tuned.

A recursive learning training strategy is applied from the procedure within Figure 20. This recursive learning algorithm creates a training set of data from known inputs and outputs. Additionally, the training continues to update weights as new data is compounded. The initial data is collected within the following equations:

$$p = (ZZ^T)^{-1} \tag{5.4}$$

$$w = pZY \tag{5.5}$$

Where Z is a matrix of known outputs within the training data, p is the training output value matrix, and w is a matrix containing the weighting coefficients. The following three equations update the weights recursively from information contained in the new input and output data. This is the algorithm within the actual runtime loop for the dynamic control.

$$p = p - \frac{pzz^T p}{1 + z^T pz} \tag{5.6}$$

$$k = pz \tag{5.7}$$

$$w = w + k(y - z^T w) \tag{5.8}$$

As the simulation progresses, the NN updates as expected, closely matching the ground truth. The following figure is an experiment to calculate the dynamic pressure data of the environment, dependent on velocity and altitude.

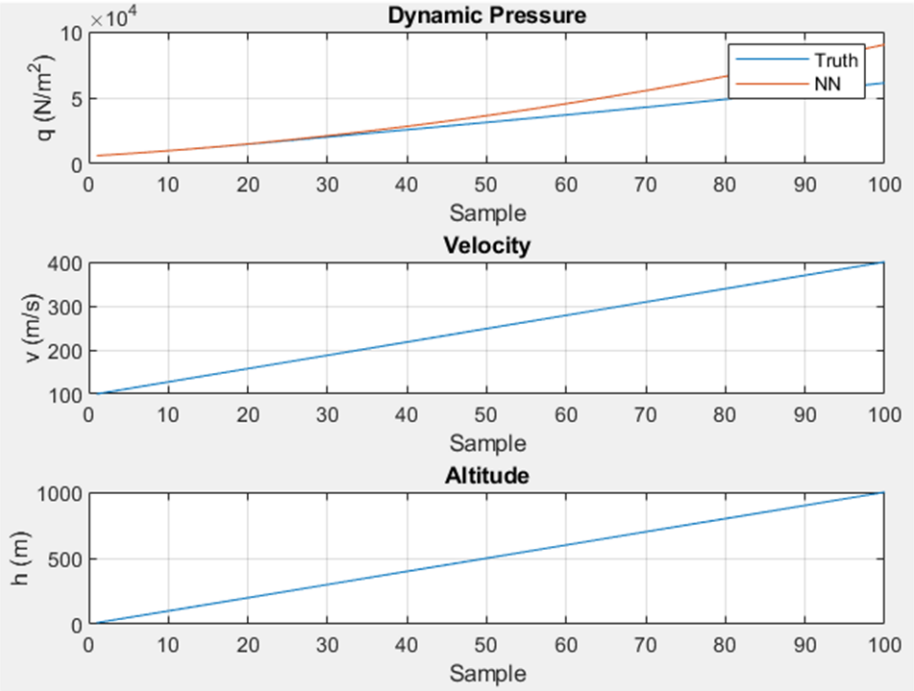

Figure 25: NN Estimation and Ground Truth with NN inputs

The NN can successfully capture and represent the dynamic pressure expected of the PPC, which ultimately contains additional terms not calculated in the simple estimation of dynamic pressure within other simulations. The PPC is dependent on dynamic pressure to calculate lift, drag, and pitching moment aerodynamics.

5.6 Results Discussion

Applying the successfully trained NN with the method from Figure 20, the NN is capable of correcting for errors in the linear PID controller. The first figure below captures the simulation path with no learning control, operating solely from PID control. The PPC is seen to overshoot the desired altitude setpoint value, then slowly return to settle at a steady state.

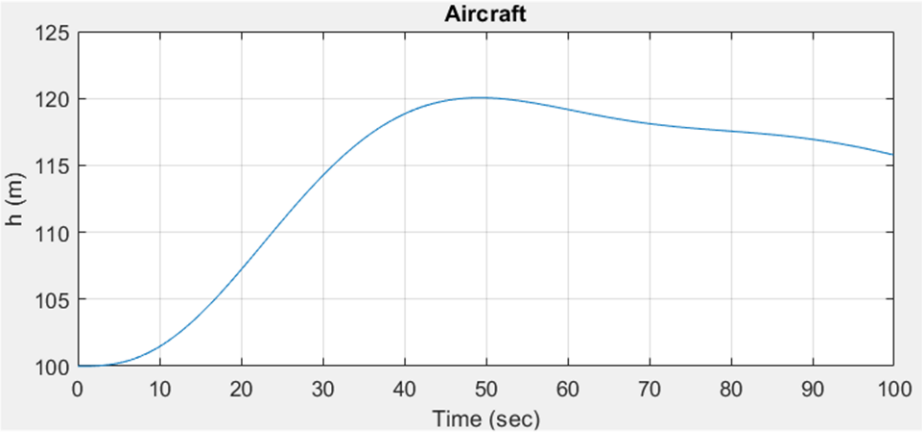

Figure 26: PID Control Performance

When NN learning control augments the PID control signal, the controller system initiates a climb to the reference setpoint without overshoot. It maintains a stable altitude until the end of the simulation.

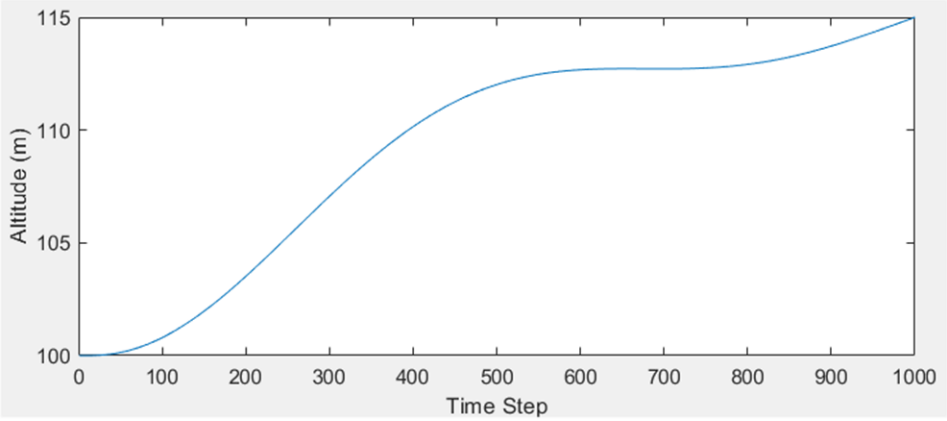

Figure 27: PID with NN Learning Adaptation Active

The NN control output generates a smoother and more robust response of the PPC. Of the three control signals developed, the PID signal change is smaller than the NN controller and the dynamics inversion control signals. It is possible that a better tuned

PID control could improve control response, but this would likely be unnecessary when learning from the NN is applied.

The PPC states that measurements from the simulation are captured in the following figure:

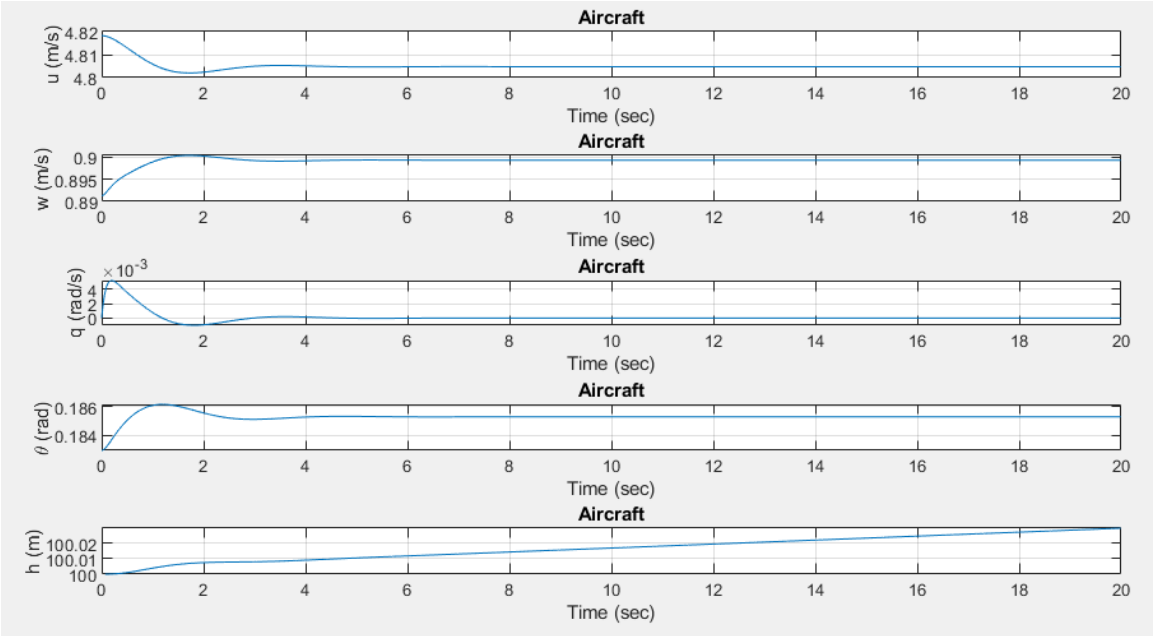

Figure 28: Simulation Results for PPC State vs Time

The PPC can successfully fly to the setpoint and maintain stable and level flight. The calculation for lift and drag convergence is also detailed:

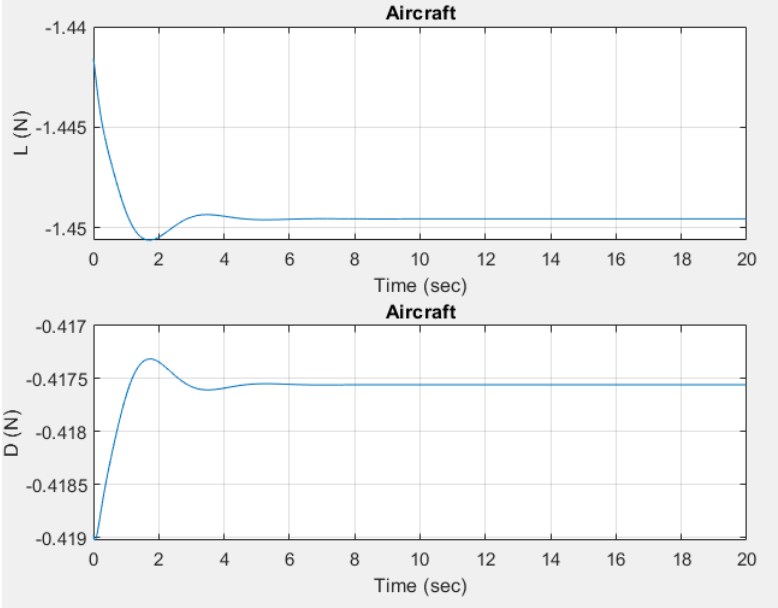

Figure 29: Lift and Drag force calculations vs Time Results (NN)

Lastly, the following two figures compare learning control values between the non-learning simulation and a simulation with the NN active.

