«««< HEAD ======= »»»> origin/master

«««< HEAD ======= »»»> origin/master «««< HEAD ======= »»»> origin/master

AALBORG UNIVERSITY
STUDENT REPORT

ED5-3-E16

Hovering control of a quadcopter

Students:
Alexandra Dorina Török
Andrius Kulšinskas

Supervisors:
Christian Mai

December 6, 2016

AALBORG UNIVERSITY
STUDENT REPORT

School of Information and Communication Technology
Niels Bohrs Vej 8
DK-6700 Esbjerg
http://sict.aau.dk

Title:
Hovering control of a quadcopter

Theme:
Scientific Theme

Project Period:
Autumn Semester 2016

Project Group:
ED5-3-E16

Participant(s):
Alexandra Dorina Török
Andrius Kulšinskas

Supervisor(s):
Christian Mai

Copies: x

Page Numbers: 41

Date of Completion:
December 6, 2016

Abstract:

xxx

Contents

»»»> origin/master

Preface

The project entitled *Hovering control of a quadcopter* was made by two students from the Electronics and Computer Engineering programme at Aalborg University Esbjerg, for the P5 project during the fifth semester.

From hereby on, every mention of 'we' refers to the two co-authors listed below.

Aalborg University, December 6, 2016.

Andrius Kulšinskas
<akulsi14@student.aau.dk>

Alexandra Dorina Török
<atarak14@student.aau.dk>

Chapter 1

Introduction

1.1 Introduction

The theme of this semester's project lies within *Automation*. Automation can be simply described as being the use of diverse control systems for fulfilling a certain task with little to no human interaction. As known from the previous semester, a control system is an instrument which has the role of adapting the behaviour of a system according to a desired state, also knows as steady-steady or reference. Any control system has three components: measurement, control and actuation. Without one of these, automation would not be possible. Essentially, the measure reflects the current state of the system, the controller is the brain that given the measurement decides which action will be performed and the actuator is the one executing the action.

Project ideas around the topic of automation are unlimited, since it is so widely spread. Having discussed a few of them that would meet the semester's requirements, we finally decided to work on the control of a quadcopter. Our decision was, for the most part, based on the fact that the university had the required equipment available, which enabled us to start working on the project right away.

UAVs have been attracting attention for many decades now. Powered UAVs were, at first, utilized by the military to execute reconnaissance missions. Nowadays, they have found other uses, such as aerial photography, search and rescue, delivery, geographic mapping and more. While there are different types of UAVs, our focus is on the multirotors. Multirotor can be defined as a rotorcraft with more than two motors. Based on this, the four most common types are tricopter, quadcopter, hexecopter and octocopter, each having 3, 4, 6 and 8 motors respectively. Each type of multirotor has its ups and downs - more motors mean higher liftforce and more reliable stability, but they also increase both price and damage caused in case of an error. Due to their price, size and ease of setup, quadcopters are the most popular type. They are popularly referred to as drones. Our goal for the project is to design a control system that makes it possible for the quadcopter to be stable - hovering mid-air - and to also act according to the user's input - maneuvering. Basically, our input will be a certain height and the quadcopter will have to automatically adjust to that height and maintain its stability when no further inputs are given. A safety feature - obstacle avoidance - will also be implemented.

Chapter 2

Prototype

This chapter will give a general overview on the pieces of technical equipement which we are using and show what the purpose of each one is. In addition, a diagram of the wiring of the components can be found in Appendice X.

2.1 Prototype Hardware

Motors

Controlling a quadcopter can be done efficiently by using high-quality motors with fast response, which will ensure more of a stable flight. The motors must also be powerful enough to be able to lift the quadcopter and perform the required aerial movements.

The motor that we are using is the Turnigy Multistar Brushless Motor seen in Figure 2.1.

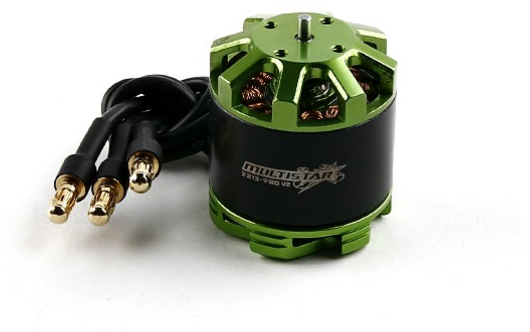

Figure 2.1: Turnigy Multistar 2213-980 V2 Brushless Motor.

Propellers

The propellers don't have such strict requirements as the motors. They are needed to be light and have a size and lift potential in order for the quadcopter to hover at less than 50% of the motor capacity. For our quadcopter, we are using plasting 10x4.5" propellers with light weight - 60g. They have a length of 254 mm and a pitch inclination of 114mm. They can be seen in Figure 2.2.

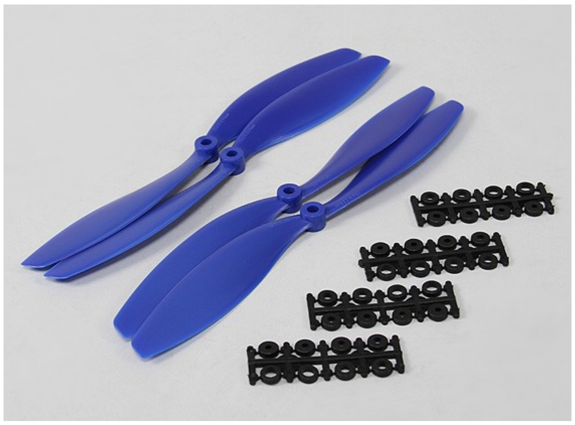

Figure 2.2: Hobbyking Slowfly Propeller 10x4.5.

Electric Speed Controller

Electronic Speed Controller (ESC) is a widely used device in rotorcrafts. The purpose of an ESC is to vary the electric motor's speed. They also come with programmable features, such as braking or selecting appropriate type of battery. We need the ESC to have a fast response, for the same reasons mentioned for the motors in Section **??**. The ESC that we are using is the TURNIGY Plush 30A which is shown in Figure 2.3.

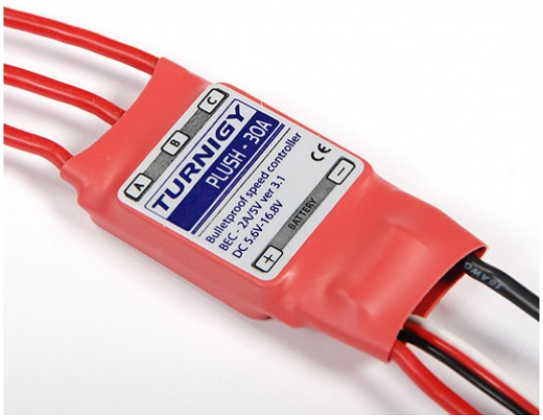

Figure 2.3: TURNIGY Plush 30A Speed Controller.

APM Flight Controller

ArduPilotMega (APM) is an open source unmanned aerial vehicle (UAV) platform which is able to control autonomous multicopters. It is illustrated in Figure 2.4. The system was improved uses Inertial Measurement Unit (IMU) - a combination of accelerometers, gyroscopes and magnetometers. The "Ardu" part of the project name shows that the programming can be done using Arduino open-source language.

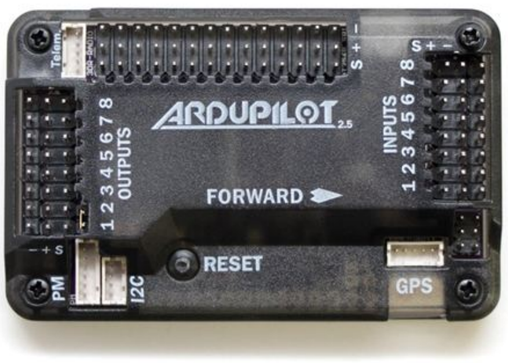

Figure 2.4: APM 2.5 board.

Power Distribution Board

To reduce the number of connections straight to the battery, we used the a power distribution board made for a previous project. A board like this is an easy solution since it enables us to connect the four ESCs directly to the board and then connect the board to the battery.

