Summer 2016

Thrust sensing for small UAVs

Christopher Scott Marchman

Department:

THRUST SENSING FOR SMALL UAVS

by

CHRISTOPHER SCOTT MARCHMAN

A THESIS

Presented to the Faculty of the Graduate School of the

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

In Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

2016

Approved by

Steve E. Watkins, Advisor
Kakkattukuzhy M. Isaac
Chang-Soo Kim

ABSTRACT

Unmanned aerial vehicles (UAVs) have become prevalent in both military and civilian applications. UAVs have many size categories from large-scale aircraft to micro air vehicles. The performance, health, and efficiency for UAVs of smaller sizes can be difficult to assess and few associated instrumentation systems have been developed. Thrust measurements on the ground can characterize systems especially when combined with simultaneous motor power measurements. This thesis demonstrates the use of strain measurements to measure the thrust produced by motor/propeller combinations for such small UAVs. A full-bridge Wheatstone circuit and electrical resistance strain gauges were used in conjunction with constant-stress cantilever beams for static tests and dynamic wind tunnel tests. An associated instrumentation module monitored power from the electric motor. Monitoring the thrust data over time can provide insights into optimal propeller and motor selection and early detection of problems such as component failure. The approach provides a system for laboratory or field measurements that can be scaled for a wide range of small UAVs.

ACKNOWLEDGMENTS

I would like to thank Dr. Steve E. Watkins for his advisement and guidance over the course of this thesis project. Dr. Watkins also provided equipment and laboratory space used in this project. I would like to thank Jeff Birt for providing materials and machining for the cantilever beams and James Schneider for his help with the experimental testing in the wind tunnel. Finally, I would also like to thank my committee members, Dr. Kakkattukuzhy M. Isaac and Dr. Chang-Soo Kim, for their support and technical expertise.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

1. INTRODUCTION

The use of smaller sized (1-5kg) unmanned aerial vehicles (UAVs) have become increasingly common for both military and civilian applications. Militarily, the UAVs are often used for reconnaissance and can be fitted with cameras and other sensors for detecting people or objects. On the civilian side, small UAVs are used for inspecting hard to reach areas, inexpensive film shots, and even package delivery. All these applications require the UAVs to carry equipment and fulfill their mission repeatedly over time. Due to their small size and relative low cost (compared to large missile carrying UAVs) little effort has gone into detecting faults and damages caused by the wear and tear of the environment. Preventative maintenance needs to be implemented as it is cheaper and safer to replace a faulty motor than to wait for a UAV to crash with potentially expensive equipment on board or cause injury to those below. Also, many combinations of motors, propellers, and batteries may be fitted on given airframes. Experimental verification of performance can aid in the selection of these components. This project demonstrates a thrust stand and instrumentation to record the performance, health, and efficiency of a UAV before and after a flight as well as over the lifespan of the UAV.

This thesis demonstrates using thin-foil strain gauges to test various motor and propeller combinations as well as a full micro-UAV. These tests occurred on the ground using a thrust stand consisting of a cantilever beam. The UAV or motor was mounted to the beam which then flexed in response to the thrust produced. The strain gauges, mounted in a full-bridge Wheatstone configuration, and instrumentation converted the mechanical strain to an electrical signal. The strain gauges also detected the vibrations produced by the UAV and motor. Detecting the change in the vibrations over time can give insight to the structural health of the aircraft. Instantaneous power was measured using dedicated voltage and current sensors to evaluate the efficiency and health of the motors. To control the motor and record data during testing, an instrumentation module consisting of a microcontroller, electronic speed controller (ESC), and sensors was constructed. The overall system consisting of the thrust stand, strain circuitry, and power instrumentation provides a scalable capacity for laboratory and field tests.

The project tested motors and UAVs in both a static and dynamic environment. The static environment consisted of the cantilever beam mounted over a laboratory table. For a dynamic environment, a low-speed wind tunnel on campus was utilized. Three individual brushless DC motors were tested in combination with five propellers of various size. A full micro-UAV was also tested. The design criteria addressed measurements of thrust in a range of 0 – 3 N with a resolution of 0.01 N. Power was measured in a range of 0 – 200 W with a resolution of 0.1 W. Strain and power data during the tests needed to be recorded at a set rate. The strain data needed to be sampled at a rate high enough to detect vibrations of several hundred hertz. Finally, the design setup needed to be portable for use in the wind tunnel.

This thesis includes a review of literature outlining previous work done in the field followed by a background section. The design chapter gives details of the design of the cantilever beams for the thrust stand and the instrumentation module for electrical power measurements. Calibration and test repeatability were very important to obtaining accurate and comparable results. The calibration details and testing producers are introduced in the methodology chapter. Finally, the results section gives the details of the findings and conclusions as they relate to the goals of the thesis.

2. REVIEW OF LITERATURE

The complexity and capabilities of unmanned aerial vehicles (UAVs) have increased to meet the requirements of applications in both military and civilian arenas [1]. UAV technologies include contributions from many areas including aeronautics, robotics, electronic hardware, software, and application instrumentation. As these systems have become more commonplace, associated safety protocols and operator requirements have been developed and educational activities for engineering students have been expanded [2] [3]. The size of UAV systems range from aircraft comparable with manned aircraft to extremely small vehicles. Examples of these small systems include the fixed wing Raven and the quadcopter DJI Phantom that weight 2.0 kg and 1.3 kg, respectively [4] [5]. Experimental development and support for larger UAVs can typically use similar infrastructure as conventional manned aircraft. Experimental support for smaller UAVs is less available.

Small UAVs are often highly configurable. Various loads, such as camera systems may be installed and various battery, motor, and propeller combinations are available. To verify that the performance of a particular configuration is adequate for the application, direct experimental testing is an attractive tool. UAVs, whether auto-piloted or remote-controlled, have significantly higher accident rates than their human piloted counterparts. In particular, small sized (0.1-5 kg) UAVs typically have no redundancies to prevent accidents in emergency situations [6]. By monitoring the overall capabilities and health of a given UAV system, damage and wear can be repaired and components can be replaced before they become fatal to the vehicle.

Strain measurement can provide useful insight into the performance and health of a vehicle's structure, such as a UAV. For example, the work by Bazen et al. involved the application of strain gauges to the frame of a UAV quadcopter to monitor individual motor thrusts in flight [5]. Their work found that strain monitoring could improve high speed maneuvers and improve a UAV's response to a critical failure, such as losing a propeller or a motor shutting down mid-flight. Mounting the strain gauges to the frame of the UAV gives excellent data about the immediate condition of the aircraft which is useful for the flight controller. However, the data does not reveal small fatigue damages that occur over

the life of the UAV. These damages can only be found through long-term testing. Separate motors or entire UAVs can be mounted to test stands to monitor the thrust of the UAV before and after flight with the purpose of identifying damages such as motor breakdown, battery degradation, and structural fatigue. Structural fatigue can be found by evaluating the vibrations produced by the UAV. Work summarized by Dimarogonas et al. show that cracks can be detected in a structure by evaluating changes to its natural frequency [7].

To measure the thrust of the UAV on the ground, a thrust stand is required. Thrust stands for large motors generally use load cells to measure the force directly using strain gauges mounted strategically on the load cell, converting the mechanical strain into an electrical signal. The type of load cell will determine the cost, accuracy, and scalability of the measurement. A thrust stand was developed for a 222.5-N (50-lbf) rocket motor using strain gauges and four cantilever beams for support [8]. Full-bridge Wheatstone circuits were used as instrumentation for the strain gauges for temperature compensation and increased sensitivity. The advantages of this approach include the low cost and mature technology associated with electrical strain gauge instrumentation.

Traditional thrust stands for measuring the very small thrusts (μN – mN range) of satellites involve complicated pendulum setups with various configurations. These systems are designed to minimize the signal noise caused by the weight of the thruster [9]. An alternative to the pendulum is a strain gauge thrust stand. A team demonstrated that a strain gauge thrust stand could be constructed to have an accuracy of ± 1 mN with a resolution of 0.12 mN [10]. The strain gauge technique is superior to the pendulum designs for its simplicity and small size, which allow the setup to fit easier inside vacuum chambers. This thesis utilizes both the large and small scale techniques to measure the thrust of motors in in the 100-mN to 10-N range. However, the techniques used can easily be adapted to increase or decrease the required sensing range adjusting the variables in the thrust stand such as beam thickness or length.

This thesis adds upon the previous work of the author which focused on using a webcam to record the displacement of a cantilever beam in real-time to monitor thrust. This apparatus was capable of measuring displacements with a resolution of 40 μm and forces of 1.63 mN [11]. By using the two systems together, the camera system could verify the thrust readings of the strain gauges.

3. BACKGROUND

This chapter contains background information relevant to the thesis to support the understanding of material presented in future chapters. Topics covered include stress and strain, electrical strain gauges, cantilever beams, simulations, propellers, brushless DC motors, and instrumentation.

3.1. STRESS & STRAIN

This thesis involved measuring strain to determine the thrust of various motors and propellers. It was therefore important to know the theory behind stress and strain and the techniques on how these forces are measured. Also presented is the theory behind electrical resistance strain gauges and their configurations.

3.1.1. Theory on Stress & Strain. Stress is the force per unit area that a load exerts on an object, shown in Equation 3-1 where P equals force. In this project, the stress induced on a beam equals the thrust produced by the motor and propeller when approximated as a point force. Stress has units of $\frac{N}{m^2}$ whereas thrust has units of newtons. Because stress is not directly measurable, experimental determination involves measuring strain with a known modulus of elasticity.

Tensile strain is the change in length of an object as it deforms under the applied load and is defined by Equation 3-2, the change in length divided by its original length. Strain is unitless, and is positive for an elongation of the object and negative for a compression. Most materials experience a linear relationship between stress and strain under moderate stresses. Using Hooke's law, the relationship between stress and strain can be seen in Equation 3-3 where E equals the modulus of elasticity. Modulus of elasticity, also known as Young's modulus, is the slope of a stress-strain curve and represents how much a material can resist being deformed permanently when a force is applied. The higher the modulus, the stiffer the material. Typical values for strain are quite small and have units of micro-strain.

$$\sigma = \frac{P}{A} \tag{3-1}$$

$$\epsilon = \frac{\Delta L}{L} \tag{3-2}$$

$$\epsilon = \frac{\sigma}{E} \tag{3-3}$$

All conductive materials experience a resistance change when subjected to strain. When the length of the material changes, the resistance also changes. The electrical resistance of a conductor is shown in Equation 3-4, where R = resistance, ρ = resistivity, l = length, and A = cross-sectional area.

$$R = \frac{\rho l}{A} \tag{3-4}$$

When a conductor is stretched, its length becomes longer and its cross-sectional area is reduced as shown in Figure 3.1. Both of these effects constructively compound to increase the overall resistance of the conductor. Similarly, if the conductor is compressed, the length is shortened and its cross-sectional area is increased resulting in a decrease to the overall resistance.

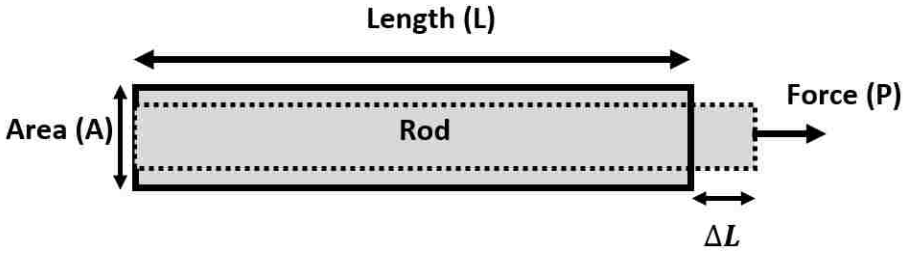

Figure 3.1. Effect of an External Force on a Rod

3.1.2. Electrical Resistance Strain Gauge. Electrical resistance strain gauges take advantage of the electrical resistance property to change their resistance when flexed. While there are different types of strain gauges, each with their own advantages, the ones used and discussed in this project are electrical resistance gauges.

The typical electrical resistance strain gauge is made by etching away a conductive foil layer into a grid as seen in Figure 3.2. The grid pattern increases the total length of the

sensing element, giving it a high enough resistance value to be useful. The pattern also serves to maximize the strain sensitivity in one direction, while minimizing it in the perpendicular direction. By only measuring strain in one direction, the operator is able to get accurate strain measurements that are not influenced by other forces.

Electrical resistance strain gauges are characterized by their overall resistance and their gauge factor (GF). Common resistance values for gauges include 120 Ω, 350 Ω, and 1000 Ω. These resistances must be matched up appropriately with the instrumentation to give accurate readings. GF's are characterized by the change in resistance divided by the change in length, shown in Equation 3-5. The value represents the sensitivity of the gauge with higher values being more sensitive. GF's are unitless and are affected by temperature. Typical values for metallic strain gauges are around two. The GF for the strain gauges used in this project is 2.085.

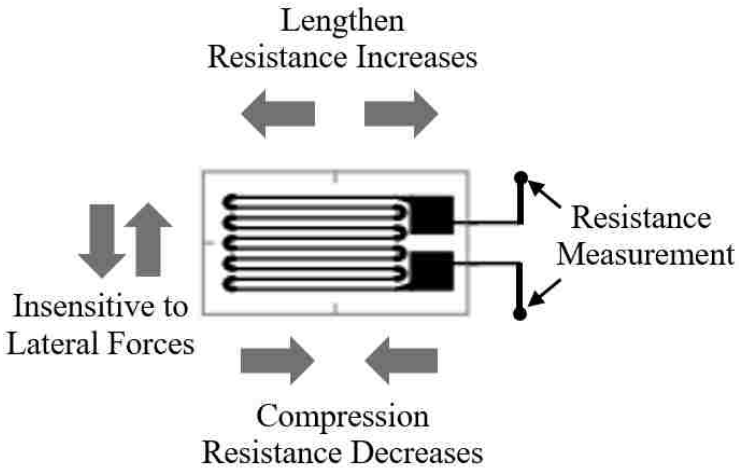

Figure 3.2. Model of Resistance Strain Gauge

$$GF = \frac{\Delta R/R}{\Delta L/L} = \frac{\Delta R/R}{\epsilon} \tag{3-5}$$

Strain gauges are applied to samples through an adhesive that enables the strain gauge to bend and move with the sample. The application of strain gauges is vital to receiving relevant data. If not properly mounted, the strain will not be transferred from the

sample, through the adhesive and backing to the conductive foil on the surface. The application procedure for the strain gauges can be found in Appendix B.

3.1.3. Strain Gauge Configuration. Because the resistance change recognized by a strain gauge is very small (less than 1 Ω), the measurement of this change requires special instrumentation. A Wheatstone bridge, shown in Figure 3.3, utilizes four resistances and a voltage measurement to find a very small change in resistance of one or multiple resistors. The resistance change is measured using a Wheatstone bridge circuit in either quarter-, half-, or full-bridge configurations. When the voltage of V_O is zero, the ratios of $\frac{R_1}{R_2}$ and $\frac{R_3}{R_4}$ are equal. By replacing one of the resistors with an unknown resistor, such as a strain gauge, the voltage measured can be used to find the unknown resistance as seen in Equation 3-6. Comparing the unknown resistance to the original unstressed resistance will give the strain values.

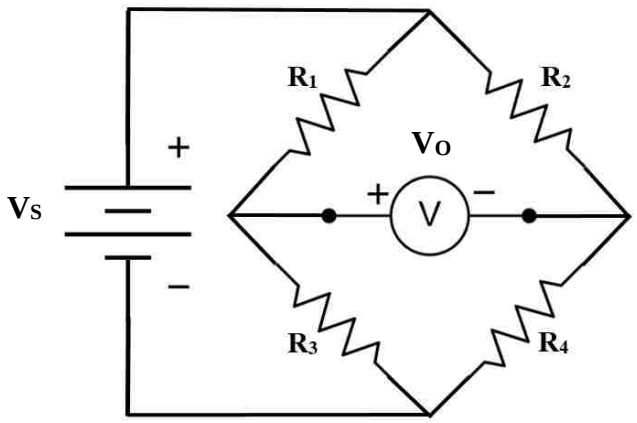

Figure 3.3. Wheatstone Bridge

$$V_o = \left(\frac{R_3}{(R_1+R_3)} - \frac{R_4}{(R_2+R_4)}\right) \times V_s \qquad (3\text{-}6)$$

By replacing one resistor, in this case R_4 with a strain gauge, a quarter-bridge circuit is created as seen in Figure 3.4. The three other resistors serve as reference resistors.

Because the three reference resistors are equal, $R = R_1 = R_2 = R_3$, the Wheatstone bridge equation can be simplified to Equation 3-7.

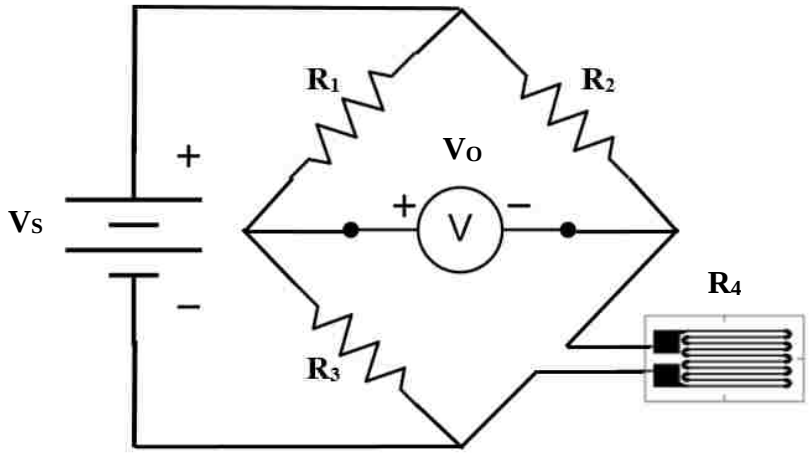

Figure 3.4. Quarter-Bridge Strain Gauge Circuit

$$V_o = \left(\frac{1}{2} - \frac{R_4}{(R+R_4)}\right) \times V_s \qquad (3\text{-}7)$$

This project utilizes a full-bridge Wheatstone configuration, shown in Figure 3.5, which replaces all four reference resistors with strain gauges. The full-bridge configuration allows for maximum sensitivity to strain changes while improving the signal-to-noise ratio. Because all four strain gauges are mounted at approximately the same location, the configuration also compensates for temperature. Two electrical strain gauges were mounted to the top of the cantilever beam and two were mounted to the bottom in the configuration shown in Figure 3.6. The gauges are mounted so when two gauges experience tension, the others experience compression. Because all four reference resistors have now been replaced by strain gauges, the final equation remains the same as Equation 3-6.

3.1.4. Strain Gauge Equipment. The Vishay P-3500 Strain Indicator was used to measure strain for this project. The P-3500 is a portable, battery-powered indicator for use with electrical resistance strain gauges in quarter-, half-, and full-bridge configurations.

As shown in Figure 3.7, the indicator has an LCD screen to display strain results, along with various knobs and buttons to adjust input settings. By default, the indicator has internal 120 Ω and 350 Ω reference resistors and is compatible with the 120 Ω gauges used in the project. According to the instruction manual, the indicator has an accuracy of one micro-strain [12]. The indicator also has an analog coaxial port which was used to import the strain data to the instrumentation module. While the LCD display on the module only has a refresh rate of a couple hertz, the coaxial output is analog allowing the data to be sampled at a much higher rate. The calibration and operation procedures can be found in Appendix B.

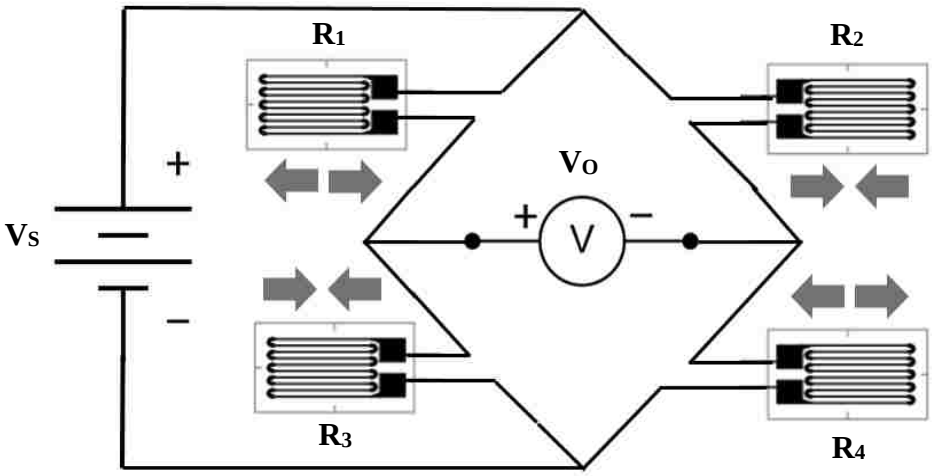

Figure 3.5. Wheatstone Full-Bridge Circuit

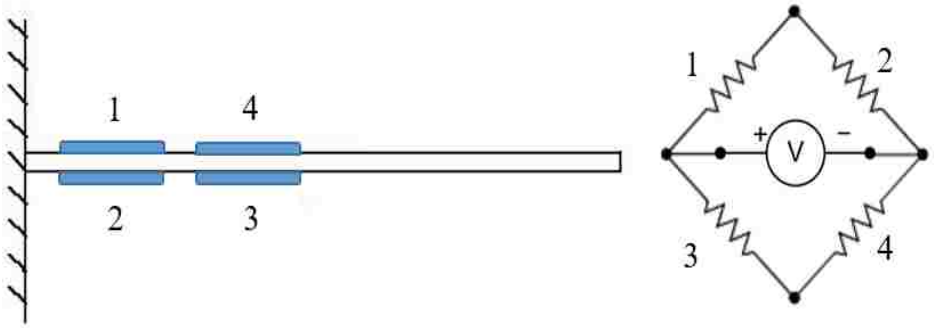

Figure 3.6. Mounting of Full-Bridge Strain Gauges

Figure 3.7. P-3500 Strain Indicator

3.1.5. Cantilever Beam. A cantilever beam is a fixture that is supported at only one end with a load perpendicular to the beam. The simplest form of a cantilever beam is a constant-width beam as shown in Figure 3.8. An important characteristic of the beam is that the surface stress on the beam varies linearly starting at zero at the location of the load. The normal stress on the beam's surface can be calculated by Equation 3-8 where P equals the load (N), b equals the beam width (m), t equals the beam thickness (m), and X equals the location on the beam [13].

Using Hooke's Law, the longitudinal strain at any point, X, of the cantilever beam is given by Equation 3-9. The equation shows the strain varying when moved along the length of the cantilever. The varying strain is important because the placement of the strain gauge on the beam will directly impact the received measurements. When using a half- or full-bridge configuration, the gauges must be carefully aligned to ensure the proper strain is being read by all gauges. Furthermore, the strain gauge will give an average of the linearly changing strain over its area which must be taken into consideration.

$$\sigma(x) = \frac{6PX}{bt^2} \tag{3-8}$$

$$\epsilon(x) = \frac{6PX}{Ebt^2} \tag{3-9}$$

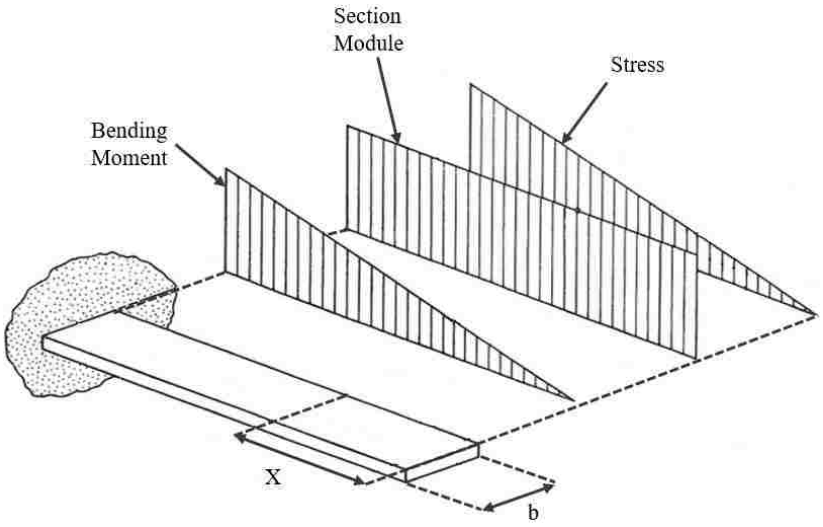

Figure 3.8. Linear Stress Profile for Cantilever Beam

3.1.5.1 Uniform-stress beam. The goal of a constant-stress beam is to have a stress profile that is uniform across the entire length of the beam when a point load is applied at the end. The uniform-stress beam differs from the constant-width cantilever beam, which had a linearly variable stress profile. To construct a beam with a uniform stress profile, the stress equation needs to equal a constant as shown in Equation 3-10. To find a shape that satisfies the condition, the thickness of the beam was fixed and the width was allowed to vary in proportion to X. The progression can be seen in Equations 3-11, 3-12, and 3-13 where K_1 is a constant representing how quickly the beam narrows in width [13].

$$\sigma(X) = \frac{6PX}{bt^2} = \text{constant} \tag{3-10}$$

$$b(X) = K_1 X \quad (t = \text{constant}) \tag{3-11}$$

$$\sigma(X) = \frac{6PX}{K_1 X t^2} = \frac{6P}{K_1 t^2} = \text{constant} \tag{3-12}$$

$$\epsilon(X) = \frac{6P}{EK_1 t^2} = \text{constant} \tag{3-13}$$

As seen in Figure 3.9, the beam now has a triangular appearance with the point load being applied at the very end of the triangle. Because a load applied at a single point is not

practical, a section of the beam material can be added to the end for mounting and applying the load. As long as the load is applied to the convergence point of the triangle, Equation 3-13 is still satisfied. This model represents the constant stress beams used in the project.

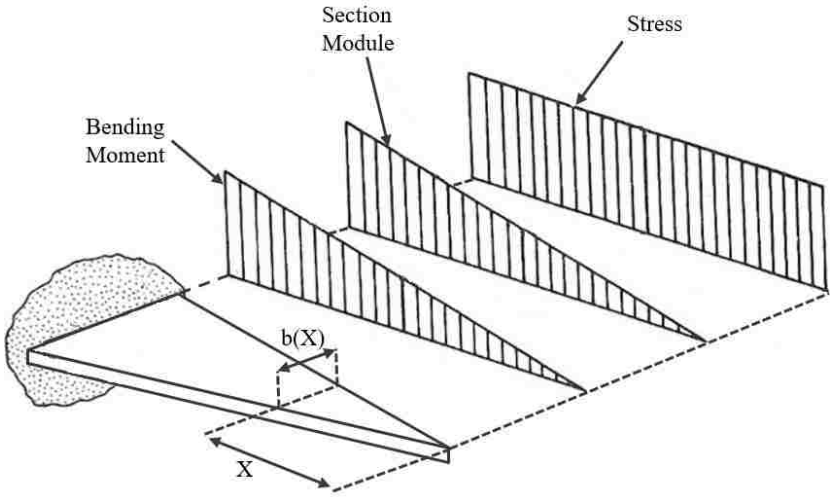

Figure 3.9. Stress profile for Constant-Stress Beam

3.1.5.2 SolidWorks simulations. The program SolidWorks (version 2014 Student Edition) was used to design and simulate the cantilever beams for the thrust stand using finite element analysis. The goal of the simulations is to predict the stresses and strains the cantilever beams will experience under various loads. With this data, the designs can be verified before fabrication. Three different measurements were simulated in SolidWorks: stress, strain, and displacement.

