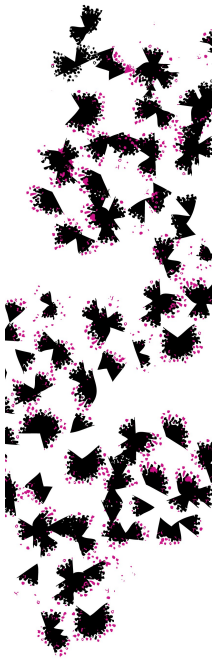


Design and Empirical study of tilting propellers of an over-actuated quadrotor

B. (Boi) Okken

BSc Report

Committee:
Dr.ir J.B.C. Engelen
R. Hashem, MSc
Dr.ir. P.C. Breedveld
Dr. L.D. de Santana

July 2017

025RAM2017
Robotics and Mechatronics
EE-Math-CS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Abstract

An over-actuated quadrotor that can track arbitrary position and orientation trajectories in space has several benefits for interactive service robotics. For construction of an over-actuated quadrotor a robust, lightweight tilting system for the propulsion motors needs to be designed. This report details the design, construction and measurements of a tilting rotor design, also investigating empirically whether secondary aerodynamic forces are of negligible impact when compared to a non-tilting design. Different system components of the measurements setup are analysed for the use of this design. Lastly, modelling parameters were extracted for later implementation in a computer model and construction of a control system.

Keywords - Aerial robotics, Aerodynamical modelling, Over-actuation, Quadrotor unmanned aerial vehicles (UAVs), aerodynamics, System identification

Index

1 Introduction

1.1 Context and Project background

SPECTORS is an innovation program involving 20 companies and education facilities. The goal of the project is to exploit the existing commercial drone technology market using improvements and innovations in the field of civilian drones.
The task of the Robotics and Mechatronics group of the University of Twente within the SPECTORS project, is to design an aerial robot that can achieve robust contact with a mechanical surface of a structure for inspection and interaction.

As an example, inspection on wind turbines used to be done by sending someone up the wind turbine, and making the person descend by rope, doing the visual inspection and using tools if necessary. This has changed in recent years with most visual inspection being done by drones. This is cheaper, safer and faster. However, for detection of for example hairline cracks, measurement equipment needs to be applied to the turbine surface. Currently this is still being done by sending a person up with a rope to physically work with the surface. This is because of the inherent problem of a quad rotor, namely that it is an under actuated platform, capable to move in 6 degrees of freedom, but cannot orient itself in 6 degrees of freedom with the 4 degrees of freedom as steering inputs. i.e. a quadcopter cannot apply force in the horizontal plane, and thus cannot consistently and for a longer period of time apply force to the vertical wind turbine surface.

The proposed solution of the project is to add more degrees of freedom to the flying platform by tilting the rotors of the quad copter. This would over actuate the platform, making it able to also apply (consistent) pressure in more than just the vertical plane, increasing the capabilities of the drone not only in wind turbine inspection, but in all types of work that require a pressure in non-vertical direction to be applied in a dangerous environment. Of course the development of such a drone is not limited to just inspection and interaction, but also for movement within constrained corridors. As an example think of a doorway and a flying platform of 1.5 m width. It usually would not fit through the doorway, but if it can orient itself in 6 degrees of freedom it could fit through with ease.

As a contribution to the SPECTORS project, drones with a tilting propeller architecture will be analysed and designed.

1.2 Problem statement

There are a lot of secondary aerodynamic forces acting upon the quadrotor and its motors. In a normal quadrotor these are considered negligible. Are these forces negligible, and can they be considered negligible for a tilting rotor system as well?

For a tilting rotor quadrotor, a design has to be made for tilting the rotors. This design has to be used for the investigation of said secondary aerodynamic forces. Preferably the design should also be able to be used within a flying prototype drone. Can a robust, easy to produce and easy to use tilting system be designed?

As a secondary problem objective, it would be beneficial to have parameters that can be used for modelling a tilting drone in software. Can model parameters be extracted from the retrieved data?

1.3 Project goals

The goal of my contribution within the project, is the analysis of existing tilting rotor mechanisms, and design of an easy to use and produce, robust tilting mechanism for measurements and preferably a flying prototype.

With the use of this tilting rotor mechanism, an investigation will be carried out whether the secondary aerodynamic forces can be neglected in a tilting rotor system when compared to a non tilting rotor system, and to what extent they can be neglected.

As a secondary goal, the extraction of modelling parameters useful in later computer modelling of an over-actuated quadrotor is set. This includes investigating that if propellers can be reversed, if the dynamic response is the same when reversing direction.

To answer these questions, two different measurement set-ups will be constructed, one to make measurements on the brushless DC motor without any tilting mechanism, and a measurement setup with a tilting system.

As for the final testing set-up, the following functions should be achieved:

Must have:

- *Measure torque and force induced by the BLDC via an arm or known length*
- *Control BLDC motor by PWM percentage*
- *A variable tilting mechanism with servo as control*
- *Be able to (independently) control BLDC and servo states according to a script*
- *Solid motor mounting*
- *Use both existing batteries and power supply for power*

Should have:

- *Robust tilting mechanism ready to implement on a flying craft*
- *Run of a single power source*
- *Ability to reverse propeller direction*
- *Measure RPM*

Could have:

- *Control motor by RPM*
- *Control the measurement set-up by serial port*

Will not have:

- *Custom ESC*
- *Custom PCB for micro*
- *Closed loop servo control*
- *Closed loop DC motor tilting control*
- *Custom arm such that the tilting axis is in line with the centre of thrust*
- *Nice computer GUI for control*

2 Background

In this section, background information will be provided needed to understand the rest of the report. Readers with a basic experience in (quadrotor) aerodynamics and basic quadcopter electronics can skip sections 2.1 and 2.2, however, it is still recommended to read section 2.3, describing existing tilting rotor designs.

2.1 Propeller/quadrotor aerodynamics

Very important to the goals of this project is the understanding of the aerodynamic forces involved. In particular the expected aerodynamic forces, and the main contributors to secondary aerodynamic forces.

2.1.1 Lift of wings and rotors

Wings and other air-foils produce lift by 'pushing' air into the direction opposite to the direction of the thrust vector. The rotor of a helicopter or quadcopter acts essentially the same as a regular air-foil, instead of cutting through the air with airspeed *v* because of the movement of the craft, it has an angular speed ω due to the rotation of the blades.

The most common way to calculate the lift of a wing, is using the lift equation.[1][2] This generally described as follows:

$$L = \frac{1}{2}\rho v^2 A_{wing} C_L \quad eq.\ 1$$

With L being the lift force in Newton, ρ being the air density in kg/m^3, *v* the wing speed in m/s, A the wing area in m^2 and C_L the lift coefficient of the wing in arbitrary units. For helicopters this equation changes slightly to the modified form:

$$L = \frac{1}{2}\rho \omega^2 A_{rot} C_L \quad eq.\ 2$$

As can be seen, the directional velocity changes into a rotational velocity, and the area changes as well. This change in area, is because the lift force is calculated because of a change in pressure. With a normal wing this is the wing area, but with a rotor this is the entire rotor over which the pressure is formed. Important to see in the equation is that the total amount of lift is dependent on constants, and on the rotor velocity squared.

2.1.2 Dis-symmetry of lift

However do note that the previously mentioned lift equation assumes an equal amount of lift over the entire rotor. In rotor-craft aerodynamics there is however a phenomena called 'dis-symmetry of lift'. This implies that during a rotation of the rotor a larger amount of lift will be observed at the advancing blade half. When viewing the rotor from above, the advancing blade is the blade which is moving against the airspeed, thus having a higher relative airspeed as when compared to the retreating side.
Now considering zero airspeed over the craft, the propellers will have the same airspeed moving in either the retreating or advancing side. This changes however when the vehicle starts moving. The airspeed of the vehicle will cause the relative airspeed of the advancing side to be higher since it has to 'push' into the relative wind of the vehicle. Consequently, the retreating side will have a lower relative airspeed because it is moving in the same direction as the relative wind of the vehicle.

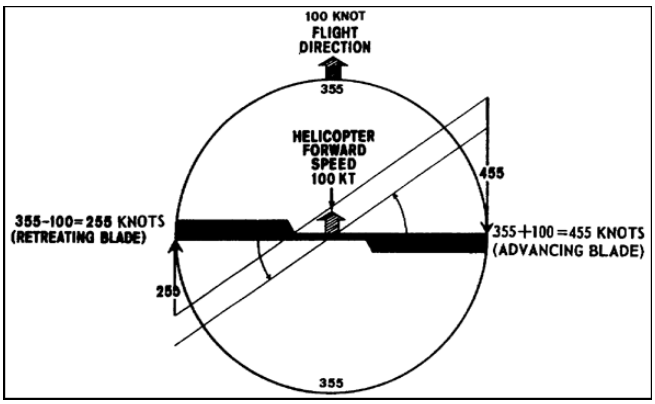


Figure 1: Top view of a rotor showing the difference of air speed for the retreating and advancing rotor blades. Note that if a flight speed of 355 knots is reached, the retreating blade will stall[3]

This generates a torque around the rotor centre because of the dissimilar lift on both sides of this centre. This torque is not desirable for vehicle stability, and if the forward airspeed is large enough, can cause the retreating edge of the blade to go into a condition called ‘stalling’ in which it will produce no more lift. The helicopter will tip over and crash into the ground if no proper handling is applied to deal with this problem.

In normal helicopters the dissymmetry of lift is largely compensated for by a phenomena called *blade flapping* which will be discussed more in depth later. Because of the quadcopter design, there are always two propellers moving in opposite rotational directions, this is done such that there is a net zero torque around the rotor blades and essentially functions as the quadcopter analogy of the tail rotor with normal helicopters. Looking at figure 2, it can be seen that the rotation direction is shown by the arrows, and forward movement direction by the "FRONT" arrow. The advancing side of a rotor blade is indicated with a red half circle, whilst the retreating edge is indicated by a blue half circle. Note that the CW rotating propellers have a 'mirrored' retreating and advancing edge when compared to the CCW rotating propellers.
As can also be seen if an imaginary axis is drawn through the craft in the direction of movement, the retreating and advancing edges will all compensate each other. Because of the symmetry of the aircraft this will be the case in all directions of movement in the plane. Thus, there is no need for blade flapping to stabilize the craft when moving in the horizontal plane.

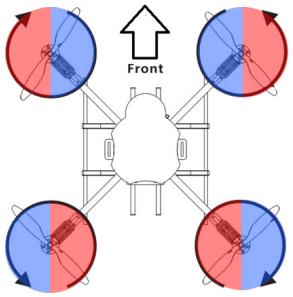


Figure 2: A schematic overview of a quadcopter, colours showing the compensation of the dis-symmetry of lift because of rotor rotation.[4]

2.1.3 Blade flapping

As explained before blade flapping is usually implemented on purpose in conventional helicopter aircraft. However it can also be a secondary, non-designed aerodynamic force due to the flexibility of the prop.

The way blade flapping works, is that it will make the blade tilt slightly up with more airflow due to drag, reducing the effective lift. In the same reasoning, on the retreating side, the relatively lower airflow will make the blade pitch less, increasing lift. This compensates the dis-symmetry of lift. The difference in amount of lift generated by both sides of the rotors causes a torque around the axis in line with the relative wind. This is often referred to as the *rolling moment*.

Because of the compensation due to quadrotor symmetry, blade flapping is an unwanted effect. Quadcopter rotors are made as stiff as possible to resist the blade flapping phenomena, however this cannot be fully resisted, there will always be some flapping in the blades. Effectively, due to the blade flapping there will be a misalignment between the rotor plane and the rotor hub. This will in turn modify the induced airflow during relative air displacement in the horizontal plane.[4]
Also, the load cycles are at the same frequency of the blade rotation, thus there can be a resonance here. This will induce a phase shift of 90 degrees with respect to the load location, which will redirect the thrust vector 'away' from the relative wind.[5]

Seeing how the dis-symmetry of lift is compensated for by quadcopter symmetry, this is obviously an unwanted secondary aerodynamic effect that should be investigated.

2.1.4 Induced drag

Another phenomena to be considered is induced drag[7]. This phenomena makes the effective airflow shift from the free air stream, to a modified one at a slight angle. This change of angle in the induced down wash, results in an induced drag of the blade This is an apparent force 'pushing' the blade against the rotation direction. Essentially, this phenomena is also caused by blade flapping.

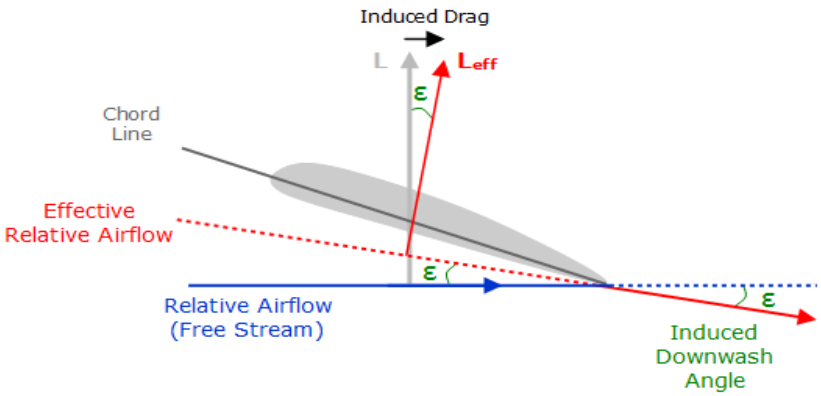


Figure 3: Diagram illustrating the induced drag phenomena[7].

2.1.5 Hub force

The hub force is a drag force that can normally be neglected when hovering or near hovering, but becomes more dominant when moving faster in the lateral direction. It is a force that acts on the rotor shaft (is perpendicular to it), and opposes the lateral velocity component.[11][19]

2.2 ESCs, brushless DC motors and servos

It is important to consider the various amount of achieving thrust for a quadcopter. Almost all commercial quad-rotors are electrically powered, and use BLDC motors for propulsion. They will be shortly discussed in the following section.

2.2.1 Brushless DC motors

Because of the limited amount of space, and tight weight limits the choice for practically all quad copters are brushless DC motors, or BLDCs for short. In contrary to 'regular' brushed DC motors, there is no electrical contact between the moving rotor, and the stationary hub. In a regular DC motor there is a connection using brushes which are often made of carbon. These brushes reverse the polarity of the current, such that the magnetic field switches direction.

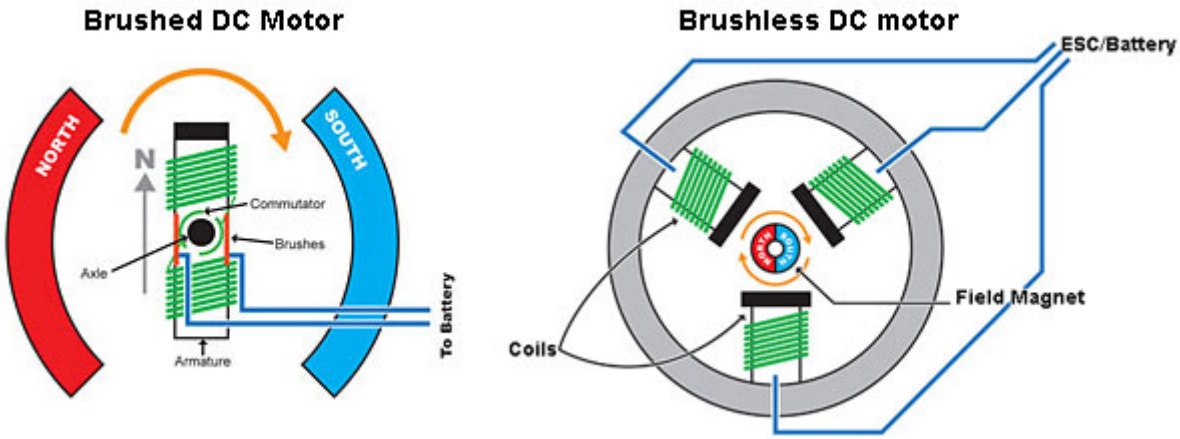


Figure 4: Comparison on internal connections of regular DC motors and BLDC motors.[8]

Due to the absence of these brushes, BLDCs have higher efficiency and lower mechanical wear, at the disadvantage of cost and more complex control electronics.[9] The high efficiency results in a very high power density which is ideal for quad rotors and other electrically powered aerial platforms.

As seen before, rotational velocity is most important when considering lift. The maximum rotational velocity can be calculated using a combination of the motor voltage, and its 'KV rating'. The KV rating indicates how many rounds per minute the motor will turn per volt, without a load. When attaching a propeller this number will reduce.

2.2.2 Brushless DC motor control

As mentioned previously, BLDCs require more complex driver electronics. This is because the 'commutation' of the poles is not done by brushes, it is done electronically. This requires sensors, and some form of processing to send pulses to the driver electronics.

For the purpose of the report, it is not of interest to dive into the details of BLDC control, instead this task is handled by an "Electronic speed controller", or ESC. These are widely available from the civilian drone market and can handle most brushless motors. Generally they use a battery voltage input, a standard 'servo' input for control, and have three output terminals to connect to the BLDC. All the complex electronic computation and motor control loops are handled internally.

2.2.3 Servos

In general, a servo motor is a rotary motor designed for accurate angular (and sometimes linear) position control. It incorporates a motor, some form of position sensing and a feedback control circuit. There are of-the-shelve integrated servos available in various different sizes. Servos are widely used in the remote controlled aircraft world to actuate flight surfaces such as the rudder. These integrated servos use a small DC motor, a reduction gear, a potentiometer and a feedback circuit to provide accurate position control.

Figure 5: Cutaway view of a servo, note the gear reduction, control circuitry and DC motor.[10]

2.3 Tilting rotor designs

Important to consider is how the tilting mechanism will be connecting the brushless motor to the actual flying platform. The choice of actuator positioning will determine ease of construction, response time of the tilting mechanism, maximum deliverable thrust and the maximum angle that can be reached by the tilting system.

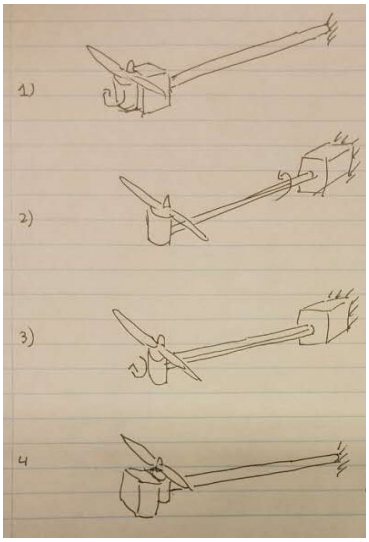


Figure 6: The four types of actuator placement

There are four different ways that the actuator can be attached between the brushless motor and the flying frame. These are illustrated above in figure 6. There are two main ‘groups’, type 1 and 4 are with the actuator close to the propulsion motor, and the other two have the actuator close to the main body.

Type 1:
Propulsion motor is directly attached to the servo body, which is also attached to the arm. The actuator is ‘sandwiched’ between the two.

Type 2:
The propulsion motor is fixed to the arm, and the entire arm can be rotated to achieve a rotation with respect to the flight frame.

Type 3:
Similar to type 2, except that now there is an internal axle to transfer the rotation to the motor, the outside tube is fixed to the motor with a free rotating joint.

Type 4:
Similar to type 1, except that the actuator is positioned on the outside of the flying frame.

Considering existing literature, especially type 2 and 4 are prevalent.

In the paper by *M. Ryll et al., 2014*[11], these two types are both present. The first iteration discussed shows a type 4 set-up using a standard RC servo. This design is validated and used for the measurements.

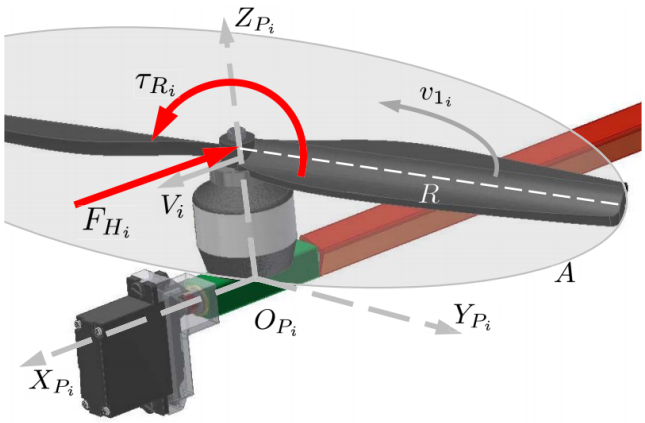


Figure 7: First iteration tilting design proposed by M. Ryll et al.[11]

After doing measurements and making a practical over actuated quadrotor with this design, a second design is proposed. It is dubbed the *Holocopter Version 2*, and is constructed around a type 2 mechanism, with custom bent arms such that the centre of thrust is in line with the centre of rotation. It uses custom DC motors with position feedback to steer these. The actuators are positioned near the centre of mass of the robot, providing a lower inertia during movement of the aerial robot.

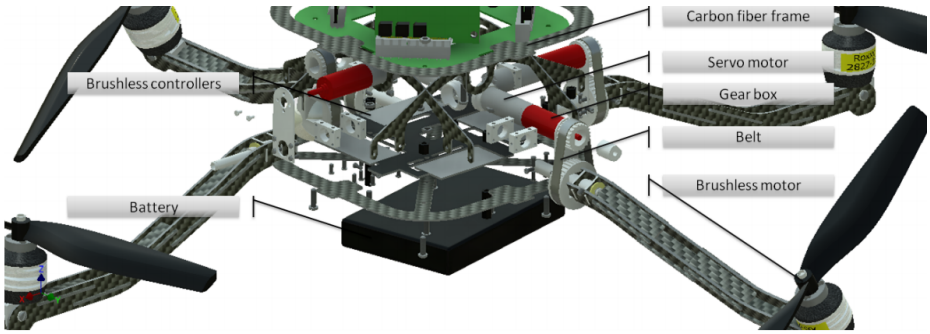


Figure 8: Holocopter version 2, showing a clear type 2 construction[11]

A design shown by *A. Oosedo et al., 2015*[12], shows a type 2 design as well. However, this design does not make use of bent arms to put the centre of thrust in line with the axis of arm rotation. It does use standard RC servos for its design. Note that the attachment of ‘rotor rods’ is fairly large and heavy.

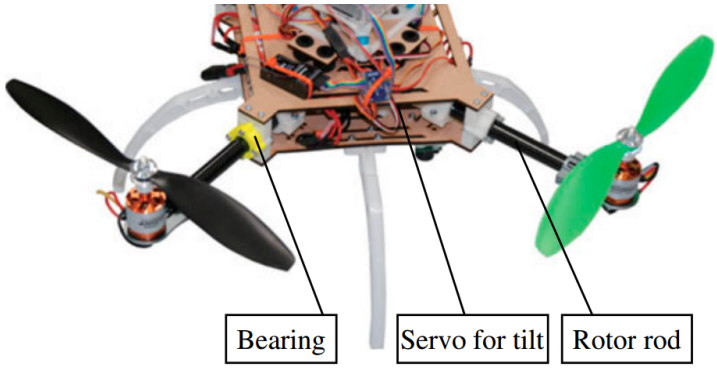


Figure 9: A clear type 2 construction shown in “Large attitude change flight of a quad tilt rotor unmanned aerial vehicle”[12]

The company *Skybornetech technologies*[13] is developing a three motor UAV design, with two titling rotors. Their design is of type 2 as well, also not using custom bent arms. It is however, using standard commercially available servos.

Figure 10: The Skyborntech rotating arm mounting system[13]

Another paper proposes a type 2 design as well.[14] Using commercial servos and a fairly bulky rotation mechanism. The design by *A. Nemati et al., 2014* does however use standard of the shelve available parts and is relatively easy to produce.