Battery

To power up our quadcopter, we will use a TURNIGY nano-tech Lipoly battery, which can be seen in Figure 2.5. Higher voltage under load, straighter discharge curves and excellent performance are the factors that make it suitable for our project.

Figure 2.5: Turnigy nano-tech 6000mah 3S 25 50C Lipo Pack.

2.2 Prototype Measurements

mass, lenght, height, moment of inertia

Chapter 3

Quadcopter Model

Quadcopter control is a complex, yet interesting problem. One of the reasons why this control problem is challenging is the fact that a quadcopter has six degrees of freedom, but only four inputs which affects the linearity of the dynamics and makes the quadcopter underactuated.

In this chapter we will take a look at the kinematics and dynamics equations that describe our quadcopter system for which we have drawn the block diagram shown in Figure .

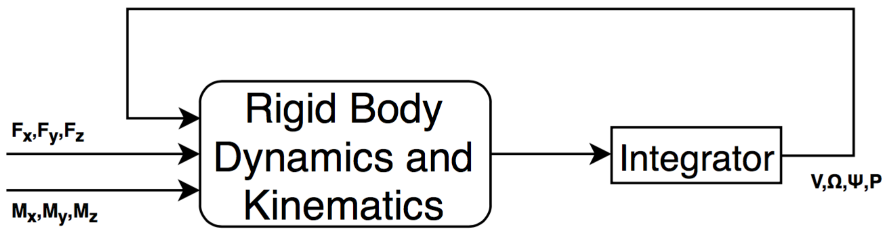

Figure 3.1: Dynamics and Kinematics Block Diagram

3.1 Quaternions and Euler angles

In Section , we have explained how the position of the quadcopter is expressed in the inertial frame and how the velocity of the quadcopter is express in the body-fixed frame. Therefore, we need to be able to find a relationship between the two, so that we can move from one to the other.

Euler angles have to be applied as a sequence of rotations. This report will use the *roll, pitch, yaw* convention. Therefore, the roll rotation will be $(R(\phi)^T)$, the pitch rotation will be $(R(\theta)^T)$ and the yaw rotation will be $(R(\psi)^T)$. Equations , and describe the quadcopter's orientation relative to the inertial frame in matrix form:

$$R(\phi)^T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\phi & sin\phi \\ 0 & -sin\phi & cos\phi \end{bmatrix} \tag{3.1}$$

$$R(\theta)^T = \begin{bmatrix} cos\theta & 0 & -sin\theta \\ 0 & 1 & 0 \\ sin\theta & 0 & cos\theta \end{bmatrix} \tag{3.2}$$

$$R(\psi)^T = \begin{bmatrix} cos\psi & sin\psi & 0 \\ -sin\psi & cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

Merging the three rotations as:

$$S = R(\phi)^T R(\theta)^T R(\psi)^T \tag{3.4}$$

gives the rotation matrix - S, which expresses a vector from the inertial frame to the body-fixed frame:

$$S = \begin{bmatrix} cos\theta cos\psi & cos\theta sin\psi & -sin\theta \\ cos\psi sin\phi sin\theta - cos\phi sin\psi & sin\psi sin\theta sin\phi + cos\phi cos\psi & cos\theta sin\phi \\ cos\psi cos\phi sin\theta & cos\phi sin\theta sin\psi - cos\psi sin\phi & cos\theta cos\phi \end{bmatrix} \tag{3.5}$$

We can notice that if we were to have $\theta = \Pi/2$, the rotation matrix brings out a singularity by turning into:

$$S = \begin{bmatrix} 0 & 0 & -1 \\ sin(\phi - \psi) & cos(\phi - \psi) & 0 \\ cos(\phi - \psi) & -sin(\phi - \psi) & 0 \end{bmatrix} \tag{3.6}$$

As a result, one degree of freedom in the three dimensional space is lost. In addition, a change in either ϕ or ψ will now have the same effect, which causes confusion. In order to be able to avoid this problem, a quaternion-based method can be applied instead.

A quaternion can be expressed as $q = [q_0, q_1, q_2, q_3]^T$, which yields:

$$q = \begin{bmatrix} cos(\phi/2)cos(\theta/2)cos(\psi/2) + sin(\phi/2)sin(\theta/2)sin(\psi/2) \\ sin(\phi/2)cos(\theta/2)cos(\psi/2) - cos(\phi/2)sin(\theta/2)sin(\psi/2) \\ cos(\phi/2)sin(\theta/2)cos(\psi/2) + sin(\phi/2)cos(\theta/2)sin(\psi/2) \\ cos(\phi/2)cos(\theta/2)sin(\psi/2) - sin(\phi/2)sin(\theta/2)cos(\psi/2) \end{bmatrix} \tag{3.7}$$

The quaternion and Euler angles are equivalent in terms of attitude, therefore we can convert from one representation to the other. The rotation matrix can be then rewritten as:

$$S_q = \begin{bmatrix} 1-2(q_2^2+q_3^2) & 2(q_1q_2+q_3q_0) & 2(q_1q_3-q_2q_0) \\ 2(q_1q_2-q_3q_0) & 1-2(q_1^2+q_3^2) & 2(q_2q_3+q_1q_0) \\ 2(q_1q_3+q_2q_0) & 2(q_2q_3-q_1q_0) & 1-2(q_1^2+q_2^2) \end{bmatrix} \tag{3.8}$$

3.2 Quaternions Representation

Knowing the position of the quadcopter relative to the inertial frame, the linear velocity in the body frame and the rotation matrix, a relationship between the three can be identified as:

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{bmatrix} = S_q^T \begin{bmatrix} U \\ V \\ W \end{bmatrix} \tag{3.9}$$

The quadcopter's attitude can be written using the angular velocities vector $\Omega^B = [P, Q, R]^T$:

$$\begin{bmatrix} \dot{q_0} \\ \dot{q_1} \\ \dot{q_2} \\ \dot{q_3} \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 0 & -P & -Q & -R \\ P & 0 & R & -Q \\ Q & -R & 0 & P \\ R & Q & -P & 0 \end{bmatrix} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \tag{3.10}$$

Because we chose to neglect all other forces acting on the quadcopter except of the propeller thrust and gravity, we can further express the forces produced by the propellers along the u_z axis as $[F_1, F_2, F_3, F_4]^T$. Therefore, the force in the body-fixed frame can be written as $F^B = [F_x, F_y, F_z]^T = [0, 0, -\sum_{i=1}^{4} F_i]^T$. Each force produces a moment around the axes of the quadcopter and as a result the moment in the body-fixed frame can be expressed as: $M^B = [M_x, M_y, M_z]^T$. These moments are explained in more detail in Chapter x.

The dynamics of the quadcopter in regards to the rotations are given by $I\dot{\Omega} = -\Omega \times I\Omega + M_B$, which yields:

$$\begin{bmatrix} \dot{P} \\ \dot{Q} \\ \dot{R} \end{bmatrix} = \begin{bmatrix} \frac{M_x}{I_x} \\ \frac{M_y}{I_y} \\ \frac{M_z}{I_z} \end{bmatrix} - \begin{bmatrix} \frac{(I_z-I_y)QR}{I_x} \\ \frac{(I_x-I_z)QR}{I_y} \\ \frac{(I_y-I_x)QR}{I_z} \end{bmatrix} \tag{3.11}$$

where $I = diag(I_x, I_y, I_z)$ is the inertia matrix.

Finally, by applying Newton's second law, we obtain $ma^B = F^B + mS_q g^I - \Omega \times V^B$, where $a^B = \dot{V}^B = [dotU, dotV, dotW]^T$ is the acceleration vector and $g^I = [0, 0, g_0]^T$ is the gravity vector with $g = 9.81 m/s^2$.

