Politecnico di Torino

Corso di Laurea Magistrale in Ingegneria Aerospaziale

A.A. 2022/2023

Sessione di Laurea Dicembre 2023

Data driven Mathematical Modelling and Analysis of a VTOL UAV system

Relatori:

Prof. Giorgio Guglieri

Ing. Daniele Camatti

Candidato:

Giacomo Cavallero 280271

ABSTRACT

Thanks to new technologies in terms of design of software and hardware, Unmanned Aerial Vehicles (UAVs) or drones are spreading all over the world. Drones are effective and compatible to many different fields such as agriculture, military operations, surveillance, scientific studies, cinema and many more. Due to poor Multicopters' flight efficiency, companies and purchasers are still sceptical of their endurance. Vertical Take Off and Landing (*VTOL*) drones have been introduced as a solution for both endurance, flight distance and restricted landing and take-off area. These drones are capable of extreme flight conditions but only if correctly controlled. Many companies and hobbyist around the world use standardized components compatible with much software. Open-source software often provide basic flight controls and sluggish drones' dynamic responses. These control laws need to cover a whole variety of different configurations and are not specific for each drone characteristics. ArduPilot is an open-source extremely versatile software that offers many different drone capabilities. Despite being robust, it provides simple and limited flight control which makes many flights phases energy expensive and not optimized.

Control laws must be created to optimize some of these manoeuvres and minimize the time spent, energy loss. The main drive must always be to increase safety. Future control laws must also cooperate with the main flight control system inside the autopilot to improve its overall performances.

In a brief description of the current State of Art of the drones' *VTOL* configurations, the differences between the available configurations are highlighted.

In this thesis, two drones of two different configurations, Tailsitter and Bellysitter, are introduced and the first one is further analysed. As no data of these drones was available beforehand, the most fundamental quantities such as mass, inertia, centre of gravity and neutral point were calculated.

A flight campaign has been performed and many different flight conditions have been studied. The data was generated from on-board sensors and later collected inside Flight Logs. These Logs were then postprocessed in MATLAB. Through the usage of Flight Logs Data, Bench Tests and many other tools, the aerodynamic coefficients were estimated.

In order to study the many possible control algorithms, a six Degrees of Freedom Simulator is developed in MATLAB environment.

After many simulations and tests, the landing algorithm has been written both in Lua programming language and in C++, to be inserted directly into the firmware to cooperate with the original in these difficult manoeuvres. Tests in lab have been performed to check the consistency of the results obtained and the robustness of the algorithms.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

This work of Thesis has the objective of modelling and controlling two different Vertical Take of Landing Drones during all their flight phases. The drones belong to Pros3 Company, have 4.5 kg and 3.5 kg masses and 1.2 m wingspan. To effectively create control laws for such a complex system, simulations need to be performed. Real tests not backed up by simulations can be dangerous, unpredictable, and expensive for the company since the components can be expensive and easy to damage. To create a simulator, a lot of data regarding the drone's dynamics needs to be obtained.

The work started more than one year ago, and no previous data of the drones was provided but a 3D model on SolidWorks. Therefore, everything had to be calculated. The first step was to obtain the most fundamental information such as mass, surfaces dimensions and distances between peculiar points. Subsequently the centre of gravity was calculated as well as a rough estimate of the neutral point. Using the 3D model, the inertia matrix was obtained. After checking these data through simple yet important tests, the focus shifted onto obtaining the aerodynamic coefficients.

Starting from the airfoil while gathering all the available data. Unfortunately, the coefficients found couldn't successfully predict the whole drone dynamics. Therefore, a more precise procedure needed to be implemented.

The other method that was found was to take advantage of the possibility of flight tests.

After finding the correct equations of motion, many specific flight tests were performed to obtain the data needed. For example, to calculate the coefficient of lift at the different angle of attack angles, cruise flights were performed at different speeds.

After the flight tests were finished, the data was post processed. The procedure was found to converge at plausible and acceptable values only in specific flight conditions as no "clean" data could be obtained in extreme, yet important, flight conditions. For this reason, the coefficient curves had to be filled with data from other sources such as Datcom, AVL or CFD. A six degrees of freedom simulator in Simulink was developed. The first flight phase that wanted to be studied and improved was the final one as multiple landings in the past ended crashing the drone's blades into the ground.

A PID algorithm was created in Simulink to control the landing phase. After many simulations and checks, the same algorithm was rewritten in Lua, a C-based programming language capable of being inserted into the Autopilot's firmware.

Although being tested, the algorithm needed to be tuned so the drone was held on the ground by two hinges. Tests to tune the PID were performed.

The simulator is also ready to model the transition between *Copter* and *Plane* mode. This transition was extremely energy expensive as the drone was gaining a lot of altitude because of it, and the subsequent descent was done in *Copter* Mode which consumed a lot more energy than the Plane Mode.

Future control laws must minimize both altitude gain and time while control the X and Y position. To perform a precise automatic landing, the *Copter* phase must be as short as possible because of its high energy costs.

During the last year I had the possibility of interacting with the drones directly as well as test all its systems and understand advantages and disadvantages from both an engineer and a pilot point of view. I was able to work as a Ground Controller to check and change the drone's behaviour in real time, work as a mechanic to fix it every time something broke, work on its electrical system, to 3D-print parts, analyse logs and many other aspects of a typical engineer work in this field.

2 Overview on the current drones' state of art

This chapter tries to briefly analyze the current drones' state of art.

The drone industry has undergone significant evolution in recent years, revolutionizing the technological landscape and opening new opportunities in a wide range of sectors. Unmanned Aerial Vehicles (UAVs), or drones, represent a technological innovation that has unlocked new possibilities in multiple fields.
Given the immense number of existing drone configurations, only electric powered drones will be considered. The analysis first describes the two main configurations: rotary and fixed wing and later will compare their advantages and disadvantages showing how one of the most suited industry sectors for the hybrid *VTOL*s' configuration is the delivery sector. The analysis will give some examples on these different configurations and compare the main configuration used in delivery called quadplane, against *VTOL*s.

2.1 21st Century Drones' Market Growth

Throughout the forecast window from 2023 to 2030, the Global Drone market will reasonably see a substantial growth. The market demonstrated steady expansion in 2022, and the anticipated implementation of strategies by major industry players is projected to drive further growth across the forecasted period. [1]
According to the analysis of the Commercial UAV News of June 2023, in 2022, the global Drone market reached a valuation of USD 276 Million. Forecasts suggest a substantial increase, projecting the market to escalate to over 800 USD Million by 2028, showcasing a Compound Annual Growth Rate (CAGR) of 20.02% during the period from 2022 to 2028.
The annual survey conducted by Drone Industry Insights has engaged over 1,100 companies globally.
The survey findings reveal that startups and small to medium-sized enterprises largely dominate the drone industry, representing nearly 70% of the respondents. Nevertheless, it's important to note that larger markets often support the growth of well-established companies. For example, in China, more than 57% of companies have workforces surpassing 50 employees. Despite the prevalence of startups, a notable percentage (5%+) of companies have staff sizes exceeding 5,000 individuals.