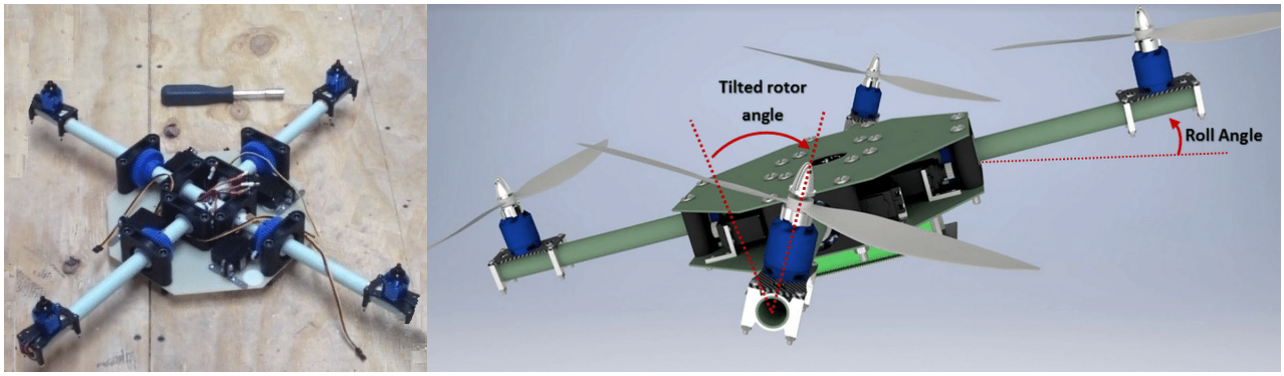


Figure 11: Another type 2 design showing relative ease of construction[14]

3 Core chapters

3.1 Measurement system overview

For the measurement system it is required to be flexible and easy to put together and use. In figure 12 a proposed overview can be seen. This system overview clearly shows all the subsystems required, and how they interconnect with each other. Note that the servo and servo interactions are to be left out for the non-tilting measurements. For the system to be easy to build, easy to use and robust, special attention has to be paid to the interfaces. How these are constructed and fit together is imperative to the working of the system and success of the research.

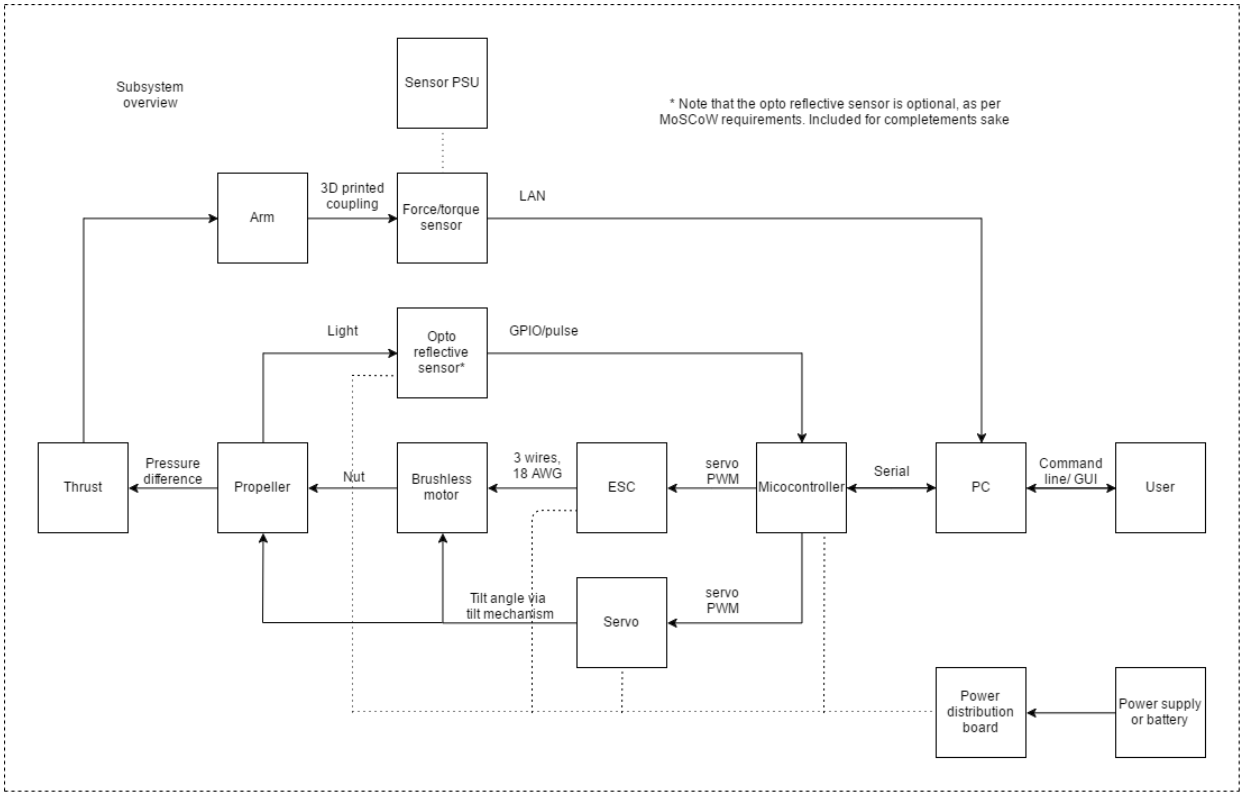


Figure 12: Schematic system overview showing all the subsystems

To further explore the system and find out how and what to build, a modified N2 diagram is constructed to identify the interfaces between the subsystems. This is useful in identifying the modules to build, constructing the wiring harness, recognizing components that are needed and how everything fits together in the bigger picture. Of the interfaces, predetermined interfaces are shown in red, so it is known what is free to design, and what is not. Also the flow of signals, power and mechanical connections can easily be identified. It can be seen that most of the design work is in the interfaces between subsystems.

Figure 13: N2 diagram showing the interconnections between the subsystems

3.2 Subsystem analysis and design

Of all the subsystems their purpose will be described, as well as possible choices made.

PC

The PC coordinates the measurements. Ideally it would use an RS232 interface connected to the microcontroller to send commands about servo and engine throttle controls. The minimum requirement is that it should record sensor data provided by the force/torque sensor and log this data into a file readable by matlab. Further processing of the data is also done on this PC, although not real time using the software package *Mathworks - Matlab*[20].

Microcontroller

The microcontroller is the brains of the measurement setup. It will control the BLDC throttle via the ESC, and also do the servo rotation for the titling measurement setup. Any possible other processing will also be handled by the microcontroller. The microcontroller itself does not need to be excessively powerful because it isn't doing any significant number crushing or real time control, thus ease of use is of primary concern. It should be able to be programmed fast and easily. If time is left over, the microcontroller can send positional data back to the PC, and receive commands from the PC over an RS232 link.

A comparison is made between common microcontroller boards available on the market. As discussed above, the most important criteria is that it should be easy to use. Power is not of main concern, thus price is a more important key driver.

Microcontroller	*Arduino*	*AVR*	*Frdm k64f*	*STM discovery*	*Intel Edison*
Price	--	++	+	++	---
Power	--	-	+++	++	++++
Features	-	-	++	+	+++
Ease of use	+++	+	+++	--	+
Total	*-2*	*1*	*9*	*3*	*5*

Table 1: Comparison table of common microcontroller development platforms/boards

Thus from this table, the NXP (Formerly Freescale) FRDM K64F development board is used for the measurement setup. Specifications of the board are shown in table 2.

Specification	
Part	*NXP (formerly Freescale) FRDM K64F*
Operating speed [MHz]	*120*
No. of Bits	*32*
Memory	*256KB SRAM, 1MB flash*
Operating current [mA]	*<150*
Operating voltage [V]	*5*
Interfaces	*Ethernet, PWM, SPI, I2C, I2S, SDHC, USB*

Table 2: Specifications of the FRDM K64F development board

Important is to note, that an external power supply can easily be supplied to the headers of the board. It can be programmed using an online development environment called *mbed*[21], enabling cross platform portability. There is no need for complicated set up procedures, and it enables the use of a hardware abstraction layer such that there is no need to program registers. It has a build in programmer with USB bootloader, thus a binary file is downloaded from the web IDE, and can directly be uploaded onto the board. On board LEDs and push buttons are useful for debugging.

Servo

The servo is the part that will make the BLDC turn to a specified angle in the tilting measurement setup. A servo was already provided for the project. The following specifications are known of the servo:

Specifications	
Part	*Hitec HS-5085MG Metal gear digital servo*
Operating voltage	*5 [V]*
Speed	*0.17 ~ 0.13 [sec per 60 degrees]*
Maximum torque	*0.353 ~ 0.422 [Nm]*
Maximum current (at stall)	*2.15 [A]*
Weight	*21.9 [gr]*
Size	*29.0 x 13.0 x 30.0 [mm]*

Table 3: Specifications of the Hitec HS-5085MG servo provided for the project

Important are the current draw and operating voltage for the power supply specifications required, and also the maximum speed and torque available for the design of the tilting mechanism. The size and weight are important for future research when the tilting mechanism may be integrated into a flying platform.

BLDC

A BLDC has already been selected and provided for the project. Specifications of this part are important for ESC selection, and weight and size dimensions are important for the final design of an aerial robot. These are shown in table 4.

Specifications	
Part	*T-Motor "Germany Aerolab" MT2212*
KV-rating	*750*
Voltage [V]	*11.1 – 14.8*
Maximum current [A]	*16*
Maximum power [W]	*200*
Optimum efficiency	*>83%*
Weight [gr]	*55*
Dimensions (diameter vs height) [mm]	*27.5 x 28.5*

Table 4: Specifications of the BLDC motor provided MT2212-750

Propeller

The choice of propeller (or prop for short) is important, since it not only affects the total amount of thrust the system can deliver, but also response, and secondary aerodynamic effects. Propellers have been selected already as well, the T-motor 11*3.7 carbon fibre propellers are used. Because the propellers are constructed using carbon fibre they are more stiff than comparable size propellers made of other types of materials such as plastic or wood.

Power supply and battery

A battery has already been provided to the project. The *Tattu 1300mAh 14.8V (4S) 45C* battery is used. In table 5 a list of specifications can be found. This selected battery imposes constraints on other parts of the system such as ESC and distribution board selection. Other specifications such as weight and size are not as important for the measurement system, but are for the final design.

Specifications	
Part	*Tattu 1300mah 14.8V 45C*
Capacity [mAh]	*1300*
Voltage [V]	*14.8 (4S)*
Discharge rate[C]	*45*
Weight [gr]	*146*
Size [mm]	*73*32*28*

Table 5: The selected battery specifications

For most of the measurements however, a rack power supply will be used. This prevents the need to charge batteries and can provide a more stable voltage independent of both load and discharge time. The battery will still be taken into account such that the other electronics are compatible with the battery selected.

The power supply available in the flight lab is the *Delta power supplies SM52-AR-60*. Capable of providing 1500 watts of continuous power. A cable is already available with the same *XT60* connector to attach to the wire harness that will be constructed.

ESC

For the measurement setup, 3 different ESCs were provided which could be used. The different types are shown in figure 6. After considering what is most important the ESC with the best fit will be used for the project.

ESC name	*Price (€)*	*Size (lxwxh) [cm]*	*Weight [gr]*	*Voltage range [V]*	*Maximum current [A]*	*Direction reversal*
Maytech 30A opto	12.35	4.50x1.75x1.00	26	7.2 - 22.2	30	Y
T-motor air 20	19.99	5.24x2.15x0.70	14	11.1 - 14.8	20	N
DYS SN 20A	10.05	2.30x1.20x0.45	7.6	7.2 - 14.8	20	Y

Table 6: Comparison of the different evaluated ESCs.

As can be seen all ESCs have enough current to drive the selected MT2212-750 motor. The main difference between them is price and size. However, one also needs to take into account that the direction of the motor can be reversed. A large voltage range is also a good option to increase flexibility.

ESC name	*Maytech 30A opto*	*T-motor air 20*	*DYS SN 20A*
Price	+	-	++
Size	+	-	++
Weight	-	+	++
Voltage range	++	-	+
Current	++	-	-
Direction reversal	++	--	++
Ease of use	+++	+++	+
Total	*10*	*-2*	*9*

Table 7: Comparison table for ESCs provided to the project

Table 7 clearly shows that the Maytech and DYS both seem best suited for the application. However, the larger current headroom and ease of reprogramming resulted in the Maytech being chosen for the measurement setup. The DYS is recommended to use with the final design due to the smaller physical dimensions and lighter weight.

Arm

Since there is already an intended airframe, the tilting system has to be adapted to suit said system. The particular airframe used is the *Tarot Iron man FY650.* It is a carbon fibre frame, with carbon fibre tubes to which the brushless motors attach. The attachment from the arm to the airframe is already in place, and the attachment of the brushless motor mounts to the arm is removable, making it a relatively simple procedure to design a custom tilting mount. The effective arm length is about 30 centimetres.

Figure 14: The Tarot Iron man FY650 airframe kit[15]

Force/Torque sensor and associated power supply

The force torque sensor *ATI Industrial Automation F/T Mini40E*[16] available at the Robotics and Mechatronics lab is available to use with the project. Care should be taken not to overstress the sensor maximum specifications (see table 8), since this could lead to permanent damage of the sensor. Next to damaging the sensor this way, it is also possible to insert screws within the mounting holes too far, which would also damage the sensor.

Specifications	
Part	*ATI Industrial Automation Mini40E*
Maximum allowed axis force (x, y, z) [N]	*±810, ±810, ±2400*
Maximum allowed axis torque (x, y, z) [Nm]	*±19, ±19, ±20*
Maximum screw depth from surface [mm]	*5.0*
Dimensions (diameter x height) [mm]	*40 x 12.2*
Weight [gr]	*49.9*
Sample rate [Hz]	*50*

Table 8: Specifications of the ATI Industrial Automation F/T Mini40E sensor

RPM sensor

There are various common ways on how RPM can be measured. The most popular ways to measure RPM without a physical connection to the rotating body are shown in table 9. Because of the time restrictions imposed on the project, and limited budget available special attention needs to be paid to the key-drivers price and ease of implementation.

Type of measurement sensor	*Hall effect*	*Inductive sensing*	*Opto-reflective*	*Otto-interrupter*	*Wire tap*	*Laser reflective*	*High speed imaging*
Price	++	+	++	++	+++	–	---
Ease of implementation	++	-	++++	++++	+	---	---
Accuracy	+	-	+++	+++	-	++++	++++
Robustness against dirt	+	++	-	-	+++	-	+++
Robustness against environment changes	++	++	+	++	+	++	-
Total	*8*	*3*	*9*	*10*	*7*	*1*	*0*

Table 9: Comparison of different types of sensing

From the total score three options are left. Especially ease of implementation is very important due to the limited amount of time available during the execution phase of the project. Thus if we look at making a wire tap, it would require special signal conditioning circuitry to get a reliable signal from the ESC wires. Next to this, the wire tap is not as accurate since it does not measure the speed directly from the rotating body. This selects either the opto reflective or the opto interrupter implementation. The opto interrupter would work by having a light gate near the blades, through which the blade would pass. This will have implementation issues due to blade flex, and would also be in the way of the thrust. A opto reflective sensor would only need to be positioned close to the BLDC motor. Thus it is chosen to implement the RPM sensor using an opto-reflective type sensing.

Power distribution board

As seen in other subsystems, it is required to have a 5V line (microcontroller, servo) and a high power battery line for the ESCs. Thus it is needed to split the battery input voltage for the (eventually) 4 ESCs, and to provide onboard 5V power. This is traditionally handled by a so called “Power distribution board” or PDB, sometimes called power hub. For the project, two different PDBs were available to use. Their specifications are listed below in table 10.

Specification	*Mini Power hub W/ BEC 5V & 12V V3*	*LED & Power hub 5in1 V3*
Input voltage	*7.2 – 26V (2-6S Li Po)*	*9 – 26V (3-6S LiPo)*
Output current (cont)	*20A (peak 25A)*	*20A (peak 25A)*
Regulated output voltages	*5V, 12V*	*5V, 12V*
Regulated output currents	*3A, 2A (3A peak)*	*3A, 0.5A (@16V in)*
Size	*36x36x4mm*	*46x36x6mm*
Weight	*6gr*	*9gr*
Price	*5$*	*7.5$*
Note	*For the 12V BEC, at least 4C is needed*	*12V BEC is linear reg* *5V LED lights may be abused to use with servos*
Special features	*Has camera V bridge*	*Has LED light controller* *Has lost plane finder* *Has low voltage alarm*

Table 10: Comparison of power hub specifications

Main requirements of the power distribution board are the motor output voltage and 5V rail for the servo and microcontroller. As can be seen the specifications for both boards in these areas are similar. Another important parameter is input voltage. As discussed before, the system is intended to run of a 4S battery. A larger input voltage range would give good flexibility though. The mini power hub is smaller, more compact and lighter, however, the Power hub has LED facilities on board and a lost plane finder. Most of the extra features both hubs offer are geared towards outdoor flight, whilst the prototype will be used mostly indoors. It should be noted that the LED lighting voltage may be abused as a servo line when necessary. The camera bridge will also probably not be used. The high current capability of the 12V BEC line of the mini power hub could be useful when connecting other control boards externally, that use 12V instead of 5V.
Thus the Mini Power hub seems to be the best fit for the measurement set up system. The LED & Power hub 5in1 V3 offers no significant advantages over the Mini power hub that are relevant to the measurements, but may be a good option if the prototype is intended to fly with marking LEDs.

3.3 Stationary rotor system and analysis

The first measurement setup that will be discussed is the non-tilting measurement setup. From this measurement setup a ‘baseline’ of secondary dynamic forces is made which can later be compared with the tilting measurement setup forces. A conclusion can be drawn from this whether the forces are negligible or not in the tilting rotor case. From the stationary rotor setup, also most modelling parameters for the BLDC motor + propeller combination can be retrieved.

3.3.1 Analysis

Definition of axis

For the measurements, it is important to define the reference frame. It is shown in figure 15. As can be seen in this figure, a torque is defined as being positive for a clockwise torque looking in the positive direction of the axis. These axis are found from the sensor data-sheet[16], and thus also used here to avoid confusion. From this point onwards, the previously discussed axis orientation will be used in discussing all forces and torques.

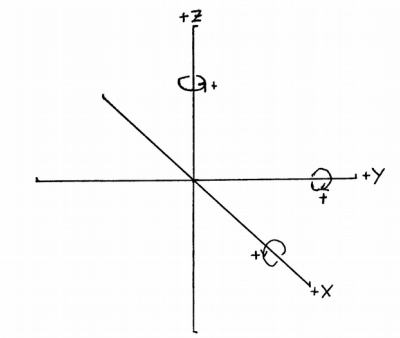


Figure 15: Axis definition for the measurements

For the stationary rotor analysis all the forces and torques are measured at the end of the arm. This will give some implications in the measurements taken. Namely that all forces in the motor, will be translated to forces in the sensor, but also into torques with the length of the arm. For example, take an arm with the sensor 0.5 [m] away from the brushless motor. Aligning the reference frame as shown in figure 16, if there would be an upwards pointing thrust vector of 10 [N] in the Y axis, it would be measured as both a force in the X-axis and a torque around the Y-axis of 10*0.5=5 [Nm].

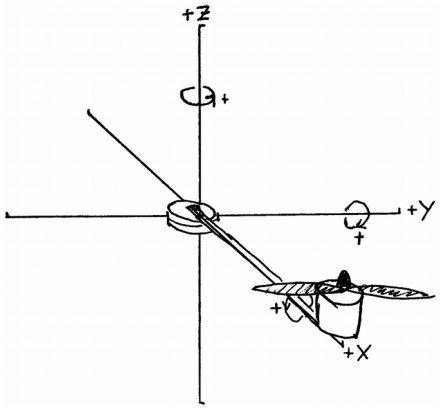


Figure 16: Axis with respect to motor and sensor positioning

Expected forces and torques

The main component that will be measured is the thrust the motor will deliver. In the fixed rotor case, from the viewpoint of the BLDC this will ideally be a pure force in the Z axis. As discussed before, because of the arm, it will also be seen as a torque around the Y axis.

Considering the data provided by the manufacturer[17], an estimated coefficient of lift can be determined. The same motor and propeller combination is used (MT2212-KV750 with T-MOTOR 11*3.7CF propellers). Do note that the thrust values are highly dependent on the voltage supplied to the motor.

Throttle (%)	*Thrust, Fz [N]*	*Torque, Tz (25cm) [Nm]*
50	*4.22*	*1.06*
65	*5.79*	*1.45*
75	*7.06*	*1.77*
85	*8.34*	*2.09*
100	*9.22*	*2.31*

Table 11: Manufacturer data[17] on thrust vs throttle, measured at a motor voltage of 14.8[V]

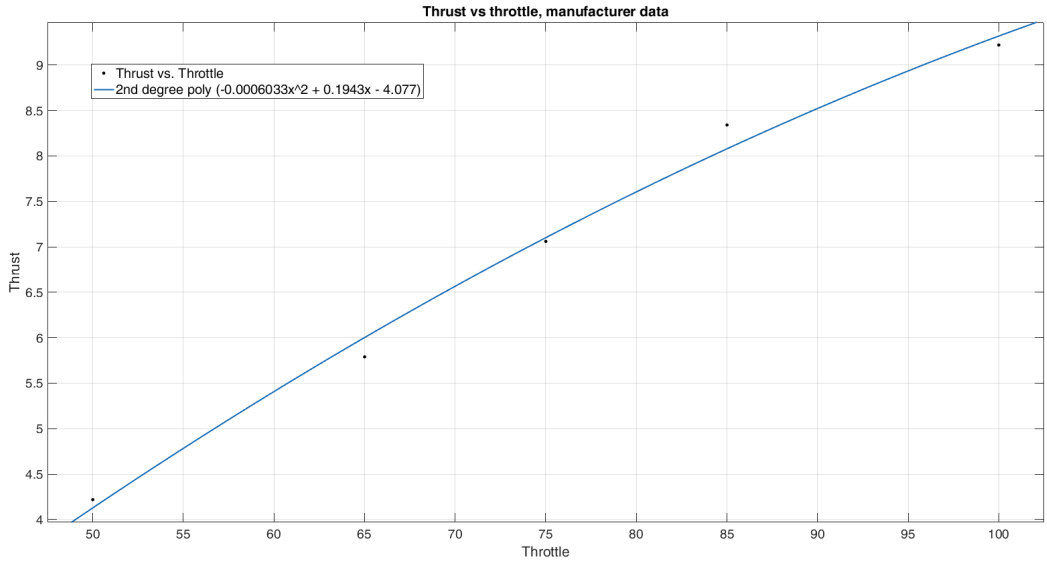


Figure 17: Manufacturer data on thrust vs throttle, with second degree poly fit

A second degree polynomial fit is made to this provided data, such that it can later be compared to the results found in this research.

Normally the thrust vector will be pointing only in the Z-axis, however, secondary aerodynamic effects will cause the thrust vector to veer slightly off centre. This will primarily be due to blade flapping and hub force. A difference in relative airflow on the retreating and advancing edge of the quadrotor will cause a different induced drag on both sides of the rotor, thus changing the direction of the thrust vector. These will appear as forces in the X and Y axis. The X component will generate a torque in the Y axis with as the arm the Z distance between the motor and the sensor. The force in the Y axis will generate a torque in the Z axis. Both the hub force and the blade-flapping will cause the same forces and torques to occur, thus they will be considered as a single cause of secondary aerodynamic effects since they cannot be distinguished from the measurements. Also, distinguishing them is not of interest in the goals of the research, only its significance.

A misalignment between the sensor top plane and BLDC top plane can also cause the thrust vector to have force components in the X and Y axis. The same goes for flex in the quadcopter arm and other mechanical mismatched and non-idealities.

With a relative wind flowing over the rotor, the dissymmetry of lift will cause a rolling moment around the X axis of the rotor hub. This will be translated as a torque in the X axis of the sensor as well.

The drag torque caused by the motor will be a torque in the Z direction, which thus would also be visible as a torque in the Z direction for the sensor.[18]

The following assumptions were made during the experiment in terms of mechanical forces and causes of the mechanical forces:
- Infinite rigidity in the quadcopter arm
- Rotational speed of the rotor blades is linear with PWM percentage
- All joints and 3D prints are infinitely rigid
- Alignment errors between sensor xy plane and brushless motor xy plane are not significant
- Power supply voltage does not vary over load
- The Z distance between the motor and sensor is small, and thus the torque coming from the force in the X axis from the motor will be negligible (under the noise floor)

Thus the phenomena of interest are found in the following forces and torques:
- Force in X axis
Blade flapping, Hub force
- Force in Y axis
Blade flapping, Hub force
-Force in Z axis
Thrust

- Torque in the X axis
Rolling moment (Dissymmetry of lift)
- Torque in the Y axis
Thrust (mapped from Fz)
- Torque in the Z axis
Drag torque
Blade flapping, Hub force (mapped from Fy)

Expected primary and secondary aerodynamic forces

Most of the forces discussed have already been abundantly discussed in literature, and on the basis of this, the following expectations are constructed.
For the blade flapping, hub force and trust, a dependability of blade rotational velocity squared is expected[11]. Thus graphs will be tried to fit with a second degree polynomial fit.