Therefore:

$$\begin{bmatrix}\dot{U}\\ \dot{V}\\ \dot{W}\end{bmatrix} = \frac{1}{m}\begin{bmatrix}F_x\\ F_y\\ F_z\end{bmatrix} + g_0\begin{bmatrix}2(q_1q_3 - q_2q_0)\\ 2(q_2q_3 + q_1q_0)\\ 1-2(q_1^2+q_2^2)\end{bmatrix} - \begin{bmatrix}QW-RV\\ RU-PW\\ PV-QU\end{bmatrix} \tag{3.12}$$

3.3 Euler Representation

In order to make the Euler angle rates correspond to the angular velocity vector, we can write:

$$\begin{bmatrix}P\\ Q\\ R\end{bmatrix} = R(\phi)^T R(\theta)^T R(\psi)^T \begin{bmatrix}0\\ 0\\ \dot{\psi}\end{bmatrix} + R(\phi)^T R(\theta)^T \begin{bmatrix}0\\ \dot{\theta}\\ 0\end{bmatrix} + R(\phi)^T \begin{bmatrix}\dot{\phi}\\ 0\\ 0\end{bmatrix} \tag{3.13}$$

Solving for Euler angles rates finally gives:

$$\begin{bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{bmatrix} = \begin{bmatrix}1 & tan\theta sin\phi & tan\theta cos\phi\\ 0 & cos\phi & -sin\phi\\ 0 & sin\phi/cos\theta & cos\phi/cos\theta\end{bmatrix}\begin{bmatrix}P\\ Q\\ R\end{bmatrix} \tag{3.14}$$

3.4 BLDC Motor

The mathematical model of a BLDC motor is is many ways similar to the one of a conventional DC motor. The main difference is represented by the added phases which affect the resistive and inductive components of the BLDC arrangement.

Therefore, we will start of by describing the mathematical modelling of a DC motor and then change it to fit the BLDC motor.

Figure illustrates a DC electromechanical system.

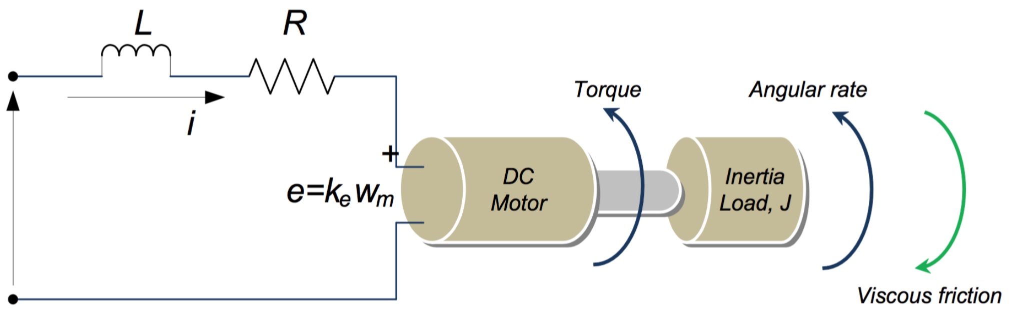

Figure 3.2: Typical DC electromechanical system

The components of the electrical circuit are the armature resistance - R, the armature inductance - L and the back EMF - e. By applying KVL, we obtain:

$$V_s = Ri + L\frac{di}{dt} + e \tag{3.15}$$

where V_s - DC source voltage and i - armature current.

Moving on to the mechanical part, further equations can be written using Newton's second law of motion:

$$J\frac{d\omega_m}{dt} = \Sigma T_i \tag{3.16}$$

$$T_e = k_f + J\frac{d\omega_m}{dt} + T_L \tag{3.17}$$

where T_e - electrical torque, k_f - friction constant, J - rotor inertia, ω_m - angular velocity and T_L - load torque.

The back EMF and the electrical torque can be described as:

$$T_e = k_t\omega_m \tag{3.18}$$

$$e = k_e \omega_m \tag{3.19}$$

where k_e - back EMF constant and k_t - torque constant.

Rewriting equations 3.15 and 3.16 gives:

$$\frac{di}{dt} = -i\frac{R}{L} - \frac{k_e}{L}\omega_m + \frac{1}{L}V_s \tag{3.20}$$

$$\frac{d\omega_m}{dt} = i\frac{k_t}{J} - \frac{k_f}{J}\omega_m + \frac{1}{J}T_L \tag{3.21}$$

Taking the Laplace transform of 3.20 and 3.21 yields:

$$si = i\frac{R}{L} - \frac{k_e}{L}\omega_m + \frac{1}{L}V_s \tag{3.22}$$

$$s\omega_m = i\frac{k_t}{J} - \frac{k_f}{J}\omega_m + \frac{1}{J}T_L \tag{3.23}$$

The next equation is obtained by substituting i from equation 3.23 into 3.22.

$$(\frac{s\omega_m + \frac{k_f}{J}\omega_m - \frac{1}{J}T_L}{\frac{k_t}{J}})(s + \frac{R}{L}) = -\frac{k_e}{L}\omega_m + \frac{1}{L}V_s \tag{3.24}$$

Assuming there is no load, equation 3.24 can be rewritten as:

$$\{(\frac{s^2 J}{k_t} + \frac{sk_f}{k_t} + \frac{sRJ}{k_t L} + \frac{k_f R}{k_t L}) + \frac{k_e}{L}\}\omega_m = \frac{1}{L}V_s \tag{3.25}$$

Solving 3.25 gives:

$$V_s = \frac{s^2 JL + sk_f L + sRJ + k_f R + k_e k_t}{k_t}\omega_m \tag{3.26}$$

The transfer function can be obtained as the ratio between the angular velocity ω_m and the source voltage V_s:

$$G_s = \frac{\omega_m}{V_s} = \frac{k_t}{s^2 JL + (RJ + k_f L)s + k_f R + k_e k_t} \tag{3.27}$$

Considering that k_f tends to zero, the transsfer function can finally be written as:

$$G_s = \frac{\omega_m}{V_s} = \frac{k_t}{s^2 JL + RJs + k_e k_t} \tag{3.28}$$

In order to obtain the mechanical and electrical time constants, we will need to manipulate equation 3.28, which gives:

$$G_s = \frac{\frac{1}{k_e}}{\frac{RJ}{k_e k_t}\frac{L}{R}s^2 + \frac{RJ}{k_e k_t}s + 1} \tag{3.29}$$

where the mechanical time constant is:

$$\tau_m = \frac{RJ}{k_e k_t} \tag{3.30}$$

and the electrical time constant is:

$$\tau_e = \frac{L}{R} \tag{3.31}$$

Therefore, substituting equations 3.30 and 3.31 into equation 3.29 yields:

$$G_s = \frac{\frac{1}{k_e}}{\tau_m \tau_e s^2 + \tau_m s + 1} \tag{3.32}$$

Equations 3.29 - 3.31 will now be changed in order to fit a BLDC motor. Therefore, they will become:

$$\tau_m = \Sigma \frac{RJ}{k_e k_t} = \frac{J \Sigma R}{k_e k_t} \tag{3.33}$$

$$\tau_e = \Sigma \frac{L}{R} = \frac{L}{\Sigma R} \tag{3.34}$$

Having a symmetrical arrangement and a three phase motor, the constants will finally be:

$$\tau_m = \frac{J3R}{k_e k_t} \tag{3.35}$$

$$\tau_e = \frac{L}{3R} \tag{3.36}$$

Taking into account the phase effects, τ_m will be rewritten as:

$$\tau_m = \frac{J3R_\phi}{k_e k_t} \tag{3.37}$$

with k_e now being the phase value of the back EMF voltage time constant described by:

$$k_e = k_{e(L-L)}/\sqrt{3} \tag{3.38}$$

A relationship between k_e - electrical torque and k_t - torque constant is found in equation 3.39by using the electrical power and mechanical power formulas.

$$k_e = k_t \times 0.0605 \tag{3.39}$$

The next step in coming up with a final transfer function for our motor is to find τ_m and τ_e, which are functions of k_e, k_t, R_ϕ, J and L.

A numerical value for k_e can be obtained using equation 3.38:

$$k_e = \frac{11.1}{1.73} = 6.416 \tag{3.40}$$

Since k_t is a function of k_e, its numerical value can also be found using equation 3.39:

$$k_t = \frac{6.416}{0.0605} = 106.04 \tag{3.41}$$

For R_ϕ, we have to measure the line-to-line resistance (between any two wires of the motor) and divide the result by 2 in order to get its value.

experiments on J and L / take them from other reports

3.5 Quadcopter Dynamics

To start describing drone's dynamics, we illustrate two frames - the inertial frame, which is defined by the ground, where the negative z direction shows represents gravity and the body frame. In the body frame, the motor axes point in the positive z direction and the arms point in the x and y directions. These two frames can be seen in figure 3.3.

