RVC-1901

Vertical Take-off and Landing Autonomous Aircraft Design

A Major Qualifying Project Report
submitted to the Faculty of the WORCESTER POLYTECHNIC INSTITUTE
in partial fulfillment of the requirements for the
Degree of Bachelor of Science

in Aerospace Engineering
by

Quintin Barker

Troy Bergeron

Jacob Goldsberry

Romelle Jack

Bailey Waterman

Zachary Zolotarevsky

Approved by:
Raghvendra V. Cowlagi, Advisor
David J. Olinger, Advisor
Aerospace Engineering Program, WPI

in Aerospace Engineering and Robotics Engineering
by

Benjamin Pasculano

Approved by:
Raghvendra V. Cowlagi, Advisor
David J. Olinger, Advisor
Aerospace Engineering Program, WPI

William R. Michalson, Advisor
Robotics Engineering Program, WPI

in Aerospace Engineering and Electrical & Computer Engineering
by

Tyler Weiss

Approved by:
Raghvendra V. Cowlagi, Advisor
David J. Olinger, Advisor
Aerospace Engineering Program, WPI

William R. Michalson, Advisor
Electrical & Computer Eng. Dept., WPI

Abstract

This project addresses the design, analysis, and and construction of an autonomous fixed-wing/rotorcraft hybrid aircraft capable of vertical take-off and landing. Autonomy is addressed to enable obstacle avoidance, visual detection of a landing target area, and accurate landing. The designed aircraft consists of a ducted main rotor and two smaller tilt-rotors, and is based on a similar design from the literature. This report provides detailed analyses of the aircraft's aerodynamic and structural properties, dynamics and stability, propulsion, and power. The development of onboard autonomy using a 3D depth sensor is presented. Simulations of stabilizing controllers are presented. The construction of a prototype aircraft and its preliminary flight test results are reported.

Table of Authorship

Chapter	**Author**	**Project Work**
Chapter 1: Background		
Section 1.1: Project Goals	TB	N/A
Section 1.2: Project Design Requirements, Constraints, and Other Considerations	TB	N/A
Section 1.3: Project Management	All	N/A
Section 1.4: Literature Review	TB QB	All
Chapter 2: Analytical Tools		
Section 2.1: Aerodynamics & Structures	QB TB RJ ZZ	QB RJ ZZ BW
Section 2.2: Simulation & Software	BW TB	BW TB
Section 2.3: Power & Propulsion	TW	TW JG ZZ
Section 2.4: Autonomy & Control	JG BP	BP TW
Chapter 3: System Design		
Section 3.1: Aerodynamics & Structures	QB ZZ RJ TB	QB JG ZZ RJ TB
Section 3.2: Simulation & Software	BW TB	BW TB
Section 3.3: Power & Propulsion	TW	TW JG ZZ
Section 3.4: Autonomy & Control	BP JG	BP JG TW
Chapter 4: System Development		
Section 4.1: Aerodynamics & Structures	ZZ RJ	JG ZZ RJ QB BW
Section 4.2: Power & Propulsion	TW ZZ	ZZ JG
Section 4.3: Autonomy & Control	BP TW BW TB	BP TW BW TB
Chapter 5: Results		
Section 5.1: Aerodynamics & Structures	JG	JG ZZ RJ QB BW
Section 5.2: Power & Propulsion	TB	ZZ JG
Section 5.3: Autonomy & Control	BP TW BW TB	BP TW BW TB
Section 5.4: Flight Test Results	ZZ JG	JG ZZ RJ QB
Chapter 6: Conclusions, Recommendations, and Broader Impacts		
Section 6.1 Broader Impacts	QB	N/S
Section 6.2 Conclusions	TB BW JG ZZ TW BP	N/A
Section 6.3 Recommendations	TB JG ZZ TW	N/A

Contents

List of Figures

List of Tables

Chapter 1

Background

In recent years the development, of small unmanned aerial vehicles (UAVs) has been a large area of research due to their commercial appeal. These vehicles are typically used for surveillance, filming, search and rescue, agriculture, and cargo transport. Being unmanned, these vehicles are designed to either be piloted remotely, or are equipped with sensors and programmed to run autonomously. UAVs can be launched in various ways including from a canon, on a typical runway, or through vertical lift. These vehicles can be powered through various methods including liquid fuel and electric motors. The intended use of a UAV influences the choice of design features mentioned above. For instance in situations where an area is too dangerous or it's simply impractical, an autonomous UAV will be favored over one that has to be remotely operated. For many tasks UAVs are used for, the space needed to perform a conventional take-off and landing is not available so a vehicle with vertical capabilities may be preferred, and the duration of the UAVs mission will often determine the fuel source used. This project determines the appropriate UAV characteristics for a package pickup and drop off simulation and then goes on to design, build and test the proposed vehicle.

1.1 Project Goals

The objective of this project is to develop a fully autonomous UAV capable of vertical take-off and landing as well as traditional fixed wing flight to achieve increased speed and range over a traditional multirotor design.

1.2 Project Design Requirements, Constraints, and Other Considerations

The design requirements for the system as a whole were:

- The system must be capable of vertical take off and landing.
- The system must be capable of transitioning to forward flight.
- The system must be capable of obstacle detection and avoidance
- The system must be capable of identifying a colored circular target on the ground.

The design requirements of the aerodynamic subsystem were:

- The wings must generate enough lift such that the vehicle can fly sufficiently slow in forward flight mode for the obstacle detection and avoidance algorithm to be effective.
- Aerodynamic control surfaces must provide sufficient control during forward flight mode to execute the obstacle avoidance trajectories.

The design requirements of the structural subsystem were:

- The frame must be structurally sound and large enough to store all components.
- The frame must allow easy access to internal components.
- The frame must provide mounting surfaces for all motors and servos.

The autonomy and control subsystem requirements were:

- The optical camera must interface with the onboard computer.
- The Realsense depth camera must interface with the onboard computer.
- The onboard computer must interface with the flight controller.
- The onboard computer must be able to run the target detection algorithm using information from the optical camera.
- The onboard computer must be able to run the obstacle detection and avoidance algorithm using information from the optical camera.

The design constrains were:

- The total project cost was limited to $ 2000 provided by the WPI Mechanical Engineering Department
- The vehicle must comply with FAA regulations for UAVs of this class.

The non-technical considerations were:

- No non-technical considerations drove the design of the vehicle.

1.3 Project Management

Organization:

The project used two different subdivision schemes throughout the project. During A and B term, when the project was in the research, design, and early development stages, the team consisted of four subgroups. The groups were, Aerodynamics & Structures, Power & Propulsion, Software & Simulation, and Controls & Autonomy. These subgroups were responsible for research and design work in their respective fields as well as the early development work. During C term the team was subdivided into two groups, the Build Team and the Software Team. Build

Team was responsible for implementing and refining the physical UAV design. The Software Team was responsible for implementing the designed software architecture and computing and sensing components.

Management Approach of MQP:

The task of team management was split apart based on each sub group. In the first two terms, each sub group typically had the person with the greatest understanding of the topic in charge. This resulted with Zolotarevsky in charge of Aerodynamics & Structures, Weiss in charge of Power & Propulsion, Bergeron in charge of Software & Simulation, and Pasculano in charge Controls & Autonomy. On a roughly weekly basis the team would meet as a whole to keep the other subgroups up to date on their progress. The team leaders would lead their teams with goals and deciding the next steps for the sub team to make.

During the actual build phase in C term, the teams were split such that the Build Team consisted of the members from the Aerodynamics & Structures group with some overlap from Power & Propulsion while the Software Team consisted of the remaining members. This was decided by previous experience in the areas required by each of the teams. With Goldsberry and Zolotarevsky having the most knowledge on constructing similar, small quadcopters and UAV's they worked together in guiding the Build Team. For the Software Team, Pasculano and Weiss had the most experience and thus took charge in that field of development.

Technical Contributions by Member:

Bergeron's technical contributions included research on different design schemes for hybrid VTOL fixed wing UAV's as well as the design of dynamic simulations for both the TURAC and VertiKUL designs in and Simulink. Bergeron was also responsible for much of the aerodynamic wind tunnel testing and subsequent data analysis. During C term, Bergeron worked on the software team to integrate the various hardware/software components that make up the controls, autonomy, and sensing portions of the project.

Pasculano's technical contributions included researching methods for aerial autonomous

navigation of fixed wing vehicles, and other aerial vehicles. Next he researched components including single board computers, cameras, stereo depth cameras, 2D and 3D rotational lidars, and other components required to build a fully autonomous system. From this research he selected the required components based on several factors including weight, cost, and overall compatibility. Pasculano was also the primary contributor to the setting up and debugging of technical difficulties encountered in this process for which there were many.

Waterman's technical contributions included equation based aerodynamic analysis to verify the accuracy of data obtained from XFLR5 in the early stages of the project. Later on her contributions included hot wire cutting wings to assist the build team, dynamical modelling of a hybrid fixed wing tilt-rotor design in Matlab and design of an lqr based controller for this design using Matlab and Simulink. Later on in the project she assisted with the control and autonomy software integration.

Zolotarevsky's technical contribution included leading the Aerodynamics & Structures Team during the initial phase of the project. He performed aerodynamic and stability analyses in XFLR5 and some structural research and design. Aerodynamics work included designing and testing all the iterations of the vehicle in XFLR5 to confirm aerodynamic stability and finding aerodynamic coefficients to use in the Matlab stability analysis. He also tested 3D printed models in the wind tunnel for confirmation of the aerodynamic performance of each iteration. Structural work included researching materials to construct the test vehicles and then designing and building those vehicles. During the build phase, Zolotarevsky was a co-lead of the Build Team. His work included iterative design and test of the physical vehicle. Much of his work included working on the Pixhawk, due to his past experiences working with it, which meant researching and editing the parameters in the Pixhawk to achieve the desired performance of the test vehicle. His experience with the Pixhawk firmware and Mission Planner software allowed him to work on the electronics side of the physical build and working with the Software Team on occasion to aid in their use of the Pixhawk. He also helped to design some of the 3D printed parts of the vehicle and hot wire cut wings for testing. Finally, he was the team purchaser who

was responsible for communicating with the Mechanical Department to buy parts required by the team during the build phase.

Barker's technical contribution included being apart of the Aerodynamics & Structures Team during the initial phase of the project. He investigated the aerodynamic forces that act on the aircraft in flight and helped in calculating aerodynamic coefficients and non-dimensional derivatives to achieve aerodynamic stability. He also contributed in selecting the proper airfoil for the aircraft through wing tunnel testing and XFLR5 analysis. As part of the build team, he assisted in constructing the frame and body of the aircraft as well as laser/hot wire cutting wings. He also contributed in building multiple test models for wind tunnel, glide, and hover tests.

Jack's technical contribution included supporting the Aerodynamics & Structures Team throughout the entirety of the project. He lead the structural modeling of the project using SolidWorks to ensure and confirm accurate and efficient modeling of each component design such as the body design as well as smaller component designs. He aided in designing and testing smaller wind tunnel test models to verify aerodynamic properties. He served as a member of the project Build Team and created the body structure of the project design as well as ensuring proper and efficient internal component structures. Most of his work consisted of modeling and hands-on manufacturing of the project. He served as a link for creating various detailed models for components such as the wings,so that other members could manufacture them through methods such as hot wire cutting and laser cutting. In terms of manufacturing the project, he served as a co-member to the Build Team. As a member of the Build Team, his most important task was to ensure proper construction of the project through various means, as well as being a pivotal member to the testing of each design iteration over the course of the project.

Goldsberry's technical contribution was split between build team, propulsion, and software-in-the-loop. Throughout the beginning of the project, he focused primarily on theoretical propulsion calculations and design philosophies while developing our SITL testing methods

on the side. He worked in support of Weiss and Zolotarevsky to determine an appropriate propulsion system, balancing their theoretical knowledge with his hobbyist experience. He worked in support of Jack to build Solidworks models and, once the models were developed, handled the majority of the 3D printing. He also handled a significant portion of the construction of the glide test models and shared his knowledge of foam board molding with build team to expedite the future construction of vehicles. Once the team transitioned to building our final vehicle, Goldsberry co-lead the build team in the testing and development of our vehicle.

Weiss's technical contribution was both in software development, particularly in the development of our sensor systems and intra-component communication, and propulsion design. Because of his background in Electrical & Computer Engineering, Weiss spearheaded the propulsion system design and selection. This included endurance modelling, thrust estimations, and performance optimization. He worked closely with Waterman, Pasculano, and Bergeron to develop our image processing and autonomous control methods.

Presentations:
Bi-Weekly presentations were put together by the team members for meetings with the project advisors Cowlagi, Michalson, and Olinger. These presentations were primarily put together by Weiss and Pasculano, with most team members contributing to the presentations with their own teams' work.

1.4 Literature Review

In order for our team to achieve our project objective, we researched pre-existing hybrid E-VTOL/Fixed wing designs. Our research revealed two basic design philosophies. In the first system, some or all of the rotors that provide thrust to the vehicle can tilt or rotate to provide vertical thrust for takeoff, while also providing horizontal thrust for flight during fixed wing mode. This type of design is referred to as a tilt-rotor vehicle. In the second system, a wing

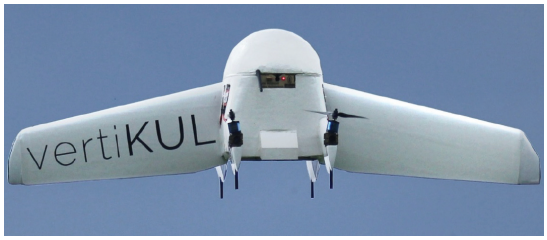
is attached to a quadrotor with the trailing edge of the wing pointed in the same direction as the rotors. This vehicle must perform a maneuver in which it flips during flight to transition to horizontal flight mode. Due to this maneuver, the design is referred to as a tail-sitter. Our team analyzed an example of each of these design philosophies to determine pros and cons and to ultimately determine which design was the best fit for our project.

The TURAC is a VTOL UAV aimed at combining the advantages of unmanned helicopter and conventional UAVs. It has the capability of vertical take-off and landing (VTOL) as well as conventional take-off and landing (CTOL) [3]. The design of the TURAC includes two electric tilt rotors in the front, attached to a fixed-wing and a fixed electric coaxial fan located in the rear of the TURAC body. The coaxial fan is used during the VTOL and hovering flight modes to provide lift. The two tilt rotors are used in all flight phases. During vertical take-off and landing the tilt rotors are oriented to provide thrust perpendicular to the wing. The tilt rotors are then rotated 90 degrees for its vertical orientation to point horizontally for forward flight in fixed wing mode. The TURAC utilizes a tail-less, flying wing, design. There are elevons on the wings to control pitching and rolling motion during forward flight as well as rudders on the winglets to control yaw [20]. There are many advantages to this design. In hover mode it can be flown as a traditional tri-copter. After transitioning to fixed wing mode, it is flown like many other tailless flying wing aircraft. This configuration is ideal, as it does not sacrifice functionality in either mode. Unfortunately, the presence of aerodynamic control surfaces as well as the tilt rotors makes this a more mechanically and mechatronically complicated design.

The VertiKUL is a hybrid design that consists of a quadrotor with a wing attached parallel to the direction of the thrust from the rotors. This design relies on a mid-air flip to transition to fixed wing flight. The vehicle starts from hover and accelerates upward while pitching over. The maneuver ends with the vehicle flying horizontally above stall speed. Once in this flight mode, all of the control is from the four rotors as there are no aerodynamic controls surfaces on this design. This design is mechanically simpler to build. The physical systems on the VertiKUL are identical to a traditional quadrotor with the addition of a wing. However, the

transition maneuver is more complicated than that of the TURAC and the flight controls in fixed wing mode are very different from standard fixed wing flight controls. Overall, this is a less versatile design. It sacrifices utility in both hover and fixed wing mode. [8]

(a) Turac Design [20]



(b) VertiKUL Design [8]

Figure 1.1: Images of the Turac (left) and VertiKUL (right) that are the two case study designs that were researched.

Chapter 2

Analytical Tools

2.1 Aerodynamics & Structures

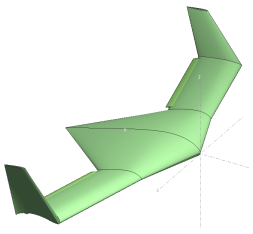
2.1.1 Aerodynamics

Using provided information regarding the airfoils and wings, full wing models for both the TURAC and VertiKUL drones were created in XFLR5. All major iterations of the design can be seen in Figure 2.1. Aerodynamic coefficients and non-dimensional derivatives were calculated using XFLR5. Several of these coefficients were also calculated analytically to cross check the validity of the simulation results. This data was critical to the creation of accurate dynamical simulations of both designs as described in section 3.4.

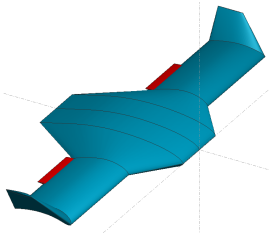
In order to cross check the validity of the XFLR5 analysis for the TURAC, we used the textbook Aircraft Dynamics: From Modeling to Simulation by M. R. Napolitano. From this text we were able to find:

$$C_{l_\alpha} = \frac{2\pi R}{2 + \sqrt{\left[\frac{AR^2(1-Mach^2}{k^2}\left(1 + \frac{\tan^2(\Lambda)}{(1-Mach^2)}\right)\right] + 4}} \tag{2.1}$$

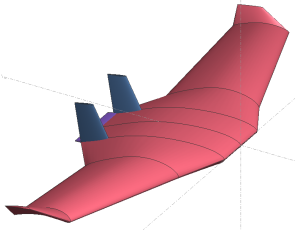
$$C_{m_\alpha} = C_{l_{\alpha_w}}\left(x_{CG} - x_{AC_{wb}}\right) - C_{l_{\alpha_h}}\eta_h\frac{S_h}{S}\left(1 - \frac{\delta\epsilon}{\delta\alpha}\right)\left(x_{AC_h} - x_{CG}\right) \tag{2.2}$$



(a) First Iteration



(b) Intermediate Iteration



(c) Final Iteration

Figure 2.1: XFLR5 models of all design iterations that were tested.

$$C_{d_\alpha} = \frac{2c_{l_1}}{\pi ARe} C_{l_\alpha} \tag{2.3}$$

Since the TURAC does not have a horizontal tail, we made the assumption that the tail terms in both equations were negligible. Using this assumption, as well as the equations above, we will be able to pick several airfoils for testing as our main wing airfoil.

2.1.2 Structures

Structural models of both the VertiKUL and TURAC designs were created in Solidworks. The purpose of this was two fold. First, physical parameters not reported by the papers would be calculated by Solidworks. Second, physical parameters that were reported could be cross-checked for accuracy. Computer Aided Design (CAD) for structural modeling of the aircraft is an essential tool to ensure and confirm structural integrity, as well as to model potential designs prior

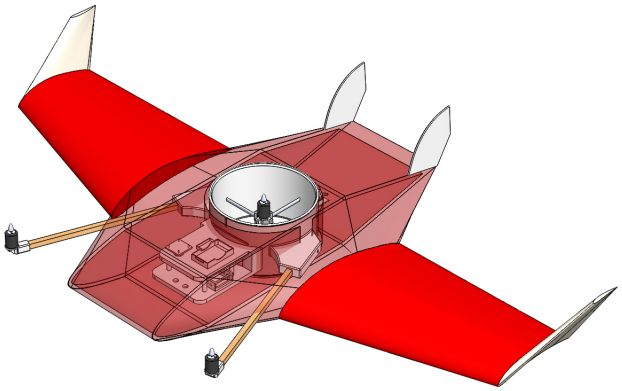


Figure 2.2: The SOLIDWORKS model of our vehicle that resulted from the iterative design process.

to manufacturing a final design. SOLIDWORKS is the main CAD suite used to structurally model the aircraft due to many factors. The main factor being that SOLIDWORKS offers a very general and easy to use suite of analysis tools to help understand the impact of aspects such as material selection and moments of inertia, just to name a few. Using SOLIDWORKS, the structures team was able to accurately model an aircraft structure that could support our electrical components while also being relatively easily manufactured. The primary concern was ease of construction and the ability to hold all of the components. The SOLIDWORKS model can be found below:

Once a full model was built and assembled, each component material was defined to accurately reflect our intended build materials. We then were able to obtain the moments of inertia for the aircraft using SolidWorks 'Mass Properties' tool. The initial moments of inertia values calculated by SolidWorks were important to further run analysis such as stability analysis on our aircraft.

2.2 Simulation & Software

For the purposes of testing and simulation of control systems, complete dynamical models of both designs were created in Matlab/Simulink. These models can be used to calculate trim conditions as well as simulate controllers. The flow chart of these models can be found below:

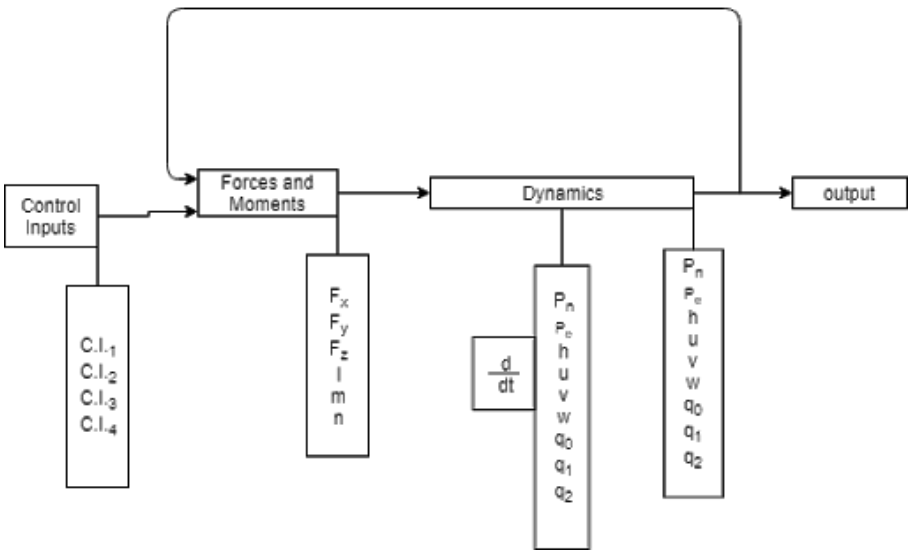


Figure 2.3: The Simulation Flow Chart that describes the control system.

Figure 2.3 shows an overview of the flow of computations in the dynamic simulations. The details of the "Forces and Moments" and "Dynamics" blocks vary depending on the aerodynamic and propulsion system design of the vehicle.

The dynamics of the design can be described using modified versions of the linearized equations from "Small Unmanned Aircraft: Theory and Practice" [3]. These equations have been modified to account for the difference in propulsion system between the TURAC design and a traditional fixed wing MAV. For the sake of linearization, the angle of the tilt rotors are taken to be constants for a particular flight regime (0° for fixed wing flight, 90° for hover). In fixed wing mode, the central coaxial fan is off and the tilt rotors are fixed horizontally. This means that, like a traditional fixed wing aircraft, the roll pitch and yaw motion of the vehicle are controlled

using the elevons and rudder. This standard system is an appealing part of this design as it is relatively simple and well understood. This breakdown of the dynamics of the VertiKUL design reveals several key issues. First, this design does not allow for the direct control of rolling motion in horizontal flight mode. This is because of the fixed orientation of the propulsion system and lack of aerodynamic control surfaces. Furthermore, the only way for this design to transition between hover and horizontal flight is through an elaborate arc maneuver. This transition is significantly more difficult to execute than the transition for the TURAC. In fixed wing mode, the lack of aerodynamic control surfaces reduces the controllability of the system making it a less useful fixed wing aircraft. After comparing both systems, the TURAC is the superior design from a dynamics and controls perspective. The two designs behave the same and are equally functional in hover mode. However, the TURAC has a simpler and easier to execute transition maneuver, and is a far superior design for fixed wing flight. In fixed wing mode it is more versatile and controllable making it the clearly superior design. The TURAC is also configured in such a way that conventional take-off and landing are possible.