If we map the Fx and Fy vectors as a result of the blade flapping and hub force against Fz, the expected dependency is in the range of 10%.[11]

When mapping Fz to Ty, it should match up perfectly, and this can thus be used to verify that there are no secondary aerodynamic effects that affect the torque in the Y axis. If there are they will show up as a difference between expected and measured torque.

In previous research by S. Park et al.[23], it was found experimentally that the drag force is about 1/50th of the thrust force. It can be found by subtracting the mapped torque in the Z axis from the force in the Y axis.

As for the rolling moment, without any relative wind flow this is expected to be zero. There could still be some torque present due to air bouncing of walls and other non-idealities.

As for the step response of the motor, the system itself consist mostly of a rotating inertia and friction. The elastic effects are considered to not be of large effect. This would in turn make one think that the expected response is an (almost) first order response. However, this would neglect the dynamics added by the ESC. Considering that the ESC does speed control, it probably has a controller build into it. The control system technique most often used for ESCs is a so called PID controller. This inherently has a second order response, and thus the result of a step is expected to be second order as well.

Measurement profile

Important in getting the correct data, is the measurement profile that will be used. Looking at the requirement, there are two types of data that are of interest. The first is steady state data at different throttle positions. These can give an indication of secondary aerodynamic effects, and can be used to extract modelling parameters such as lift and drag coefficient. A second set of measurement data is needed to extract the system dynamics. This can give an idea of how the system would respond to a change in throttle.

For the steady state ‘slow’ measurements, a slowly increasing ramp is chosen that goes from 0% to 100% throttle. The duration of this ramp is chosen to be long (in the order of 100 seconds) such that the effect of system dynamics is negligible. It will however, create a smooth curve of throttle versus torques and forces, because time is linear with throttle percentage. This removes the need to feed back absolute throttle positions to the computer, which then need to be mapped to the data.

For the measurement profile the assumption is made that it behaves as a linear, time invariant system. To find the dynamics of the system, the first most obvious choice would be a delta-Dirac pulse. This is because of the unique property of the delta-Dirac pulse that a linear time invariant system responds with its own impulse response. However, a pulse of infinitesimally small width and a surface area of one is hard to generate. Instead the anti-derivative of the delta-Dirac function is used, a unit step input. By stepping the throttle input, the dynamics of the system will show up in this response. Standard system identification methods can then be used to approximate the s-transfer function of the system.

Ideally the same step response should also be measured whilst inverting the rotor direction. This response could very well be different to a response to a step in the same direction. With the ESCs chosen rotor direction can be done, however in practice it did not seem to work well and thus was not used for the measurement profile.

Positioning of the measurement setup

To avoid secondary aerodynamic effects caused by phenomena that are not under investigation, the measurement setup needs to be as much into ‘free air’ as possible. Some of the aerodynamic effects are the ground effect (increased lift and decreased aerodynamic drag close to a fixed surface such as the floor) and influences of walls on the air currents. Here another assumption arises, namely that the walls and floor are sufficiently far away to have a negligible impact on the measurements.

Parts to be designed for the measurement setup.

As can be seen in the N2 diagram in figure 13, a few mechanical links still have to be made. For the non tilting measurements it was chosen to use the BLDC attachment plates provided with the motor to attach the motors to the arm. Thus what still needs to be designed is:

- Wiring harness
- Arm-to-sensor attachment
- sensor-to-environment attachment
- RPM sensor and associated electronics
- Software for controlling the ESCs

3.3.2 Design and realization

For the design phase, the subsystem interconnections will be investigated that will be used in the non-tilting measurement setup.

Wiring harness and electronics organization

In the N2 diagram it is clear that there are a lot of wiring interconnections visible between subsystems. These all need to be put together into a wiring harness. This will prevent entanglement of wires, and also make it easier to rebuild the setup, or change parts if needed. It prevents confusion when things do not work as expected and make it easier to find errors in wiring.

After putting all the electronic subsystems on a sheet, connections between subsystems are drawn and organized. Care is taken to make the system easy to take down and build up, and to prevent wire entanglement. The complete wiring harness can be found in appendix *A*. Signal wires are routed with signal wires and power signals with power signals. In the wiring diagram connectors are shown with a yellow rectangle. At these points subsystems can be detached from each other for easy replacement in case of part failure, and ease of construction. For example, if the arm needs to be detached currently just 2 electrical connections need to be unplugged. Do note that in the wiring harness the connections for the RPM sensor are still present.

The wiring harness is constructed using heavy gauge wire for the power connections, and smaller flexible wire for the lower power connections. Regular 2.54mm headers are used for connections of lower power electronics. Signal wires plug into the freedom board with regular male pin connections. For the heavy power connections (PSU to ESC and ESC to BLDC) two types of connections are used. Between the PSU and ESC, an XT60 connector is used, because it is a standard battery connector in quadcopter systems. Between the ESC and BLDC, standard ‘bullet’ connectors are used, covered with heat-shrink to prevent shorting to other parts of the system.

As for the organization of the electronics, during the first set of measurements, it was found that a solid mounting plate would make the setup more reliable. During the first set the microcontroller and power distribution board were dangling on their own wires, which of course increases the chance of a short circuit. This was remedied by fashioning a small MDF plate with stand-offs to mount the PCBs.

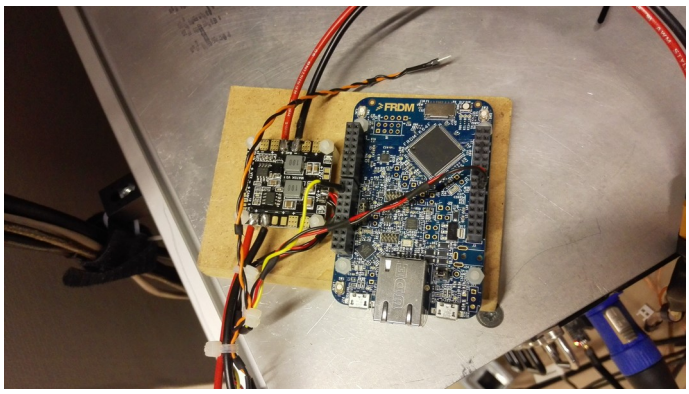


Figure 19: Close up of the MCU and power distribution electronics

Software

The measurement software for the non-tilting measurements follows a predetermined curve programmed into the MCU. It runs this code once after which it waits until reset. The curve that is run, is as discussed in the analysis a ramp, followed by a pulse. This should be sufficient to gather the data required.

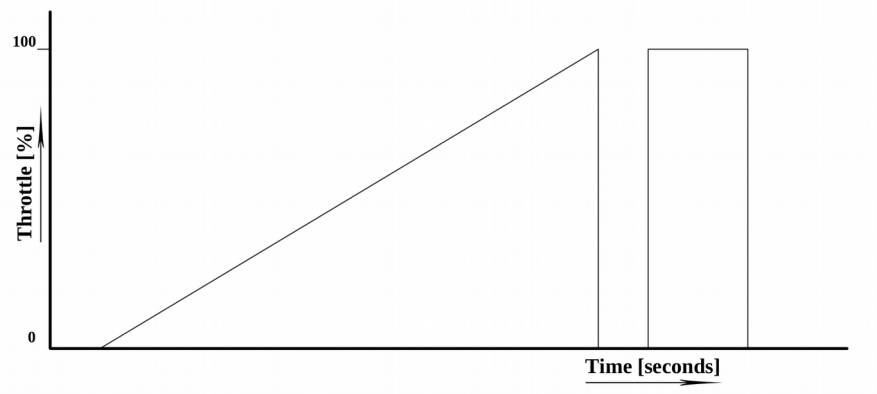


Figure 20: The non-tilting measurement profile

As a safety precaution, the blue LED is always off until the measurements are over. Thus it can easily be seen if there is no chance of the motor spinning up.

Sensor to solid surface attachment

As discussed before, it is important to have the measurement setup in a place with as little interference from unintended forces as possible. It was chosen to attach the measurement setup to the ridable card available in the flight lab. This is a convenient mounting location because it provides a reasonably large distance to the floor (comparable as to the distance the prototype would fly), would make it easy to move the measurement setup to the middle of the room and away from the walls, and would also make for a convenient table to put the electronics and other hardware needed for the mounting and construction. Additionally, the power-supply is mounted inside the cart.

The cart itself is constructed using *Boikon* aluminium construction profiles. These use standard mounting screws and connections can easily be made to the existing profile. An additional horizontal aluminium profile was attached to the side of the cart, sticking out of the front. To this profile an adapter plate made of perspex was then attached. This perspex plate also has mounting holes for the sensor such that the sensor is lined out properly.

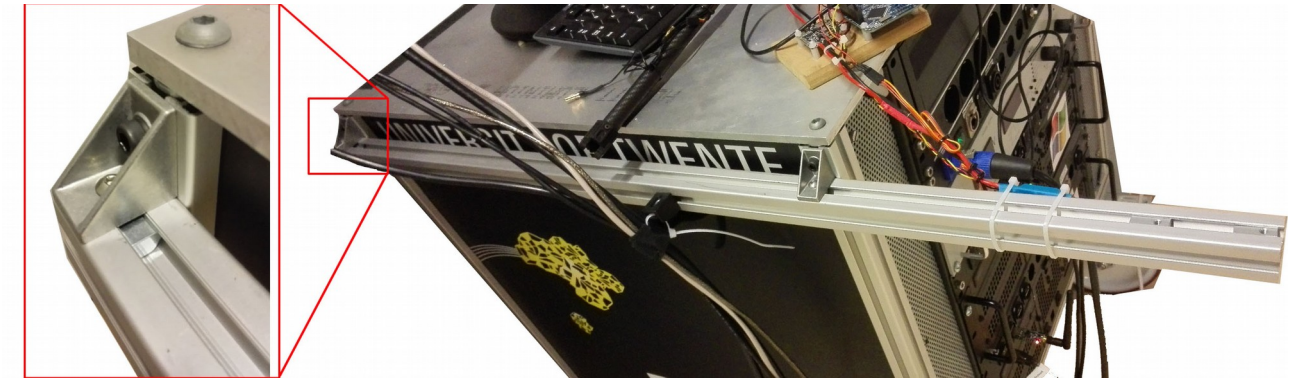


Figure 21: The Boikon mounting profiles, and construction technique, left the 90 degree interconnects used to attach the beam to the cart

F/T sensor mounting

To attain the theoretical And adapter plate is constructed to mount the sensor to the arm of the quadcopter. Requirements are mostly that it needs to be solid in construction, and easy to screw onto the sensor/arm. This is because the F/T sensor is used by multiple projects within the RAM group, thus the sensor is shared and can only be used for a certain amount of time. Quick and easy mounting and dismounting would save time in the long run.

The F/T sensor to arm mounting system went through two iterations. The first iteration (shown in figure 22) has two solid ‘loops’ through which the arm is pushed. The third loop can be tightened using a bolt and nut to prevent the arm from rotating due increased friction.

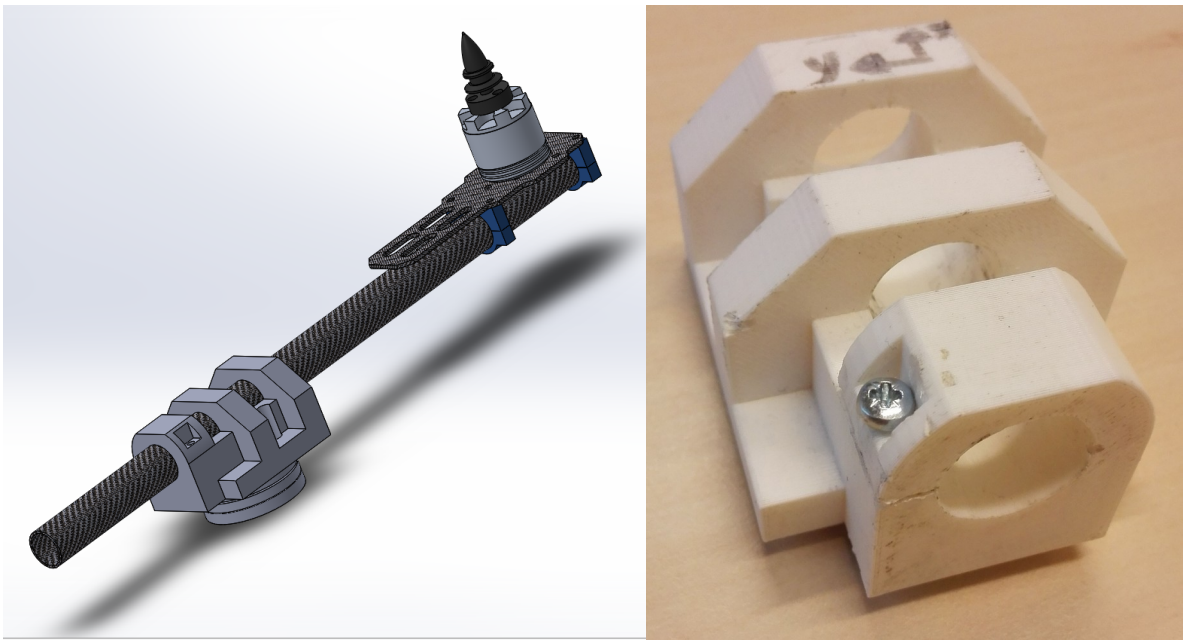


Figure 22: First iteration of the solid mounting system, left the solidworks model, battered 3d print to the right

First measurements have been done using this setup, however, it was not deemed to be sufficiently satisfying requirements due to a few points, and thus a second iteration was developed.

First of all, the mounting plate had the wrong sensor orientation, thus making it have a non ideal angle from the measuring cart to which the sensor was attached (clear in figure 24). The arm did have the correct orientation with respect to the sensor. This was not the only problem with the system. It was designed with the idea in mind to easily slide out the arm if it needed to be detached. In practice however, the 3D fit was harder than expected, and the back of the carbon fibre tube had a solid piece attached to it (normally connects the tube to the quadcopter) thus the entire BLDC motor had to be detached to slide out the tube. Because one of the sensor mounting screws is hidden under the tube if it is in place in the mount, the entire measurement setup needed to be broken down to detach the sensor.

These two problems were addressed in the second iteration of sensor to arm mounting. This system consists of two identical halves. The first half stays connected to the sensor, and the second half is flipped 180 degrees and mounted on top of the other one using four bolts. In between the two halves the arm is sandwiched.

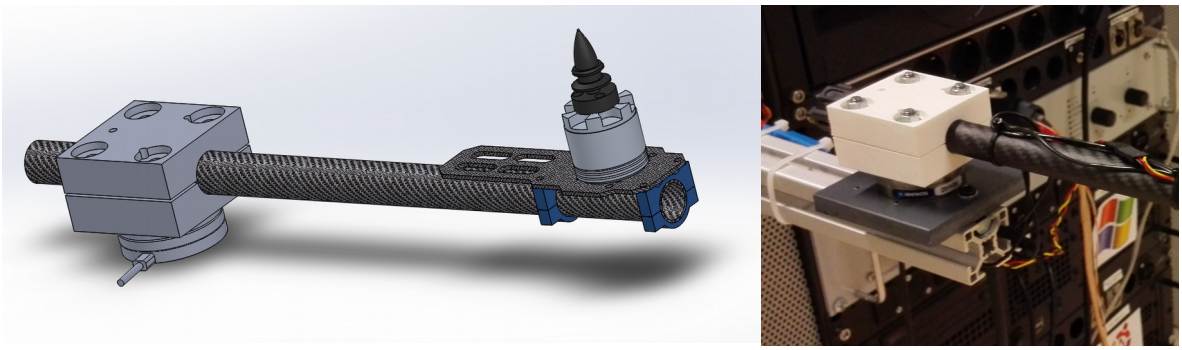


Figure 23: Second iteration arm to F/T sensor attachment.

In practice this significantly simplified mounting and dismounting of the sensor. Just the four bolts holding the halves together need to be undone to reach the sensor mounting screws. Furthermore, the direction the arm is facing now is as intended. This iteration was considered good enough to be used with the tilting measurement setup as well.

RPM sensor

The opto-reflective RPM sensor was determined to be used for the measurements. However, it still remains a should have and did not have priority. Preliminary design was done for the sensor, but the idea had to be dropped due to time constraints. The results achieved so far will still be discussed since they are useful for future work.

The RPM sensor works by sending out (infrared) light, and measuring how much light is returned from the object close by. The first idea was to mount the sensor upwards, to detect whether there was a rotor blade in front of the sensor or not. However, the curvature of the blade made this hard to accomplish. Instead, it was chosen to face the sensor close to the brushless motor, and attach a (light) coloured piece of tape to the BLDC motor. Because different colours reflect different amounts of light, it can be detected when the piece of tape (of known length) passes the sensor. The output of the sensor is analog, so it is fed into a comparator circuit to digitalize the signal.

Figure 18: Oscilloscope screenshot of the sensor about 6mm away from the BLDC motor.

Figure 18 shows the practical attained results measuring with an oscilloscope. The BLDC was run full speed to see if the sensor is fast enough to register the edges. As can be seen in figure 18, a simple threshold comparator around the midway point of the signal would digitize it sufficiently. The 171.3Hz frequency of the signal is the equivalent of about 10300 RPM, which is around the maximum speed of the brushless motor without load.

What was not tested, is whether the sensor is sensitive to ambient light. Sunlight and certain types of artificial light contain large amounts of (near) infrared energy, which could give false readings or prevent the sensor from working. This is a point for further research and development of the RPM sensor. Also note that the distance had a large impact on the voltage range of the signal. Placing the sensor as close as possible to the BLDC would significantly increase the voltage range (and thus noise rejection) as well as minimizing the effect of ambient light.

Measuring the RPM from this signal can be done in three different ways. The first is to have a piece of tape of a known length, thus how long the pulse is corresponds to the actual rotational speed. However, since there is only one piece of tape on the motor, one can also measure the length between two rising, or two falling edges. This method is not dependent on tape length and thus more robust. Lastly, the number of pulses can be counted for a known time. The last two counting methods are both standard techniques for frequency counters.

First iteration measurement setup

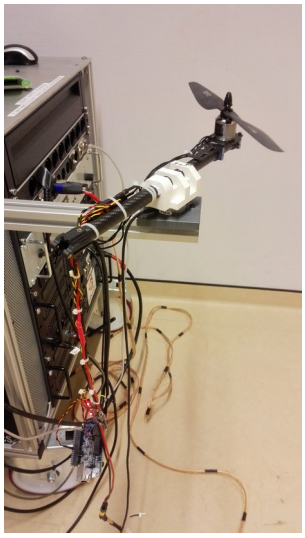


Figure 24: Measurement setup as used with the first iteration arm-to-sensor mount.

Note that in figure 24, the sensor direction is correct, yet the propeller is closer to the cart and wall because the sensor to bar mounting is not done as intended. Having the propeller closer to the wall and cart is not ideal since secondary aerodynamic effects could occur from winds and wind currents from effects of said walls and cart. Another note is that the connection is close to the motor. This is done to confirm that the maximum torque is within sensor specifications before extending the arm length.

Another non ideal part of this system is that the electronics are hanging in the air. The wiring harness is partly completed. This is also addressed in the second iteration of the non-tilting measurement system.

Second iteration measurement setup

The second iteration measurement setup was used to achieve the final results for the non tilting measurements. Because it worked well for these measurements, the arm-to-sensor and sensor-to-boikon mounting system was used for the tilting measurements as well.

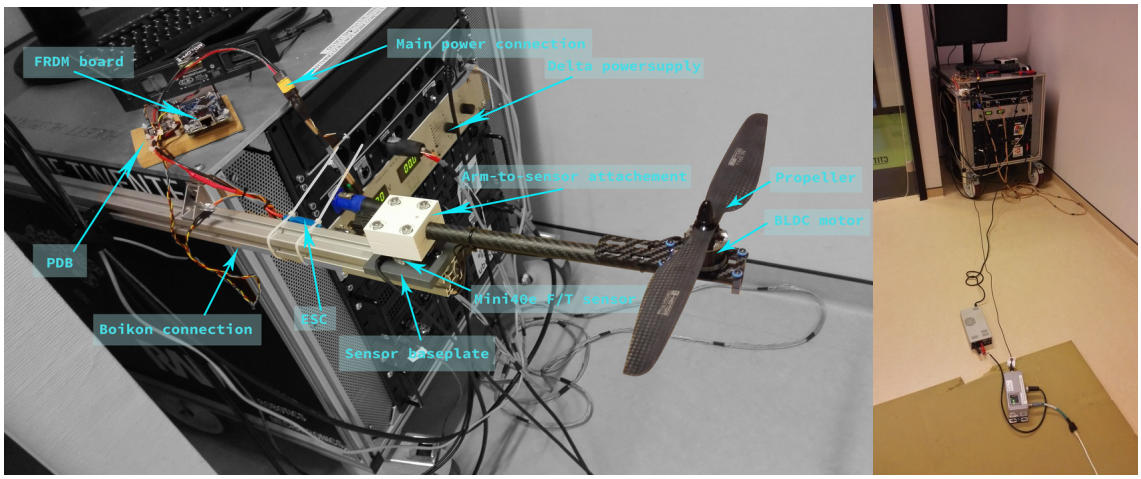


Figure 25: Overview of the measurement setup with all parts indicated. On the right, the F/T sensor base and power supply. Note the second iteration arm-to-sensor mount.

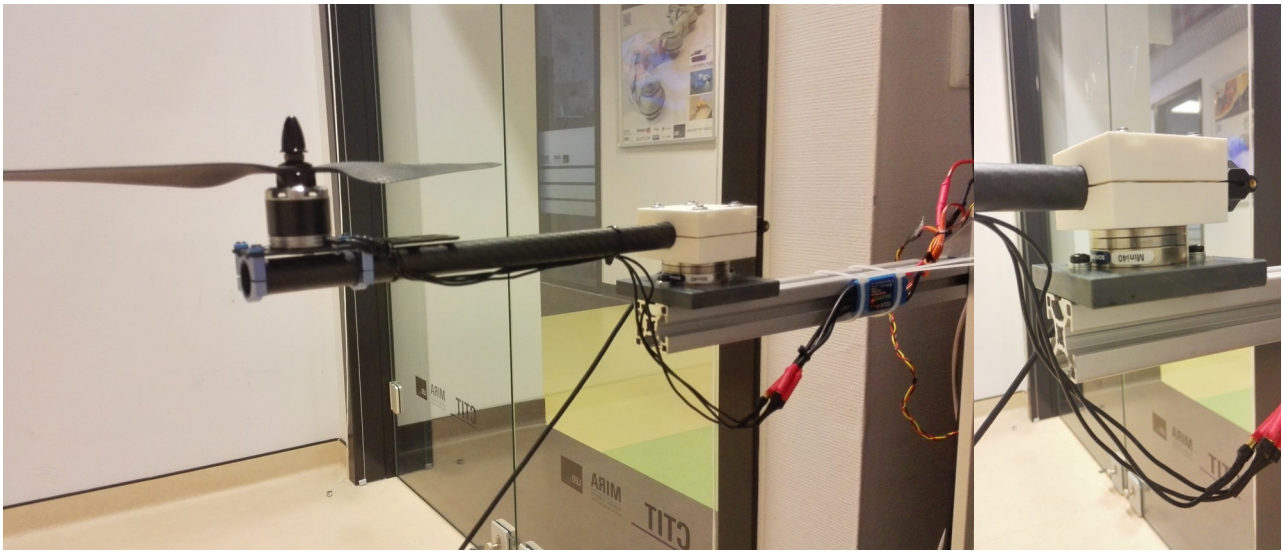


Figure 26: Close up of the arm, and the sensor attachment, the ESC (blue) attached to the BLDC motor. On the right, the sensor-to-arm attachment and perspex sensor-to-boikon plate.

3.3.3 Results

The first measurements being shown are the ramp profile measurements to identify secondary aerodynamic forces and possibly extract modelling parameters.

Ramp profile measurements

Using the second iteration measurement setup, four runs were done to retrieve four sets of data to use. The data was afterwards split into two sections, one to be used for the ramp profile, the other for the step response. For the ramp measurements the four runs were used to average out any errors in measurements and some process variations. The used voltage during this measurement is around 13.8V. This voltage should have been 14.8V, this is explained later in the discussion.

Figure 27 shows an overview of the four datasets. They are numbered dataset 2 through 5, this is because dataset 1 was invalid because the propeller was a clockwise rotating propeller, whilst the motor was spinning counter clockwise.