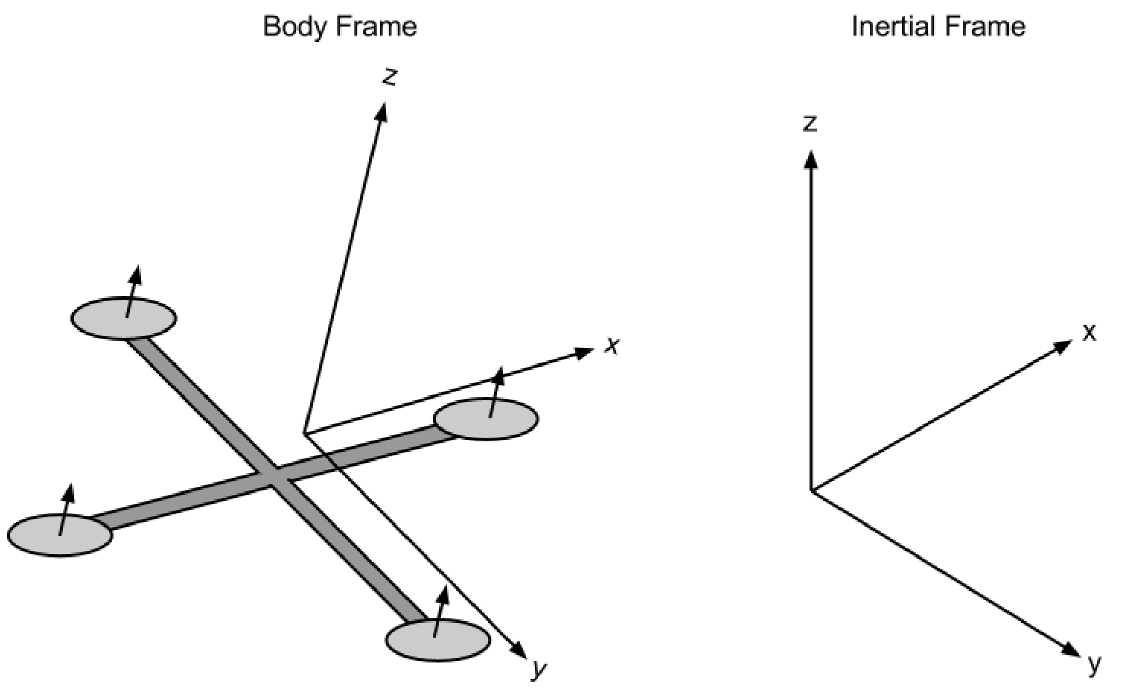

Figure 3.3: Quadcopter Body Frame and Inertial Frame

3.6 Kinematics

The drone's position in inertial frame is defined as $x = (x, y, z)^T$. Similarly, the velocity is $\dot{x} = (\dot{x}, \dot{y}, \dot{z})^T$. In the body frame, we have the roll, pitch and yaw angles given to be $\theta = (\phi, \theta, \Psi)^T$. Then, the angular velocities are equal to $\dot{\theta} = (\dot{\phi}, \dot{\theta}, \dot{\psi})^T$. It should be noted that the angular velocity vector ω is not equal to the angular velocity $\dot{\theta}$ itself. It is possible to convert angular velocity into angular velocity vector using the following expression:

$$\omega = \begin{bmatrix} 1 & 0 & -s_\theta \\ 0 & c_\phi & c_\theta s_\phi \\ 0 & -s_\phi & c_\theta c_\phi \end{bmatrix} \dot{\theta} \tag{3.42}$$

where ω is the angular velocity vector.

Both frames can be related using a rotation matrix R, which goes from one frame to another. The matrix is obtained using the ZYZ Euler angle conventions.

$$R = \begin{bmatrix} c_\phi c_\psi - c_\theta s_\phi s_\psi & -c_\psi s_\phi - c_\phi c_\theta s_\psi & s_\theta s_\psi \\ c_\theta c_\psi s_\phi + c_\phi s_\psi & c_\phi c_\theta c_\psi - s_\phi s_\psi & -c_\psi s_\theta \\ s_\phi s_\theta & c_\phi s_\theta & c_\theta \end{bmatrix} \tag{3.43}$$

Therefore, a vector $\overrightarrow{v}$ in the body frame can be defined as $R\overrightarrow{v}$ in the inertial frame.

3.7 Motor Dynamics

A brushless motor produces torque given by equation 3.44.

$$\tau = K_t(I - I_0) \tag{3.44}$$

Here, τ is the torque, I is the input current, K_t is the constant describing torque proportionality and I_0 is the no-load current.

The voltage on the motor is the sum of a resistive loss and the Back-EMF, giving equation 3.45.

$$V = IR_m + K_e\omega \tag{3.45}$$

where V is the voltage across the motor, R_m is the motor resistance, ω is the angular velocity and K_e is a constant, describing the back-EMF generated per RPM. We can then combine equations 3.44 and 3.45 and put them into the power equation, to get a new equation 3.46.

$$P = IV = \frac{(\tau + K_t I_0)(K_t I_0 R_m + \tau R_m + K_t K_e \omega)}{K_t^2} \tag{3.46}$$

To help simplify the model, we can assume that the motor resistance is very small and therefore can be disregarded. This gives us a new power equation 3.47.

$$P \approx \frac{(\tau + K_t I_0) K_e \omega}{K_t} \tag{3.47}$$

We can also, for the purposes of simplifying the model, assume that $K_t I_0 \ll \tau$. This is a reasonable assumption due to the fact that I_0 describes the current with no load on the motor and is usually a small number. Based on this, we can then obtain the final equation 3.48 for power.

$$P \approx \frac{K_e}{K_t} \tau \omega \tag{3.48}$$

3.8 Forces

From the conservation of energy, it is known that the energy that the motor expends during some period of time is equal to the force generated by the propeller multiplied by the distance the it displaces moves, given by the equation 3.49.

$$P \times dt = F \times dx \tag{3.49}$$

We can then say that the power is also equal to the thrust multiplied by the air velocity, as seen in equation 3.50.

$$P = F\frac{dt}{dx} = Tv_h \tag{3.50}$$

Here - T is the thrust and v_h is the air velocity. Since the model is describing a hovering quadcopter in a closed area, only v_h is an acting velocity (i.e. vehicle speed has no effect and free stream velocity is equal to zero).

Momentum theory gives equation 3.51, for the hover velocity v_h.

$$v_h = \sqrt{\frac{T}{2\rho A}} \tag{3.51}$$

where A is the area swept out by the blade and ρ is the air density. In the case of this model, τ is proportional to thrust T by a constant K_τ. Keeping this in mind, we can now use the simplified power equation 3.48 and derive a new equation 3.52.

$$P = \frac{K_e}{K_t}\tau\omega = \frac{K_e K_\tau}{K_t}T\omega = \frac{F^{\frac{3}{2}}}{\sqrt{2\rho A}} \tag{3.52}$$

Solving for thrust T, we can derive a new equation 3.53.

$$T = (\frac{K_e K_\tau \sqrt{2\rho A}}{K_t}\omega)^2 = k\omega^2 \tag{3.53}$$

where k is a constant, describing all other constants used in the equation.

The final equation 3.53 now describes the thrust T generated by the motor with angular velocity ω as an input.