2.3 Power & Propulsion

Selecting a rotary system that most efficiently provides the propulsive power necessary to power the craft is essential to having a good overall design. In addition, a battery needs to be selected such that the vehicle has enough energy to perform the maneuvers required in testing yet light enough to not weigh the craft down. If the battery has excess energy than what is needed for in flight maneuvers, then the battery will be taking up weight that could be used for a different aspect of the design. On the other hand, if the battery selected is too small, the aircraft will exhaust its battery before it can complete the mission. Therefore, it is important to accurately estimate the power consumption of the craft for the mission. Power will be drawn from both the on-board electronics on the aircraft as well as the four motors in order for the craft to ascend, descend and hover. In order to estimate how much power is required for a given mission,

power draw from the electronics and the motors will have to be estimated. To estimate power draw for the electronics, current draw will be measured from our electronics while they are running the same functionality as they would be during a mission. This current draw will then be multiplied by the expected run time of the electronics to calculate the power consumption, as in the equation below.

$$E = It\ [mAh] \tag{2.4}$$

Moreover, the power draw from motors for propulsive energy can be calculated using the following equations [13], first for a single motor:

$$P = \frac{1}{FM}\left(\frac{T^{3/2}}{\sqrt{2\rho A}}\right)\ [Watts] \tag{2.5}$$

And then for a ducted co-axial fan:

$$P = \frac{1}{FM}\left(\frac{T^{3/2}}{\sqrt{4\rho A}}\right)[Watts] \tag{2.6}$$

Where T is force of thrust [N], ρ is the air density constant [kg/m^3], A is area of the prop attached to the motor [m2] and FM is Figure of Merit. For now, the force of thrust required to hover will be consider to be equal to the force of weight acting against the aircraft. Moreover, FM is a an efficiency metric defined as the ratio of minimum power required(Ideal Case) to the actual power figured to hover [13]. Well-designed rotary systems have an FM value of anywhere between 0.6 and 0.8 depending on what application the system was specifically designed for while less efficient rotary systems have an FM closer to 0.5 [13]. FM is described as:

$$FM = \frac{P_{ideal}}{P_{actual}} = \frac{C_T^{3/2}}{\sqrt{2C_P}} \tag{2.7}$$

Where C_T is the coefficient of Thrust and C_P is the coefficient of power. These values must be determined experimentally using an apparatus such as a static thrust stand. In addition, power can be calculated using an alternative equation to equations 4 and 5 [13]:

$$T_i = C_T \omega_i^2 \tag{2.8}$$

Where ω_i is the angular speed of the rotor [rad/s]. However as previously stated C_T must be determined experimentally, thus this equation cannot be used if the motor is not available for testing. Due to the financial constraints of this project, equations 4 and 5 will be used when determining what motor to purchase online and then analyzed by using equations 19 & 20 with an FM of .6 and .8. By conducting this analysis, a reasonable range of thrust outputted from a motor can be determined analytically. If the motor is found to be within design criteria, the motor will be purchased and then tested using a static thrust stand, where the coefficient of thrust can be determined. Then equation 7 can be used to experimentally determine the expected thrust output of the motor.

2.4 Autonomy & Control

For most UAV applications, the aircraft is flown using the line-of-sight of the operator, however there are many situations where it is impractical for a UAV to be remotely operated. Therefore, developing an autonomous system capable of completing a predefined mission is a necessity in modern UAV design, especially when the vehicle must be able to track and avoid obstacles in real time. The depth of our autonomy scripting was determined by the modern definition of full autonomy. As reported in [4], an autonomous vehicle is one that is "capable of understanding higher level intent and direction." We interpreted this to mean that an autonomous vehicle is one that is fed a predetermined mission which it executes independently of human control and is

capable of adapting to changes in the environment. From this understanding, our autonomy and control research was split into two main subgroups: Sensor packages for finding information about the environment, and autonomy scripting for UAV's.

2.4.1 Sensors

To determine a sensor package we researched what our environment would look like, and what others have done in similar environments. Indoor testing for flight in hover mode means no GPS signal so the POZyx GPS spoofing setup will be used. The indoor area will likely be small and have obstacles which will take the shape of tall posts with foam coverings. This type of environment is ideal to use Robot Operating System (ROS) paired with SLAM. Outdoor testing for flight in fixed wing mode will allow for a GPS connection but faster forward flight. Faster flight requires faster processing for obstacle avoidance. These requirements for testing indirectly dictates the specifications of the sensors and on-board computers needed for the vehicle to perform the desired maneuvers.

In our research the typical autonomous vehicles used either rotational lidar or stereo camera systems. A rotational lidar will continuously scan the environment in every direction. This is typically a strong choice for SLAM because it is able to give information about the vehicle's surroundings in all directions. Another common choice is using two cameras in fixed location on the vehicle to determine the location of surrounding obstacles, this is often called stereo vision. Since this projects objective was a VTOL UAV, minimizing weight was extremely important. Other important factors included maximum and minimum sensing distance, power requirement, data bus, and price. 2.2 shows these attributes for each sensor researched.

From looking at the table you will see that the RealSense camera was the lightest choice by far. While this option only allows us to detect in one direction it also will give us the best data in the vertical Z direction when looking ahead. Also, reaction speed is extremely important in fixed wing mode, so by only looking in the direction of travel we can focus on the obstacles ahead instead of obstacles to the sides, above, below, and behind because those are not a current

Lidar	Weight (g)	Power (V)	Bus	Detectable Range (m)	Price
Hokuyo URG-04LX-UG01 Scanning Laser Rangefinder	160	5	USB mini B	0.02 - 5.6	$ 1080
YDLIDAR X4 360° Laser Scanner	189	5	USB mini B UART	0.12 - 11	$ 100
Robotics 360° Laser Distance Sensor LDS-01	125	5	USB mini B UART	0.12 - 3.5	$ 180
YDLIDAR F4 Pro 360° Laser Scanner	186	5	USB mini B	0.08 - 12	$ 170
YDLIDAR G4 360° Laser Scanner	214	5	USB mini B	0.08 - 16	$ 325
Intel Realsense 400 Series	72	5	USB 3.0	0.1 - 10	$ 190

Table 2.1: Comparison of different Lidars

threat. When in hover mode the vehicle can rotate to detect objects in its surrounding. This makes the RealSense the best choice for our vehicle.

Distance Sensor	Weight (g)	Power (V)	Bus	Distance (m)	Divergence (deg)	Price
Sharp GP2Y0D810Z0F Digital Distance Sensor	1.3	5	3 Pin (Power, Ground, Signal)	0.1	Unlisted	$ 7
Benewake TFMINI Micro LIDAR Module	6	5	UART	12	2.3	$ 40
LIDAR-Lite 3 Laser Rangefinder	22	5	I2C	40	0.5	$ 130
SF10/A	35	5	Serial, I2C, USB	25	0.4	$ 279

Table 2.2: Main Sensor Research

Once the main sensor was chosen the vehicle still required sensors to determine the vehicles height above the ground. To determine the height above the ground laser and IR distance sensors. Again emphasizing light weight components parts were researched and a table was made of possible sensors. Other important factors included maximum and minimum sensing distance, power requirement, data bus, and price. 2.2 shows these attributes for each sensor researched. After looking at many sensors it was concluded that with the IR sensors they have very limited range. This narrowed down the options to the laser range finders. The LIDAR-

Lite 3 Laser Rangefinder because of its range of distances and relative light weight for those distances.

The sensors to be chosen for localization, object avoidance,and target detection were the most suitable based on weight and ability to achieve our goals. For localization indoors we chose to use the Pozyx system because it had been used previously by MQP teams and is still being used by WPI research students. For outdoor localization a much larger system needed to be used, so we chose to use GPS for outdoor localization. Finally we chose to use the Raspberry Pi Pi cam because it was light weight at 9 grams, and already available in the lab.

2.4.2 Computer

Choosing an on board computer is a difficult task, especially when the software that will be running on that computer has not been written yet. To make an educated choice on the computer requirements we looked at the computers used by similar projects. The main project we based our computer choice on was on an obstacle avoiding flying wing [2], because that part of our mission we predicted would require the fastest computing time. In this paper Barry used 2 Odroid XU3 computers on board. That computer is most similar to the last computer listed in the table below. While using 2 Odroid computers was a strong choice the Jetson TX2 was a more powerful option in terms of computing cores and RAM than 2 seperate Odroid computers.

SBC	**Weight (g)**	**CPU**	**RAM (GB)**	**CPU Speed**	**Memory**	**Ubuntu**	**Physical Camera Support**
Raspberry Pi	45	ARM Cortex-A53 (64-bit)	1	1.4 GHz Quad core	32 GB MicroSD	Yes	5
Jetson TX2	85	4x ARM Cortex-A57 + NVIDIA Denver2 (dual-core) (64-bit)	8	2 GHz HexaCore	32GB & 128GB MicroSD	Yes	2
Up Core	45	Intel x86_64 (64-bit)	4	1.92 GHz Quad core	64GB	Yes	3
Intel Compute Stick	54	Quad-core Intel Atom Processor	1	1.33 GHz Quad Core	8GB & 64GB MicroSD	Yes	1
Odroid-XU4	60	ARM Cortex-A15 (32-bit)	2	2 GHz Octa Core	64GB MicroSD	Yes	3

Table 2.3: Computer Research

2.4.3 Software in the loop, Mission Planner, OpenCV, & Dronekit

Over the past ten years, several consumer options for autonomous UAV control have become available. While the face of the consumer market has been DJI since the advent of low-cost commercial UAV's, there are several open-source options that allow for unparalleled control and creativity with vehicle design. The predominant software for autonomous control is ArduPilot, an open-source package that contains tools for controlling a variety of vehicles (cars, planes, quadcopters, submarines, etc) as well as resources for simulating vehicle missions. However, ArduPilot does not have native support for OpenCV and while it does support python scripting, it does not include a refined library for controlling the vehicle. In order to reduce our workload while creating scripts for the vehicle, we will be using a tool for interfacing with UAVs called Dronekit. For example, when writing python scripts without the dronekit library, we would have to spoof radio transmitter pulse-width modulation values to control the vehicle. With Dronekit, we write vehicle.simple_takeoff(altitude) to have the vehicle takeoff to a specified altitude after the motors are armed. Due to our limited experience in autonomous control, we utilized ArduPilot's native support for Software-In-The-Loop (SITL) simulation. This allows us to safely practice our autonomy scripts on a vehicle so that we may optimize them before buying the parts to build. Thus, simulation testing keeps us safe and keeps us from damaging the vehicle during the extensive testing required by writing autonomy scripts. Furthermore, we used the program Gazebo as a physics sandbox for operating our vehicle in a known environment to better visualize how different control inputs affect the vehicle we have designed. This simulation can be seen in Figure 2.4. To check our vehicle's status, we utilized MavProxy, a stripped-down ground control station that would output the simulated state of the vehicle as it performed its mission in Gazebo. Furthermore, since Gazebo interfaces with the Robot Operating System (ROS), we were able to simulate stereo camera outputs and point clouds as outputted by Intel's Realsense camera for use with OpenCV. This allowed us to practice object detection in the simulation. Finally, we were able to remotely interact with the simulated vehicle and upload missions in Mission Planner as shown in Figure 2.5. We effec-

tively proved the functionality and value of SITL, but ultimately decided to focus our efforts elsewhere as the project progressed.

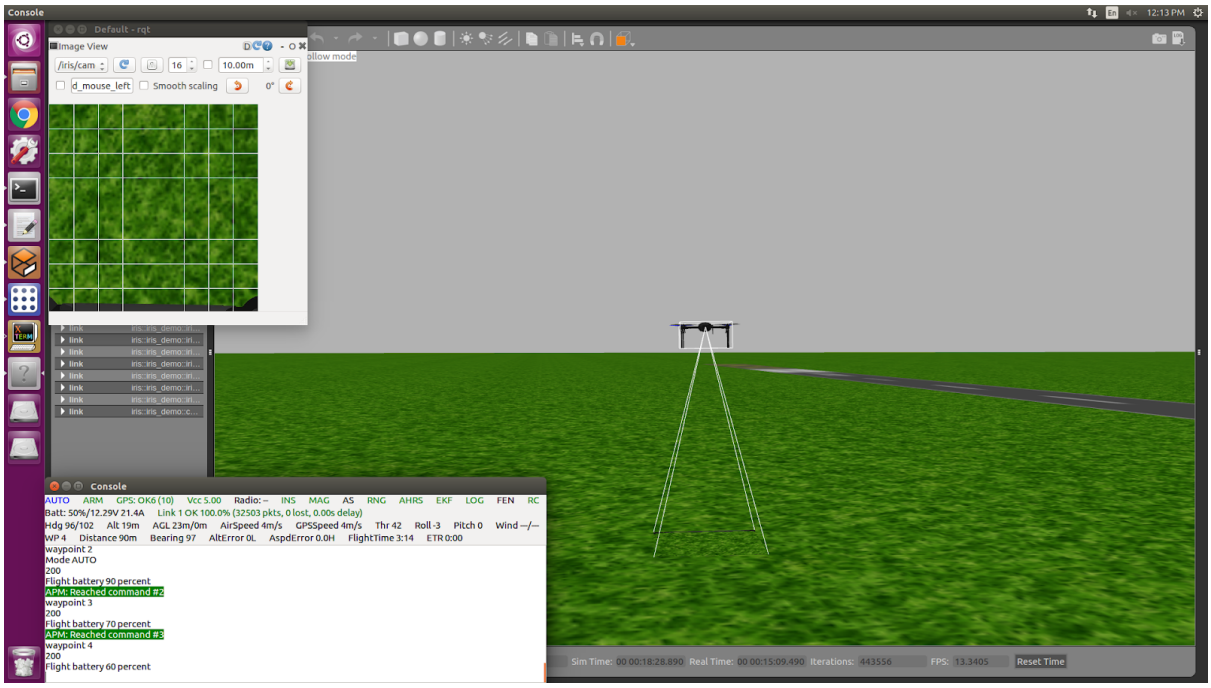


Figure 2.4: A Gazebo Simulation demonstrating how the vehicle could be tested before building it.

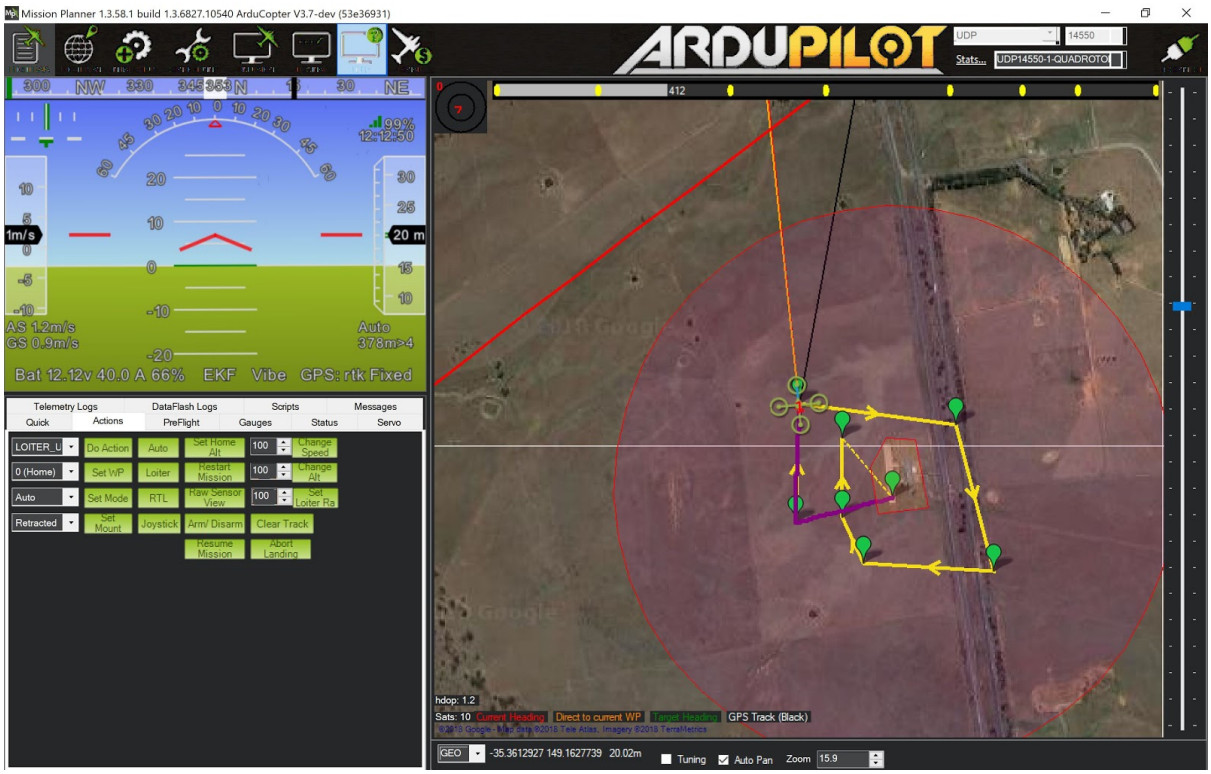


Figure 2.5: A pre-planned mission in Mission Planner running SITL.

Chapter 3

System Design

3.1 Aerodynamics & Structures

As stated by the objective of this project, the main goal is to design a hybrid micro air vehicle capable of both Vertical Take Off and Landing (VTOL) and classical fixed wing flight. Our previous analysis of the two case studies set forth by the VertiKUL and the TURAC demonstrated that the tilt-rotor design of the latter model is far more capable in regards to our main goal. The design would benefit pick up and drop off missions typical to a conventional helicopter but would also allow for much higher cruise speeds. The TURAC uses a blended body design, which allows for increased lift provided by the body and wing components. The design further allows for removable wings which will allow for the aircraft to be operated solely as a traditional quadcopter depending on the intended mission. With this design as a foundation, we performed further analysis into the aerodynamic and structural design of our own model should be scaled to an appropriate size to qualify for the competition.

3.1.1 Airfoil Selection

The airfoils we researched to use as the main wing airfoil were the NACA 34112, S1223, and the E423. The main objective in selecting an airfoil was to find a certain airfoil with

significantly high lift and low drag at low Reynolds numbers. The reason for this requirement was the high weight of our vehicle paired with the need for slow cruise speeds for the real-time image processing for obstacle avoidance. These high-lift airfoils listed, particularly the S1223 and the E423 lead to increased payloads, shortened takeoff and landing distances, and lowered stall speeds (Source). These beneficial effects of improved high-lift airfoil aerodynamics are crucial for our design. We first chose the same airfoil the TURAC's design obtained (NACA 34112) as a standard benchmark to be able to compare performance against the other airfoil selections. The NACA 34112, pictured in Figure 3.1, shows a 12 percent thickness as well as a 20 percent reflex at the camber position.

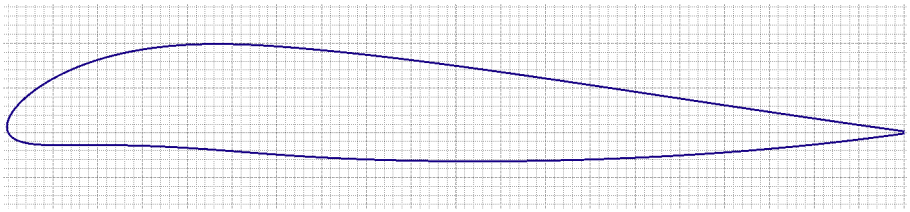


Figure 3.1: 34112 Airfoil Shape

From past MQP designs, it was shown that the Selig S1223 was one of the more common airfoils for high lifting aircraft's similar to the TURAC's NACA 34112. The Selig S1223 was once tested in the 1990s to have a Reynolds number of (2 x 10^5) while yielding a maximum coefficient of 2.2 when tested in the wind tunnel by Michael Selig. The S1223, shown in Figure 3.2, displays a max thickness of 12.1 percent of the chord length at 19.8 percent of the chord behind the leading edge, and its max camber of 8.1 percent of the chord length at 49 percent of the chord behind the leading edge.

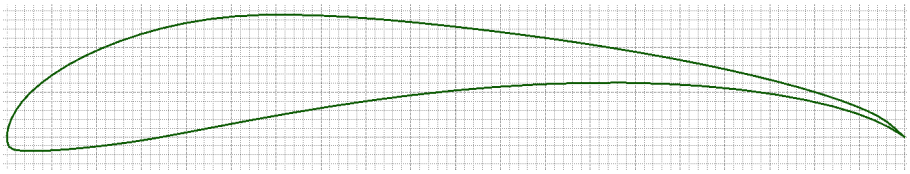


Figure 3.2: S1223 Airfoil Shape

Further research into high lift airfoils yielded the E423 as another valuable airfoil option because of its ability to produce a high lift, have a low drag and for its increased surface area. The Eppler 423, shown in Figure 3.3, displays a max thickness 12.5 percent at 23.7 percent chord and max camber 9.5 percent at 41.4 percent chord.

Figure 3.3: E423 Airfoil Shape

3.1.2 XFLR 5 Analysis

Once this list of airfoils was finalized, further analysis of the aerodynamic performance was required. These three airfoils were tested in a program known as XFLR5 for their respective capabilities in theoretical conditions to what our vehicle would experience in flight. To test aircraft performance, the theoretical values required were each airfoil's coefficient of lift at alpha, coefficient of pitching moment at alpha, coefficient of drag at alpha, and the lift to drag ratio at alpha. Each airfoil was tested at a 12m/s cruise speed over a range of angle of attacks (α). The Reynolds number tested for all airfoils was at a constant 125,000. Results of lift coefficient as a function of α from XFLR5 are shown below in Figure 3.4. The green line represents the data given by the S1223 airfoil, the red represents the E423, and the blue line represents the NACA 34122. As expected, the S1223 and the E423 had a higher lift coefficient than the TURAC's NACA 34112 from α of 6° and up. The S1223 and the E423 followed similar performance trends as both airfoils coefficients of lift increase significantly after angle of attack of 3° while the NACA 34112 only gradually increases. The S1223 performs the best of the three airfoils with a coefficient of lift of 1.1 at 12 α and the E423 is next with a value of 0.95 at the same alpha.