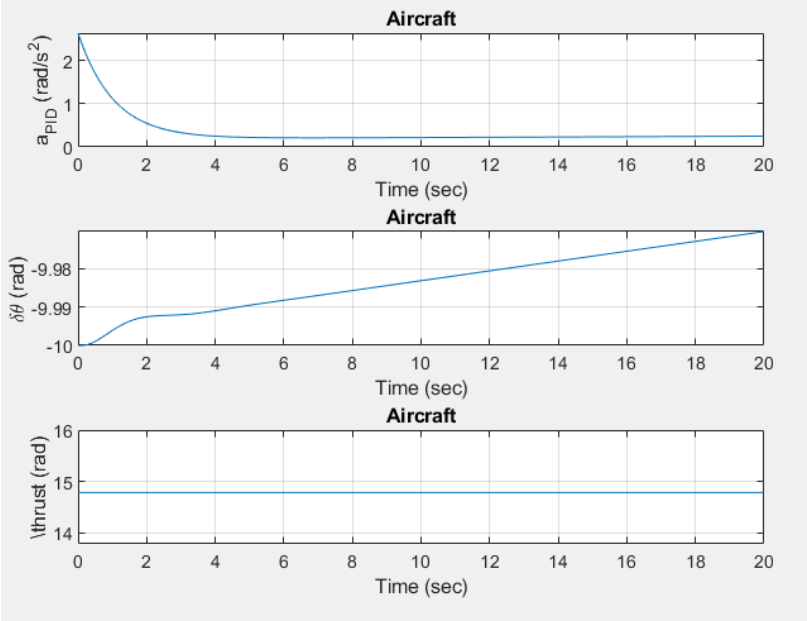

Figure 30: PID Input Only Simulation, Control Output vs Time

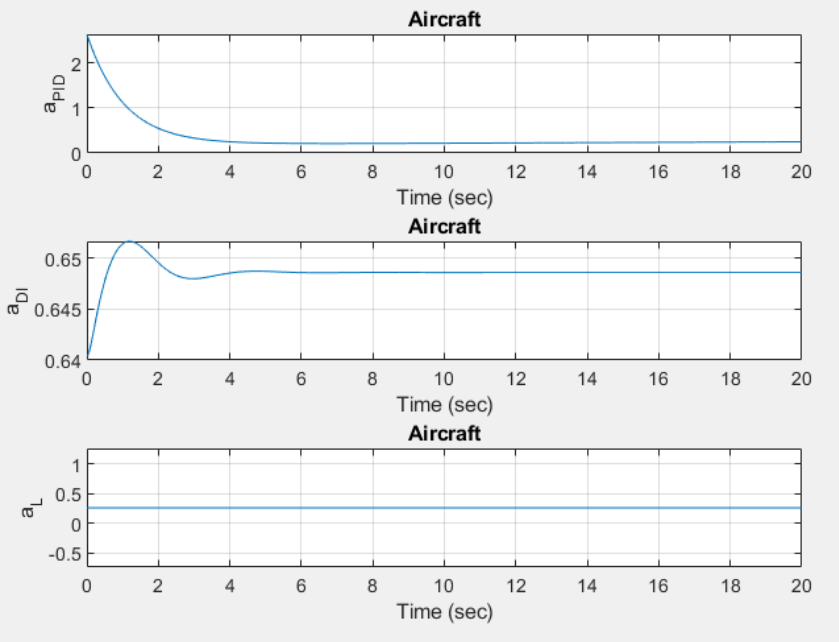

Figure 31: PID + NN + Dynamic Inversion, Control Outputs vs Time

The results show that the NN-assisted methods provide additional control inputs to fly the PPC along a more stable path. This is apparent in the comparison of Figures 26 and 27, where the PPC overshoots the setpoint by a large percentage, and in the NN-controlled simulation, the PPC flies directly to the setpoint. Furthermore, within the results, there is a minimal amount of altitude drift, this is likely associated with the learn rate selection for the simulation.

CHAPTER 6

ALTERNATIVE CONTROL SCHEME DEVELOPMENT

Many different models have been examined in detail for the traditional PPC, but alternative control methods are still being researched. The two papers which did discuss alternative control inputs proposed their new designs in Ward [10] [14] and Scheuermann [18]. Ward covers the novel idea of using weight shifting to control both lateral and longitudinal dynamics successfully and is thus far one of the only methods effectively controlling forward airspeed for a PPC without a noticeable decrease in flight efficiency. Scheuermann also proposes a novel control idea of using upper surface canopy spoilers, experimentally and numerically demonstrating that spoilers may be used for effective control and significantly improve landing accuracy.

The angle of incidence control method is also shown by Ward to increase the control envelope for glide slope in an unpowered glide and can also modify the forward airspeed for powered flight. Further examination of PPC aerodynamics shows that with an alternative design, the addition of pitch angle into the typically fixed angle thrust vector on the airframe may be able to increase or decrease the effective mass for the entire system.

6.1 Introduction to Thrust Vectoring Control

Applying a variable thrusting vector within the existing dynamic model may increase the velocity envelope within a similar range as Ward's angle of incidence control. Adding a vertical thrust component (relative to the body frame of reference) would be like modifying the effective mass by incidence angle control. A controller for thrust vectoring

could be used for driving airspeed, rate of ascent, and rate of descent in a more direct and precise method over Ward's proposed angle of incidence control. While alternative means of lateral control were not discussed in Ward's paper, a lateral thrust vectoring system could drive yaw axis control of the PPC, fix yawing moments, and reduce oscillations about the vertical axis.

Performance increases for controllable airspeed envelope and accuracy to waypoint could be substituted for the parametric on landing distances when comparing a powered parachute to an unpowered glider vehicle. In addition, the analysis same used in 6 used for developing optimal control inputs for various environments would give baseline valuable information for direct comparison to traditional models.

Paramotor control and design typically require the thrust axis to be aligned with or near the center of gravity. Maneuvers are performed as a 2-input system, where increased or decreased motor thrust results in the PPC climbing or descending, and a differential air brake input yaws the craft. Because there is a single throttle setting that will result in stable and level flight, the standard PPC will fly with a fixed forward velocity, a function of its design parameters (parafoil geometry, thrust axis, weight, and cg location). This chapter assumes that a new, controllable envelope of forward velocity is achievable during level flight by controlling the thrust axis in pitch dynamics. A downward thrust will increase the effective weight of the airframe, allowing the wing loading to increase and forward velocity to increase [1].

The mathematical model extends the work from Chapter 2 of this document. The assumption that the velocity of the paramotor will be controllable through thrust vectoring is based on the simple calculation for lift:

$$Lift = C_L \frac{\rho \cdot V^2}{2} \cdot Area \tag{6.1}$$

The dynamic model free body diagram is pictured in the following Figure 30. The coordinate plane is similarly referenced as in Chapter 2. The additional controllable input within this longitudinal model is, ε, the pitch thrust vectoring angle.

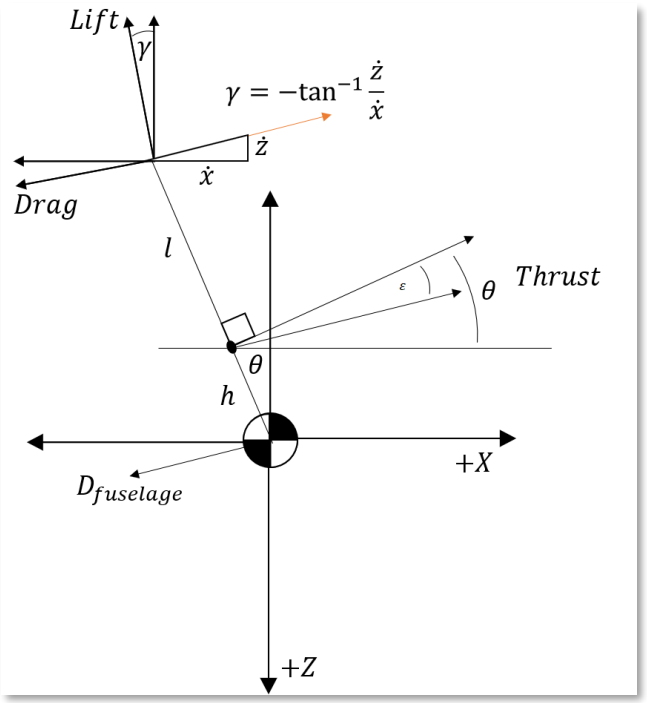

Figure 32: Thrust Vectoring Control (TVC) Free Body Diagram

In a stable and level flight condition, the wing's lift is equal to the total weight it must support. This will lead us to an equation for which we may directly control the vehicle's velocity by adjusting the equation. Where, ε, is the angle that which the thrust vectoring assembly is steered from the pitch angle of the paramotor frame and T is the

thrust produced by the motor to give the vertical component of the thrust. For straight and level flight, where: $L = W_{PPC} - T \cdot \sin(\theta - \beta)$, we can see that velocity will be a function of motor thrust and thrust angle. Although the control of forward velocity will depend on maintaining a constant horizontal thrust component, overall thrust must increase as the thrust vectoring angle increases.

Pictured below is a TVC actuation system attached to a fixed-wing plane. A swashplate configuration or rotating surface vanes may be better suited for larger-scale applications to reduce necessary actuator force.

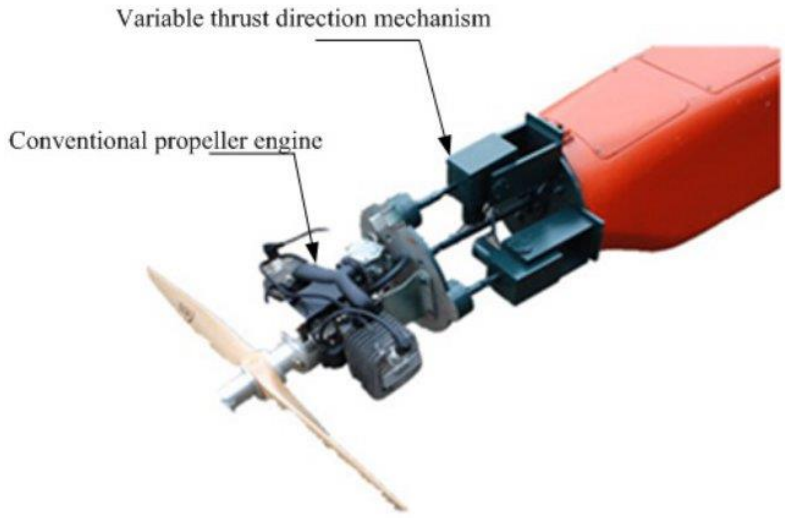

Figure 33: 2-Axis TVC Assembly, Proposed for use on PPC (image source: Y. Wang)

6.2 Thrust Vectoring PPC Simulation

The simulation of an open-loop thrust step response (*t = 5 seconds)* in Figure 32 shows that the airspeed is decoupled from the thrust input and therefore uncontrollable in a standard configuration PPG. Analysis of the step response of an increased throttle setting is shown below and is comparable to results from Chapter 2. This is done to validate the modified thrust vectoring dynamic equations.