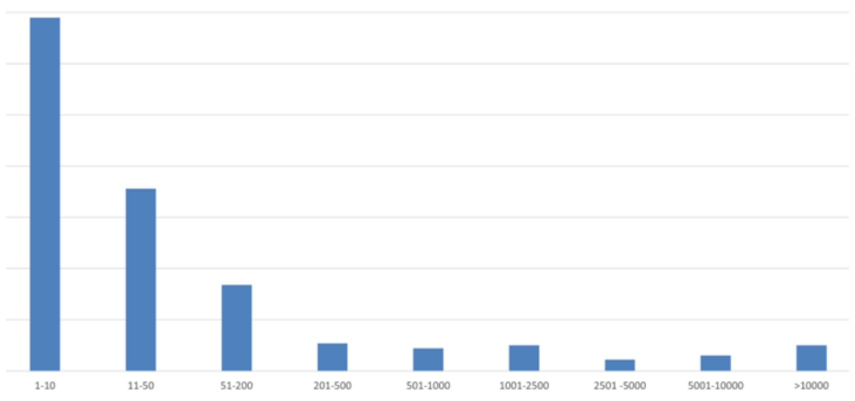


Figure 1: Number of employees in the companies in the Drone Market [1]

A focus on the delivery drone market

It is widely acknowledged that the drone delivery industry represents a continuously expanding market. According to Mahan et al. in the article "*Drone delivery: More lift than you think*", during the year 2020, 2021 and 2022 more than 660,000 commercial drone deliveries have taken place, not counting the numerous test flights that have refined and demonstrated this technology. By the beginning of 2022, approximately 2,000 drone deliveries take place every day on a global scale. This number is rapidly increasing, with forecasts indicating nearly 1.5 million deliveries expected for 2022, a significant rise from the under half a million recorded in 2021. [2] Presently, drone deliveries encompass a wide array of products, from essential medical supplies such as vaccines and blood transfusions to everyday commodities like pizza, burgers, electronics, and various others. Major industry leaders like Antwork, Flytrex, Manna, among others, have collectively garnered more than $1 billion in disclosed funding over the past decade, substantially propelling the industry's growth. [2]

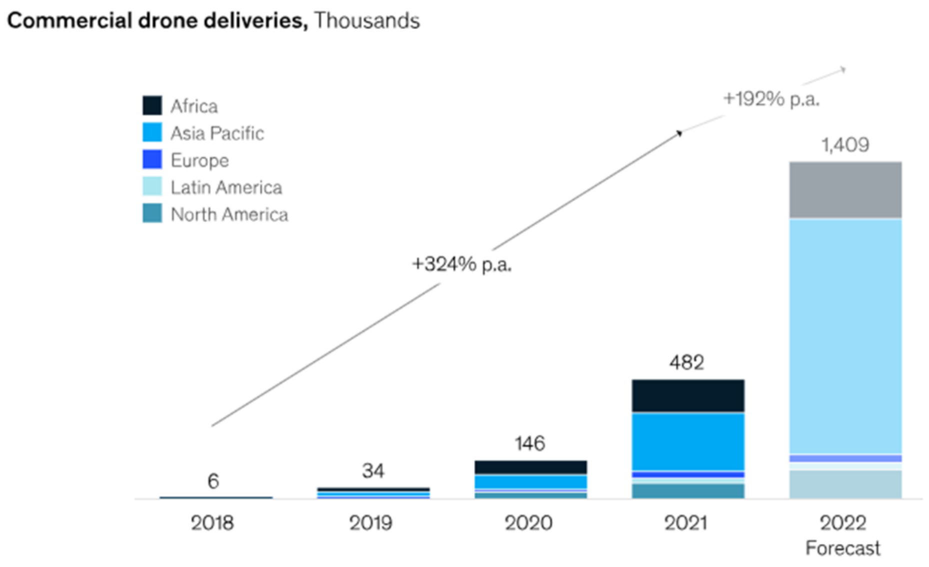


Figure 2: Growth of the number of deliveries from 2018 to 2022 [2]

Looking at the analysis of Mahan et al the future direction of direction of drone delivery relies on three crucial catalysts, that will be described in the following.

- **Regulation:** The extent and functioning of drone delivery rely on regulatory frameworks that govern allowable geographic zones, airspace usage, flight schedules, and operational parameters. These directives exert a substantial influence on associated costs.
- **Public acceptance:** The scope and functionality of drone delivery are contingent upon regulatory frameworks that oversee authorized geographical areas, airspace utilization, flight timetables, and operational criteria. These guidelines significantly impact the associated expenses.
- **Cost:** Given equal circumstances, consumers generally favour delivery options with lower costs. Nevertheless, evolving innovations such as electric and autonomous vehicles along with ground-based robots consistently drive down their expenses as they advance in development.

Coming to conclusions, it is important to emphasize that we are at a pivotal moment for the drone market. Despite the significant increase in volumes, the industry's future strongly depends on the three factors analysed above: the evolution of regulations, the increasing of public acceptance, and the alignment of cost considerations. These factors will determine whether drone deliveries will fulfil their potential to revolutionize global logistics or remain confined to specific applications.

The main companies in the delivery sector

The table below outlines key aspects of leading companies in the delivery sector employing drone technology, namely Zipline, Amazon Prime Air, Wing, DHL, UPS, Flytrex and Volansi. The factors analyzed for each company are:

- *Specs*, the table displays, in sequence, weight, cruise speed and operative range of the drones.
- *Type of delivery.*
- *Region,* the area where the company operates.

Table 1: Analysis of the main actors in the delivery sector

Company	zipline	amazon Prime Air	Wing	DHL	ups	FLYTREX	VOLANSI
Drone type	QUADPLANE	Exacopters	Dodecacopter (+ 2 pushing motors)	Quadcopter (pushpull motors	Quadcopter	Multirotor	QUADPLANE FLY/380
Specs	2-2.5kg 27 m/s 120 miles	25 kg 22 m/s 10 miles radius	5.2 kg 29 m/s 20 km	14kg 19 m/s	N/A 20 m/s 20 km	8kg 30m/s 20km	50kg 34 m/s N/A
Type of delivery	The payload descends with a rope while drone is hovering. Medications Blood (1-3 kg)	Anything less tha 2.5 kg (restrained sizes)	Food And Beverages Delivery	Anything less than 2kg (restrained size)	Payload up to 5 kg	Food And Beverages Delivery	2h/15kg payload 4h/10kg payload
Region	Rwanda USA Nigeria Cote d'Ivoire Kenya Ghana Japan	Stated in Oregon USA U.S. (Italy, UK 2025)	USA Australia Finland Ireland	Closed recently	USA	Based in USA Deliver in USA	USA

2.2 Fixed-wing Drones, Rotary-wing Drones and Fixed-wing *VTOL*: Current State of Art

There are several different drone types which are currently being studied, developed and used in industries. These different types can be divided into two main groups: Fixed-wing and rotary-wing drones. Fixed-wing and rotary-wing drones represent two distinct categories of Unmanned Aerial Vehicles (UAVs) characterized by distinct attributes and applications. Each type presents notable advantages, yet also limitations that make them more or less suitable for particular assignments. In the following paragraphs it will be provided a brief analysis of both drones' typologies.

Rotary-wing drones

Rotary-wing drones are among the most common and versatile configurations. These drones utilize rotating propellers to generate lift and move through the air. Their maneuverability and stability make them ideal for a wide range of applications.

Advantages

Looking at the advantages of the rotary-wing drones it is essential to underline that these aerial vehicles are recognized for their exceptional maneuverability and the capacity to maintain a stationary position. These attributes make them particularly suited for specialized tasks such as aerial photography, making these drones essential in the film industry and professional photography. Notably, models like the DJI Phantom 4 Pro V2.0 and Parrot Anafi USA are extensively utilized within these domains owing to their ability to capture stable and versatile imagery from varying perspectives. The DJI Phantom 4 Pro V2.0 is equipped with a 20-megapixel camera with a 1-inch CMOS sensor and 4K video at 60 fps and delivers exceptional performance for high-quality aerial shots.

Figure 3: DJI Phantom 4 Pro V2.0

Beyond their expertise in aerial imaging, rotary-wing drones serve vital roles in surveillance and security applications. Agencies involved in public safety and law enforcement harness these drones to oversee emergency situations, conduct area patrols, and execute search and rescue missions. The drones' hovering capabilities facilitate comprehensive monitoring of specific areas and expedite responses to critical incidents.