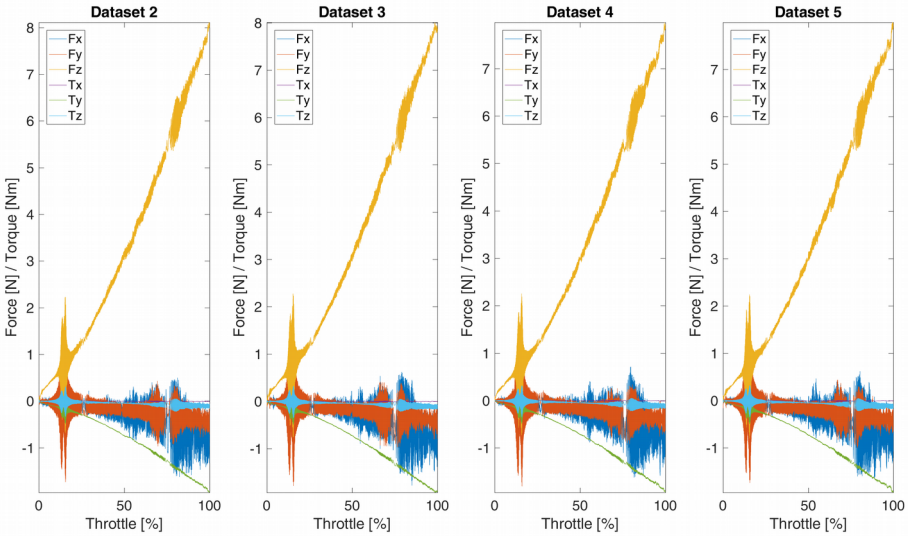


Figure 27: Overview of the four datasets

The first interesting thing to note is the oscillations near 14% throttle. The most probable explanation for these oscillations is that this is near a system resonance frequency. This resonance frequency could not be identified since the sampling frequency of the sensor is too low, only aliasing could be observed.

Having a closer look at the force in the Z direction and the torque in the Y direction, it can be seen that the expected squared relationship is indeed visible in the data. For this, second degree polyfits were made to fit the experimental data. All four datasets seem to generate polyfits reasonably close together. The average polyfit can be used for a model. Additionally, a lift coefficient solely dependent on x^2 was also fitted. This gives a lift coefficient of about 0.0009 (with a throttle percentage between 0 and 100, and as output force in Newtons).

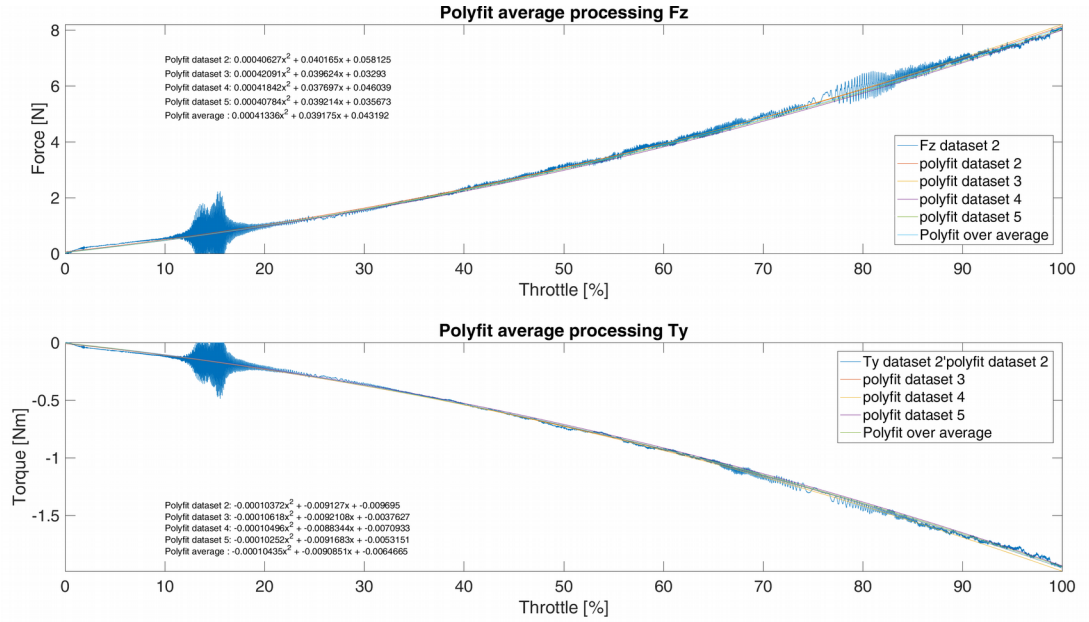


Figure 28: Fz and TY data together with polyfits

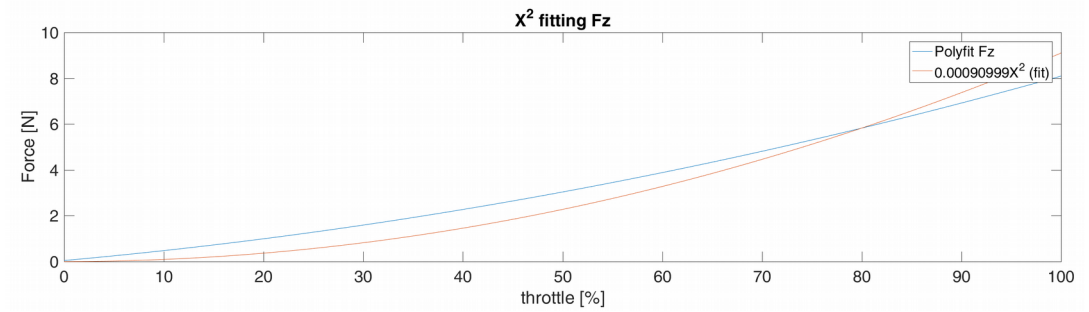


*Figure 29: Fitting an a*x^2 to the average polyfit for Fz*

After multiplying Fz with the approximate (measured) arm length of 24 cm, it can be seen that it maps very well to Ty, which confirms that there are negligible secondary aerodynamic effects present as torque in the Y axis.

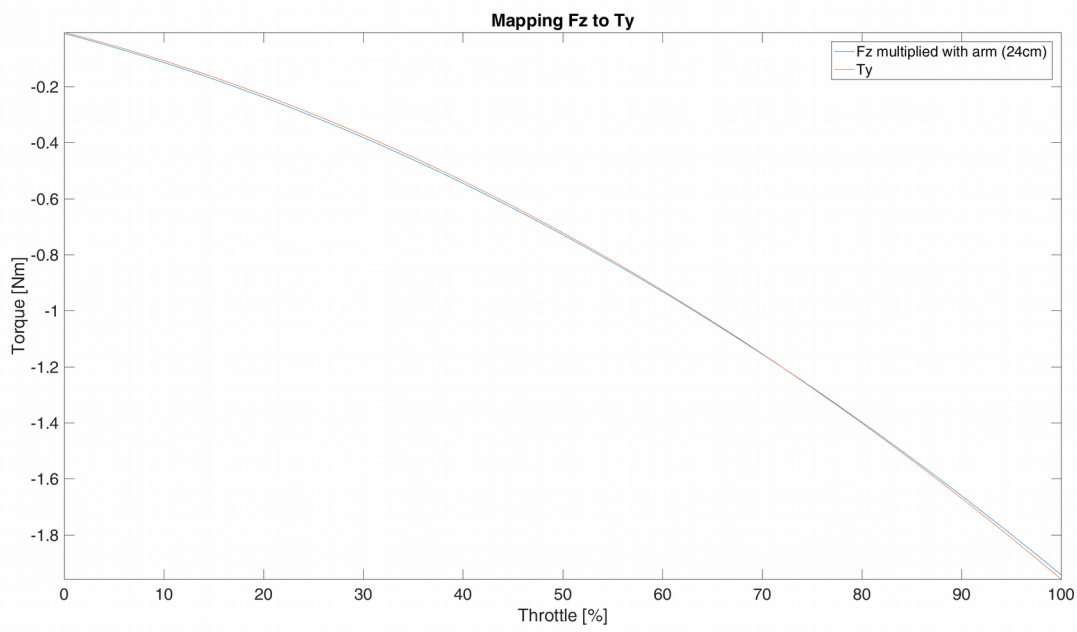


Figure 30: Fz multiplied with approximate arm length compared to Ty

As manufacturer data was available for comparison, it can be plotted against the found polyfit for Fz as well. This is shown in figure 31. As can be seen the shape matched reasonably well with the found results, although the manufacturer data shows higher thrust values. This could easily be explained that they have used a higher motor voltage for their experiments.

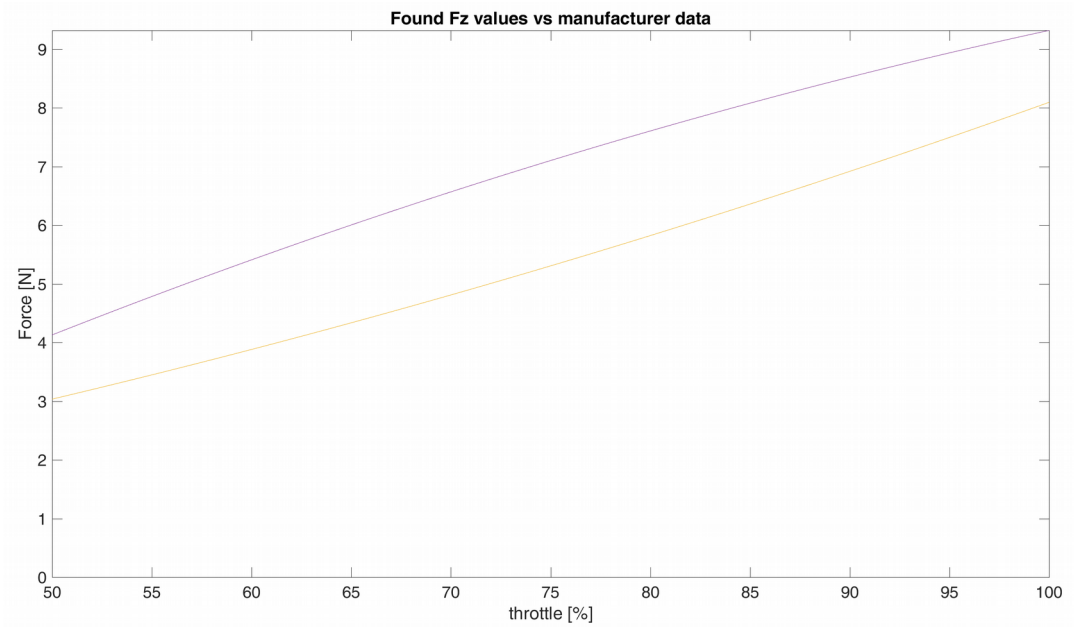


Figure 31: The polyfit of the manufacturer for Fz (purple), compared to the found polyfit for Fz (yellow).

The noise of both Fz and Ty was also tried to be approximated for a possible model. As seen the noise is not white noise, but approximating it as white noise may be sufficient for the model. The values for the mean and variance for this are shown in figure 32.

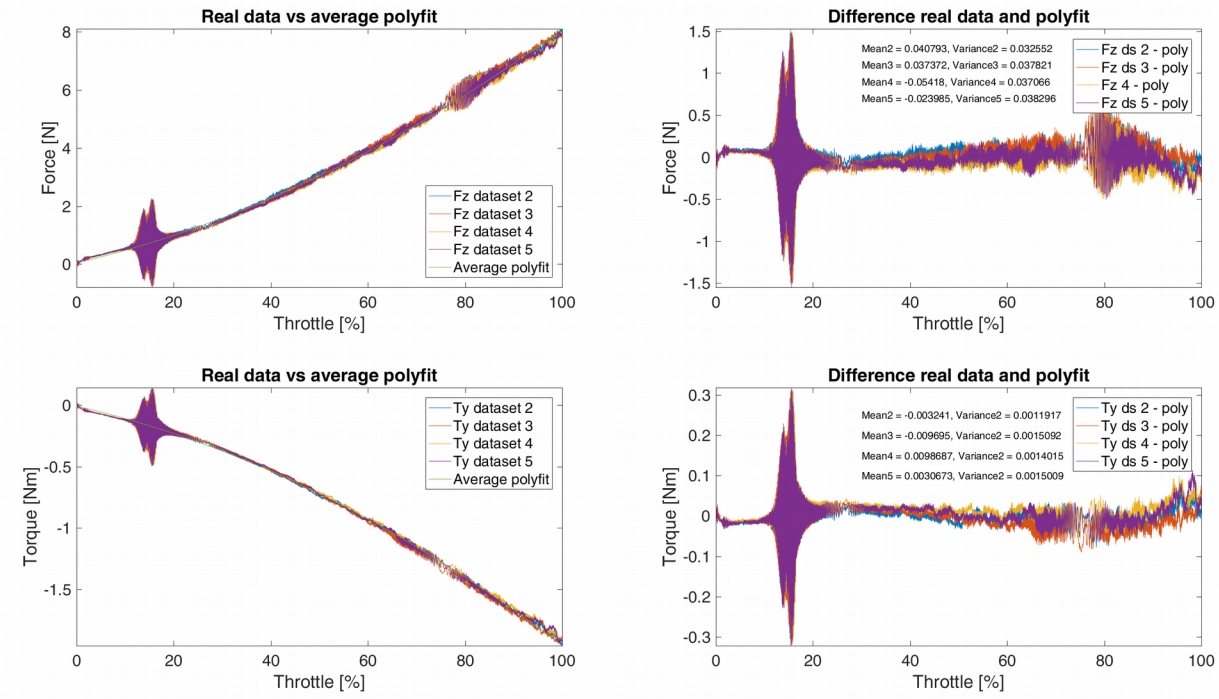


Figure 32: Noise analysis of the real data vs the polyfit

In Fx and Fy, it is expected to see the effects of the hub force and blade flapping phenomena. These are shown in figure 33, together with polyfits for all the dataset and a polyfit average.

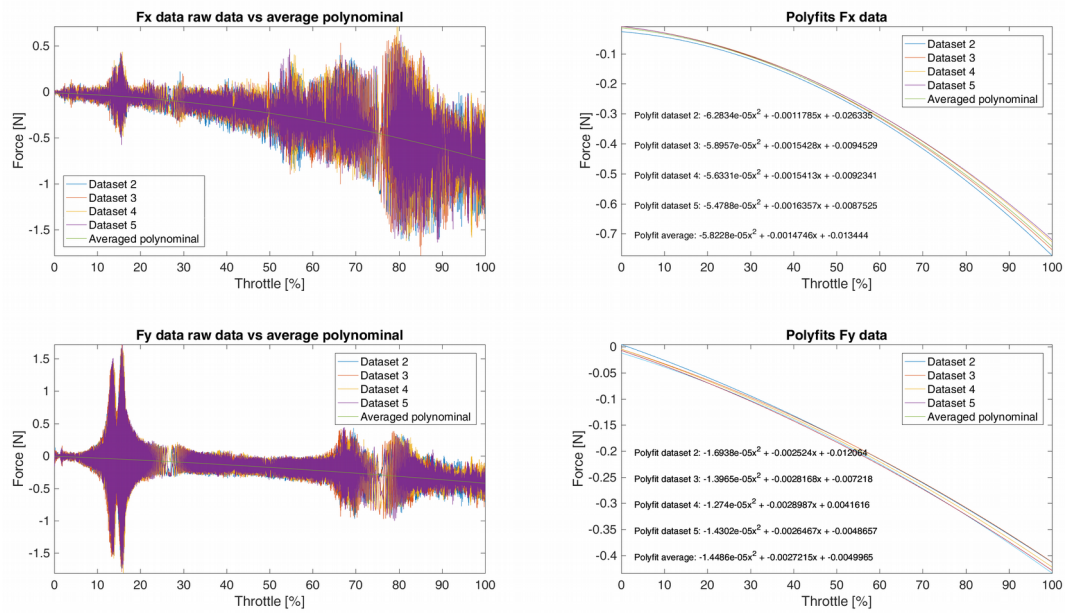


Figure 33: Polyfits for the Fx and Fy data

This data is then plotted as a percentage of Fz, which yields the following. From about 10% throttle onwards, both show around 10% the force of Fz, which is as expected and consistent with existing previous research.
What is interesting is that the force in the X axis increases with throttle percentage, whilst Fy drops slightly. The increase in the X axis is probably because the effect of blade flapping is largest here, but more research should be done to find the exact cause.

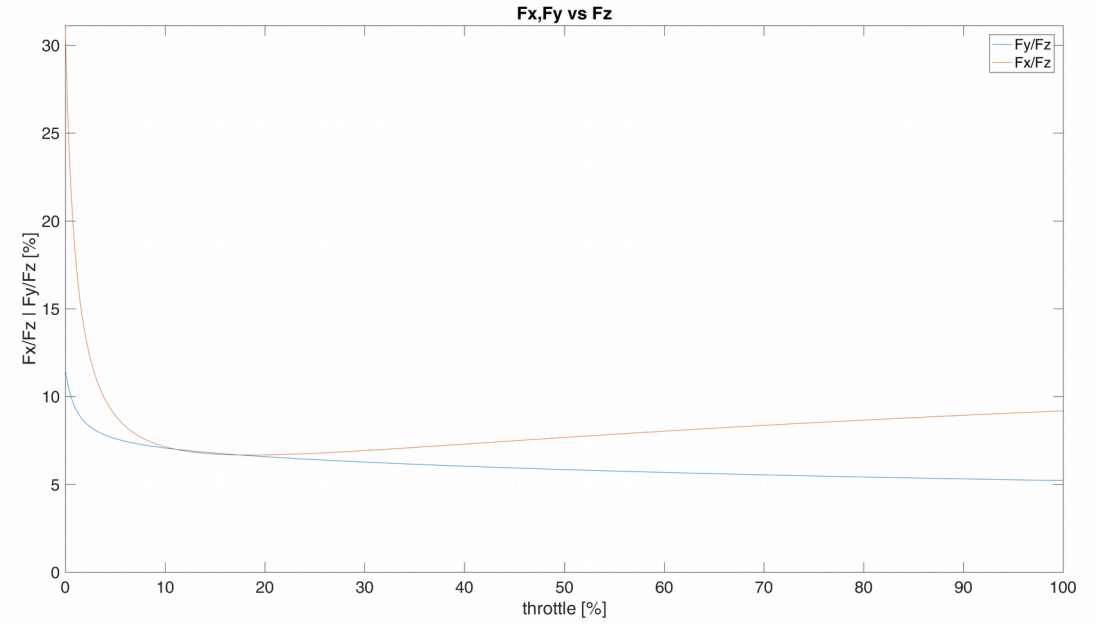


Figure 34: Fx and Fy as a percentage of Fz

Of Tx, no signal is expected. And as seen in the graph below, there indeed is only noise on this channel. After calculating both the mean and the variance it is seen that it has (nearly) zero mean, with a certain variance that can be used in the final model.

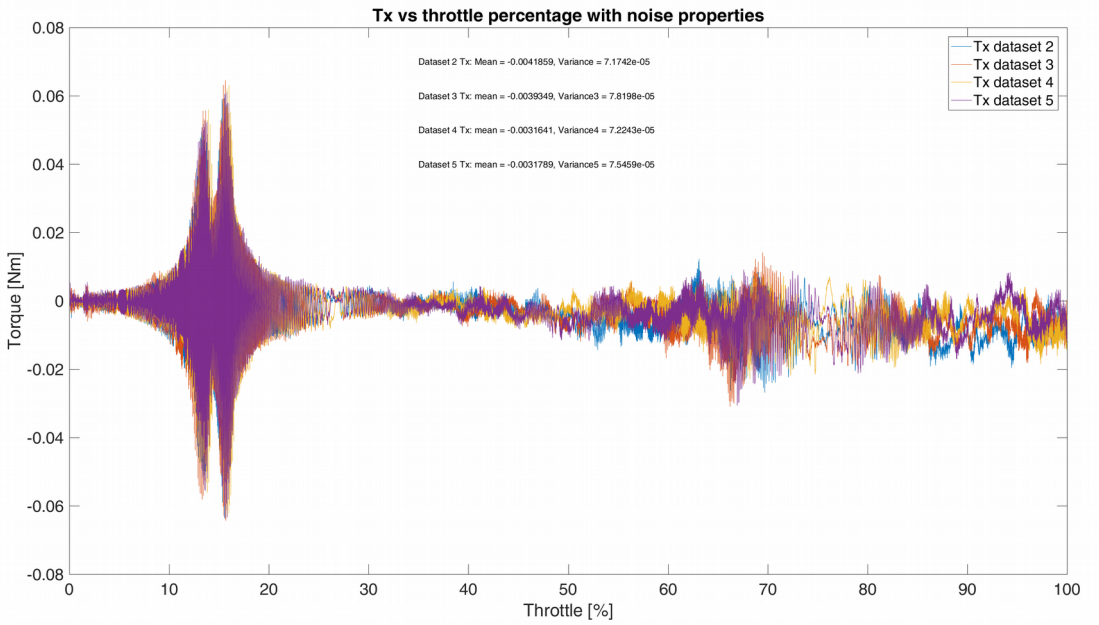


Figure 35: Tx noise properties

The previously discussed force in the Y direction maps to a torque in the Z axis. However, we expect to find another torque in the Z-axis, namely the drag torque. After subtracting the mapped torque from the force in the Y direction with an arm of 24cm, the following curve is left:

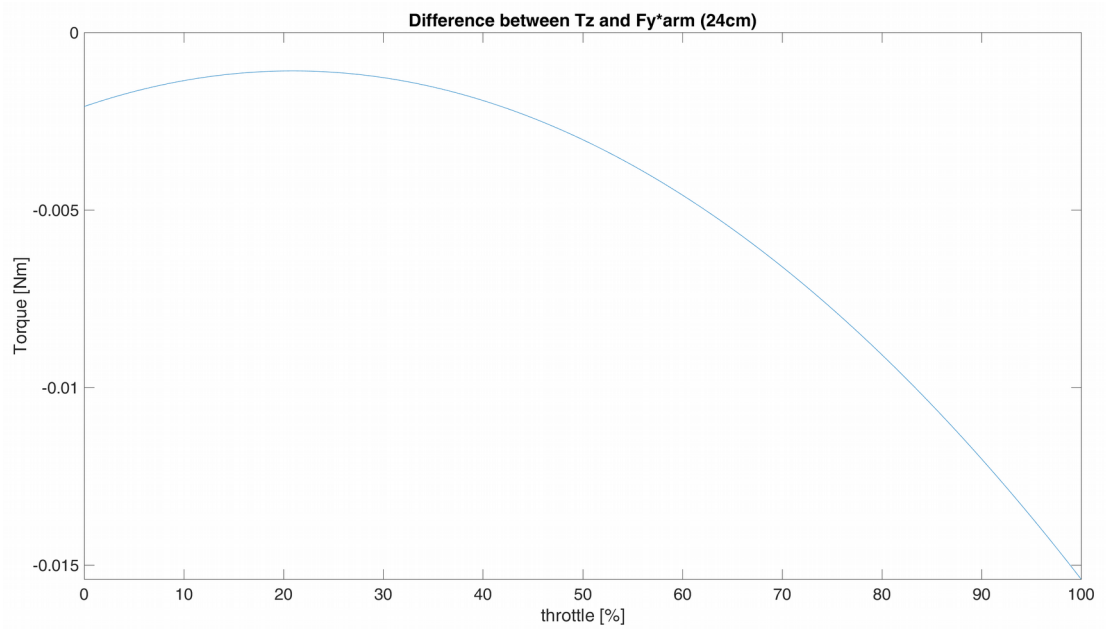


Figure 36: Difference between measured Tz and mapped Tz from Fy. Expected is the drag torque

However, the found torque is around 0.19% with respect to the thrust force. This is much lower than expected. Therefore, no polyfit was made to this corrected Tz.

Step response measurements

As discussed in the expected results, due to the nature of the PID controller in the ESC the response is expected to be second order. The data retrieved for the response is shown in figure 37. As can be seen, it has a slight overshoot which also indicates a second order system.