To create the simulations several parameters needed to be specified. After the part and material were selected, the fixed geometry points were specified and are shown with the green arrows. A set load was applied to a point located at the end of the beam, shown with purple arrows. Combined, the cantilever beam has one fixed end and one moveable end. The mesh was then set, with a finer mesh giving more accurate results but with a longer computing time. A detailed comparison between the simulation results and actual results for can be found in Section 6.1

Stress was simulated to determine the yield strength of the designs. SolidWorks calculates an estimated yield strength and maximum stress for the material and shape of a design. Yield strength is the point at which the material will not return to its original shape when the load is removed. The goal is for the beams to be flexible enough to generate high strain readings without exceeding their yield strength. The designs were simulated using 6061 aluminum as the material. The maximum estimated load for the 1.27-mm (0.050-in) thick beam was 5 N. With a load of 5 N, the simulated maximum stress is approximately half of the yield strength, a large enough margin that it is safe to assume that the material will return to its original shape. The same is true for the 0.8128-mm (0.032-in) thick beam with a maximum load of 2 N. Figures 3.10 shows the simulated stress for a constant-width and a constant-stress 6061 aluminum beam of 0.050-in thickness and 22.86 cm (9 in) in length. The estimated yield strength for each beam is also given.

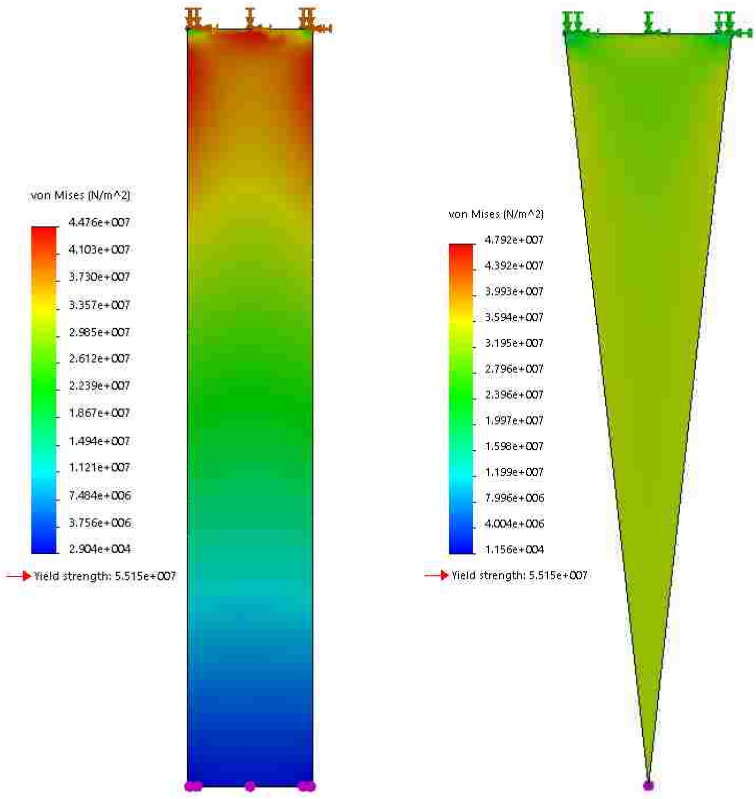

Figure 3.10. Constant-Width & Constant-Stress Beam Stress

3.2. PROPELLERS

A propeller is a “device for providing a force or thrust at the expense of power generated by a motor” [14]. The thrust is produced by moving a mass of fluid (air) in the opposite direction that the craft is being propelled and effectively converts the rotational speed of the motor into linear speed. The propeller operates on the same principle as an airfoil and acts as a rotating wing. The size of the propellers used in this project are expressed as the “diameter” x “pitch” in inches. For example, an 8x6 propeller has a diameter of 8 inches and pitch of 6 inches. The “E” in front of a propeller dimension stands for electric and is means the propeller is designed for electric motors rather than gas turbines. All the propellers used in this project have two blades. The less blades on a propeller, the more efficient it is because there are less blades that have to cut through the air. Oversizing the propeller is a common problem for users. Because the electric motor will attempt to pull more current to maintain the current revolutions per minute (rpm’s), the motor can easily overheat and burn out.

The diameter dimension controls the thrust produced by the propeller as well as the rpm of the motor at a certain power. The pitch is the angle of the blade and is the measure of how far the propeller would move in one full revolution if it was screwed into a solid as demonstrated in Figure 3.11 [14]. The pitch of the propeller controls the speed of the aircraft. Therefore, in Figure 3.11, the 10 x 8 propeller would move the airplane faster than the 10 x 4 propeller even though the diameters are identical.

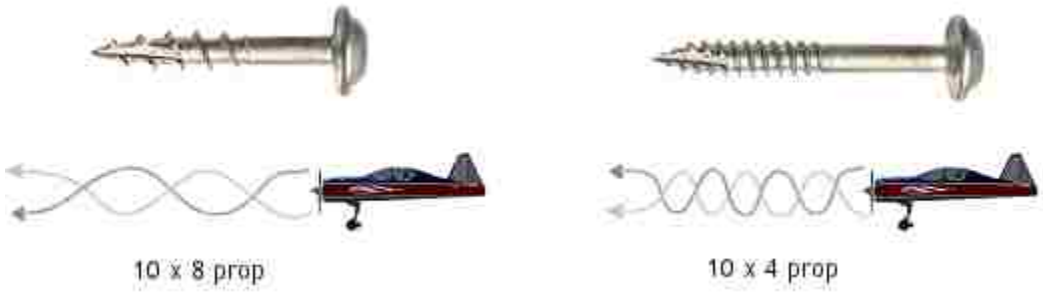

Figure 3.11. Propeller Pitch [15]

The power absorbed by a propeller is influenced by the rpm of the motor, and the diameter and pitch of the propeller. This relationship can be used to calculate the power necessary for an electric motor to turn a propeller at a given speed and is given in Equation 3-14, where power is measured in watts, K equals the propeller constant, the diameter and pitch are represented in feet and rpm has units in thousands [14]. The ideal speed of the aircraft can be determined by Equation 3-15, where pitch is in inches, K equals the propeller constant, and the numbers convert the units, inch per minute into mph [14].

$$\text{Power} = K \times \text{Pitch} \times \text{RPM}^3 \times \text{Diameter}^4 \tag{3-14}$$

$$\text{Speed (mph)} = \frac{60 \times \text{RPM} \times \text{Pitch}}{K \times 12 \times 5280} \tag{3-15}$$

3.3. BRUSHLESS DC MOTORS

BLDC (Brushless DC) motors are synchronous machines with a permanent magnet rotor and stator coils compared to a brushed DC motor which contains rotor coils and a permanent magnet stator. Figure 3.12 shows the difference between the two electric motor types. By reversing the rotor and the stator, a BLDC motor can remove the need for brushes.

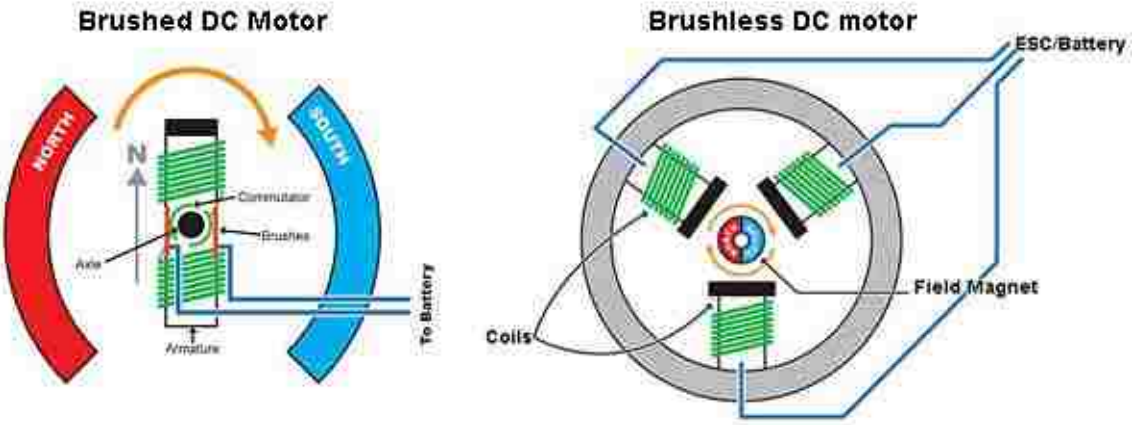

Figure 3.12. Difference Between Brushed and BLDC Motors [16]

A BLDC rotates due to the interaction between the electromagnets and the permanent magnet. When DC power is applied to one of the stator coils, the opposite poles of the stator and rotor are attracted to each other which pulls the rotor magnet towards the energized stator [17]. Alternating which coils are energized causes the rotor to spin. The concept can be applied in reverse, with a stator coil pushing the rotor rather than pulling it. Most BLDC controllers incorporate both of these techniques to provide maximum power to the motor. Figure 3.13 shows the polarity of the voltage of the stator coils needed for a continuous 360 degree rotation. For brushed motors, the timing control for each stator coil is done manually by the commutator. For BLDC motors, the timing is controlled electronically. Many large BLDCs utilize embedded Hall effect sensors to send position information back to the controller for more accurate control, however the ESC and motors used in this thesis do not utilize this technique [15].

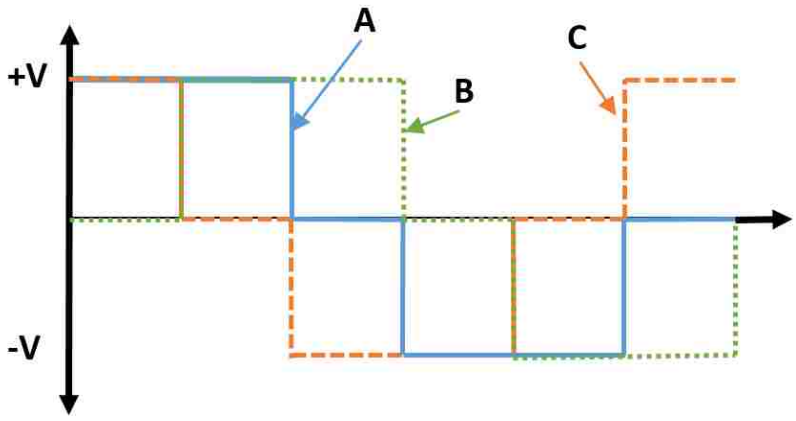

Figure 3.13. DC Voltage Required for Each Stator Coil

There are two classes of BLDC motor, the inrunner and the outrunner. Inrunners have their permanent magnet rotors located inside of the electromagnet stator and which allow them to operate at high speeds. The high speeds allow the inrunners to be more efficient than the outrunner motors but cause the inrunners to produce lower torque, requiring a gearbox. Therefore, outrunner motors, with their permanent magnet rotors outside the electromagnet stator are typically used for propeller propulsion [15]. The three

large independent motors used in this project are outrunners, while the small standalone UAV has inrunner motors.

BLDC motors have many advantages over their traditional brushed counterparts. They have no brushes, which wear out over time and require maintenance. These brushes are also noisy (audibly and electrically) and can cause sparking. BLDCs are also more efficient due to their lower weight and are superior at high speed applications over 5000 rpm [17]. BLDC motors also have a constant torque profile. These characteristics make them the ideal motor for small UAV applications. The main disadvantage to BLDC motors are their high price and complicated control electronics due their need of an electronic commutator.

For BLDC motors used for UAV applications, several key specifications are given. One of the common specifications listed on small brushless motors is K_v, which represents rpm per volt. A high K_v equates to a motor with a high speed and lower torque profile while a low K_v equals to a lower speed and higher torque [18]. The K_v required depends on the application. Generally, a higher K_v at the same voltage is more efficient and preferred. One of the motors used in this project has a K_v of 1200 and is used at 11.1 volts. Therefore, the max rpm of the motor can be found to equal 13320 rpm. This value is for a no-load condition.

3.4. INSTRUMENTATION

An instrumentation module was constructed to record data from the sensors and to control the motor. This section gives background information on various components and sensors used in the module.

3.4.1. Analog-Digital Converter (ADC). An ADC converts an analog voltage signal into a useable digital value. The ADC operates by repeatedly comparing the incoming voltage to a known internal voltage. The more bits the ADC has, the more comparisons can be made and the more accurate the result will be. The Arduino Micro microcontroller used in the instrumentation module has a 10 bit ADC, giving $2^{10} = 1024$ possible values. A typical microcontroller will multiplex multiple pins into one ADC. A constant capture rate is needed to have an accurate time scale for the data. As shown in

Section 5.1.4.1, the maximum frequency the microcontroller can accurately record data is 1000 Hz (1 ms period).

3.4.2. Communications. Computer communications are broken up into two categories, parallel and serial. While parallel communications are faster they require more input/output pins and overhead. Serial communications are more often seen on microcontrollers such as the Arduino Micro. This microcontroller has built-in serial architecture that connects easily to a host computer through USB. The baud rate identifies how fast data is sent over the connection. As long as both the transmitter and receiver use the same baud rate, data can be transferred. However, too high of baud rates will result in missing or incorrect data. The Arduino Micro can successfully transmit data up to 230400 bits per second (bps).

The Arduino Micro microcontroller acts as the transmitter for the serial connection. The receiver is a software program called CoolTerm which runs on the host computer. CoolTerm is a free program that can store all data sent over the serial port into a text file [19]. The data can then be imported into Microsoft Excel for analysis. The program can accurately receive data from baud rates up to 230400 bps making it compatible with the microcontroller.

3.4.3. Voltage Sensor. The goal of the voltage sensor is to detect the voltage being produced by the power source. The voltage is used to calculate power and to prevent under and over voltage conditions. A commercial op-amp based sensor was chosen rather than using a voltage divider with two resistors. The op-amp has a very high input impedance, meaning very little current will be drawn from the circuit. The lower the current draw, the less impact the sensor has on the circuit.

3.4.4. Current Sensor. A Hall effect sensor was used to measure the current being consumed by the motor. The sensor generates a voltage in response to a magnetic field with the magnetic field being generated by the current flowing out of the power source. The Hall effect is the process of generating a voltage in the presence of a magnetic field. Due to the Lorentz force, the electrons flowing through a conductor deviate in the magnetic field and create a voltage difference. The magnetic field is placed perpendicular to the current generating the Hall voltage. When the magnetic field and voltage are known, the values can be used to find the current. One of the main benefits to a Hall effect sensor

is the electrical isolation between the mains circuit and the sensing circuit. Because of the high (up to 20 A) currents being measured, a malfunction could unintentionally send the high current into the low-current environment of the microcontroller. Figure 3.14 shows the chosen Allegro ACS715 current integrated circuit and its implementation of the Hall effect.

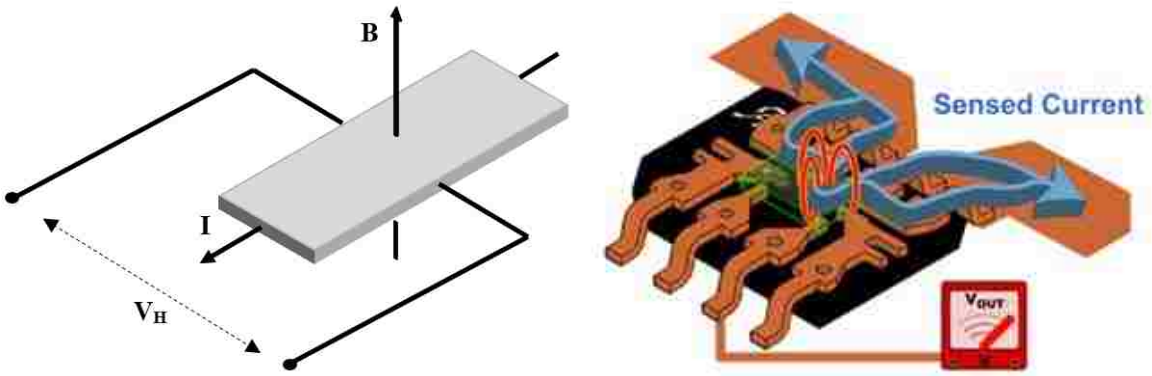

Figure 3.14. Hall Effect Integrated Circuit [20]

3.4.5. Electronic Speed Controller (ESC). BLDC motors require an AC signal to rotate which is provided by the ESC. The ESC energizes two of the three motor wires at a time. As the motor spins, the third wire generates a voltage that corresponds to the current speed. The ESC uses the information to adjust the timing of the output to keep the motor rotating at the selected speed.

The speed of the ESC is controlled by a PPM (pulse position modulation) signal, similar to the more common PWM (pulse width modulation). PWM changes the duty cycle or "on time" of the pulse to convey information. For example, if a signal has a 50 percent duty cycle, then the on and off times are equal which represents a perfect square wave. PPM differs by adjusting the time in between fixed width pulses. The difference can be seen in Figure 3.15. On the Arduino platform, a PWM signal can be written with a simple AnalogWrite command, however a PPM signal is produced by the servo library and uses a timer interrupt to output the correct pulse frequency. The microcontroller has eight bits of

resolution for AnalogWrite and the servo library which gives 255 different positions for the duty cycle.

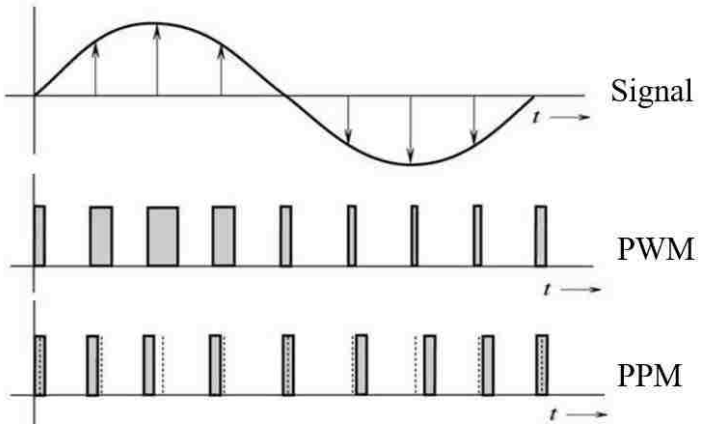

Figure 3.15. PPM versus PWM

3.5. WIND TUNNEL

An open-loop wind tunnel was used to test the motors, propellers, and thrust stand under real-world conditions. The wind tunnel, located in room 334 of Toomey Hall at the Missouri University of Science and Technology, was an open-loop wind tunnel. The open loop design drew ambient air through a vent on the roof of the building, pulled the air through the tunnel, and then expelled the air through a separate vent on the roof. A diagram of the wind tunnel used in this project can be seen in Figure 3.16. The tunnel had a square test section of 45.72 cm (18 in), shown as the transparent section in Figure 3.17, and had a maximum air velocity of 31.29 m/s (70 mph). The tunnel also featured a honeycomb structure and several filters to smooth the incoming air to reduce turbulence. The less turbulence in the tunnel, the more accurate the measurements. Air was moved through the tunnel using a large radial air compressor that relied on centrifugal force to move the air. A pitot tube was used to measure the air velocity of the tunnel and worked by measuring the difference between the static and total pressure.

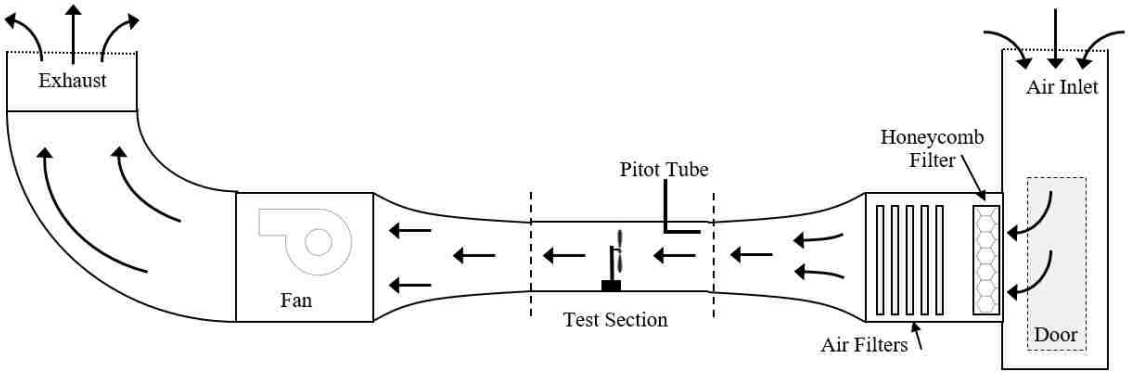

Figure 3.16. Wind Tunnel Diagram

Figure 3.17. Wind Tunnel Picture

4. DESIGN

This chapter describes the design process and the final design of the cantilever beam and instrumentation module as well as the specific motors and propellers used in this thesis. The goal of this work was to satisfy the following criteria. The design needed to measure the thrust in range of 0 N – 3 N with a resolution of 0.01 N. Strain and power data during the tests needed to be recorded by the instrumentation module. The data needed to be sampled at a high enough rate to detect vibrations. Finally, the design setup needed to be portable for field testing and sized for use in the wind tunnel as described in Section 5.2.2. Three individual motors and one micro-UAV (unmanned aerial vehicle) were chosen for testing. Three aluminum cantilever beams were constructed with thickness of 0.8128 mm (0.032 in), 1.27 mm (0.050 in), and 1.5748 mm (0.062 in). Figure 4.1 gives an overview of the entire design.

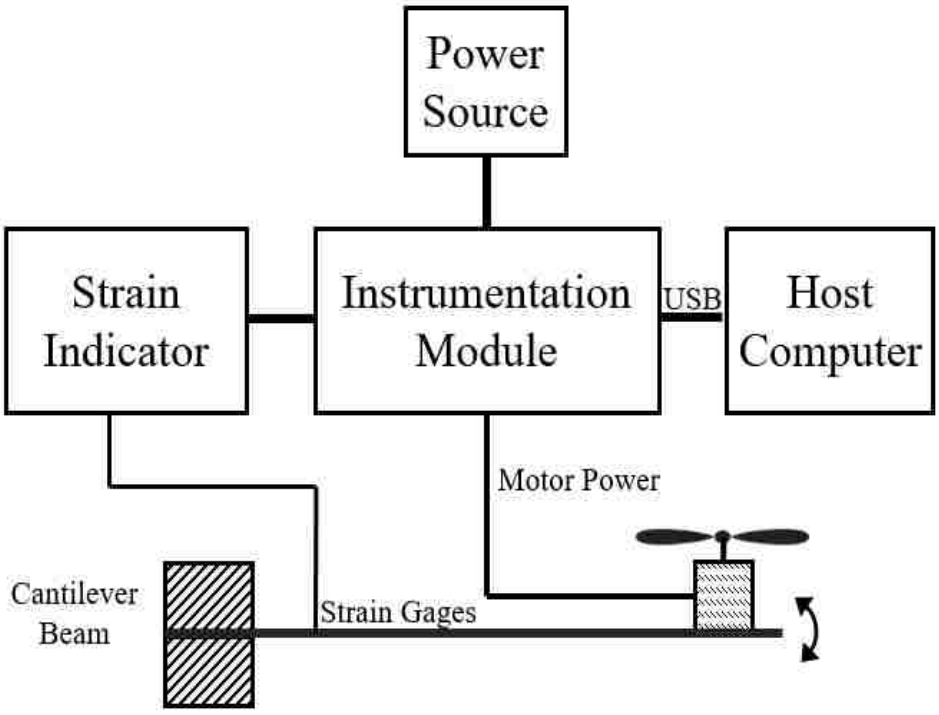

Figure 4.1. Overview of Design

4.1. MOTORS

The motors selected were chosen to meet the thrust range criteria of having a thrust range of 0 N – 3 N. Three individual motors were chosen, approximately the same physical size but with three different K_v ratings. These motors are typically used for 30 to 50 mph fixed wing aircraft or for quadcopters. Along with the individual motors, a micro-UAV was chosen to test the system with a full, physical UAV.

4.1.1. Standalone Motors. Three separate motors were tested using the 0.050-in and 0.062-in cantilever beam. Multiple motors were tested to identify how they affected the thrust produced by the different propellers. Manufacturers' data on each of these motors can be found in Appendix A. The motors were chosen because they had the same coil diameter (28 millimeters) but had different K_v ratings.

The first motor, shown on the left in Figure 4.2, is a NTM 28-26 1200 K_v motor and was repurposed from a previous project. The second motor, shown on the right in Figure 4.2 is a Turnigy Park300 1080 K_v motor. This motor was chosen to evaluate the effect of the small motor length on thrust and power. The third motor, seen in Figure 4.3 is the Turnigy SK3 980 K_v motor. This motor was chosen to serve as a comparision to the NTM 1200 K_v motor, i.e. the motor has the same size, but has a different K_v rating.

Figure 4.2. NTM 28-26 1200 K_v Motor & Turnigy Park300 1080 K_v Motor

Figure 4.3. Turnigy SK3 980 K_v Motor

4.1.2. Small UAV Motors. The Syma S107 UAV, pictured in Figure 4.4, was chosen to identify whether the testing apparatus was able to scale down to smaller motors with lighter thrust potentials. The UAV is small enough to show the testing of a full UAV and not just the individual motors. It contains three small motors, two for the main rotor and one for the rear rotor. These motors are inrunner brushless motors rather than the typical outrunner variety giving them a much higher speed but less torque as explained in Section 3.3. As seen in Figure 4.5, these motors require a gear box to lower the motor rpm to a useable level. The tail rotor has a much smaller propeller and a smaller torque constraint so does not require a gearbox. Unfortunately, since the motors are encased in the frame of the UAV, they could not be run from the instrumentation module and the only obtainable data is strain and approximate throttle position.

4.2. PROPELLERS

Five propellers were chosen to be tested with each of the three individual motors. The propellers were chosen because they fit the recommended range of all three motors and had thrust data provided by the manufacturer for comparisons. Figure 4.6 shows the propellers used for testing. From left to right the dimensions are E4.5x4.5, E5x5, E7x6, E8x4, and E8x6, with the first number equaling the length and second equaling the pitch, both in inches. Manufacturers' data on each propeller can be found in Appendix A.

Figure 4.4. Syma S107 UAV

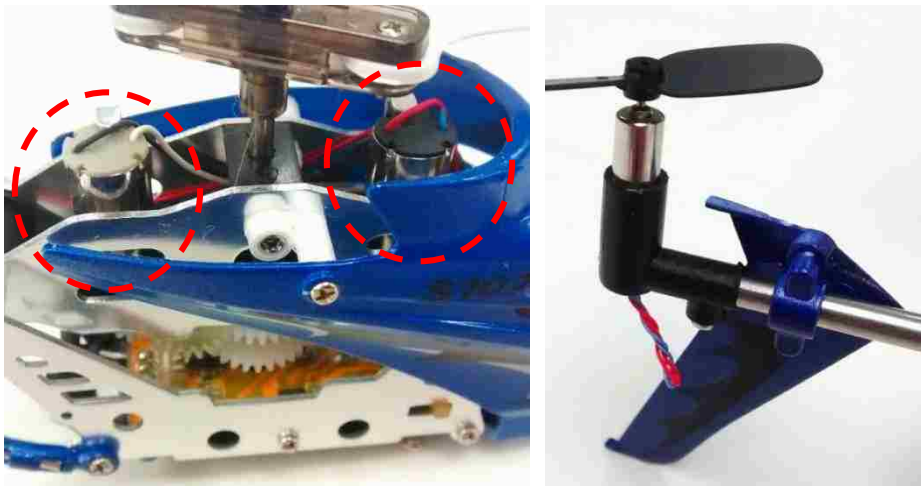

Figure 4.5. Syma S107 Motors

Figure 4.6. Five Propellers used in Tests

4.3. CANTILEVER BEAMS

The goal of this project is to measure the thrust produced by motors and propellers by measuring strain on a cantilever beam. A beam was designed to meet the following criteria. The beam needed to have an area for four strain gauges to utilize the full-bridge configuration for the Wheatstone circuit. It needed to fit in the cross section of the wind tunnel and attach to the given wind tunnel mounting bracket as well as mount to the metric table for static testing in the lab. The beam needed to be light weight and easy to manufacture. Finally, the beam would need the capacity to test all three individual motors as well as full UAVs.