Moreover, another important field of application of rotary-wing drones is precision agriculture. Within precision agriculture, rotary-wing drones have revolutionized crop management methodologies. Their functionalities encompass field mapping, crop health monitoring, and precise delivery of fertilizers or pesticides. This enhances operational efficiency, minimizes resource usage, and fosters sustainable agricultural practices.

Furthermore, rotary-wing drones play a pivotal role in scientific research endeavors. Their adeptness in data collection within challenging environments and environmental surveillance in remote or inhospitable areas makes them invaluable assets in scientific pursuits due to their adaptability and access to challenging locales.

Disadvantages

Rotary-wing drones also present certain drawbacks.

Their autonomy is frequently constrained, resulting in shorter flight durations in comparison to fixed-wing drones. This limitation makes them less ideal for tasks necessitating coverage of extensive geographic areas or long-range flights. Additionally, the maintenance of rotating propellers may demand more meticulous attention compared to fixed-wing components.

Fixed-wing drones

Fixed-wing drones generate the lift needed for flight through their wings and are designed for long-range flights.

Advantages

Fixed-wing drones are recognized for their remarkable autonomy and capability to cover extended distances. Their prolonged flight capacity makes them particularly well-suited for missions requiring long aerial exploration or the surveillance of expansive geographic regions. Another notable strength of fixed-wing drones lies in their capacity for detailed data collection. These drones are proficient in remote sensing missions such as topographic mapping and environmental surveillance. Their ability to operate at higher altitudes compared to rotary-wing drones allows them to gather intricate data across extensive territories, proving invaluable in fields like cartography and geology.

Additionally, fixed-wing drones demonstrate suitability for the aerial delivery of lightweight goods. Companies like DHL are testing the use of fixed-wing drones for transporting medicines and medical supplies to remote regions. This application underscores the potential of fixed-wing drones in enhancing the logistics of medical supplies within hard-to-access areas.

Disadvantages

Analyzing the disadvantages of fixed-wing drones it is important to underline that fixed-wing drones demonstrate lesser maneuverability in comparison to rotary-wing drones, making them less suitable for tasks demanding stationary flights or complex maneuvers. Furthermore, their flight configuration necessitates adequate space for both takeoff and landing, which poses constraints in urban environments or areas characterized by restricted spaces.

In conclusion, rotary-wing and fixed-wing drones present distinct advantages and drawbacks, making them applicable to diverse tasks. The choice between these types, hinges upon the precise objectives of drone utilization, flight duration, requirements for covering expansive geographical regions, and the intricacy of necessary maneuvers. Both classifications of drones persistently progress and discover novel applications, underscoring the expanding significance of UAV technology across various sectors.

In addition to rotary-wing and fixed-wing drones, there exists a hybrid configuration which combines the traits of both: *Fixed Wing VTOL*. This drone typology will be described in the following paragraph.

Fixed Wing VTOL

The *VTOL* (Vertical Takeoff and Landing) represents a unique category capable of vertical takeoff and landing, thereby offering remarkable versatility. The Fixed Wing *VTOL* integrates both multirotor and conventional wing capabilities, addressing certain challenges posed by these respective configurations. With a robust thrust-to-weight ratio, it accomplishes vertical and horizontal flights adeptly, leveraging its wing design to achieve extended distances with minimal power consumption.
In these drones, either the trailing edge of the wing or the fuselage serves partially or entirely as the landing gear, providing notable advantages. However, this implies great complexity. The transition between vertical and horizontal flight, and vice versa, poses considerable control challenges, particularly during the landing phase, which will be described later in more detail. Within this configuration, the primary subcategories are Bellysitters and Tailsitters, denoting drones that land on their lower fuselage or the tail/trailing edge of the wing, respectively.
Both configurations can incorporate vectored thrust technology. Vectored-thrust means that the propeller can adjust its direction thanks to a rotary motor servo linked to the motor. Adjusting the angle of the propellers offers agility and versatility, facilitating performing complex maneuvers at higher acceleration thanks to the high control power due to the nature of this control.

Bellysitter VTOL drones

Bellysitter *VTOL* drones are designed to enable takeoff using multiple approaches:

a) Propelled by hand (the pilot must physically throw the drone into the air)
b) Landing Gear with wheels or some kind sliding cushion
c) Tailsitter-like takeoff

The SenseFly eBee X is a Bellysitter *VTOL* drone, used in precision agriculture for field mapping and crop monitoring. It offers considerable endurance, high-resolution maps, and efficient coverage of extensive areas.

Figure 4: SenseFly eBee X

SenseFly eBee X vs Strategy

The main differences between SenseFly eBee X and the drone under analysis in this thesis (which will be introduced at the end of this chapter), Strategy, are:

- The thrust of the eBee is non vectored and the landing sequence is both much easier and risker.
- eBee requires a greater landing distance compared to Strategy and needs a flat landing surface with no obstacles, which might not be available everywhere.
- The other big difference is that eBee needs to be thrown to take off.

Tailsitters VTOL drones

Tailsitters drones land and take off on their tail, the drone's body remains vertical during descent and ascent. This configuration offers ease of takeoff and landing at a higher aerodynamic resistance due to often-used booms to stabilize the landing.
An example of Tailsitter, beside Strategy is the Quantix *VTOL*. This drone is specifically designed for precision agriculture, allowing farmers to monitor crops and collect valuable data for production optimization.

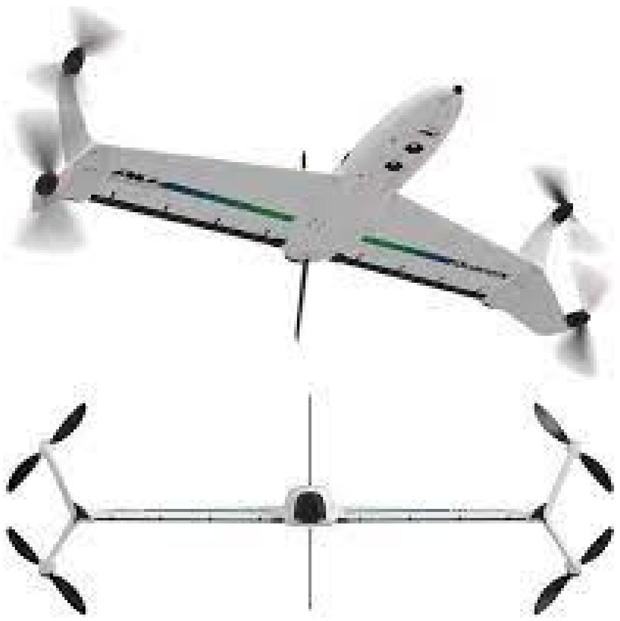


Figure 5: Tailsitters VTOL drone

2.3 Rotary wing vs Fixed Wing vs *VTOL*

This section summarizes the advantages and disadvantages of these three configurations.

The specifications which have been selected are:

1) *Power Consumption*. It refers to the overall power needed during levelled flight. It is not only important for the endurance but also due to its impact on the internal components. Greater power demands result in higher currents, necessitating larger and more expansive components.
2) *Speed.* It refers to both the range of flight speeds reachable by the aircraft and the maximum speed.
3) *Distance*. It is the maximum distance from the landing point.
4) *Hovering Capability.* Arguably the most important aspect of this comparison as in the delivery sector is mandatory.
5) *Runway need.* Another important aspect which might differ from the previous specifications in many configurations.
6) *Complexity.* This detail represents the most important factor from a company perspective.

Table 2: Comparison between Multirotor, Fixed Wing and Fixed Wing VTOL

CONFIGURATION	**Multirotor**	**Fixed Wing**	**Fixed Wing VTOL**
Power Consumption (levelled flight)	High	Low	Low
Speed	Narrow Range	Wide Range	Wide Range
Distance	Short	Long	Long
Hovering Capability	Yes	No	Yes
Runway need	No	Yes	No
Complexity	Low	Low	High

Please note that the column 'Fixed Wing *VTOL*' refers only to Bellysitters and Tailsitters *VTOL*.