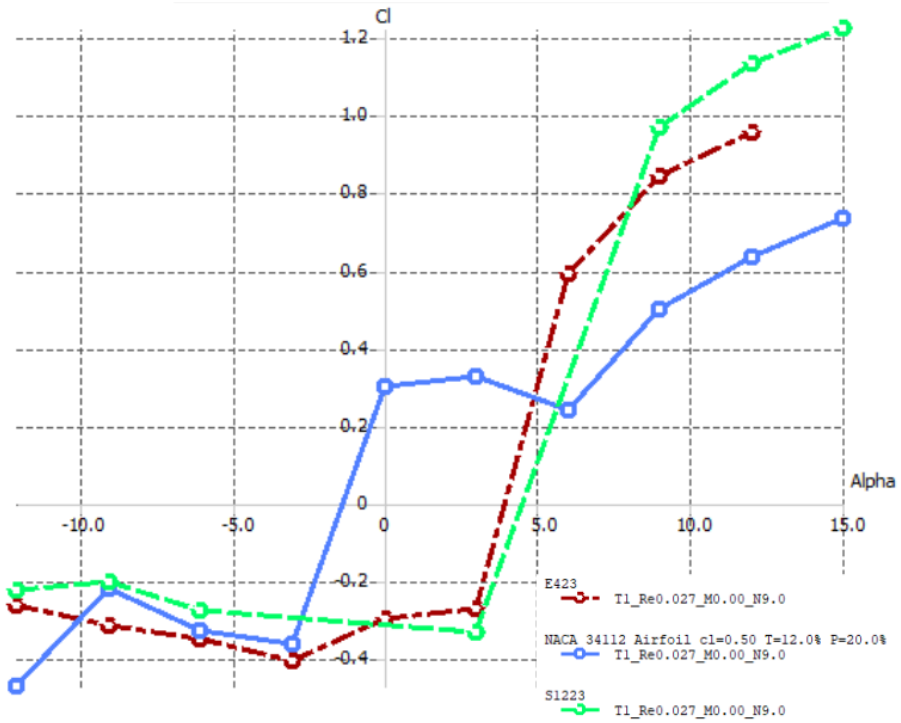


Figure 3.4: XFLR5 Coefficient of Lifts

The next value we tested for was the coefficient of pitching moment due to α. The static stability of an aircraft is dependent on this value and thus we wanted to ensure that whatever airfoil we chose would be able to provide this type of stability mid-flight. The XFLR5 results are shown below in Figure 3.5. As seen in the graph, the trim angle (the angle where pitching moment is zero) for both the S1223 and E423 is -6° to -7°. For the NACA 34112, the trim angle is closer to -3°. While it is more convenient for the trim angle to be closer to zero for stability purposes, this is not overly hard to correct for. A larger deflection angle for the elevator during flight will be needed in order to shift the moment curves of the S1223 and E423 as compared to the NACA 34112.

Another value we looked for was the coefficient of drag due to α. The XFLR5 results are shown below in Figure 3.6. The E423 and the S1223 demonstrated about the same drag coefficients due to α until the 13 ° angle of attack where the S1223 had a higher drag coefficient at 0.223 compared to the E423 at 0.187. It is important, particularly with our vehicle, that the airfoil minimizes drag as much as possible due to its high weight and low speed of flight.

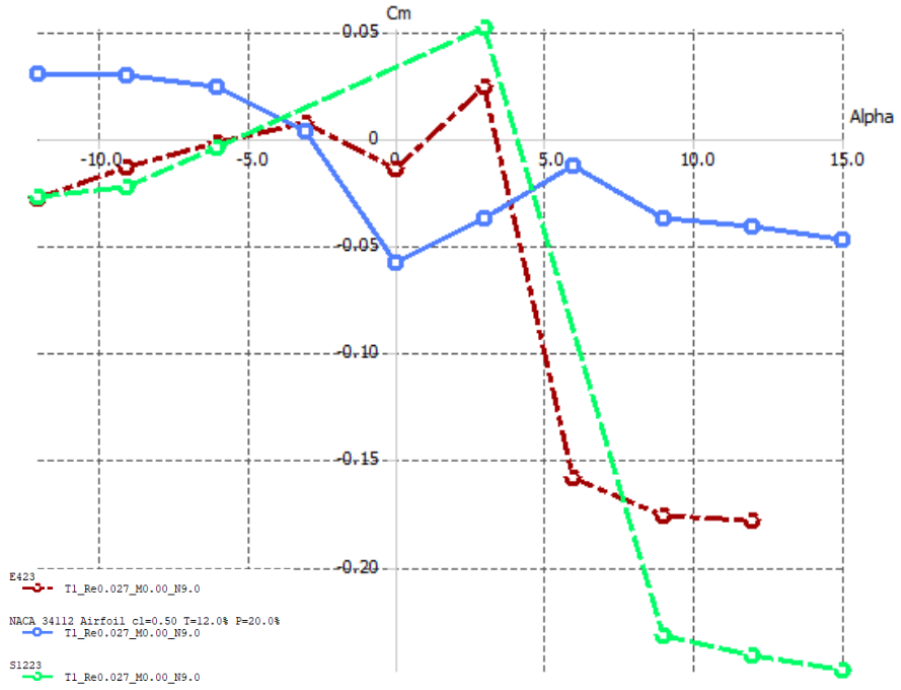


Figure 3.5: XFLR5 Moment Coefficients

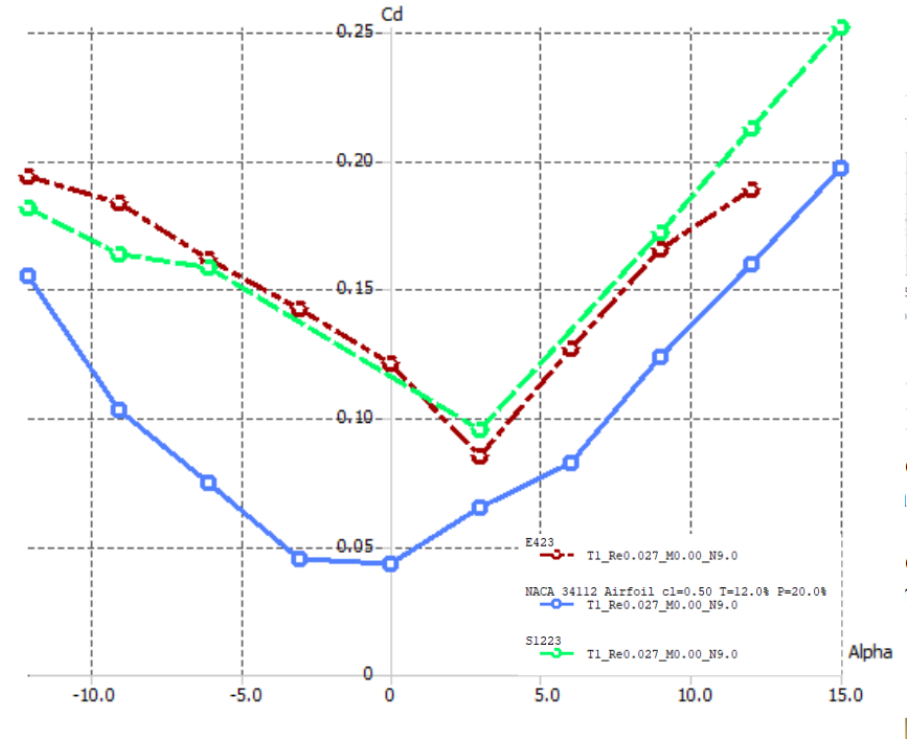


Figure 3.6: XFLR5 Drag Coefficients

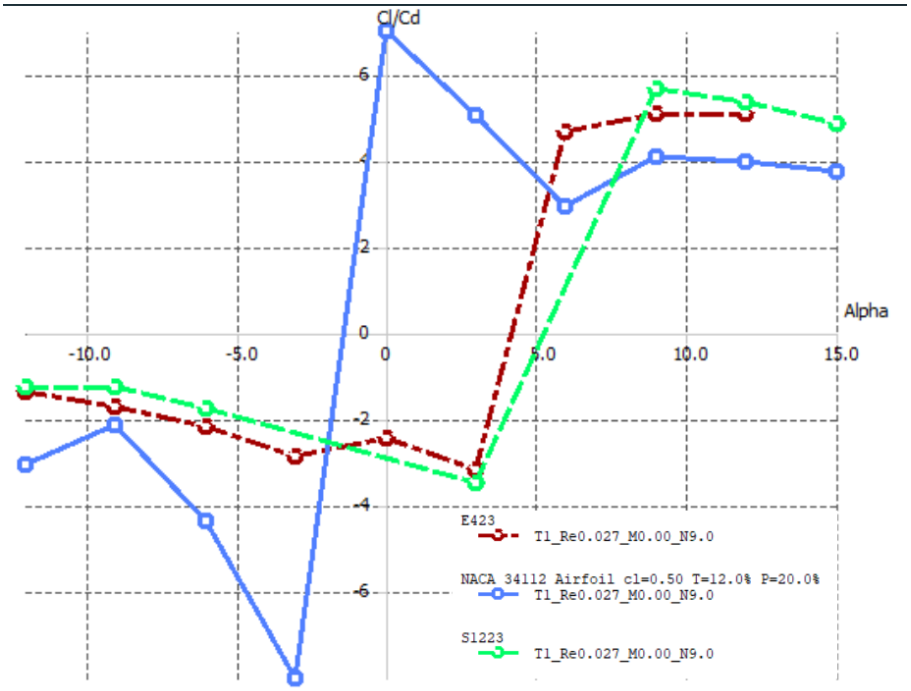


Figure 3.7: XFLR5 C_L/C$_D$ curves

Lastly, our team tested for the Lift to Drag Ratio for all three airfoils based off the same cruise speed, Reynolds number, and alpha we used for the lift, moment, and drag Coefficients. The XFLR5 results are shown below in Figure 3.7. The graph below shows the NACA 34122 having its highest lift to drag ratio of 6.62 at zero degrees angle of attack. The E423 and the S1223 performed class to each other peaking at around 9°s angle of attack where the S1223 had a higher lift to drag ratio of 5.86 compared to E423 lift to drag ratio of 5.24.

3.1.3 Wind Tunnel Testing Analysis

The next step in selecting the main wing airfoil was to get experimental values of lift, pitching moment, and drag to compare with the values we received from XFLR5. Three 3D printed models of our vehicle with the three different airfoils were tested in a wind tunnel over a range of flight conditions in order to obtain these values. The 3D printed models were 1/3rd size model of our vehicle and were created using SolidWorks with only the airfoil shape varying. Each airfoil was tested at speeds ranging from 10 m/s to 30 m/s increasing by 2 m/s increments.

The angle of attack (α) ranged from -12° to +15°s (limited by the physical motion of the mounting mechanism) moving at 3-degree increments. Results of lift coefficient as a function of alpha from wind tunnel testing are shown below in Figure 3.9. The orange line represents the data given by the S1223 airfoil, the blue represents the E423, the grey line represents the NACA 34122, and the yellow line is the new body design we eventually had to develop for use over the streamlined body of the TURAC design. The reason for this "new body" design was the lack of enough space for all electrical and mechanical components in the body. The values obtained from wind tunnel underwent post-processing as the wind tunnel provided only the normal and axial forces as well as the pitching moment about a point other than the center of gravity of the models. The following equations were used to convert those forces and moments to the actual aerodynamic forces and moments we needed.

$$D = N \sin \alpha + A \cos \alpha \tag{3.1}$$

$$L = N \cos \alpha - A \sin \alpha \tag{3.2}$$

$$M_{AC} = M_P - BN \tag{3.3}$$

Where N is the Normal force measurement, A is the Axial force measurement, α is the angle of attack, M_P is the measured pitching moment and B is the distance between the measured moment and the aerodynamic center.

Once these values were converted, they then needed to be non-dimensionalized in order to obtain the coefficients we were looking to compare. That was done using the following expression.

$$\frac{1}{2}\rho v^2 S \tag{3.4}$$

Where ρ is the density of air, v is the airspeed, and S is the wetted area of the wing or model being tested

Finally, after non-dimensinalizing results, graphs could be produced for each model and the

coefficients could be compared. Graphs were created for each coefficient in two ways. The first being the graph where the average of all airspeeds for each data point of α were taken and then graphed. The second graph, the ones labelled with "20+ m/s", are the average for all airspeeds 20m/s and above for all alphas. The reason we did this is due to the higher accuracy of the wind tunnel data readout at higher Reynolds numbers, as pointed our to us by the co-advisor to this project (Professor Olinger). Thus, we thought it best to create separate graphs of just the averages of those values for comparison for accuracy's sake.

Observing the graphs for coefficient of lift vs α, as expected during the testing, the S1223 had the highest lift coefficient followed by the E423 and then the NACA 34122. Ultimately, both the E423 and the S1223 outperformed the Turac design, which was expected, based off the XFLR5 analysis. As seen in both Figures 3.8 and 3.9, the S1223 performs best. However, the E423 is a very close second. While the exact numbers from the XFLR5 simulation do not align with this real-world test, the general trends followed by each airfoil remain generally the same. While comparing the theoretical data with the experimental data, keep in mind that the numbers from XFLR5 are ideal and do not account for the effects of the body of the vehicle while the wind tunnel tests do. This physical difference accounts for the difference in numbers. Considering that fact, the numbers from the wind tunnel test align closely enough to the results achieved in the XFLR5 simulation and back up that analysis of lift coefficients.

Results of the pitching moment as a function of α from wind tunnel testing are shown below in Figure 3.10. The performance of all three airfoils were similar in that the pitching moment occurs at negative angle of attack, which was also reported Ozdemir's, "Design of a Commercial Hybrid VTOL UAV System" for the TURAC design. Comparing both the experimental data (wind tunnel tests) to the XFLR5 results, we notice that both sets of trends have negative slopes, denoting static stability. Also, for the pitching moment data, the actual values of the wind tunnel tests follow relatively close to the magnitude and trends of XFLR5 values, again pointing towards a correct original analysis. The one point of note with this set of data is the value of trim angle of attack. For the S1223 and E423, the trim alphas are -6° and -8°, respec-

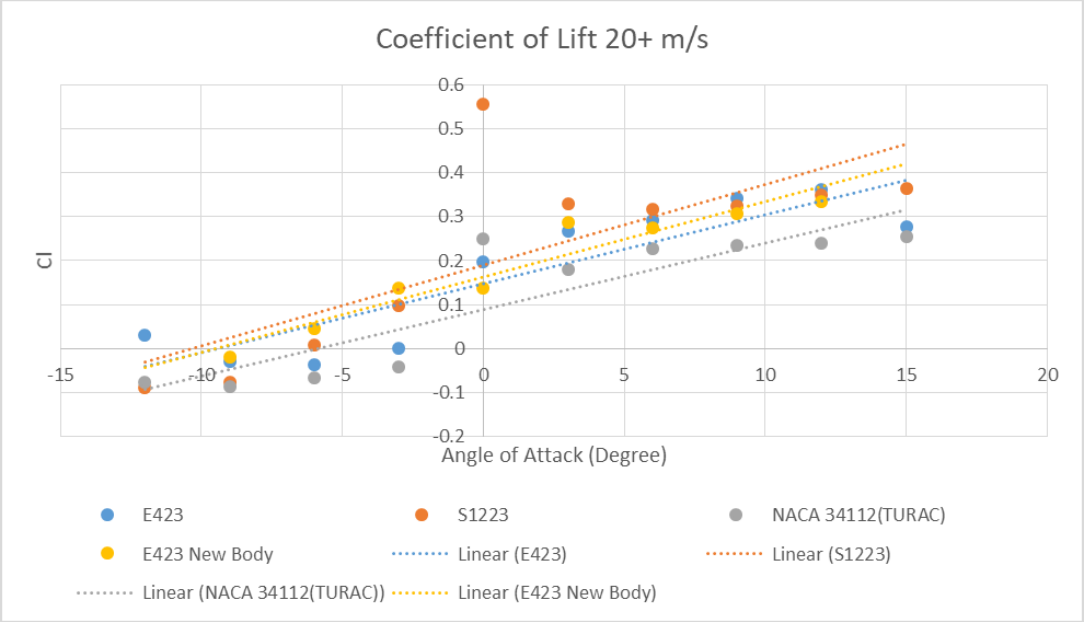


Figure 3.8: Measured Lift Coefficient All Airspeeds

tively. Similarly to the XFLR5 values seen in Figure 3.5, these values of trim are not convenient for flight, but can be corrected for with the right elevator deflection angle.

Results of the drag as a function of α from wind tunnel testing are shown below in Figure 3.12. The experimental data again follows similar trends to the XFLR5 data, with the NACA 34112 creating the most drag over the range of angles of attack and the E423 and S1223 performing similarly to each other.

Lift to drag ratio as a function of α results from wind tunnel testing are shown below in Figure 3.14. The peak of CL/CD occurs at the similar places for all three airfoils and the highest peak was from the S1223 at 6°. These graphs have similar appear similar when compared to Figure 3.7 analysis by XFLR5 except for the values of the NACA 34112. This variance arises from a difference in drags values which could be reasoned for by XFLR5 not accounting the body of the models.

Overall, both the S1223 and the E423 outperformed the NACA 34112 in all categories as expected. This allowed us to rule out the NACA 34112 airfoil from our airfoil selection. When it came down to the E423 and the S1223, even though the S1223 generated a higher lift, the

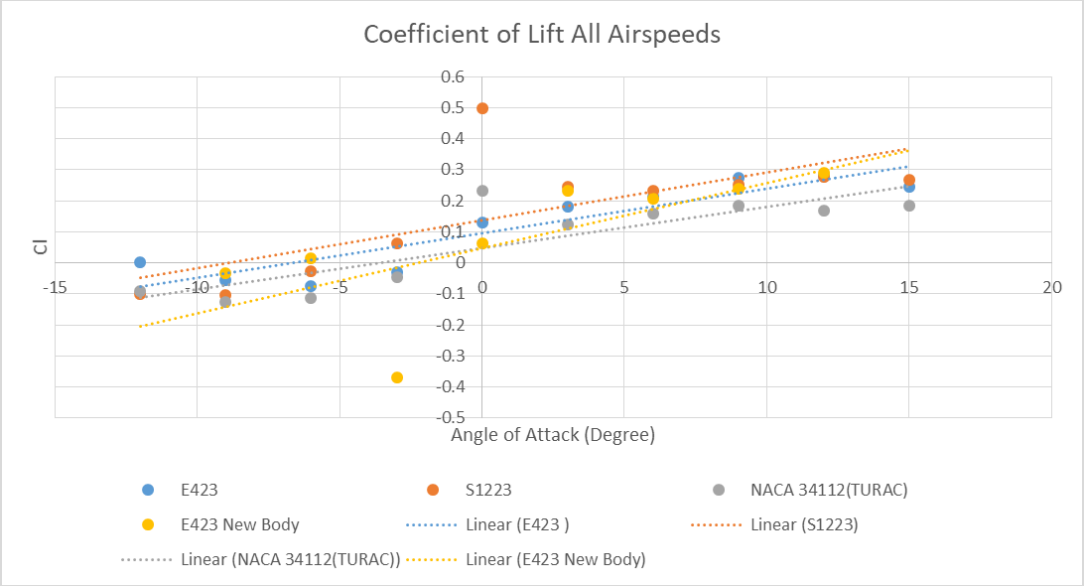


Figure 3.9: Measured Lift Coefficient 20+ M/S

E423 has a similar pitching moment, drag, and lift to drag ratio. The final deciding factor is in fact not based on aerodynamics, but ease of manufacturability. When creating the airfoils, the S1223 has a fragile trailing edge that tapers off to a very fine point, while the E423's trailing edge remains more modest in comparison. This allows the E423 to be a stronger and easier build. Ultimately, through the XFLR5 performance numbers, wind tunnel analysis, and manufacturing a structural sound aircraft, the team decided to select the E423 as the vehicle's main wing airfoil.

With this airfoil selected, the team then produced multiple test models. After the first two iterations of glide test models were tested as mentioned in section 5.1, the team decided that a redesign was required to fix the unstable dutch roll mode for stick-fixed flight. For this redesign, the wings were shaped into delta wings with a leading edge sweep. The wingspan was increased by about 2 inches on each side and a 5° dihedral was added for roll and yaw stability. The reason delta wings were chosen over the original strait wing design was because the delta wings provided a greater surface area and thus a great lift capacity in order to maintain the low Reynolds numbers at the high weight that we would be flying at. An iterative test process in XFLR5 of design and test was utilized to refine some of the specifics in order to get the exact