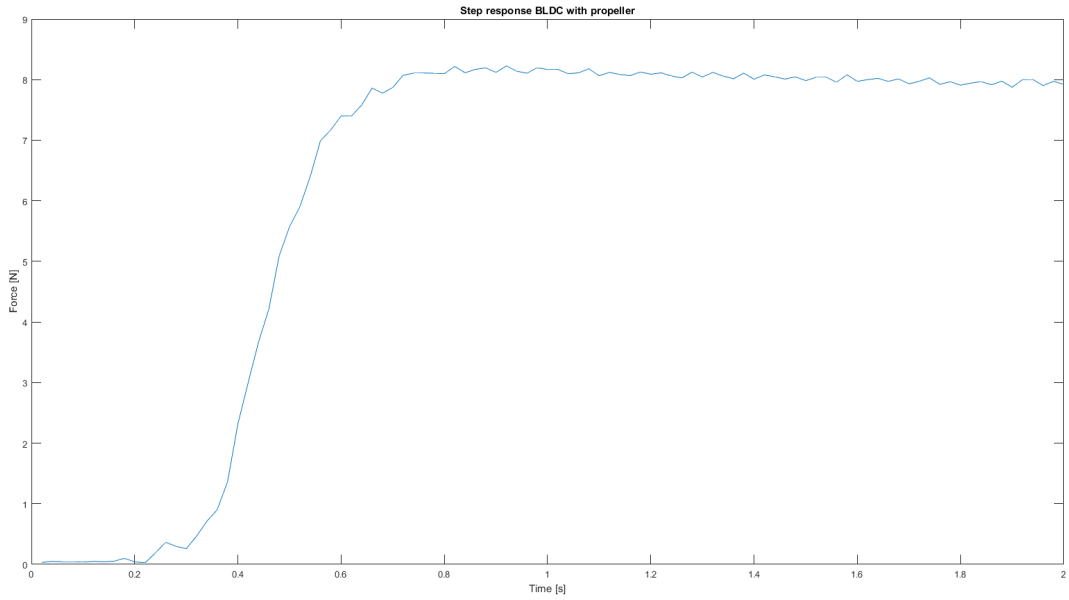


Figure 37: Response to a step input, 0 to 100% throttle.

To characterize the dynamic response of the BLDC motor, we can try to fit parameters to a typical second order transfer function in the form of:

$$F_z = K \frac{\omega^2}{s^2 + 2\zeta\omega s + \omega^2} \quad eq.\ 3$$

These parameters can be determines by looking at the step response.

The damping term is relativly easy to read from the step response, as it is given by the steady state after the step response. This is (for a given input voltage of 13.66V for the ECU) an output of around 8[N] for a throttle of 1, if it is mapped from 0 to 1. Thus the gain K of the system is 8.

As for the damping, a look can be had at standard second order transfer graphs with different damping. These curves can then empirically be chosen.

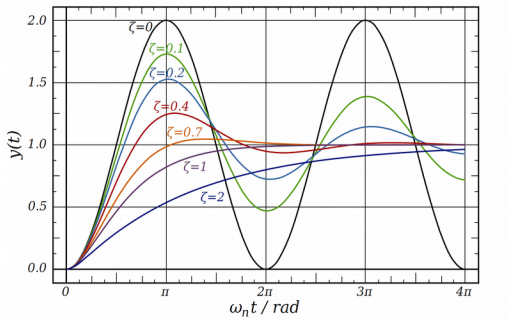


Figure 38: Second order systems with different damping rates[22]

From this, the damping appears to be just a bit above 0.7, so for our system 0.8 is chosen. This value can be varied and tuned later.

The last parameter of interest is the rise time. If we have a look at the step response, the rise time is about 0.9-0.2=0.7[s]. If we look at the standard step responses this matches with about 2 pi radians. With this information we can calculate omega.

$$\omega = \frac{2\pi}{t_{rise}} = \frac{2\pi}{0.7} \approx 7.2722[rad/s], \zeta \approx 0.8, K \approx 8 \quad eq.\ 4$$

Filling this into the standard second order equation yields:

$$F_z = K\frac{\omega^2}{s^2 + 2\zeta\omega s + \omega^2} = \frac{1007}{s^2 + 17.95s + 125.9} \quad eq.\ 5$$

If we now plot this response to see how it matches with our original data set:

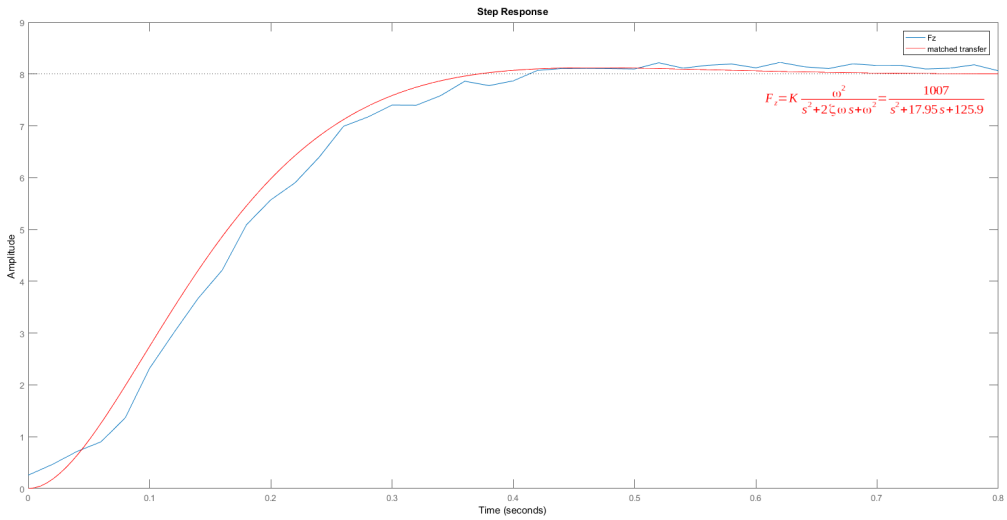


Figure 39: Step response with fitted second order S transfer functions

As can be seen, this transfer function is not a perfect match, but gives a decent starting point for building a model of the BLDC motor, ESC, arm and propeller.

Often it is necessary to have a first order step response, because these are in general easier to work with mathematically. For completeness sake a first degree S transfer is also fitted to the step response.

First of all, we have to identify the gain and time constant of the system. The gain is relatively easy to find, it would be 8.097.
Since this will be an equation in the form of $F = u(t)(1 - e^{-t/\tau})G = 8u(t)(1 - e^{-t/\tau})$ *eq. 6*
We can find that the factor $1 - e^{-t/\tau}$ will be 0.86 for t=2τ . Thus:
F = 0.86*8 = 6.88 for t=2*tau.

0.22s -> 0.02385
1.04s -> 8.097

8.07315*0.86=6.942909

Which is at about t=0.56
This gives 2τ=(0.56-0.22)=0.34
τ = 0.17

Thus a first order response would be in terms of laplace:

$$F = G\frac{1}{\tau s+1} = 8.07315\frac{1}{0.17s+1} \quad eq.\ 7$$

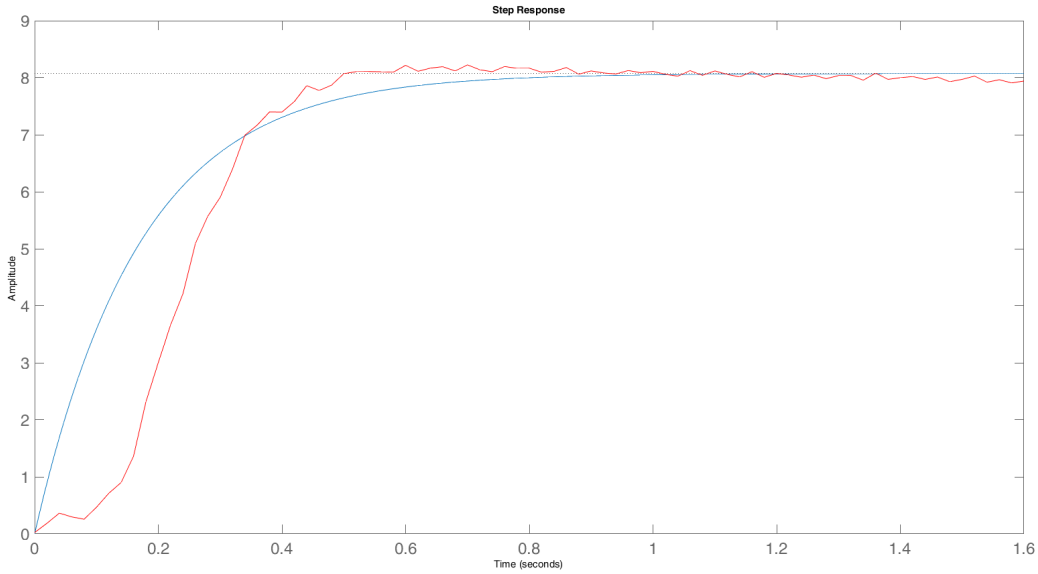


Figure 40: Step response with first order s-transfer fit

This is obviously a less good fit than the second order response, but could possibly be good enough for the intended model. This decision is left up to future work.

The measurement setup was also used to gain knowledge about the ESC used. The ESC takes a servo signal with a pulse-width that varies between 1 and 2 ms. The frequency of the signal is about 50hz, although the exact frequency is not of importance because the ESC reacts to absolute changes in pulse width. The ESC has a ‘middle’ position, in which it gives 0 throttle. From this middle position, a deviation towards 1ms would be turning the rotor in one direction, towards 2ms pulse width turning the motor the other direction. The exact middle point was not at the expected 1.5ms, but at 1.45ms.

After doing some initial experimentations, the ESC/motor combination only allowed limited propeller reversal. When going up from 1.45 towards 2ms, it behaved as expected, however when going down from 1.45ms to 1ms, the motor sometimes jams to a halt very suddenly. After this discovery, it was deemed as requiring too much time to figure out the root problem, and thus reverse propeller direction measurements were dropped.

3.4 Tilting rotor system and analysis

The second part of the experiment involves the tilting rotor measurements, these results will be later compared to the non-tilting results as to find whether the secondary aerodynamic effects are significant. This section will also contain the design of the tilting tip mechanism.

3.4.1 Analysis

Definition of axis

The definition of the axis remains the same as in the case of the stationary rotor. There is no apparent reason for changing the axis definition, and keeping it the same will simplify comparison between tilting and non-tilting results.

Expected forces and torques

The same goes for the expected forces and torques. However, the thrust vector will now not be a pure force in the Z direction, but a vector with a component in the Y and Z direction. Note however, that for the measurements it is only important to have a relative wind flowing over the rotor. The tilting measurements are done by making brushless DC motor go through a sine type movement (which will be discussed later) around the axis of the arm. If this is done with the BLDC pointing up as the centre point of the sine, then the maximum angular tilting velocity will be reached here. Without direct angle position feedback, it is hard to retrieve the positioning data. But it is known that the blades will see the largest relative wind with the highest tilting angular velocity, thus in the upright position. Absolute positioning isn't of importance because at this largest relative rotor wind, the phenomena of interest can still be investigated. In this case, the forces and torques are present in the sensor the same way as for the non-tilting measurements.

Expected results

The same omega squared dependency is expected for most forces as with the non-tilting measurements. However, a larger effect is expected of forces that are dependent on relative wind over the rotor, such as dissymmetry of lift.

Next to these effects, also extra mechanical effects are expected in the results. Namely the gyroscopic effect (resistance to turning of a spinning inertia) and inertia of the rotating motor.

Measurement profile

As discussed before, for the tilting measurements the ESCs will be turned on with 100% throttle, whilst there will be a sine wave on the servo. This will make it possible to compare the relative forces and torques found in the non-tilting measurements if the rotor is pointed upwards.

Something to take into account for the sine wave, is the maximum speed the servo can attain. This is the limiting factor in the sine wave frequency that can be reached. The servo has a specified maximum turn rate of 60 degrees in 0.17~0.13[s]. Taking the lowest figure of 0.17 seconds, this corresponds to about 353 degrees per second, which is 6.161 rad/s.
The used function is the sine function expressed as

$$\alpha = A\sin(\omega t) = A\sin\left(\frac{2\pi}{T}t\right) \quad \textit{eq. 8}$$

with α =the angle, A=maximum angle that can be achieved, ω = speed of the sine.
Also known is the derivative of this function (after applying the chain rule):

$$\frac{d\alpha}{dt} = A\cos\left(\frac{2\pi}{T}t\right)\frac{2\pi}{T} \quad \textit{eq. 9}$$

This, together with the fact that it is known that

$$max\left(\frac{d}{dt}\sin\left(\frac{2\pi}{T}t\right)\right)=\frac{d}{dt}\sin\left(\frac{2\pi}{T}t\right),t=\frac{T}{2} \quad eq.\ 10$$

brings us to conclude that the maximum angular velocity is found at:

$$max\left(\frac{d\alpha}{dt}\right)=A\cos\left(\frac{2\pi}{T}\frac{T}{2}\right)\frac{2\pi}{T}=A\frac{2\pi}{T}\cos(\pi)=-A\frac{2\pi}{T} \quad eq.\ 11$$

Note that the maximum diversion of the sine wave is not important, as this only scales the maximum speed. Thus if a smaller diversion is used, it will result in a higher frequency that is needed to achieve the same maximum change in angle. In order not to make sure that the measurement setup operates within servo angle limits (somewhere around +-90 degrees), the arbitrary deviation of 45 degrees to either side was selected to use.
The maximum angle that can be achieved is 353 degree/s, solving for this with A=45 yields T=-0.8. Thus the maximum attainable frequency is 1.25 [Hz]. A safety margin is used such that the maximum rotation velocity is never reached, thus 1.2 [Hz] is used as the maximum frequency.

To find an additional phenomena, next to the maximum sine wave slower sine waves are also used. If there is a speed dependent effect, this would show up in these phenomena.

The final measurement profile used is thus putting the ESCs at full throttle, and then varying the frequency of the sine wave in steps. First the maximum speed is used of 1.20 hz which is then decreased further and further to investigate tilting angular velocity dependent effects. The following sine wave frequencies are used (all in hertz):
1.2 – 1.0 – 0.5 – 0.2 – 0.1
All of these will be done for a duration of 30 seconds, with a 5 second break in between during which the BLDC motor will keep blowing on full thrust.

Tilting actuator positioning

For the design of the tilting rotor, different actuator positions have to be analysed. From the literature discussed, a few different actuator types have arisen. These can now be analysed to determine which one would suit the measurement setup the best.

The advantage of having the actuator close to the main body, such as with type 2 and 3, is mass centralization. Due to having less mass at the outside of the quadcopter, it has decreased inertia whilst turning and thus a possible increase in mobility and response.
Another advantage is that there is no actuator in the airflow of the brushless motor, therefore allowing for maximum thrust.

Type 3 has as an additional advantage that the mounting of the tube to the frame is easy (no need for a rotational joint here). However, type 2 has as an advantage that a curved arm could be used, which would make it possible to put the centre of thrust, in-line with the centre of rotation of the arm. This would improve/simplify the control system for arm rotation.

However, having the actuator at the brushless motor side, allows for an actuator that needs to provide less torque (since it does not need to turn the motor and also the arm). There is no need to consider flex and torsion in the tube, and the mounting of the tube to the frame can be easily done with existing hardware. There is also less stress on the joint between the arm and the body, since the servo mass is close to the propulsion motors.
Type 2 and 3 would be the best option to implement in a final design, because there is no actuator reducing the thrust. Of these, 2 takes the preference due to the ability to design a custom arm which

would improve the control of the system. This is the final preferred option.

Of the different options, type 1 or 4 are easiest to implement, considering there is an existing airframe that is used. The way the airframe is build right now, would prohibit an axis to go through the middle of the tube. Since for the measurement set-up no new arms will be designed, type 2 and 3 would offer no advantages over type 1 or 4. The choice between type 1 or 4 depends on the type of actuator used, which suits best for the configuration.

Parts to be designed for the measurement setup

As the same sensor mounting system will be used as with the non-tilting measurements, a tilting tip mechanism needs to be designed. The system will be using a servo to rotate the brushless DC motor an equal amount of degrees in both directions with respect to the Z axis, around the X axis. Preferably the tilting system is light and robust enough to be used on a flying platform.

3.4.2 Design and realization

Iteration I

The first iteration of tilting design was made before the decision was made to go for a type 1 or 4. It was still available in case a backup was needed in the flyable iterations of the design. In practice, it did not seem to be needed.

The iteration is build up as a type 2 design, therefore a fairly complicated mounting plate would have been needed to attach the arm to the sensor.

The entire axle is rotated by attaching a solid custom gear via an outer fitting to the axle. This gear then driven by a gear attached to the servo motor.

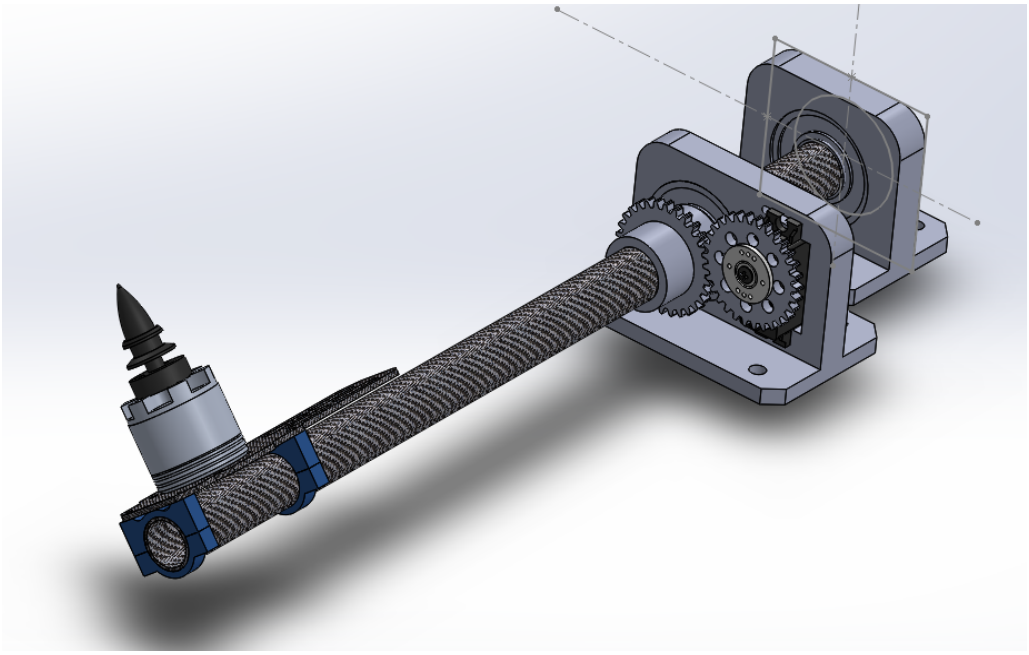


Figure 41: First iteration 'back up' design, clearly based on type 2.

Iteration II

The second iteration is build up as a type 4 design. It was chosen for this because of the construction of the servo. Essentially, an outer sleeve is needed to rotate around an inner rod.

Because of the cylinder-in-cylinder construction, it would not make sense to put the servo in the middle as in a type 1. It could be done but it would be hard to transfer all of the mechanical force provided by the thrust, to the arm reliable.

The design of the part is based on 2 main ‘sections’.
The first section is an inner axle that slides into the existing carbon tube. This inner axle is then fixed in place by bolts going through the axle and an associated sleeve over the axle. The reason for this is to prevent the inner axle from rotation. This section stays fixed with the quadrotor. At the end of the axle the a turning disc of the servo is glued.

The second section of the part is a sleeve that goes over the inner axle. The inner diameter of this sleeve is large enough, such that two bearings can be friction fitted between the inner axle, and the outer sleeve. The body of the servo itself is rigidly attached to the the second section as well. This whole section rotates (thus the servo body rotates with the BLDC motor).
Also attached on top of the second section, is a housing for the RPM sensor if it would be used.

As the part is designed now, almost all mechanical force is translated to the inner axle by means of the two bearings. This in effect makes sure that the servo does not need to transmit a high amount of mechanical load, and only performs the tilting.

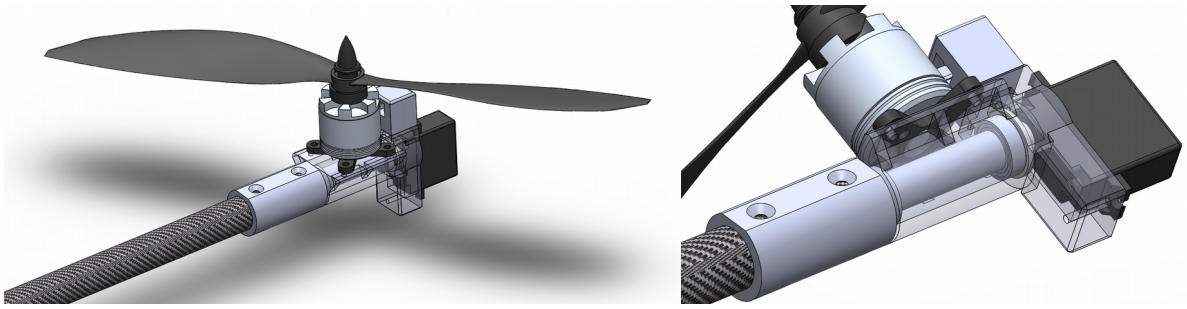


Figure 42: Third iteration tilting tip design, used for the measurements. On the right a close-up of the mechanical construction. Note the presence of the RPM sensor housing on top facing the BLDC.

The design was 3D printed, however, difficulties were found during construction in part assembly. It was tricky to both attach the servo wheel, and after that the servo in place because almost all was shielded by the second part section. A small amount of space was taken into account to place the wheel, however in practice it seemed quite tricky to glue it on. Thus the decision was taken to saw the part in half, attach the wheel and then glue the part back together. This worked for the measurement setup, but is of course not ideal for the final version, thus a third iteration was designed. Do note that all tilting measurements are done with this second iteration part.

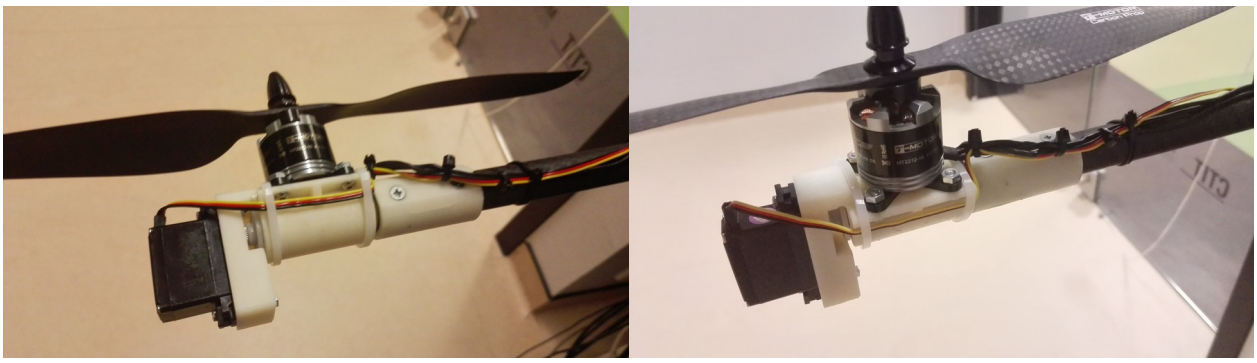


Figure 43: Second iteration part as used in the tilting measurements. The tie-wraps are used to keep the servo wires from dangling in the air and getting snatched in the rotor blades. Note the hole designed for easy access of the turning disc of the servo as well.

Iteration III

For the final iteration, the inner axle has been kept the same as in the second iteration. However, now the outer-sleeve can be disassembled into two parts for ease of construction. This would greatly improve the speed at which the part can be put together, and increase reliability as well. This design was not tested in tilting measurements, but should generally behave the same as the second iteration.