The main competitor of Fixed Wing *VTOL*s are Quadplanes, that combines Multirotor and Fixed Wing aspects. Quadplanes are also considered to be *VTOL*s as they can easily perform both vertical and horizontal takeoff while flying as a fixed wing plane. The difference between Fixed Wing *VTOL*s and Quadplanes is that the latter are less complex but also less advantageous in terms of power consumption, size and cost.

Quadplanes utilize four vertical mounted motors to lift off and land. When the drone has reached the desired height, the main propeller, mounted horizontally, will start and once it has gained sufficient speed to generate the necessary lift, the vertical mounted propeller will cease to work. The same process, but reversed, happens when transitioning between horizontal and vertical flight.Quadplanes offer great intrinsic stability in both vertical and horizontal flight which makes them the greatest competitor to Fixed Wing *VTOL*'s.

The following table summarizes the main differences between Fixed Wing *VTOL* and Quaplanes.

Table 3: Comparison between Fixed Wing VTOL and Quadplane

CONFIGURATION	Fixed Wing VTOL	Quadplane
Aerodynamic Efficiency	High	Low
Production Cost	Medium	High
Size	Medium	Large
Payload/MTOW	Low	High
Complexity	High	Medium-Low

Again, the main drawbacks are the higher complexity and the lower ratio between payload and MTOW. This second drawback has not been fully described as it mainly depends on the thrust to weight ratio and therefore on the choice of the motor, which may impact on the other aspects both positively and negatively. For example, a bigger propeller may increase the thrust to power ratio which may lower the power consumption during flight, on the other hand a smaller propeller would decrease the weight during *Copter* mode flight. The following image has the objective of showing the high potential of multi frame firmware like ArduPilot. The image shows only the *Copter* -like frame, many others can be implemented.

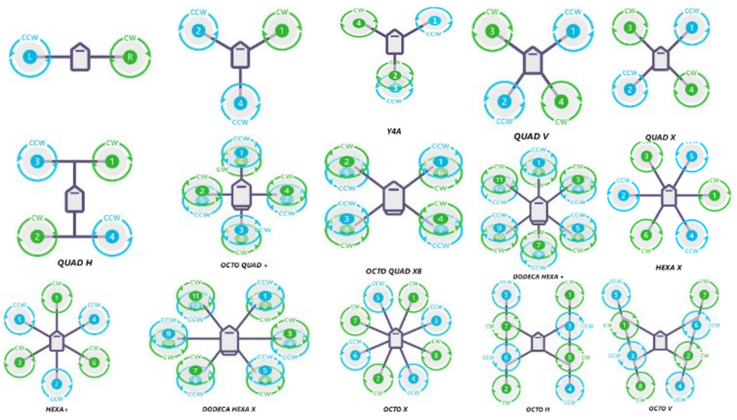


Figure 6: ArduPilot Supported Copter frames

3 DYNAMICS OF *VTOLS*

The drones that were analyzed and tested during this work are Strategy and Vanity.
These two drones are all-wing type and have only one controllable surface: the Elevons. This controllable surface works as both Aileron and Elevator, respectively reaching opposite and equal angles.
The following chapter will focus on Strategy and Vanity.

3.1 STRATEGY AND VANITY

As stated before, the drones studied in this thesis are Strategy and Vanity.
Although both being *VTOL*s, the first one is a Bellysitter type and the second one is a simpler Tailsitter. Strategy is the "big brother" of Vanity since it weighs more, has a bigger wing surface and a bigger wingspan. The biggest difference between the two drones are Strategy's tilting motors.
This extra option also brings more complexity to the system but increases the overall operational capabilities of the drone. As previously explained, Bellysitters land on the fuselage as for Tailsitters on the stern. The landing phase of Bellysitter includes the other type's landing but reaches a much more stable position.
Also, Tailsitter often needs to enlarge the back of the fuselage or wings to have a bigger surface to land on. Vanity adopts another solution, which consists in a thin pole attached to the vertical empennage. This not only decreases flight performance as it increases the overall mass, inertia along the X axis and aerodynamic drag, but also worsen Vanity's design which can be very important for a company's market. For this reason, a Strategy is somewhat the evolution but from the start many concerns were raised because of the Tilting Motors. The second most important advantage in having Tiltable motors is the possibility of maintaining attitude conditions even in the presence of wind or other disturbances.

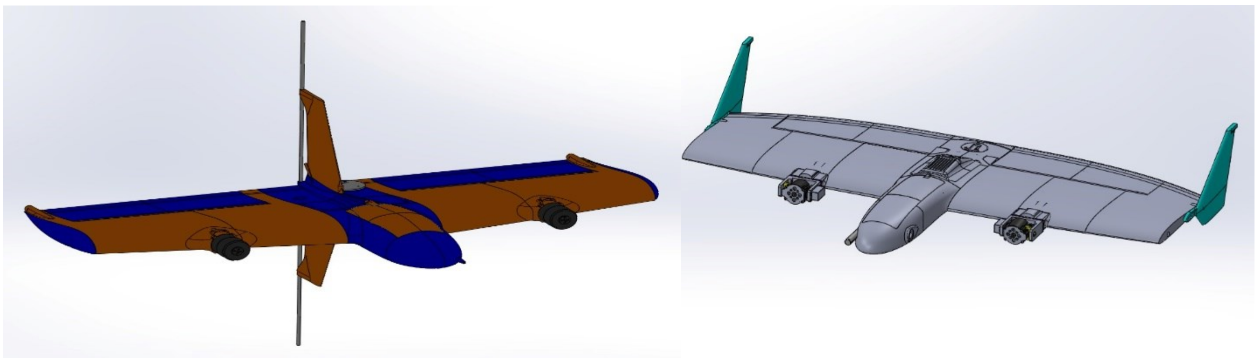


Figure 7: Vanity (on the left) and Strategy (on the right)

3.1.1 *Plane* Mode

In *Plane* mode, the drone follows the common commands of typical aircraft.

Pitching motion is created with the Elevon moving as they were elevator surfaces. Therefore, they move creating the same angle with respect to the chord.

The PWM is inverted for the right Elevon as the two Servos must rotate in opposite directions to create the same Elevon angle (so that if the left side reaches 1800 PWM, the right side is at 1200 PWM). This issue can be easily solved thanks to an ArduPilot parameter, addressable through Mission Planner.

In *Plane* mode, the roll is given by differential Angle at the Elevon, as they were moving as ailerons. Since the Servos are already rotated 180°, the PWM needed to reach the same angle (positive on one side and negative on the other) is the same.

The Yaw in *Plane* mode is provided by the differential Thrust as no Vertical Empennage is present. Therefore, each propeller rotational speed can be addressed individually.

Lift and drag are generated from both wings and fuselage although the latter's contribution being a lot smaller.

The later aerodynamic force is generated from both the winglets and the fuselage and cannot be controlled as there are no movable surfaces along the Z Axis.

3.1.2 *Copter* Mode

In *Copter* mode the Roll is given by differential thrust, yaw from the Elevons moving as ailerons and Pitch from the Elevons moving as Elevators, as before.

In *Copter* mode the Motor Tilts can be used to create different movements such as Pitch and Roll Movements. The Tilts Servos can be addressed individually and therefore the Thrust generated by each propeller can be directed in any direction between -90° to + 90° (with respect to the X_{Body}).

In Plane mode the Motor Tilts aren't used as they are not needed. The necessary Thrust to keep the aircraft in a levelled flight is along the X_{Body} Axis and it's less the half with respect to the Thrust needed in *Copter*.

3.1.3 Flight controls

The vector of commands with their respective limitation can already be outlined.