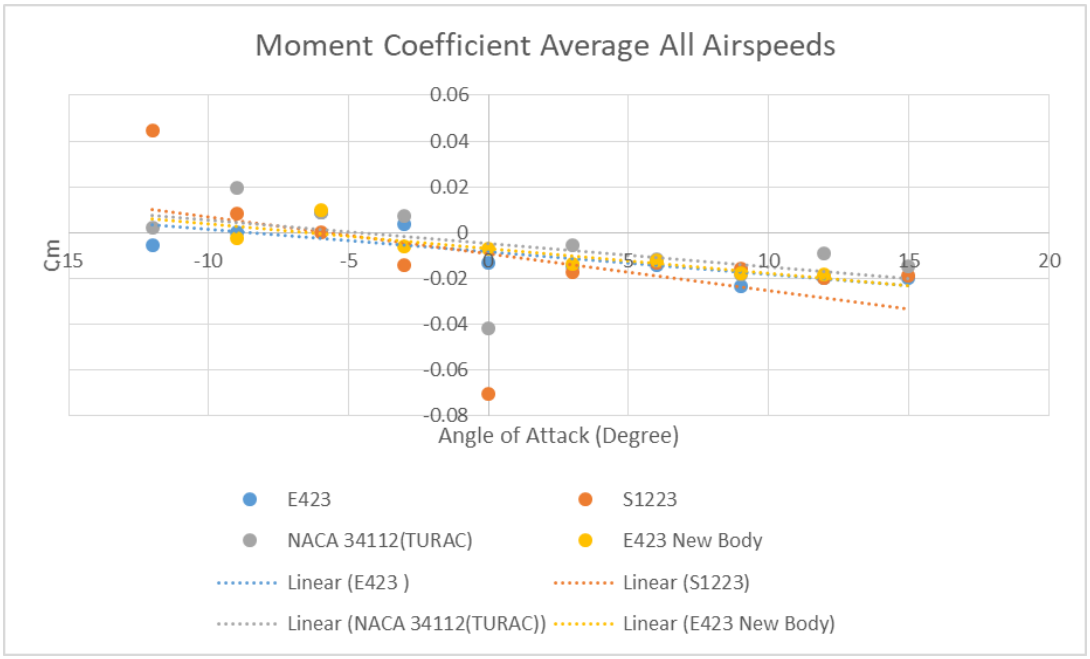


Figure 3.10: Moment Coefficient All Airspeeds

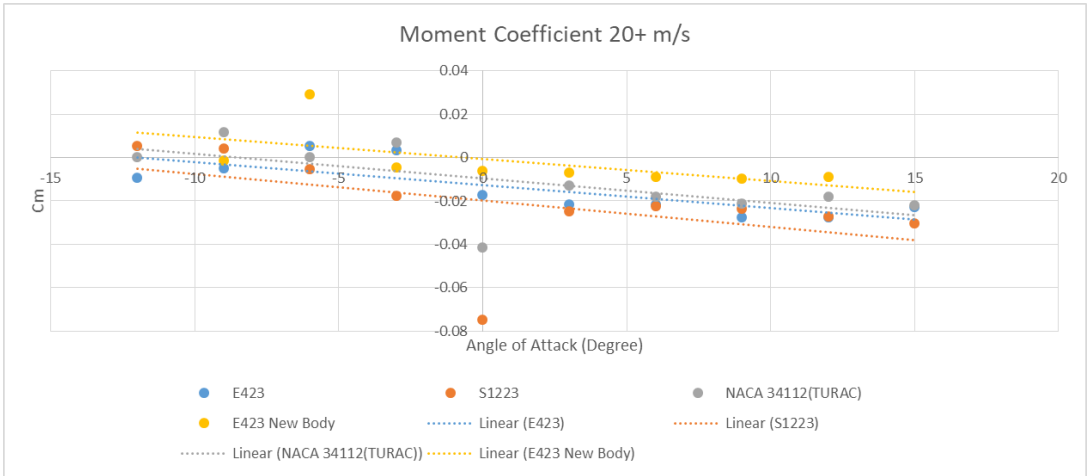


Figure 3.11: Moment Coefficient 20+ m/s

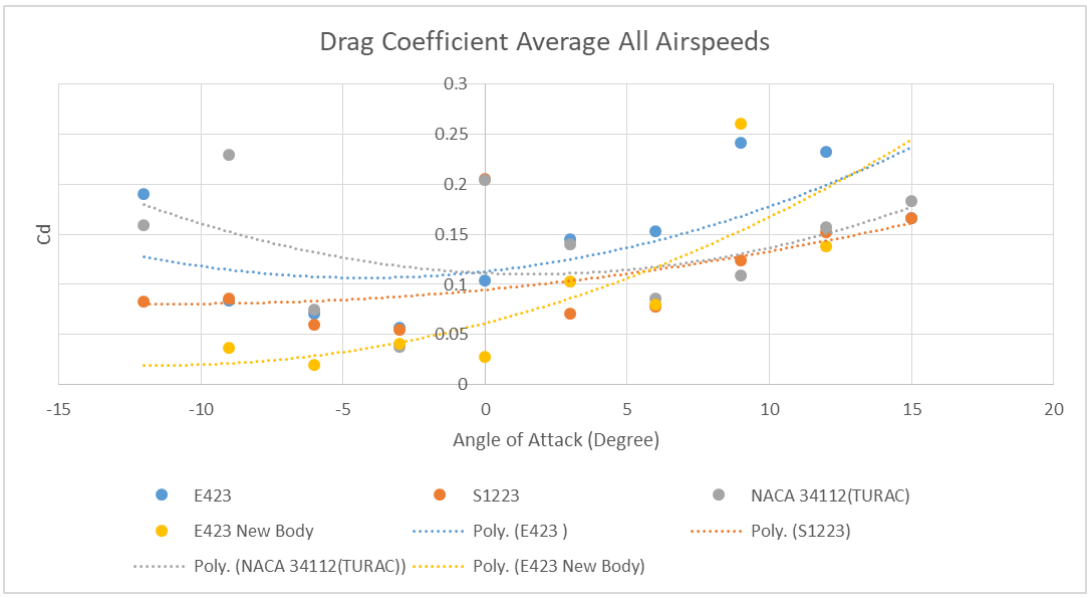


Figure 3.12: Drag Coefficient All Airspeeds

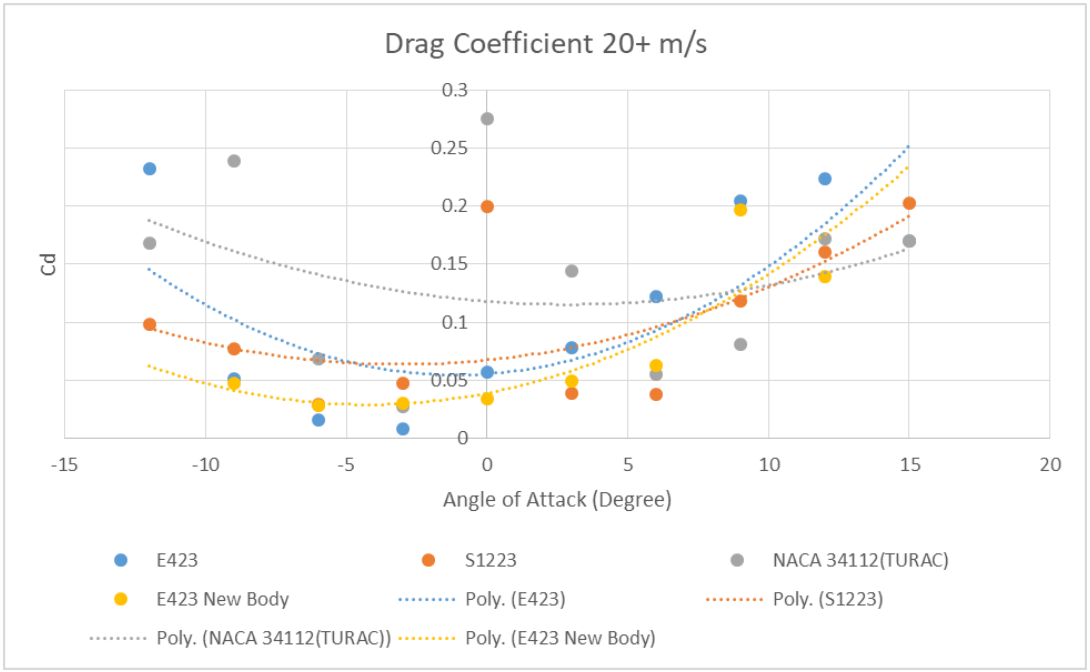


Figure 3.13: Drag Coefficient 20+ m/s

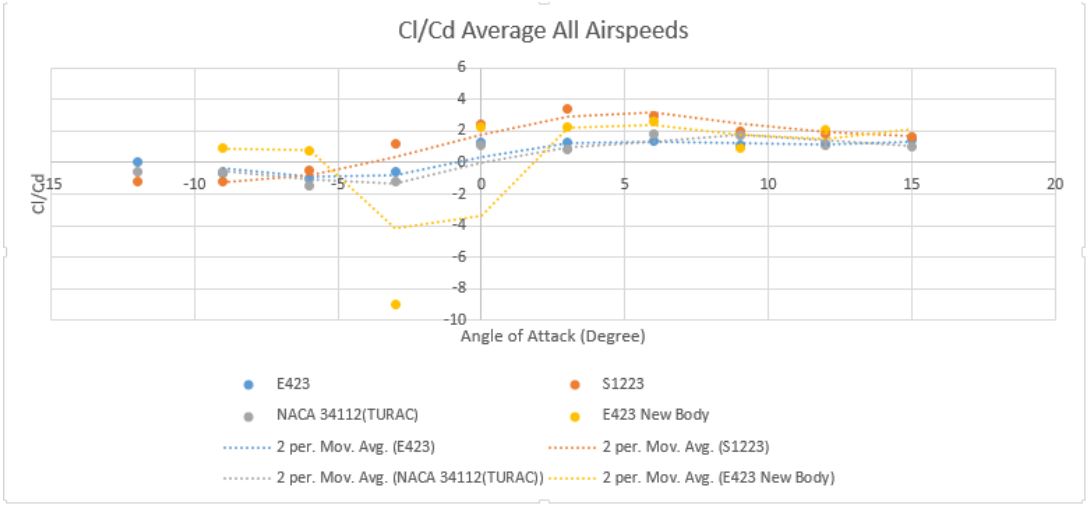


Figure 3.14: Cl/Cd All Airspeeds

sizing chosen for the wings. The final design and aerodynamic curves can be seen in Figures 3.16 and 3.17 below.

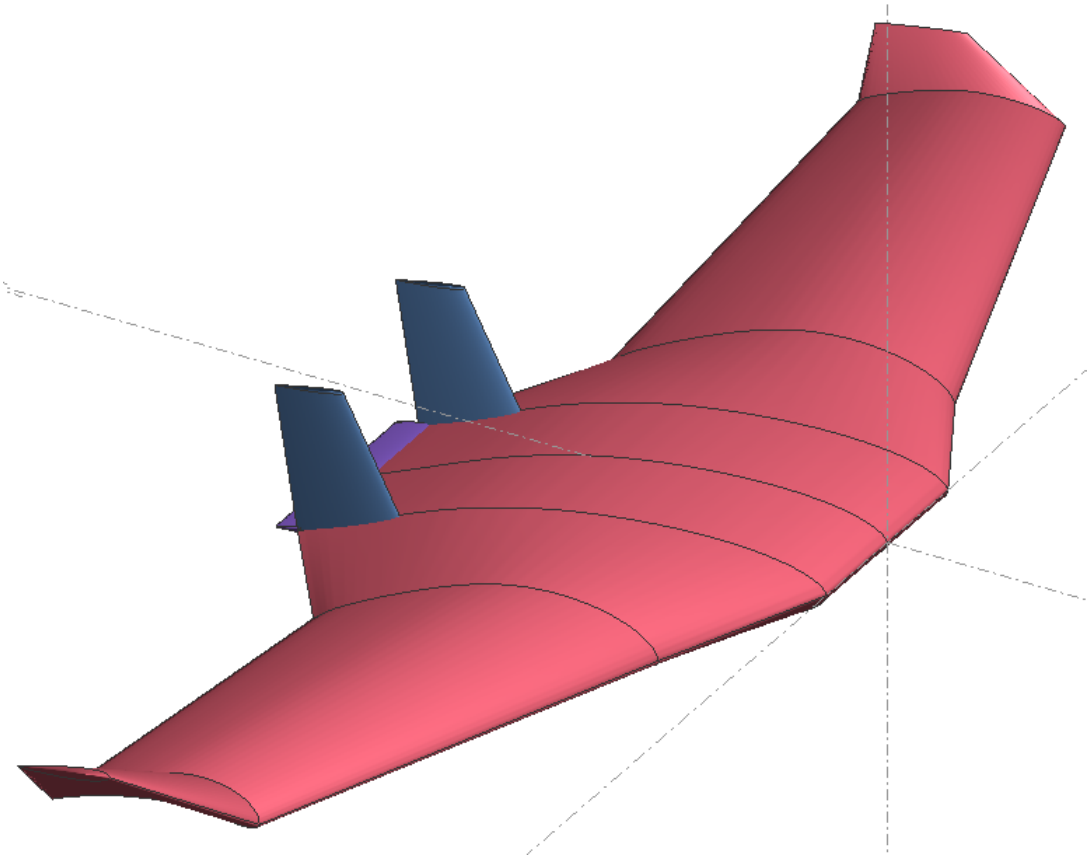


Figure 3.16: New Model Design

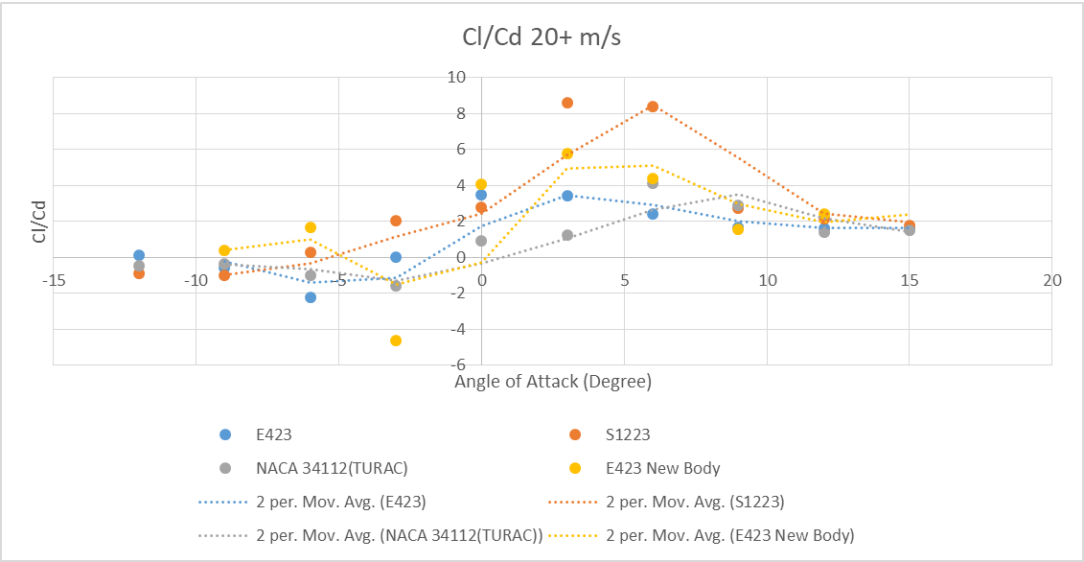


Figure 3.15: Cl/Cd 20+ m/s

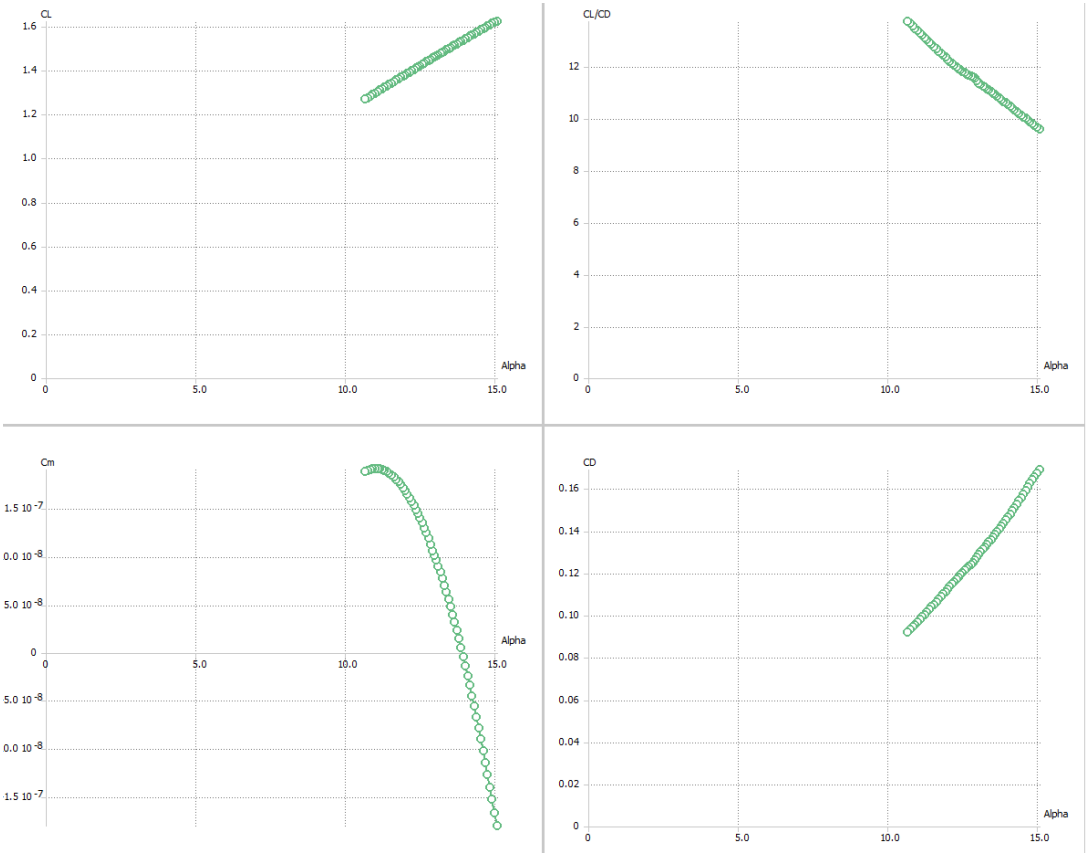


Figure 3.17: XFLR5 Full Plane Stability

3.2 Simulation & Software

3.2.1 Trim State

The objective of this simulation was to design a controller about a trim condition. In order to do this we first needed to find a trim condition. The dynamic model of the system at a trim condition is expressed as:

$$\begin{aligned}
\dot{h} &= V_a \sin\gamma \\
\dot{u} &= 0 \\
\dot{v} &= 0 \\
\dot{w} &= 0 \\
\dot{\phi} &= 0 \\
\dot{\theta} &= 0 \\
\dot{\psi} &= \frac{V_a}{R}\cos\gamma \\
\dot{p} &= 0 \\
\dot{q} &= 0 \\
\dot{r} &= 0
\end{aligned} \tag{3.5}$$

Where V_a is the trim airspeed, R is the radius of turn, and γ is the trim climb angle. The next step was to calculate the trim state and control inputs for the system given a specified V_a, R and γ. In order to narrow down the number of variables in the system we used angle of attack (α), sideslip angle (β), and roll angle (ϕ) as the inputs to the trim function we created. The entire state was calculated using these six values given the equations:

$$
\begin{aligned}
u &= V_a \cos\alpha \cos\beta \\
v &= V_a \sin\beta \\
w &= V_a \sin\alpha \cos\beta \\
\theta &= \alpha + \gamma \\
p &= -\frac{V_a}{R} \sin\theta \\
q &= \frac{V_a}{R} \sin\phi \cos\theta \\
r &= \frac{V_a}{R} \cos\phi \cos\theta \\
\psi &= 0
\end{aligned}
\tag{3.6}
$$

Note Γ_x refers to the inertia matrix products defined in Beard & McLain [3] Then using the fully defined dynamical model of the aircraft from the Beard & McLain textbook [3] we calculated the control inputs and trim condition.

$$
\begin{aligned}
\dot{p}_n &= (\cos\theta\ \cos\psi)u + (\sin\phi\ \sin\theta\ \cos\psi - \cos\phi\ \sin\psi)v + +(\cos\phi\ \sin\theta\ \cos\psi + \sin\phi\ \sin\psi)w \\
\dot{p}_e &= (\cos\theta\ \sin\psi)u + (\sin\phi\ \sin\theta\ \sin\psi + \cos\phi\ \cos\psi)v + (\cos\phi\ \sin\theta\ \sin\psi - \sin\phi\ \cos\psi)w \\
\dot{h} &= u\sin\theta - v\sin\phi\ \cos\theta - w\cos\phi\ \cos\theta \\
\dot{u} &= rv - qw - g\sin\theta + \frac{\rho V_a^2 S}{2m}C_{x_0} + C_{x_\alpha}\alpha + C_{x_q}\frac{cq}{2V_a} + C_{x_{\delta_e}}\delta_e + \frac{\rho S_{prop}C_{prop}}{2m}(k_{motor}\delta_t)^2 - V_a^2 \\
\dot{v} &= pw - ru + g\cos\theta\sin\phi + \frac{\rho V_a^2 S}{2m}C_{y_0} + C_{y_\beta}\beta + C_{y_p}\frac{br}{2V_a} + C_{y_r}\frac{br}{2V_a} + C_{y_{\delta_a}}\delta_a + C_{y_{\delta_r}}\delta_r \\
\dot{w} &= qu - pv + g\cos\theta\cos\phi + \frac{\rho V_a^2 S}{2m}C_{z_0} + C_{z_\alpha}\alpha + C_{z_q}\frac{cq}{2V_a} + C_{z_{\delta_e}}\delta_e \\
\dot{\phi} &= p + q\sin\phi\tan\theta + r\cos\phi\tan\theta \\
\dot{\theta} &= q\cos\phi - r\sin\phi \\
\dot{\psi} &= q\sin\phi\sec\theta + r\cos\phi\sec\theta \\
\dot{p} &= \Gamma_1 pq - \Gamma_2 qr + \frac{1}{2}\rho V_a^2 SbC_{p_0} + C_{p_\beta}\beta C_{p_p}\frac{bp}{2V_a} + C_{p_r}\frac{br}{2V_a} + C_{p_{\delta_a}}\delta_a + C_{p_{\delta_r}}\delta_r \\
\dot{q} &= \Gamma_5 pr - \Gamma_6(p^2 - r^2) + \frac{\rho V_a^2 Sc}{2J_y}\left[C_{m_0} + C_{m_\alpha}\alpha + C_{m_q}\frac{cq}{2V_a} + C_{m_{\delta_e}}\delta_e\right] \\
\dot{r} &= \Gamma_7 pq - \Gamma_1 qr + \frac{1}{2}\rho V_a^2 SbC_{r_0} + C_{r_\beta}\beta + C_{r_p}\frac{bp}{2V_a} + C_{r_r}\frac{br}{2V_a} + C_{r_{\delta_a}}\delta_a + C_{r_{\delta_r}}\delta_r
\end{aligned}
\tag{3.7}
$$

From the dynamic model of the system at trim condition shown above we knew that $\dot{p}$, $\dot{q}$ and $\dot{r}$ all equal zero. Modifying these equations for elevon control surfaces instead of aileron and elevator and after some math, we calculated the control inputs to be :

$$
\begin{bmatrix} \delta_{e_1} \\ \delta_{e_2} \\ \delta_r \end{bmatrix} = \begin{bmatrix} C_{p_{\delta_e}} & -C_{p_{\delta_e}} & C_{p_{\delta_r}} \\ C_{m_{\delta_e}} & C_{m_{\delta_e}} & 0 \\ C_{r_{\delta_e}} & -C_{r_{\delta_e}} & C_{r_{\delta_r}} \end{bmatrix}^{-1} \begin{bmatrix} -\frac{2}{\rho {V_a}^2 Sb}(\Gamma_2 qr - \Gamma_1 pq) - C_{p_0} - C_{p_\beta}\beta - \frac{C_{p_p}bp}{2V_a} - \frac{C_{p_r}br}{2V_a} \\ -\frac{2J_y}{\rho {V_a}^2 Sc}(\Gamma_6(p^2 - r^2) - \Gamma_5 pr) - C_{m_0} - C_{m_\alpha}\alpha - \frac{C_{m_q}cq}{2V_a} \\ -\frac{2}{\rho {V_a}^2 Sb}(\Gamma_1 qr - \Gamma_7 pq) - C_{r_0} - C_{r_\beta}\beta - \frac{C_{r_p}bp}{2V_a} - \frac{C_{r_r}br}{2V_a} \end{bmatrix}
\tag{3.8}
$$

The last control input is solved for by setting $\dot{u}$ equal to zero which results in the equation:

$$
\delta_t = \sqrt{\frac{2m(-rv + qw + g\sin\theta) - \rho V_a^2 S\left[C_{x_0} + C_{x_\alpha}\alpha + C_{x_q}\frac{cq}{2V_a} + C_{x_{\delta_e}}(\delta_{e1} + \delta_{e2})\right]}{\rho S_{prop}C_{prop}k_{motor}^2} + \frac{V_a^2}{k_{motor}^2}}
\tag{3.9}
$$

The output of this trim state function is the difference between the state derivatives and the

desired state derivatives V_a, R, γ, α, β and ϕ. Using the fsolve function in Matlab we got a numerical value for the trim α, β and ϕ. Using these values we calculated the state, control inputs and state derivatives at the trim condition using the same equations as above.

3.2.2 LQR Controller

Once the trim state has been calculated we need to design a controller to minimize disturbances around that condition. In order to do so we had to determine the A and B matrices of the system. The Beard and McLain textbook [3] breaks these matrices into their longitudinal and lateral components however, due to the coupled motion of the elevons we thought it would be better to use the full matrices and only have one controller.