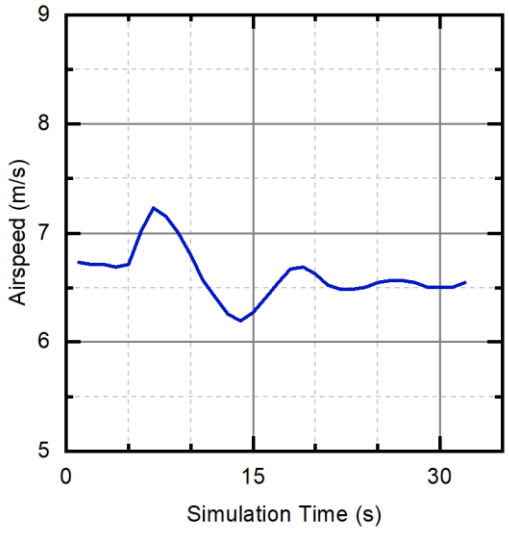

Figure 34: Thrust Vector Model Validation

The newly derived TVC model was also validated, including an uncontrolled step response simulation for an increase in the thrust vector angle. The step response is actuated at $t = 30\,sec$ and is displayed in the following figure of horizontal and vertical displacements over time.

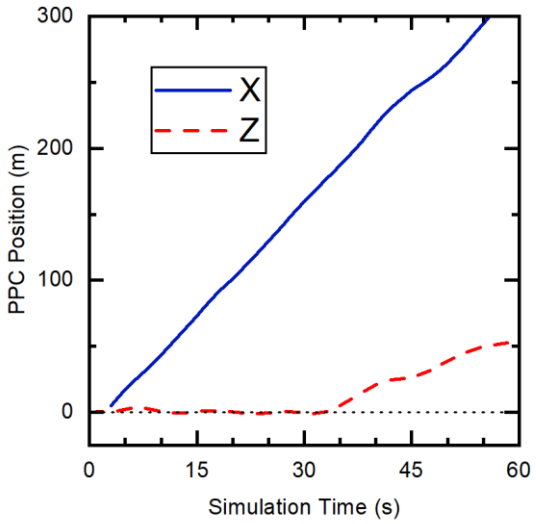

Figure 35: Example Thrust Vectoring Step Input

In the first 30 seconds of this simulation, before the thrust vectoring input is actuated, the PPC is in straight and level flight ($\dot{z}, \ddot{x} = 0$). After the step response input, $\dot{z}$ is increased, and $\dot{x}$ remains unchanged.

A decrease in thrust value shows a similar oscillation and a fixed descent rate in similar validation simulations. Removing thrust entirely shows the model reaches a stable descent. These various simulations prove the assumption that airspeed will no longer remain fixed despite an increase or decrease in motor thrust when a thrust vectoring input is applied. A final uncontrolled step response simulation validates the new control envelope available by thrust vectoring actuation. For the PPC described in Chapter 2 and within the appendix of this document, an ~4 m/s envelope in horizontal airspeed is now possible in straight and level flight. The following two figures detail these results.

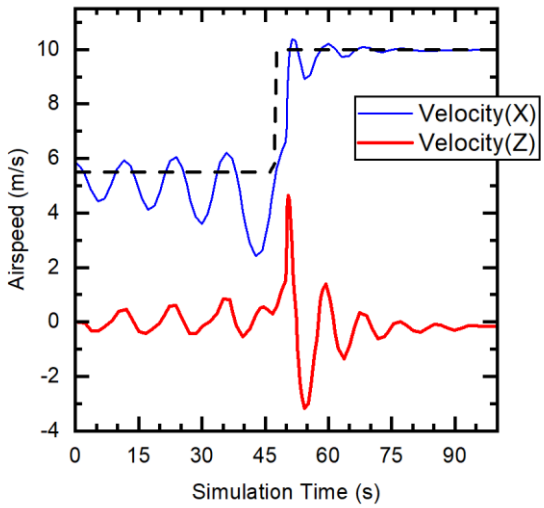

Figure 36: TVC Actuation Example of Increased Forward Velocity

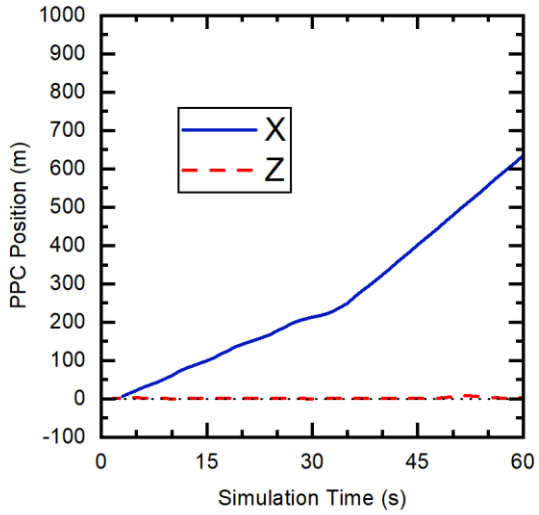

Figure 37: Thrust Vector Control Increase of Airspeed (Positional Graph)

A PID feedback loop was implemented within MATLAB, controlling the absolute altitude of the PPC as a desired feedback input. A step response is applied to change the thrust angle ε, with the PID control modulating input T, throttle, to maintain altitude. The PID

control improves system response time, oscillations, and allows for a larger controllable range compared to the uncontrolled simulations. Figure 32 shows a significant decrease in the system's oscillations, with a significant increase controllable range of forward velocity achievable.

The PPC with PID control is shown to increase the stability of the controllable region by approximately 5 m/s compared to the zero controllable airspeed of a traditional fixed thrust angle PPC. In addition, the feedback controller removes the oscillation and transient response present in the comparable Figure 5 without feedback control.

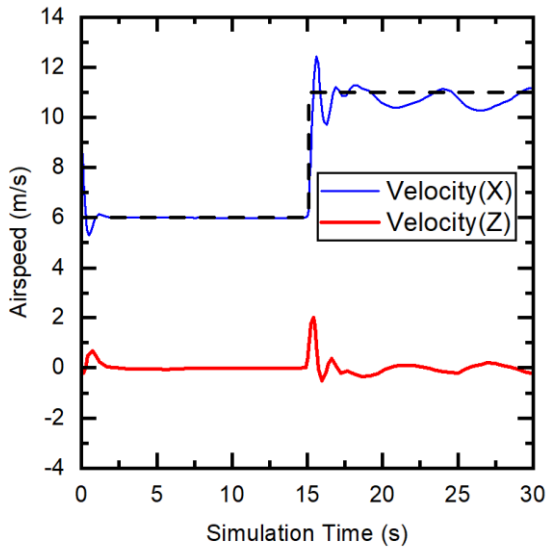

Figure 38: PID Controlled Example of TVC Actuation

As with the model in Chapter 2, simulation dynamics do not account for exceeding the critical angle of attack or operation within the near stall region, which would result in a collapse of the assumed dynamics and potential instability. This is especially problematic for the non-rigid structure of the paramotor wing, which may achieve a partial airfoil

collapse, as seen in testing with variable payload weights that operate near maximum wing loading capacity.

6.3 Lateral TVC

This section discusses the use case, dynamic modifications, and requirements for implementation of lateral control of the PPC with thrust vectoring. Up to this point, all lateral control has been actuated with airbrakes by initiating some moment about the vertical ‘yaw’ axis for directional control of the PPC. As mentioned in previous chapters, the main benefit of this standard control is that it dramatically simplifies dynamics and stability requirements for designing an autonomous PPC at the cost of directional agility. The ability to perform bank and pitch turning by a PPC is unnecessary for most autonomous mission use cases. However, this method of lateral control is possible and used as the primary control in the recreational paragliding community. Within the PPC platform, the most significant benefit of moving to a thrust vectoring lateral control method would be the complete removal of the airbrake system, including simplifying the parafoil design, removing extra rigging, and deletion of the airbrake servo actuators.

A PPC capable of full directional and lateral control only with a simplified thrust vectoring mechanism could prove to be both cheaper to manufacture and easier to deploy for end users.

Figure 39: A Manned Paraglider Performs an Aerobatic Roll (source: storytrender.com)

The most significant modification to the forces and moments dynamics within Chapter 2 will be the calculation of roll moment, *l*. The general equation for the calculation of *l* is given below:

$$l = \frac{1}{2} V_a^2 S b C_l(\beta, p, r, \delta_{TVC}) \tag{6.2}$$

This is further refined with the inclusion of aerodynamic coefficients for roll stability and damping provided by the parafoil. The primary design consideration is that the force moment applied by the TVC in the roll axis must be large enough to overcome the pendulum stability of the PPC within that yaw axis without necessarily applying a large amount of forward thrust, causing a pitching moment. This necessitates the TVC assembly having a large displacement angle, likely 30˚ or more. If an additional applied moment is

still needed, it may be necessary to include aero surfaces like a rudder. The detailed lateral TVC modification is as follows:

$$l = \frac{1}{2}\rho V_a^2 Sb[C_{l_o} + C_{l_\beta}\beta + C_{l_p}\frac{b}{2V_a}p + C_{l_{TVC}}F_{prop}\sin\vartheta] \tag{6.3}$$

Where ϑ is the thrust vectoring angle in the lateral axis, all other variables remain consistent in their definitions within Chapter 3.

Furthermore, a similar process to developing a 'bank and pitch' turning controller will be necessary for designing the autopilot controller. As such control is very similar to the methods used for autonomous fixed-wing aircraft, this method will not be covered further.

6.4 TVC Conclusion

This chapter discussed the control of a thrust vectoring design for powered parachute aircraft within the pitch axis. The main finding is that thrust vectoring for a paramotor can effectively increase the region of controllable airspeed and allow a PPC to operate in higher wind conditions or more significant turbulence potentially. Thrust vector control also improves the ascent and descent rates of the PPC and can shift airspeed slightly for optimizing lift and drag for a given parafoil geometry and wing loading.