They will be later analyzed in detail:

3.1

$$\vec{u} = \begin{Bmatrix} -\pi/4 \\ -\pi/4 \\ -\pi/2 \\ -\pi/2 \\ 0 \\ 0 \end{Bmatrix} \leq \begin{Bmatrix} \delta_{Elevon_{Right}} \\ \delta_{Elevon_{Left}} \\ \delta_{Tilt_{Right}} \\ \delta_{Tilt_{Left}} \\ \delta_{Motor_{Right}} \\ \delta_{Motor_{Left}} \end{Bmatrix} \leq \begin{Bmatrix} \pi/4 \\ \pi/4 \\ \pi/2 \\ \pi/2 \\ 1 \\ 1 \end{Bmatrix}$$

During a levelled turn, both Aileron and Elevator powers are needed. The combination of the movements results in a linear combination of the deflections. For example, if the required aileron angle is 8° and the required elevator angle is 2°, the elevons will be moved 10° and -6° respectively.

$$\delta_{aileron} = 8° \quad \rightarrow \quad \delta_{Elevator} = 2°$$

$$\delta_{Elevon_{Left}} = 10° \quad \rightarrow \quad \delta_{Elevon_{Right}} = -6°$$

Therefore, the equations can be written:

$$\delta_{aileron} = \frac{\delta_{Elevon_{Left}} + \delta_{Elevon_{Right}}}{2}$$

$$\delta_{Elevator} = \delta_{aileron} - \delta_{Elevator_{Right}}$$

Using these two commands in different ways, can provide the pilot with all the necessary maneuver that the drone must perform. The elevons are moved by two servos which can tilt them from -45° to 45° meanwhile the motors can provide horizontal (Vanity, Strategy) and vertical Thrust (only Strategy) through the Tilts Servos.

Strategy Motors' Servos can tilt between +90° and – 90°, they are more robust than the Elevon servos although a bit slower.

3.2 Autopilot Signals

All the controls are commanded by the autopilot through the usage of PWM signals. Pulse Width Modulation (PWM) is a modulation method employed to regulate the "strength" of an analogic signal via a digital signal. In practice, a PWM signal takes the form of a square wave, where the duration of the pulse (the period during which the signal is "high") varies in direct proportion to the analogic signal's intensity. A longer pulse corresponds to a higher analogic signal intensity. This technique finds broad applications in electronics, where precise management of motors is demanded. PWM signals are used for their straightforwardness, efficiency, and accuracy when it comes to governing electrical devices.

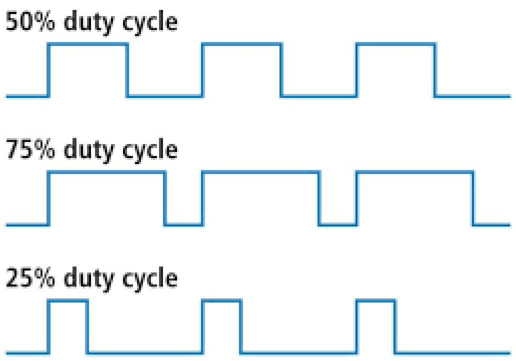


Figure 8: PMW signal

The PWM signal ranges between 1000 and 2000 (sometimes 800-2200) and at different frequencies, depending on the ESC. This means that 1000 PWM equals to 0% Duty Cycle which means a flat signal. Instead, a 2000 PWM signal corresponds to a 100% Duty Cycle.
In the image above the curves correspond respectively to 1500, 1750 and 1250 PWM.
The commands can be also written in terms of limitation for the actual servos and motors in terms of PWM:

$$\vec{u} = 1000 \leq \left\{\begin{matrix} \delta_{Elevon_{Right}} \\ \delta_{Elevon_{Left}} \\ \delta_{Tilt_{Right}} \\ \delta_{Tilt_{Left}} \\ \delta_{Motor_{Right}} \\ \delta_{Motor_{Left}} \end{matrix}\right\} \leq 2000$$

3.3 Strategy's Sensors

Multiple sensors inside the fuselage provide real time data to the autopilot at high frequencies, allowing the UAV to correctly control its own behavior.
Listed below are the sensors of Strategy:

- **GPS**: It's the main source of position and velocity. Most of the navigation checks are based on it and the autopilot prevents the drone from arming without it. The main satellite constellation is used, and data is provided within 5-10 Hz.

- **Accelerometers**: Provide instantaneous acceleration values which are then integrated twice to obtain velocity and position. They suffer from integration bias and their data needs to be filtered.
- **Gyroscopes**: Provide instantaneous rotational acceleration. They also suffer from integration bias but also gimbal lock which for this specific drone is an enormous problem.
- **Pitot Tube**: Provides instant angle of attack values. Unfortunately, it suffers from a high level of noise which can make the whole sensor useless.
- **Barometer**: Two different barometers measure the pressure to calculate the altitude with respect to the take of area.
- **Compass**: Two different compasses measure the angle with respect to the north to provide additional information of the drone’s attitude.

4 AERODYNAMIC COEFFICIENTS CALCULATION

Aerodynamic coefficients have a fundamental role in describing the drone's dynamics. Despite Strategy's low Aerodynamic efficiency, these coefficients are extremely important. In most of the emergency scenarios in which the drone might end up, the thrust created from the propellers is vital for its sustenance. By default, the software is set up so that if something is not working, the *Copter* mode is initiated and a *QRTL* (Quad Plane Return to Launch) is performed. Given the extreme attitudes in which the drone can fly, different strategies have been developed to describe the aerodynamic.

The procedure was therefore specialized for two types of configurations: *Copter* and *Plane*.

1. The *Plane* configuration is much easier to describe and test, the equations of motion describing this type of dynamics have already been described countless times and are easily linearizable.
2. The *Plane* configuration is much harder to correctly simulate. To make things worse it also incorporates different maneuvers and transients which do not follow common aerodynamic principles.

Due to the aforementioned importance of the *Copter* configuration in a *VTOL* drone, the search of its coefficients is mandatory. Fortunately, some *Copter* coefficients correspond to *Plane* so that the procedure only has to be performed once and the best flight data can be chosen.

Different options have been evaluated to create a 6 *DoF* Model of the drone. The one which has been firstly followed through is with the usage of Data provided from the logs. As this type of drone is extremely (economically) cheap to fly, flight testing is a convenient option. The aim of this model is to successfully recreate the dynamics of the drone to be able to test different flight maneuvers and control laws before performing them. The objective of the following method is to obtain the aerodynamic coefficients from the data gathered by flight campaign.

4.1 GENERAL DESCRIPTION

The first attempt consists in inverting the equation of motion which describes the aircraft. For each instant of the log, all the needed variables are taken from the log and modified in order to be consistent with one another inside the equation. After calculating all six coefficients, the data is plotted, and results are described. The first thing that needs to be done is to find the correct equations of motion. Since it's a 6 Degree of Freedom model, 3 linear acceleration and 3 rotational accelerations are calculated respectively from accelerometers and gyroscopes inside the autopilot.

The equations of motion must describe both the *Plane* and *Copter* dynamics including: aerodynamics, thrust characteristics, movable surfaces and if needed sensors simulation.
At first a non-linear model is created, afterwards a linearized model in a forward nominal flight condition is created as it becomes very convenient for different control laws. It's beyond the extent of this thesis to find the correct linearization of the *Copter* dynamics.

4.2 COMPARISON BETWEEN THE CHOSEN PROCEDURE AND ALTERNATIVES

To calculate the aerodynamic coefficients, different types of procedures have been analyzed and followed. This section will briefly describe each one of them.