We can now sum the thrust over all motors. This sum is then given by

$$T_B = \sum_{i=1}^{4} T_i = k \begin{bmatrix} 0 \\ 0 \\ \sum \omega_i^2 \end{bmatrix} \tag{3.54}$$

We can model a simplified version of fluid friction as a force proportional to the linear velocity in each direction. Then, drag forces will have another force term in their model:

$$F_D = \begin{bmatrix} -k_d\dot{x} \\ -k_d\dot{y} \\ -k_d\dot{z} \end{bmatrix} \tag{3.55}$$

3.9 Torques

Every motor produces some torque about the z axis in the body frame. This torque is the requirement for the torque produces by the motor to keep the motor providing thrust by spinning the propeller. The spinning propeller overcomes frictional forces and creates angular acceleration. The frictional force is acquired from fluid dynamics as follows:

$$F_D = \frac{1}{2}\rho C_D A v^2 \tag{3.56}$$

Here, ρ is the fluid density, A is the propeller cross-section area, C_D is a dimensionless constant. We can then derive an equation for torque produced due to drag force:

$$\tau_D = \frac{1}{2}R\rho C_D A v^2 = \frac{1}{2}R\rho C_D A(\omega R)^2 = b\omega^2 \tag{3.57}$$

where b is an appropriately dimensioned constant. While we assume that force all of the force is applied at the top of the propeller, our only concern is that the drag torque is proportional to the angular velocity. Torque for the i^{th} motor about the z axis can then be described as follows:

$$\tau_z = (-1)^{i+1}b\omega^2 + I_M\dot{\omega} \tag{3.58}$$

Here, I_M is the moment of inertia about the z axis of the motor, $\dot{\omega}$ is the angular acceleration of the propeller and the term $(-1)^{i+1}$ is describing the direction of rotation of the i^{th} motor (i.e. whether it is spinning clockwise or counter-clockwise). Since in a steady state there is no acceleration, we can say that $\dot{\omega} \approx 0$ and drop the term. The simplified equation then becomes

$$\tau_z = (-1)^{i+1}b\omega^2 \tag{3.59}$$

Then, we can derive an equation for total torque about the z axis.

$$\tau_\psi = b(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \tag{3.60}$$

If we select motors 1 and 3 to be on the roll axis, we can the following roll torque equation:

$$\tau_\phi = \sum r \times T = L(k\omega_1^2 - k\omega_3^2) = Lk(\omega_1^2 - \omega_3^2) \tag{3.61}$$

Here, L defines the distance from the propeller to the centre of the quadcopter, assuming all motors are placed at equal distances. Similarly, we can obtain the pitch torque for the other two motors:

$$\tau_\theta = Lk(\omega_2^2 - \omega_4^2) \tag{3.62}$$

A matrix describing all torques in the body frame can be seen in equation 3.63.

$$\tau_B = \begin{bmatrix} Lk(\omega_1^2 - \omega_3^2) \\ Lk(\omega_2^2 - \omega_4^2) \\ b(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{bmatrix} \tag{3.63}$$

3.10 Equations of Motion

Quadcopter's acceleration in the inertial frame is caused by gravity, thrust and linear fircion. We can obtain thrust vector in the inertial frame through the use of rotation matrix R to map thrust vector from one frame to the other. Linear motion can then be described by equation 3.64.

$$m\ddot{x} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} \tag{3.64}$$

Since we prefer to express rotations about the centre of the vehicle instead of about inertial centre, it is more reasonable to use rotational equations of motion in body frame. We can derive these equations from Euler's equations for rigid body dynamics. The equation 3.65 is expressed in vector form.

$$I\dot{\omega} + \omega \times (I\omega) = \tau \tag{3.65}$$

It can then be rewritten as equation 3.66.

$$\dot{\omega} = \begin{bmatrix} \dot{\omega_x} \\ \dot{\omega_y} \\ \dot{\omega_z} \end{bmatrix} = I^{-1}(\tau - \omega \times (I\omega)) \tag{3.66}$$

A drone model can be understood as two lines crossed together at origin with a point mass as motors at the end of each line. From this, we can derive a diagonal inertia matrix 3.67.

$$I = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} = I^{-1}(\tau - \omega \times (I\omega)) \tag{3.67}$$

Taking everything into consideration, a final result for the rotational equations of motion in body frame are given as equation 3.68.

$$\dot{\omega} = \begin{bmatrix} \tau_\phi I_{xx}^{-1} \\ \tau_\theta I_{yy}^{-1} \\ \tau_\psi I_{zz}^{-1} \end{bmatrix} - \begin{bmatrix} \frac{I_{yy}-Izz}{I_{xx}}\omega_y\omega_z \\ \frac{I_{zz}-Ixx}{I_{yy}}\omega_x\omega_z \\ \frac{I_{xx}-Iyy}{I_{zz}}\omega_x\omega_y \end{bmatrix} \tag{3.68}$$

3.11 Control

Having a mathematical model of the system makes the development of controller easier. With only input being the angular velocities of each motor, we can write the system as a first order differential equation in state space.

$$\dot{x_1} = x_2 \tag{3.69}$$

$$\dot{x_2} = \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + \frac{1}{m} R T_B + \frac{1}{m} F_D \tag{3.70}$$

$$\dot{x_3} = \begin{bmatrix} 1 & 0 & -s_\theta \\ 0 & c_\phi & c_\theta s_\phi \\ 0 & -s_\phi & c_\theta c_\phi \end{bmatrix}^{-1} x_4 \tag{3.71}$$

$$\dot{x_4} = \begin{bmatrix} \tau_\phi I_{xx}^{-1} \\ \tau_\theta I_{yy}^{-1} \\ \tau_\psi I_{zz}^{-1} \end{bmatrix} - \begin{bmatrix} \frac{I_{yy} - Izz}{I_{xx}} \omega_y \omega_z \\ \frac{I_{zz} - Ixx}{I_{yy}} \omega_x \omega_z \\ \frac{I_{xx} - Iyy}{I_{zz}} \omega_x \omega_y \end{bmatrix} \tag{3.72}$$

Here, x_1 is the position of the vehicle in space, x_2 is the drone's linear velocity, x_3 denotes the angles of pitch, roll and yaw and x_4 shows the angular velocity vector.

Chapter 4

Sensors

4.1 Model of the sensors

This chapter will, step by step, introduce the sensors, describe how they function and define the methods used to extract the data from them. The accelerometer and gyroscope that we are using are both part of the built-in IMU of the Ardupilot, fitted on 6-axis MPU6000 14-bit chip. The accelerometer works by detecting a force that is actually the opposite of the acceleration vector. This force is not always caused by acceleration, but it can be. It just happens that acceleration causes an inertial force that is captured by the force detection mechanism of the accelerometer. The gyroscope measures the rotation around one of the axes.

Accelerometer

To measure the direction of the gravity vector and define the pitch and roll angles, an accelerometer is an easy to use solution. The raw ADC values obtained from the accelerometer first have to be converted into some acceleration m/s^2. To make this happen, a resolution equation is defined that will describe the relationship between the raw values and the acceleration. For this particular accelerometer, some general acceleration a_θ along some axis θ can be found using equation 5.1.

$$a_\theta = \frac{ADC_\theta * g}{bits - 1} \tag{4.1}$$

where ADC_θ is the raw value measured by the sensor, g is the gravitational force and $bits$ is the number of bits the module is running at. The values measured by the accelerometer will be stored in vector $\bar{a}^B = [\bar{a}_x, \bar{a}_y, \bar{a}_z]^T$. Additionally, these values will be converted into degrees. This process will be discussed in the programming section.

Gyroscope

The gyroscope has some similar functionality as the accelerometer, being able to measure the angular velocities expressed as $°/s$. The equation to convert raw gyroscope values into angular velocity will be covered in the programming section as well. The measured values of gyroscope will be stored in vector $\bar{\Omega}^B = [\bar{g}_x, \bar{g}_y, \bar{g}]_z^T$.

4.2 Programming Accelerometer and Gyroscope

It is important to understand how the sensors can actually deliver us the data. Usually, they fall into two categories: analogue and digital. The one we are using is digital and it can be programmed using I2C, SPI or USART communication.

//insert lines of code that makes the spi communication possible

Before getting into the calculations of the angles, we need to get the raw values from both accelerometer and gyroscope and that is done by accessing their registers.

//insert lines of code that read the raw values from acc and gyro

If we want to calculate the inclination of a device relative to the ground for example, we can calculate the angle between the force vector and one of the z-axis. We can do that by manipulating the raw values and by degree conversion. The functions that take care of converting the raw values from both accelerometer and gyroscope are the following:

//insert lines of code for the math part

We have used the Serial Monitor within the Arduino IDE environment to read the outputs from both sensors and they seem to be accurate. The difference in the values of the accelerometer output is because accelerometers are more sensitive to noise and vibrations. The figure below shows the printed outputs at steady state, that is when the quadcopter is parallel to the ground.