Because of the difference in thrust ranges for the individual motors versus the micro-UAV, two identical cantilever beams were manufactured with the only difference being thickness. The thinner beam experienced no problems with testing. The 0.050-in beam failed during static testing and was replaced with a second version which included more supports and a 0.062-in thick profile.

4.3.1. First Iteration. The first iteration cantilever beams, made in 0.032-in and 0.050-in thicknesses were designed to meet the established criteria. The constant-stress beam shape was designed to allow placement of two strain gauges on each side of the beam for a total of four gauges. In Figure 4.7, the two top gauges can be seen on the top half of the beam. A constant-stress beam allowed for the gauges to be placed at any location in the constant-stress region. The beam's length was determined by the wind tunnel 18-in cross section. The beam's 9.0-in length placed the center of the motor at the center of the wind tunnel. The mounting circle at the end of the beam was designed for the motor mounts of the individual motors. The other end of the beam was designed to be mounted to the wind tunnel mount and the metric table for static testing. The beams were cut from 6061-sheet aluminum with a thickness of 0.032 in and 0.050 in by a CNC machine. Aluminum was used because of its high sensitivity to strain, availability, and easiness to machine. In particular, 6061-aluminum was used because of its availability. Half inch aluminum blocks were cut to support the beam during static testing.

The beams were prepared for testing by applying the electrical strain gauges to each beam using the procedure outlined in Appendix B. Superglue was used to bond the strain gauge wires to the beam and 3.5-mm connectors were used to connect the strain gauges to

the strain indicator module. Using the connector rather than bare wires gave a better connection and could be connected and disconnected quickly, multiple times without risk of damage.

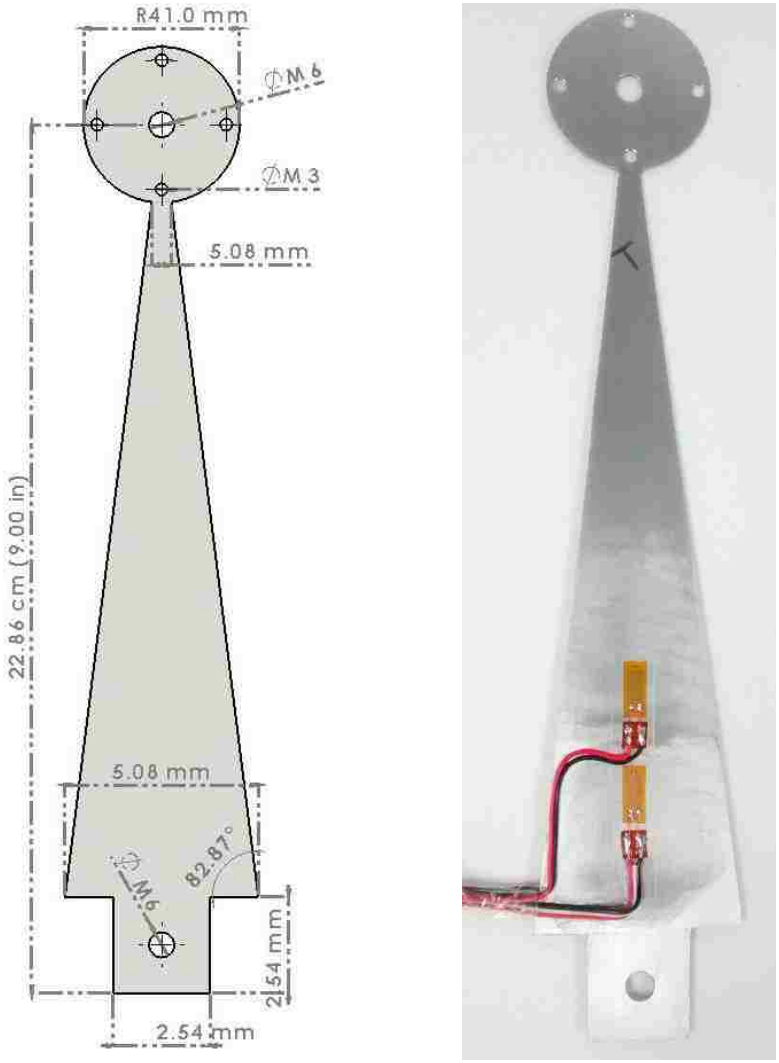

Figure 4.7. First Iteration 0.032-in & 0.050-in Beam

The first-iteration beam was used successfully with the small Syma S107 UAV on the 0.032-in beam. Several tests were run with no issues with results shown in Section 6.2.4. However, while testing the 0.050-in beam with the large motors, excessive torsion caused a permanent failure of the beam as seen in Figure 4.8. The combination of the

Turnigy Park300 1080-K_v motor and the 7x6 propeller caused the end of the beam to twist uncontrollably, resulting in the motor mount section being ripped from the rest of the beam. The damage was permanent and the beam had to be redesigned to counter the problem. The orientation of the strain gauges makes them insensitive to the torsion so torsion data was not recorded.

Figure 4.8. 0.050-in Beam Torsion Failure

4.3.2. Second Iteration. Because of the permanent torsion failure, the beam to test the individual motors was redesigned. The second iteration beam can be seen in Figure 4.9. This beam had a wider connection point between the beam and the motor mount area to keep the beam from twisting. The connection point is curved to prevent the stress buildup at the sharp corners of the first beam. The thickness was also increased from 0.050 in to 0.062 in to decrease the sensitivity to strain. The thrust potential of the larger propellers was underestimated and the 0.050-in beam was too thin. The beam generated an excess amount of strain on the beam that went over the maximum thrust range and resulted in the microcontroller clipping the data. A lower sensitivity setting on the strain indicator had to be used in these cases. Using a thicker beam allowed the strain indicator and microcontroller to record the full range of thrust at the maximum sensitivity setting. The 0.062-in beam was calibrated and each motor and propeller was tested on the new beam.

Table 4.1 gives the thrust range for each of the three beams in newtons and grams as well as the resolution in newtons and grams per ADC value. The range was determined by finding the thrust that would give an ADC value of 1023. Resolution was found by dividing the maximum thrust value by 1023, the total number of ADC values.

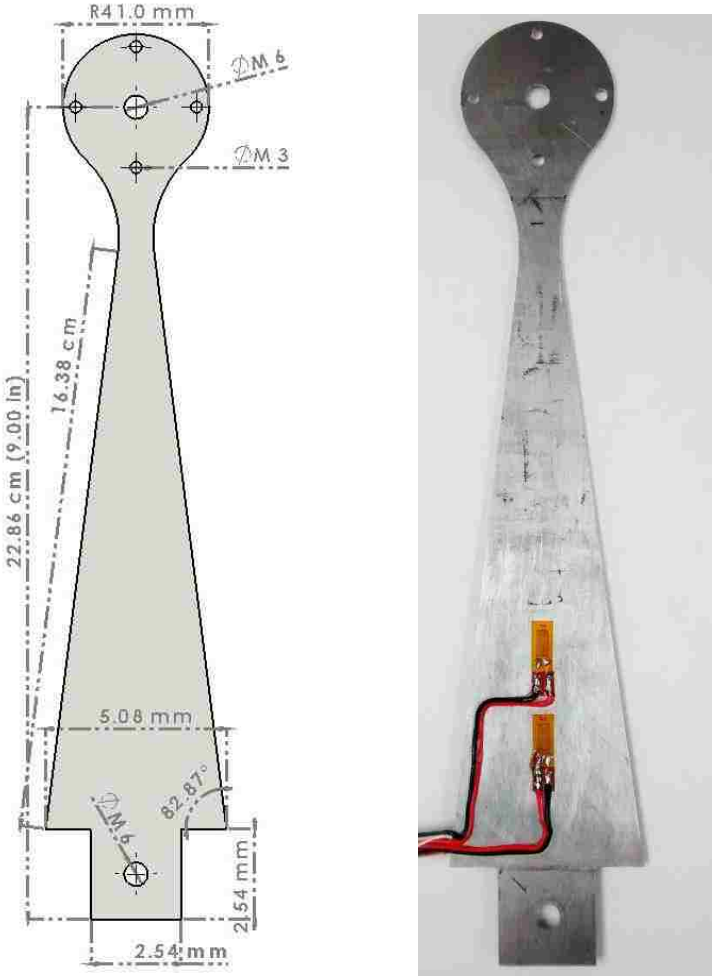

Figure 4.9. Second-Iteration 0.062-in Beam with Side Support

Table 4.1. Cantilever Beam Thrust Range & Resolution

Beam	Thrust (g)	Thrust (N)	Resolution (g/ADC value)	Resolution (mN/ADC value)
0.032”	0 – 222	0 – 2.17	0.217	2.13
0.050”	0 – 560	0 – 5.48	0.547	5.36
0.062”	0 – 1061	0 – 10.40	1.04	10.17

4.3.3. Vibration Damping. The 0.062-in beam produced a significant number of vibrations compared to the other two beams that in a couple cases prohibited testing. To lessen the vibrations caused by the motor and propeller, rubber washers were added on either side of the cantilever beam as shown in Figure 4.10. These mounting components reduced the vibrations enough to complete testing on the 1200-K_v and 980-K_v motors. However, the 1080-K_v motor, the same motor that caused the torsion failure of the 0.050-in beam experienced strong enough vibrations on the 8x4 and 8x6 propeller tests that the tests could not be completed. During these tests, the motor and propeller caused the beam to vertically vibrate at its resonant frequency. The center of the beam flexed about 0.5 in at an approximate frequency of 100 Hz. Further details can be found in Section 6.1. Attaching a 100-g weight to various points of the beam to alter the resonant frequency did not decrease the vibrations.

Figure 4.10. Rubber washers

4.4. INSTRUMENTATION

The instrumentation module was designed to control the motor consistently over multiple tests and record all data for further analysis. The module included a voltage and current sensor, an input for the strain gauge indicator, and controls for the electronic speed

controller (ESC). A full schematic of the module can be found in Appendix D. Manufacturers' data on all parts used can be found in Appendix A. Figure 4.11 shows a block diagram the instrumentation module. Figure 4.12 shows the inside and outside of the completed module.

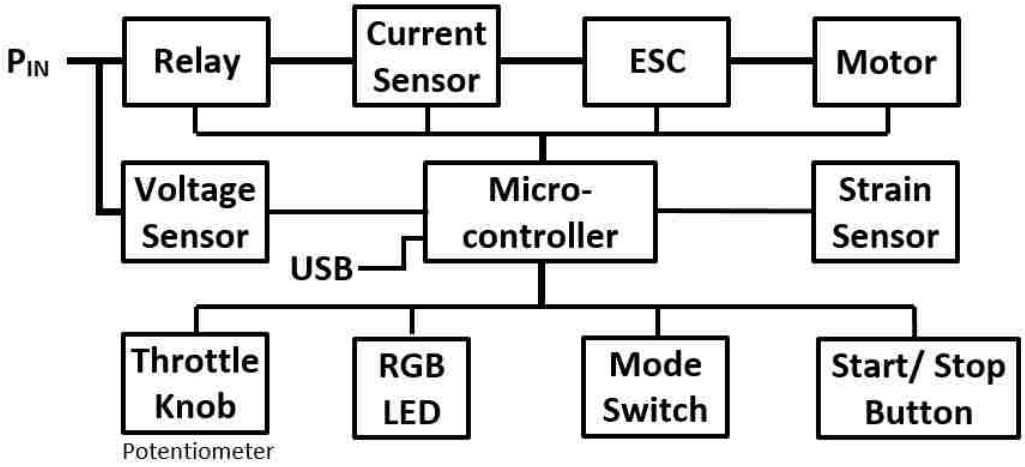

Figure 4.11. Instrumentation Module Block Diagram

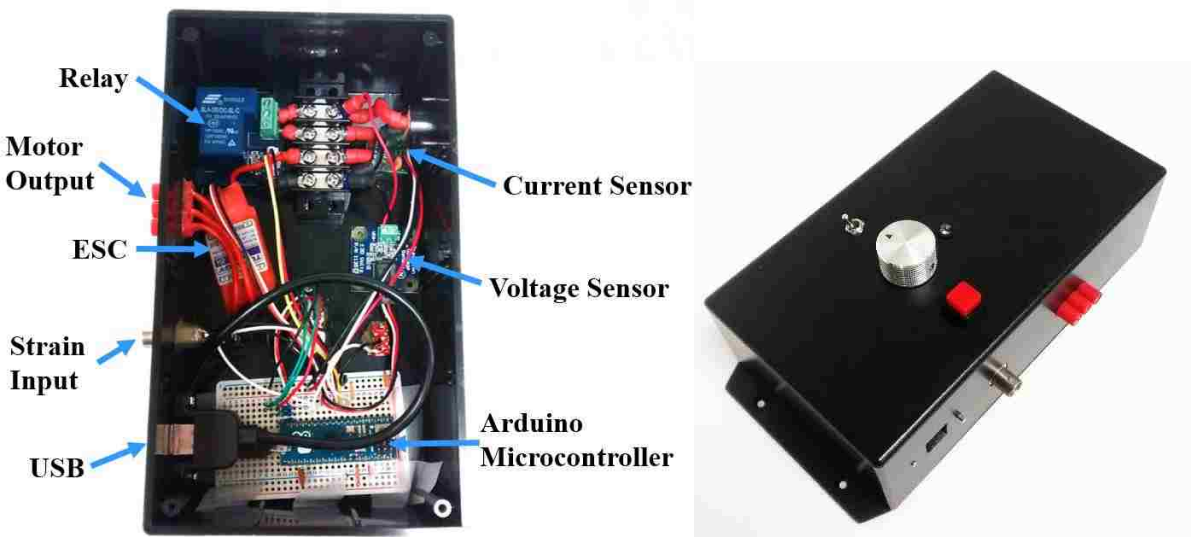

Figure 4.12. Instrumentation Module Overview

4.4.1. Operation of Module. The module has two operating modes, manual and automatic. In manual mode, the speed of the motor can be controlled by the potentiometer knob and was intended for testing functionality and for use in the wind tunnel. In automatic mode, the microcontroller will run the uploaded test profile and automatically record and send data through the serial USB connection.

4.4.2. Microcontroller. The microcontroller's function is to receive and display input, to and from the user, to control the brushless DC motor through the ESC, gather data from the voltage, current, and strain sensors, and transmit the data through a USB connection to a host computer. To meet all these requirements, the Arduino Micro microcontroller, shown in Figure 4.13, was chosen. It is based on the ATmega32U4 architecture developed by Atmel which allows for native USB functionality without relying on a separate processor [21]. Combining these functions allows for faster serial communications and lower costs. The processor also supports 20 digital input/output (I/O) pins with seven of these having pulse width modulation (PWM) capabilities and 12 total analog inputs [22]. These specific specifications are necessary due to the modules' need for four analog inputs, two digital inputs, four PWM outputs, and one digital output for a combined 11 I/O pins.

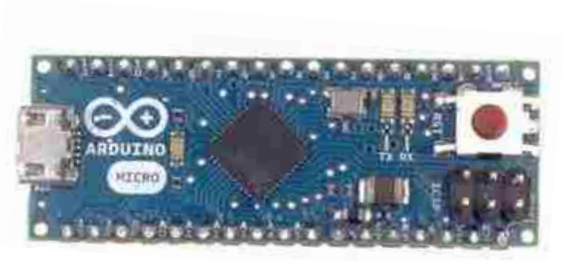

Figure 4.13. Arduino Micro Microcontroller

Compatibility with the Arduino IDE (integrated development environment) software was a large benefit that was considered when selecting a microprocessor. The IDE is a C-based environment with a large user base that provided function libraries that did not have to written from scratch. The microcontroller also has a small footprint which

enabled the overall module to be more compact. The microcontroller was also inexpensive (less than U.S. $20), allowing for replacements to be purchased had they been necessary.

4.4.3. Voltage Sensor. A voltage sensor was included in the instrumentation module to provide information on the supply voltage during tests. The voltage data can be used to determine the health of the battery and ensure the battery and motor are not damaged by prolonged testing. Combined with a current sensor, the voltage data was used to calculate the instantaneous power of the motor. The sensor was required to have a minimum sensing range of 0 to +15 VDC to accommodate the required voltage range of the ESC.

The Phidgets precision voltage sensor, shown in Figure 4.14, is able to convert a ±30 VDC range to a linearly proportional 0 to 5 VDC range which can be read by the microcontroller. A commercial voltage sensor constructed of operational amplifiers was chosen over using a simple voltage divider circuit due to its increased accuracy, linear output, and ability to interface with the ADC without additional circuitry. The voltage sensor includes a 2.5 V offset to account for the negative portion of the sensing range. When reading a supply voltage of 0 V, the microcontroller ADC reads 2.5 V.

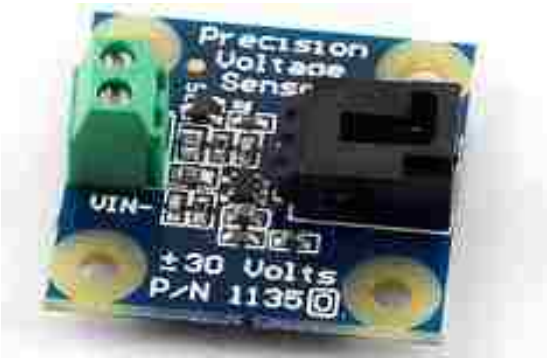

Figure 4.14. Phidgets Precision Voltage Sensor

4.4.4. Current Sensor. One of the goals of the instrumentation module was to calculate the power produced by the motor during tests and involved using a current sensor combined with the voltage sensor. The sensor needed to measure DC current up to 20 A and have a compatible interface to the microcontroller. Because of the high currents

involved, a Hall effect sensor was desired and ultimately the Allegro ACS715 integrated circuit was chosen. The breakout board for the 30-A version, designed by Pololu, was purchased for US $9.95 and can be seen below in Figure 4.15.

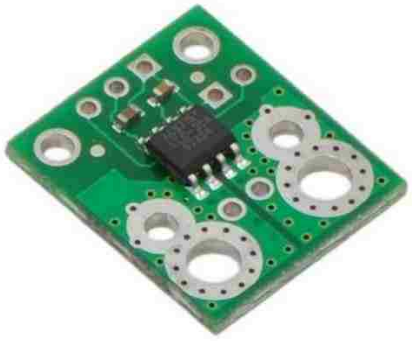

Figure 4.15. 30-A Breakout Board for Current Sensor

The Hall effect sensor was chosen over another typical current method, a shunt resistor. The ACS715 output signal is electrically isolated from the high sensed current, increasing safety and lowering the risk of damage to the microcontroller. Hall effect sensors also have no added resistance to the wire, unlike a shunt resister, and therefore consume less power. The ACS715 was placed on the high side, or between the power source and the ESC and can detect any downstream failures such as a short circuit and shut off the module before damaging the components.

4.4.5. ESC. The ESC, a Turnigy Plush 25A ESC, was used to control the brushless DC motor. The original interface was a standard radio transmitter that transmitted to a receiver connected to the ESC. The radio transmitter was no longer operational and the Arduino microcontroller was used to bypass the receiver and directly reproduce the signals necessary for the ESC. The ESC has a max current rating of 25 A which is well above 17 A, the maximum current for 1200-K_v motor. Figure 4.16 shows the ESC used in this project.

The ESC requires a PPM (pulse position modulation) signal as an input, similar to how a servo is positioned. The Arduino IDE contained a servo library to run the ESC with

minimal code. A common ground and a signal connection are connected between the Arduino and the ESC. No VCC power (5 V) is connected between the Arduino and the ESC because the ESC was connected to the supply voltage and the connection would result in damage to the microcontroller. This connection is normally used to power the wireless receiver which was not necessary for this setup. The three phase output of the ESC was run to a three terminal female bullet connector that connected to the exterior of the module. Each brushless motor was connected to a male bullet connector, allowing the motors to be swapped easily.

As a safety feature, the ESC cannot be activated until the correct startup sequence has been inputted. This feature ensures that the motor cannot accidentally be started if the throttle joystick was not left at minimum. The startup procedure can be found in Appendix B. After several attempts, the ESC was successfully armed with both an AC/DC adapter, as well as the battery pack. The ESC was calibrated to the microcontroller as shown in Section 5.1.2.

Figure 4.16. Turnigy Plush 25 A ESC

4.4.6. Power Source. The ESC and motor determined the input power requirements for the instrumentation module. The microcontroller is powered independently off the USB connection. The ESC requires a voltage range between 5.6 VDC to 16.8 VDC and can handle up to 25 A. Two power sources were chosen, a wall outlet

AC/DC adapter to run during setup and a 3-cell lithium polymer (LiPo) battery to run during static and wind tunnel tests.

The AC/DC adapter was an old laptop charger and has a voltage output of 12 VDC and a max current output of 4.16 A. A wall adapter was useful for running multiple tests of the functionality of the each component of the instrumentation module and during troubleshooting. A battery would need to be recharged multiple times when running many tests throughout the day. The adapter is also a more stable power source which is ideal for running calibration tests because of the voltage drop as power is consumed. This model was chosen for its high output current and because the output voltage is in the middle of the acceptable voltage range. When running the startup procedures, the ESC checks the voltage of the power supply to determine the number of cells the battery contains. Because, the voltage is very similar to a 3-cell battery, the ESC can be tricked into accepting the adapter. The connector to the adapter was replaced with an XT60 connector to be compatible with the battery.

A Turnigy 3-cell 2200-mAh LiPo battery, designed for hobbyist applications, was chosen as the primary power source for the system. These batteries are designed to interface directly with the ESC using the XT60 connectors. The high current discharge of the battery enables the motor to run at full power. A Turnigy battery charger was purchased to charge the battery. The special charger is necessary to ensure all three cells are balanced, which prolongs the life of the battery. Figure 4.17 shows an image of the battery and charger used in the setup.

4.5. SOFTWARE

Four programs were written for the Arduino Micro microcontroller including ESC throttle calibration, sensor calibration, static testing, and wind tunnel testing. The full code for all four programs can be found in Appendix C. These programs were written in the Arduino IDE (version 1.6.5) [22]. One of the main advantages to using the Arduino microcontroller was the large user base, and many prebuilt libraries. Several such libraries were used to expedite the coding process.

Figure 4.17. Turnigy 2200-mAh Battery and Charger

4.5.1. ESC Throttle Calibration. This program calibrates the ESC throttle position. The throttle is controlled by sending a servo value from 0 to 255. The program enters the throttle calibration mode of the ESC by transmitting the maximum throttle value. Once this value is accepted (one beep, about 2.5 seconds), the low throttle position is sent. The ESC beeps twice if the data is accepted. The entire program is written in the setup function because it only needs to be run once.

4.5.2. Sensor Calibration. This program was written to calibrate the strain, voltage, and current sensors used in this project. Only one sensor can be calibrated at a time and is chosen by changing the sensorPin value. The data is sampled at a specified rate and transmitted over the serial connection to be recorded by the CoolTerm software.

4.5.3. Static Testing. Software for the static testing portions of this thesis was designed to meet the following criteria. The software must connect with the ESC to run the motor, read data from the voltage, current, and strain sensor, transmit the data to the host computer, and communicate statuses to the user. An overview of the program is shown in Figure 4.18. The program was split into three sections: setup, loop, and interrupt service routine (ISR). The setup section programs all the necessary settings into the microcontroller including mode selection and contains code to activate the ESC. The voltage drop of the battery was negligible over the 15 second test interval and only needs to be sampled at the beginning of the test in the setup section. The loop section contains

code to update the ESC speed and to test for safety conditions. The ISR section contains code to sample and transmit the sensor data to the host computer. The ISR is activated every millisecond by the MsTimer2 library. The MsTimer2 library, written by Javier Valencia, utilizes the Arduino Micro microcontroller's Timer 2 to run the ISR at a prescribed interval [23]. The section also contains a counter that allows the loop section to increment through the test profile.

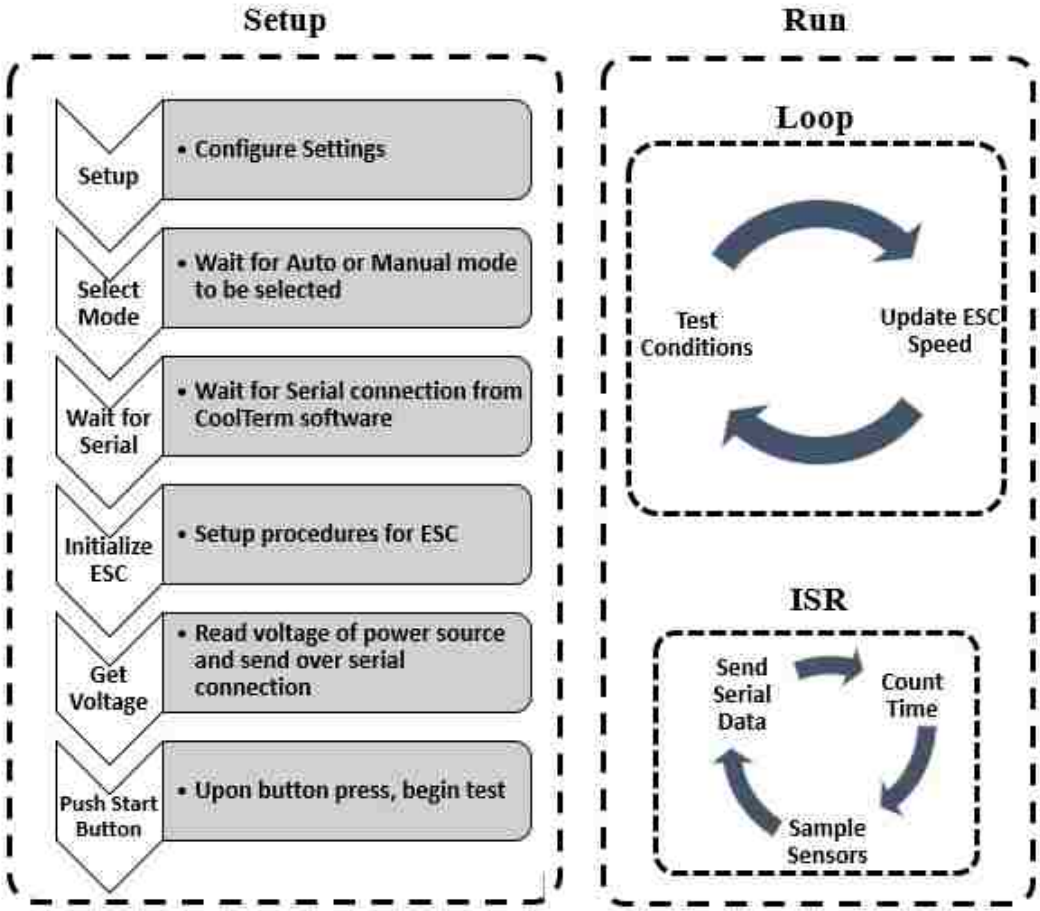

Figure 4.18. Static Testing Software Overview

The software incorporates two separate modes that are selected by the user using a toggle switch during the setup section. The first is manual mode and is used for preliminary testing. This mode allows the user to control the speed of the motor manually using the

potentiometer knob. The second mode is automatic mode which allows the speed of the motor to be controlled by the microcontroller. The automatic mode is used for final static testing because the same test profile can be run for each test ensuring consistency for accurate comparisons.