4.2.1 LOG

ArduPilot's logs have the advantage of having more than 500 variables logged simultaneously, which can be found in the ArduPilot website. [3] [4]
Therefore, the complete status of the vehicle is easily accessible at any time. The information ranges from waypoint navigation, position and its derivatives, sensor raw and filtered data, estimations, sensors status, calibration and control parameters, flight modes, input commands, ESC status and others.
Fortunately, this type of Drones requires minimum costs to fly which include a licensed pilot and a ground controller. It is both effective and cheap. For this reason, the first path chosen to calculate the aerodynamic coefficients is, indeed, log based.

4.2.2 2D Fluid Dynamics

Initially, fluid dynamics simulations were avoided due to their tendency to require significant time and computational resources.
Its accuracy in estimating the aerodynamic behavior can be greater than the log's-based method although the solution might suffer from divergence at specific motion fields which lead. In the end, this method has been partially used to extend the range of the angle of attack.

4.2.3 Open-Source Software

To validate the results obtained, two different open-source software have been used:
Tornado and AVL. Whereas Tornado coefficients have been calculated in another colleague's thesis, [5], AVL has instead been implemented to check the exactness of the two results.
These open sources only work at low angles of attack which is unfortunately a very small range compared to what's needed for this drone's dynamics. On the other hand, they offer a great variety of calculations which range from Trim calculation, dynamic response, eigenvalue, and others, all of which have been taken advantage of.

5 MOTOR MODEL

ArduPilot logs the PWM channels' output from the Radio Controller. Despite being useful, this value is not sufficient to correctly calculate the thrust from the propellers. The electric current, voltage, speed and other specifications are necessary to have a better estimation of both *Thrust* and *Torque* provided by the propellers.

The data needed from the battery is the resistance, the current output, and the voltage.

From the ESC, the resistance is taken from the datasheet.

The motor is described by its most important value, the K_V and its resistance.

Since the only logged values which vary over time are voltage, current and PWM, they are "synced" with the rest of the data from the other parts of the model through a time interpolation. After the time-interpolation, these signals are sent inside a function as inputs.

The model calculates the forces acting on the propellers at each instant and derives the rotational acceleration which is then integrated. The rotational speed is then used to estimate the torque and the thrust. Torque and Thrust are calculated by interpolation of the rotational speed. The interpolating coefficients have been calculated through the data obtained from the bench tests. After calculating the Static Thrust, a second calculation is performed using the airspeed. As will be shown, the generated thrust strongly depends on the airspeed.

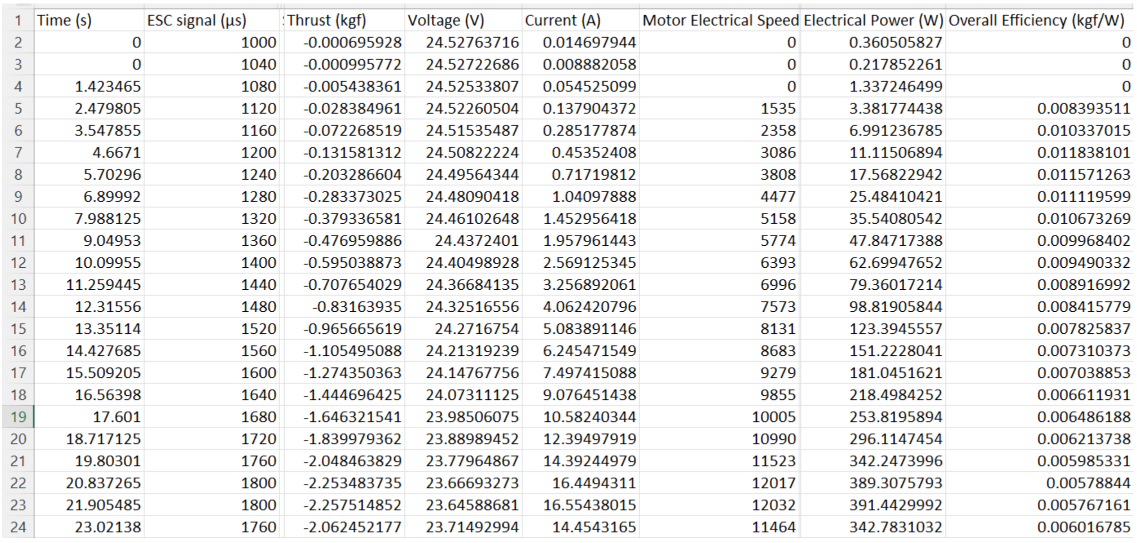
5.1 BENCH TESTS

After a simple estimate of the required thrust to keep both drones in a levelled, stationary flight, it appeared obvious that the data provided by the producing company was extremely optimistic. Therefore, conducting *Bench Tests* became the only solution to successfully obtain the necessary coefficients. The tests were performed both in a lab at Polytechnic of Turin and in lab at Pros3. The gathered data was later post processed.

Initially, *static tests* were performed to precisely calculate the Thrust and Torque at different Rotational Speed. Afterward, dynamic tests were carried out to have the reference data needed to correctly create the Motor Model. This led to an accurate model of the motor at zero velocity, but no tests were performed for dynamic thrust estimations.

The propeller, a XOAR 16x6, is made of wood, is fixed, and designed for electrical motors only. Various sensors—including a load cell, ammeter, voltmeter, chronometer, and IR sensors were used to calculate the data. Afterward, the data was collected inside multiple Excel pages it was later postprocessed in MATLAB to obtain the result that will be described in the following.

During the tests the quantities calculated were Time [s], ESC [PWM], Thrust [kgf], Voltage [V], Current [A], Rotational Speed [RPM], POWER [W], Efficiency [kgf/W]. The following figures displays the data obtained from the bench tests.



1	Time (s)	ESC signal (µs)	Thrust (kgf)	Voltage (V)	Current (A)	Motor Electrical Speed	Electrical Power (W)	Overall Efficiency (kgf/W)
2	0	1000	-0.000695928	24.52763716	0.014697944	0	0.360505827	0
3	0	1040	-0.000995772	24.52722686	0.008882058	0	0.217852261	0
4	1.423465	1080	-0.005438361	24.52533807	0.054525099	0	1.337246499	0
5	2.479805	1120	-0.028384961	24.52260504	0.137904372	1535	3.381774438	0.008393511
6	3.547855	1160	-0.072268519	24.51535487	0.285177874	2358	6.991236785	0.010337015
7	4.6671	1200	-0.131581312	24.50822224	0.45352408	3086	11.11506894	0.011838101
8	5.70296	1240	-0.203286604	24.49564344	0.71719812	3808	17.56822942	0.011571263
9	6.89992	1280	-0.283373025	24.48090418	1.04097888	4477	25.48410421	0.011119599
10	7.988125	1320	-0.379336581	24.46102648	1.452956418	5158	35.54080542	0.010673269
11	9.04953	1360	-0.476959886	24.4372401	1.957961443	5774	47.84717388	0.009968402
12	10.09955	1400	-0.595038873	24.40498928	2.569125345	6393	62.69947652	0.009490332
13	11.259445	1440	-0.707654029	24.36684135	3.256892061	6996	79.36017214	0.008916992
14	12.31556	1480	-0.83163935	24.32516556	4.062420796	7573	98.81905844	0.008415779
15	13.35114	1520	-0.965665619	24.2716754	5.083891146	8131	123.3945557	0.007825837
16	14.427685	1560	-1.105495088	24.21319239	6.245471549	8683	151.2228041	0.007310373
17	15.509205	1600	-1.274350363	24.14767756	7.497415088	9279	181.0451621	0.007038853
18	16.56398	1640	-1.444696425	24.07311125	9.076451438	9855	218.4984252	0.006611931
19	17.601	1680	-1.646321541	23.98506075	10.58240344	10005	253.8195894	0.006486188
20	18.717125	1720	-1.839979362	23.88989452	12.39497919	10990	296.1147454	0.006213738
21	19.80301	1760	-2.048463829	23.77964867	14.39244979	11523	342.2473996	0.005985331
22	20.837265	1800	-2.253483735	23.66693273	16.4494311	12017	389.3075793	0.00578844
23	21.905485	1800	-2.257514852	23.64588681	16.55438015	12032	391.4429992	0.005767161
24	23.02138	1760	-2.062452177	23.71492994	14.4543165	11464	342.7831032	0.006016785

Figure 9: Bench tests' outputs

Please note that the RPM must be divided by two because the bench was mistakenly configured with twice the number of coils.