The thrust vectoring method used in this paper can be further developed for lateral control of PPC, enabling heading direction changes without the necessity for complicated

rigging or inefficient air brake control. Future research will combine longitudinal and lateral control in a single dynamic model.

CHAPTER 7

EXPERIMENTAL RESULTS AND DISCUSSION

7.1 First Prototype

The initial prototype was designed using a heavily modified commercial remote-control paramotor frame, used in successful initial testing to develop the PixHawk autopilot system for a PPC. The chute has a wing area of 0.5 m^2, and the system has a final flying weight of 330 grams. Empirical testing has shown that the original parafoil has a dimensionless Coefficient of Lift calculated at approximately 0.15, which can be assumed to be relatively constant [1]. This prototype was then improved upon, moving to a larger area 1.5 m^2 parafoil with a maximum payload weight of 3.3 kg. The following figure shows a closeup view of the small PPC prototype in flight. The nylon line rigging, servo actuator arms, propellor and telemetry antenna on the front of the body is clearly visible.

Figure 40: Small PPC Prototype in Flight.

The following screen capture is an example of the freely available *Mission Planner* software that is one of the options for integration with the Pixhawk PX4 flight controller.

This software provides the programming and integration of the PX4 system, control modes, and live telemetry transfer directly to a ground station. One of the flight modes for the autonomous flight testing is displayed wherein the PPC is flying a circle of a fixed radius about a determined point on the ground. The minimum stable turning radius for the small PPC prototype was approximately 25 meters; however, the PPC could perform much tighter turns with good stability during manual flight mode.

Figure 41: Mission Planner Software PPC Testing

Telemetry datalink is made through a 900 MHz band serial data transceiver on the PX4 and the ground station laptop. This two-way data link is crucial for updating mission navigation and control parameters during testing. Speed, location, PPC orientation, and sensor health are transmitted back to the ground station. In addition, Mission Planner software can update PID control tunes through the telemetry link during flight. An

additional redundant manual R/C transmitter and receiver are connected to the PX4 to toggle between flight modes and provide manual control if necessary.

The PX4 controller serves as the flight computer system, which contains integrated sensors to capture the dynamic states of the PPC. A GPS receiver is connected directly to the PX4 unit. Initial flights of the system are running on a modified version of the built-in *fixed wing* control model; however, this has been updated to a custom firmware mixing for better control of the PPC airframe. Servo actuators located in the PPC body for lateral airbrake steering are rigged with nylon lines to the parafoil. The PPC brushless motor directly drives a carbon fiber propellor, the motor actuated by an Electronic Speed Control (ESC), which receives a PWM signal proportional to the desired thrust control from the PX4.

The motor is mounted in a ‘pusher’ configuration, standard for large-scale paramotors. Additional body configurations where the motor is mounted in a ‘tractor’ configuration and vertical ‘aero’ stabilizers are being experimented with to provide torsional damping to the body as it hangs under the parafoil. This configuration would further reinforce the dynamic rigid body assumption without modifying the underlying dynamic equations.

The following image is a picture of the initial autonomous prototype in flight over a testing field in Mesa, AZ. This image was captured from a commercial drone being flown to chase the PPC.

Figure 42: Initial PPC Prototype Flight Testing

Previous testing has also shown that the model is the most stable to pitch perturbations when the center of gravity is below the attachment point for the parafoil. However, too low will affect the ability of the motor to provide an adequate moment to pitch the model for a climb. This becomes problematic for a rigid body assumption.

Analysis of flight videos and recordings from the accelerometer shows that the paramotor is stable and well-damped for the roll axis. Even on the small parafoil prototype, the roll rarely exceeds 10 degrees of angle. Most of the turn is performed through yaw, consistent with the longitudinal control methods described in [1]. Overall performance of the small PPC was acceptable in winds less than 3 m/s; however, desire for a larger payload capacity and ability to fly in more extreme winds led to the development of a larger parafoil prototype. The following screen capture is from an onboard camera transmitting video back to a ground station:

Figure 43: Onboard Camera of PPC Prototype

This still frame was selected to show an example of a maximum bank angle during a minimum radius turn. By measuring the horizon within the image, we can approximate that the maximum bank angle during this turn is approximately 15˚. The PPC quickly corrects bank angle, but small rolls will be experienced in all flight modes because of adverse roll conditions and wind. It is important to note that while these minor disturbances cannot be corrected in flight, the results of roll and pitch oscillations can be removed in post-process for capturing video or other sensitive data as long as the aircraft state is recorded.

7.2 Energy Harvesting Development

The PPC is capable of additional features and attachments for improving mission capabilities. In addition to its large payload capacity for carrying cameras, remote sensing

instrumentation, or radio repeater equipment, the PPC is also uniquely suited for attaching energy harvesting missions for extended endurance roles. The following image captured from a prototype test flight shows two lightweight and flexible solar panels mounted on the top of the parafoil as a concept.

Figure 44: Energy Harvesting Concept Prototype PPC

From flight testing, this prototype 0.5 m^2 PPC can maintain level flight at approximately 26-30 watts. The most efficient solar panels produce approximately 150 W/m^2. However, the flexible and lightweight panels tested here produce much less power. A parafoil wing designed with cutting edge film solar panels could potentially produce 5-

10 W/m^2, not enough with current technology to sustain indefinite flight, but enough to potentially increase flight times by 15-30%. An efficiently designed PPC aircraft could fly indefinitely in areas that experience large thermal or wind-related updrafts, as is typical for aircraft gliders. A PPC aircraft could operate onboard electronics and actuators and charge batteries when a thermal soaring technique is applied with onboard solar panels.

7.3 Second Prototype Development and Testing Results

An additional PPC prototype was designed and flown, creating improvements on the initial prototype. The original PX4 flight controller and programming were carried into the next PPC. The Opale Models H1.5 parafoil was selected as an improvement to payload capacity and wind resilience. This new parafoil has three times the surface area of the original and was selected to increase payload capacity and battery size for long-endurance missions. The large PPC wing used for the second prototype was selected for the geometry measurements in the CFD simulations of the second chapter. It is characterized as a high lift airfoil capable of extremely slow flight (< 2 m/s) when loaded to the minimum payload.

During flight testing, the time aloft was more than 1.5 hours before returning to landing. Because of this prototype's more significant form factor, this prototype accepts a larger and more efficient motor and propellor combination. The motor combination has more static thrust than the smaller prototype, even at a lower and more efficient rpm. The following closeup picture is of the larger 1.5 m^2 prototype:

Figure 45: Large Prototype PPC on Approach for Landing

The large PPC was tested in various environmental conditions, ranging from stable weather to moderate wind conditions with updrafts. In contrast to the small PPC, the second prototype was much more dynamically stable in both pitch and roll tendencies. In combination with the new motor, the large PPC climbed significantly faster. Several experimental tests were performed in measuring the aerodynamic performance of the PPC.

The aircraft was flown to 100 m above ground level, after which the motor was shut off, and the PPC began to glide. The ground track distance for each trial run was taken in

the four cardinal directions to account for wind drift effects, and the PPC was loaded to a total weight of 3.0 Kg.

Figure 46: Lift to Drag Ratio Experiment

The results of this experiment show that the CFD simulations of the parafoil predicted with reasonable accuracy the lift and drag of the parafoil. At the 3.0 kg mass, the lift force is approximately 30 N. This amount of lift correlates to the 16˚ angle of attack simulation runs, where the L/D coefficient is measured at 5.9:1. However, in the glide testing, the PPC was observed to glide approximately 380 meters when averaged in all directions for wind speed, giving a final glide ratio of 3.8:1. There are several reasons for this discrepancy, mainly that the CFD analysis did not account for the drag induced by the parafoil rigging lines or the PPC airframe. It is possible to estimate the effect of the line effects and PPC airframe effects, but there would be a large margin of error at the slow flight speeds the PPC is operating.

In conclusion, autonomous control of a standard PPC is viable on COTS hardware, with parafoil selection being a critical decision for autonomous performance and overall efficiency. Future designs that encompass additional control methods paired with efficient fuselage design could outperform both multi-rotor and fixed-wing unmanned systems for specific missions.

CHAPTER 8

PPC HARDWARE IN THE LOOP SIMULATION

This chapter covers the development and application of autonomous PPC Hardware-in-the-Loop (HITL) simulation in the MATLAB/SIMULINK environment. MATLAB's UAV Toolbox contains hardware packages for multiple autonomous flight controllers, including the popular Pixhawk PX4 flight controller computers. PX4 open-source autopilot software also includes support for several airframes, focused mainly on quadcopter and fixed-wing airframes. In addition, MATLAB developers have generated a hardware package for PX4 flight computers, which allows for straightforward deployment of code generated within SIMULINK directly onto a PX4 FC. The HITL simulations in this chapter use the Simulink dynamic model and controllers developed within Chapters 2 and 3 onto a PX4 flight controller for testing and validation purposes.

Figure 47: PX4 Pixhawk 4 Flight Controller and GPS Antenna (Source: pixhawk.org)

This simulation technique is beneficial for validating controller compatibility on real hardware without conducting flight testing. In addition, the use of the MATLAB and

SIMULINK environments allows control systems engineers to have access to developing autonomous systems without advanced software development methods.

8.1 MATLAB PX4 Autopilot Controller Deployment

The UAV Toolbox Support Package for PX4 Autopilots is a development environment that carries support for direct code generation from SIMULINK worksheets. The hardware support contains access to the PX4's sensor suite, MAVLINK message bus, state estimators, and telemetry communication hardware. To deploy a custom firmware controller onto a PX4 FC, a prototype developer or tester should be familiar with state estimation, navigation and guidance, position and rate controller, and the actuator mixer blocks.

Developing a controller for the PPC airframe is the same architecture as autopilot controllers for multirotor or fixed-wing aircraft; however, a custom position and rate controller and actuator mixer must be included for successful implementation. The following image shows a general PX4 block diagram of an autopilot controller within Simulink.