The ESC accepts the throttle position signal as a servo input (pulse position modulation). The ESC signal is normally created by the standard Servo library which is included by default in the Arduino IDE software. However, the Servo library uses interrupts to generate the very specific timing requirements for the signal. The static testing program also needs a timer interrupt (MsTimer2) to sample data at a consistent rate. These two interrupts conflict because when an ISR is entered, all other interrupts are disabled until the current ISR is completed. The Adafruit_TiCoServo library uses a dedicated timer/counter pair to send the servo signal rather than using interrupts [24]. Once configured, zero instruction cycles are spent on the task. This method allows both the servo and timer interrupt to be used simultaneously.

4.5.4. Wind Tunnel Testing. The code for the wind tunnel testing is very similar to the static testing code, but with a few changes to tailor the code to the testing environment. The automatic test mode was removed because the device only needed to be operated manually. The tests needed to run for an extended period of time, therefore the battery voltage needed to be recorded at periodic intervals rather than once at the start of the test. The throttle position of the motor was also recorded.

4.5.5. Data Recording. The USB connection on the Arduino microcontroller was used to send serial data to the host computer. This data is recorded on the host computer by the CoolTerm software. The speed of this data transmission is the bottleneck in collecting and sending measurements. Therefore, the higher the baud rate, the better. Sensor data from the current sensor, voltage sensor, and strain sensor were formatted to be easily converted to a Microsoft Excel document once saved as a text file on the host computer. The raw values were separated by commas and formatted as a string. The string is then sent to the CoolTerm software. No manipulation is done to the ADC values on the microprocessor. Therefore, each value will be a 10 bit value between 0 and 1023. The procedure for capturing the incoming data using the CoolTerm software is shown in Appendix B.

4.5.6. Safety Features. Several safety features have been designed to prevent injury and damage to the equipment. When in manual mode, the motor cannot be activated until the potentiometer is first brought to the minimum throttle position. This feature ensures that the motor is only activated when intended. The start button also functions as an emergency stop that when pressed, deactivates the relay, cutting power to the motor. The button works in both manual and automatic modes. The microcontroller periodically checks the source voltage for under or over voltage conditions. If the voltage is out of range, the relay will be tripped and power disconnected from the motor which can prevent damage to the ESC.

5. METHODOLOGY

This chapter serves to explain the methods behind how the experimental tests were prepared and conducted. The methods include calibration of sensors and equipment and procedures for the static and wind tunnel tests. Three cantilever beams with thicknesses of 0.8128 mm (0.032 in), 1.27 mm (0.050 in), and 1.5748 mm (0.062 in) were calibrated. All three were used during static testing; the 0.062-in thick beam was used in the wind tunnel test.

5.1. CALIBRATION

To convert strain readings to thrust each of the three cantilever beams required separate calibration due to their different thicknesses. The ESC (electronic speed controller) was calibrated to the desired output of the microcontroller. There are many different types of error that can be introduced into a system. For this project, the most significant errors stem from strain gauge and instrumentation inaccuracies. Errors in these devices were decreased through calibration.

5.1.1. Strain Indicator Calibration. Before any other testing or calibration could begin, the strain indicator unit needed to be calibrated. By using the internal 5000 $\mu\epsilon$ shunt resistor, the gauge factor (GF) is compensated for the lead wires. All the lead wires are the same length (24 in) therefore, the calibration only needed to be performed once. The calibration procedure can be found in Appendix B. The result of the calibration changed the GF from 2.085 to 2.084. The new GF was used with all three beams.

5.1.2. Strain to Thrust Calibration. A gram-force (g_F) is equal to the magnitude of the force exerted by one gram in a standard Earth gravity of 9.806 m/s^2. The SI unit of force is the newton (N) is defined as the force necessary to accelerate a one kg object one m/s^2. Converting between these two units is shown in Equation 5-1.

$$Newtons = grams * 0.009806 \tag{5-1}$$

A precision weight set was used to calibrate each cantilever beam to establish the relationship between force and strain. The relationship is dependent on the thickness of

each beam with the thinner beam being more sensitive, i.e. producing more strain with the same force. From the data, trendlines were created for each dataset to find the calibration equations. For the 0.032-in thick beam, Equation 5-2 shows the conversion from strain to thrust in newtons. These strain values are for the full-bridge Wheatstone circuit. Likewise, Equations 5-3, 5-4 show the conversion for the 0.050-in, and 0.062-in thick beams respectively.

$$\text{Thrust (N)} = [0.0399 \times \text{strain } (\mu\epsilon) - 0.0136] \times 0.009806 \quad (5\text{-}2)$$

$$\text{Thrust (N)} = [0.1006 \times \text{strain } (\mu\epsilon) + 0.015] \times 0.009806 \quad (5\text{-}3)$$

$$\text{Thrust (N)} = [0.1908 \times \text{strain } (\mu\epsilon) - 0.0691] \times 0.009806 \quad (5\text{-}4)$$

5.1.3. ESC Calibration. The ESC required calibration before it could be effectively controlled by the microcontroller. The calibration involved programming the high and low throttle positions. To simplify the controls and calculations, the design had 100 positions of throttle control. Since the microcontroller has eight bits of PPM (pulse position modulation) output, the 100 positions could be located anywhere between zero and 255. A maximum throttle position of 125 and a minimum throttle position of 25 were chosen.

5.1.4. Instrumentation. All the sensors in the module needed calibration to produce accurate results. This section focuses on the calibration steps needed for each device. The microcontroller only outputs raw ADC values to the host computer. All calibrated equations shown in this section were applied on the host computer in Microsoft Excel.

5.1.4.1 ADC. Without calibration, the ADC on the microcontroller will experience both offset and gain errors. The setup included a precision power supply, oscilloscope, and the coaxial splitter. The output from the power was split off to verify the voltage. When using the external 5.0 V USB voltage, the ADC produced significant offset and gain errors. However, when using the 2.56 V internal reference voltage, the ADC produced no significant offset or gain errors.

The maximum sample rate for the Arduino ADC was tested to determine the sample rate to use for data collection during tests. A signal generator was attached to the analog

input of the microcontroller as shown in Figure 5.1. A 25-Hz sine wave with a peak to peak voltage of 2.000 V and a 1.000 V offset was sent through a coaxial cable to the ADC and sampled at regular intervals as determined by the external timer interrupt. The MsTimer2 library was used to trigger the interrupt at the correct interval. The code was sampled at several intervals to see if the sine wave was still discernable. The highest sample rate possible with the MsTimer2 library was 1000-Hz. As shown in Appendix E, the 1000-Hz sample rate is consistent and accurate.

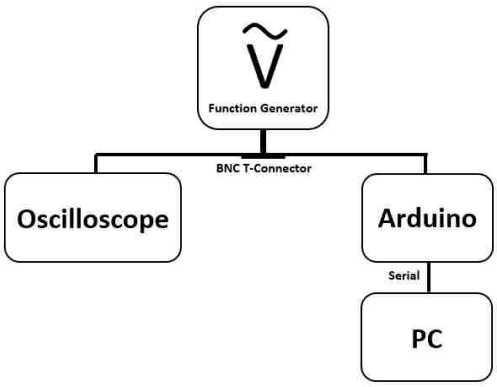

Figure 5.1. ADC Calibration Setup

5.1.4.2 Communications. The bottleneck in the Arduino programs is the sensor data transmission over the serial connection in the ISR. It is therefore imperative that the transmission occurs as fast as possible. The max baud rate that the CoolTerm recording software can record at is 230400 bps (bits per second). The Arduino microcontroller was tested to see whether it can transmit accurately at this baud rate. A precision voltage of 1.000 V was sent to the ADC to hold a steady value. The MsTimer2 library was used to get a sample rate of 1000-Hz as shown in the above section. The ADC value was then transmitted at 230400 bps for about three seconds and saved with the CoolTerm software. As shown in Appendix E, the ADC value held steady over the entire sampled range only varying by one ADC value. The data confirms that the selected baud rate is accurate and can be used for testing.

5.1.4.3 Voltage sensor. The voltage sensor had to be configured to work with the 2.56 V internal reference voltage. Because the voltage sensor has an offset of 2.50 V when reading zero volts, the polarity needed to be reversed. A precision voltage supply was used to supply the voltage to the sensor. The ESC was disconnected during the test to prevent interference and possible damage. When the voltage is increased, the ADC decreased as a result of the reverse polarity. The calibrated equation that relates voltage to the ADC value is shown in Equation 5-6 and found by taking a trendline of the data in Microsoft Excel.

$$\text{Voltage} = -0.0367 \times (\text{ADC value}) + 37.082 \tag{5-6}$$

5.1.4.4 Current sensor. The current sensor was calibrated by attaching a constant current source to the current sensor. A multimeter was also placed in series with the sensor to verify the current source's output, as shown in Figure 5.2. The current was varied from 50 mA to 1000 mA and the sensor values were recorded for ten seconds using the CoolTerm software. The calibration data when sensing 1000 mA can be seen in Appendix E and shows that the noise of the sensor is approximately four ADC values. For calibration, the entire ten second sample range was averaged to find an approximate value. The averaged data can be seen in Appendix E. Taking a trendline of the data yielded the calibration Equation 5-7.

$$\text{Current (mA)} = 18.168 \times (\text{ADC value}) - 3815.6 \tag{5-7}$$

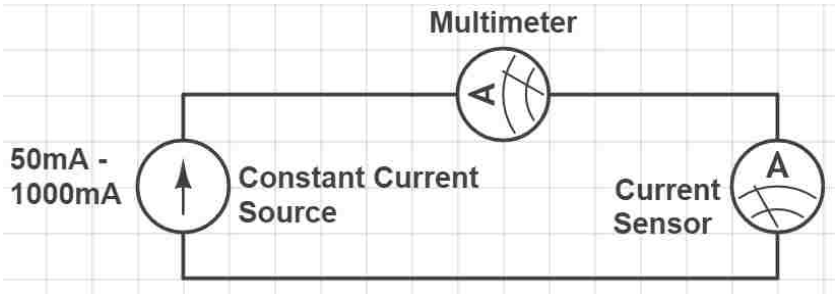

Figure 5.2. Current Sensor Calibration Setup

5.1.4.5 Strain indicator output. The output from the strain indicator needed to be calibrated to the corresponding ADC values. The strain indicators sensitivity was set to maximum to give a theoretical sensitivity of 440 μV per με according to the manual [12]. A strain gauge reference model was used to provide the strain data for calibration. A micrometer at the end of the beam was rotated to place a constant stress on the beam. Multiple readings were taken and can be seen in Appendix E. The calibration equation was found by taking a trendline of the data in Microsoft Excel and can be seen in Equation 5-8. The 15 με offset is produced by the strain indicator unit.

$$\text{Strain } (\mu\epsilon) = 5.4218 \times (\text{ADC value}) + 15.789 \tag{5-8}$$

5.2. TESTING

Multiple tests were run on the three motors and five propellers a static environment. One motor and two propellers were tested in a dynamic (wind tunnel) setting. This section details the procedures followed during the static and dynamic tests to ensure the consistency of each test.

5.2.1. Static Testing. The majority of the testing was done in a controlled static testing environment with no wind and consistent temperatures which allowed for high repeatability between tests. As shown in Figure 5.3, the selected beam was mounted horizontally to the metric table between two half-inch aluminum blocks. The mount ensured that only the motor end of the beam was movable. The beam was suspended off the edge of the table to reduce oscillations caused by the turbulent air between the rotor and table.

5.2.2. Wind Tunnel Testing. To simulate real world conditions, the second-iteration 0.062-in cantilever beam was placed in a subsonic wind tunnel. The wind tunnel used was a subsonic wind tunnel with a maximum speed of 70 mph and a square cross section of 18 in. Figure 5.4 shows the setup with the cantilever beam and motor in the wind tunnel. The tunnel was an open-loop wind tunnel, with air being pulled in and exhausted out ducts in the ceiling

The procedure for the test included securing the beam with the motor to the clamp in the wind tunnel pull-out section as shown in Figure 5.5. The section was then secured

inline with the wind tunnel with the heights of each section matched. A pitot-static tube was used to measure the air velocity inside the wind tunnel and was calibrated to the outside temperature and pressure at the time of the tests. Before each test, the tunnel was ran for a few minutes to equalize the temperature. The strain indicator was zeroed to 2000 με to allow the recording of positive and negative strains values.

For each test, the wind tunnel air velocity was incremented by 3.048 m/s (10 ft/s, 6.8 mph) until 70 ft/s was reached. At each speed, the motor was activated and strain, voltage, and current values were recorded. The motor was turned off while adjusting the speed of the wind tunnel to preserve the battery. The first test completed was a drag test. This test included measuring the drag of the thrust stand and 1200-K_v motor at each speed. No propeller was on the motor when running the drag test. The second test utilized the 7x6 propeller and the third test used the 5x5 propeller. The strain values were averaged over each test to obtain an average thrust at the required speed.

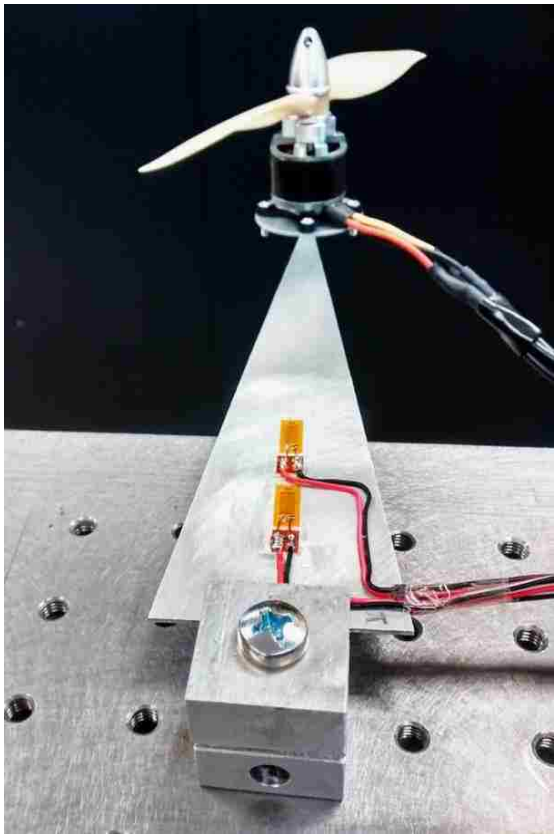

Figure 5.3. Static Testing of Individual Motor

Figure 5.4. Cantilever Beam and Motor inside Wind Tunnel

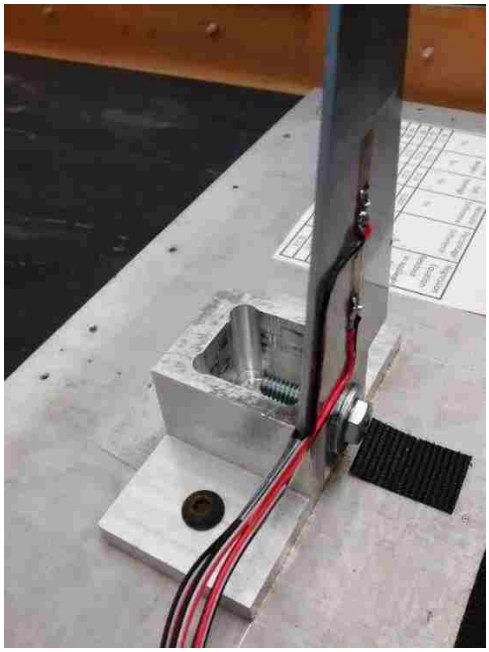

Figure 5.5. Wind Tunnel Beam Mount

5.2.3. Motor Testing. Each of the three individual motors and the small UAV were static tested. The small UAV was mounted to the 0.032-in thick beam as shown in Figure 5.6. Because it used a wireless receiver to receive throttle instructions, the motor could not be controlled by the instrumentation module and only strain data could be obtained. Two different tests were run; the first test measured the thrust of the two main rotors. The wireless transmitter was used to slowly increase the throttle from the minimum to the maximum position over several seconds. The throttle was held at maximum, then slowly lowered to the minimum position over several seconds. The second test measured the thrust produced by the tail rotor. The throttle was to maximum then the directional control stick was then pulsed up to activate the tail rotor.

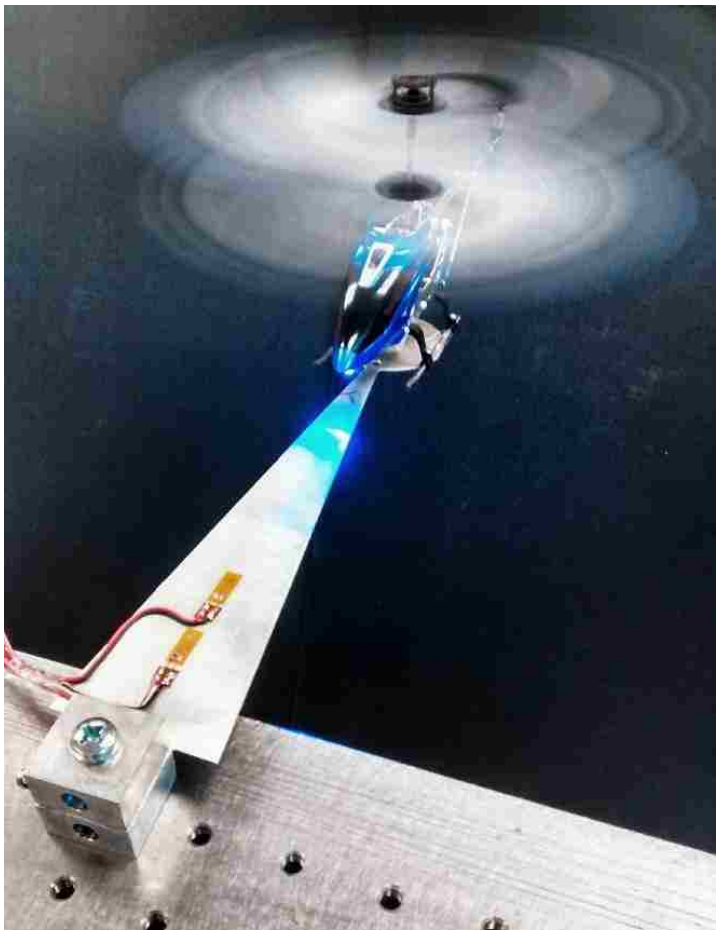

Figure 5.6. Small UAV Test

The three individual motors were mounted on the 0.050-in and 0.062-in thick cantilever beams. All three motors utilized the same mount and each motor was tested with the five propeller sizes for comparison. The instrumentation module was used for the each test in automatic mode and controlled the ESC and collected data simultaneously. For each test, the ESC would accelerate the motor from minimum to maximum speed over five seconds, hold the maximum speed for five seconds, and then decelerate the motor from maximum to minimum speed over five seconds. With each motor test utilizing identical test profiles, accurate comparisons could be made.

6. RESULTS & DISCUSSION

This chapter gives the results and analysis for the data collected during testing. The cantilever beams were tested to verify the simulations and strain gauges. Results from all the static and wind tunnel tests were also tabulated. The data collected from the microcontroller was raw ADC values and all calibrations and unit conversions were completed in Microsoft Excel.

6.1. CANTILEVER BEAMS

This section gives the results of simulations and verifies the calibration equations of the 0.032-in, 0.050-in, and 0.062-in cantilever beams. The resonant frequencies of the three beams were also found after testing and produced excessive vibrations in the thrust results.

To verify that the designed beam shape was a constant stress beam, each of the four gauges was measured separately under a constant applied load of 100 g. Table 6.1 gives the results of the test. As shown, each gauge gave a very similar value. Gauge 1 and 3 only varied by 2 $\mu\epsilon$ and were both located on the top side of the beam to measure tension. Similarly, gauge 2 and 4 only varied by 1 $\mu\epsilon$ and were located on the bottom to measure compression. These results verify that the beam was a constant-stress beam.

Table 6.1. Individual Strain Gauge Test

Strain Gauge	Strain ($\mu\epsilon$)
1	135
2	- 130
3	137
4	- 131

The 0.032-in, 0.050-in, and 0.062-in beam were compared to verify the calibration equations using the 1200-K_v motor. The 4.5x4.5 propeller was used for the test because it was the only propeller that was in the 0.032-in beam thrust range. Ideally, after calibration

the beams should read identical thrust output. Figure 6.1 shows a graph with the thrust results for the three beams. The three beams produced almost identical results with the 0.032-in beam producing the least amount of vibrations.

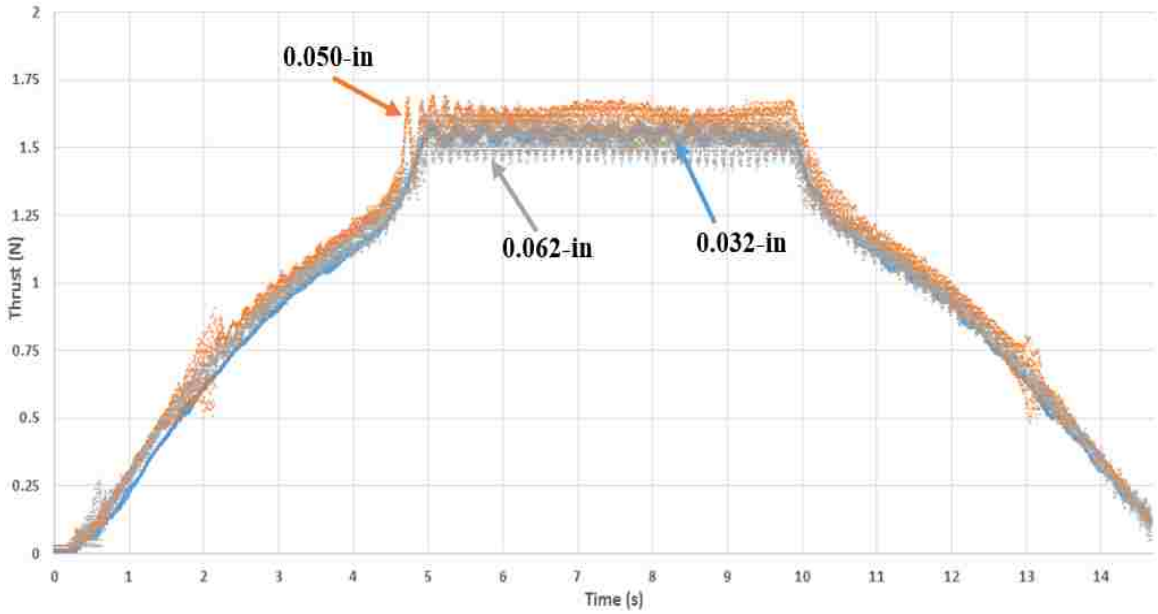

Figure 6.1. Cantilever Beam Thrust Comparison

6.1.1. Simulations. Each cantilever beam was simulated in SolidWorks to predict the strain of a known weight, which can be converted to thrust. The simulations aided in the design of the beams, especially the thickness, as the thrust range was able to be determined ahead of time. The simulations of the 0.032-in and 0.062-in beam are shown in Figure 6.2. A 100 g (0.9806 N) weight was simulated at the end of the beam at the center of the motor mount. The strain in the constant-stress region was then found and compared to the measured results, shown in Table 6.2. To match the strain of the full-bridge Wheatstone circuit, the simulated strain was multiplied by four, one for each gauge. The simulated strain for each beam was less than the measured strain by about 2 %.

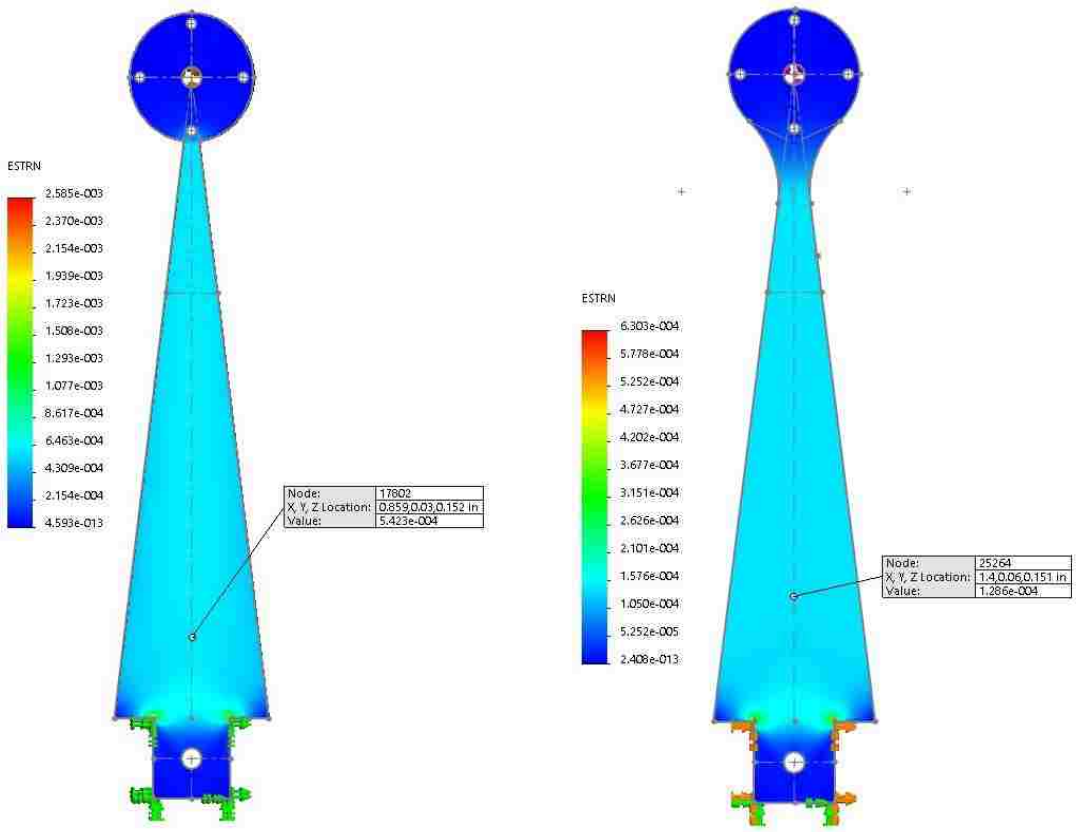

Figure 6.2. 0.032-in & 0.062-in Beam Simulations

Table 6.2. Beam Simulation Results

Beam	Simulated Strain (με)	Measured Strain (με)	Percent Error (%)
0.032-in	2169.2	2202	1.49
0.050-in	964.8	995	3.04
0.062-in	514.4	522	1.48

6.1.2. Resonance. Both the 0.050-in and 0.062-in beams experienced excessive vibrations at specific throttle positions. The vibrations were caused by the motor rpm hitting the resonant frequency of the beams. The frequency was different for the different beams with the thicker, 0.062-in beam having a higher frequency than the 0.050-in beam. The 0.062-in beam also had a small resonant frequency at a low throttle position. Figure 6.3 (1200-K_v 5x5 propeller, 980 K_v 7x6 propeller) show the excess vibrations at the resonant spots of the 0.050-in beam and the 0.062-in beam. The vibrations also occur as

the throttle was decreased, confirming the location. Higher resolution images of the resonant locations can be found in Section 6.2.1.1.