//insert printscreen of the values

//mention sensitivity

Chapter 5

Actuators

Actuators are an important part of any control system because they are the ones responsible for bringing it to the desired state. They do this by applying forces on the system. In our case, the actuators are the motor and the propellers. Figure x displays the actuators' configuration. Ardupilot, which is the "brain" of our control system and has the controller implemented on it is connected to the speed controller, which is in charge of controlling the motor through a PWM signal.

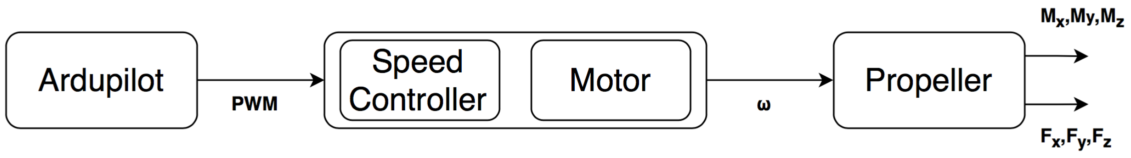

Figure 5.1: Actuators Block Diagram.

Each motor produces an angular velocity ω, therefore the propeller spins at that defined angular velocity. In this chapter, we will be introducing more comprehensible models for both motor and propeller as well as some experiements regarding the PWM signal - motor speed relationship.

5.1 Propeller model

For this part of the report, we have decided not to take into account the aerodynamic forces that act on the propeller, therefore neglecting the air friction, the flapping of the blased and the ground effect. This enables us to create a simpler model, which is easy to understand and implement.

The thrust F_i and the moment across the M_i along the u_z axis can be written as functions of angular velocities such as:

$$F_i = K_T\omega_i^2 \tag{5.1}$$

$$M_i = K_M \omega_i^2 \tag{5.2}$$

where the constant related to thrust K_T and the constant related to the moment K_M can be describes as:

$$K_T = c_T \frac{\rho D^4}{4\Pi^2} \tag{5.3}$$

$$K_M = c_P \frac{\rho D^5}{8\Pi^2} \tag{5.4}$$

with D= being the diameter of the propeller, ρ being the air density, c_T and c_P being the thrust and power coefficient. We are considering these values to have small values (<0.2) based on what we have read from other projects.

Therefore:

$$K_T \approx 1.45 \times 10^{-5} \tag{5.5}$$

$$K_M \approx 3.5 \times 10^{-7} \tag{5.6}$$

We can write $K \approx K_M/K_T = 4.1 \times 10^{-7} = 0.024$. The forces and moments can now be expressed as:

$$F_i = K_T \omega_i^2 \tag{5.7}$$

$$M_x = (\omega_2^2 - \omega_4^2) K_T D = (F_2 - F_4) D \tag{5.8}$$

$$M_y = (\omega_1^2 - \omega_3^2) K_T D = (F_1 - F_3) D \tag{5.9}$$

$$M_z = (\omega_1^2 + \omega_3^2 - \omega_2^2 - \omega_4^2) K_M = (F_1 + F_3 - F_2 - F_4) K \tag{5.10}$$

where D is expressed in centimeters.

5.2 Motor model

Providing a signal to the motor is done through Ardupilot, using PWM, which enables providing an analog singal by digital means. In order to create a PWM signal, we have used the *write.Microseconds()* function from the*Servo.h* library withing the Arduino IDE environment. The role of the speed controllers is to turn the PWM singal into a three-phase signal to feed the BLDC motors.

Each motor receives a three-phase signal that is proportional with the angular velocity ω. In general, the behaviour of a motor can be analysed by looking at the electrical and mechanical part of its structure. These parts are represented in Figure .

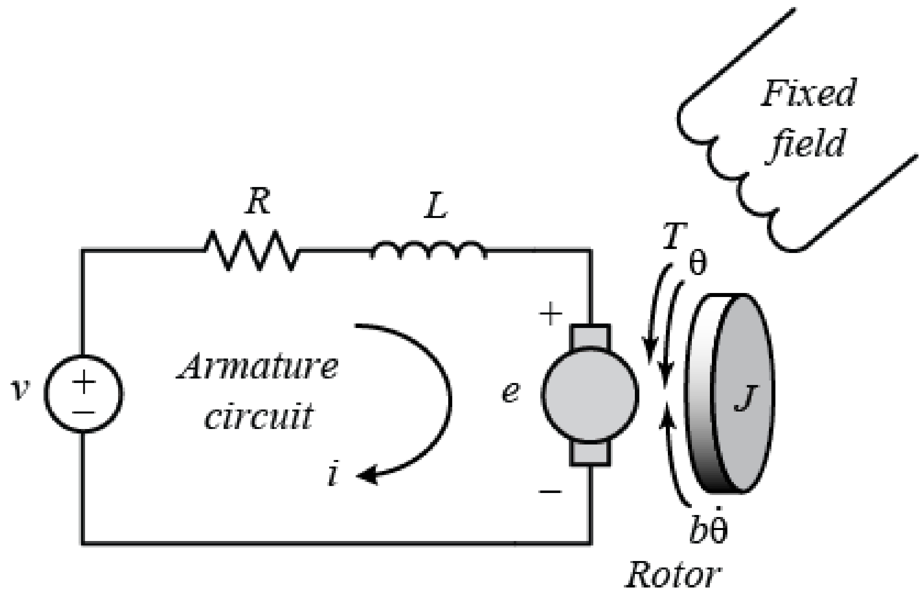

Figure 5.2: Electrical and Mechanical part of the motor.

A second order system can be obtained by writing the equations that describe the electrical and mechanical part as:

$$V = RI + L\frac{dI}{dt} + K_e\dot{\theta} \tag{5.11}$$

$$K_eI = J\ddot{\theta} + b\dot{\theta} \tag{5.12}$$

The transfer function can be taking the Laplace transform of each equation and manipulating it into:

$$\frac{\omega}{PWM} = \frac{K_e}{(Js+b)(Ls+R)+K_e^2} \tag{5.13}$$

where ω is the angular velocity of the motor, PWM is the signal to the motor, K_e is the electromotive constant, J is the rotor's moment of inertia, b is the damping ratio of the mechanical system, L is the inductance, R is the resistance and I is the measuremement of the current.

Equation can be further simplified and written as a first order system, since it is difficult to measure all the motor parameters:

$$\frac{\omega}{PWM} = \frac{k_i}{\tau s + 1} \tag{5.14}$$

where τ is the time constant of the system and k_i is the DC gain.

5.3 Motor Identification

This section will show the experiements that we have carried out in regards to the motor.

5.3.1 Expected and Real Motor Performance

The motors used in the prototype specify to be rated at K_v of 980. K_v is a constant describing the ration between RPM and the applied voltage and is expressed as $K_v = \frac{RPM}{V}$. Derived from this, voltage's effect on the RPM can be seen in figure 5.3.

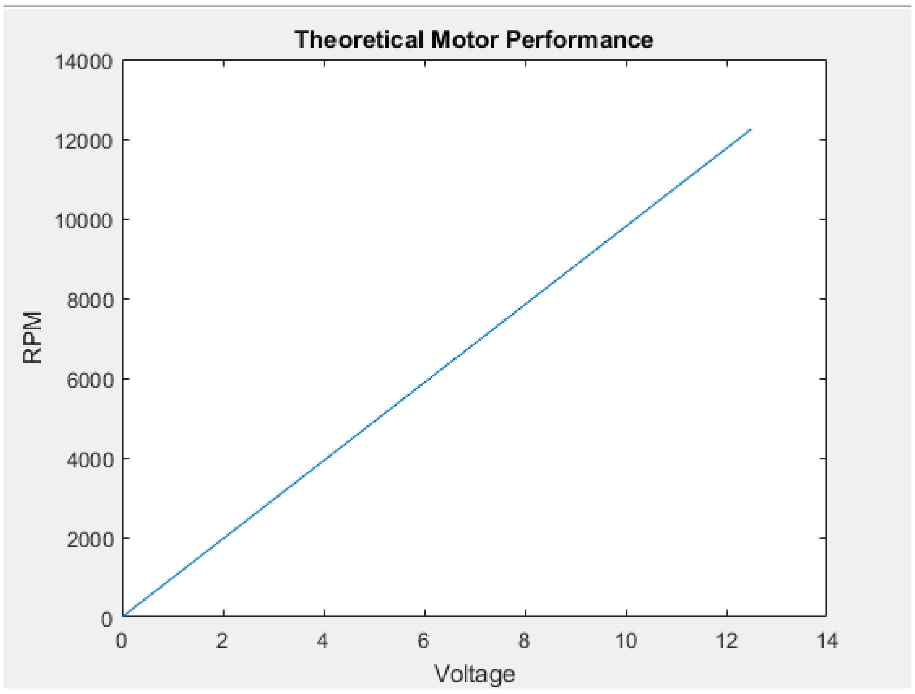

Figure 5.3: Expected motor performance

With a fully charged battery, the RPM is expected to be $980 \times 11.1V = 10878RPM$.