Using the lateral and longitudinal state space models from the textbook:

$$\begin{bmatrix} \dot{u} \\ \dot{w} \\ \dot{q} \\ \dot{\theta} \\ \dot{h} \end{bmatrix} = \begin{bmatrix} X_u & X_w & X_q & -g\cos(\theta^*) & 0 \\ Z_u & Z_w & Z_q & -g\sin(\theta^*) & 0 \\ M_u & M_w & M_q & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \sin(\theta^*) & -\cos(\theta^*) & 0 & u^*\cos(\theta^*) + w^*\sin(\theta^*) & 0 \end{bmatrix} \begin{bmatrix} u \\ w \\ q \\ \theta \\ h \end{bmatrix} + \begin{bmatrix} X_{\delta_e} & X_{\delta_e} & X_{\delta_t} \\ Z_{\delta_e} & Z_{\delta_e} & 0 \\ M_{\delta_e} & M_{\delta_e} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_{e_1} \\ \delta_{e_2} \\ \delta_r \end{bmatrix}$$

$$\begin{bmatrix} \dot{v} \\ \dot{p} \\ \dot{r} \\ \dot{\phi} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} Y_v & Y_p & Y_r & g\cos(\theta^*)\cos(\phi^*) & 0 \\ L_v & L_p & L_r & 0 & 0 \\ N_v & N_p & N_r & 0 & 0 \\ 0 & 1 & \cos(\phi^*)\tan(\theta^*) & q^*\cos(\phi^*)\tan(\theta^*) - r^*\sin(\phi^*)\tan(\theta^*) & 0 \\ 0 & 0 & \cos(\phi^*)\sec(\theta^*) & p^*\cos(\phi^*)\sec(\theta^*) - r^*\sin(\phi^*)\sec(\theta^*) & 0 \end{bmatrix} \begin{bmatrix} v \\ p \\ r \\ \phi \\ \psi \end{bmatrix} + \begin{bmatrix} Y_{\delta_e} & -Y_{\delta_e} & Y_{\delta_r} \\ L_{\delta_e} & -L_{\delta_e} & L_{\delta_r} \\ N_{\delta_e} & -N_{\delta_e} & N_{\delta_r} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_{e_1} \\ \delta_{e_2} \\ \delta_r \end{bmatrix}$$

Where

$$\begin{bmatrix} A_{long} \end{bmatrix} = \begin{bmatrix} X_u & X_w & X_q & -g\cos(\theta^*) & 0 \\ Z_u & Z_w & Z_q & -g\sin(\theta^*) & 0 \\ M_u & M_w & M_q & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \sin(\theta^*) & -\cos(\theta^*) & 0 & u^*\cos(\theta^*) + w^*\sin(\theta^*) & 0 \end{bmatrix} \tag{3.10}$$

and

$$\left[\ A_{lat}\ \right] = \begin{bmatrix} Y_v & Y_p & Y_r & g\cos(\theta^*)\cos(\phi^*) & 0 \\ L_v & L_p & L_r & 0 & 0 \\ N_v & N_p & N_r & 0 & 0 \\ 0 & 1 & \cos(\phi^*)\tan(\theta^*) & q^*\cos(\phi^*)\tan(\theta^*) - r^*\sin(\phi^*)\tan(\theta^*) & 0 \\ 0 & 0 & \cos(\phi^*)\sec(\theta^*) & p^*\cos(\phi^*)\sec(\theta^*) - r^*\sin(\phi^*)\sec(\theta^*) & 0 \end{bmatrix} \tag{3.11}$$

Similarly,

$$\left[\ B_{long}\ \right] = \begin{bmatrix} X_{\delta_e} & X_{\delta_e} & X_{\delta_t} \\ Z_{\delta_e} & Z_{\delta_e} & 0 \\ M_{\delta_e} & M_{\delta_e} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{3.12}$$

and

$$\left[\ B_{lat}\ \right] = \begin{bmatrix} Y_{\delta_e} & -Y_{\delta_e} & Y_{\delta_r} \\ L_{\delta_e} & -L_{\delta_e} & L_{\delta_r} \\ N_{\delta_e} & -N_{\delta_e} & N_{\delta_r} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{3.13}$$

The constants in this state space model are defined below:

$$Y_v = \frac{\rho S b v^*}{4m{V_a}^*}(C_{Y_p}p^* + C_{Y_r}r^*) + \frac{\rho S v^*}{m}(C_{Y_0} + C_{Y_\beta}\beta^* + C_{Y_{\delta_{ev}}}(\delta^*_{ev1} - {\delta_{ev2}}^*) + C_{Y_{\delta_r}}{\delta_r}^*)$$
$$+ \frac{\rho S C_{Y_\beta}}{2m}(\sqrt{u^{*2} + w^{*2}})$$
$$Y_p = w^* + \frac{\rho {V_a}^* S b}{4m}C_{Y_p}$$

$$
\begin{aligned}
Y_r &= -u^* + \frac{\rho {V_a}^* Sb}{4m} C_{Y_r} \\
Y_{\delta_{ev}} &= \frac{\rho {V_a}^{*2} S}{2m} C_{Y_{\delta_{ev}}} \\
Y_{\delta_r} &= \frac{\rho {V_a}^{*2} S}{2m} C_{Y_{\delta_r}} \\
L_v &= \frac{\rho S b^2 v^*}{4{V_a}^*}(C_{p_p} p^* + C_{p_r} r^*) + \rho S b v^* (C_{p_0} + C_{p_\beta} * \beta^* + C_{p_{\delta_{ev}}}(\delta^*_{ev1} - {\delta_{ev2}}^*) + C_{p_{\delta_r}} {\delta_r}^*) \\
&\quad + \frac{\rho S b C_{p_\beta}}{2}(\sqrt{u^{*2} + w^{*2}}) \\
L_p &= \Gamma_1 q^* + \frac{\rho S {V_a}^* b^2}{4} C_{p_p} \\
L_r &= -\Gamma_2 q^* + \frac{\rho S {V_a}^* b^2}{4} C_{p_r} \\
L_{\delta_{ev}} &= \frac{\rho {V_a}^{*2} S b}{2} C_{p_{\delta_{ev}}} \\
L_{\delta_r} &= \frac{\rho {V_a}^{*2} S b}{2} C_{p_{\delta_r}} \\
N_v &= \frac{\rho S b^2 v^*}{4{V_a}^*}(C_{r_p} p^* + C_{r_r} r^*) + \rho S b v^* (C_{r_0} + C_{r_\beta} * \beta^* + C_{r_{\delta_{ev}}}(\delta^*_{ev1} - {\delta_{ev2}}^*) + C_{r_{\delta_r}} {\delta_r}^*) \\
&\quad + \frac{\rho S b C_{r_\beta}}{2}(\sqrt{u^{*2} + w^{*2}}) \\
N_p &= \Gamma_7 q^* + \frac{\rho V_a S b^2}{4} C_{r_p} \\
N_r &= -\Gamma_1 q^* + \frac{\rho V_a S b^2}{4} C_{r_r} \\
N_{\delta_{ev}} &= \frac{\rho {V_a}^{*2} S b}{2} C_{r_{\delta_{ev}}} \\
N_{\delta_r} &= \frac{\rho {V_a}^{*2} S b}{2} C_{r_{\delta_r}} \\
X_u &= \frac{u^* \rho S}{m}(C_{x_0} + C_{x_\alpha}\alpha^* + C_{x_{\delta_{ev}}}(\delta^*_{ev1} + \delta^*_{ev2})) - \frac{\rho S w^* C_{x_\alpha}}{2m} + \frac{\rho S c C_{x_q} u^* q^*}{4m{V_a}^*} - \frac{\rho S_{prop} C_{prop} u^*}{m} \\
X_w &= -q^* + \frac{w^* \rho S}{m}(C_{x_0} + C_{x_\alpha}\alpha^* + C_{X_{\delta_{ev}}}(\delta_{ev1} + \delta_{ev2})) + \frac{\rho S c C_{x_q} w^* q^*}{4m{V_a}^*} + \frac{\rho S C_{x_\alpha} u^*}{2m} \\
&\quad - \frac{\rho S_{prop} C_{prop} w^*}{m} \\
X_q &= -w^* + \frac{\rho {V_a}^* S C_{x_q} c}{4m} \\
X_{\delta_{ev}} &= \frac{\rho {V_a}^{*2} S C_{x_{\delta_{ev}}}}{2m} \\
X_{\delta_t} &= \frac{\rho S_{prop} C_{prop} \delta_t^*}{m}
\end{aligned}
$$

$$Z_u = q^* + \frac{u^*\rho S}{m}(C_{z_0} + C_{z_\alpha}\alpha^* + C_{z_{\delta_{ev}}}(\delta^*_{ev1} + \delta^*_{ev2})) - \frac{\rho S C_\alpha w^*}{2m} + \frac{u^*\rho S C_{z_q} c q^*}{4m{V_a}^*}$$

$$Z_w = \frac{w^*\rho S}{m}(C_{z_0} + C_{z_\alpha}\alpha^* + C_{z_{\delta_{ev}}}(\delta^*_{ev1} + \delta^*_{ev2})) + \frac{\rho S C_{z_\alpha} u^*}{2m} + \frac{\rho w^* S c C_{z_q} q^*}{4m{V_a}^*}$$

$$Z_q = u^* + \frac{\rho {V_a}^* S C_{z_q} c}{4m}$$

$$Z_{\delta_{ev}} = \frac{\rho {V_a}^{*2} S C_{z_{\delta_{ev}}}}{2m}$$

$$M_u = \frac{u^*\rho S c}{J_y}(C_{m_0} + C_{m_\alpha}\alpha^* + C_{m_{\delta_{ev}}}(\delta^*_{ev1} + \delta^*_{ev2})) - \frac{\rho S c C_{m_\alpha} w^*}{2J_y} + \frac{\rho S c^2 C_{m_q} q^* u^*}{4J_y {V_a}^*}$$

$$M_w = \frac{w^*\rho S c}{J_y}(C_{m_0} + C_{m_\alpha}\alpha^* + C_{m_{\delta_{ev}}}(\delta^*_{ev1} + \delta^*_{ev2})) + \frac{\rho S c C_{m_\alpha} u^*}{2J_y} + \frac{\rho S c^2 C_{m_q} q^* w^*}{4J_y {V_a}^*}$$

$$M_q = \frac{\rho {V_a}^* S c^2 C_{m_q}}{4J_y}$$

$$M_{\delta_{ev}} = \frac{\rho {V_a}^{*2} S c C_{m_{\delta_{ev}}}}{2J_y}$$

where * indicates the trim condition. These matrices were combined to form the 10x10 A matrix and the 10x6 B matrix:

$$\begin{bmatrix} A \end{bmatrix} = \begin{bmatrix} A_{long} & 0 \\ 0 & A_{lat} \end{bmatrix} \tag{3.14}$$

$$\begin{bmatrix} B \end{bmatrix} = \begin{bmatrix} B_{long} & 0 \\ 0 & B_{lat} \end{bmatrix} \tag{3.15}$$

In order to use the LQR controller we must also have a Q and R matrix. These matrices tell the controller how much weight to put on state control versus input control and are determined for each trim condition. For this we used a 10x10 identity matrix for Q and a 6x6 identity matrix for R. With this information we used the ”lqr” command in Matlab to produce an optimized gain matrix. We then simulated this controller using Simulink, the results of which can be seen in the following Figures:

As you can see, the simulation shows the response of the system tending to zero when there is a small disturbance as expected.

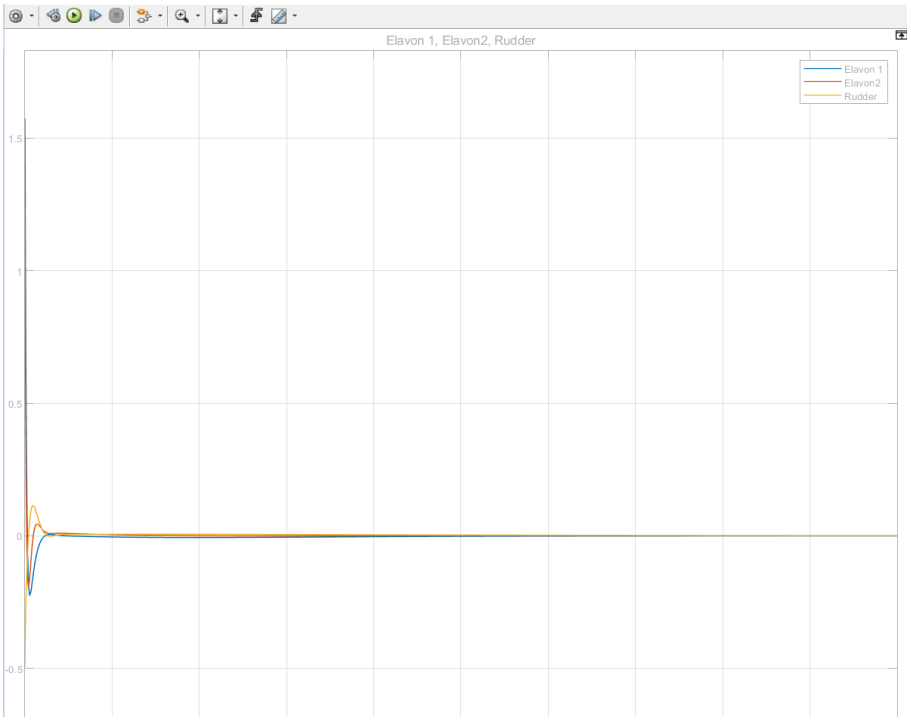


Figure 3.18: Response of control surfaces to a small disturbance

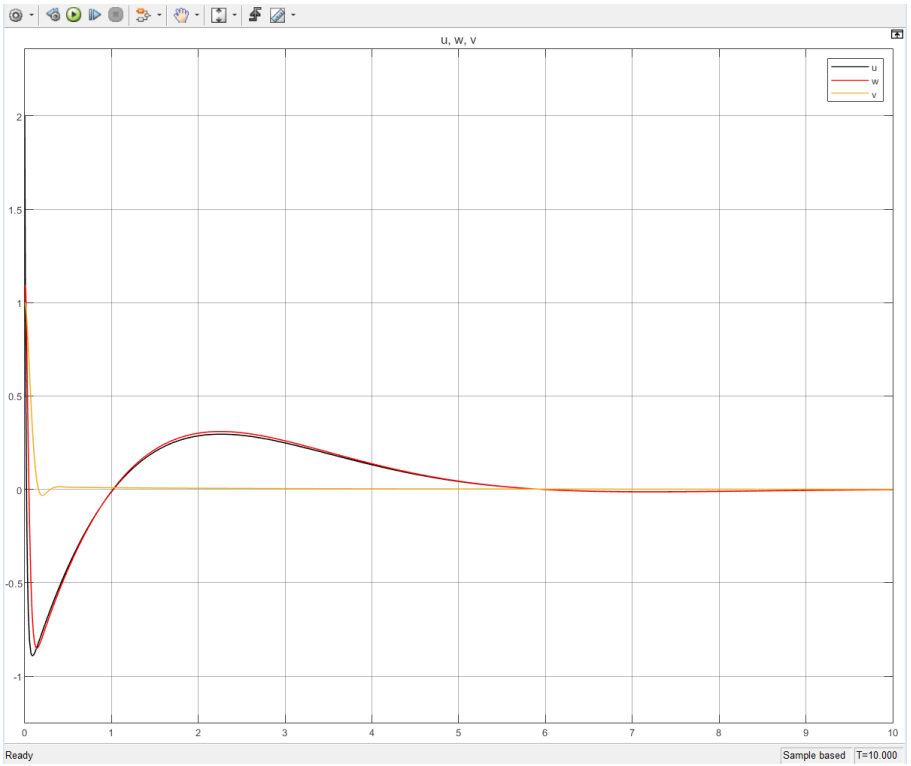


Figure 3.19: Response of velocity components to a small disturbance

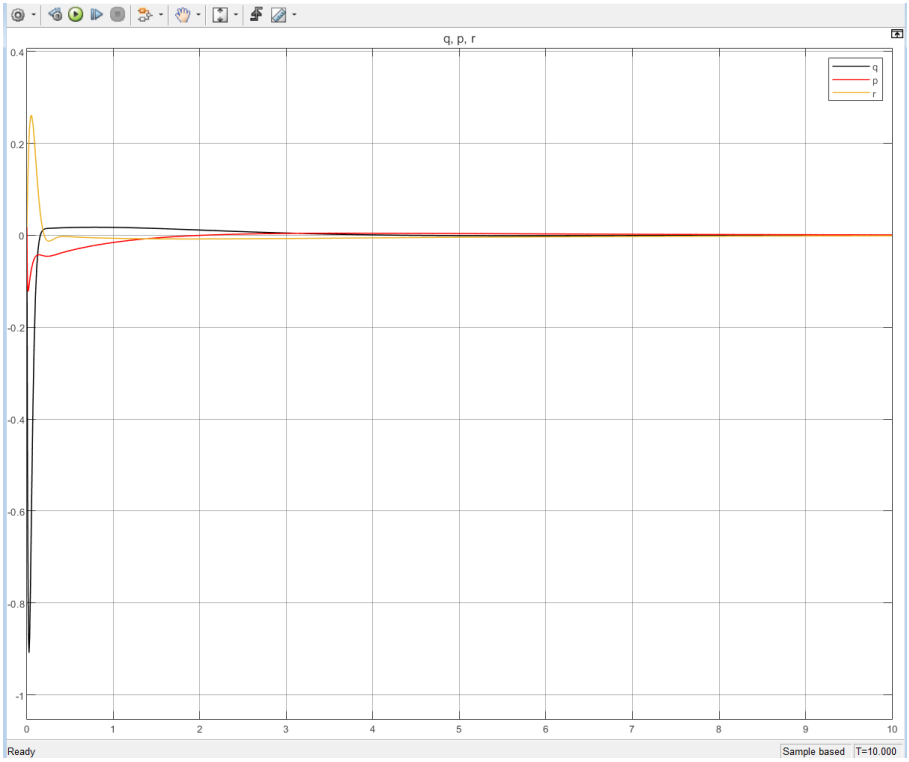


Figure 3.20: Response of angular rates to a small disturbance

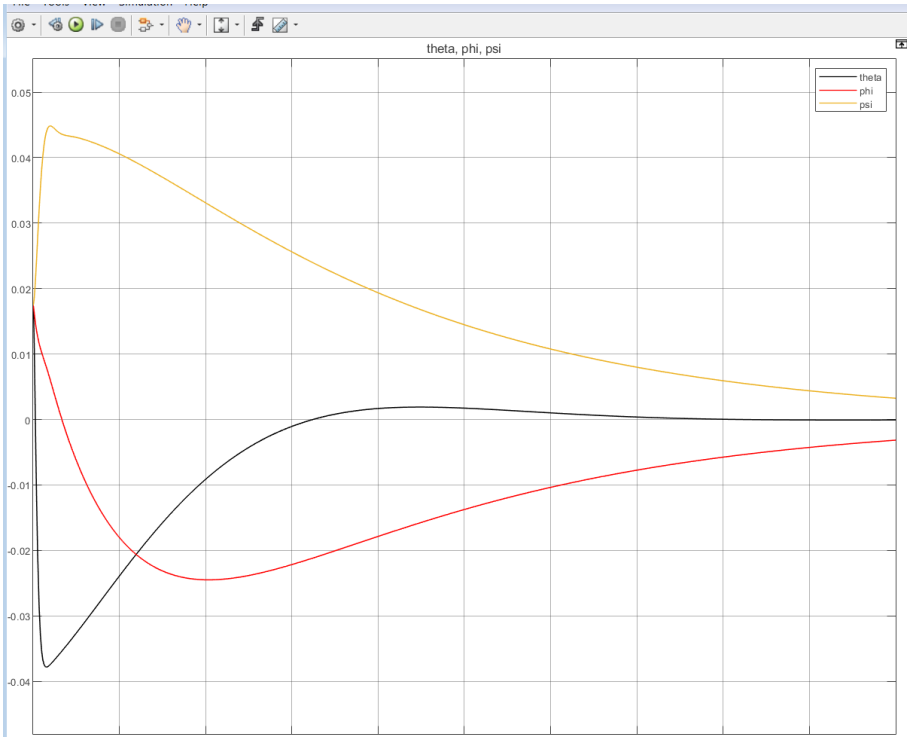


Figure 3.21: Response of Euler angles to a small disturbance

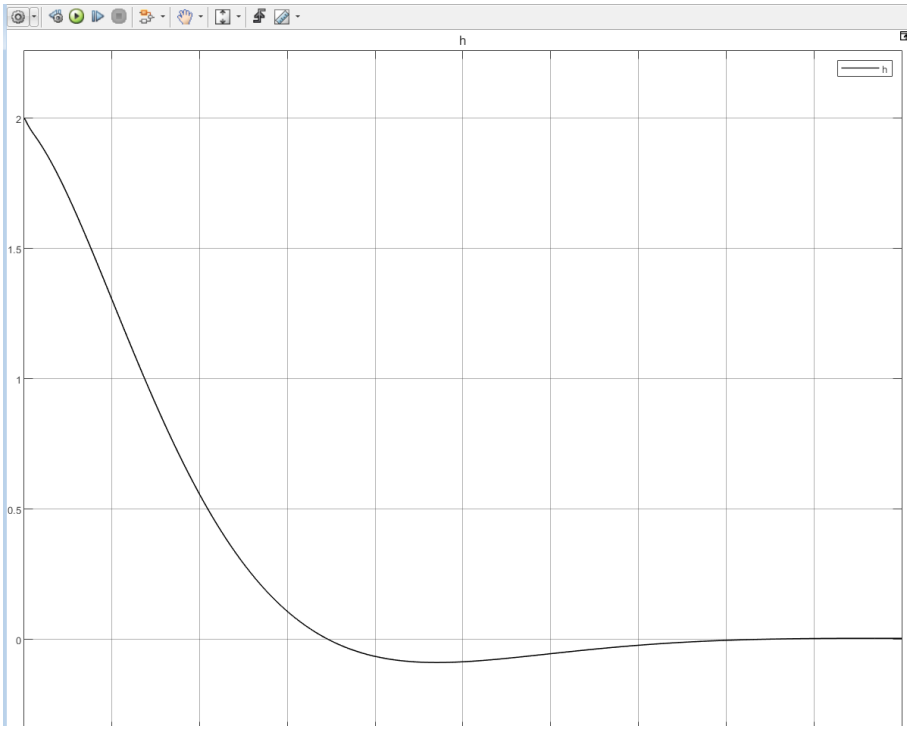


Figure 3.22: Altitude response to a small disturbance

3.3 Power & Propulsion

3.3.1 Propulsion System

In order to properly size the motors for the craft, an accurate weight budget for the aircraft must be established. Below is the current weight budget for the craft with all of it's onboard components accounted for. With the weight budget established, a set of four motors were selected that could approximately match the force of double the craft's weight. The "doubling" was done as a precaution in order to ensure we would have more than enough thrust to lift off the ground should our design change in some way. The motors that were selected were the "Tmotor MN 2212 KV920-2.0 Navigator Type" motors. According to the motor specifications, these motors output 918 g of thrust each when running at full throttle on 14.8 V and with 9.5 inch props. This results in a total expected thrust output of 3672 g, without accounting for any thrust augumentation due to ducting. This was over 800g more than the design requirement of having a capable thrust output of double the craft's mass.

Weight Budget				
Component	*Mass (g)*			
Frame Foam Esitmate	500			
Jetson TX2 on Carrier Board	150			
Realsense D435	72			
Picam	3			
Max Amps 3250 mah 4s Lipo (14.8)	327			
PixHawk	10			
Servos (*5)	60			
Lumenier BLHeli_S 35A 4-in-1	14			
Pozyx tag	12			
Motors	260			
	1396	*g*m*2 = Thrust*	27.3616	(N)
		Thrust	2792	(g)
		100% Throttle thrust output	3672	(g)
		(No added efficency for the fan)		

Figure 3.23: Weight Budget for the Aircraft

Using equations (4) and (5), the propulsion systems of a regular quadrotor and the Turac were analyzed for a fixed mission of rising up to an altitude of 3.25 m over a 4 second period, hovering at that altitude for 60 seconds, and then descending back down at a rate of 60 seconds. This analysis assumes a FM of 0.7 for both systems, a thrust distribution of 15%, 15%, and 70% (front two tilt rotors, ducted coaxial fan) for the Turac propulsion system, and even thrust distribution of 25 % per motor for the quadrotor propulsion system. The results of this analysis are represented the figure below. The current consumed over time can be found by taking the area under the curve. According to this analysis, a traditional quadrotor's propulsion system will consume approximately six times as much power as the Turac's propulsion system to complete the same mission. After this conducting this analysis, it was decided that our aircraft would have a propulsion that closely mimic's the Turac's, with a ducted coaxial fan and two tilt rotors mounted in the front of the aircraft.