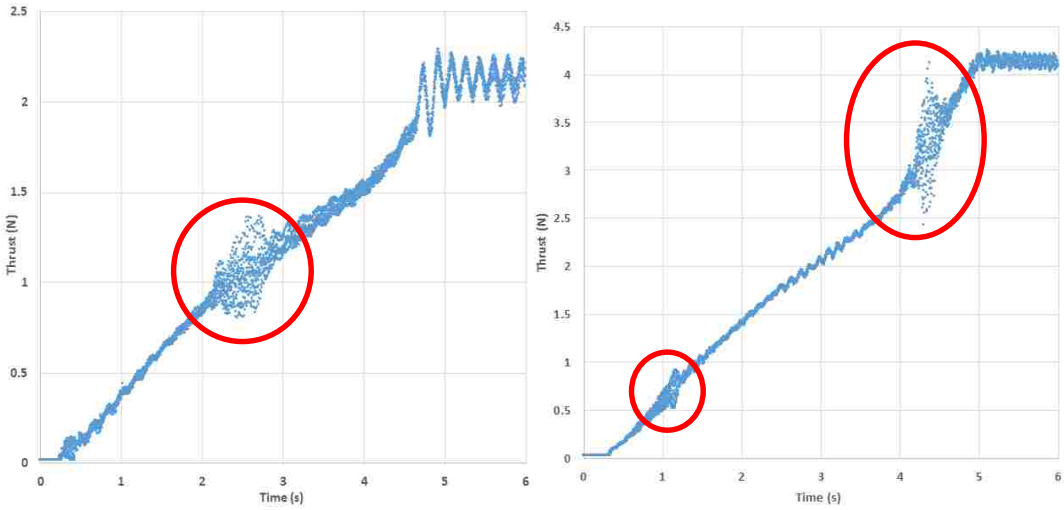

Figure 6.3. 0.050-in Beam & 0.062-in Beam Resonance

Using the Fast Fourier Transform (FFT), the 0.050-in beam resonance frequencies of the 0.050-in beam were found to be 157 Hz and 471 Hz. For the 0.062-in beam, the upper section had resonance frequencies of 70 Hz and 488 Hz. The lower section experienced identical frequencies confirming the resonance frequency of the beam.

The 1080-K_v motor experienced excessive vibrations with all propellers unlike the other two motors. As shown in Figure 6.4 (1080-K_v 4.5x4.5 propeller), at the maximum throttle, the beam experienced large vibrations of a frequency of 407 Hz on the 0.062-in beam and was only slightly lower than the Figure 6.3, 488-Hz resonance point due to the lower weight of the motor. For an unknown reason, the 1080-K_v motor triggers the beam to vibrate at its resonant frequency with a much greater amplitude than the other two beams.

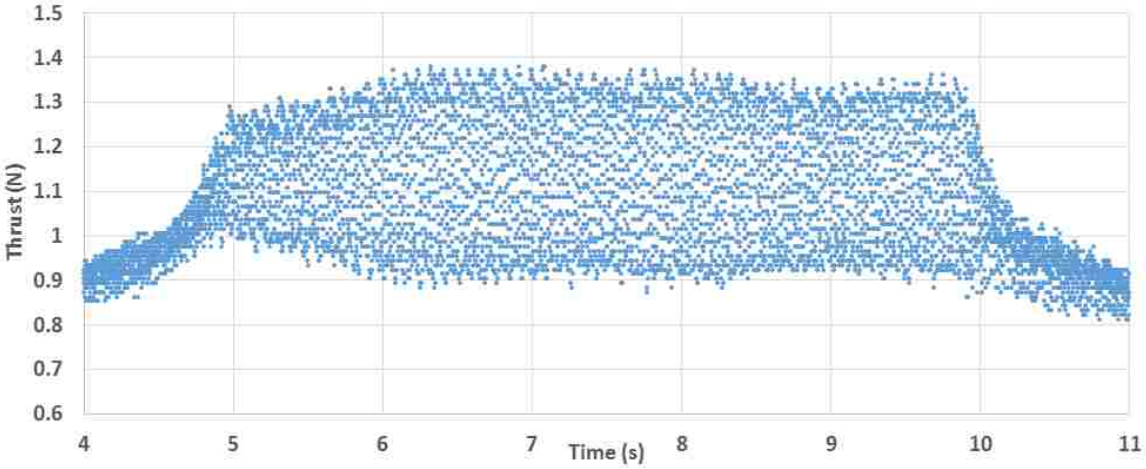

Figure 6.4. Resonance at Full Throttle

6.2. STATIC TESTING

This section gives the results of the static tests conducted on the metric table in the laboratory. Included in this section are detailed results of the 1200-K_v motor with the 7x6 propeller using the 0.050-in beam. All other results are given in Appendix F. For each test, the throttle was incremented from its minimum position to its maximum over five seconds, held at maximum for five seconds, then decremented to the minimum throttle position over five seconds. More details about the test setup can be found in Section 5.2.1. Data for the three motors and five propellers was captured successfully with the exception of two 1080-K_v motor results. Excessive vibrations caused these results to be incomplete.

6.2.1. Individual Motor Case. The 1200-K_v motor results were selected for detailed analysis due to smaller vibrations compared to the 980-K_v and 1080-K_v motors. In particular, the 1200-K_v motor with the 7x6 propeller and 0.050-in beam were chosen because this combination produced the cleanest results with the least vibrations and noise.

6.2.1.1 Thrust data. Thrust data was captured in real-time by recording the strain on the cantilever beam at a frequency of 1000 Hz. This data was then converted to thrust in newtons using the calibration equations. Figure 6.5 shows the thrust output during the entire test. The thrust increased almost linearly through the first five seconds. Once the throttle reached maximum at the five second mark, the thrust experienced underdamped oscillations while the beam settled to its equilibrium point. The thrust then decreases at the ten second mark, mirroring the first five seconds. Figure 6.6 shows a detailed view of

the underdamped oscillations upon reaching full throttle. The average thrust at maximum throttle was 5182 mN. Running at an estimated rpm of 11000, the manufactures data suggests the thrust should be 600.5 g (5888 mN). This thrust was higher than the experimental results which suggests a lower actual rpm. Lowering the rpms down to 10300, gives close to the experimental thrust output with a thrust of 5118 mN.

Figure 6.7 shows the resonant frequency region of the beam. Constructive interference caused the oscillations to get increasing large. By taking the Fast Fourier Transform, the beam was found to oscillate at a frequency of 47 Hz as the motor throttle was increased. Figure 6.8 zooms in on the data even further to show only 0.1 s of data at the resonant frequency of the beam. At this magnification, the individual oscillations could easily be seen along with smaller oscillations of approximately 240 Hz.

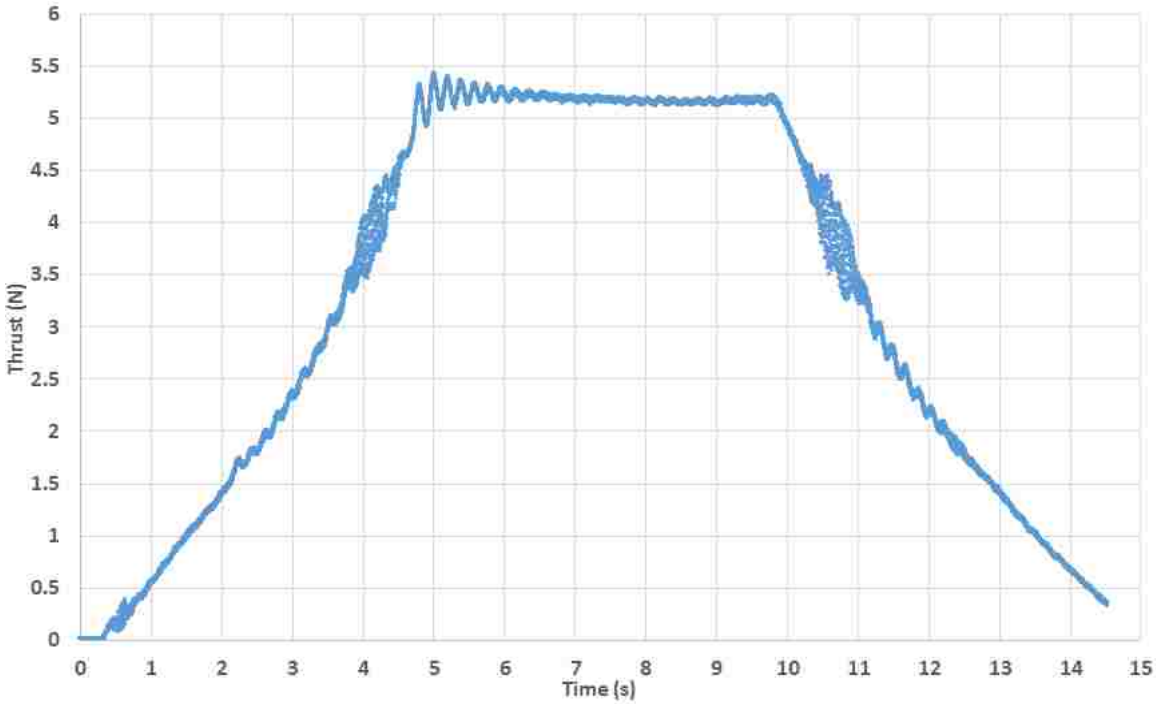

Figure 6.5. 1200-K_v 7x6-Propeller 0.050-in Beam Thrust Results

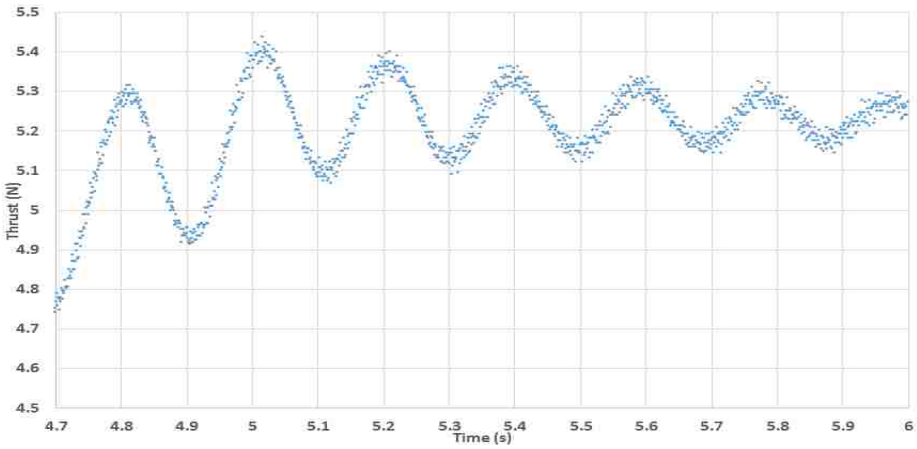

Figure 6.6. Oscillations at Full Throttle

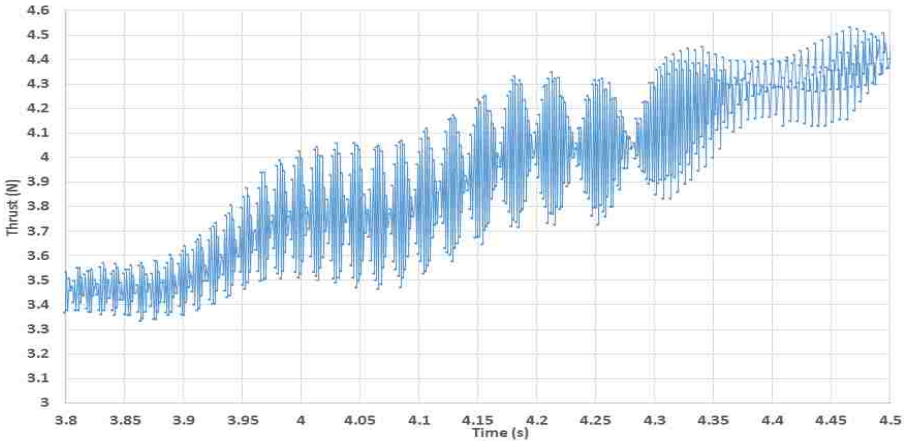

Figure 6.7. Resonant Frequency Oscillations

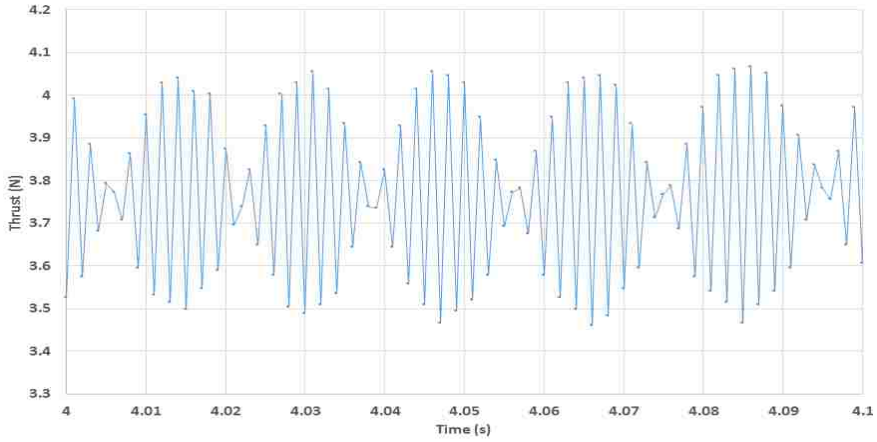

Figure 6.8. Close-up on Oscillations

6.2.1.2 Power data. Real-time power data was collected by measuring the initial voltage and the current. The voltage was multiplied by the current to find the power. Plotting this data gives the results shown in Figure 6.9. As shown, the sensor was very noisy due to interference from the motor. However, the results are also consistent, with each motor and propeller combination producing a very similar graph. The power increased linearly until it plateaued at the peak thrust level. At this level, the power experienced noise of 60 W but was still very consistent with very strict bounds being observed.

The maximum power for this motor and propeller combination was found by averaging the plateaued section between the five and ten second interval. For the 1200-K_v motor and 7x6 propeller, the maximum power was found to be 115.6 W. Because the power data for all the tests was similar to Figure 6.9, the averaging technique allowed the data to be accurately compared.

Figure 6.10 shows a zoomed-in portion of the graph right before it hits the plateaued section. The data varies significantly by approximately 60 W due to the noise. The red line on the figure shows a moving average calculation that removes the noise to show the actual linearly increasing power.

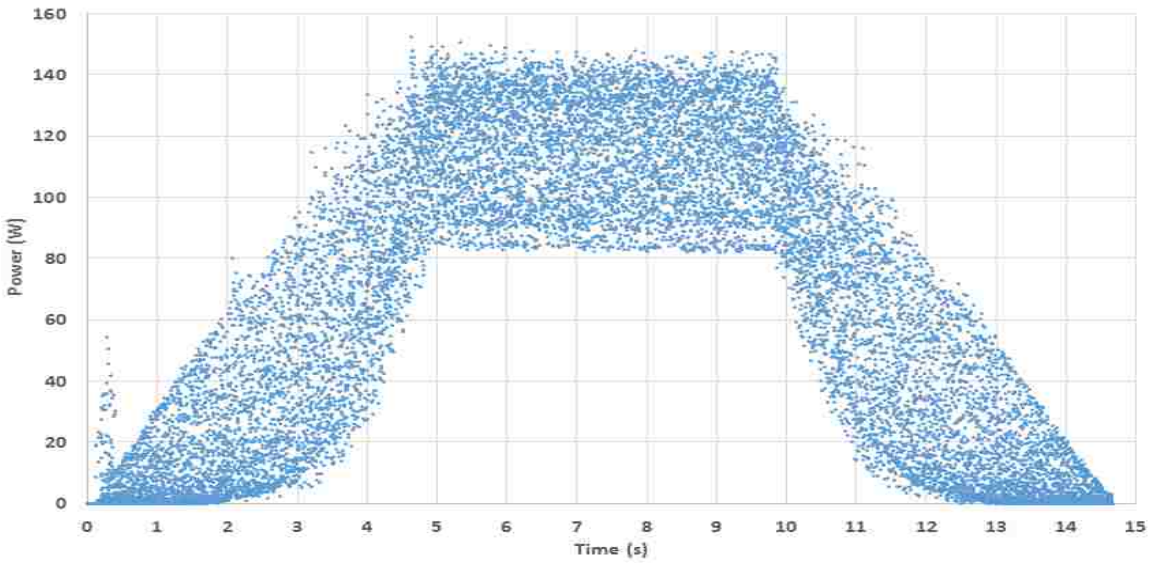

Figure 6.9. Power Data for 1200-K_v 7x6 0.050-in Beam Test

The voltage of the battery during the test was measured at 12.5 V with the current equaling 9.25 A. The Manufacturers data, shown in Appendix A, shows that the power for this motor and propeller should equal 101.3 W at a voltage of 12.5 V. In contrast, the power measured by the instrumentation module was 13.2% higher at 115.6 W. As shown in Figure 6.11, the manufacturers' data falls within the noise suggesting that the actual power of the motor was not located at the center of the noise data.

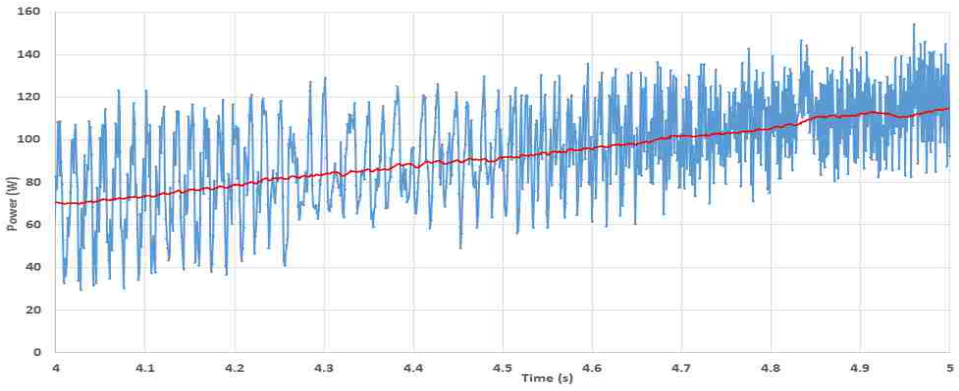

Figure 6.10. Moving Average of Noise for Power Data

6.2.1.3 Thrust & power data. The thrust and power data from Figures 6.5 and 6.9 were graphed together on Figure 6.11 to demonstrate the relationship between thrust and the power consumed by the motor. The thrust data and the lower edge of the power data mirror each other for the entirety of the test.

6.2.2. Motor Comparisons. The three motors were compared using the 7x6 propeller to determine the differences in thrust output. Figure 6.12 gives a visual comparison of the three motors with the 7x6 propeller on the 0.062-in beam. The 1200-K_v motor was the most powerful in terms of thrust output followed by the 980-K_v motor which produced 21.4% less thrust because of its smaller K_v rating. Even though the motors were approximately the same size, the 980-K_v motor runs at lower rpms, reducing its thrust output. The weakest motor was the 1080-K_v motor which produced 36.4% less thrust than the 1200-K_v motor and 20.0% less thrust than the 980-K_v motor. The lower thrust output

was caused by the smaller height of the motor and lower torque. The K_v rating of the motor can only be used for comparisons when the motor sizes are identical.

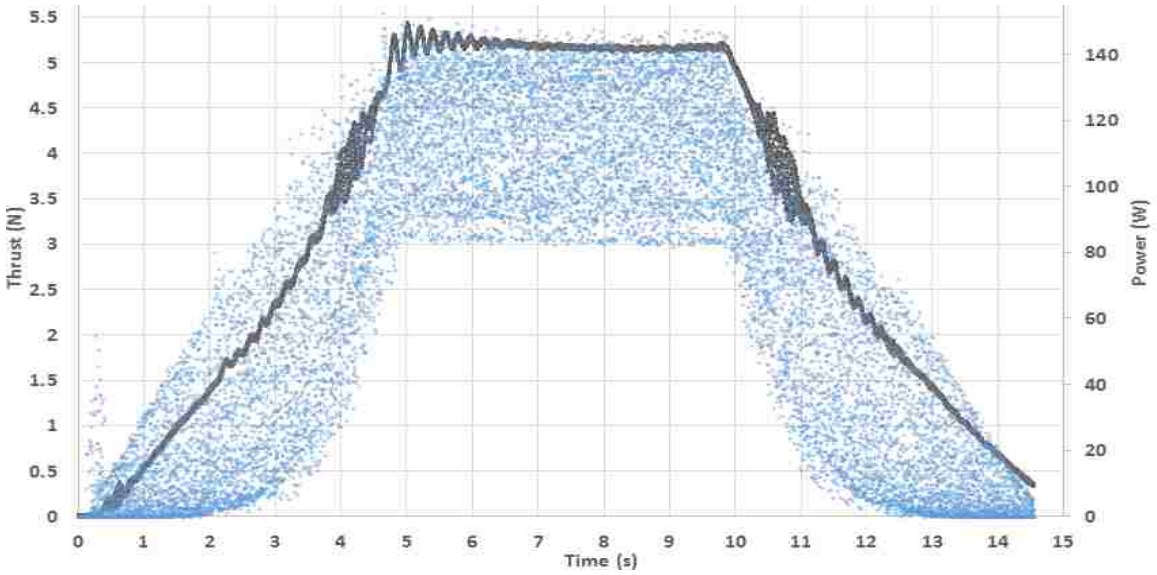

Figure 6.11. Power and Thrust Data

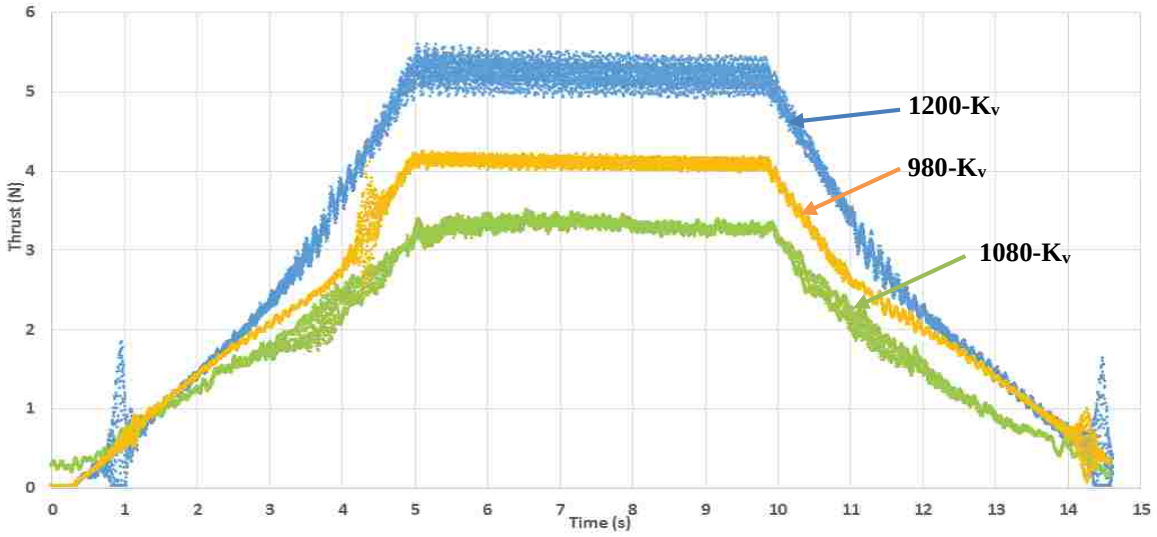

Figure 6.12. Motor Comparison of 0.062-in Beam with 7x6 Propeller

6.2.3. Propeller Comparisons. All five propellers were compared using the 1200-K_v motor to show the differences in thrust. As seen in Figure 6.13, the diameter of the propeller has a larger impact on the thrust output than the pitch. Jumping up two inches in diameter from five to seven produced the largest increase in thrust.

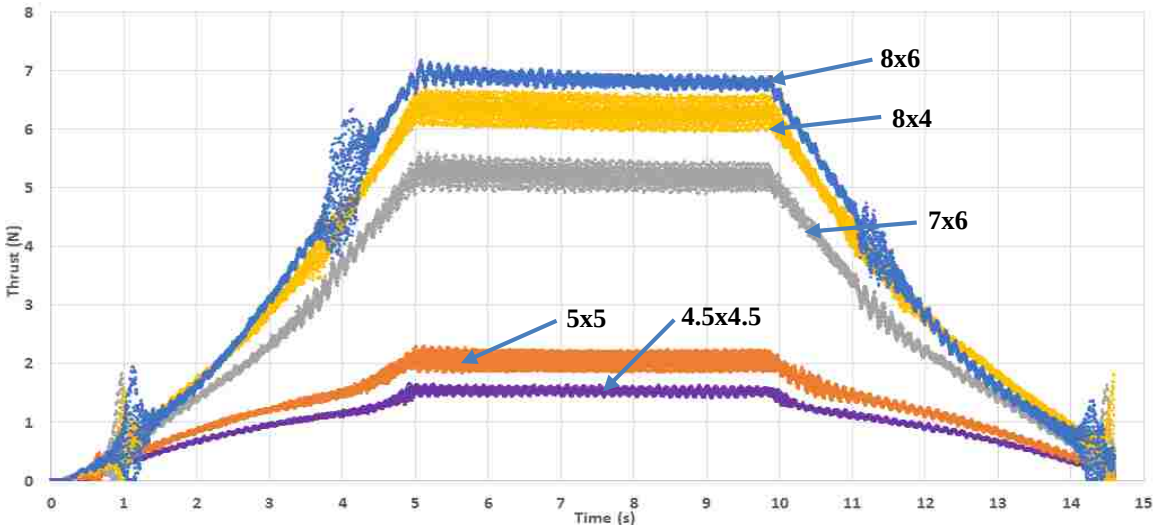

Figure 6.13. 1200-K_v Motor with all Five Propellers

6.2.4. Tabulated Results. This section includes all of the collected data on the maximum thrust, power, and the power-to-thrust ratio for the static testing. Table 6.3 gives the maximum thrust data. This data was collected by averaging the thrust produced when the motor was running at full throttle. The 1200-K_v motor data was very similar between the two beam tests, however the other two motors produced different thrust data for the same tests. Table 6.4 gives the power data collected when the motors were operating at full throttle. As with the thrust data, the 1200-K_v data was similar between the two beam tests unlike the other two motors.

Table 6.5 contains the power-to-thrust ratio for each test while operating at full throttle and was found by dividing the power data from the thrust data. This data was more consistent across beam tests showing that while the motors were producing different thrusts, they were also correspondingly consuming different amounts of power. The most efficient motors were the 1080-K_v and 980-K_v motor because of their higher power-to-

thrust ratio. While the 1200-K_v motor was more powerful in thrust output, it was less efficient overall.