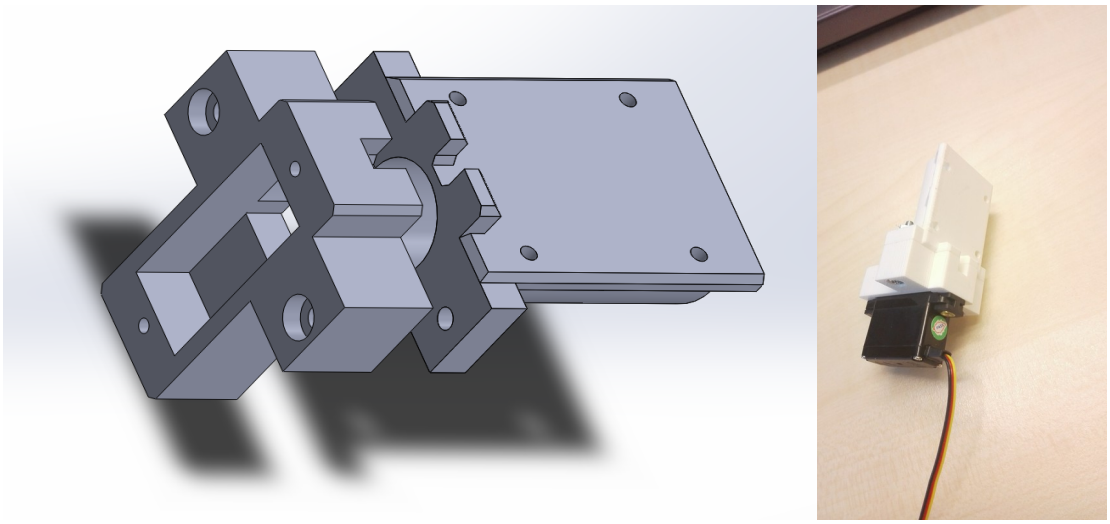


Figure 44: Third iteration part, 3D model on the left, 3D printed prototype on the right, ready to be tested on a tilting platform.

Measurement setup as used to gather the results

An overview of the measurement setup how it is used to gather the results is shown in figure 45. Most of the setup is kept the same as to the non-tilting measurements. For the electronics side the servo is now plugged into the wiring harness and the code modified.

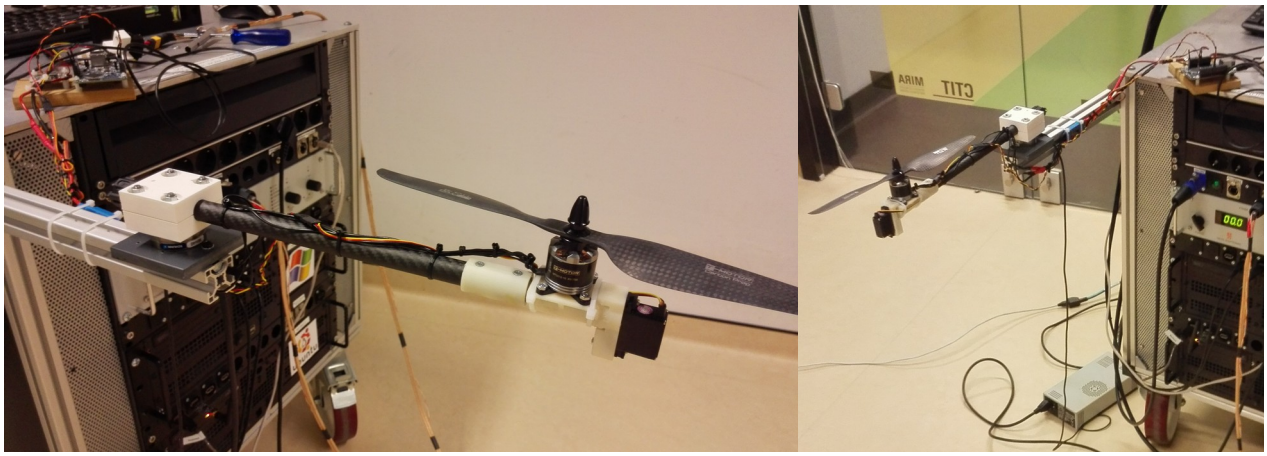


Figure 45: Tilting tip design as used for the measurements

3.4.3 Results

Tilting tip design

From the MoSCoW criteria on the measurement setup, all the must haves have been implemented, and also most of the should haves. Due to time constraints RPM measurements were dropped. Could haves could not be implemented due to lack of time.

Completed

Must have:

- ☑*Measure torque and force induced by the BLDC via an arm or known length*
- ☑*Control BLDC motor by PWM percentage*
- ☑*A variable tilting mechanism with servo as control*
- ☑*Be able to (independently) control BLDC and servo states according to a script*
- ☑*Solid motor mounting*
- ☑*Use both existing batteries and power supply for power*

Should have:

- ☑*Robust tilting mechanism ready to implement on a flying craft*
- ☑*Run of a single power source*
- ☑*Ability to reverse propeller direction*
- □ *Measure RPM*

Could have:

- □ *Control motor by RPM*
- □ *Control the measurement set-up by serial port*

It was still sufficient to reach the goal of attaining all desired measurements, as well as delivering an airborne ready tilting tip design. The non completed tasks are left as future work.

Tilting measurements

Measurement results of the tilting measurements are shown in figure 46. Five subsequent measurements showed similar results.

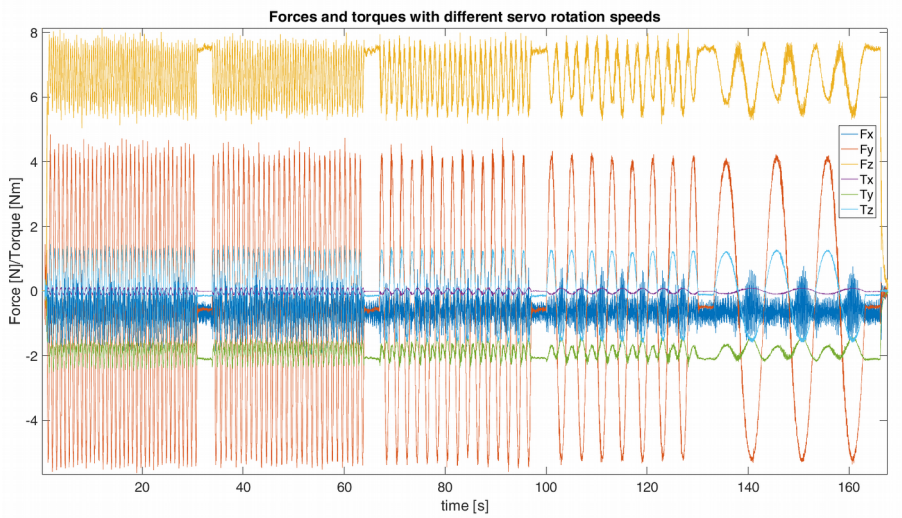


Figure 46: Measurement results of the titling setup. The different tilting speeds can clearly be seen, from 1.2hz to 0.1hz

No numerical analysis could be done on the data, instead it will be visually examined. This is done in two phases, first the tilting measurements are compared to the non-tilting data, after which the results of two different tilting velocities are compared.

Note that for analysis between the tilting and non-tilting measurements, only relative differences between forces and torques can be analysed, and no absolute values.

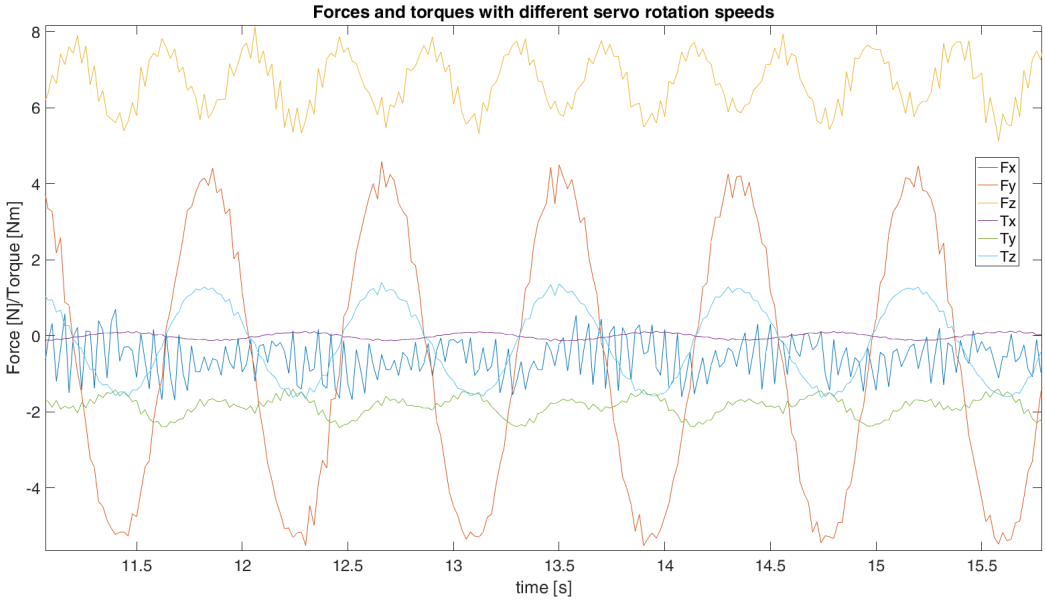


Figure 47: Section of measurement data at the fastest angle rotational velocity (1.2hz)

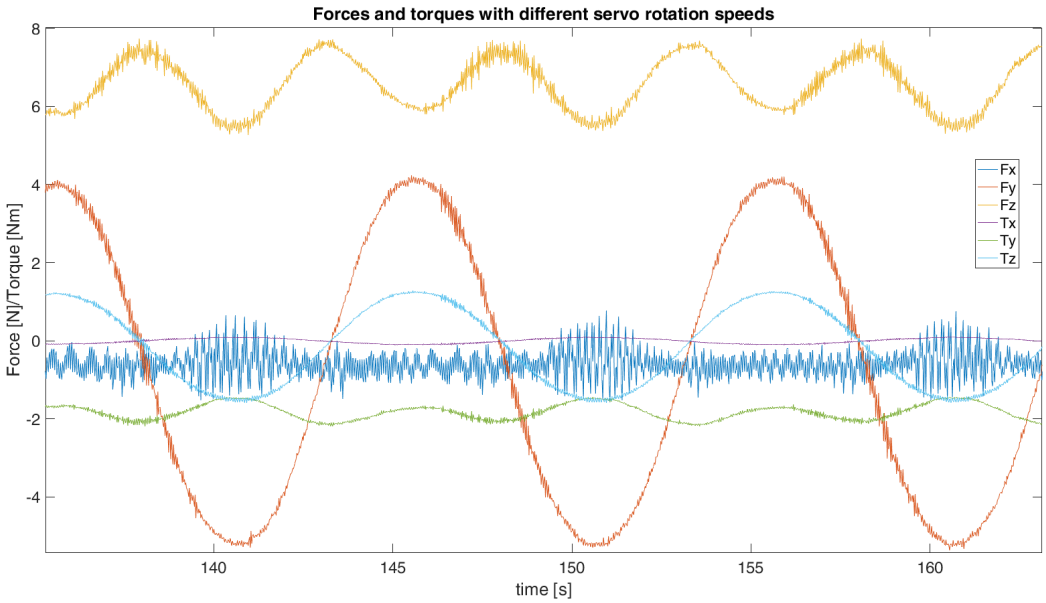


Figure 48: Section of measurement data at the slowest angle rotational velocity (0.1hz)

When observing Fz, it can be noted that the valleys do not have the same amplitude. There is a mismatch between Z force component when tilting to either the left, or right side. This is most probably because there is no way to directly set servo position, merely a given pulse-width that is assumed to correspond to a position. Most likely the platform does not tilt with a same angle to either left or right because a same deviation in pulse width from a ‘rest’ pulse width does not coincide with an equal angular displacement.

When looking at the peaks of Fz, we are looking at the point where the rotor is pointing straight upwards. Because of the sine measurement profile this is the same position where the rotor will have the largest angular velocity.

Considering at the point where the rotor is pointed upwards, this is the same situation as with the non-tilting measurements, thus a rough comparison can be made in relative differences between forces and torques. The slow tilting measurements (0.1 [Hz]) were taken to be compared with the non-tilting measurements, such that if these results match up, it can be concluded that the non-tilting measurements can be considered the same as stationary. This will make it able to compare the fast tilting measurements to a stationary case.

Mapping the peak of Fz of about 7.6 [N] to a torque in the Y axis with an arm length of about 30 [cm], gives a torque of around 2.28 [Nm] which matches up with the data which gives a torque of around -2.15 [Nm]. As with the non-tilting measurements, this difference could be due to the non exact known length of the arm.

During the non-tilting measurements Fx was around 10% of Fz at full throttle, whilst Fy was around 5%. Due to the noisy results, Fx data is not reliable, however when looking at the results the bias seems to be around 0.8 [N], which corresponds approximately with the expected 10% of Fz (7.6 [N]). The sinusoidal waveform of Fy is at its highest change per unit of time at the comparison point, thus a small change in time results in a fairly large change of Fy. Fy still seems to hover near the expected 0.3 [N].

Considering that after subtracting Fy times the arm length from Tz, the results was around -0.015 [Nm] at 100% throttle, doing these same operations on the tilting data is not useful. Mostly because of the large amount of noise found on the signals, and the signal being at the point with the largest change per unit time.

Now we can compare the fastest tilting measurements, with the slowest. The slowest measurements are so slow that these can be considered stationary for comparison purposes, and give an indication in what an increase in tilting angle velocity will do with observed forces and torques.

When comparing the slower and faster tilting setup, the torque in the Y axis seems to be slightly larger with the larger angular tilting velocity. This could be because of other mechanical causes such as inertia and gyroscopic effects.

Another observation is that there is an increase in torque in the X direction. This is because in this axis there is now an actuator moving a propeller with both inertia and a gyroscopic effect. This could resist this change in angle and cause a torque in the X axis.

There are no observable differences in the other torques and forces.

4 Conclusion

4.1 Conclusion

Tilting tip design

A measurement set up was created to produce measurements for both non-tilting and tilting measurements. All ‘must haves’ of the design have been fulfilled, together with most of the ‘should have’ requirements of the system. The tilting tip designed is robust enough for implementation in an over-actuated unmanned aerial vehicle.

Non-tilting measurements

The non-tilting measurements found results that mostly lined up with existing literature. However, it was found that the drag torque did not match up with expectations done by previous research.

Modelling parameters were extracted for most forces and torques were applicable. Next to the static modelling parameters, dynamic response has been characterized for the given ESC, BLDC and propeller combination. The response and parameters for propeller reversal could not be done due to ESC limitations.

Tilting measurements

No proper numerical analysis could be done of the tilting measurements. However, visual analysis was done. When comparing relative forces and torques to the relative forces and torques found in the non-tilting measurements, there does not seem to be a significant observable change in secondary aerodynamic forces and torques.

When comparing different tilting speeds, there are two torques that exhibit different behaviour, these are probably due to mechanical elements. Mechanical forces seem to have a bigger effect on the change in behaviour with larger tilting speed. There seem to be no significant changes or abnormalities in other forces and torques, and thus it is concluded that the secondary aerodynamic forces caused by the tilting rotor are negligible.

4.2 Discussion

One might think that the effect of Fz being different when either being tilted left or right might be due to the influence of walls, measurements were done to confirm or deny this suspicion. It was found that this was indeed not the case. The change of tilting left and right of Fz seemed independent when varying the distance to the walls.

It was attempted to do numerical analysis even though there was a lack of time, it is shown in appendix F. This is however not the proper way to do statistical analysis. Another suggested way to analyse the tilting data is covered in the recommendation and future work section.

The sampling rate used for the F/T sensor was a low (50 Hz). The readings were affected a lot by noise, especially in the case of the fastest tilting measurements. Redoing these measurements with 10x the sample rate and using filtering would be a good idea.

The tilting and non-tilting measurements use a different supply voltage supplied to the motors. This was a mistake and due to lack of time the measurements could not be repeated with the same motor voltage. However, this will only make a difference for absolute force and torque values, the relative differences should still be around the same when varying supply voltage (and thus rotational velocity) slightly.

4.3 Recommendations and Future work

Before implementing the tilting design on an actual airframe, long term stress testing is recommended. The design as build has survived around half an hour of ‘flight time’ in bursts of 2 minutes during measuring. This is not enough to conclude long term robustness of the design and construction techniques. The design did however show no visible signs of cracking or weakening.

Implement a type 2 measurement setup with custom arms is recommended. This would increase the maximum thrust the motors can deliver, whilst also improving the dynamic response of the tilting features.

It is now assumed that the servos track position accurately and precisely (because they are inherently positionally tracking devices). Can this assumption be made? And to what extent are they accurate? Could an improved tilting measurement setup be made using other forms of position controllers? And could existing servos be hacked to retrieve the angle from the internal potentiometer?

Further develop the opto-reflective sensor is needed. It only needs to be attached to the measurement setup and the electronics soldered and debugged. As for the circuit suggested, a simple comparator such as an LM393 which compares the RPM sensor output value to a known voltage. The circuit should also implement a hysteresis to prevent eradicate switching of the output at switching points. In essence this will form a Schmitt trigger with adjustable triggering voltage.

It is recommended to verify if the throttle percentage is linear with the rotational velocity of the blades. After finding a rotational velocity versus thrust, the thrust results of this research can be mapped to rotational velocity of the motor. What would be preferred is to get the RPM sensor working and repeat the measurements whilst recording rotational velocity as well.

An investigation into the found oscillations at 14% throttle is recommended. Are the oscillations present in the results a system resonance frequency? What are the primary factors contributing to this system resonance frequency (is it the measurement setup, or a combination of the BLDC and propellers which mostly influence this)? If the system resonance is independent of the measuring setup, it is worth considering this when building the controller.

More work into system identification would be recommended. System identification was not the primary purpose of this research, thus more effort in this field would significantly improve the model parameters and thus how well they model real life. Especially the dynamic response of motors can be done better, maybe optimize the transfer function by software instead of doing it by eye-balling it. The models found however, are considered to be close enough to be useful for a model.

More investigation into the lower than expected drag torque is needed to confirm either an error in measurements, error in the literature used as reference or a difference in the results between literature and found data because of the set up and components used.

Getting the propeller reversal to work and doing step measurements on this reversal and compare them with the dynamic response of the non-reversal case would be interesting.

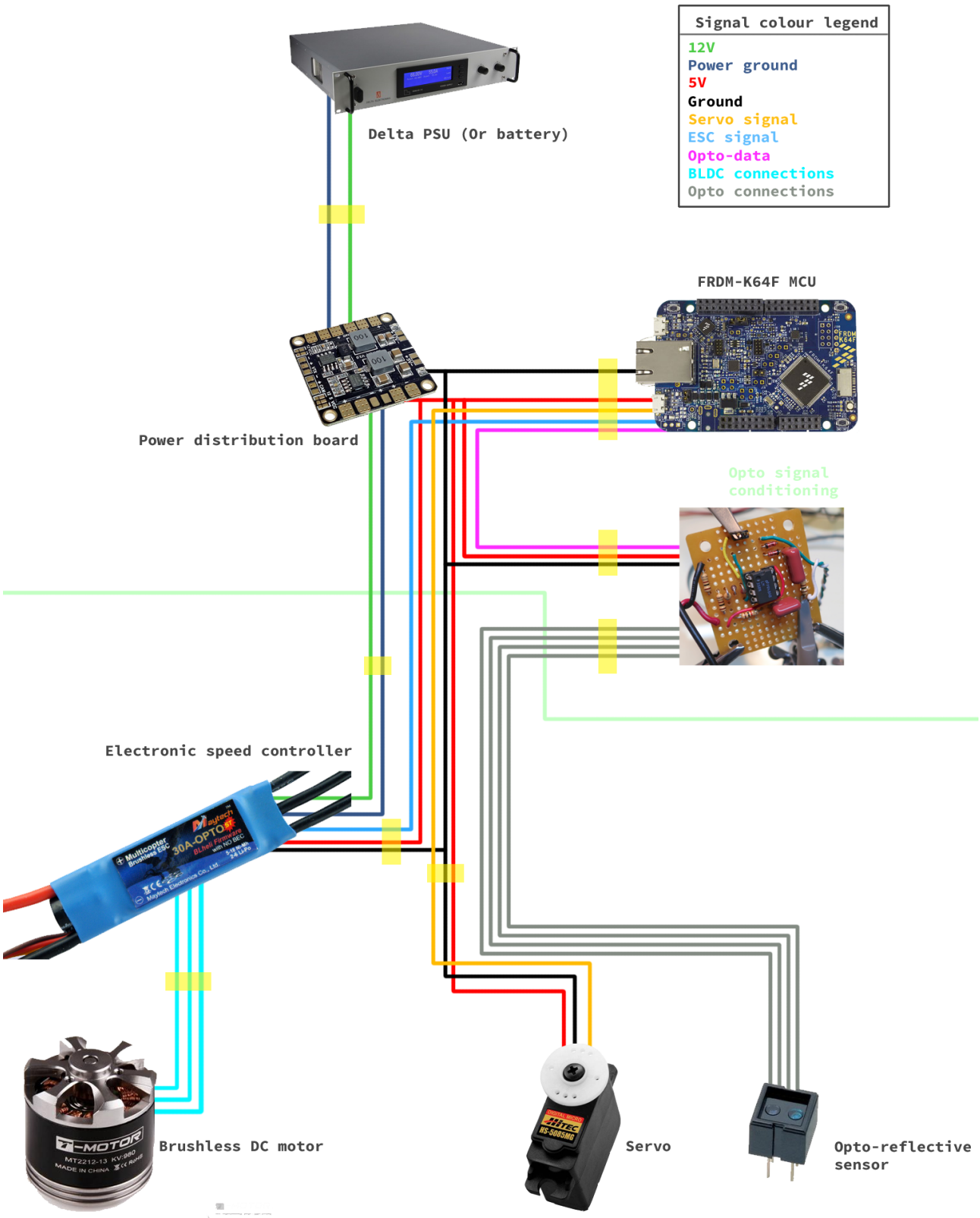
As for the tilting measurements, how large is the impact of gyroscopic effects, inertia and other mechanical forces and torques on the tilting measurements?

Try to confirm or deny the low drag torque, and get a coefficient for the drag torque from this.

As discussed before, there was not enough time to properly analyse the tilting measurements. It is recommended to use a different technique to properly analyse these measurements.
A better way to process the measurements is to draw fitting curves around Fz measurements. This is because Fz varies with the known phase difference compared to the angle of the measurements. This together with the fact that the maximum Fz corresponds to the rotor pointing straight upwards (rotating angle is zero) makes it possible to reconstruct the tilting angle. Using these fitting curves a function can be made of the measurement profile (angle). Then the found forces and torques in the sensor can be calculated back to forces and torques in the BLDC frame of reference using a rotation matrix. The output of this, can then be used to create the same comparisons as done with the non-tilting measurements. Also, because sine wave used gives a known velocity at a known position of the tilting system, the data can be used to create graphs of forces versus change in tilting angle.