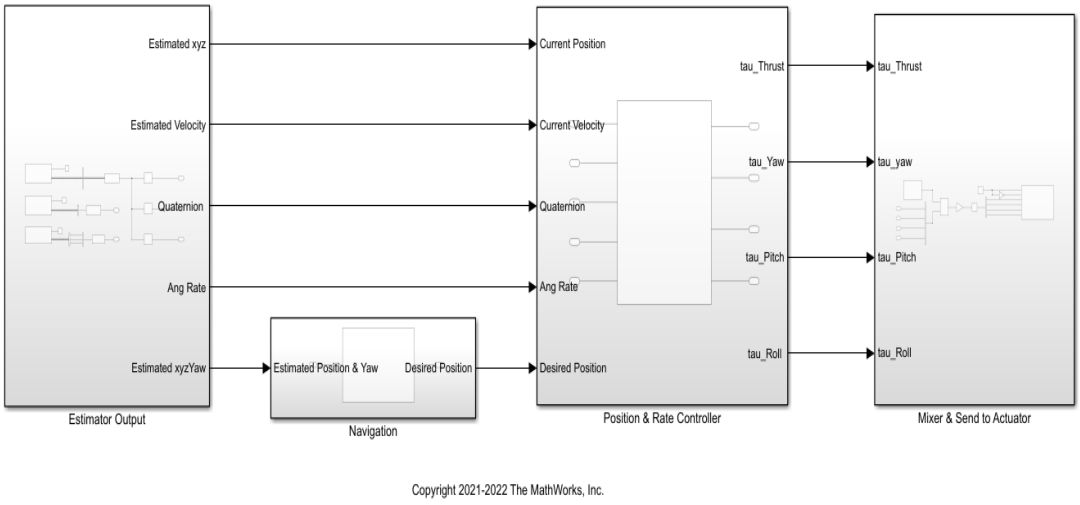

Figure 48: MATLAB Example of Hardware Deployable Autopilot for a Multirotor (Source: https://www.mathworks.com/help/supportpkg/px4/ref/hitl-simulink-plant-example.html)

Within this controller example, two main changes for the PPC are made. First, the position and rate controller must be modified to apply PID Feedback loops to generate necessary thrust and yaw commands for a standard PPC. Additionally, a custom mixer must be developed for transforming the commands into actuator signals applicable to the PPC.

The PID control defined in Chapter 3 of this document detail the position and rate controllers and are further modified for compatibility with the PX4 structure. For the PPC, the lateral and longitudinal controls should stay decoupled. The pitch control dynamics calculate the difference in desired altitude and measured altitude. The desired altitude is drawn from the ground control software waypoint, and the measured altitude is from the PX4's Extended Kalman Filter (EKF) state estimator module. This summing difference is

passed into a tuned PID block and provides a pitch control output magnitude. This pitch control output is then passed to the actuator mixing function to generate a PWM control output correlating to the desired thrust of the PPC. Applying the same process generates a yaw control output; however, the desired heading of the PPC is differenced by the estimated PPC heading combined with the sideslip due to wind. The following two figures display block diagrams of the altitude and heading controllers for deployment on the PX4.

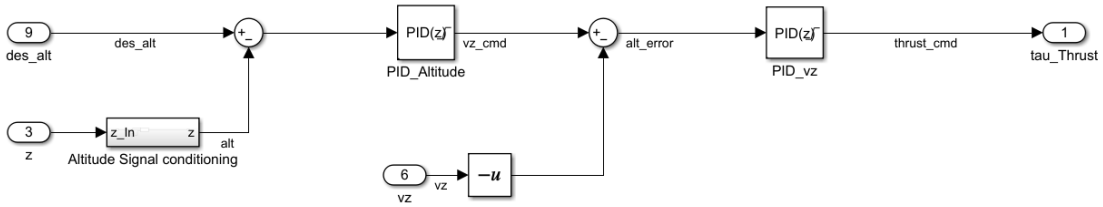

Figure 49: PPC Altitude Controller Design

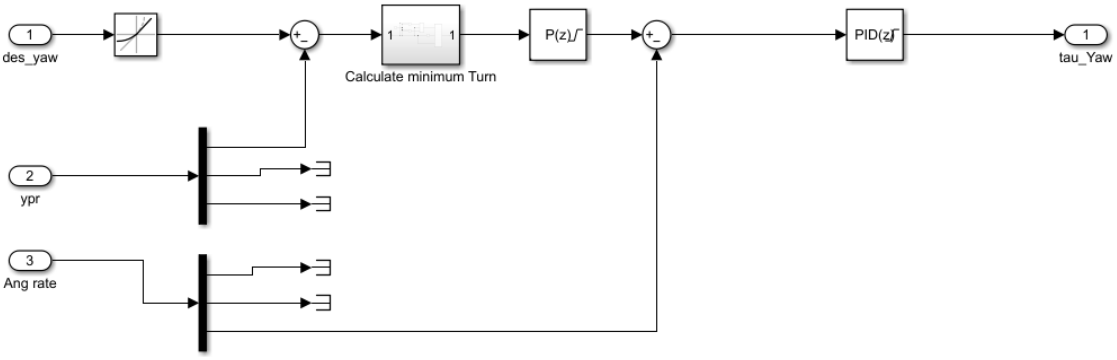

Figure 50: PPC Lateral Controller Design Diagram

The output of both the altitude and heading controllers must be converted to PWM signals which drive the physical actuators on the PPC. In the PX4 environment, this is accomplished in two steps. First, the mixing function details the combination of controls

and cross controls that generate dynamic forces on the aircraft. In the standard PX4 development environment, this is generated from a ‘mixing’ file to detail the actuator output commands. An example of a mixing file for the PPC is included in the figure below and further referenced in the PX4 software development wiki.

```
save as PPC.main.mix

Aileron/rudder/elevator/throttle mixer for PX4FMU
================================================

CH2: THROTTLE MIXER
-------------
Two scalers total (output, pitch).
#S: <group> <index> <-ve scale> <+ve scale> <offset> <lower limit>
<upper limit>
# apply throttle slowly limit to 3 seconds.
M: 1
S: 0 1  -10000  -10000      0 -5000  30000

CH3: AIRBRAKE MIXER
-------------
Two scalers total (output, yaw).
#S: <group> <index> <-ve scale> <+ve scale> <offset> <lower limit>
<upper limit>
# mix both controls for airbrake servos together, one for left turn
one for right turn

M: 2
S: 0 2  0  20000  0 -10000  10000
S: 0 2  -20000  0  0 -10000  10000
```

Figure 51: PX4 Example Actuator Mixing Function for PPC

A different approach is used for mixing the actuator outputs within the MATLAB environment. A mixing matrix is created to specify cross control commands detailed in the following matrix multiplication equations:

$$[Throttle \quad Airbrake(L,R)] = [Thrust \quad Yaw]\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} \tag{8.1}$$

As mentioned in Chapters 2 and 3, the PPC airframe is not fully controllable with standard control actuators; this is seen with the controllability matrix not having 'full rank.' This mixing matrix is then applied to the SIMULINK model, multiplied by the desired control outputs. The following figure is a block diagram for the mixing control applied in Simulink.

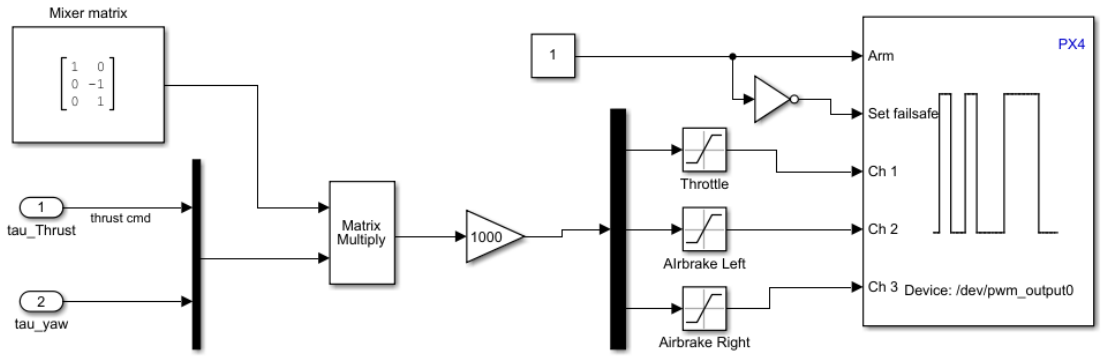

Figure 52: Actuator Mixing Diagram

The mixing of 'Airbrake Left' and 'Airbrake Right' controls is decoupled. This is done to reduce the amount of roll experienced by the PPC during the application of an airbrake. The airbrake controls should be defined on separate channels so that the left airbrake does not actuate during a right-hand turn and vice versa. The same PWM control could potentially actuate the different servos, but this would function more like a type of aileron control for a fixed-wing airplane than differential braking control.

This fully defines the controller that can now be deployed on the PX4 within MATLAB. The UAV Hardware Toolbox handles the generation and deployment of code onto a specific hardware target, most commonly a PX4 Pixhawk 1 or Pixhawk 4. In the case of developing a prototype model for actual flight testing, all necessary prerequisites have been completed, and the prototype is now ready to test with the experimental flight controller. Hardware in the loop testing requires an additional dynamic simulation of the PPC; this method is detailed within the next section.

8.2 MATLAB HITL Dynamic Plant Model

A 6 Degree of Freedom dynamic simulation is run on the host computer while connected to the PX4 hardware operating in HITL simulation mode. The dynamic model is cloned from the dynamic model generated in Chapter 3 of this document and modified to work with the inputs and outputs of the PX4 software. These I/O modules are defined within the hardware support package. They include the serial port connections to the PX4 hardware and additional signal conditioning to interface between the MAVLINK connection of the PX4 and Simulink model connections.

Additional functions must be defined prior to the dynamic model for computation of the amount of actuator deflection from the PWM outputs provided by the PX4 hardware. The general PPC HITL simulation block diagram is included in the figure below and interfaces to PX4 inputs and outputs.