Table 6.3. Experimental Thrust Data

Propeller	0.050-in Beam			0.062-in Beam		
	980-K_v	1080-K_v	1200-K_v	980-K_v	1080-K_v	1200-K_v
4.5 x 4.5	1155 mN	837 mN	1573 mN	1038 mN	1132 mN	1522 mN
5 x 5	1644 mN	1128 mN	2108 mN	1450 mN	1532 mN	2048 mN
7 x 6	3331 mN	-	5182 mN	4098 mN	3318 mN	5215 mN
8 x 4	3792 mN	-	6230 mN	5049 mN	-	6292 mN
8 x 6	4241 mN	-	6784 mN	5504 mN	-	6847 mN

Table 6.4. Experimental Power Data

Propeller	0.050-in Beam			0.062-in Beam		
	980-K_v	1080-K_v	1200-K_v	980-K_v	1080-K_v	1200-K_v
4.5 x 4.5	21.0 W	15.8 W	32.3 W	19.5 W	18.4 W	31.1 W
5 x 5	29.9 W	22.9 W	47.0 W	27.6 W	27.3 W	46.6 W
7 x 6	68.3 W	-	115.6 W	76.4 W	62.7 W	113.8 W
8 x 4	70.9 W	-	120.1 W	81.7 W	-	120.2 W
8 x 6	97.6 W	-	152.5 W	110.8 W	-	159.3 W

Table 6.5. Power-to-Thrust Ratio

Propeller	0.050-in Beam			0.062-in Beam		
	980-K_v	1080-K_v	1200-K_v	980-K_v	1080-K_v	1200-K_v
4.5 x 4.5	55.0	53.0	48.7	53.2	61.5	48.9
5 x 5	55.0	49.2	44.9	52.5	56.1	43.9
7 x 6	48.7	-	44.8	53.6	52.9	45.8
8 x 4	53.5	-	51.9	61.8	-	52.3
8 x 6	43.5	-	44.5	49.7	-	43.0

6.2.5. Small UAV Results. The Syma S107 UAV was tested on the thinner 0.032-in beam because of its lower thrust output. The main rotor thrusts were tested by slowly moving the throttle from minimum to maximum position, holding at maximum,

then slowly lowering to minimum as shown in Figure 6.14. The maximum thrust of the rotors was calculated by taking an average of the plateau portion of the graph, from eight seconds to eleven seconds and gave a value of 549.6 mN of thrust. Because the thrust was for both rotors combined, the value was divided by two to give approximately 275 mN of thrust for each rotor. The results were very stable, with the UAV only varying by about 8 mN of thrust when at full power.

As shown in Figure 6.15, the UAV produced large oscillations at startup when power was applied to the rotors rapidly. This information could be used to program how the UAV begins flight for a smoother start. Likewise, in Figure 6.14, once the rotors hit the 200 mN threshold, the motors turned completely off which can influence landings.

Three other tests were conducted with the data being tabulated in Table 6.6. With the main rotors at full power, the tail rotor was pulsed to measure its thrust. The next two tests measured the effect right and left turns have on thrust. Making a right or left turn slows down one of the main rotors so they are no longer spinning synchronously and caused the UAV to turn either right or left depending on which rotor was spinning faster. Interestingly, the right turn produces more thrust than the left turn.

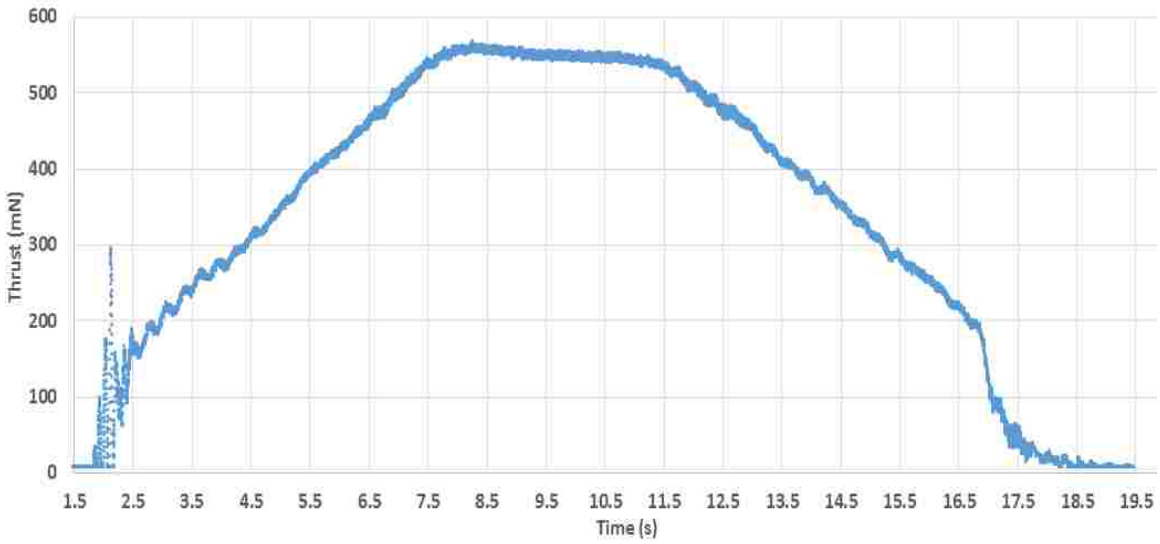

Figure 6.14. Syma UAV Main Rotors Thrust Output

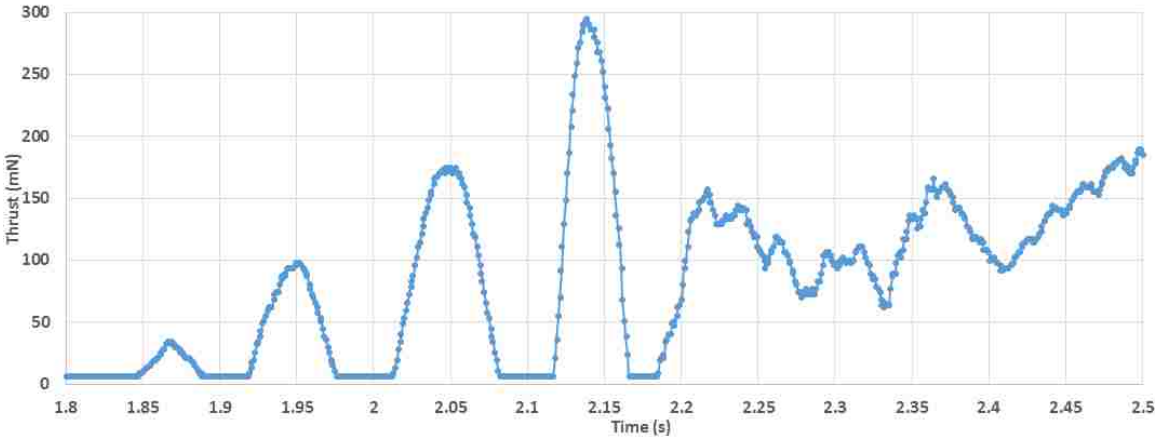

Figure 6.15. Syma UAV Main Rotors Thrust Startup

Table 6.6. Syma S107 UAV Thrust Data

Rotor	Max Thrust (mN)
Main	549.6
Tail	36.8
Left Turn	349.2
Right Turn	423.2

6.3. WIND TUNNEL TESTING

Two propellers were tested in the wind tunnel, the 7x6 and 5x5 propellers. The tunnel is described in Section 3.5. The two propellers were tested in conjunction with the 1200-K_v motor and 0.062-in beam. To calibrate the pitot tube, the atmospheric pressure and temperature of the outside air were found to be 98.98 kPa (29.23 inHg) and 284.8 K (11.67 °C). The drag test was ran first with no propeller from 0 to 23.38 m/s (0 to 80 ft/s). The results were plotted in Figure 6.16 and a quadratic trendline was found. The trendline equation was used as the drag values for the two propeller tests. The 7x6 propeller was run at 55% throttle capacity due to large vibrations at higher speeds. The 5x5 propeller was able to run at 100% throttle capacity. The excess thrust, shown in Figure 6.17, was found by adding the positive thrust values and the negative drag values at each speed. The maximum speed was found by finding the x-intercepts of the trendlines for the data. Equation 6-1 shows the quadratic trendline for the tare drag while Equation 6-2 and 6-3

show the trendline equations for the 7x6 and 5x5 propellers. Figure 6.18 shows the power used by the motor at each speed. It can be seen that the motors draw constant power until their maximum speed was reached. At this point, the propellers also act as a turbine, with the wind helping to turn the blades, reducing the power needed by the motor.

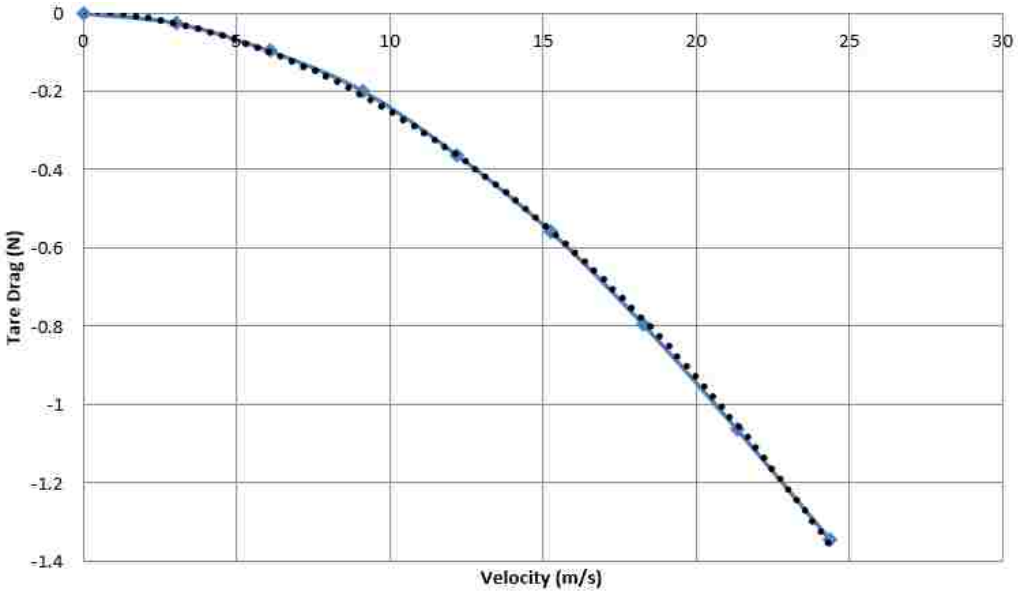

Figure 6.16. Tare Drag of 0.062-in Beam with 1200-K_v Motor

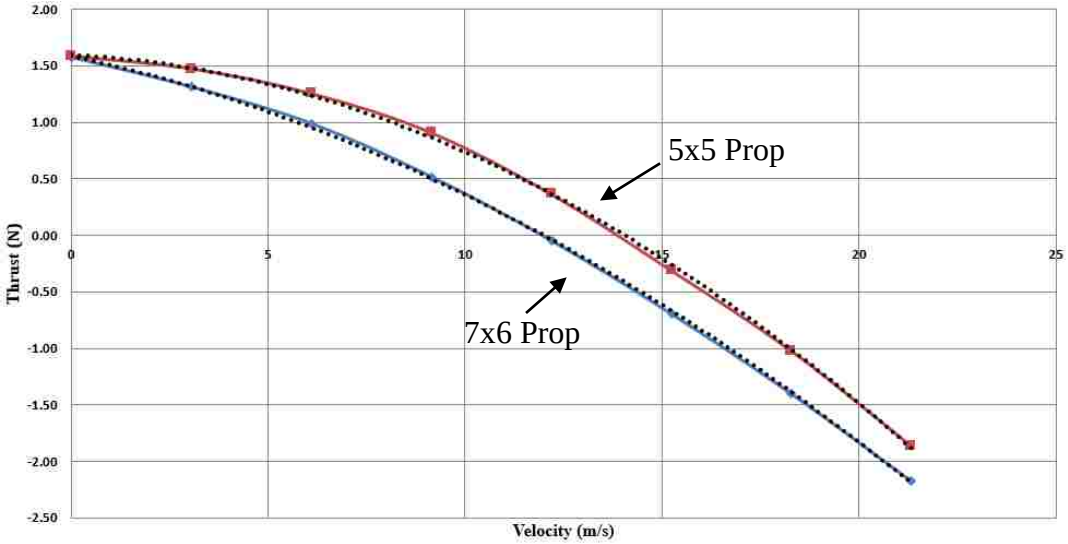

Figure 6.17. Excess Thrust Wind Tunnel Test

$$\text{Drag (mN)} = -2.1059 \times \text{Velocity}^2 - 4.739 \times \text{Velocity} + 7.079 \quad (6\text{-}1)$$

$$\text{Thrust (mN)} = -4.814 \times \text{Velocity}^2 - 74.563 \times \text{Velocity} + 1586.6 \quad (6\text{-}2)$$

$$\text{Thrust (mN)} = -6.830 \times \text{Velocity}^2 - 17.627 \times \text{Velocity} + 1597.4 \quad (6\text{-}3)$$

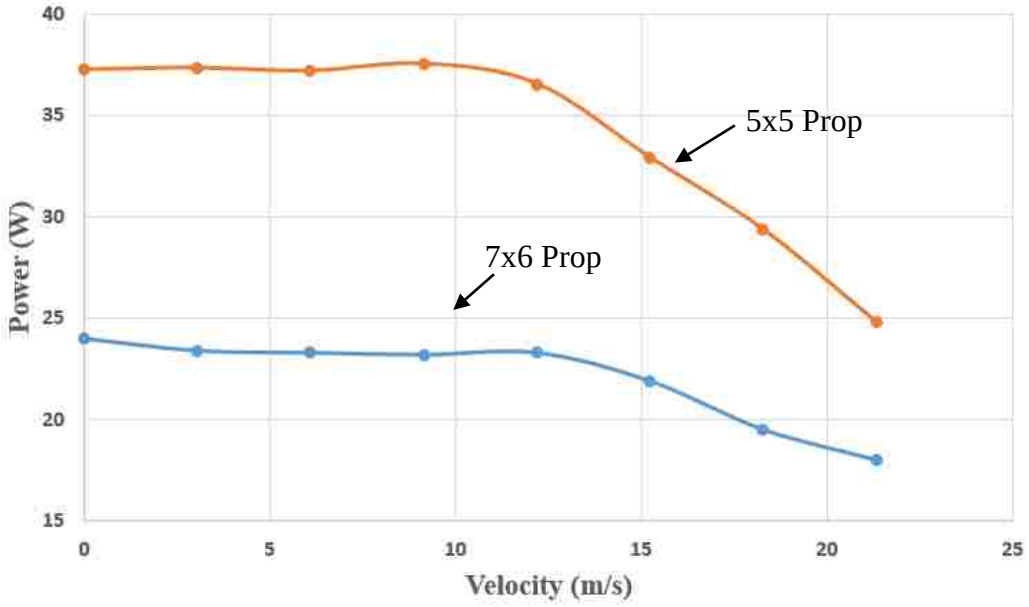

Figure 6.18. Power Consumed During Wind Tunnel Tests

6.4. DISCUSSION AND APPLICATIONS

The thrust and power data shown in this chapter demonstrate that the strain data from a cantilever beam can be used to find the thrust of small motors and propellers. The thrust data was averaged to remove the vibrations induced on the beam. As shown, the 1200-K_v motor was the most reliable of the three motors, producing equivalent thrusts on both beams. The thrust data also matched the power data with a linearly increased power output corresponding to a linearly increased thrust output. The power and thrust data together could be used to sense the health of the UAV. An onboard voltage and current sensor could provide the power data. By comparing the power-to-thrust ratio over time, the battery and motor health could be monitored.

The design of the cantilever beam was very important in determining the thrust range, sensitivity and resonant frequencies. The thrust range and sensitivity are influenced by the length and thickness of the beam. Resonance of the beam was also an important

factor that was induced by certain motor rpm's. Due to resonance, the 1080-K_v motor tests could not be completed. Future cantilever beams could be designed to reduce this resonance. UAVs should be tested to ensure they do not induce resonance in the beam.

The thrust sensing apparatus can be scaled up or down depending on the application. A thicker and shorter beam would reduce the sensitivity but would increase the range to allow the testing of larger motors. Likewise, using a thinner and longer beam, would reduce the range but would increase the sensitivity for smaller motors. Thinner beams are also less prone to vibrations and more accurate than the thicker beams, allowing them to be used to test the very small thrusts produced by satellite micro-thrusters.

7. SUMMARY

This thesis addressed the lack of adequate methods of testing the performance, health, and efficiency of smaller sized UAVs by developing a constant-stress cantilever beam and electrical strain gauge thrust stand. A full-bridge Wheatstone circuit was used to measure the produced force. Also, associated power data was monitored for the electric motors.

Three constant-stress cantilever beams, with thicknesses of 0.032 in, 0.050 in, 0.062 in, were developed to measure the thrust produced. Calibration testing confirmed the strain simulations of the three beams. After the 0.050-in beam experienced a permanent torsion failure, the 0.062-in beam was designed to reduce the effect of torsion and increased in thickness to increase the thrust range. The natural frequency of the beams were determined as it can have a negative result on the strain and thrust readings. An instrumentation module was constructed to control the motors and collect data from the strain, voltage, and current sensors.

Three individual motors with a maximum thrust output of 5 N to 6 N were tested in combination with five propellers ranging from 4.5 in to 8 in in diameter and 4 in to 6 in in pitch. All motors and propellers were tested during the static tests and the 1200-K_v motor and 7x6 propeller were tested in the wind tunnel. Power and thrust data was collected and compared with the power-to-thrust ratio being the final metric for each motor and propeller combination. Watching the power-to-thrust ratio over time gives information on the health of the motor and battery system.

This work met all the criteria that were specified at the start of the project. A test stand was developed for both a static and dynamic testing environment. The design measured thrust from a range of 0 – 10.40 N, well above the required 3.0 N target. Even at this thrust range, the required 0.1 N resolution was still met. For the two thinner beams, the resolution was higher with less range. Power data was recorded for the 0 – 200 W target using a voltage sensor and a current sensor. The resolution of the power measurements was 0.018 W at a constant voltage, well over the required 0.1 W resolution. Both strain and power data were recorded at a set interval (1 ms) for accurate timing and

high enough that vibrations of approximately 500 Hz were recorded on the beam. Finally, the thrust stand was successfully implemented in the wind tunnel.

There were some limitations of the presented research that could be addressed in the future. Power data for the micro-UAV could be gathered in the future by integrating the battery and controls into the instrumentation module. Vibration modeling of the cantilever beams could be completed to reduce the motors activation of the beams' resonant frequency. This would enable the thrust stand to test all motor and propeller combinations of the appropriate size. Finally, long term testing of the motors and airframe could be completed in the future to demonstrate the thrust stand's ability to identify faults during field conditions before they become fatal.

Applications for the work presented in this thesis relate to component selection and to the health monitoring of small sized UAVs. Various motor, propeller, and battery combinations can be evaluated. Long term component failures such as motor breakdown and battery degradation can easily be tested. Monitoring the vibrations produced by the UAV could be used to detect cracks in the frame of the aircraft. The test setup could also be scaled down to address the problem of measuring the thrust of micro-thrusters for satellites.

APPENDIX A.
MANUFACTURERS' DATA

This appendix gives information on the components and equipment used in this thesis including motors, propellers, previously unmentioned design components, and calibration equipment.

A.1. MOTOR DATA

This section includes manufacturers' data on the motors and UAV tested in this project. When provided by the manufacturers, thrust and power data is given in Tables A.1, A.2, and A.3. The dimensions of the motors are given and correspond to dimensions of Figure A.1.

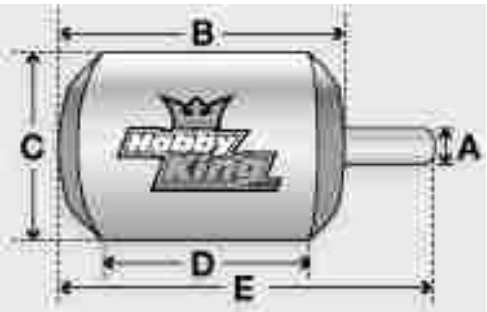

Figure A.1. Motor Dimensions [25]

Table A.1. Motor Specifications [25]

Motor:	NTM 28-26	Turnigy Park300	Turnigy SK3
K_v:	1200 rpm/V	1080 rpm/V	980 rpm/V
Max Current:	17 A	9 A	10 A
Max Power:	215 W @ 12 V	100 W @ 11.1 V	96 W
Weight:	57.6 g	25 g	44 g
A: Shaft Diameter	3 mm	3 mm	3 mm
B: Length	26 mm	16 mm	28 mm
C: Diameter	28 mm	28 mm	28 mm
D: Can Length	15 mm	8 mm	12 mm
E: Total Length	42 mm	36 mm	40 mm

Table A.2. NTM 1200kv Manufacturer Prop Tests [25]

Prop	Voltage (v)	Current (A)	Power (W)	Thrust (g)
7x6	11.1	8	90	-
8x4	11.1	9	100	600
8x6	11.1	13	145	-

Table A.3. Turnigy 1080kv Manufacturer Prop Tests [25]

Prop	Voltage (v)	Current (A)	Power (W)	Thrust (g)
8x4	11.1	5.5	55.5	360

A.2. PROPELLER DATA

Manufacturers' data was found for each propeller used in this thesis. Figure A.2 gives the thrust data as a function of revolutions per minute (rpm) for the 4.5x4.5 and 5x5 propellers. Similarly, Figures A.3, and A.4 give the data for the 5x5, 7x6, 8x4, and 8x6 propellers. This data was gathered by users from flybrushless.com.

A.3. CALIBRATION EQUIPMENT

The following equipment was used to calibrate the instrumentation module. The signal generator was a Tektronix AFG 5101. The signal generator was used to calibrate the ADC of the microcontroller. Attached to the signal generator was a Tektronix PS 5010 programmable power supply. The power source was used to calibrate the current sensor in conjunction with a handheld multimeter by utilizing its constant current feature. Also attached was a Tektronix PS 5004 Precision Power Supply which was used to calibrate the voltage sensor. A HP 54600A oscilloscope was used to verify the signals from the equipment. Figure A.5 shows the standard weights set that was used in the cantilever beam calibrations. The weights ranged from 1 g to 100 g. Figure A.6 shows the strain gauge reference unit which was used to apply a known strain to calibrate the strain indicator unit. Figure A.7 shows the AC/DC converter used to power the instrumentation module for extended periods of time.

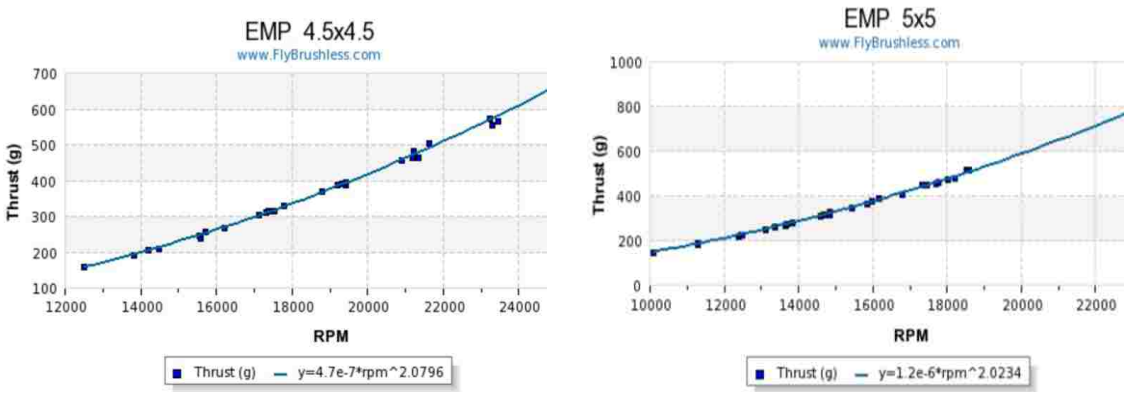

Figure A.2. 4.5x4.5 & 5x5 Propeller Thrust [26]

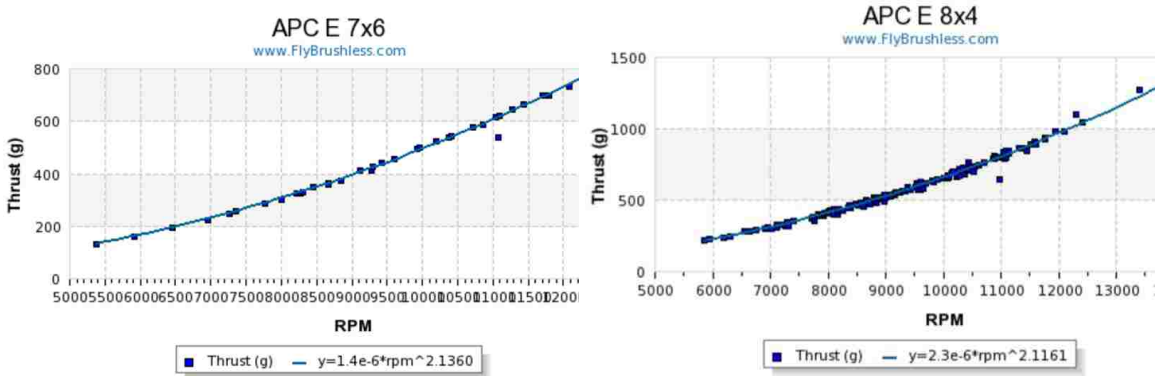

Figure A.3. 7x6 & 8x4 Propeller Thrust [26]

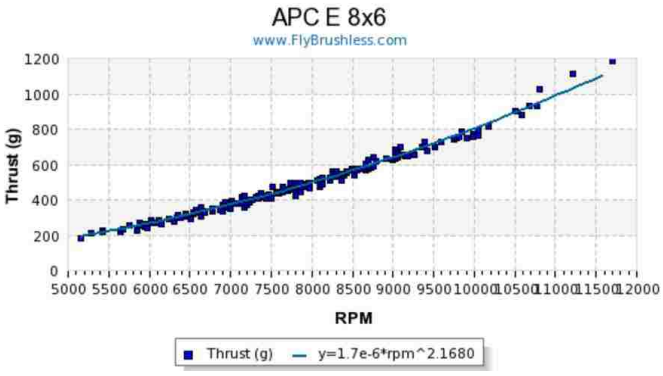

Figure A.4. 8x6 Propeller Thrust [26]

Figure A.5. Standard Weights Set

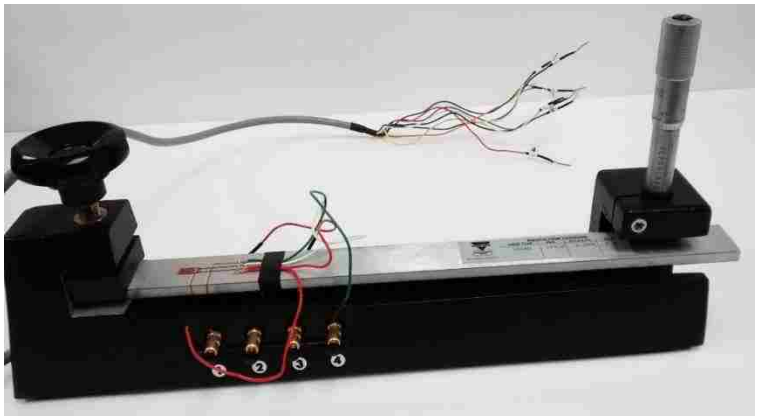

Figure A.6. Strain Gauge Reference Unit

Figure A.7. Power adapter with XT60 Connector

A.5. MISCELLANEOUS COMPONENTS

This section gives details on all components not previously mentioned in the body of the paper. The strain gauges used were Vishay Micro-measurements, model EA-06-250BG-120. The gauges have a resistance of 120.0 ± 0.15% and a GF of 2.085 ± 0.5%.