5 Appendices

Appendix A: Wiring harness



Appendix B: Microcontroller software code for the non-tilting measurements

```
#include "mbed.h"
//#include <cmath>

#define PI 3.14159265359

// Mode of the controller, only make one of these 3 true. the rest must be false
#define MODERAMP true
#define MODESTEP false
#define MODESTEPREV false
#define MODESTAIR false

PwmOut ESC(D9);
DigitalOut led1(LED2);
DigitalIn buttonup(PTC6); //SW2
DigitalIn buttondw(PTA4); //SW3
Serial pc(USBTX, USBRX);



// Sets the output servo signal to somewhere between 0-100%
void setESC(float setpoint){
    //float dutyCycle = ((setpoint*1.4)+80)/100000; // Add the 1mS offset to the scaling of 0-100% (map to 1 to 2ms)
    float dutyCycle = ((setpoint)+100)/100000; // Add the 1mS offset to the scaling of 0-100% (map to 1 to 2ms)
    ESC.pulsewidth(dutyCycle); // We set by pulsewidth because servos dont care about absolute duty cycle %, but about pulsewidth
}

int main(){
    led1 = true;

    float cnt = 45;
    setESC(cnt);

    wait(2);


    /*
    while(true){

        if(buttonup == 0 && cnt<200){
            led1 = false;
            cnt+=5;
            setESC(cnt);
            wait_ms(500);
            led1 = true;
        }else if(buttondw == 0 && cnt>-100){
            led1 = false;
            cnt-=5;
            setESC(cnt);
            wait_ms(500);
            led1 = true;
        }
    wait_ms(1);
    }
    */

    // Ramp profile
    while(MODERAMP){
        for(float ramp=0; ramp<50; ramp+=0.003){
            setESC(cnt+ramp);
            wait_ms(10);
        }
        setESC(cnt+50);
        wait(5);
        setESC(cnt);
        wait(5);
        setESC(cnt+50);
        wait(5);
        setESC(cnt);

        led1 = false;
```

```
        while(true){};
    }

    // Ramp profile
    while(MODESTEP){

        setESC(cnt+40);
        wait(2);
        setESC(cnt+20);
        wait(2);
        setESC(cnt);

        led1 = false;

        while(true){};
    }

        // Ramp profile
    while(MODESTEPREV){

        setESC(cnt+20);
        wait(2);
        setESC(cnt-20);
        wait(2);
        setESC(cnt);

        led1 = false;

        while(true){};
    }

    // Step profile
    while (MODESTAIR) {

        led1 = false;
        setESC(10);
        wait(5);
        setESC(20);
        wait(5);
        setESC(30);
        wait(5);
        setESC(40);
        wait(5);
        setESC(50);
        wait(5);
        setESC(60);
        wait(5);
        setESC(70);
        wait(5);
        setESC(80);
        wait(5);
        setESC(90);
        wait(5);
        setESC(100);
        wait(5);
        setESC(0);

        led1 = false;

        while(true){};
    }
}
```

Appendix C: Microcontroller software code for the tilting measurements

```
#include "mbed.h"
//#include <cmath>

#define PI 3.14159265359

PwmOut ESC(D9);
PwmOut SERVO(A5);
DigitalOut led1(LED2);
DigitalIn buttonup(PTC6); //SW2
DigitalIn buttondw(PTA4); //SW3
Serial pc(USBTX, USBRX);



// Sets the output servo signal to somewhere between 0-100%
void setESC(float setpoint);

// Sets the output servo signal to somewhere between 0-100%
void setServo(float setpoint);

// This function implements a sine on the servo for 15 seconds before exiting, max frequency that can be used with the servo is 1.2484 hz
void sineservo(double frequency);

int main(){
   led1 = true;

   float zero = 45;
   setESC(zero);

   wait(10); // enough time to get clear of the room


   while(true){

      setESC(zero+50); // Turn on the motor full power

      sineservo(1.2); // Set frequency to just under theoretical max frequency

      setServo(43); // Set servo to middle and wait 3 secs
      wait(3);

      sineservo(1); // Set frequency to just under theoretical max frequency

      setServo(43); // Set servo to middle and wait 3 secs
      wait(3);

      sineservo(0.5); // Set frequency to just under theoretical max frequency

      setServo(43); // Set servo to middle and wait 3 secs
      wait(3);

      sineservo(0.25); // Set frequency to just under theoretical max frequency

      setServo(43); // Set servo to middle and wait 3 secs
      wait(3);



      sineservo(0.1); // Set frequency to just under theoretical max frequency

      setServo(43); // Set servo to middle and wait 3 secs
      wait(3);

      setESC(zero); // Turn off motor
      led1 = false; // Indicate safe to handle

      while(true){};
   }

}
```

```
// Sets the output servo signal to somewhere between 0-100%
void setESC(float setpoint){
   //float dutyCycle = ((setpoint*1.4)+80)/100000; // Add the 1mS offset to the scaling of 0-100% (map to 1 to 2ms)
   float dutyCycle = ((setpoint)+100)/100000; // Add the 1mS offset to the scaling of 0-100% (map to 1 to 2ms)
   ESC.pulsewidth(dutyCycle); // We set by pulsewidth because servos dont care about absolute duty cycle %, but about pulsewidth
}


// Sets the output servo signal to somewhere between 0-100%
void setServo(float setpoint){
   float dutyCycle = ((setpoint*1.4)+80)/100000; // Add the 1mS offset to the scaling of 0-100% (map to 0.8 to 2.2ms)
   SERVO.pulsewidth(dutyCycle); // We set by pulsewidth because servos dont care about absolute duty cycle %, but about pulsewidth
}

// This function implements a sine on the servo for 15 seconds before exiting, max frequency that can be used with the servo is 1.2484 hz
void sineservo(double frequency){

// Max servo speed is about 60degrees in 0.17sec, so with some trigonetry, the max frequency of the sine going from 45 to -45 is 1.2484 hz
// THe max deviation for +-45 is 14 to 72% (43% middle) found empircally for the setServo percentage

 double timeconst = (2*PI*frequency); // This is the cosine time constant used

 int cntdwn = 3000; // 30 second countdown timer before exit

 // Now devide the stepsize for the sine up in steps of 10ms each

 double timecount = 0;

 double sineout = 0;

   while(cntdwn>0){

   // Calculate the sine scalign factor
   sineout = sin(timeconst*timecount);

   // Scale the output to +45 to -45 degreees, which is 14 to 72 (middle is 43) in percentages
   sineout = (sineout*29) + 43;

   setServo(sineout);

   wait_ms(10);
   cntdwn--;
   timecount+=0.01;
   }

}
```

Appendix D: Non-tilting measurements matlab script

```
%%
% BEFORE RUNNIGN THE SCRIPT, PLEASE IMPORT RAMP2 UNTIL RAMP 5 INTO THE
% WORKSPACE

% Directory to save pictures in
figStr = 'C:\Sync\Dropbox\SPECTORS\boi okken\Data\Results 02062017\Data seperated\Ramp\Graphs final\';
%figStr = 'F:\dropbox sync\Dropbox\SPECTORS\boi okken\Data\Results 02062017\Data seperated\Ramp\Graphs final\';

%%
%Generate a thrust scale for all data files
elements2 = size(RDTSequence2);
throttle2 = RDTSequence2/(elements2(1)/100);
elements3 = size(RDTSequence3);
throttle3 = RDTSequence3/(elements3(1)/100);
elements4 = size(RDTSequence4);
throttle4 = RDTSequence4/(elements4(1)/100);
elements5 = size(RDTSequence5);
throttle5 = RDTSequence5/(elements5(1)/100);

%Plot them
figure(1);

subplot(1,4,1);
plot(throttle2,Fx2);
hold on;
plot(throttle2,Fy2);
hold on;
plot(throttle2,Fz2);
hold on;
plot(throttle2,Tx2);
hold on;
plot(throttle2,Ty2);
hold on;
plot(throttle2,Tz2);
hold off;
title('Dataset 2');
xlabel('Throttle [%]');
ylabel('Force [N] / Torque [Nm]');
axis([0 100 -inf inf])
legHan = legend('Fx','Fy','Fz','Tx','Ty','Tz');
set(legHan,'Location','northwest');
set(gca,'FontSize',18);

subplot(1,4,2);
plot(throttle3,Fx3);
hold on;
plot(throttle3,Fy3);
hold on;
plot(throttle3,Fz3);
hold on;
plot(throttle3,Tx3);
hold on;
plot(throttle3,Ty3);
hold on;
plot(throttle3,Tz3);
hold off;
title('Dataset 3');
xlabel('Throttle [%]');
ylabel('Force [N] / Torque [Nm]');
axis([0 100 -inf inf])
legHan = legend('Fx','Fy','Fz','Tx','Ty','Tz');
set(legHan,'Location','northwest');
set(gca,'FontSize',18);

subplot(1,4,3);
plot(throttle4,Fx4);
hold on;
plot(throttle4,Fy4);
hold on;
plot(throttle4,Fz4);
hold on;
```

```
plot(throttle4,Tx4);
hold on;
plot(throttle4,Ty4);
hold on;
plot(throttle4,Tz4);
hold off;
title('Dataset 4');
xlabel('Throttle [%]');
ylabel('Force [N] / Torque [Nm]');
axis([0 100 -inf inf])
legHan = legend('Fx','Fy','Fz','Tx','Ty','Tz');
set(legHan,'Location','northwest');
set(gca,'FontSize',18);

subplot(1,4,4);
plot(throttle5,Fx5);
hold on;
plot(throttle5,Fy5);
hold on;
plot(throttle5,Fz5);
hold on;
plot(throttle5,Tx5);
hold on;
plot(throttle5,Ty5);
hold on;
plot(throttle5,Tz5);
hold off;
title('Dataset 5');
xlabel('Throttle [%]');
ylabel('Force [N] / Torque [Nm]');
axis([0 100 -inf inf])
legHan = legend('Fx','Fy','Fz','Tx','Ty','Tz');
set(legHan,'Location','northwest');
set(gca,'FontSize',18);

set(gcf, 'Position', get(0,'Screensize'));
saveas(gcf,strcat(figStr,'Rampinputdata.fig'));
saveas(gcf,strcat(figStr,'Rampinputdata'),'png');

%%
%Generate polyfits Fz
polyfitFz2 = polyfit(throttle2, Fz2, 2);
polyfitFz3 = polyfit(throttle3, Fz3, 2);
polyfitFz4 = polyfit(throttle4, Fz4, 2);
polyfitFz5 = polyfit(throttle5, Fz5, 2);
%Generate polyfits Ty
polyfitTy2 = polyfit(throttle2, Ty2, 2);
polyfitTy3 = polyfit(throttle3, Ty3, 2);
polyfitTy4 = polyfit(throttle4, Ty4, 2);
polyfitTy5 = polyfit(throttle5, Ty5, 2);

%%
%%%%%%%%%%%%%%%% FZ AND TY POLYNOMINAL AVERAGING
%%%%% Process force in X axis
%Generate a throttle scale for discritization
throttle = (0:0.1:100);

%Generate the polynomicals for the Fz
p2 = @(x) polyfitFz2(1)*x.^2 + polyfitFz2(2)*x + polyfitFz2(3);
p3 = @(x) polyfitFz3(1)*x.^2 + polyfitFz3(2)*x + polyfitFz3(3);
p4 = @(x) polyfitFz4(1)*x.^2 + polyfitFz4(2)*x + polyfitFz4(3);
p5 = @(x) polyfitFz5(1)*x.^2 + polyfitFz5(2)*x + polyfitFz5(3);

%Discretize and average
p2discr = p2(throttle);
p3discr = p3(throttle);
p4discr = p4(throttle);
p5discr = p5(throttle);
polyavr = (p2discr + p3discr + p4discr + p5discr)/4;

%Poly fit the average
polyfitavr = polyfit(throttle,polyavr,2);
%And generate a polynominal function with this
```

```
Fzpoly = @(x) polyfitavr(1)*x.^2 + polyfitavr(2)*x + polyfitavr(3);

%Discretize to plot on the throttle scale
pavrdiscr = Fzpoly(throttle);
figure(2);

subplot(2,1,1);

plot(throttle2,Fz2);
hold on;
plot(throttle,p2discr);
hold on;
plot(throttle,p3discr);
hold on;
plot(throttle,p4discr);
hold on;
plot(throttle,p5discr);
hold on;
plot(throttle,pavrdiscr);
hold off;

title('Polyfit average processing Fz');
xlabel('Throttle [%]');
ylabel('Force [N]');

text(10, 7, ['Polyfit dataset 2: ' num2str(polyfitFz2(1)) 'x^2 + ' num2str(polyfitFz2(2)) 'x + ' num2str(polyfitFz2(3))]);
text(10, 6.5, ['Polyfit dataset 3: ' num2str(polyfitFz3(1)) 'x^2 + ' num2str(polyfitFz3(2)) 'x + ' num2str(polyfitFz3(3))]);
text(10, 6, ['Polyfit dataset 4: ' num2str(polyfitFz4(1)) 'x^2 + ' num2str(polyfitFz4(2)) 'x + ' num2str(polyfitFz4(3))]);
text(10, 5.5, ['Polyfit dataset 5: ' num2str(polyfitFz5(1)) 'x^2 + ' num2str(polyfitFz5(2)) 'x + ' num2str(polyfitFz5(3))]);
text(10, 5, ['Polyfit average : ' num2str(polyfitavr(1)) 'x^2 + ' num2str(polyfitavr(2)) 'x + ' num2str(polyfitavr(3))]);

axis([0 100 0 inf])
legHan = legend('Fz dataset 2','polyfit dataset 2','polyfit dataset 3','polyfit dataset 4','polyfit dataset 5','Polyfit over average');

set(legHan,'Location','southeast');
set(gca,'FontSize',18);

%%%%% Now the same for the torque in the Y axis
%Generate the polynomicals for the Ty
p2 = @(x) polyfitTy2(1)*x.^2 + polyfitTy2(2)*x + polyfitTy2(3);
p3 = @(x) polyfitTy3(1)*x.^2 + polyfitTy3(2)*x + polyfitTy3(3);
p4 = @(x) polyfitTy4(1)*x.^2 + polyfitTy4(2)*x + polyfitTy4(3);
p5 = @(x) polyfitTy5(1)*x.^2 + polyfitTy5(2)*x + polyfitTy5(3);

%Discretize and average
p2discr = p2(throttle);
p3discr = p3(throttle);
p4discr = p4(throttle);
p5discr = p5(throttle);
polyavr = (p2discr + p3discr + p4discr + p5discr)/4;

%Poly fit the average
polyfitavr = polyfit(throttle,polyavr,2);
%And generate a polynominal function with this
Typoly = @(x) polyfitavr(1)*x.^2 + polyfitavr(2)*x + polyfitavr(3);

%Discretize to plot on the throttle scale
pavrdiscr = Typoly(throttle);

subplot(2,1,2);

plot(throttle2, Ty2);
hold on;
plot(throttle,p2discr);
hold on;
plot(throttle,p3discr);
hold on;
plot(throttle,p4discr);
hold on;
plot(throttle,p5discr);
hold on;
plot(throttle,pavrdiscr);
hold off;
```

```
title('Polyfit average processing Ty');
xlabel('Throttle [%]');
ylabel('Torque [Nm]');

text(10, -1.4, ['Polyfit dataset 2: ' num2str(polyfitTy2(1)) 'x^2 + ' num2str(polyfitTy2(2)) 'x + ' num2str(polyfitTy2(3))]);
text(10, -1.5, ['Polyfit dataset 3: ' num2str(polyfitTy3(1)) 'x^2 + ' num2str(polyfitTy3(2)) 'x + ' num2str(polyfitTy3(3))]);
text(10, -1.6, ['Polyfit dataset 4: ' num2str(polyfitTy4(1)) 'x^2 + ' num2str(polyfitTy4(2)) 'x + ' num2str(polyfitTy4(3))]);
text(10, -1.7, ['Polyfit dataset 5: ' num2str(polyfitTy5(1)) 'x^2 + ' num2str(polyfitTy5(2)) 'x + ' num2str(polyfitTy5(3))]);
text(10, -1.8, ['Polyfit average : ' num2str(polyfitavr(1)) 'x^2 + ' num2str(polyfitavr(2)) 'x + ' num2str(polyfitavr(3))]);

axis([0 100 -inf 0])
legend('Ty dataset 2"polyfit dataset 2','polyfit dataset 3','polyfit dataset 4','polyfit dataset 5','Polyfit over average');
set(gca,'FontSize',18);

set(gcf, 'Position', get(0,'Screensize'));
saveas(gcf,strcat(figStr,'PolyfitFzTy.fig'));
saveas(gcf,strcat(figStr,'PolyfitFzTy'),'png');

%%
%%%%%%%%%%%%%%%% FZ NOISE PROFILE
%INPUTS:
%Fz is the input data array that should be supplied for the script to work
%RDTSequence is the input data array containing sample numbers

%Discretize over the input data array length
Fzpolydiscr2 = Fzpoly(throttle2);
Fzpolydiscr3 = Fzpoly(throttle3);
Fzpolydiscr4 = Fzpoly(throttle4);
Fzpolydiscr5 = Fzpoly(throttle5);
Fzpolydiscr = Fzpoly(throttle);

figure(3);

subplot(2,2,1);
plot(throttle2, Fz2);
hold on;
plot(throttle3, Fz3);
hold on;
plot(throttle4, Fz4);
hold on;
plot(throttle5, Fz5);
hold on;
plot(throttle, Fzpolydiscr);
hold off;
title('Real data vs average polyfit');
axis([0 100 -inf inf])
xlabel('Throttle [%]');
ylabel('Force [N]');
legHan = legend('Fz dataset 2','Fz dataset 3','Fz dataset 4','Fz dataset 5','Average polyfit');
set(legHan,'Location','southeast');
set(gca,'FontSize',16);

%Subtract to leave the rest
left2 = Fz2 - Fzpolydiscr2;
left3 = Fz3 - Fzpolydiscr3;
left4 = Fz4 - Fzpolydiscr4;
left5 = Fz5 - Fzpolydiscr5;

%Model as white noise (not entirely accurate but good enough for a model)
mn2 = mean(left2);
vr2 = var(left2);
mn3 = mean(left3);
vr3 = var(left3);
mn4 = mean(left4);
vr4 = var(left4);
mn5 = mean(left5);
vr5 = var(left5);

%Plot this too
subplot(2,2,2);

plot(throttle2, left2);
```

```
hold on;
plot(throttle3, left3);
hold on;
plot(throttle4, left4);
hold on;
plot(throttle5, left5);
hold off;
title('Difference real data and polyfit');
axis([0 100 -inf inf])
xlabel('Throttle [%]');
ylabel('Force [N]');

text(30, 1.3, ['Mean2 = ' num2str(mn2)  ', Variance2 = ' num2str(vr2)]);
text(30, 1.1, ['Mean3 = ' num2str(mn3)  ', Variance3 = ' num2str(vr3)]);
text(30, 0.9, ['Mean4 = ' num2str(mn4)  ', Variance4 = ' num2str(vr4)]);
text(30, 0.7, ['Mean5 = ' num2str(mn5)  ', Variance5 = ' num2str(vr5)]);

legend('Fz ds 2 - poly','Fz ds 3 - poly','Fz 4 - poly','Fz ds 5 - poly');
set(gca,'FontSize',16);

%%%% Now do the same for the torque
%Discretize over the input data array length
Typolydiscr2 = Typoly(throttle2);
Typolydiscr3 = Typoly(throttle3);
Typolydiscr4 = Typoly(throttle4);
Typolydiscr5 = Typoly(throttle5);
Typolydiscr = Typoly(throttle);

subplot(2,2,3);

plot(throttle2, Ty2);
hold on;
plot(throttle3, Ty3);
hold on;
plot(throttle4, Ty4);
hold on;
plot(throttle5, Ty5);
hold on;
plot(throttle, Typolydiscr);
hold off;
title('Real data vs average polyfit');
axis([0 100 -inf inf])
xlabel('Throttle [%]');
ylabel('Torque [Nm]');
legend('Ty dataset 2','Ty dataset 3','Ty dataset 4','Ty dataset 5','Average polyfit');

%Subtract to leave the rest
left2 = Ty2 - Typolydiscr2;
left3 = Ty3 - Typolydiscr3;
left4 = Ty4 - Typolydiscr4;
left5 = Ty5 - Typolydiscr5;

%Model as white noise (not entirely accurate but good enough for a model)
mn2 = mean(left2);
vr2 = var(left2);
mn3 = mean(left3);
vr3 = var(left3);
mn4 = mean(left4);
vr4 = var(left4);
mn5 = mean(left5);
vr5 = var(left5);
set(gca,'FontSize',16);

%Plot this too
subplot(2,2,4);

plot(throttle2, left2);
hold on;
plot(throttle3, left3);
hold on;
plot(throttle4, left4);
hold on;
plot(throttle5, left5);
```

```
hold off;
title('Difference real data and polyfit');
xlabel('Throttle [%]');
ylabel('Torque [Nm]');
axis([0 100 -inf inf])

legend('Ty ds 2 - poly','Ty ds 3 - poly','Ty ds 4 - poly','Ty ds 5 - poly');

text(30, 0.25, ['Mean2 = ' num2str(mn2)  ', Variance2 = ' num2str(vr2)]);
text(30, 0.20, ['Mean3 = ' num2str(mn3)  ', Variance2 = ' num2str(vr3)]);
text(30, 0.15, ['Mean4 = ' num2str(mn4)  ', Variance2 = ' num2str(vr4)]);
text(30, 0.10, ['Mean5 = ' num2str(mn5)  ', Variance2 = ' num2str(vr5)]);
set(gca,'FontSize',16);

set(gcf, 'Position', get(0,'Screensize'));
saveas(gcf,strcat(figStr,'NoiseFzTy.fig'));
saveas(gcf,strcat(figStr,'NoiseFzTy'),'png');

%%
%%%%%%%%%%%%%%%% Plot Fz vs Ty with arm
figure(4);

% discitize input data
Fzpolydiscr = Fzpoly(throttle);
Fzarm = Fzpolydiscr*-0.24;

Typolydiscr = Typoly(throttle);
Tyarm = Typolydiscr;

% Plot it
plot(throttle, Fzarm);
hold on;
plot(throttle, Tyarm);
hold off;

title('Mapping Fz to Ty');
xlabel('Throttle [%]');
ylabel('Torque [Nm]');
axis([0 100 -inf inf])

legend('Fz multiplied with arm (24cm)','Ty');
set(gca,'FontSize',18);

set(gcf, 'Position', get(0,'Screensize'));
saveas(gcf,strcat(figStr,'FzTyarm.fig'));
saveas(gcf,strcat(figStr,'FzTyarm'),'png');

%%
%%%%%%%%%%%%%%%% Tx NOISE PROFILE

%Plot the Tx
figure(5);

plot(throttle2, Tx2);
hold on;
plot(throttle3, Tx3);
hold on;
plot(throttle4, Tx4);
hold on;
plot(throttle5, Tx5);
hold off;
title('Tx vs throttle percentage with noise properties');
xlabel('Throttle [%]');
ylabel('Torque [Nm]');
legend('Tx dataset 2','Tx dataset 3','Tx dataset 4','Tx dataset 5');

%Calculate the mean and variance of the torque in the X axis
mn2 = mean(Tx2);
vr2 = var(Tx2);
mn3 = mean(Tx3);
vr3 = var(Tx3);
mn4 = mean(Tx4);
vr4 = var(Tx4);
```

```
mn5 = mean(Tx5);
vr5 = var(Tx5);

text(35, 0.07, ['Dataset 2 Tx: Mean = ' num2str(mn2)  ', Variance = ' num2str(vr2)]);
text(35, 0.06, ['Dataset 3 Tx: mean = ' num2str(mn3)  ', Variance3 = ' num2str(vr3)]);
text(35, 0.05, ['Dataset 4 Tx: mean = ' num2str(mn4)  ', Variance4 = ' num2str(vr4)]);
text(35, 0.04, ['Dataset 5 Tx: mean = ' num2str(mn5)  ', Variance5 = ' num2str(vr5)]);
set(gca,'FontSize',18);

set(gcf, 'Position', get(0,'Screensize'));
saveas(gcf,strcat(figStr,'TxNoise.fig'));
saveas(gcf,strcat(figStr,'TxNoise'),'png');

%%
%%%%%%%%%%%%%%%% Tz is useful for the drag torque constant

polyfitTz2 = polyfit(throttle2, Tz2, 2);
polyfitTz3 = polyfit(throttle3, Tz3, 2);
polyfitTz4 = polyfit(throttle4, Tz4, 2);
polyfitTz5 = polyfit(throttle5, Tz5, 2);

p2 = @(x) polyfitTz2(1)*x.^2 + polyfitTz5(2)*x + polyfitTz5(3);
p3 = @(x) polyfitTz3(1)*x.^2 + polyfitTz5(2)*x + polyfitTz5(3);
p4 = @(x) polyfitTz4(1)*x.^2 + polyfitTz5(2)*x + polyfitTz5(3);
p5 = @(x) polyfitTz5(1)*x.^2 + polyfitTz5(2)*x + polyfitTz5(3);

%Discretize and average
p2discr = p2(throttle);
p3discr = p3(throttle);
p4discr = p4(throttle);
p5discr = p5(throttle);
polyavr = (p2discr + p3discr + p4discr + p5discr)/4;

%Poly fit the average
polyfitavr = polyfit(throttle,polyavr,2);
%And generate a polynominal function with this
Tzpoly = @(x) polyfitavr(1)*x.^2 + polyfitavr(2)*x + polyfitavr(3);

figure(6);

subplot(2,1,1);
plot(throttle2,Tz2);
hold on;
plot(throttle3,Tz3);
hold on;
plot(throttle4,Tz4);
hold on;
plot(throttle5,Tz5);
hold on;
plot(throttle, Tzpoly(throttle));
hold off;

title('Tz data vs average polyfit');
xlabel('Throttle [%]');
ylabel('Torque [Nm]');
legend('Tz dataset 2','Tz dataset 3','Tz dataset 4','Tz dataset 5', 'Tz polyfit average');
axis([0 100 -inf inf])
set(gca,'FontSize',18);

subplot(2,1,2);

plot(throttle, p2(throttle));
hold on;
plot(throttle, p3(throttle));
hold on;
plot(throttle, p4(throttle));
hold on;
plot(throttle, p5(throttle));
hold on;
plot(throttle, Tzpoly(throttle));
hold off;

title('Polyfitted Tz data');
```

```
xlabel('Throttle [%]');
ylabel('Torque [Nm]');
legend('Tz polyfit dataset 2','Tz polyfit dataset 3','Tz polyfit dataset 4','Tz polyfit dataset 5', 'Tz polyfit average');
axis([0 100 -inf inf]);

text(10, -0.06, ['Polyfit dataset 2: ' num2str(polyfitTz2(1)) 'x^2 + ' num2str(polyfitTz2(2)) 'x + ' num2str(polyfitTz2(3))]);
text(10, -0.07, ['Polyfit dataset 3: ' num2str(polyfitTz3(1)) 'x^2 + ' num2str(polyfitTz3(2)) 'x + ' num2str(polyfitTz3(3))]);
text(10, -0.08, ['Polyfit dataset 4: ' num2str(polyfitTz4(1)) 'x^2 + ' num2str(polyfitTz4(2)) 'x + ' num2str(polyfitTz4(3))]);
text(10, -0.09, ['Polyfit dataset 5: ' num2str(polyfitTz5(1)) 'x^2 + ' num2str(polyfitTz5(2)) 'x + ' num2str(polyfitTz5(3))]);
text(10, -0.1, ['Polyfit average : ' num2str(polyfitavr(1)) 'x^2 + ' num2str(polyfitavr(2)) 'x + ' num2str(polyfitavr(3))]);
set(gca,'FontSize',18);

set(gcf, 'Position', get(0,'Screensize'));
saveas(gcf,strcat(figStr,'TzDragTorque.fig'));
saveas(gcf,strcat(figStr,'TzDragTorque'),'png');

%%
%%%%%%%%%%%%%%%% FX FY

figure(7);

hold on;
set(gca,'FontSize',18);

% Generate polyfits for the Fx and Fy of all datasets
fxpoly2 = polyfit(throttle2, Fx2, 2);
fypoly2 = polyfit(throttle2, Fy2, 2);
fxpoly3 = polyfit(throttle3, Fx3, 2);
fypoly3 = polyfit(throttle3, Fy3, 2);
fxpoly4 = polyfit(throttle4, Fx4, 2);
fypoly4 = polyfit(throttle4, Fy4, 2);
fxpoly5 = polyfit(throttle5, Fx5, 2);
fypoly5 = polyfit(throttle5, Fy5, 2);

fxpolyeq2 = @(x)fxpoly2(1)*x.^2 + fxpoly2(2)*x + fxpoly2(3);
fypolyeq2 = @(x)fypoly2(1)*x.^2 + fypoly2(2)*x + fypoly2(3);
fxpolyeq3 = @(x)fxpoly3(1)*x.^2 + fxpoly3(2)*x + fxpoly3(3);
fypolyeq3 = @(x)fypoly3(1)*x.^2 + fypoly3(2)*x + fypoly3(3);
fxpolyeq4 = @(x)fxpoly4(1)*x.^2 + fxpoly4(2)*x + fxpoly4(3);
fypolyeq4 = @(x)fypoly4(1)*x.^2 + fypoly4(2)*x + fypoly4(3);
fxpolyeq5 = @(x)fxpoly5(1)*x.^2 + fxpoly5(2)*x + fxpoly5(3);
fypolyeq5 = @(x)fypoly5(1)*x.^2 + fypoly5(2)*x + fypoly5(3);

%Discretize and average
fxpolyeq2discr = fxpolyeq2(throttle);
fypolyeq2discr = fypolyeq2(throttle);
fxpolyeq3discr = fxpolyeq3(throttle);
fypolyeq3discr = fypolyeq3(throttle);
fxpolyeq4discr = fxpolyeq4(throttle);
fypolyeq4discr = fypolyeq4(throttle);
fxpolyeq5discr = fxpolyeq5(throttle);
fypolyeq5discr = fypolyeq5(throttle);

fxpolyavrdiscr = (fxpolyeq2discr+fxpolyeq3discr+fxpolyeq4discr+fxpolyeq5discr)/4;
fypolyavrdiscr = (fypolyeq2discr+fypolyeq3discr+fypolyeq4discr+fypolyeq5discr)/4;

fxpolyavr = polyfit(throttle, fxpolyavrdiscr, 2);
fypolyavr = polyfit(throttle, fypolyavrdiscr, 2);

fxpolyeqavr = @(x)fxpolyavr(1)*x.^2 + fxpolyavr(2)*x + fxpolyavr(3);
fypolyeqavr = @(x)fypolyavr(1)*x.^2 + fypolyavr(2)*x + fypolyavr(3);

% Plot the regular data
subplot(2,2,1);

plot(throttle2, Fx2);
hold on;
plot(throttle3, Fx3);
hold on;
plot(throttle4, Fx4);
hold on;
plot(throttle5, Fx5);
hold on;
```

```
plot(throttle,fxpolyeqavr(throttle));
hold off;

title('Fx data raw data vs average polynominal');
xlabel('Throttle [%]');
ylabel('Force [N]');
legHan = legend('Dataset 2','Dataset 3','Dataset 4','Dataset 5','Averaged polynominal');
set(legHan,'Location','southwest');
axis([0 100 -inf inf]);
set(gca,'FontSize',14);

subplot(2,2,3);

plot(throttle2, Fy2);
hold on;
plot(throttle3, Fy3);
hold on;
plot(throttle4, Fy4);
hold on;
plot(throttle5, Fy5);
hold on;
plot(throttle,fypolyeqavr(throttle));
hold off;

title('Fy data raw data vs average polynominal');
xlabel('Throttle [%]');
ylabel('Force [N]');
legend('Dataset 2','Dataset 3','Dataset 4','Dataset 5','Averaged polynominal');
axis([0 100 -inf inf]);
set(gca,'FontSize',14);

%Now plot the polyfits
subplot(2,2,2);

plot(throttle,fxpolyeq2(throttle));
hold on;
plot(throttle,fxpolyeq3(throttle));
hold on;
plot(throttle,fxpolyeq4(throttle));
hold on;
plot(throttle,fxpolyeq5(throttle));
hold on;
plot(throttle,fxpolyeqavr(throttle));
hold off;

text(3, -0.3, ['Polyfit dataset 2: ' num2str(fxpoly2(1)) 'x^2 + ' num2str(fxpoly2(2)) 'x + ' num2str(fxpoly2(3))]);
text(3, -0.4, ['Polyfit dataset 3: ' num2str(fxpoly3(1)) 'x^2 + ' num2str(fxpoly3(2)) 'x + ' num2str(fxpoly3(3))]);
text(3, -0.5, ['Polyfit dataset 4: ' num2str(fxpoly4(1)) 'x^2 + ' num2str(fxpoly4(2)) 'x + ' num2str(fxpoly4(3))]);
text(3, -0.6, ['Polyfit dataset 5: ' num2str(fxpoly5(1)) 'x^2 + ' num2str(fxpoly5(2)) 'x + ' num2str(fxpoly5(3))]);
text(3, -0.7, ['Polyfit average: ' num2str(fxpolyavr(1)) 'x^2 + ' num2str(fxpolyavr(2)) 'x + ' num2str(fxpolyavr(3))]);

title('Polyfits Fx data');
xlabel('Throttle [%]');
ylabel('Force [N]');
legend('Dataset 2','Dataset 3','Dataset 4','Dataset 5','Averaged polynominal');
axis([0 100 -inf inf]);
set(gca,'FontSize',14);

subplot(2,2,4);

plot(throttle,fypolyeq2(throttle));
hold on;
plot(throttle,fypolyeq3(throttle));
hold on;
plot(throttle,fypolyeq4(throttle));
hold on;
plot(throttle,fypolyeq5(throttle));
hold on;
plot(throttle,fypolyeqavr(throttle));
hold off;

text(3, -0.2, ['Polyfit dataset 2: ' num2str(fypoly2(1)) 'x^2 + ' num2str(fypoly2(2)) 'x + ' num2str(fypoly2(3))]);
text(3, -0.25, ['Polyfit dataset 3: ' num2str(fypoly3(1)) 'x^2 + ' num2str(fypoly3(2)) 'x + ' num2str(fypoly3(3))]);
```

```
text(3, -0.3, ['Polyfit dataset 4: ' num2str(fypoly4(1)) 'x^2 + ' num2str(fypoly4(2)) 'x + ' num2str(fypoly4(3))]);
text(3, -0.35, ['Polyfit dataset 5: ' num2str(fypoly5(1)) 'x^2 + ' num2str(fypoly5(2)) 'x + ' num2str(fypoly5(3))]);
text(3, -0.4, ['Polyfit average: ' num2str(fypolyavr(1)) 'x^2 + ' num2str(fypolyavr(2)) 'x + ' num2str(fypolyavr(3))]);

title('Polyfits Fy data');
xlabel('Throttle [%]');
ylabel('Force [N]');
legend('Dataset 2','Dataset 3','Dataset 4','Dataset 5','Averaged polynominal');
axis([0 100 -inf inf]);
set(gca,'FontSize',14);

set(gcf, 'Position', get(0,'Screensize'));
saveas(gcf,strcat(figStr,'FxFyPoly.fig'));
saveas(gcf,strcat(figStr,'FxFyPoly'),'png');

%%
%%%%% GENERATE A PLOT THAT SHOWS THE PERCENTAGE OF FX/FY VS FZ

fypoldiscr = fypolyeqavr(throttle);
fxpoldiscr = fxpolyeqavr(throttle);
fzpoldiscr = Fzpoly(throttle);

% Calculate the percentages asociated with the discritized data
fyfzpercentage = -fypoldiscr./(fzpoldiscr./100);
fxfzpercentage = -fxpoldiscr./(fzpoldiscr./100);

figure(8);



plot(throttle,fyfzpercentage);
hold on;
plot(throttle,fxfzpercentage);
hold off;

title('Fx,Fy vs Fz');
xlabel('throttle [%]');
ylabel('Fx/Fz | Fy/Fz [%]');
legend('Fy/Fz','Fx/Fz');
axis([0 100 0 inf]);

set(gca,'FontSize',18);

set(gcf, 'Position', get(0,'Screensize'));
saveas(gcf,strcat(figStr,'FxFyvsFz.fig'));
saveas(gcf,strcat(figStr,'FxFyvsFz'),'png');


%%
%%%%% GENERATE A PLOT THAT SHOWS THE PERCENTAGE OF TZ VS FZ

tzpoldiscr = Tzpoly(throttle);
fzpoldiscr = Fzpoly(throttle);

% Calculate the percentages asociated with the discritized data
tzfzpercentage = -tzpoldiscr./(fzpoldiscr./100);

figure(9);

plot(throttle,tzfzpercentage);

title('Tz vs Fz');
xlabel('throttle [%]');
ylabel('Tz/Fz [%]');
axis([0 100 0 inf]);

set(gca,'FontSize',18);

set(gcf, 'Position', get(0,'Screensize'));
saveas(gcf,strcat(figStr,'TzvsFz.fig'));
saveas(gcf,strcat(figStr,'TzvsFz'),'png');
```

```
%% Fit an X^2 curve to the polyfits of Fz, Tz to gain the coefficient of lift and drag

% First define the function to fit to
fitxsquarefunc = @(c,x)c*x.^2;

%discritize the functions
fzpoldiscr = Fzpoly(throttle);
tzpoldiscr = Tzpoly(throttle);

% arbirtarily set initial point x0 at 1
x0 = 0;

% Do the fitting
[C1,resnorm,~,exitflag,output] = lsqcurvefit(fitxsquarefunc,x0,throttle,fzpoldiscr);
[C2,resnorm,~,exitflag,output] = lsqcurvefit(fitxsquarefunc,x0,throttle,tzpoldiscr);

figure(10);
subplot(2,1,1);
plot(throttle,fzpoldiscr);
hold on;
plot(throttle,fitxsquarefunc(C1,throttle));
hold off;

legend('Polyfit Fz',[num2str(C1) 'X^2 (fit)']);

title('X^2 fitting Fz');
xlabel('throttle [%]');
ylabel('Force [N]');

set(gca,'FontSize',18);

subplot(2,1,2);
plot(throttle,tzpoldiscr);
hold on;
plot(throttle,fitxsquarefunc(C2,throttle));
hold off;

legend('Polyfit Tz',[num2str(C2) 'X^2 (fit)']);

title('X^2 fitting Tz');
xlabel('throttle [%]');
ylabel('Torque [Nm]');

axis([0 100 -inf 0]);

set(gca,'FontSize',18);

set(gcf, 'Position', get(0,'Screensize'));
saveas(gcf,strcat(figStr,'xsquarefittingfztz.fig'));
saveas(gcf,strcat(figStr,'xsquarefittingfztz'),'png');


%% Polyfit Fy*arm vs subtract Tz

%discritize
tzpoldiscr = Tzpoly(throttle);
fypoldiscr = fypolyeqavr(throttle);

%multiply with arm
fypoldiscrarm = fypoldiscr*0.24;

% subtract
differencetzfy = tzpoldiscr-fypoldiscrarm;

figure(11);

plot(throttle, differencetzfy);

axis([0 100 -inf 0]);

title('Difference between Tz and Fy*arm (24cm)');
xlabel('throttle [%]');
```

```
ylabel('Torque [Nm]');

set(gca,'FontSize',18);

set(gcf, 'Position', get(0,'Screensize'));
saveas(gcf,strcat(figStr,'diffTzFyarm.fig'));
saveas(gcf,strcat(figStr,'diffTzFyarm'),'png');


%%
%plotting the found results of Fz to a curve fit of manufc data

%manufac data polyfit
fzpolymanufdata = @(x)-0.0006033*x.^2 + 0.1943*x  - 4.077;

%discritize
fzpoldiscr = Fzpoly(throttle);
fzpolymanufdatadiscr = fzpolymanufdata(throttle);

%plot

figure(12);

plot(throttle, fzpoldiscr);
hold on;
plot(throttle, fzpolymanufdatadiscr);

axis([50 100 0 inf]);

title('Found Fz values vs manufacturer data');
xlabel('throttle [%]');
ylabel('Force [N]');

set(gca,'FontSize',18);

set(gcf, 'Position', get(0,'Screensize'));
saveas(gcf,strcat(figStr,'ManufacFzvsMine.fig'));
saveas(gcf,strcat(figStr,'ManufacFzvsMine'),'png');
```