In order to confirm this, the actual RPM was measured using SHIMPO DT-205 digital tachometer. A piece of reflective paper was taped to one of the motors so that the tachometer would have something to lock onto, as seen in figure 5.4.

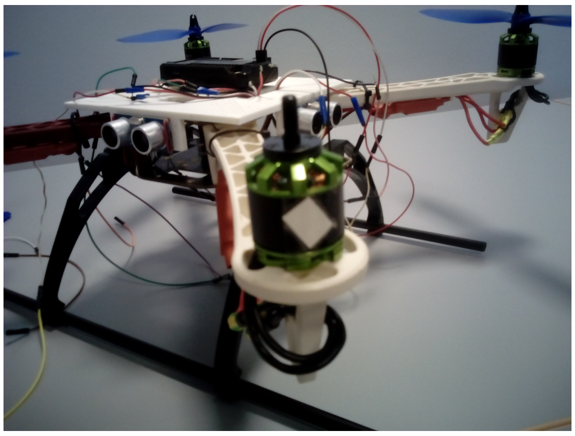
Figure 5.4: Reflective paper on the motor

By sending the maximum signal from the board, the RPM was measured to be 10812, with a small possible error. In conclusion, the motors' K_v coefficient found in the datasheet is in fact correct.

5.3.2 APM Output and Motor Speed Relationship

In order to use the board's output signal as an input in a mathematical model, it is necessary to find an equation to translate it into an angular rate. First, an equation has been found that describes the relationship between motor's RPM, ESC's output signal frequency and number of motor poles [3]:

$$RPM = \frac{120 \times f}{n} \tag{5.15}$$

Here, f is the frequency and n is the number of poles (in the case of this project - 14).

Therefore, it is possible to measure the frequency at various output values, which can then be used to define an equation that uses the board's signal length as an input value and results in an RPM.

The frequency was measured using an (NAME) oscilloscope and by connecting the probe's ground clip to the ground of the battery and the probe tip to any one of the wires between motor and the ESC. Then, by providing different output signals from the flight controller, the frequency was measured on the oscilloscope screen. An on-board filter was used to filter out some noise, especially at lower output signals. The recorded frequency was later converted into RPM using equation 5.15, for later comparison. The output signal lengths and the corresponding frequencies converted into RPM can be seen in table 5.1.

$Output, \mu s$	RPM
762	2825
770	3189
780	3689
790	4211
800	4683
810	5161
820	5546
830	5874
840	6223
850	6532
860	6795
870	7008
880	7301
890	7534
900	7730
925	8233
950	8552
975	8786
1000	9129
1050	9566
1100	9814
1150	10071
1200	10260

Table 5.1: Frequency measured at ESC output, converted into RPM

While measuring, it was observed that the frequency would never reach a steady-state and therefore, a sizeable error is possible between measurements. Because of that, RPM was manually measured using the tachometer at same output values, as seen in table 5.2.

$Output, \mu s$	RPM
762	2754
770	3170
780	3664
790	4210
800	4715
810	5223
820	5617
830	5965
840	6285
850	6578
860	6846
870	7100
880	7368
890	7555
900	7790
925	8265
950	8614
975	8910
1000	9160
1050	9576
1100	9860
1150	10089
1200	10261

Table 5.2: RPM measured at different values

While measuring with tachometer, the error appeared to be less significant.

The two tables 5.2 and 5.1 were then plotted to see the difference, which is seen in figure 5.5.

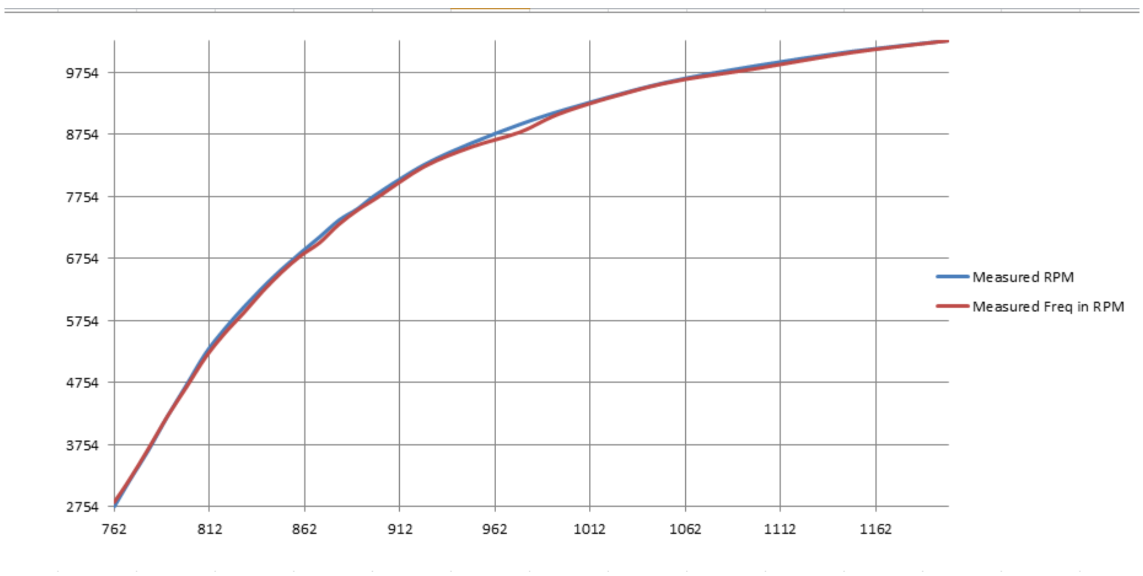

Figure 5.5: Plotted values of manually measured and frequency-based RPM

From the plotted graphs, it can be seen that the difference is very much noticeable, especially at certain points. Due to the fact that measurements with tachometer provided smaller errors than measurements with an oscilloscope, it was decided to use the values measured with the tachometer for later calculations.

In order to make use of these findings, the RPM values have to be converted into rad/s for use as angular rate. The conversion equation is:

$$\omega = \frac{2\pi \times RPM}{60} \tag{5.16}$$

The values measured with tachometer were then converted into rad/s and plotted in Matlab. Then, by utilising the *polyfit* function, a 2nd order polynomial equation fitting the original graph was obtained. The equation seen in Figure can then be used as an approximation for the conversion of the sent signal in μs into angular rate.

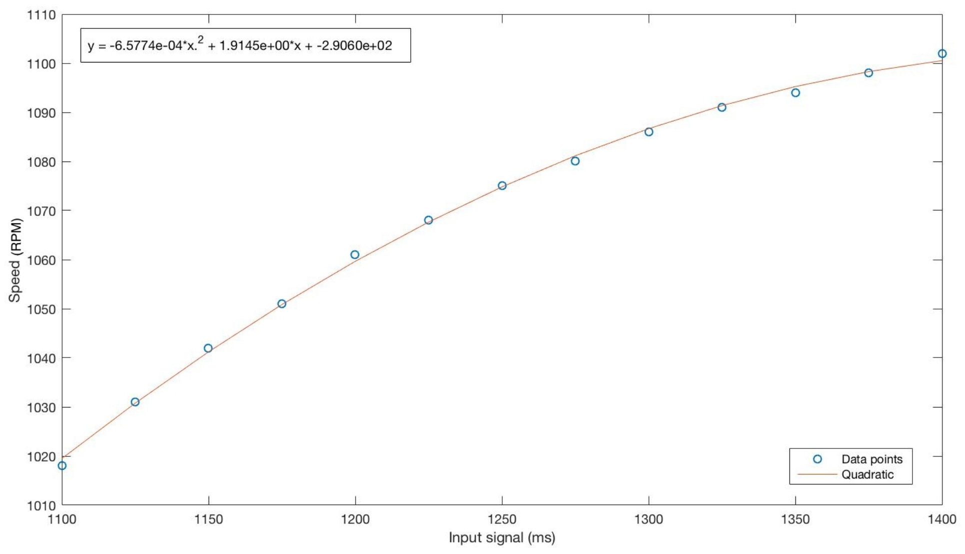

Figure 5.6: Converted and plotted RPM values and the trendline

5.4 Electronic Speed Controllers

...