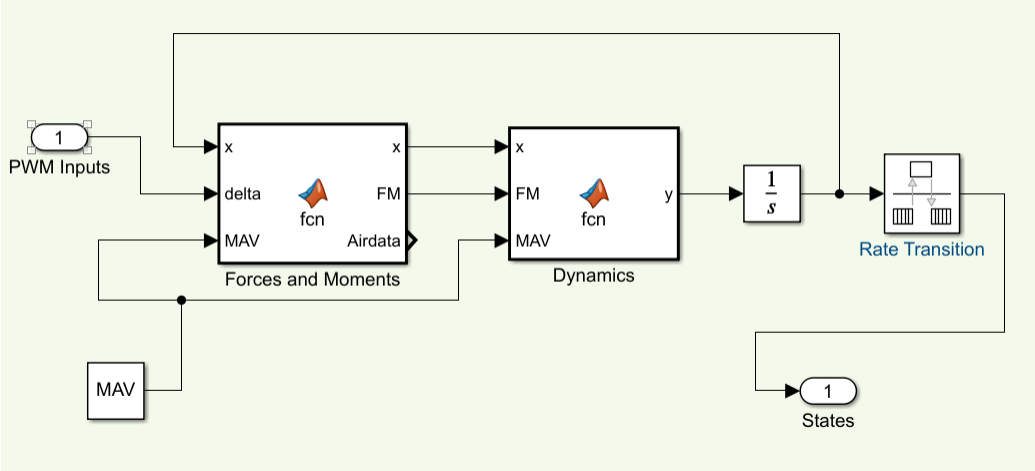

Figure 53: HITL Dynamic Host Simulation Block Diagram of PPC

This PPC simulation block above is a subfunction contained within the dynamic simulation block in the following diagram:

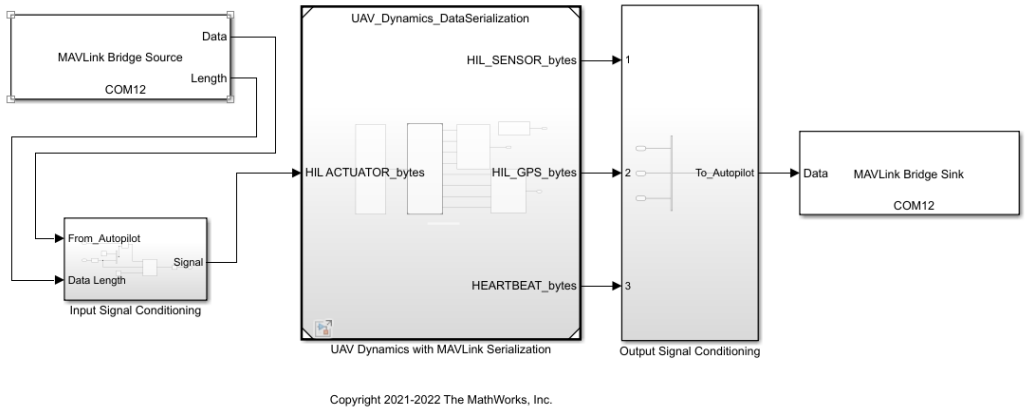

Figure 54: Host Computer Simulink Model for HITL Simulation (source: https://www.mathworks.com/help/supportpkg/px4/ref/hitl-simulink-plant-example.html)

When the computer host side HITL is initialized, it generates a serial port connection to the Q Ground Control software, an open-source user interface software used to apply autonomous drones. The following section will display the results of the HITL simulation.

8.3 HITL Testing and Results

After the autopilot controller is deployed on the PX4 hardware and the dynamic simulation is initialized on the host computer, Q Ground Control will begin to read the MAVLINK protocol telemetry via serial port and update positional data on the map for visualization. Q Ground Control is the user interface software allowing the selection of waypoints for guidance, on-screen telemetry information, and mission parameter adjustments.

The following screen capture is a HITL run showing the autonomous controller guiding the aircraft to the first waypoint.

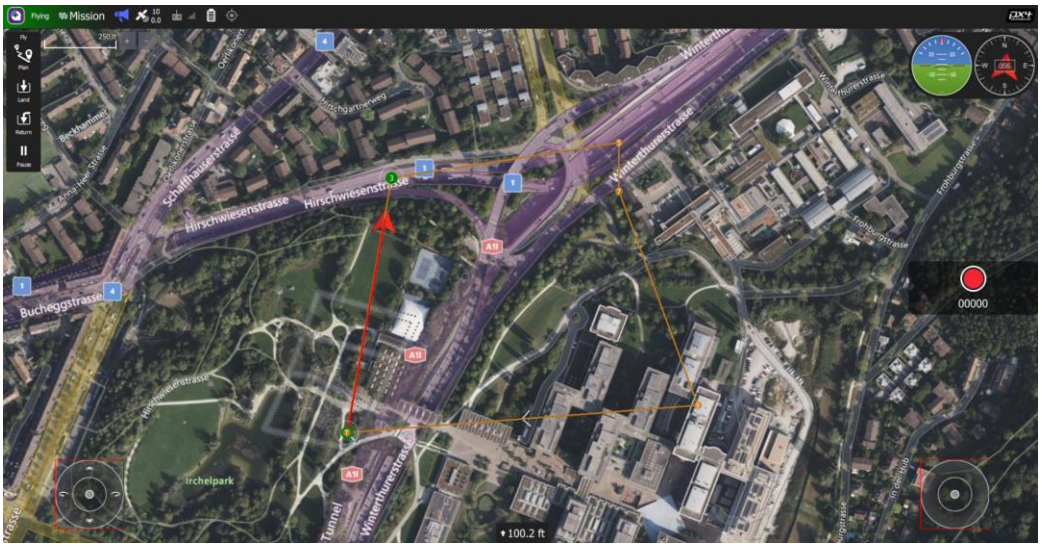

Figure 55: HITL Simulation Following Waypoints in QGC

The HITL simulation shows that the controller is successfully implemented within the PX4 firmware stack and proves that the controller is ready to be implemented in experimentation. Within the simulation, the user can select specific commands to change the guidance of the UAS midflight. The following screen capture shows an example of the UAS turning back to the takeoff location after a ‘Return to Home’ command is issued via the user interface.

Figure 56: Return to Home Command Issued and the Subsequent Flight Path

Additional HITL simulations are available within the MATLAB PX4 toolbox, such as integration and visualization within the Unreal® Engine. This was not performed during research for this document, but it encouraged future research into mapping, surveying, and visual guidance.

CHAPTER 9

CONCLUSION AND FUTURE WORK

Powered Parachute autonomous vehicles continue to have a few unique challenges associated with use before they become prevalent in practical applications. Additional research into additional control schemes by angle incidence control, weight shifting, and variable thrust angle control, could be further refined into a robust airframe for a more straightforward control design. This document experimented with and simulated advanced adaptive control techniques like MRAC and Neural Network dynamic control, proving the potential for these methods to be integrated into further projects. Application of the Neural Network control precisely captures non-linear relationship dynamics that are challenging to characterize and model. When incorporated with additional experimental data, the flight performance of a Neural Network controller should be significantly improved compared to the simulation testing.

This document included a successful procedure for modeling the aerodynamic coefficients from geometric data, guiding modeling parameters, and estimates of the accuracy of those results with flight testing. In addition to capturing aerodynamic parameters, the CFD simulations included valuable information on weight and airspeed estimates for overall system design. From these simulations and verification with flight data, it is recommended to design the PPC system wing loading between 0.1-0.2 g/cm^2.

The introduction of novel thrust vectoring control actuation to the PPC was successfully integrated into the standard parafoil dynamic model and improved the possible flight airspeed envelop of the PPC. The benefit of this type of control actuation is two-fold. First, the PPC system can benefit from simplified control methods, providing for easier to

develop parafoils that do not have complex rigging and aerodynamic surfaces. Second, the ability to expand the stable control envelope of the PPC can provide for a more stable platform of onboard sensors and increase the environmental variance the PPC can tolerate. These extra capabilities make the PPC a more competitive choice against multirotor and fixed-wing unmanned systems from a system architecture standpoint.

Chapter 8 included hardware integration and testing information within open-source software systems and successfully demonstrated the integration of PPC dynamic controllers. This chapter demonstrates the viability of the MATLAB hardware support package for the rapid prototype development and testing of a new control algorithm for non-traditional airframes. Applying Hardware-in-the-Loop simulations within the development chain affirms controller viability and performance without risking expensive equipment.

As small Unmanned Aerial Systems become increasingly prevalent, the capabilities for specific airframe configurations will be further tested and soon approach various technological limitations. The PPC has endurance, efficiency, and portability characteristics that have not yet been utilized to their full potential. Continuing research may prove small autonomous powered parachute aircraft to be viable alternatives in the emerging sUAS world. A few examples of future work recommendations are covered in the next section.

8.1 Future Work

There are several future work topics for the continuance of robust control development and guidance within the powered parachute domain. Many of these topic ideas were developed from the research for this dissertation but not extensively pursued. These topics consist of novel design considerations for PPC airframes and additional control algorithms to enhance capability. The following ideas are most recommended for their perceived potential for improving the PPC platform.

Within the scope of airframe sensors and dynamic modeling, estimation of the relationship between parafoil to the airframe body could be critical to understanding differences in the ‘modes’ of powered parafoil flight. For example, accurately measuring or estimating the canopy's angle could improve flight performance during pitch maneuvers. Measuring the canopy angle could be performed by either incorporating a rotary motion encoder on the pivot joint between the airframe and parafoil or using an upward-facing camera paired with machine learning. When this measured parafoil state information is used as an input to a neural network dynamic control algorithm, the neural network could capture the non-linear dynamic relationship between canopy and airframe without requiring the development of a non-linear dynamic model of the interaction. This method has the potential to provide an extremely robust non-linear controller for the PPC system by reducing the error between desired and actual flight paths.

From a system design viewpoint, improving guidance schemes and environmental energy capture may benefit powered parachutes to become a more prevalent choice for the UAS designer. Small efficiency gains can be realized from navigation algorithms that include identifying and following thermal updrafts. The slow flight characteristics and

large wing area of midsize PPC aircraft make it ideal for energy capture missions where time aloft is a top priority. When used in combination, efficient parafoil design, a low drag airframe, and energy harvesting flight planning or solar panels could significantly extend range and time aloft for specific missions. Tethered autonomous parafoils have additional potential to operate nearly indefinitely over a stationary position or be used as an alternative energy source in an area with prevailing winds.

Lastly, there is potential for using a parafoil wing in conjunction with alternative UAS airframes. One proposal is to develop a hybrid PPC and multirotor aircraft to capture the strengths of both individual airframes. A standard multirotor copter suspended vertically below a parafoil can actuate thrust for pitch control and lateral control without any additional motors or servo actuators. Research into this hybrid airframe could ensure that a multirotor can extend its time aloft until it releases away from the parafoil and performs in a mission role not possible with a standard PPC.

REFERENCES

[1] J. Chambers, "Longitudinal Dynamic Modeling and Control of Powered Parachute Aircraft," Rochester Institute of Technology, 2007.

[2] J. M. Stein, "Parachute Testing for the NASA X-38 Crew Return Vehicle," NASA Johnson Space Center, Houston, 1999.

[3] N. J. Slegers, "Dynamic Modeling, Control Aspects and Model Predictive Control of a Parafoil and Payload System," Oregon State University, Corvallis, June 2005.

[4] J. D. Nicolaides, "Performance Estimates for Powered Parafoil Systems," NTIS U.S. Department of Commerce, Springfield, VA, 1974.

[5] M. Watanabe and Y. Ochi, "Modeling and motion analysis for a powered paraglider (PPG)," in *SICE Annual Conference*, Kagawa University, Japan, 2007.

[6] R.-V. Mihai, R. C. Pahonie and I.-R. Edu, "A Practical Method to Estimate the Aerodynamic Coefficients of a Small-Scale Paramotor," *INCAS,* vol. 6, no. 4, pp. 63-73, 2014.

[7] J. Umenberger, "Guidance, Navigation and Control of a Small-Scale Paramotor," in *Proceedings of Australasian Conference on Robotics and Automation*, Victoria University of Wellington, New Zealand, December 2012.