Appendix E: Tilting measurements matlab script

```
time = RDTSequence*0.02;

figure(1);

plot(time, Fx);
hold on;
plot(time, Fy);
hold on;
plot(time, Fz);
hold on;
plot(time, Tx);
hold on;
plot(time, Ty);
hold on;
plot(time, Tz);
hold off;


title('Forces and torques with different servo rotation speeds');
xlabel('time');
ylabel('Force [N]/Torque [Nm]');

legend('Fx','Fy','Fz','Tx','Ty','Tz');

set(gca,'FontSize',18);

%% plot Fx/Fy versus Fz Fx/Fy versus Fz
figure(2);

% Calculate the percentages asociated with the discritized data
%fyfzpercentage = -Fy./(Fz./100);
%fxfzpercentage = -Fx./(Fz./100);


%plot(time,fyfzpercentage);
%hold on;
%plot(time,fxfzpercentage);
%hold off;

%title('Fx,Fy vs Fz');
%xlabel('throttle [%]');
%ylabel('Fx/Fz | Fy/Fz [%]');
%legend('Fy/Fz','Fx/Fz');
%axis([0 inf 0 100]);
```

Appendix F: Analysis tilting measurements

As discussed before, it is a relatively hard task to map or fit a function to the forces and torques without angle position feedback. Thus is is chosen to analyze the data when the thrust vector is pointing in the Z direction, where the BLDC motor has the highest rotational velocity.

Now looking at peaks of Fz at the rising edge of Fy, we pick 4 points at the fastest rotational speed, and four points at the slowest rotational speed and average.
Averaging and comparing the results of this process gives the following data:

Forces and torques	*Dataset 1*	*Dataset 2*	*Dataset 3*	*Dataset 4*	*Average*
Fx [N]	*-0.6372*	*-0.9527*	*-1.078*	*-0.9265*	*-0.9*
Fy [N]	*-0.06737*	*-0.2392*	*-0.183*	*-0.4662*	*-0.24*
Fz [N]	*7.698*	*7.761*	*7.581*	*7.464*	*7.63*
Tx [Nm]	*-0.02061*	*0.01849*	*-0.0141*	*0.033357*	*0*
Ty [Nm]	*-2.19*	*-2.251*	*-2.151*	*-2.098*	*-2.17*
Tz [Nm]	*-0.006695*	*-0.02556*	*-0.03615*	*-0.1114*	*-0.04*

Table 12: Slowest rotational speed (0.1hz)

Forces and torques	*Dataset 1*	*Dataset 2*	*Dataset 3*	*Dataset 4*	*Average*
Fx [N]	*-0.06306*	*-0.04049*	*-1.544*	*-1.545*	*-0.8*
Fy [N]	*0.0042*	*0.06362*	*-0.2302*	*-0.04895*	*-0.05*
Fz [N]	*7.575*	*7.685*	*7.598*	*7.672*	*7.63*
Tx [Nm]	*-0.01697*	*-0.01392*	*-0.004424*	*-0.01655*	*-0.01*
Ty [Nm]	*-2.377*	*-2.394*	*-2.356*	*-2.374*	*-2.38*
Tz [Nm]	*-0.01982*	*0.01721*	*-0.04541*	*0.05977*	*0*

Table 13: Fastest rotational speed (1.2hz)

And now comparing the averages and calculating the increase yields:

Forces and torques	*Slowest rotational speed*	*Fastest rotational speed*	*Absolute difference*	*Percentage increase*
Fx [N]	*-0.9*	*-0.8*	*-0.1*	*-11.11*
Fy [N]	*-0.24*	*-0.05*	*-0.19*	*-79.17*
Fz [N]	*7.63*	*7.63*	*0*	*0*
Tx [Nm]	*0.004*	*-0.01*	*0.01*	*-350*
Ty [Nm]	*-2.17*	*-2.38*	*0.21*	*9.68*
Tz [Nm]	*-0.04*	*0.003*	*-0.04*	*-107.5*

Table 14: Comparison of force and torque increases

As for interpretation of these results, the absolute values of Tz is too small to call significant. When looking at the graphs, it is clear that the noise has a larger variance than this. It makes it impossible to assess the phenomena of drag torque, however, the point of the research is to find out whether this force is significant or negligible and not how large these torques and forces might be.

It can be clearly visually observed in the graph that there is almost no change in Tx in the slowest rotational speed, whilst there is clearly a torque present in the fastest rotational speed.

This component is here most likely because of the previously mentioned rotational inertia and gyroscopic effect that needs to be countered.

Also something to note is the large amount of noise of Fx, deviating about 1N from the average trend.

Looking at Fy, it is on the rising edge of this force. This means the propeller is turning from the negative Y axis to the positive Y axis. Because this is the steepest part of the ‘function’ of Fy, a small variation here in time can mean a large deviation in Fy found. This is especially the case in the faster rotational velocity due to the limited sample rate used.

The values found for Ty do seem to be fairly consistent. This will mean that there is an increase of about 10% in torque in the Y axis.
This can also be seen if we map the force in the Z axis to the torque in the Y axis. The arm length with attached titling mechanism is around 30cm. Thus:

$$T_{y,mapped} = l F_z = 0.3 F_z \quad \textit{eq. 12}$$

	Fz	*Ty (mapped from Fz)*	*Ty*	*Difference Ty and Tymapped*
Slow, dataset 1	*7.698*	*2.3094*	*2.19*	*0.1194*
Slow, dataset 2	*7.761*	*2.3283*	*2.251*	*0.0773*
Slow, dataset 3	*7.581*	*2.2743*	*2.151*	*0.1233*
Slow, dataset 4	*7.464*	*2.2392*	*2.098*	*0.1412*
Fast, dataset 1	*7.575*	*2.2725*	*2.377*	*-0.1*
Fast, dataset 2	*7.685*	*2.3055*	*2.394*	*-0.0885*
Fast, dataset 3	*7.598*	*2.2794*	*2.356*	*-0.0766*
Fast, dataset 4	*7.672*	*2.3016*	*2.374*	*-0.0724*

Table 15: Comparing mapped Tz versus found Tz for fast and slow tilt

Note with this table, that the found Ty is slightly less than the expected Ty when mapped from Fz. This is probably because of the non exact arm length. However, this is not of importance for the result. What is to be noted is that the torque in the Y axis is for the fast tilting measurements larger than the expected torque when mapping the force in Z axis.

The increase in the torque in the Y axis is most probably due to the inertia and gyroscopic effect of the rotor. The centre of thrust is not exactly in the middle of the axis of rotation, and thus there will be a rotating inertia. At the point where Fz is pointing upwards, the Z component of this inertia needs to change direction. The added inertia to the already existent thrust in the Z direction is what probably causes the increase in torque in the Y axis.

6 References

[1] *MIT - Theory of Flight.* (16/03/1997) Retrieved 20/06/2017, from http://web.mit.edu/16.00/www/aec/flight.html
[2] *NASA – The Lift Equation.* (05/05/2015) Retrieved 20/06/2017, from https://www.grc.nasa.gov/www/k-12/airplane/lifteq.html
[3] *Global Security – HELICOPTER FUNDAMENTALS.* (N/A) Retrieved 20/06/2017, from http://www.globalsecurity.org/military/library/policy/army/accp/al0966/le3.htm
[4] *UT-Q650 QUADCOPTER ASSEMBLY GUIDE.* (28/7/2017) Retrieved 20/06/2017, from http://www.unmannedtech.co.uk/manuals/ut-q650-assembly-guide
[5] Bangura M., Melega M., Naldi R., Mahony R. 2016. *Aerodynamics of Rotor Blades for Quadrotors*
[6] Sydney N., Smyth B., Paley D.A. N/A (est. 2013). *Dynamic Control of Autonomous Quadrotor Flight in an Estimated Wind Field*
[7] *Lift-induced drag.* (3/6/2017) Retrieved 20/06/2017, from *https://en.wikipedia.org/wiki/Lift-induced_drag*
[8] *Brushed vs Brushless motors.* (N/A) Retrieved 20/06/2017, from *http://www.thinkrc.com/faq/brushless-motors.php*
[9] *Brushless DC electric motor.* (14/6/2017) Retrieved 20/06/2017, from *https://en.wikipedia.org/wiki/Brushless_DC_electric_motor*
[10] *Servo Motor Control: The Servo Motor.* (N/A) Retrieved 20/06/2017, from *http://www.pyroelectro.com/tutorials/servo_motor/servomotor.html*
[11] Ryll M., Bülthoff H. H., Giordano R. P. *N/A (30/07/2014). A Novel Over-actuated Quadrotor Unmanned Aerial Vehicle: Modelling, Control and Experimental Validation*
[12] *Oosedo A., Abiko S., Narasaki S., Kuno A., Konno A., Uchiyama M. N/A (26/5/2015). Large attitude change flight of a quad tilt rotor unmanned aerial vehicle*
[13] *Skyborntech Technologies – UAV development.* (N/A) Retrieved 21/06/2017, from *http://skybornetech.com/rnd/*
[14] *Nemati A., Soni N., Sarim M., Kumar M. N/A (12/10/2016). Design, Fabrication and Control of a Tilt Rotor Quadcopter*
[15] Tarot RC*: Tarot Iron man FY650.* (25/02/2013) Retrieved 22/06/2017, from *http://www.tarot-rc.com/index.php?main_page=product_info&cPath=65_66&products_id=197*
[16] *ATI Industrial Automation Mini40.* (N/A) Retrieved 22/06/2017, from *http://www.ati-ia.com/products/ft/ft_models.aspx?id=Mini40*
[17] *T-Motor MT2212 KV750 BLDC motor.* (N/A) Retrieved 28/06/2017, from https://www.aerolab.de/t-motor-mt2212-kv750/brushless-motoren/t-motor-mt-serie/a-101003/#technicaldata
[18] *Andrew Gibiansky. N/A (23/11/2012). Quadcopter Dynamics, Simulation, and Control (Retrieved 27/06/2017 from http://andrew.gibiansky.com/downloads/pdf/Quadcopter%20Dynamics,%20Simulation,%20and%20Control.pdf)*
[19] Bruno Siciliano and Oussama Khatib. *Springer Handbook of Robotics – 26.5 Rotorcraft modeling and design. 2nd ed. Verlag Berlin Heidelberg*: Springer, 2016.
[20] *Mathworks - Matlab* (N/A) Retrieved 28/06/2017, from *https://nl.mathworks.com/products/matlab.html*
[21] *Mbed online compiler* (N/A) Retrieved 28/06/2017, from https://www.mbed.com/en/
[22] Wikibooks - *Control systems, second order systems* (N/A) Retrieved 28/-06/2017, from https://en.wikibooks.org/wiki/Control_Systems/Examples/Second_Order_Systems
[23] *Park S., Her J., Kim J., Lee D. N/A (14/9/2016). Design, Modeling and Control of Omni-Directional Aerial Robot*