A normally-open relay, model SLA-05VDC-SL-C was used as a safety feature to either establish or break the connection from the power source to the ESC. The relay ensured that the ESC and motor received power but could be shut off quickly if needed. The relay was rated at 30 A for 30 VDC which was within the operating range of all other components. It was controlled by the microcontroller with a digital signal. Because the relay was a normally-open relay, any problem with the microcontroller would disable the relay.

A potentiometer knob allowed the user to control the speed of the motor manually rather than running a test profile. It functioned as a simple voltage divider circuit and was connected to an analog input on the Arduino microcontroller. It had a range of 0 to 11.4 kΩ. A toggle switch was used to alternate between manual and automatic modes. The switch was connected to a digital input on the Arduino microcontroller. A push button served a dual purpose as a start button and as an emergency stop button. Pressing the button began the test when in auto mode or disconnected the power supply with the relay if a test was in progress. This button was connected to the Arduino microcontroller. An RGB (red, green, blue) LED was used to communicate the status of the device. The LED was a common cathode LED with the three connections sharing a common ground. A 470 Ω current limiting resistor was included on each LED anode to prevent damage. The power input connectors used were XT60 connectors. These connectors are commonly used in RC applications and the LiPo battery pack shipped with these connectors preinstalled. The AC/DC wall adapter was also fitted with this connector for compatibility. A BNC (Bayonet Neill-Concelman) connector was used to connect the strain indicator to the instrumentation module. This allowed a standard coaxial cable to connect the two devices.

APPENDIX B.
PROCEDURES

This appendix gives procedures that were too long for the main paper. This includes the strain gage application, ESC, strain indicator, and coolTerm software procedures.

B.1. STRAIN PROCEDURES

Strain gages were applied to the cantilever beams with the following procedure:

Prepare Surface:

- Abrade the surface to create scratch marks. This helps adhesion by providing a larger surface area for bonding
- Begin with 150 grit sandpaper and use a circular motion
- Wet sand with 400 grit sandpaper

Clean Surface:

- Add water, then wipe with a towel in one direction until no shavings remain
- Do the same with acetone
- Rinse and dry with water

Mark strain gauge positions:

- Use a scribe to score the sample to mark the desired position
- Clean the sample with water and dry

Apply strain gages:

- Handle with tweezers to keep the oils off the gauges
- Align the gauges on the sample. To keep them in position use cellophane tape
- Pull the tape up on one side to move the gauge off the sample
- Apply adhesive (super glue) to the specimen. Put the gauge back onto the sample
- Push on the gage with a finger and hold until set (2-3 minutes)

Apply Bonding Pad:

- Apply with same technique as strain gauges
- Solder ribbon leads to the gauges and bonding pads. Use tape to hold in place
- Solder wire to the bonding pads

Coat with an enamel spray

The strain indicator operation followed the following procedure. The strain gauges were attached to the binding posts inputs in the full-bridge configuration. While the beam was unstressed, the indicator was then zeroed by pressing the Amp Zero button and using the control to set the reading to ± 0000. The strain gauge GF was set with the appropriate controls. After pressing the run button, the balance was adjusted with the balance controls to get a reading of ± 0000. The beam was then loaded with an external force and strain reading was displayed on the LCD display and sent to the instrumentation module through the analog output.

The strain indicator was calibrated with the shunt resistor calibration procedure. One of the beam gauges was attached in a quarter-bridge Wheatstone configuration to the strain indicator. The run button was pressed and the gauge was zeroed. The calibration button (CAL) was pressed which connected the internal calibration resistor of 5000 με in parallel with the gauge. The ideal reading with no lead resistance should be in the range 4793 με to 4798 με from Equation B-1 [12]. The gauge factor dial was rotated to bring the strain value into this range. The new gauge factor was then recorded.

$$5000\mu\epsilon \times \frac{2.000}{\text{GF } (2.085)} \pm 0.05\% \qquad \text{(B-1)}$$

B.2. ESC PROCEDURES

The ESC was activated using the following procedure. From the Turnigy manual, the start-up sequence for the ESC included providing power to the ESC, sending a minimum throttle signal (as determined by calibration), then waiting for confirmation. The ESC communicated through a set of beeps generated by sending high frequency pulses through the motor. Once power was applied, a long beep was emitted to tell that the power supply was within range. Finally a set of three short beeps signals that the ESC had been successfully armed.

To enter the programming mode on the ESC, a high throttle was sent before the device was turned on. This procedure differs from the normal procedure that sends a low throttle position at startup. Once activated, the user was required to wait two seconds and then send the new low throttle position value. A long beep confirmed the throttle settings were accepted.

B.3. COOLTERM SOFTWARE

The following procedure was used to capture the incoming data using the CoolTerm software. First, the appropriate serial port needed to be opened. Once the serial connection was established, a software trigger on the Arduino microcontroller noticed the opened serial connection and began the process of initializing the ESC. At this point, the user pressed CTLR + R buttons, which started saving the incoming serial stream to a text file. Next, the user would hit the start button on the instrumentation module to run the test in either auto or manual mode. Once the test was completed, the microcontroller stopped sending data. In CoolTerm, CTLR + SHIFT + R would then be pressed to close the serial connection and finish creating the text file. The file could then be imported into Microsoft Excel as a CSV (comma separated value) and the calibration equations could be applied to the raw data.

APPENDIX C.
ARDUINO CODE

Appendix C gives the various Arduino programs used in the project. This includes ESC throttle calibration, sensor calibration, static testing, and wind tunnel code. The code was written in the Arduino IDE, a C-based language. Figure C.1 gives the ESC throttle calibration code. Figure C.2 shows the sensor calibration code. Figure C.3 gives the static testing code. Finally, Figure C.4 shows the wind tunnel testing code. Program descriptions and comments can be found in each figure.

ESC_Throttle_Calibration

```
/*    ESC THROTTLE CALIBRATION
 *    Chris Marchman
 *
 *    PROGRAM DESCRIPTION
 *
 *    This program calibrates the ESC (Electronic Speed Controller) throttle position.
 *    The throttle is controlled by sending a servo value from 0 to 255. This program
 *    takes a minimum value and maximum value and sets these values as the minimum and
 *    maximum throttle positions.
 */

#include <Servo.h>

const int ESCpin = 5;     //IO Pin for the ESC Output
const int relayPin = 3;   //IO Pin for the relay

const int minSpeed = 19;  //Min speed for throttle position
const int maxSpeed = 132; //Max speed for throttle position

Servo ESC;  //Creates a servo instance

void setup() {
  pinMode(ESCpin, OUTPUT);    //Sets the ESCpin to output mode
  pinMode(relayPin, OUTPUT);

  Serial.begin(9600);  //Sets the data rate for the Arduino serial connection

  while(!Serial);  //Waits for a serial connection to be established. This waits for the
                   //the CoolTerm software to be ready.

  digitalWrite(relayPin, HIGH);  //Turns the relay on
  delay(100);
  ESC.attach(ESCpin);     //Attaches the servo to the correct pin
  ESC.write(maxSpeed);    //Sends a servo signal to the ESC. By sending the max speed value
                          //first, the ESC enters the throttle programming mode

  delay(2500);            //Wait 2.5 seconds for the ESC to save the throttle data
  ESC.write(minSpeed);    //Sends the min throttle value to the ESC
  delay(5000);            //Wait 5 seconds for the ESC to save the throttle data. The ESC
 }                        //will beep twice if the data is accepted.

void loop() {
  //The program does not require loop code. The code will not compile without it.
}
```

Figure C.1. ESC Throttle Calibration Code

Sensor_Calibration

```
/*    SENSOR CALIBRATION
 *    Chris Marchman
 *
 *    PROGRAM DESCRIPTION
 *
 *    This program is used to calibrate the three sensors: Strain, Voltage, and Current.
 *    The appropriate sensor pin needs to be selected and uploaded before beginning. Data
 *    is sampled at a specified rate and transmitted over the serial connection to be
 *    recorded by the CoolTerm software.
 */

#include <MsTimer2.h> //Timer 4 on Arduino Micro

const int currentPin = A8;  //IO pin for the current sensor input
const int voltagePin = A7;  //IO pin for the voltage sensor input
const int strainPin  = A6;  //IO pin for the strain sensor input
const int relayPin   = 3;   //IO pin for the relay output

//*************Set Sensor Pin*****************
const int sensorPin = currentPin; //Set sensor pin to the chosen sensor input

const int dataInterval = 1; //Sets the sample rate in milliseconds


void setup() {
  analogReference(INTERNAL);  //Sets the ADC reference voltage to the internal 2.56V

  pinMode(sensorPin, INPUT);  //Sets the sensorPin to input
  pinMode(relayPin,  OUTPUT);

  Serial.begin(230400); //Sets the data rate for the Arduino serial connection
  while(!Serial); //Waits for a serial connection to be established. This waits for the
                  //CoolTerm software to be ready.
  delay(100);

  MsTimer2::set(dataInterval, isr); //Configures timer 4 to be triggered. (Sample rate in
                      //ms, interrupt service routine function)
  MsTimer2::start(); //Begins the timer
}

//The program does not require loop code. The code will not compile without it.
void loop() {
 }

//Interrupt Service Routine,is called at the sample rate specified by MsTimer2 library
void isr(){
  //Reads the data from the sensor, then transmits it over the serial connection
  Serial.println(analogRead(sensorPin));
}
```

Figure C.2. Sensor Calibration Code

Static_Testing

```
/*   THESIS PROJECT
 *   Chris Marchman
 *
 *   PROGRAM DESCRIPTION
 *
 *   This program serves to run and collect data for the test apparatus. The ESC is
 *   controlled by the servo command. Data is collected at a set interval as
 *   determined by the MsTimer2 interrupt. This data is sent over a serial connection
 *   to be read by the CoolTerm software a host computer.
 *
 *   FUNCTION & LIBRARY DICTIONARY
 *
 *   sensors() - ISR(intterrupt service routine) that runs when the external interrupt
 *      is triggered. This interrupt is controlled by MsTimer2. The function reads the
 *      data from the sensors, converts to them to a string seperated by commas, and
 *      sends them to the host computer over the serial connecion.
 *   updateESC() - updates the ESC speed servo signal. If the program is in manual mode,
 *      the ESC speed is controlled by the poteniotmeter. If in auto mode, the ESC speed
 *      is updated in the ISR and then transmitted to the ESC in this function.
 *   testConditions() - tests for safety conditions. Turns off the module if there is a
 *      problem.
 *   shutDown() - turns off the device when the test is complete.
 *   servo library - library that generates the pulse pulse modulation. This library is
 *      included in the Arduino IDE.
 *   MsTimer2 library - Library developed by Javier Valencia to easily utilized the timer 4
 *      of 10ms gives a sample rate of 100Hz.
 *      on the Arduino Micro. Enables the ISR to be called at a set interval. An interval
 */

#include <Adafruit_TiCoServo.h>  //Run servo without interrupts
#include <MsTimer2.h> //Timer 4 on Arduino Micro

// Variables for IO pins
const int ESCpin     = 5;
const int voltagePin = A7;
const int currentPin = A8;
const int strainPin  = A6;
const int buttonPin  = 7;
const int togglePin  = 2;
const int potPin     = A11;
const int relayPin   = 3;
const int redPin     = 9;
const int greenPin   = 10;
const int bluePin    = 11;

const int sweepSpeed = 5*10; //seconds to sweep from min-max, max-min, hold
const int minSpeed = 25; //Minimum
const int maxSpeed = 125;

int ledState = LOW; //For blinking LEDs
unsigned long previouaMillis = 0;

volatile int ESCspeed = minSpeed;     //Global variable to update motor speed
volatile int counter = 1;             //Global variable to update every 1 ms
volatile int holdCounter = 1;         //Global variable to count hold period
volatile boolean mode;                //Global variable for mode, LOW=manual, HIGH=automatic
volatile int profile;
String sensorString;

Adafruit_TiCoServo ESC; //Creates a servo instance for the ESC
```

```
61
62 void setup() {
63   analogReference(INTERNAL);  //Sets the ADC reference voltage to the internal 2.56V
64   //Sets the pins to input or output
65   pinMode(ESCpin,     OUTPUT);
66   pinMode(voltagePin, INPUT);
67   pinMode(currentPin, INPUT);
68   pinMode(strainPin,  INPUT);
69   pinMode(buttonPin,  INPUT_PULLUP); //Sets the internal pullup resistor for binary input
70   pinMode(togglePin,  INPUT_PULLUP);
71   pinMode(potPin,     INPUT);
72   pinMode(relayPin,   OUTPUT);
73   pinMode(redPin,     OUTPUT);
74   pinMode(greenPin,   OUTPUT);
75   pinMode(bluePin,    OUTPUT);
76
77   unsigned long currentMillis; //Used for blinking LEDS without using the delay() function
78
79 //********Select Mode (Manual or Auto)********
80   //Wait for Button Push
81   while(digitalRead(buttonPin) == HIGH){
82     if(digitalRead(togglePin) == HIGH){
83       digitalWrite(bluePin, HIGH);
84       digitalWrite(greenPin, LOW); //Automatic
85     }else{
86       digitalWrite(bluePin, HIGH); //Manual
87       digitalWrite(greenPin, HIGH);
88     }
89    }
90   mode = digitalRead(togglePin); //Get mode from toggle pin
91
92   Serial.begin(230400); //Activates serial connection
93
94 //********Wait for Serial Connection*******
95   while(!Serial){
96     //Blink the LED
97     currentMillis = millis();
98
99     if(currentMillis - previousMillis >= 500){
100      previousMillis = currentMillis;
101
102      if (ledState == LOW){
103        ledState = HIGH;
104      }else{
105        ledState = LOW;
106      }
107      digitalWrite(redPin, ledState);
108    }
109  }
110  digitalWrite(redPin, ledState);
111  delay(100);
112
113 //********Initialize ESC********
114  digitalWrite(relayPin, HIGH); //Turn on relay
115  delay(100);
116  ESC.attach(ESCpin);
117  ESC.write(10);
118  delay(5000); //Wait for ESC to verify power source
119
120  //Reset colors
```

```
  digitalWrite(bluePin, LOW);
  digitalWrite(greenPin, LOW);

//********If in Manual, Wait for knob to be set to zero*********
  if(mode == LOW){
    Serial.println("Reset Dial to Zero");
    while(analogRead(potPin) >= 2){
       //Blink the LED
      currentMillis = millis();

      if(currentMillis - previousMillis >= 100){
        previousMillis = currentMillis;

      if (ledState == LOW){
        ledState = HIGH;
      }else{
        ledState = LOW;
      }
      digitalWrite(redPin, ledState);
      }
    }
  }

//******Wait for Start Button to Begin Test******
  digitalWrite(bluePin, LOW); //Turn off other two lights
  digitalWrite(redPin, LOW);
  digitalWrite(greenPin, HIGH);
  Serial.println("Push Start Button");
  while(digitalRead(buttonPin) == HIGH){ //Wait for start button
    currentMillis = millis();
    if(currentMillis - previousMillis >= 500){
      previousMillis = currentMillis;

      if (ledState == LOW)
        ledState = HIGH;
      else
        ledState = LOW;

      digitalWrite(greenPin, ledState);
    }
  }

//******Get Voltage******
  digitalWrite(bluePin, LOW);  //Set color to green
  digitalWrite(greenPin, HIGH);
  digitalWrite(redPin, LOW);
  Serial.println("Voltage:");
  for(int i=0; i <=10; i++){ //Loop to read the voltage 10 times to obtain average
    Serial.println(analogRead(voltagePin));
  }
  Serial.println("************");  //Sepeartes the voltage values from the current/strain values
  Serial.println("Current, Strain"); //Tells order of the data

  MsTimer2::set(1, sensors); //period, ISR
  MsTimer2::start();
}

//Most of the code is in the ISR. The loop only updates the ESC and checks the test conditions
void loop() {
  updateESC();
```

```
  testConditions();
}

/* ISR (Interrupt Service Routine)
 * Function called every millisecond by MsTimer2 library. Read data from the current and strain
 * sensors, combine into a single string, then output over the serial conneciton. Also, update
 * timing counter.
 */
void sensors(){
  String currentString;
  String strainString;

  sensorString = ""; //Clear string
  currentString = String(analogRead(currentPin)); //Convert current value to string
  strainString = String(analogRead(strainPin));   //Convert strain value to string
  sensorString += currentString; //Add current value to string
  sensorString += ','; //Add comma
  sensorString += strainString; //Add strain value
  Serial.println(sensorString); //Send data over serial connection

  counter++; //Update timing counter
}

/* Checks time to run test profile. Counter updated every 1 ms.
*   Case 1: Increase ESC speed every 50ms 100 times for a total of 5 seconds
*   Case 2: Hold Max ESC speed for 5 seconds, holdcounter undated everty 50ms
*   Case 3: Decrease ESC speed every 50ms 100 times for a total of 5 seconds
* Once Case 3 is complete, the interrupts are stopped and the test is complete.
*/
void updateESC(){

  int val = minSpeed;
  static int lastVal = val;

  //Automatic Mode
  if(mode == HIGH){
    if(counter == 50){ //Update ESC speed every 50ms
    switch(profile){ //Select mode, default is Case 1
     case 1:
       ESCspeed++;
       if(ESCspeed == maxSpeed) //If at max ESC speed, enter mode 2
          profile = 2;
      break;

      case 2:
        if(holdCounter == 100) //Wait for 5 seconds, then enter mode 3
          profile = 3;
        else
          holdCounter++;
      break;

      case 3:
         ESCspeed--;
         if(ESCspeed == minSpeed){ //If at min ESC speed, turn off relay and stop timer
           profile = 4;
           shutDown(); //Shuts down the program. Test is complete.
         }
      break;
      }
    ESC.write(ESCspeed); //write speed to ESC
```

```
    counter = 1;
    }
  }else{
    val = analogRead(potPin); //Read potentiometer
    val = map(val,0,1023,minSpeed-5,maxSpeed+5); //convert from 0-1023 to 0-100
    if((lastVal == val) ||(lastVal == val+1) ||(lastVal == val-1)){ //keep values from jumping
      ESC.write(val);
      lastVal = val;
    }
  }
}

//Read switch & button, test for safety conditions
void testConditions(){
  if(digitalRead(buttonPin) == LOW){ //If the stop button is pressed, open the relay
     digitalWrite(relayPin, LOW);

     Serial.println("ERROR: Emergency stop button was pressed");
     MsTimer2::stop(); //Disables the interrupt and data collection

     while(1){ //Wait for reset
       digitalWrite(redPin, HIGH);
       digitalWrite(greenPin, LOW);
       digitalWrite(bluePin, LOW);
     }
  }
}

//Shuts down test
void shutDown(){
  MsTimer2::stop(); //Disables the interrupt
  digitalWrite(relayPin, LOW); //Opens the relay

  while(1){
     digitalWrite(redPin, HIGH);
     digitalWrite(greenPin, HIGH);
     digitalWrite(bluePin, HIGH);
  }
}
```

Figure C.3. Static Testing Code

Wind_Tunnel §

```
/*  WIND TUNNEL TESTING
 *
 *  Simplified code for manual control inside the wind tunnel.  Changes include
 *  a slower data sample time to accommodate transmitting voltage, current,
 *  strain, and thrust position.
 */

#include <Adafruit_TiCoServo.h>  //Run servo without interrupts
#include <MsTimer2.h> //Timer 4 on Arduino Micro

// Variables for IO pins
const int ESCpin = 5;
const int voltagePin = A7;
const int currentPin = A8;
const int strainPin = A6;
const int relayPin = 3;
const int potPin = A11;
const int buttonPin = 7;

const int minSpeed = 25;
const int maxSpeed = 125;

volatile int counter = 1;
volatile int throttle = minSpeed; //Throttle Position from Potentiometer

Adafruit_TiCoServo ESC; //Create a servo instance for the ESC

void setup() {
  analogReference(INTERNAL);  //Sets the ADC reference voltage to the internal 2.56V
  pinMode(ESCpin, OUTPUT);
  pinMode(voltagePin, INPUT);
  pinMode(currentPin, INPUT);
  pinMode(strainPin, INPUT);
  pinMode(relayPin, OUTPUT);
  pinMode(buttonPin, INPUT_PULLUP);
  pinMode(potPin, INPUT);

  Serial.begin(230400);  //Sets the data rate for the Arduino serial connection
  while(!Serial); //Waits for a serial connection to be established. This waits for the
  delay(500);     //CoolTerm software to be ready.
  Serial.println("Voltage, Current, Strain, Throttle");

  //********Initialize ESC********
  digitalWrite(relayPin, HIGH);
  delay(100);
  ESC.attach(ESCpin);
  ESC.write(10);
  delay(5000); //Wait for ESC to verify power source

  MsTimer2::set(10, sensors); //period, ISR. Slower to accommodate more data
  MsTimer2::start();
}

void loop() {
    updateESC();
    testConditions();
}

/* ISR (Interrupt Service Routine)
 * Function called every millisecond by MsTimer2 library. Read data from the current and strain
 * sensors, combine into a single string, then output over the serial conneciton. Also, update
 * timing counter.
 */
```

```
void sensors(){
  Serial.print(analogRead(voltagePin));
  Serial.print(",");
  Serial.print(analogRead(currentPin));
  Serial.print(",");
  Serial.print(analogRead(strainPin));
  Serial.print(",");
  Serial.println(throttle);
}

// Updates the ESC speed, called every 20 ms
void updateESC(){
  int val;

  val = analogRead(potPin); //Reads the potentiometer
  throttle = map(val,0,1023,minSpeed-2, maxSpeed); //Maps the potentiometer value to the ESC values
  ESC.write(throttle); //write speed to ESC
  counter = 1;
}
//Read switch & button, test for safety conditions
void testConditions(){
  if(digitalRead(buttonPin) == LOW){ //If the stop button is pressed, open the relay
     digitalWrite(relayPin, LOW);

     Serial.println("ERROR: Emergency stop button was pressed");
     MsTimer2::stop(); //Disables the interrupt and data collection

     while(1){ //Wait for reset

     }
  }
}
```

Figure C.4. Wind Tunnel Testing Code

APPENDIX D.
SCHEMATICS

This section gives the full schematic of the instrumentation module, shown in Figure D.1. Pins on the Arduino microcontroller not used in this project are a smaller font and italicized. Pins with an "A" indicate analog pins while pins without are digital. A schematic of the terminal block layout is shown in Figure D.2. Finally, a schematic of the full-bridge Wheatstone configuration used on the cantilever beams is given in Figure D.3. Gauges T1 and T2 are located on the top side of the beam while B1 and B2 are on the bottom.

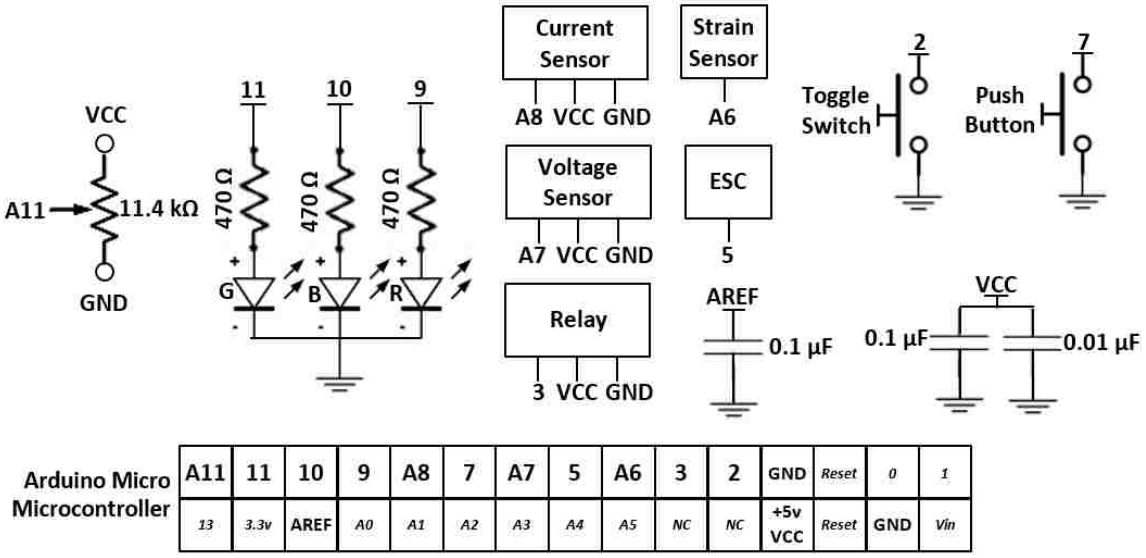

Figure D.1. Instrumentation Module Schematic

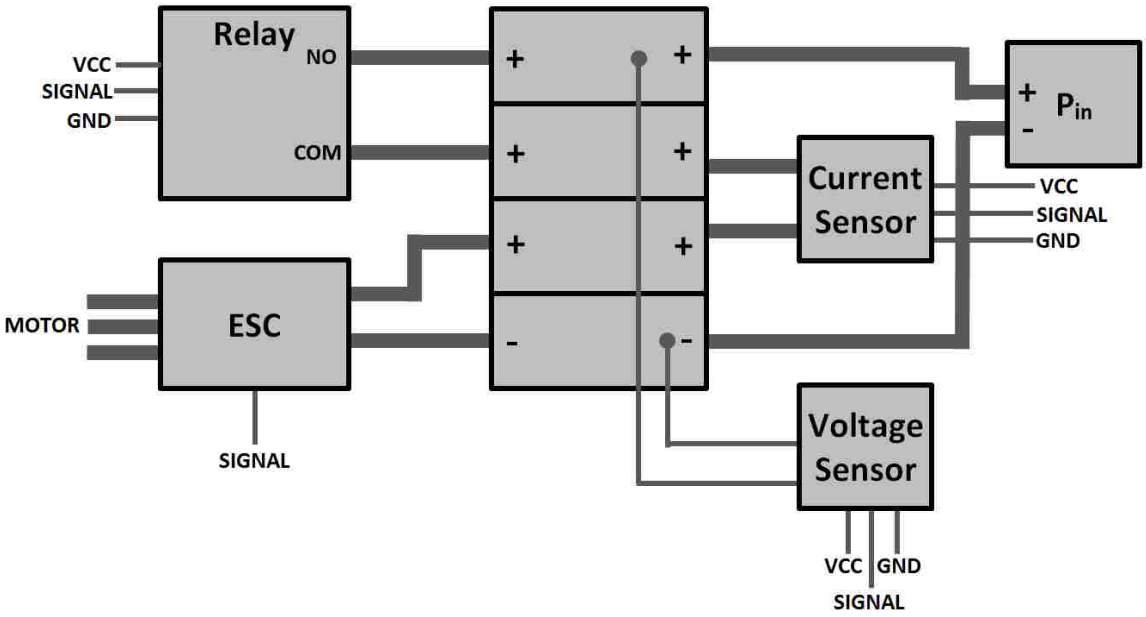

Figure D.2. Terminal Block Schematic

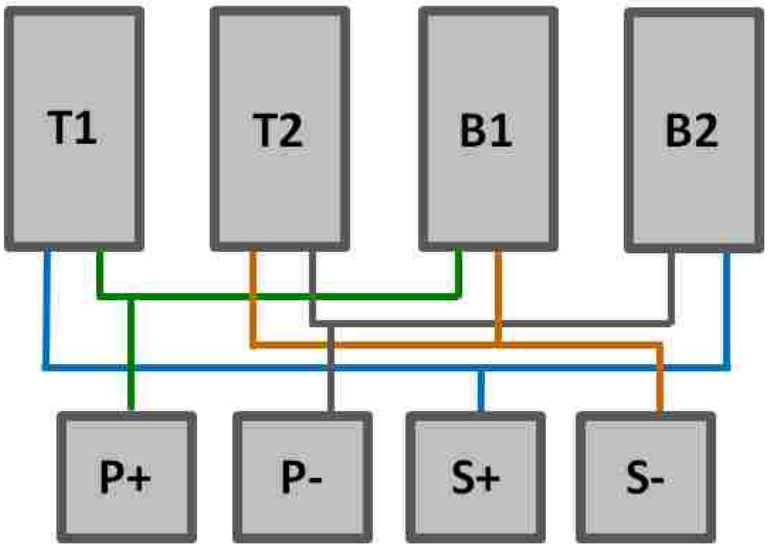

Figure D.3. Strain Gauge Schematic

APPENDIX E.
CALIBRATION DATA

This appendix gives the data used to calibrate the instrumentation such as the strain indicator, microcontroller and sensors. The data was used to calibrate the ADC (analog-digital converter) and to verify settings. For the sensors, the data was used to obtain equations to convert the raw ADC to the desired units. Due to the large quantities of data generated while sampling at 1000 Hz, figures showing the data are used rather than presenting the raw data. Each cantilever beam's response to weight was also recorded.