[8] G. Van der Kolf, "Flight Control System for an Autonomous Parafoil", Thesis, Stellenbosch University, 2013.

[9] Y. Aoustin and Y. Martynenko, "Control Algorithms of the Longitude Motion of the Powered Paraglider," in *Proceedings of the ASME 2012 11th Biennial Conference On Engineering Systems Design and Analysis*, Nantes, France, July 2012.

[10] M. Ward, S. Culpepper and M. Costello, "Parafoil Control Using Payload Weight Shift," *Journal of Aircraft,* vol. 51, no. 1, pp. 204-215, 2014.

[11] Y. Ochi, "Flight Control of a Powered Paraglider by MIMO PID Control based on Two-Degree-of-Freedom Integral-type Optimal Servomechanism," in *19th IFAC Symposium on Automatic Control in Aerospace*, Würzburg, Germany, 2013.

[12] K. Tanaka, M. Tanaka, Y. Takahashi and H. O. Wang, "A Waypoint Following Control Design for a Paraglider Model With Aerodynamic Uncertainty,"

IEEE/ASME TRANSACTIONS ON MECHATRONICS, vol. 23, no. 2, pp. 518-523, April 2018.

[13] T. Jann, "Advanced Features for Autonomous Parafoil," in *18th AIAA Aerodynamic Decelerator Systems Technology Conference and Seminar*, Braunschweig, Germany, 2005.

[14] M. Ward, "Adaptive Glide Slope Control for Parafoil and Payload Aircraft," Thesis, Georgia Institute of Technology, 2012.

[15] H. Zhu, Q. Sun, X. Liu, J. Liu, H. Sun, W. Wu, P. Tan and Z. Chen, "Fluid–structure interaction-based aerodynamic modeling for flight dynamics simulation of parafoil system," in *Nonlinear Dynamics*, https://doi.org/10.1007/s11071-021-06486-0, 2021, p. 3445–3466.

[16] R. W. Beard and T. W. McLain, "Small Unmanned Aircraft: Theory and Practice," Princeton: Princeton University Press, 2012.

[17] Q. Quan, X. Dui and S. Wang, "Multicopter Design and Control Practice," Singapore: Publishing House of Electronics Industry, 2020.

[18] E. J. Scheuermann, "Autonomous Control of Parafoil and Payload Systems Using Upper Surface Canopy Spoilers," Thesis, Georgia Institute of Technology, August 2015.

[19] V. Prisacariu and I. Circiu, "The Analysis of the Flying Wing in Morphing Concept," *INCAS*, vol. 5, no. 2, pp. 43-52, 2013.

[20] A. Nagy and J. Rohacs, "Unmanned Measurement Platform for Paragliders," in *28th International Congress of the Aeronautical Sciences*, Montréal, Québec, 2012.

[21] C. Ippolito, "Integration and Control of Morphing Wing Structures for Efficiency/Performance," NARI Technical Report, WBS: 694478.02.93.02.11.04.21.

[22] C. E. Hanson, "Cooperative Thermal Soaring for Small Uninhabited Aerial Vehicles," California State University, Fresno, May 2008.

[23] M. J. Allen, "Autonomous Soaring for Improved Endurance of a Small Uninhabited Air Vehicle," NASA Dryden Flight Research Center, Edwards.

[24] Z. Akos, M. Nagy, S. Leven and T. Vicsek, "Thermal soaring flight of birds and unmanned aerial vehicles," Bioinspiration & Biomimetics, Budapest, Hungary, 2010.

[25] P. S. Williams-Hayes, "Flight Test Implementation of a Second Generation Intelligent Flight Control System," Technical Report NASA/TM-2005-213669. NASA Dryden Flight Research Center, November 2005.

[26] M. Paluszek and S. Thomas, "MATLAB Machine Learning," New Jersey: Apress, 2017.

[27] Gorman, "Modeling, Comparison and Analysis of Multi-body Parafoil Models with Varying Degrees of Freedom," ProQuest Dissertations Publishing, 2011.

APPENDIX

MATLAB CODE

MAV Parameters Files

```
clc
clear all

% PPC parameters

% initial conditions
MAV.pn0    = 0;     % initial North position
MAV.pe0    = 0;     % initial East position
MAV.pd0    = 0;     % initial Down position (negative altitude)
MAV.u0     = 2;     % initial velocity along body x-axis
MAV.v0     = 0;     % initial velocity along body y-axis
MAV.w0     = 0;     % initial velocity along body z-axis
MAV.phi0   = 0;     % initial roll angle
MAV.theta0 = deg2rad(5);     % initial pitch angle
MAV.psi0   = 0;     % initial yaw angle
MAV.p0     = 0;     % initial body frame roll rate
MAV.q0     = 0;     % initial body frame pitch rate
MAV.r0     = 0;     % initial body frame yaw rate

MAV.Cond0 = [
MAV.pn0
MAV.pe0
MAV.pd0
MAV.u0
MAV.v0
MAV.w0
MAV.phi0
MAV.theta0
MAV.psi0
MAV.p0
MAV.q0
MAV.r0
];
```

```
%physical parameters of airframe
MAV.gravity = 9.81;
MAV.mass = 3.0;   %Kg
MAV.Jx   = 0.824; %fix
MAV.Jy   = 1.135; %fix
MAV.Jz   = 1.759; %fix
MAV.Jxz  = 0.120; %fix
MAV.S_wing       = 1.5; %wing area
MAV.b            = 2.74; %wing span
MAV.c            = 0.55; %wing chord
MAV.S_prop       = 0.2027; %prop area
MAV.rho          = 1.2682; %air density kg/m^3
%MAV.e           = 0.9; %unused?
%MAV.AR          = MAV.b^2/MAV.S_wing; %unused? using simplified CL and CD calc
MAV.prop_H       = -0.1; %distance from motor to center of grav verify sign
MAV.chute_H      = -1.0;

% Gamma parameters from uavbook page 36
MAV.Gamma  = MAV.Jx*MAV.Jz-MAV.Jxz^2;
MAV.Gamma1 = (MAV.Jxz*(MAV.Jx-MAV.Jy+MAV.Jz))/MAV.Gamma;
MAV.Gamma2 = (MAV.Jz*(MAV.Jz-MAV.Jy)+MAV.Jxz*MAV.Jxz)/MAV.Gamma;
MAV.Gamma3 = MAV.Jz/MAV.Gamma;
MAV.Gamma4 = MAV.Jxz/MAV.Gamma;
MAV.Gamma5 = (MAV.Jz-MAV.Jx)/MAV.Jy;
MAV.Gamma6 = MAV.Jxz/MAV.Jy;
MAV.Gamma7 = (MAV.Jx*(MAV.Jx-MAV.Jy)+MAV.Jxz*MAV.Jxz)/MAV.Gamma;
MAV.Gamma8 = MAV.Jx/MAV.Gamma;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Longitudinal aerodynamic coefficients
%MAV.C_L_0        = 0.23; %zero AoA coefficient of lift
%MAV.C_L_alpha    = 5.61; %stability coeff
MAV.C_L_q        = 2.3; %stability coeff

%MAV.C_D_0        = 0.043; %zero AoA Coefficient of drag
%MAV.C_D_alpha    = 0.03; %stability coeff
```

```
MAV.C_D_q        = 1.0;   %stability coeff

MAV.C_m_0        = 0.0135; %zero AoA moment
MAV.C_m_alpha    = -2.74; %pitch stability (neg stable)
MAV.C_m_q        = -20.21; %pitch damping coefficient

%MAV.alpha0       = 0.47;
%MAV.C_L_delta_e   = 0.13; %unused PPC
%MAV.C_D_delta_e   = 0.0135;
%MAV.C_m_delta_e   = -0.99;

%MAV.epsilon      = 0.16; %unused simplified CL
%MAV.C_D_p        = 0.0; %coefficient of drag parasitic (roughly constant)

%% Lateral aero constants pg51 for stabilities

%force lateral
MAV.C_Y_0        =  0.0; %lateral force coeff (zeros)
MAV.C_Y_beta     = -0.83; %lateral force coeff sideslip
MAV.C_Y_p        =  0.0; %lateral force from roll
MAV.C_Y_r        =  0.0; %lateral force from yaw
MAV.C_Y_delta_a  =  0.0; %airbrake force lateral

%roll
MAV.C_l_0        =  0.0; %roll force coeff
MAV.C_l_beta     = -10.5; %roll stability (-neg stable)
MAV.C_l_p        = -2.8; %roll damping coefficient
MAV.C_l_r        =  0.25;
MAV.C_l_delta_a  = -0.04; %airbrake cross control derivative (roll)

%yaw
MAV.C_n_0        = 0.0; %pitch force coeff (may be slightly neg in PPC)
MAV.C_n_beta     = 0.5; %yaw stability coefficient (+stable)
MAV.C_n_p        = -0.069;
MAV.C_n_r        = -0.095; %yaw damping coefficient
MAV.C_n_delta_a  = -0.15; %airbrake primary control derivative (yaw)
```

```
%MAV.C_Y_delta_r   = 0.19;
%MAV.C_l_delta_r = 0.0024;
%MAV.C_n_delta_r   = -0.069;


%% Parameters for propulsion thrust and torque models
MAV.D_prop = 0.208;     % prop diameter in m

% Motor parameters
MAV.K_V = 145;                  % from datasheet RPM/V
MAV.KQ = (1/MAV.K_V)*60/(2*pi);   % KQ in N-m/A, V-s/rad
MAV.R_motor = 0.042;            % ohms
MAV.i0 = 1.5;                   % no-load (zero-torque) current (A)

% Inputs
MAV.ncells = 12;
MAV.V_max = 3.7*MAV.ncells;       % max voltage for specified number of battery cells

% Coeffiecients from prop_data fit
MAV.C_Q2 = -0.01664;
MAV.C_Q1 = 0.004970;
MAV.C_Q0 = 0.005230;

MAV.C_T2 = -0.1079;
MAV.C_T1 = -0.06044;
MAV.C_T0 = 0.09357;


MAVbus = Simulink.Bus.createObject(MAV); %creates simulink object
mav_bus = evalin('base',MAVbus.busName); %evaluates simulink object to bus
%use this in conjuction with a *Constant* source block and specify as bus
%input to matlab functions in simulink
```