3.3.2 Electrical System Overview

Below is an overview of the aircraft's electrical system.

The voltage coming from the battery will be 14.8 volts, which will be stepped down to 12V for the Jetson TX-2 and 5V for the Pixhawk with the two dc-dc converters that exist on the 4-way Lumenier ESC. The ESC will output the PWM signal requested by the Pixhawk to the motors

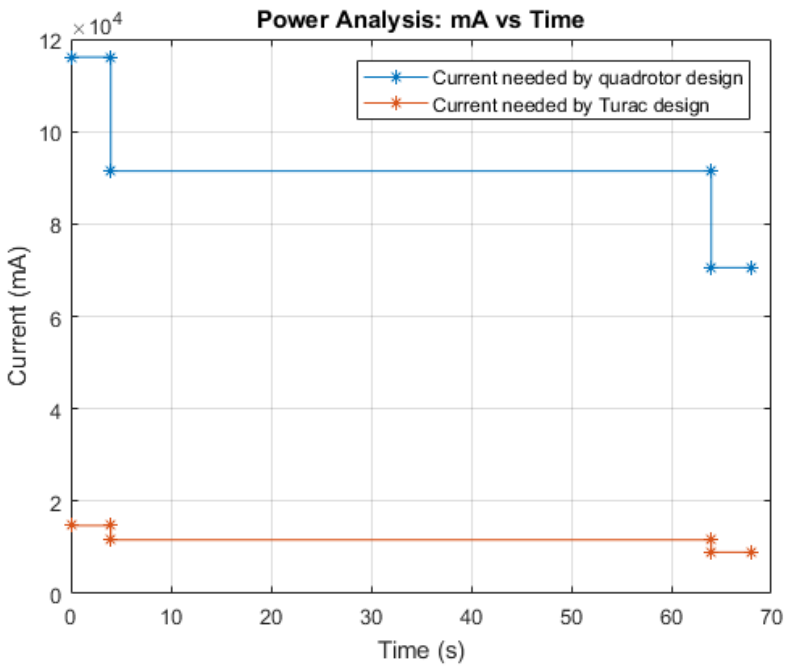


Figure 3.24: Power Analysis of Quadrotor and Turac Propulsion Systems

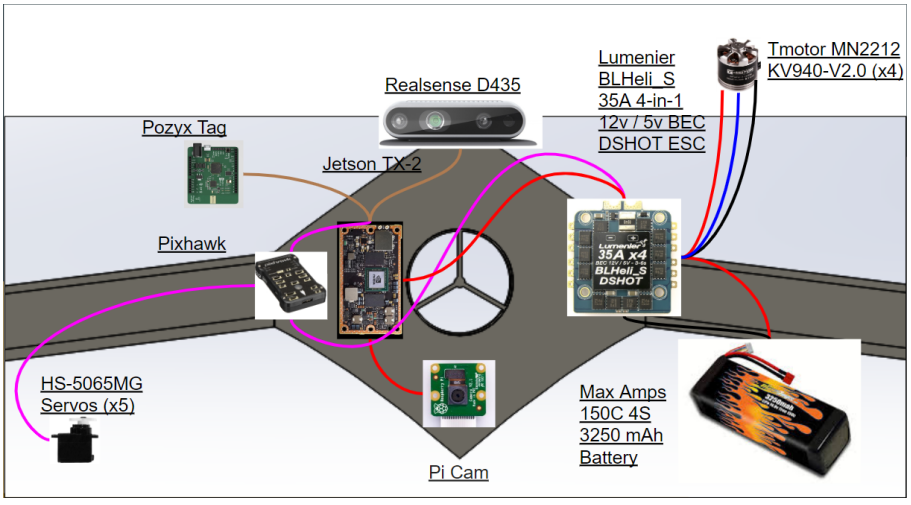


Figure 3.25: Electrical System Overview

as a part of the motor control system. The Pixhawk will also output requests to the Servos as part of the control system. Moreover, all image & sensor data from the Pi Cam and Realsense depth camera will be sent to the Jetson TX-2 with USB connections. Finally, the Pozyx tag will be connected to the Jetson TX-2 as a means of localizing indoors. All connections adhere to component's specified voltage, current and power limits.

Originally the Jetson TX-2 was going to be placed on a lighter and more compact J-120 carrier board instead of the heavier development board that the Jetson TX-2 came with. However, the carrier board was abandoned for two reasons. First, the overall size and weight nearly tripled from the original design, thus allowing for us to use the heavier development board on-board our aircraft. Secondly, while we were able to correctly power up the carrier board using a 4s and the 5 V dc-dc converter on-board the ESC, we were unable to properly flash the OS to it as we could not get the carrier board to enter "force recovery" mode.

3.3.3 DC-DC Converter Design

After conducting a battery survey, it was determined that higher voltage LiPo batteries with more cells are less energy dense that lower voltage LiPo batteries with less cells. As shown below in the battery survey, a 2s battery (7.4 Volts) weighing 250g, stores almost twice the energy a 4s(14.8) battery weighing 250g does.

Knowing this, a dc-dc converter with high efficiency could be used to step up the voltage of a more energy dense 2s battery, to the 14.8 V that the motors need to run off of. As long as the converter was light enough, this would increase the range of our craft. After conducting a search for a dc-dc converter that fit our system specifications, none where found to be satisfactory, as shown below. A red highlighted cell indicates an non-matching specification that rules the device out for our system. A yellow highlighted cell indicates a imperfect specification that may rule the device out for our system.

After failing to find a DC-DC converter that fits all of the specifications of the system,

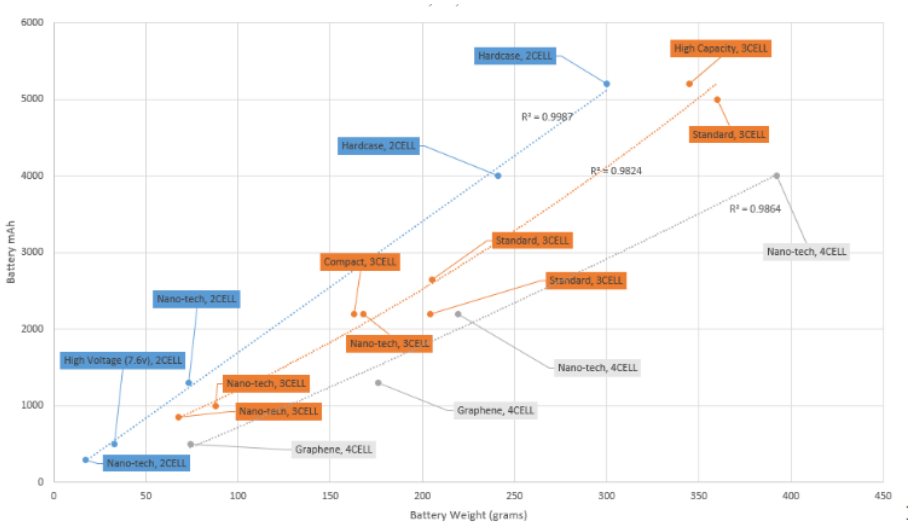


Figure 3.26: Survey of 2s, 3s and 4s Batteries, Weight [g] vs. Energy [mAh]

Name	Max Output Voltage	Min Input Voltage	Max Current Output	Efficency	Weight	Price
DF Robot DC-DC 150W	24 V	8 V	6 A	0.9	280g	$39.00
Mean Well DDR-60G-24	36 Vdc	9 V	2.5 A	?	220 g	~$30.00
VPT DVFL2800S Series	28V	16 V	20 A	0.81	86g	"Call Sales Rep"
TC230 Series	28V	10 V	?	0.9	318g	$891.00
CUI PQA50-T	12 V	18V	10 A	0.9	57g	$108.68

Figure 3.27: DC-DC Converter Survey

the design of a custom dc-dc converter was started. The following things so far have been considered in the design: A ”Soft Start” to account for inductor current limit, Capacitor ESR, realistic switch and diode losses. Below is the design written in PSPICE code.

Below are the simulated waveforms from the above PSPICE code. The output voltage is in red while the input voltage is in green.

According to the waveform, the output voltage for this design is the intended 14.8 V, with a very small voltage ripple of less than .5 V. While this analysis validates the design of the DC-DC converter, there was simply not enough time to build the converter and incorporate it into our system. Had there been more time or more team members with an electrical engineering background, the system’s overall efficiency could have been improved by introducing this DC-DC converter immediately after the battery. Should the DC-DC converter ever be built in the future it will be important to use an inductor with a high enough saturation limit, that is also

```
EVTOL BOOST DC-DC Converter Draft 2 Engineer: Tyler Weiss
V 1 0 PWL(0s, 0V, 0.5ms, 7.4V) ; Input voltage from Battery
L 1 20 .9u IC = 0; Inductor
rL 20 2  65m; Inductor ESR
S 2 0 10 0 ZW; Switch
.MODEL ZW VSWITCH(RON=0.05u ROFF = 1MEG); Switch
D 2 3 DI; Diode
.MODEL DI D(IS=6.02u RS=4.20m BV = 40.0 IBV=600u; Diode
+ CJO=2.21n M=0.333 N=1.04 TT=50.4n); Diode
C1 3 0  103.0u IC=0; Two Caps in Parallel to limit voltage ripple.
C2 3 0 103.0u IC=0
R 3 0 1.44; Load Resistance
VD 10 0 PULSE(-1 10 0 1n 1n 9.7u 14.8u); Duty Ratio: 9.7/14.8 = .66
.PROBE
.TRAN 2m 2m 0 600n UIC
.END
```

Figure 3.28: DC-DC PSPICE Code

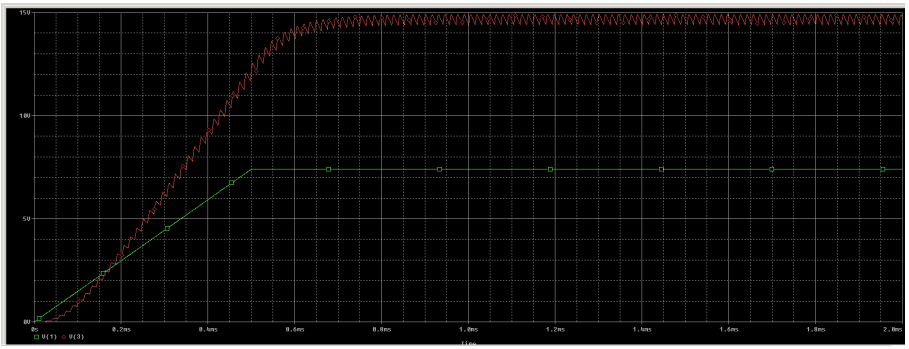


Figure 3.29: DC-DC Output Waveforms in PSPICE

light enough for our application. Capacitors with a low ESR will also need to be considered when order components.

3.4 Autonomy & Control

The goal of this eVTOL vehicle is to autonomously navigate to an unknown target location and land at that location. In order to achieve this goal separate autonomous tasks had to be completed simultaneously. The first challenge was to implement a target identification system. Second was to sense and avoid obstacles in the vehicles surroundings. With these two autonomous systems working in parallel the vehicle would be able to complete its tasks.

Figure 3.30: Initial Testing of Circle Tracking Software

3.4.1 Target Detection

The chosen target marker was red and white concentric circles. This proved to be a difficult target to detect. To detect the circles we used a camera and several image processing techniques. First the image went through a black and white filter. This allowed for less noise in the image since the target is known to be both white and red which appear white and dark gray once filtered. The next step was to put the image through a gaussian blur. This softened the edges created in the background of the image. Finally once the image was ready we used the OpenCV method of Hough Transform to detect the circles. This method identifies edges and extends a line perpendicularly out and makes each point on that line as the potential center of a circle. once it has done this for each edge it will decide the likelihood that it has actually found the center of a circle and return the center and radius. Since our target contains several circles it makes the target easier to find. an image from the initial testing video can be seen in Figure 3.30.

3.4.2 Autonomous Obstacle Avoidance

We needed to navigate two environments, the first of which is for the indoor obstacle avoidance tests, and the second is the recreational fields at Worcester Polytechnic Institute where we will complete test flights. Since our software is designed around ROS the integration of different libraries eases the development of the system. For map storage Octomap is used because it is a strong 3d mapping software that integrates easily with ROS. When mapping indoors the vehicle will always be in hover mode. This allows us to better map the environment because of the lower speeds and higher maneuverability. When hovering the vehicle can rotate and with use of the Pozyx localization, point cloud data from the Realsense camera, and inertial measurements from the IMU patch together a map of its surroundings. With this data the vehicle can then make decisions on which direction to travel and where to search for a target. When navigating outside and using the flying wing mode the task becomes more difficult. Obstacle avoidance of small flying wing UAV is a topic that has rarely been researched publicly. In a paper this topic was researched heavily with a similar aircraft to out Turac based model. In this research a similar stereo vision system was used. This stereo system was implemented to work in a push broom method. This method greatly increased computational times by only looking for obstacles a fixed distance ahead of the vehicle. This allowed the vehicle to focus ahead on the obstacles that it would be able to avoid rather than the ones that were nearby and it wouldn't have time to react. Since this is such a computationally expensive task mapping the environment is not a priority. In this mode the first priority is to not crash. The second priority is to find the target. The vehicle will fly in a zamboni like pattern because of the inability to make tight turns similar to a zamboni. while making this loop the downward facing camera will constantly be searching for the target and if it is located the position is marked and will be circled back to.

Chapter 4

System Development

4.1 Aerodynamics & Structures

4.1.1 Aerodynamics development

To develop the aerodynamics for the models used in this project, we referenced the research and analyses conducted in the design phase. For example, when creating the initial models used to test the three different airfoils (S1223, E423, NACA 34112) we referenced previous research on high lift airfoils and cross referenced their performance numbers given by online resources. After testing those models in the wind tunnel, we went back and refined the models. The refining included adding winglets to the models and ensuring all dimensions matched between the models and that the only changing variable was the airfoil shape. These models were then tested in the wind tunnel and the results are the graphs seen in section 3.1.

These results were then taken and built into an original test model in SolidWorks with all components inside. At this point, we realized that more space was needed to fit all of the electrical and mechanical parts of the vehicle's systems. We concluded that to make it work, a thicker airfoil for the central body was required and the main body needed to be reshaped into a more rectangular form. The airfoil used for the central body for the "New Model" design is a modified version of the NREL S822. A 3D printed model of the modified design was then tested

in the wind tunnel as well and the results of that testing can be seen in the graphs in section 3.1 where the data is labeled ”New Model” with a yellow line. The New Model performed similarly to the TURAC-based model, and even better in some ways, with reduced drag and marginally higher lift values at lower angles of attack. Once we confirmed the performance of this new design, we started building foam core mock-ups of the models for testing.

The wings were cut using the hot-wire cutting technique where 26 gauge nichrome wire is strung on a u-shaped bow and a current is run through the wire. This heats the wire up to a point where it can effortlessly cut through foam core and can be used to shape the foam blocks into aircraft parts including wings, winglets, body panels, etc. We utilized this technique to construct the wings of our glide test model and to roughly shape the body of that same model.

Once this full scale glide test model was fully constructed, testing commenced. Testing of the glide test model consisted of having a team member hold the model above their head and throw it in such a way that it would leave their hands at a speed higher the stall speed. However, this proved too difficult a test method as the model was large and awkward to hand launch. Thus, we scaled the model down to a half-scale size and the glide testing continued. The half-size model was much easier to throw and glided over short distances. The main issue with the glide model during testing, was its tendency to roll out of a stable glide. This can be seen toward the end of one of the videos we took during testing. To fix this unstable roll, more aerodynamic design was conducted and a new model with delta wing shape was designed which, according to XFLR5 was stable in all dynamic stability modes, especially the dutch roll which was the main one of concern before.

4.1.2 Structure development

We used two methods for the construction of our vehicle: foam board wrapping and hot wire slicing. This was due primarily to our familiarity with the methods. However, the models constructed with these methods remained durable, functional, and easily modifiable for early testing and development. We built three distinct models during our initial testing phase: a full

scale glide test model, a half scale glide test mock-up, and a hover test mock-up.

The full scale glide test model was a combination of hot wire cut wings and a foam board body with sanded insulation foam to create the tapers that connect the body with the wings as seen in 4.1. We found during testing that we were unable to accelerate the vehicle past its' stall velocity and were recommended to downsize.

Due to the overall size of the half scale model, we decided against a hollow body and instead made it entirely of sanded insulating foam. The wings were constructed using the same method as the full scale glide model. While this allowed us to accelerate the vehicle past its' stall velocity, we found that the vehicle likely had an unstable dutch roll mode, which countered the dynamic stability results we found in XFLR5. However, one of the predominant reasons we can doubt this due to the challenge of stably launching the vehicle. We resorted to a hand launching the vehicle over our heads which often resulted in a sharp yawing motion.

Our hover test model included many of the core components of the final model but without the wings. We 3D printed a fan duct and front motor boom mounts and created a foam board shell to encapsulate our electronics. This allowed for us to get a better idea of how much space we had to work with while also providing an easily reparable vehicle.

Figure 4.1: Image of the Full Scale Glide Model After Many Tests

4.1.3 Glide Test

Once we were able verify and improve the aerodynamics of our 3 initial wind tunnel test models, we manufactured a full scale test model. We conducted glide tests with the full scale model as another verification method to prove previous analysis such as stability and our aerodynamic conclusions. However, the first glide test of the full scale model was inconclusive as the model crashed without providing any evidence of a stable glide. Our advisor suggested that the vehicle likely wasn't reaching it's stall speed. To resolve this, our team constructed a half scale model which achieved semi-stable flight after shifting the center of gravity several times. This model can be seen in figure 4.2. Video analysis showed that the vehicle likely had a stable phugoid and short period mode but lacked lateral stability as it had a tendency to roll uncontrollably. This can be resolved through the use of ailerons and so we determined that the test was satisfactory.

Figure 4.2: Image of the Half Scale Glide Model

4.2 Power & Propulsion

The propulsion was successfully implemented onto the hover test model with one minor adjustment. Due to availability of genuine parts, the prop size and ducted fan size were both reduced to 9 inches. This change was made after the set of 9.5 inch props that were ordered were not compatible with the motors. This was thought to be a problem because the vehicle weight already exceeded our original expectations and a smaller prop would produce less thrust. A single motor was tested on a thrust test stand to measure exact thrust capabilities with a 9 inch prop. The results of the test can be seen below in 4.3. In addition, reducing the fan size made the ducted fan easier to 3D print and reduced the overall size of the vehicle.

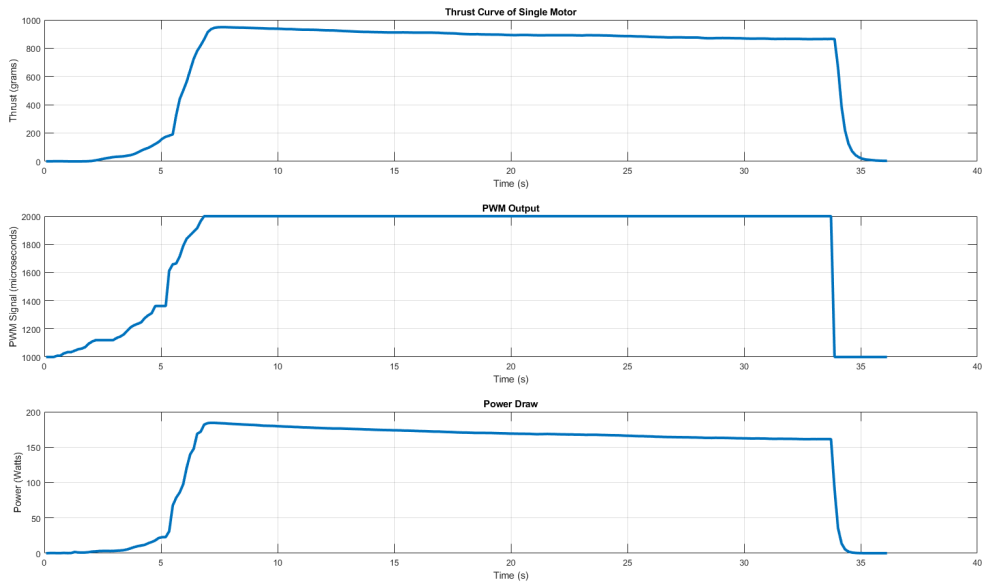


Figure 4.3: The thrust and power curves obtained from motor testing.

During early testing phases, pitch and roll problems affected the craft's ability to hover. Originally, the issues were thought to be caused by incorrect placement of the center of gravity on the test models such that the motors on the front booms were causing to great of a pitching moment. These issues were corrected by enabling quadplane abilities in the Pixhawk firmware and selecting the tricopter setup, also known as the "Nimbus" frame class. With this setup, a

stable hover was finally achieved and a push to implement tilt rotors was attempted. The tilt rotor servos were powered by connecting the 5-volt output from the electronic speed controller to the servo rail of the Pixhawk. The servos we used were the RDS3115MG bi-axial digital servos that allow up to a 180 degree motion. Forward boom mounts were made for these servos. The tilt servos were then set up for the Pixhawk and performed as expected except for a propeller clearance issue. The tilting of the motors caused the props on the forward motors to clip the booms. This was resolved by limiting the allowable deflection angle of the servos to sixteen degrees from center when rotating towards the booms. When we first attempted using the deflection angle parameter to solve the clearance issue, we found the parameter made an indeterminable difference in the deflection angle. This was caused by inaccurately centered servos, which can be caused by mounting the servos while they are powered off (they won't hold a centered position when powered off and through a parameter that limited the range of motion of the servo. With this issue addressed, we were able to achieve a stable hover. The only remaining issue was instability in the PID tuning for yaw control, which we attempted to resolve using a recently released (and relatively unproven) QAUTOTUNE feature that automatically sets PID values for each axes of motion. When we updated our firmware to the latest beta version of Arduplane, we gained access to this feature but lost the ability to spin the motors. This issue went unresolved. We ultimately returned to the 3.9.6 firmware. With this setup, a successful hover was completed and can be seen in Appendix A.

With a successful hover completed, we focused on transitioning to forward flight from hover. Unfortunately we were unable to achieve a succesful transition to forward flight before the submission of this paper. This was due to an unresolved issue with the nimbus firmware. Our team located setup documentation for selecting a switch on the transmitter to change the vehicle's flight mode. The documentation stated that once the vehicle was switched from QStabilize to any other Arduplane mode (stabilize), the vehicle would begin to transition by slowly rotating the tilt rotors forward and decelerating the fan as the vehicle's wings begin to generate lift. Instead, we found that the vehicle would freeze it's throttle values and fail to fully rotate the

tilt rotors. We attempted to resolve this by lowering the vehicle's cruise speed parameter and attempted to switch to modes other than stabilize but failed to find a successful result. The only different result came from switching from Quadplane to Manual which immediately shutoff the coaxial motors and quickly rotated the tilt rotors forward. This would result in our vehicle stalling. For these reason, we felt that attempting a transition in the air would be unsafe.