E.1. STRAIN INDICATOR GAGE FACTOR CALIBRATION

The initial strain reading for the calibration was 4793 με which was out of range for an ideal reading. The gage factor was adjusted to 2.084 to give an adjusted calibration value of 4796 με.

E.3. MICROCONTROLLER CALIBRATION

This section gives the data collected during the calibration of the microcontroller. The ADC read speed calibration data is shown in Figure E.1. Figure E.2 shows the accuracy of the serial connection when reading transmitting at 230400 bps.

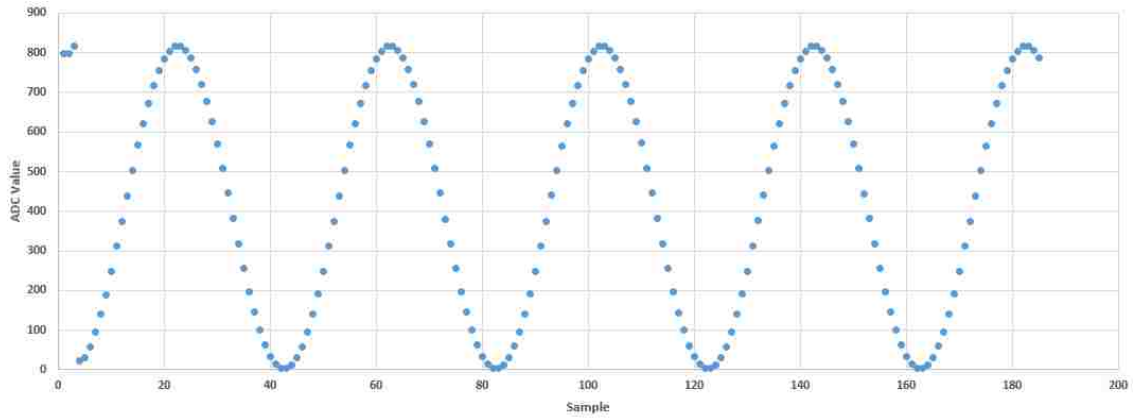

Figure E.1. 25-Hz 1.000-V Signal Sampled at 1000 Hz

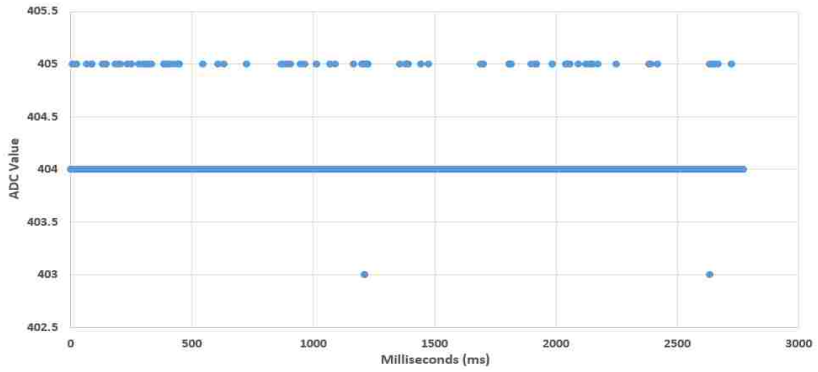

Figure E.2. 230400 Baud Rate Accuracy Test

E.4. STRAIN TO THRUST CALIBRATION

Table E.1 shows the calibration data for the strain to thrust calibration for each of the three beams.

Table E.1. Strain to Thrust Data

Weight (g)	0.032-in Beam Strain (με)	0.050-in Beam Strain (με)	0.062-in Beam Strain (με)
1	26	12	6
2	50	23	12
5	125	52	28
10	251	102	54
20	--	199	106
50	--	492	259
70	--	681	364
100	--	995	522
120	--	1194	630
150	--	1497	790

E.5. SENSOR CALIBRATION

The voltage, current, and strain sensor were calibrated to the ADC. The voltage data is given in Table E.2. Table E.3 shows the data for the current calibration. Table E.4 shows the data for the strain calibration. Figure E.3 shows the noise of the current sensor when calibrated with the constant current source.

Table E.2. Voltage Sensor Calibration

V_{in} (Power Supply) (V)	ADC Value
0.000	1009
0.500	996
1.000	982
5.000	874
10.000	737
15.000	601

Table E.3. Current Calibration Data

Current (mA)	ADC Value
50	212.9
100	215.5
150	218.3
200	221.1
250	223.8
300	226.5
500	237.4
700	248.3
1000	265.3

Table E.4. Strain Calibration Data

ADC Value	Strain ($\mu\epsilon$)
1	22
2	27
3	31
20	124
50	287

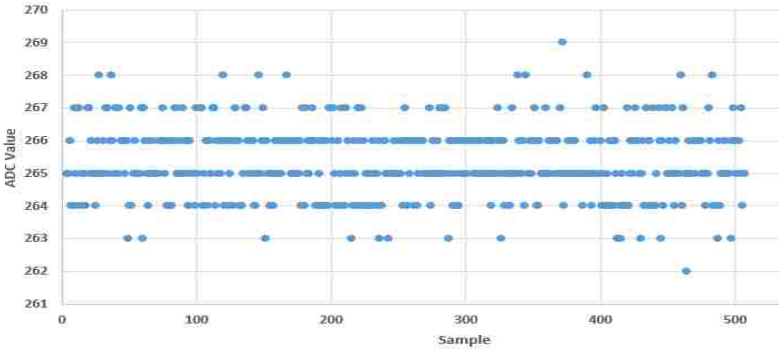

Figure E.3. 1000 mA Calibration Test

APPENDIX F.
RESULTS

This appendix gives results not shown in the results section of the paper including battery voltage tests, strain and power tests for the 1080-K_v and 980-K_v motors, and additional strain tests for the Syma micro-UAV.

F.1. BATTERY VOLTAGE TEST

The battery was tested with the 1200-K_v motor running at half power for 30 min. The voltage level was monitored over this time and readings were taken every 0.5 second. As shown in Figure F.1, the voltage decreases linearly over the first 25 min of the test until it reaches its rated voltage of 11.1 V. After this point, the voltage decreases at a rapid rate.

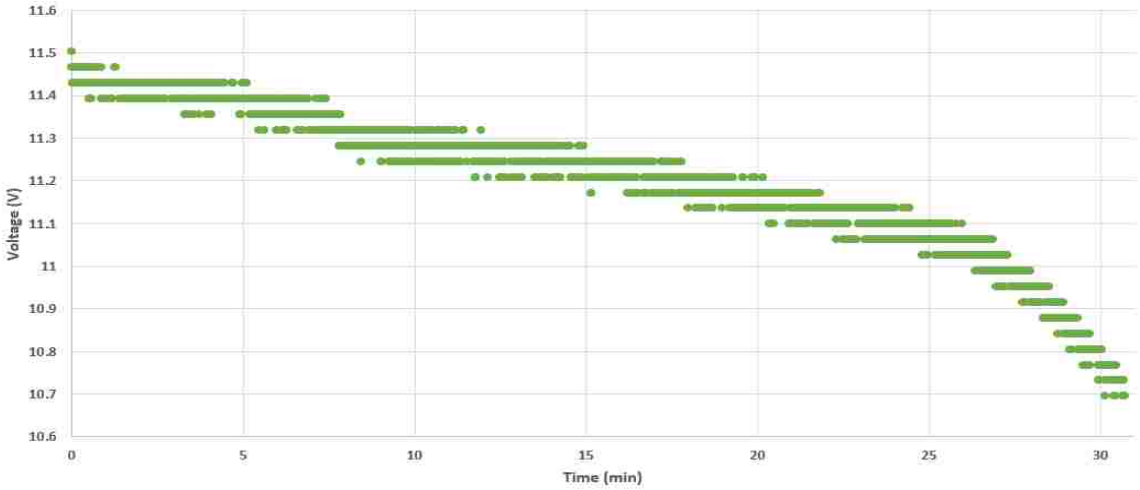

Figure F.1. 30 Minute Battery Voltage Test

F.2. STRAIN DATA FOR INDIVIDUAL MOTORS

This section gives the strain data for each of the three motors on the 0.062-in beam. Note that resonance causes spikes to appear at certain portions of the strain curves.

F.2.1. 980-K_v Motor. This section gives the strain data from the 980-K_v motor. Figures F.2, F.3, and F.4 show the 4.5x4.5, 5x5, 7x6, 8x4, and 8x6 propeller data. The 8x4 propeller produced significant vibrations compared to the other propellers. However, resonance did occur at the same portion of each graph indicating a particular motor frequency causes resonance with the 0.062-in thick beam.

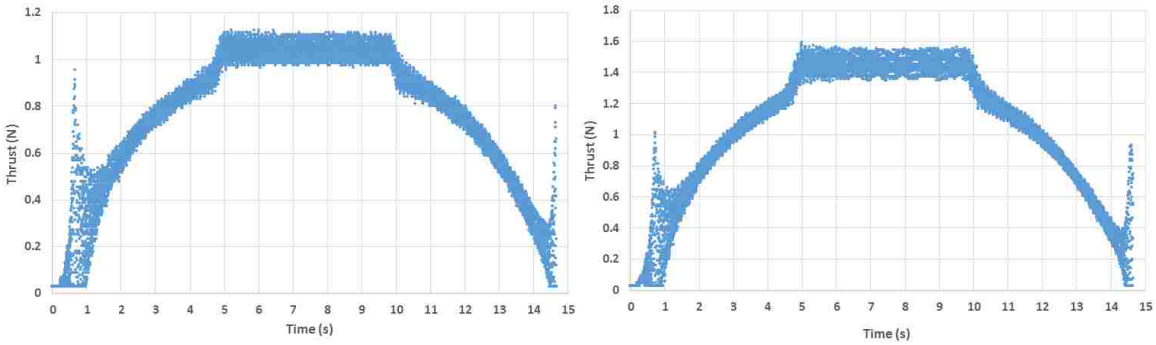

Figure F.2. 4.5x4.5 & 5x5 Strain Data

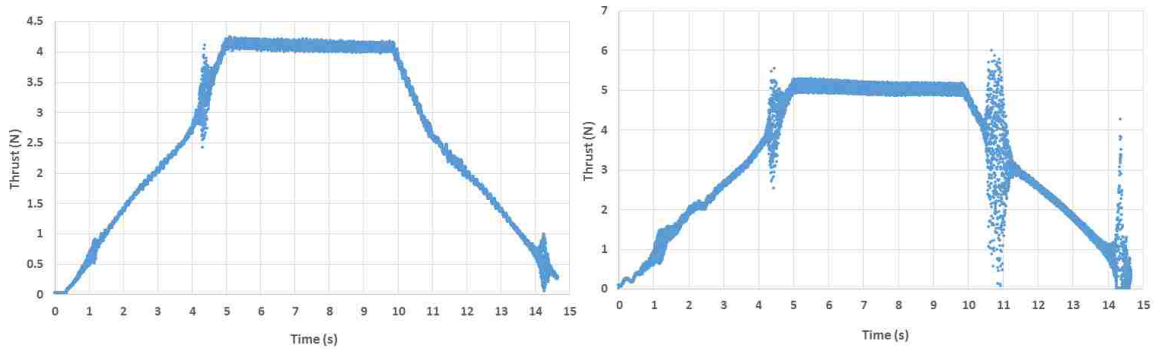

Figure F.3. 7x6 & 8x4 Strain Data

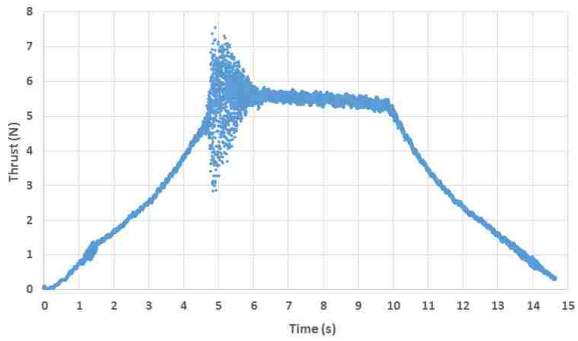

Figure F.4. 8x6 Strain Data

F.2.2. 1080-K_v Motor. This section gives the strain data from the 1080-K_v motor. Figures F.5 and F.6 show the 4.5x4.5, 5x5, 7x6, and 8x4 propeller data. Note that Figure F.5 experienced large vibrations at maximum thrust. The 8x4 plot shows the data collected before the emergency stop button was pushed due to the uncontrollable vibrations. No data from the 8x6 propeller could be collected due to the excessive vibrations.

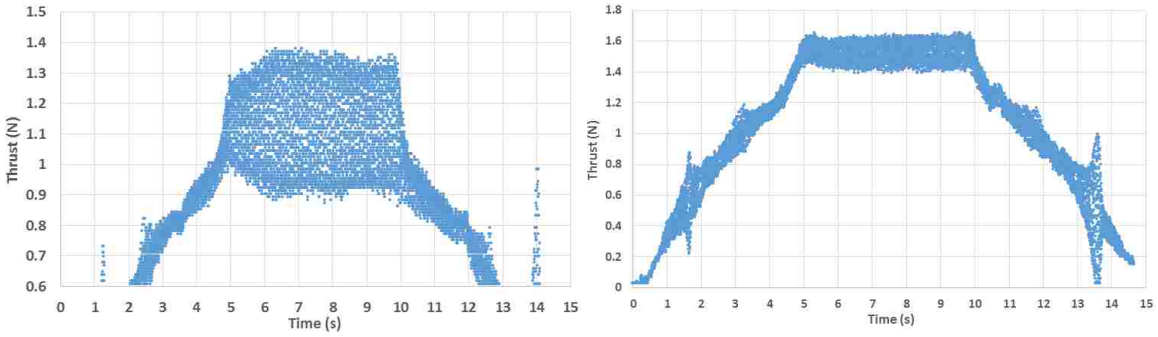

Figure F.5. 4.5x4.5 & 5x5 Strain Data

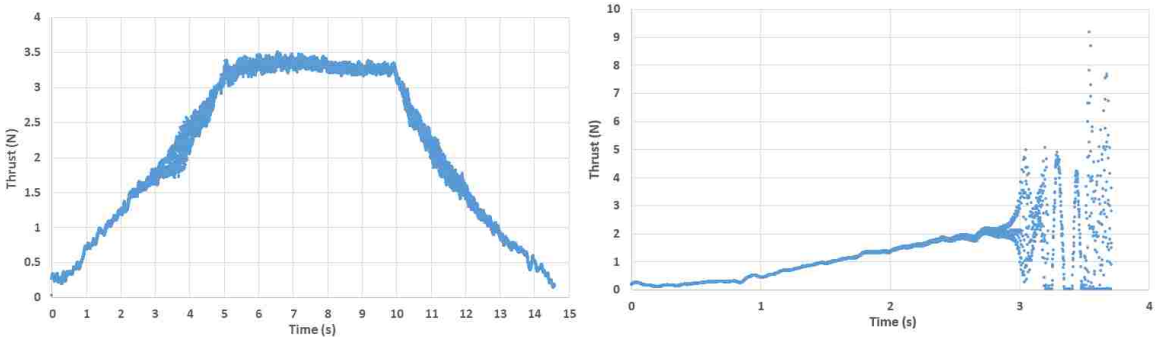

Figure F.6. 7x6 & 8x4 Strain Data

F.2.2. 1200-K_v Motor. This section gives the strain data from the 1200-K_v motor. Figures F.7, F.8, and F.9 show the 4.5x4.5, 5x5, 7x6, 8x4, 8x6 propeller data. Note that the 1200-K_v motor data was much cleaner with less vibrations than the other two motors.

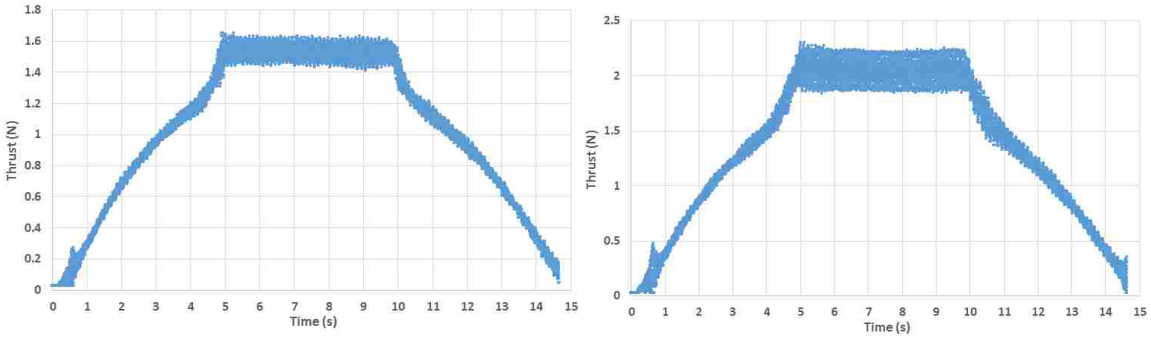

Figure F.7. 4.5x4.5 & 5x5 Strain Data

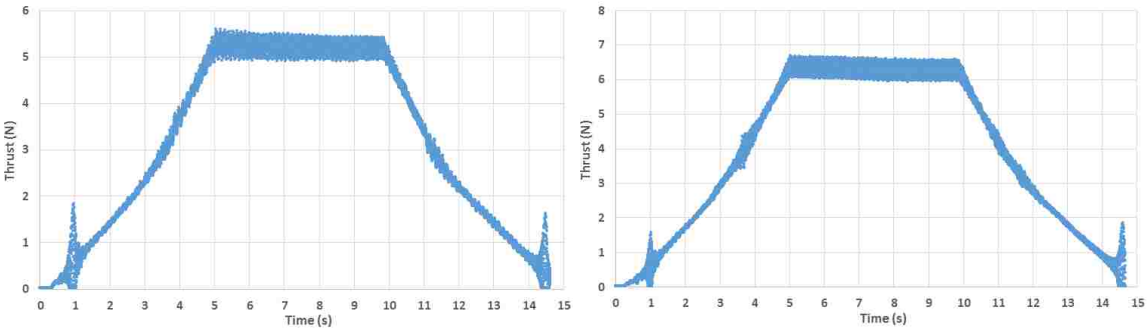

Figure F.8. 7x6 & 8x4 Strain Data

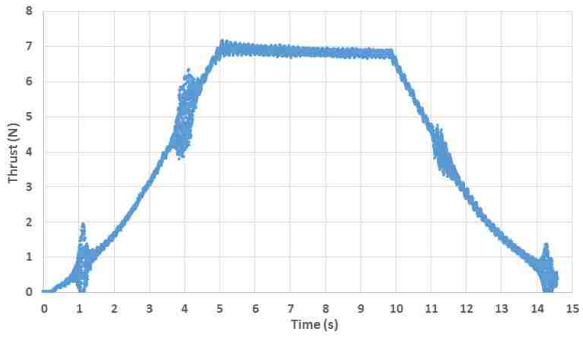

Figure F.9. 8x6 Strain Data

F.3. WIND TUNNEL DATA

This section gives the data values obtained from the wind tunnel tests. Table F.1. shows drag values obtained from the static test with the 0.062-in thick beam and the 1200-

K_v motor. The drag was recorded in increments of 10 ft/s. From the data, a second order polynomial trend equation was found. Column four of Table F.1 gives the calculated drag values from the equation. These values were used to determine the excess thrust of Table F.2. Table F.2 gives the measured and excess thrust values found for the 5x5 and 7x6 propellers at 10 ft/s intervals.

Table. F.1. Drag Values

Velocity (ft/s)	Velocity (m/s)	Tare Drag (mN)	Equation (mN)
10	3.048	-25.51	-26.93
20	6.096	-96.61	-100.07
30	9.144	-199.51	-212.33
40	12.192	-362.29	-363.73
50	15.24	-556.87	-554.26
60	18.288	-796.36	-783.92
70	21.336	-2063.91	-1052.70
80	24.884	-1346.43	-1360.62

Table F.2. Thrust Values

Velocity (ft/s)	Tare Drag (mN)	Measured Thrust (7x6)	Excess Thrust (7x6)	Measured Thrust (5x5)	Excess Thrust (5x5)
0	0	1571.99	1571.99	1584.87	1584.87
10	-26.93	1337.54	1310.61	1500.11	1473.18
20	-100.07	1087.23	987.16	1354.18	1254.11
30	-212.33	728.99	516.66	1119.31	906.98
40	-363.73	315.26	-48.47	727.43	363.70
50	-554.26	-141.15	-695.41	243.28	-310.98
60	-783.92	-617.89	-1401.80	-236.41	-1020.33
70	-1052.70	-1120.23	-2172.94	-807.96	-1860.67

BIBLIOGRAPHY

[1] J. P. How, C. Fraser, K. C. Kulling, L. F. Bertucelli, O. Troupt, L. Brunet, A. Bachrach, and N. Roy, "Increasing Autonomy of UAVs," *IEEE Robotics and Automation Magazine*, **16**(2), 43-51, (2009).

[2] R. Loh, Y. Bian, and T. Roe, "UAVs in Civil Airspace: Safety Requirement," *IEEE Aerospace and Electronic Systems Magazine*, **24**(1), 5-17, (2009).

[3] D. Erdos, A. Erdos, and S. E. Watkins, "An Experimental UAV System for Search and Rescue Challenge," *IEEE Aerospace and Electronics Systems Magazine*, **28**(5), 32-37, (2013).

[4] "UAS: RQ-11B Raven," *avinc.com*, 2015. [Online-accessed May, 2016]. Available WWW: http://www.avinc.com/uas/small_uas/raven

[5] John Bazin, Travis Fields, and Andrew J. Smith, "Feasibility of In-Flight Quadrotor Individual Motor Thrust Measurements," AIAA Atmospheric Flight Mechanics Conference, AIAA SciTech, (AIAA 2016-1760). San Diego, CA 4-8 January, 2016 (2016).

[6] D. Stojcsics, "Flight Safety Improvements for Small Size Unmanned Aerial vehicles," Intelligent Engineering Systems (INES), 2012 IEEE 16th International Conference on, Lisbon, 2012, pp. 483-487.

[7] Andrew D. Dimarogonas, "Vibration of a Cracked Structure: A State of the Art Review, " Engineering Fracture Mechanics, vol. 55, no. 5, pp 831 – 857, 1996.

[8] M. Desrochers, G. Olsen, and M. Hudson, "A Ground Test Rocket Thrust Measurement System," Journal of Pyrotechnics, vol. 2001, no. 14, pp. 50-55, 2001.

[9] A. Polk, Pancotti, T. Haag, S. King, M. Walker, J. Blakely and J. Ziemer, "Recommended Practices in Thrust Measurements," 33rd International Electric Propulsion Conference, Washington, DC, 2013.

[10] R. J. Stephen, K. Rajanna, V. Dhar, K. G. K. Kumar and S. Nagabushanam, "Thin-film Strain Gauge Sensors for Ion Thrust Measurement," IEEE Sensors Journal, vol. 4, no. 3, pp. 373-377, June 2004.

[11] C. Marchman. "A Camera Based Displacement Sensor for Thrust Stand with 40 µm Resolution," (unpublished). Missouri University of Science and Technology, 2015.

[12] Vishay Micro-Measurements, *Model P-3500 Digital Strain Indicator Instruction Manual,* Raleigh: 2003. Available WWW: http://www.caimlabs.co.za/Instrumentation%20manuals/P-3500%20manual.pdf.

[13] *E-106 Constant-Stress Beams*. Raleigh: Vishay Education Division, 1982, pp. 1-3.

[14] F. Weick, *Aircraft propeller design*. New York: McGraw-Hill Book Company, Inc., 1930.

[15] P. Carpenter, "RC Airplane Propeller Size Guide," *Rc-airplane-world.com*, 2016. [Online-accessed Jan, 2016]. Available WWW: http://www.rc-airplane-world.com/propeller-size.html.

[16] "Brushed vs Brushless Motors - Think RC," *Thinkrc.com*, 2016. [Online-accessed Feb, 2016]. Available WWW: http://www.thinkrc.com/faq/brushless-motors.php.

[17] H. Toliyat and G. Kliman, *Handbook of electric motors*. New York: Marcel Dekker, 2004.

[18] E. Eekhoff, "Brushless Motor KV Constants Explained," *LearningRC.com*, 2015. [Online-accessed Oct, 2015]. Available WWW: http://learningrc.com/motor-kv/.

[19] R. Meier. *CoolTerm*, Version 1.4.6. 2016. [Software]. Available WWW: http://freeware.the-meiers.org/.

[20] "Allegro MicroSystems - 0 to 50 A Integrated Current Sensor ICs," *Allegromicro.com*. [Online-accessed Feb 2016]. Available WWW: http://www.allegromicro.com/en/Products/Current-Sensor-ICs/Zero-To-Fifty-Amp-Integrated-Conductor-Sensor-ICs.aspx.

[21] "ATmega32U4 Datasheet,"*Atmel.com*. [Online-accessed Nov, 2015]. Available WWW: http://www.atmel.com/Images/Atmel-7766-8-bit-AVR-ATmega16U4-32U4_Datasheet.pdf.

[22] "Arduino - ArduinoBoardMicro," *Arudino.cc*, 2015. [Online-accessed Nov, 2015]. Available WWW: https://www.arduino.cc/en/Main/ArduinoBoardMicro.

[23] J. Valencia. *MsTimer2*, 2015. [Software]. Available WWW: https://github.com/PaulStoffregen/MsTimer2.

[24] P. Burgess, "Using NeoPixels and Servos Together," *Learn.adafruit.com*, 2015. [Online-accessed Feb, 2016]. Available WWW: https://learn.adafruit.com/neopixels-and-servos.

[25] "NTM Prop Drive Series 28-26A 1200kv / 286w," *Hobbyking.com*, 2016. [Online-accessed Nov, 2015]. Available WWW: http://www.hobbyking.com/hobbyking/store/__17345__NTM_Prop_Drive_Series_28_26A_1200kv_286w.html.

[26] "Find a Prop," *flybrushless.com*, 2016. [Online-accessed Mar, 2016]. Available WWW: http://flybrushless.com.

VITA

Christopher Scott Marchman received a Bachelor of Science degree in Electrical Engineering from Missouri University of Science and Technology in May 2014. He received a Master of Science degree in Electrical Engineering in July 2016. Chris was a member of IEEE, Phi Kappa Phi, and Tau Beta Pi. He received the Chancellor's Fellowship, a full-ride scholarship, to complete his graduate studies. Chris was also a runner for Missouri S&T's Cross Country and Track teams and won numerous Academic All-Conference Awards.