The results obtained from the subsequent analysis are now presented.

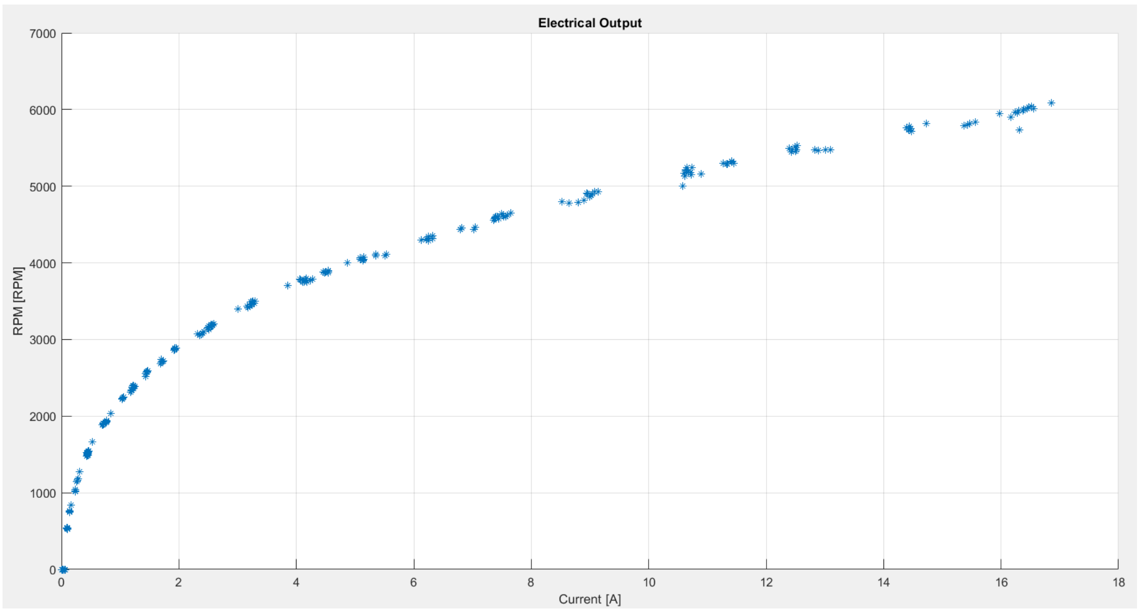


Figure 10: Drawn Current

As it can be seen from the Figure 8, the drawn current increases as the rotational speed increases. Please notices that these values were obtained at fully charged battery, 24.5-24.8 V. The tests were consequential and therefore the battery voltage decreased over time.

Next, the Thrust was interpolated with the PWM through a second-degree polynomial. The only condition that was set externally was that the curve had to pass from 0 at 1000 PWM. To achieve this condition, the data was postprocessed because the load cell was measuring miniscule values of Thrust even at 1000 PWM, when the motors were still. Therefore, these values were neglected, resulting in the interpolation displayed in the Figure 9.

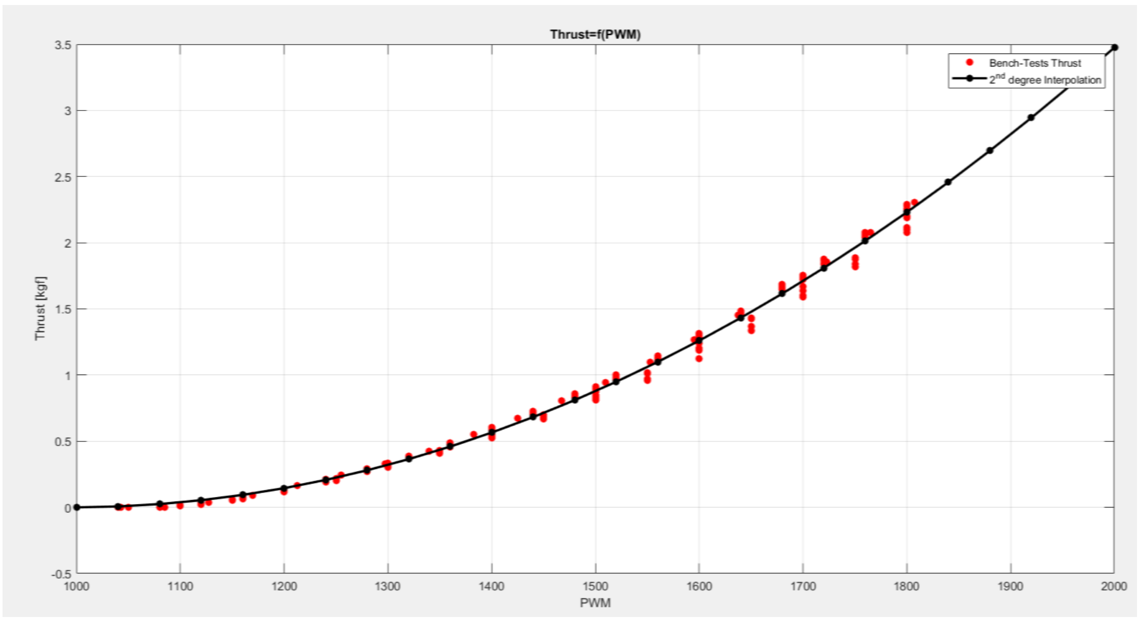


Figure 11: Thrust as a function of PWM signal

The polynomial is the following:

$$T = 0 + 0.0068 \cdot PWM + 3.3986 \cdot PWM^2$$

The bench tests were performed between 1000 and 1800 PWM with a maximum thrust of 2.25 kgf.

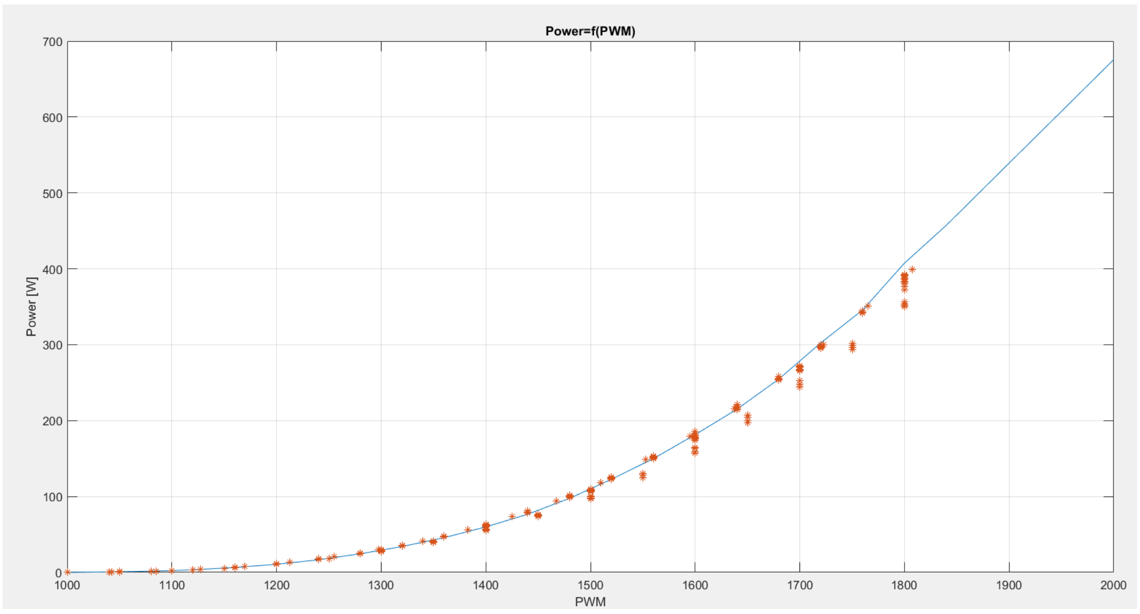


Figure 12: Power as a function of PWM signal

Normally a typical PWM value is 1400 corresponding to 50-Watt power consume and 0.5 kgf thrust. A rough estimate of the endurance of the drone can now be calculated.

$$\frac{2 \cdot P_{Motor}}{\eta_{ESC} \cdot \eta_{Mot}} + P_{Avionic} \approx 150W + 25W \approx 175W$$

Which means that with a 6S 1P 5200mAh battery, the drone can fly for 40 minutes. The calculation is not precise, its scope is to provide an insight into why this configuration can be so advantageous.