5.4.1 Calibrating and Programming ESCs

ESCs have 5 input pins, 3 of which come from the flight controller. These 3 wires supply the signal for the ESC to translate into the angle of the shaft for the motor. Before ESCs can be used, they need to be properly calibrated. Programming additional settings is optional, but in most cases necessary too. Normally, the calibration for UAVs is done by setting the throttle to full on the radio controller before powering the ESCs. However, since the board translates the throttle signal into a PWM signal, it is possible to calibrate and run the motors directly from the flight controller by sending a PWM signal using software.

The calibration is done by first sending the maximum length of signal the user wishes to use. The ESC emits sounds that are brand-specific and indicate whether the signal was successfully recognised or not. Once the maximum signal is accepted, the user the emits the minimum signal and waits for approval. Additional sound is then emitted, indicating that the calibration was successful.

Using Arduino's Servo library, a self-explanatory built-in function *servo.writeMicroseconds(int value)* is used to send the signal. By default, the library has the minimum signal set to 544μs and the maximum - to 2400μs. For this project, we shortened the range down to 700μs and 2000μs for both signals respectively. A commented code used to calibrate the ESC connected to the first pin of the board can be seen in listing 5.1.

```
#define MAX_SIGNAL 2000 //define max and min signal length
#define MIN_SIGNAL 700
#define MOTOR_PIN1 12 //pin 1 on the board
Servo motor1;
void setup() {
  Serial.begin(9600); //begin serial communication at 9600 rate
  Serial.println("Program begin...");
  motor1.attach(MOTOR_PIN1); //attach the pin to the servo variable
  Serial.println("Now writing maximum output.");
  Serial.println("Turn on power source, then wait 2 seconds and press any key.");
  motor1.writeMicroseconds(MAX_SIGNAL); //this signal should only be sent when calibrating for the first time
  while (!Serial.available()); //wait for an input, so you have time to plug in the ESC
  Serial.read();
  Serial.println("Sending minimum output");
  motor1.writeMicroseconds(MIN_SIGNAL); //send the min signal, which is the only needed signal when already calibrated
  while (!Serial.available()); //wait for the beeps to indicate calibration is done
  Serial.read();
  Serial.println("Calibrated");
}

void loop() {

}
```

Code Listing 5.1: Calibrating the ESC

Programming additional settings of the ESC is done in a similar manner - by sending minimum and maximum signals after the ESC emits particular sounds, indicating wanted selections. The programmable features and selections are brand-specific and can be found in the datasheets. For the purposes of this project, the ESCs have been programmed to include two features - brake mode off and selection of li-ion battery.

Chapter 6

State space model

Chapter 7

Experiments

7.0.1 Output Operating Range

Since there is no direct translation between the output signal from the board and the speed of the motor and the ESC allows more current than the motor can recognise, it is necessary to find the operating range for the board output that has the most noticeable effect on the RPM of the motor. Through trial and error and by measuring the RPM using SHIMPO DT-205 digital tachometer. The lowest operating point was found to be $762\mu s$, at which the motor finally begins to spin. The highest point was defined at $1200\mu s$. Higher values were found to have very small effect on the RPM. Therefore, despite the ESC recognising the values between 700 and $2000\mu s$, the most noticeable effect will be between 762 and $1200\mu s$ and will be primarily used in the project.

7.0.2 APM Frequency

Due to lack of proper documentation, it was necessary to do measure the frequency of the signals sent out by the flight controller. To do so, an oscilloscope was connected to one of the output pins of the board. Then, using the servo library, a signal was sent out. The interval between signals was found to be $20ms$, therefore, the frequency of the board is $50Hz$, as seen in figure 7.1.

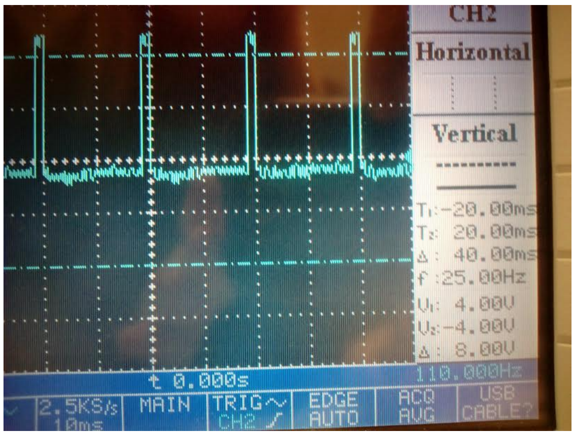

Figure 7.1: Oscilloscope measuring APM's frequency.

The second experiment on the board was then made to determine how the flight controller handles the output signals during those $20ms$. First two outputs of the APM were connected to the oscilloscope, both utilizing the Servo library to send signals of length of $2000\mu s$. Results can be seen in figure 7.2.

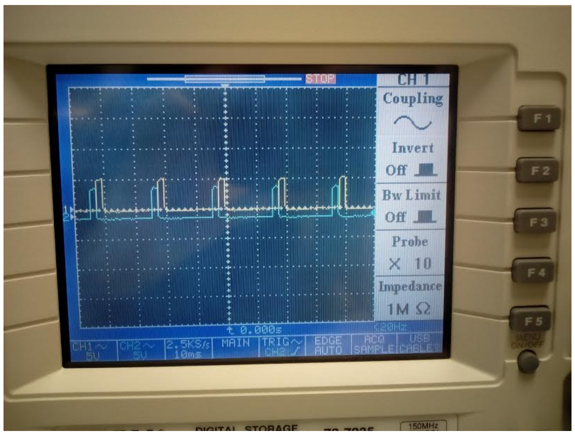

Figure 7.2: Readings of the two output signals.

Then, for further testing purposes, both outputs were given different values in two scenarios, as seen in figure 7.3.

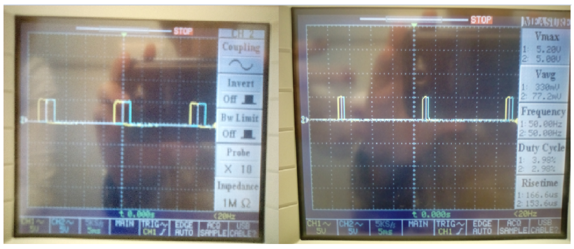

Figure 7.3: Left - signals running at $2000\mu s$; Right - $700\mu s$.

From this, two conclusions can be made:

1. If the first signal is shorter than the second one, the second signal will still follow right after the first signal ends. In other words, the APM leaves no gaps between the outputs.
2. Since the board runs at the frequency of $50Hz$ and has a period of $20ms$, this leaves $\frac{20}{8} = 2.5ms$ maximum length for each output signal. The servo library is hard-capped at $2.4ms$ and thus is well within the limits of the board.

Chapter 8

Discussion

Chapter 9

Conclusion

Bibliography

[1] Nancy Hall. Simplified airplane motion. https://www.grc.nasa.gov/www/k-12/airplane/smotion.html. (Accessed 06/11/16).

[2] Alex. The quadcopter : control the orientation. http://theboredengineers.com/2012/05/the-quadcopter-basics/. (Accessed 06/11/16).

[3] Vfds how do i calculate rpm for three phase induction motors? http://www.precision-elec.com/faq-vfd-how-do-i-calculate-rpm-for-three-phase-induction-motors/. (Accessed 19/11/16).

Chapter 10

Appendix

10.1 Appendix code