As for the electrical system, all components were successfully connected during a bench test. However, because the complete aircraft was never fully constructed, it is impossible to know how the electrical system may have behaved during various loads it would have experienced during a flight test.

4.3 Autonomy & Control

Figure 4.4 below summarizes the proposed system design for the hardware and software components necessary to carry out autonomous flight with obstacle avoidance and circle detection.

4.3.1 Jetson TX2 Setup

To properly set up our Jetson TX2 we first downloaded the desired version of Jetpack operating system onto a linux machine. After trying Jetpack versions 3.1, 3.2.1, and 3.3 we concluded that the Jetpack 3.2.1 would be the best option as it was the operating system used in the documentation we found for setting up a Realsense D435 Camera. Once downloaded we extracted the file and made it executable by running chmod +x <file name >. Running the file would open up a Jetpack installer window. First it will explain that you are using the jetpack installer to flash a Jetson Development kit with the OS you have downloaded the next button at the bottom of the window can be clicked. Next the installer needs the installation and download directories chosen and a choice needs to be made wether to enable or disable data collection. It is good practice to change the instalation directory to its own folder in the home directory, but it can be left in the default folder. Allowing or not allowing data collection does not make a

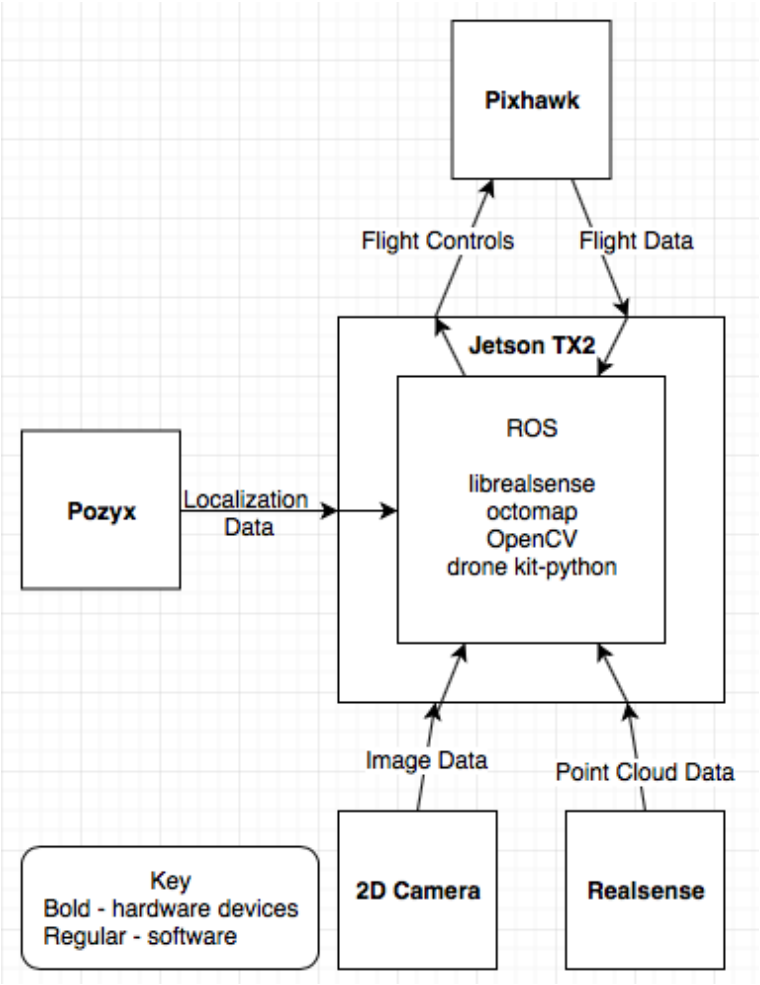


Figure 4.4: Block Diagram of Component Interactions

difference in the outcome of the instalation. The next step will be to selsect your development enviroment, we used a Jetson TX2. Then a pop-up window will open asking for the password for the host machine you are using. The next window will be the Jetpack Components Manager. Select which components you would like to install. We chose to install all the components and then selected next in the bottom right corner. Next the Terms and Conditions window popped up and Accept All was checked off in the top left followed by the accept button in the bottom right. Now the Jetpack components will begin downloading and preparing for installation. In this step we found that an error would occur while attempting to download OpenCV (see later on how to setup OpenCV). To fix this we changed the action from install to no action in the Jetpack Components Manager window and continued the install. Once the install was complete

the network layout needed to be selected. Two images of the layout are presented and for our application the second option was selected. Next the network interface selection is requested to which we chose the default setting. Now a Post Instalation window will open giving instructuond to attach the Jetson device through its USB and to put the device into force recivery mode so the Ubuntu Operating system can be flashed. After that is setup and run the devide will be ready with Ubuntu Operating system.

4.3.2 USB Camera and OpenCV

In order to setup the USB camera for circle detection we had to add openCV to the Jetson because we changed the action to no action in the previous step. To do this we went to github jetson hacks buildOpenCVTX2 and in the home directory ran the command
'git clone https://github.com/jetsonhacks/buildOpenCVTX2' to get the files from the repository onto the Jetson. Next we followed the instructions from the README.md of the github page to build and intall OpenCV those directions are as follows:

```
$ cd OpenCVTX2
$ ./buildOpenCV.sh
$ cd opencv/build
$ make
$ cd opencv/build
$ sudo make install
```

To test that OpenCV installed correctly we ran the 'test.py' and 'video_capture.py' scripts using the command 'Python' followed by the name of the file. The first listed script will open the video stream while the second script is the actual circle detection script.

4.3.3 Realsense Stereo Camera

To implement obstacle avoidance an Intel Realsense Depth camera was selected to measure distance from the vehicle to any potentially obstacles. However, the Realsense is not natively compatible with the Jetson computer. Since these are natural component choices for this sort of project there are several examples of these two components working together. This can be achieved by patching and rebuilding the Jetson kernel before building the realsense library [9].

On this page there are two options to rebuild the Kernel. The first is titled ”Rebuilding the kernel” and the second is titled ”A nasty little alternative” [9]. The ”Rebuilding the kernel method requires running multiple patch scripts to patch certain drivers and header files within the kernel. When we tried this method, we found that many of the drivers and header files that the patch script was trying to patch, did not even exist in our Jetson’s kernel in the first place. Once this was discovered, we attempted to download these drivers from an Ubuntu 16.04 kernel that was posted on Github, place each driver and header file in it’s proper directory, and then run the patch script. While some patches were successful with this method, some patches also failed and ultimately we could not successfully rebuild the Jetson’s Kernel using the ”Rebuilding the kernel” method.

After failing to rebuild the kernel using the ”Rebuilding the kernel Method”, we then moved on to trying the second method titled ”A nasty little alternative.” Upon originally trying this method, the installation of the Realsense library would fail and return an error relating to the ”Cuda”. The Cuda is parallel computing platform developed by Nvidia that speeds up computing on-board the Jetson by using GPU cores in parallel [7]. Ideally the Realsense library would be built using the Cuda, but we were unable to resolve this error without disabling the use of the Cuda. To disable the use of the Cuda during the building of the Realsense Library and successful interface the Jetson and Realsense Stereo Camera, change the flag ”-DBUILD_WITH_CUDA=true” on line 116 of ”installLibrealsense.sh” to ”-DBUILD_WITH_CUDA=false”.

4.3.4 Pixhawk

While the Jetson serves as the main onboard computer, the pixhawk serves as the flight controller. The Pixhawk must be setup to communicate with the Jetson. The first step is to setup the interface with the Jetson via ROS and MAVROS/MAVLINK. To do this, go to https://github.com/jetsonhacks/installROSTX2 and clone to the home directory. Then run the following commands:

```
$ cd ~/installROSTX2
$ ./installROS.sh -p ros-kinect-ros-base
$ sudo apt-get install ros-kintic-desktop
$ sudo apt-get install ros-kinetic-desktop-full
$ ./setupCatkinWorkspace.sh
$ cd ../catkin_ws
$ sudo apt-get install ros-kinetic-mav
$ sudo apt-get install python-pip
$ sudo pip install dronekit
```

When connecting to pixhawk through USB you need to include these lines in your code

```
connect_string = 'dev/ttyACM0'
vehicle = connect(connection_string, wait_ready=True)
```

Next, to finish setting up MAVROS, follow the instructions at
https://dev.px4.io/en/ros/mavros_installation.html
Run the following commands:

```
$ sudo apt-get install python-catkin-tools
$ cd ~/catkin_ws
$ catkin init
$ wstool init src
```

```
$ rosinstall_generator --rosdistro kinetic
mavlink | tee /tmp/mavros.rosinstall
$ rosinstall_generator -- upstream
mavros | tee -a/tmp/mavros.rosinstall
$ rosinstall_generator --upstream-development
mavros | tee -a/tmp/mavros.rosinstall
$ wstool mere -t src /tmp/mavros.rosinstall
```

respond yes to prompt

```
$ wstool update -t src -j4
$ Rosdep update
$ rosdep install --from-paths src --ignore-src-y
$ sudo ./src/mavros/mavros/scripts
/install_geographiclib_datasets.sh
$ cd ~/catkin_ws
$ rm -rf dronekit
$ catkin_make_isolate
```

Next to test if the Pixhawk is connected connected correctly, try recieving raw imu data from the pixhawk as follows:

```
$ cd /opt/ros/kinect/share/mavros/launch
$ roslaunch mavros apm.launch fcu_url:=/dev/ttyTHS2
```

if using USB to connect use '/dev/ttyACM0/ ' in separate terminal tab while above command is running the next step can be started in a new window.

```
$ cd
$ rostopic echo/mavros/imu/data_raw
```

Following the above commonds should result in successful integration of the Jetson and Pixhawk via MAVROS/MAVLINK. The results of these tests are discussed in the following section.

Chapter 5

Results

5.1 Aerodynamics and Structures

One of the constants of our project was the aerodynamics of the vehicle. Through wind tunnel, glide tests, and software analysis, we were able to validate the theoretical performance of our vehicle when flying as a traditional fixed wing. Unfortunately, we were unable to test the performance of our wing design on our full scale vehicle.

While much of the aerodynamic analysis was completed early in the project, the structure of the vehicle remained fluid throughout testing due to continually changing requirements and the discovery of limitations in our design. The extent of the vehicle at the submission of this paper was a bare bones internal structure that supported all four motors and the necessary hardware. It was primarily constructed from soft wood spars, streamlined aluminum spars, and printed components. Although rudimentary, we had designed the vehicle to be as modular as possible to allow for a simplified 'add-on' construction method as we progressed through testing. We successfully tested the air frame in a hover and had begun testing forward flight. However, a successful transition was not attempted.

5.2 Power and Propulsion

System hover tests of the skeleton frame of the final design and all electronic components except Jetson, Realsense, and USB Camera were eventually successful. Successful hover was achieved with slightly under 50% thrust. This indicates that the full system would have had more than enough thrust to hover and maneuver in VTOL mode. Achieving hover at roughly 50% thrust is inline with specifications for quadrotors and other electric hover vehicles. Since the hover configurations is the highest required thrust configuration, we can conclude that the vehicle would have had enough thrust in fixed wing mode.

5.3 Controls and Autonomy

The implementation of the controls and autonomy software and components met with varying levels of success. We were able to successfully setup the Jetson TX2 and individually interface it with each of the other components. However, these different components were never working at the same time onboard the Jetson and as a result were never tested for performance while running simultaneously. On the whole this subsystem, which comprised the primary work of the software team in the final term, is considered partially successful. Individual components worked to varying degrees and appeared they would have worked together.

5.3.1 USB Camera and Circle Tracking

We were able to successfully interface a USB camera with the jetson and run the circle detection scripts developed earlier in the term. Performance of the circle detection algorithm onboard the Jetson was similar to the performance on a desktop. Using only the optical USB camera, the algorithim successfully indentified circular objects in the field of view. It was able to do this in real time and with circles of varying size. It was also tested with moving circles, where the circular object was waved back and forth in front of the camera and move across the field of view from one end to the other. The results of these tests indicate that the USB camera and

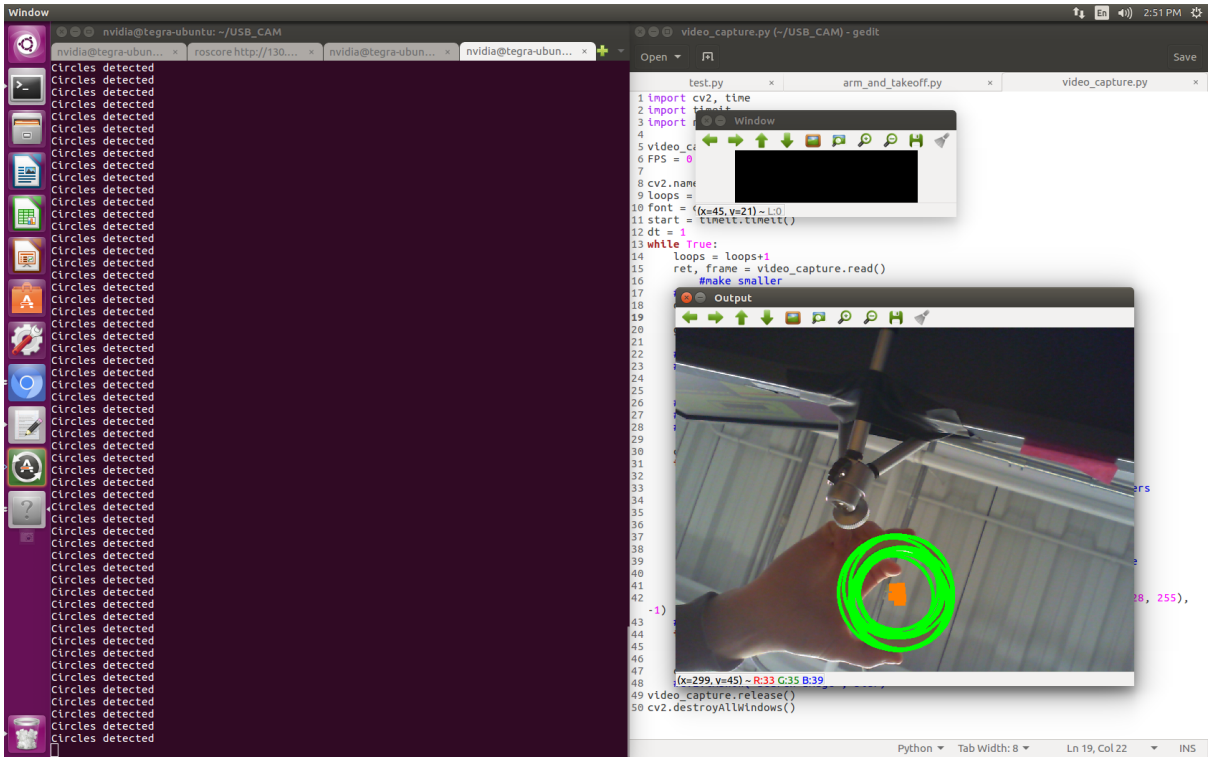


Figure 5.1: Circle Detection Onboard Jetson

circle tracking algorithm would have been sufficient to identify circular targets on the ground.

5.3.2 Realsense StereoCamera

The integration of the Realsense with the Jetson was perhaps the single most time consuming part of the software system development. As the two components are not naively designed to work together, doing so was difficult and not at all straight forward. Ultimately we were successful and were able to use run the realsense from the Jetson. We were able to get optical and stereoscoptic distance readings from the Realsense to the Jetson and view this in the Realsense user interface enviornment.

We were never able to setupt ROS in such a way that it could read the data from the realsense and as a result were never able to write or implmenet obstacle avoidance algorithms onboard the Jetson.

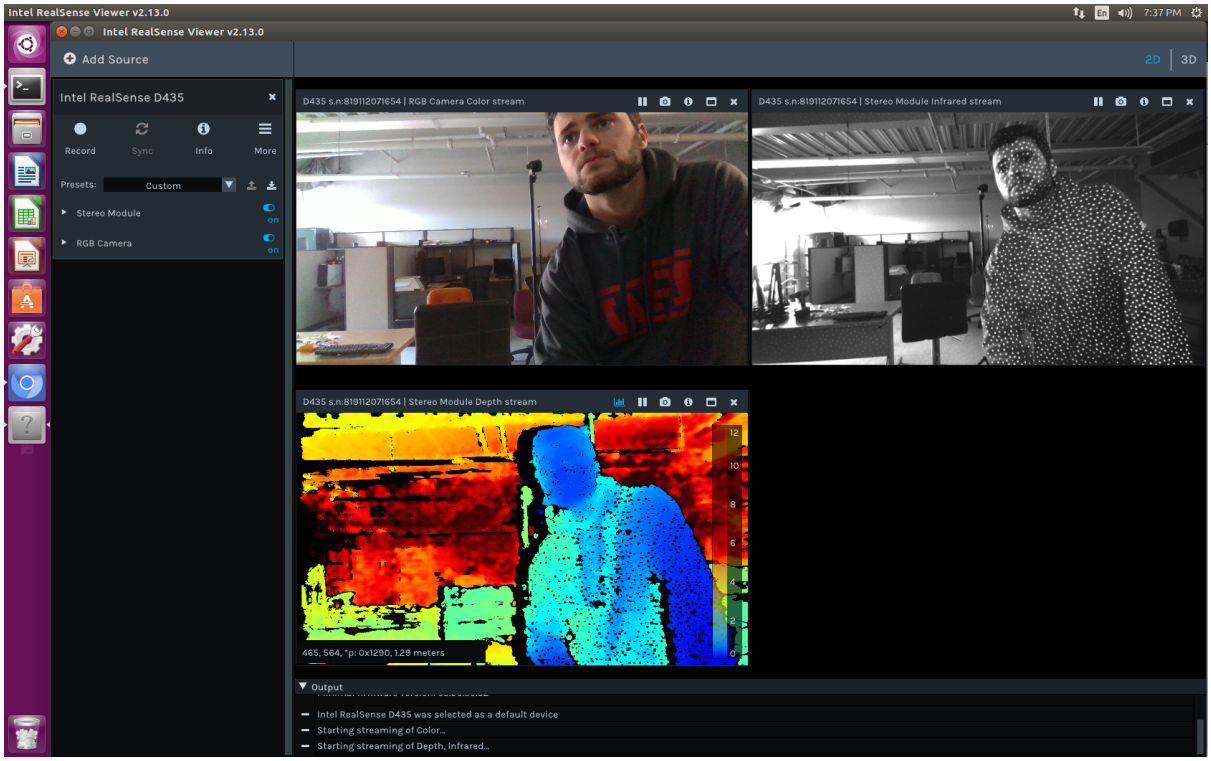


Figure 5.2: RealSense User Interface Onboard Jetson

5.3.3 Pixhawk

We were able to successfully interface the Pixhawk with the Jetson. We were able to read data from the pixhawk to the Jetson such as imu data. We also successfuly conducted bench tests in which the Jetson and Pixhawk were interfaced and then the pixhawk was connected to motors with out propellers. Using Dronekit, were able send a command to the pixhawk to take off and ascend to an altitude to of 20 meters. Upon sending the command, the motors connected to the Pixhawk would start spinning and the pixhawk would return the craft's current altitude to the Jetson. This indicates that these components would have worked on board. In a similar fashion, we were able to turn a servo connected to the pixhawk using a Dronekit script. This shows that we would be able to command servos on board our aircraft in order to transitions from hover to fixed wing flight as well as move control surfaces.

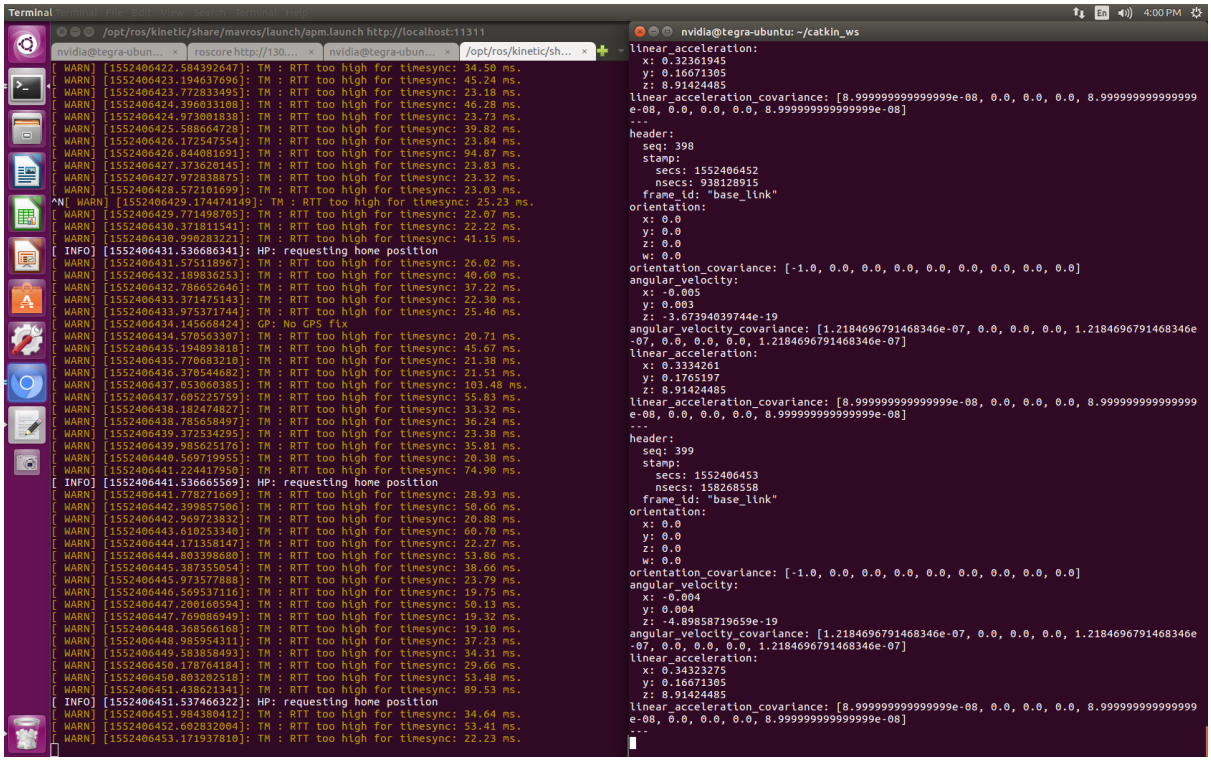


Figure 5.3: Raw IMU Data From Pixhawk to Jetson

5.4 Flight Test Results

Over the course of this project, we conducted several glide tests and flight tests to problem shoot our design and make revisions based on the results. After the initial design and analysis phase we created a large, full scale glide test model as shown in Figure 4.1. However, as shown in the video taken during the test (`https://www.youtube.com/watch?v=J_eMBDd_ewY`) and as described in Sections 3.1 and 4.1, this model was difficult to throw for testing. We thus shrunk the vehicle down to a half scale model shown in Figure 4.2 and tested it as shown in this video `https://www.youtube.com/watch?v=NXaM3ZX9_PE`. At the end of that video and in subsequent testing, that model however showed signs of rolling out of stable flight due to an unstable dutch roll mode. At that point, we decided to redesign the vehicle once again. The final design is shown in Figure 2.2. This vehicle adopted delta wings at a 5° dihedral with rudders moved to the tail of the body. Both of these revisions resulted in a stable aerodynamic analysis in XFLR5 for all dynamic stability modes. Static stability for this new model can be

seen in Figure 3.17.