Forces and Moments

```
function [x,FM,Airdata] = fcn(x,delta,wind,MAV)
%% forces and Moments
%   Computes the forces and moments acting on the airframe.
%
%   Output is
%       F     - forces
%       M     - moments
%       Va    - airspeed
%       alpha - angle of attack
%       beta  - sideslip angle
%       wind  - wind vector in the inertial frame
%


    % relabel the inputs
    pn      = x(1);
    pe      = x(2);
    pd      = x(3);
    u       = x(4);
    v       = x(5);
    w       = x(6);
    phi     = x(7);
    theta   = x(8);
    psi     = x(9);
    p       = x(10);
    q       = x(11);
    r       = x(12);


    delta_a = delta(1); %paramotor airbrake delta_a
    delta_t = delta(2); %paramotor thrust delta_t

    W_ns = wind(1); %steady state wind
    W_es = wind(2);
    W_ds = wind(3);
    U_wg = wind(4); %wind gust
```

```
    V_wg = wind(5);
    W_wg = wind(6);


    %% compute AIR DATA
    Rot_bv =[ cos(theta)*cos(psi), cos(theta)*sin(psi), -sin(theta);...
            sin(phi)*sin(theta)*cos(psi)-cos(phi)*sin(psi) ,
sin(phi)*sin(theta)*sin(psi)+cos(phi)*cos(psi) , sin(phi)*cos(theta);...
            cos(phi)*sin(theta)*cos(psi)+sin(phi)*sin(psi) , cos(phi)*sin(theta)*sin(psi)-
sin(phi)*cos(psi) , cos(phi)*cos(theta)]; %rotation matrix vehicle to body frame

    Vbw = Rot_bv *[W_ns;W_es;W_ds] + [U_wg;V_wg;W_wg];

    Vba = [u - Vbw(1), v - Vbw(2), w - Vbw(3) ];  %ur=1 vr=2 wr=3

    Va = (Vba(1)^2 + Vba(2)^2 + Vba(3)^2)^(0.5);
    alpha = atan(Vba(3)/Vba(1));
    beta = asin(Vba(2)/Va);
    Airdata = [Va,alpha,beta];

    %% Aerodynamic forces
    %compute 4.12 and 4.13 %CL(alpha) %CD(alpha)
    %BEST FIT TO PPC

    CL_ALPHA = 0.3969 + 0.1247*alpha + -0.0033*alpha^2; %coefficient of lift from
simscale
    CD_ALPHA = 0.0005*alpha^2 + 0.0135*alpha + 0.4778;
    %OLD
    %CL_ALPHA = MAV.C_L_0 + MAV.C_L_alpha * alpha;
    %CD_ALPHA = MAV.C_D_0 + MAV.C_D_alpha * alpha;

    %compute equations from 4.19
    CX   = - CD_ALPHA*cos(alpha) +  CL_ALPHA*sin(alpha);
    CX_q = -MAV.C_D_q*cos(alpha) + MAV.C_L_q*sin(alpha);

    CZ   = - CD_ALPHA*sin(alpha) -  CL_ALPHA*cos(alpha);
```

```
    CZ_q = -MAV.C_D_q*sin(alpha) - MAV.C_L_q*cos(alpha);

    %Calculate Aerodynamic forces
    FaeroX= CX + CX_q*q*MAV.c/(2*Va);
    FaeroY= MAV.C_Y_beta*beta + MAV.C_Y_delta_a*delta_a; %4.14 simplified
remove zeros
    FaeroZ= CZ + CZ_q*q*MAV.c/(2*Va);

    %prop forces
    %FpropX= 0.5*MAV.rho*MAV.S_prop*MAV.C_prop*((MAV.K_Q*delta_t)^2-Va^2);
    FpropX = 0.05*delta_t;

    % compute external forces on aircraft
    Force = [0,0,0];
    Force(1) =  -MAV.mass*MAV.gravity*sin(theta)         + ...
       0.5*MAV.rho*Va^2*MAV.S_wing*(FaeroX) + FpropX; %Fx

    Force(2) =   MAV.mass*MAV.gravity*cos(theta)*sin(phi) + ...
       0.5*MAV.rho*Va^2*MAV.S_wing*(FaeroY); %Fy

    Force(3) =   MAV.mass*MAV.gravity*cos(theta)*cos(phi) + ...
       0.5*MAV.rho*Va^2*MAV.S_wing*(FaeroZ); %Fz

%% Aerodynamic Moments


    TaeroL = MAV.b/2 * (MAV.C_l_0 + MAV.C_l_beta * beta + MAV.C_l_p * p *
MAV.b/(2*Va) +...
          MAV.C_l_r*r*MAV.b/(2*Va) + MAV.C_l_delta_a*delta_a) ; %4.15 ROLL
halved distance because only one airbrake is operating

    TaeroM = MAV.c * (MAV.C_m_0 + MAV.C_m_alpha * alpha + MAV.C_m_q * q *
MAV.c/(2*Va))+...
          (FaeroX)*MAV.chute_H; % PITCH Aerofoil pitching moment + drag moment
```

```
    TaeroN = MAV.b/2 * (MAV.C_n_0 + MAV.C_n_beta * beta + MAV.C_n_p * p *
MAV.b /(2*Va) +...
            MAV.C_n_r * r * MAV.b/(2*Va) + MAV.C_n_delta_a * delta_a); %4.16 YAW
equation halved distance

    Torque = [0,0,0];
    Torque(1) = 0.5 * MAV.rho * Va^2 * MAV.S_wing * (TaeroL); %L
    Torque(2) = 0.5 * MAV.rho * Va^2 * MAV.S_wing * (TaeroM); %M
    Torque(3) = 0.5 * MAV.rho * Va^2 * MAV.S_wing * (TaeroN); %N


FM = [Force,Torque];
end
```

Dynamics

```
function y = fcn(x,FM,MAV)

%states
   pn    = x(1);
   pe    = x(2);
   pd    = x(3);
   u     = x(4);
   v     = x(5);
   w     = x(6);
   phi   = x(7);
   theta = x(8);
   psi   = x(9);
   p     = x(10);
   q     = x(11);
   r     = x(12);

%forces and moments
   fx    = FM(1);
   fy    = FM(2);
   fz    = FM(3);
   l     = FM(4);
   m     = FM(5);
   n     = FM(6);


%position states
   pndot = w*(sin(phi)*sin(psi) + cos(phi)*cos(psi)*sin(theta)) - v*(cos(phi)*sin(psi) -
cos(psi)*sin(phi)*sin(theta)) + u*cos(psi)*cos(theta);

   pedot = v*(cos(phi)*cos(psi) + sin(phi)*sin(psi)*sin(theta)) - w*(cos(psi)*sin(phi) -
cos(phi)*sin(psi)*sin(theta)) + u*cos(theta)*sin(psi);

   pddot = w*cos(phi)*cos(theta) - u*sin(theta) + v*cos(theta)*sin(phi);

%velocity
   udot = r*v - q*w + (1/ MAV.mass)*fx;
```

```
    vdot = p*w - r*u + (1/ MAV.mass)*fy;

    wdot = q*u - p*v + (1/ MAV.mass)*fz;

%rotation
    phidot = p + q*sin(phi)*tan(theta)+ r*cos(phi)*tan(theta);
    thetadot = q*cos(phi)-r*sin(phi);
    psidot = q*sin(phi)*sec(theta)+r*cos(phi)*sec(theta);

%angular rates
    pdot = MAV.Gamma1*p*q - MAV.Gamma2*q*r + MAV.Gamma3*l +
MAV.Gamma4*n;
    qdot = MAV.Gamma5*p*r - MAV.Gamma6*(p*p-r*r) + (1/MAV.Jy)*m;
    rdot = MAV.Gamma7*p*q - MAV.Gamma1*q*r + MAV.Gamma4*l +
MAV.Gamma8*n;


y = [pndot; pedot; pddot; udot; vdot; wdot; phidot; thetadot; psidot; pdot; qdot; rdot];
```

PPC Face and vectors function for display

```
function [V,F,colors] = defineSpacecraftBody()
    % Define the vertices (physical location of vertices

%distances and lengths (meters)
  fuse_l    = .29;
  fuse_h    = .12;
  fuse_w    = .08;

  wing_l     = .4;
  wing_w     = 1.57;
  wing_w2    = wing_w/3;
  wing_h     = 0.96;
  wing_c     = 0.22;

  P1 = [fuse_l/2 -fuse_w/2 -fuse_h/2];
  P2 = [fuse_l/2 fuse_w/2 -fuse_h/2];
  P3 = [fuse_l/2 fuse_w/2 fuse_h/2];
  P4 = [fuse_l/2 -fuse_w/2 fuse_h/2];

  P5 = [-fuse_l/2 -fuse_w/2 -fuse_h/2];
  P6 = [-fuse_l/2 fuse_w/2 -fuse_h/2];
  P7 = [-fuse_l/2 fuse_w/2 fuse_h/2];
  P8 = [-fuse_l/2 -fuse_w/2 fuse_h/2];
  Pivot1 = [0 -fuse_w/2 -fuse_h/2];
  Pivot2 = [0 fuse_w/2 -fuse_h/2];

  %wing points
  P9  = [wing_l/2 -wing_w/2 -wing_h];
  P10 = [wing_l/2 -wing_w2/2 -wing_h-wing_c];
  P11 = [wing_l/2 wing_w2/2 -wing_h-wing_c];
  P12 = [wing_l/2 wing_w/2 -wing_h];
  P13 = [-wing_l/2+0.1 wing_w/2 -wing_h];
  P14 = [-wing_l/2 wing_w2/2 -wing_h-wing_c];
  P15 = [-wing_l/2 -wing_w2/2 -wing_h-wing_c];
  P16 = [-wing_l/2+0.1 -wing_w/2 -wing_h];
```

```
V = [...

P1;...
P2;...
P3;...
P4;...
P5;...
P6;...
P7;...
P8;...
P9;...
P10;...
P11;...
P12;...
P13;...
P14;...
P15;...
P16;...

Pivot1;...
Pivot2;...
];

F = [...

%fuselage
 1, 2, 3, 4;...
 1, 4, 8, 5;...
 1, 2, 6, 5;...
 2, 3, 7, 6;...
 3, 4, 8, 7;...
 5, 6, 7, 8;...

%parachute
 9 , 10, 15, 16;...
 10, 11, 14, 15;...
```

```
 11, 12, 13, 14;...

%lines
17,9,9,17;
17,16,16,17;
17,10,10,17;
17,15,15,17;

18,11,11,18;
18,12,12,18;
18,13,13,18;
18,14,14,18;
];

chutecolor = [1, 0, 0];
fusecolor =  [0, 0, 1];
linecolor =  [1, 1, 1];
colors = [...

    fusecolor;...
    fusecolor;...
    fusecolor;...
    fusecolor;...
    fusecolor;...
    fusecolor;...

    chutecolor;...
    chutecolor;...
    chutecolor;...

    linecolor;...
    linecolor;...
    linecolor;...
    linecolor;...
    linecolor;...
    linecolor;...
```

```
    linecolor;...
    linecolor;...
    ]

end
```