Once we had proved our design could glide, the group's focus turned to flight tests, particularly hover tests since that would encompass the first stage of a flight plan for our vehicle. During B term, the Aerodynamics & Structures and Power & Propulsion teams built the first iteration of hover test models and conducted the first series of tests towards the end of the term. A video of the results of this test can be seen here https://www.youtube.com/watch?v=ErdHlaw6ojI. Though this flight test was not entirely successful, the team learned that the setup for the Pixhawk required tweaking. After emailing the developers of the Pixhawk firmware, the tilt-rotor and quadplane classes were suggested to us for further research. The "Nimbus" VTOL vehicle was specifically recommended to us by the developers and further research into that frame class setup ensued. Once we felt comfortable in our knowledge of that type of setup, we set the Pixhawk up to perform in a similar fashion and tested it in a hover test at the beginning of C term as shown in this video (https://www.youtube.com/watch?v=8yCkWZajSDM). During the test, the vehicle hovers a couple of inches of the ground for about a second before the central fan support snaps due to vibrations. The central supporting shaft was essentially a beam pinned at only two ends experiencing large amounts of torque from the motors, causing the beam to rotate about its longitudinal axis. These rotations are what caused the vehicle to porpoise as shown in that video. To fix these vibrations we decided to use two streamlined aluminum spars that way the central motor mount would be pinned at four points as opposed to two and aluminum would be much sturdier than the 3D printed PLA mounts previously used in the design. A test of a single spar through the central mount was successfully conducted to confirm this theory. The test can be seen in this video https://www.youtube.com/watch?v=QEwqlIj07Ww. Another issue we encountered with the hover test model was components moving around mid-flight due to vibrations and causing the center of gravity to shift. To fix this issue, component trays were designed. These trays were used to securely mount on board electronics on the new hover test

model that was built for the last series of flight tests. More flight tests were attempted in C term but resulted in the vehicle rolling out of control. The initial hypothesis was that the PID gains were set too high but after a significant amount of testing it was discovered that the issue was incorrect connections between the Pixhawk and the electronic speed controller. The final hover tests were conducted at the beginning of D Term, the most successful of which can be seen here (`https://www.youtube.com/watch?v=nO0P3DVM9As`). During this flight test, the Build Team was attempting to tune the yaw PID gains to minimize the rocking motion the vehicle experienced once after takeoff. This video shows the most stable flight conducted during those tests. These were the last series of flight tests the team was able to conduct.

Chapter 6

Conclusions, Recommendations, and Broader Impacts

6.1 Project Broader Impacts

EVTOL micro-aerial vehicles can be used and applied in everyday operations for military, civilian, and transportation needs. In this section, we discuss the applications of EVTOL MAVs and how they aid public health/safety and improve the parcel delivery economy as well as the social, economic, and environmental challenges they present.

6.1.1 Public Health and Environmental Support

One of the biggest ongoing applications for unmanned aerial vehicles is for parcel delivery. On March 10, 2016, Flirtey, a startup MAV delivery service, became the first company to successfully delivering a package for a half-mile in Hawthorne, Nevada, via drone [23]. This was the first time a UAV has made a fully autonomous delivery in an urban setting in the United States. It wasn't too long until Amazon.com was granted a new patent by the U.S. Patent and Trademark Office for a delivery drone that can respond to human gestures. By December 7, 2016, Amazon Prime Air made its first unmanned drone delivery that was able to transport a

five pound weighted package at a cruising altitude of 400 feet in 30 minutes or less using a GPS tracking system [22]. The benefits behind drone parcel delivery include environmental safety. The decrease of using diesel-burning delivery trucks could mean a reduction in both energy consumption and release of greenhouse gases that contribute to climate change.

Another advantage is to transport valuable medicine and blood products to facilities in need. Zipline, a startup company in California, designs, builds, and operates MAVs, initially in Rwanda, for delivery of blood and medical supplies. Zipline's new delivery vehicle is an autonomous fixed-wing style airplane. The MAV is capable of flying at a top speed of 128 km/h, and a cruising speed of 101 km/h—21 km/h faster than the previous generation of aircraft—with a round trip range of 160 kilometers carrying up to 1.75 kilos of cargo. Since launching the service in Rwanda in 2016, Zipline has flown 300,000 km, delivering 7,000 units of blood over 4,000 flights, approximately a third of which have been in emergency life-saving situations [27]. Now, Zipline is looking into delivering more blood supply to outside of the capital, Kigali and opening up a second distribution center in Rwanda. Overall, with the creation of the MAV, parcel delivery and improve the environment as well as public health day in and out.

6.1.2 Social and Environmental Challenges

The invention and use of EVTOL MAV can present social and environmental challenges as well. For example, can MAVs be capable in protecting people's safety and privacy? Turns out, the Customs and Border Protection (CBP) and U.S. Southern Command, a joint military command agency, have ran successful MAV missions to target drug smugglers and imply potential local law enforcement uses. In 2006, the Los Angeles police department tested a MAV, the SkySeer, for a variety of local functions, including hovering and watching a crime scene, tracking drug dealers, searching for lost children or Alzheimer's patients in difficult terrain, aiding police in pursuits and detecting speeders at a fraction of the cost of operating a helicopter [4]. Another ongoing issue was having unmanned aerial vehicles in the air while there

are other aircrafts that are manned flying. Today, the conversation is still about keeping airspace safe for planes that carry people. In the case of runway maintenance, for instance, the drone's high-resolution camera picks up things the human eye might miss and makes it easier for engineers to spot structural changes over time. With technology on the rise, aviation industries are currently in the making of creating more efficient MAV avionics to keep it at least 1 meter away from the aircraft it's inspecting, to avoid damaging it, and feature a collision avoidance system [4]. With these advancements, MAVs will be able to dodge people or objects that might come into its path, thus making it safer for the people.

EVTOL MAVs can play a vital role when it comes to potential environmental issues that present itself everyday. Human activity has been impacting the environment for thousands of years. Industrial, chemical, and energy-related waste materials have had an exponential effect on the environment. Luckily, companies such as senseFly have been creating MAVs to survey/mapping, mining, agriculture, and for environmental protection. Their MAVs are able to help the environment by assessing rivers and floods by mapping and modeling the changes in the morphology of river basins. This will prevent any potential floods or wetlands happening in the wilderness. By using a non-visible spectrum camera, senseFly drones also can help to assess plants and their vegetation, determine soil characteristics, and estimate biomass in order to conserve plants all around the world [21]. With senseFly, there are already pathways as to how MAVs can contribute and help combat against environmental issues that have been lingering issues in the world.

6.1.3 Economic Impact

EVTOL MAVs can have an economical impact on parcel delivery. MAVs are considered a disruptive technology in that it is a new way of doing things that disrupt or overturn traditional business methods and practices. In this case, parcel delivery companies such as Amazon and UPS are currently in works of including MAV delivery services into their businesses. The theory behind this is how Amazon will set up its drone system only in cities where it will be

expected to increase efficiency and profitability [17]. For example, the drone system would be costlier and less profitable in a low density/low population city versus a high density/high population city. Urban areas with denser and larger populations of customers are expected to provide scale economies. Amazon's MAV technology is a labor saving/capital-using technology as MAVs replace labor and trucks with more specialized capital and labor in the production delivery services. In Figure 6.1, Introduction of drone technology switches production from Q1 to Q2, replacing labor with capital and increasing the capital to labor ratio. Figure 1 shows the new technology (Q2) changing from labor intensive to capital intensive. Amazon's application of MAVs will save money on labor fees by switching the traditional truck driver, where one person drives one truck at a time, to having one-person controlling/monitoring multiple drones at once.

Based off of UPS delivery costs, it would take around $1.20 for one package to be delivered taken into account expenses like driver cost and gas/toll fees [26]. In a study in Chattanooga, Tennessee, 4,943 packages less than 5 pounds were delivered on their busiest day of the year [6]. Using this number we can determine the amount of packages delivered over a 5-year span using the equation below. (Used 51 weeks due to major holidays.)

$$\frac{4943 \text{ packages}}{\text{day}} \cdot \frac{6 \text{ days}}{\text{week}} \cdot \frac{51 \text{ weeks}}{\text{year}} 5 \text{ years} = 7,562,790 \text{ packages} \tag{6.1}$$

Dividing the total cost for MAV delivery 24 hours per day by the 7,562,790 packages delivered, we have the cost to deliver a package by MAV to be .20 cents a package [15]. With the total cost for MAV delivery 12 hours per day, the cost to deliver a package by MAV would be around .40 cents a package. So with the busiest day for deliveries in Chattanooga, Tennessee, MAV delivery costs per package is a third or less of the UPS delivery cost. Overall, the MAV industry is a growing industry because of the invention being a disruptive technology that can take the place of the modern day on ground deliveries. Companies like Amazon and UPS and greatly benefit MAV delivery by saving labor and gaining capital.

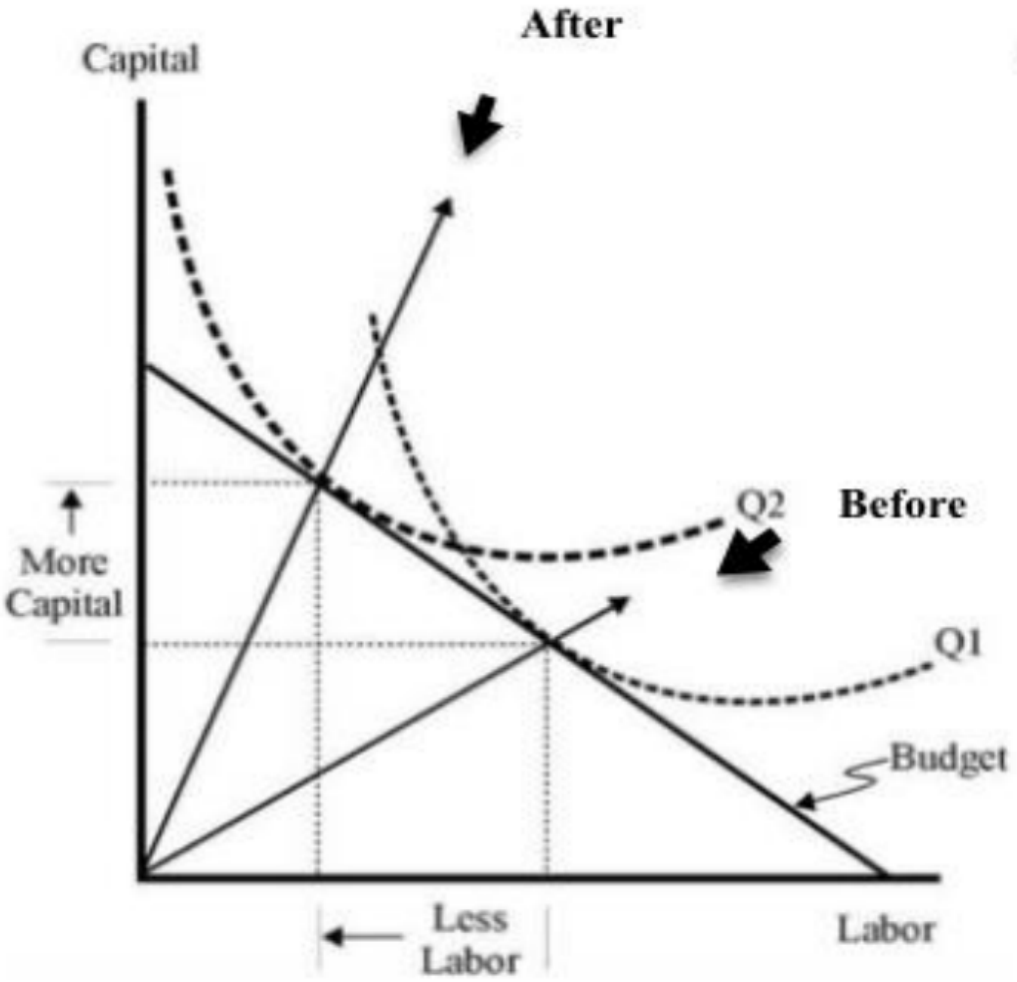


Figure 6.1: Labor Saving Innovation with Increasing Capital.

6.2 Conclusions

This project succeeded in designing a hybrid electric powered micro aerial vehicle, however it fell short when it came to the actual building and flying due to several reasons. Early on the project work was focused on building an analysis toolbox based off of two similar aircraft to our proposed design. While this was helpful for some design aspects, in some areas it hindered the forward momentum of our project. One design aspect this approach was helpful for was the aerodynamic analysis. By modelling wings in XFLR5 and then confirming the data through wind tunnel testing early on in the scope of the project, we were able to pick an airfoil that we stuck with throughout the course of the project. However, where this approach fell short was with the controller design and DC to DC converter. Our software and simulation team spent close to two terms doing dynamical modeling and designing a controller based off of a previous design with the intention to convert this controller to a language that could be put on our computer board. However, later on in the project we found that a controller pre-built into the pixhawk could sufficiently control our machine. Similarly, our power and propulsion team spent time designing a DC to DC converter to increase the voltage of a battery so we could decrease the weight allotted to that component. This converter was never used because after one of the design iterations we determined that the aircraft would be able to carry the heavier battery without issue. Instead, more time should have been spent on the actual design of the aircraft early on in the project as that ended up being a major setback to the project. More than halfway through the project, once building had started, we recognized the need for a complete redesign which wouldn't have been such a major setback if this had been realized earlier on in the project. Because of this our team spent the majority of the latter half of the project stuck in design iteration loops and only had time to test the aircraft's hover capabilities.

Another area our project fell short in was software integration. While in theory our aircraft was capable of target detection and obstacle detection and avoidance, in practice it didn't quite come together. Because we chose components early on based off of their performance specifications

on the individual component level we ended up with a mess of incompatibility. It took close to two terms worth of work to get our Realsense camera working on our Jetson board simply because the two components were not designed to work together. Because of this setback we did not have time to get the pushbroom stereo algorithm working and therefore were not obstacle avoidance capable. We were however able to get circle tracking working but not without incompatibility issues. We discovered early on that the Raspberry PiCam that we had purchased didn't have the same CSI connection type as our board and couldn't be used. After we switched to a USB camera, starting with one we found in the lab and then purchasing the Ocam, we realized that the way in which the kernel has to be patched to get the Realsense working makes it so the Jetson doesn't recognize the data coming in from the USB cam. This essentially means we couldn't run the Realsense and the Ocam on the Jetson at the same time and a new computer board would have to be added to the aircraft to run the circle detection at the same time as obstacle avoidance. If more research had been done early in the project on ease of system integration and not just that the technical specs meet our needs the software implementation may have gone smoother.

6.3 Recommendations

6.3.1 Build

In future projects, we strongly encourage teams to put a greater emphasis on preliminary vehicle design/construction and testing schedule. We believe that pairing our vehicle construction time line with our testing schedule early in the project may have pushed development and successful tests at a faster rate. This design approach is most achievable with a modular vehicle design, which allows for assembly of the bare minimum components and air frame for each step in testing. Furthermore, our first design step should have been to gather an assumed list of components and create CAD mock-ups of each. These models could be used for sizing and spacing design withing the vehicle CAD assembly. This approach may have allowed for us to

identify some design requirements earlier.

With all that said, our greatest hindrances were 3D printer speed/availability and our understanding of the Pixhawk. Having access to a 3D printer proved invaluable to our project due the speed at which we could prototype components. However, we likely relied too heavily on printed pieces and often found ourselves waiting for parts for a multitude of reasons. Using laser cut balsa as a replacement or cutting some non-structural components from foam may have reduced this reliance and allowed for even more rapid development of the vehicle. However, our prototyping speed was ultimately less of a limiting factor than our knowledge of the Pixhawk, especially when it came to using the Pixhawk for quadplane control. The documentation for VTOL capable fixed wing control with the Pixhawk was more limited than expected and we spent a significant amount of our time problem solving issues with the Pixhawk. This likely could have been avoided early on if we developed a mock testing rig to practice with the different control parameters specific to quadplanes in the ardupilot firmware.

6.3.2 Software

In future projects, greater consideration and research should be done before selecting components that are not natively mutually supportive. Getting components such as the Realsense and Jetson working together took far more time than was anticipated. In the future, the optimization of such components for parameters like weight, size, computational power, power draw etc. should be considered less important than how the components will work together. Furthermore, plans for running the controls and autonomy components onboard another vehicle should be made much earlier. Instead of trying to get each component working on the bench and then planning to integrate with vehicle at the end, each component should be developed for bench testing and then tested onboard a pre-built test platform as this process itself is not straight forward.

Another issue the software team faced was the overlapping use of the kernels drivers. As mentioned before to run the Realsense on the Jetson it was required to patch the kernel. This

means that the driver used for the USB camera is also used for the Realsense camera. When we applied the realsense patch we caused the USB camera to no longer work. We theorized two possible solutions to this issue. The first and easiest would be to use a secondary less powerful computer to interpert the camera data. With this method the secondary computer would capture the video and analyze it. then using Ehternet it could send the important information such as circle location through ROS to the main computer, the Jetson. The second and more technically challenging solution would be to duplicate the camera drivers and make a second driver so that both camera would work on this system. This idea was not thoroughly researched, but could be a possible solution.

Bibliography

[1] M. Bangura, R. Mahony, *Nonlinear Dynamic Modeling for High Performance Control of a Quadcopter*. Australian National University, Canberra, Australia. 2012. Retrieved from https://pdfs.semanticscholar.org/e899/0d6f6144a34e3ea31ff55094607bcf5b118d.pdf.

[2] Barry, Andrew. *"Flying between obstacles with an autonomous knife edge maneuver"*. PhD Thesis. Massachusetts Institute of Technology, Massachusetts, USA. 2016. Retrieved from https://dspace.mit.edu/handle/1721.1/103718.

[3] Beard, Randal W., and McLain, Timothy W. *Small Unmanned Aircraft: Theory and Practice.* Princeton University Press, Princeton (2012).

[4] Bowes, Peter, 6 June 2006, "Americas — High Hopes for Drone in LA Skies", BBC News. Retrieved from news.bbc.co.uk/2/hi/americas/5051142.stm.

[5] Chase, *"Chase1325/IPAS"*. (2019). GitHub Repository. Retrieved from github.com/Chase1325/IPAS.

[6] Cheredar, (December 27 2012), "At Peak, Amazon Sold a Whopping 306 Items per Second in 2012", Venture Beat (VB News). Retrieved February 10, 2015, from https://venturebeat.com/2012/12/27/at-peak-amazon-sold-a-whopping-306-items-per-second-in-2012/

[7] Cuda Zona. (2018, December 19). Retrieved from: https://developer.nvidia.com/cuda-zone

[8] Hochstenbach, M., Notteboom, C., Theys, B., Schutter, J.D., (2015, December). "Design and Control of an Unmanned Aerial Vehicle for Autonomous Parcel Delivery with Transition from Vertical Take-off to Forward Flight - VertiKUL, a Quadcopter Tailsitter" International Journal of Micro Air Vehicles. Volume 7 Number 4. DOI: 10.1260/1756-8293.7.4.395.

[9] Jetsonhacks & Mohammedari, (2018, September 08). "buildLibrealsense2TX", GitHub. Retrieved March 3, 2019 from `https://github.com/jetsonhacks/buildLibrealsense2TX`

[10] Jetsonhacks, (2018, April 27). "Intel RealSense Camera ROS Package - NVIDIA Jetson TX", Youtube. Available: `www.youtube.com/watch?v=-q3jGJgFZ7Q`

[11] Jetsonhacks, *"jetsonhacks/buildOpenCVTX2"*. (2018, July 5), GitHub Repository. Retrieved from `https://github.com/jetsonhacks/buildOpenCVTX2/tree/e9e0bb0814c879198e9175c7161add893a8e6c04`

[12] Jetsonhacks, (2018, April 26). *"jetsonhacks/installROSTX2"*, GitHub. Retrieved from `https://github.com/jetsonhacks/installROSTX2`

[13] Johnson, Wayne. (1980). *Helicopter Theory - 6.1.3 Power Available*. Dover Publications. Retrieved from `https://app.knovel.com/hotlink/pdf/id:kt00AZX4V2/helicopter-theory/power-available`

[14] Kamenev, *"NVIDIA-AI-IOT/redtail"*. (2019, December 13). GitHub Repository. Retrieved from `https://github.com/NVIDIA-AI-IOT/redtail/wiki/Dev-board-UART-and-Pixhawk-connection`

[15] Keeney, Tasha. (2015, May 5). "How Can Amazon Charge $1 for Drone Delivery?", Arkinvest.com. Retrieved February 22, 2019 from `https://ark-invest.com/research/drone-delivery-amazon`

[16] Lucas, A., Ermakov, V. (2019, March 3) "Mavros Extras", GitHub Repository. Retrieved from https://github.com/mavlink/mavros/tree/master/mavros_extras

[17] Mack, Eric. (2018, February 13). "How Delivery Drones Can Help Save The World", Forbes. Retrieved from https://www.forbes.com/sites/ericmack/2018/02/13/delivery-drones-amazon-energy-efficient-reduce-climate-change-polluti #44f053626a87

[18] MAVROS. (2019) "MAVROS (MAVLink on ROS)", Dronecode. Retrieved from https://dev.px4.io/en/ros/mavros_installation.html.

[19] Meier, L., Willee, H., Lukaczyk, T. (2018, 14 September). "C-UART Interface Example", GitHub Repository. Retrieved from https://github.com/mavlink/c_uart_interface_example

[20] Ozdemir, Ugur & Aktas, Yucel & Vuruskan, Aslihan & Dereli, Yasin & Farabi Tarhan, Ahmed & Demirbag, Karaca & Erdem, Ahmet & Duygu Kalaycioglu, Ganime & Ozkol, Ibrahim & Inalhan, Gokhan.*Design of a commercial hybrid VTOL UAV System*. Journal of Intelligent & Robotic Systems Vol 64 (2014). DOI 10.1007/s10846-013-9900-0.

[21] SenseFly, (2019). "Environmental Protection", Parrot Group. Retrieved January 10, 2019, from http://www.sensefly.com/industry/environmental-protection/

[22] Shaban, Hamza. (2018, March 22). "Amazon is Issued Patent for Delivery Drones that Can React to Screaming Voices, Flailing Arms". The Washington Post, WP Company. Retrieved from https://www.washingtonpost.com/news/the-switch/wp/2018/03/22/amazon-issued-patent-for-delivery-drones-that-can-react-to-screaming-

[23] Sonnor, (September 13 2018). ”Single operator runs multiple drones in Reno for first time”, Associated Press. Retrieved January 31, 2019, from `https://www.apnews.com/f475cff0b5f24992823a0ec23a25d9ab`

[24] Stumpe, (November 2017). ”Drones to the rescue”, AIAA. Retrieved October 31 2017 from `aerospaceamerica.aiaa.org/features/drones-to-the-rescue`

[25] Torvalds. (2019, February 7) *”Torvalds/Linux”*, GitHub Repository. `https://github.com/torvalds/linux`.

[26] Welch, Adrienne. *”A Cost-Benefit Analysis of Amazo* rimePAir” .n Honors Thesis. University of Tennessee at Chattanooga, Chattanooga, Tennessee. 2015. Re tieved from `https://scholar.utc.edu/cgi/viewcontent.cgi?article=1051&context=honors-theses`.

[27] Zipline. (2019). ”We’re Providing 11,000,000 People with Instant Access to Urgent Medicines”. Zipline. Retrieved from `flyzipline.com/impact/`