Finally, an interpolation of the Thrust as a function of RPM has been performed.

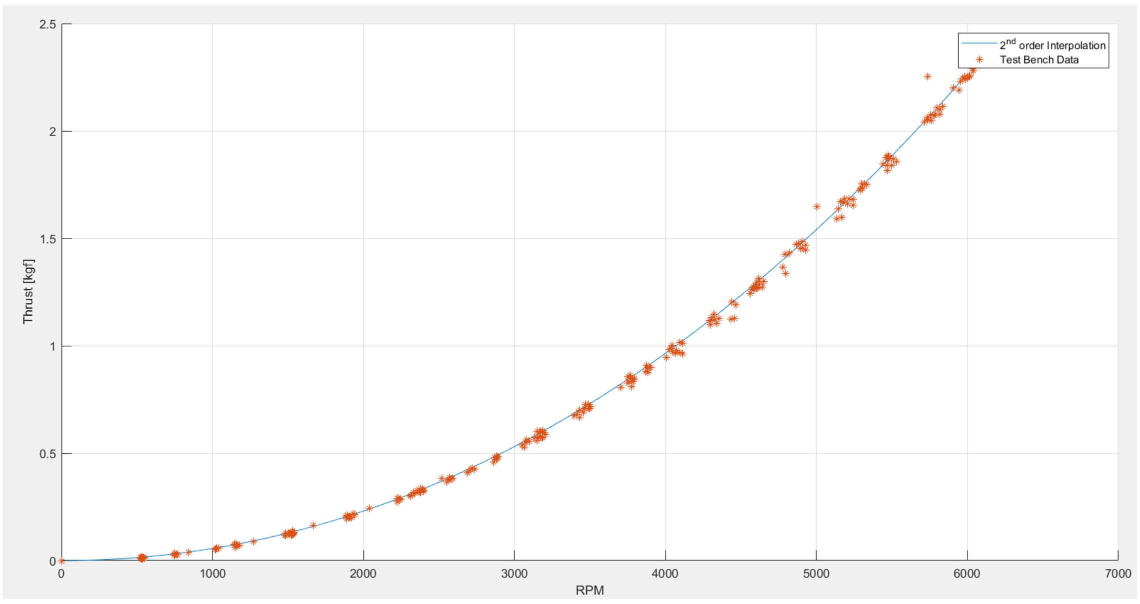


Figure 13: Thrust as a function of RPM

As it can be seen in the graph above, the highest thrust values obtained are at 6000 RPM which correspond to roughly 2.5 kgf of thrust and 1800 PWM.

The obtained coefficients are the following:

$$T = c_3 + c_2 w + c_1 w^2$$

Where $c_1 = 5.5\ 10^{-8}, c_2 = 1.0\ 10^{-14}\ c_3 = 0$

Again the c_3 value has been set to 0 to have 0 power required at 0 rotational speed.

The calculated data has been summarised in the next tables. It is important to underline that all values obtained are approximate yet meaningful.

Table 4: Propellers performance at 50% throttle

Thrust [kgf]	0.85
RPM	3800
Power [W]	100

Table 5: Propellers performance at 100% throttle

Thrust [kgf]	3.5
RPM	8000
Power [W]	700

5.2 Physical Model

Inside the motor model, the integration requires a step time significantly smaller than the logged data. For this reason, it is previously interpolated to have more time frames.

The model starts with no rotational velocity nor current.

First, the available voltage is calculated:

5.1

$$V = V_{battery} - R_{Battery} \cdot I_{battery}$$

Where:

- $V_{battery}$: is the voltage outputted by the battery, measured with the board;
- $I_{battery}$: is the current outputted by the battery, measured by the board's ammeter;
- $R_{Battery}$: is the internal resistance of the battery, which is provided by the producing company and is hypothesized to be constant.

The voltage is taken and lowered by the multiple's resistances in the circuit; the resulting value is therefore further lowered by the actual Throttle required, which is a value between 1 and 0.

5.2

$$V_{Motor} = V \cdot Throttle = \left(V_{battery} - R_{Battery} \cdot I_{battery}\right) \cdot Throttle$$

The current reaching the motor is obtained multiplying the K_V constant with the just calculated motor voltage, though it also must take into consideration the ESCs' resistances:

5.3

$$I_{Motor} = \frac{K_v \cdot V_{Motor} - w}{(R_{Motor} + R_{esc}) \cdot K_v}$$

The torque provided by the motor is subsequently calculated multiplying the obtained current by the other fundamental motor coefficient K_t.

5.4

$$Q_{Motor} = I_{Motor} \cdot K_t$$

The Drag Torque is now calculated with the Blade Element Theory:

5.5

$$Q_{Drag} = K_P \cdot \rho \cdot w^2 \cdot r^5$$

Where:

- ρ is the density of the air, calculated exactly in the same way as described in the previous equations of motion;
- w is the current rotational speed of the propeller;
- r is the radius of the propeller's blades;
- K_P is the Torque coefficient.

The rotational acceleration is then calculated as a difference of Torques divided by the moment of inertia:

5.6

$$\dot{w}^{k+1} = \frac{Q^k_{Motor} - Q^k_{Drag}}{I}$$

Where *I* is the moment of inertia, calculated as the propeller's blades were parallelepipeds, therefore:

5.7

$$I = n\frac{1}{12}bh^3$$

Where:

- n*:* is the number of blades;
- b*:* is the chord length;
- h: is the length of the blade.

The rotational acceleration is then multiplied by the time interval and added to the previous value of rotational speed:

5.8

$$w^{k+1} = w^k + \dot{w}^k\Delta t = w^k + \frac{Q_{Motor} - Q_{Drag}}{I}\Delta t$$

The Thrust is calculated in the same manner as the torque; hence, the formula is as follows:

5.9

$$T = K_T(w) \cdot \rho \cdot w^2 \cdot r^4$$

The only difference is that the constant is now dependent on the rotational speed; in the following way:

5.10

$$K_T(w) = k_{t_1}w^3 + k_{t_2}w^2 + k_{t_3}w + k_{t_4}$$

These four coefficients have been calculated from the bench tests.

The Drag Torque is calculated using the wing drag equation and, therefore, a Drag Coefficient is needed. Unfortunately, the one provided by the producing company is not accurate and needs to be recalculated.

5.3 Results

The exact model described in the paragraph 5.2 was used to and tested against the data obtained from the bench tests, as can be seen in the following figures. The same Voltage, Current and PWM trends fed to the propeller motor during the bench tests were equally fed to the model.
In the figures, the black line represents the calculated Thrust and Rotational Speed trends obtained from the Bench Test load cell, whereas the green line denotes the model calculated Thrust and Rotational Speed.
The relative error has been found to be less than 1% at high rotational speeds. At low speeds the error increase substantially, although it is not significant as no thrust or torque is produced below 1000 RPM. The absolute error has the opposite trend but remain small even at high RPM.

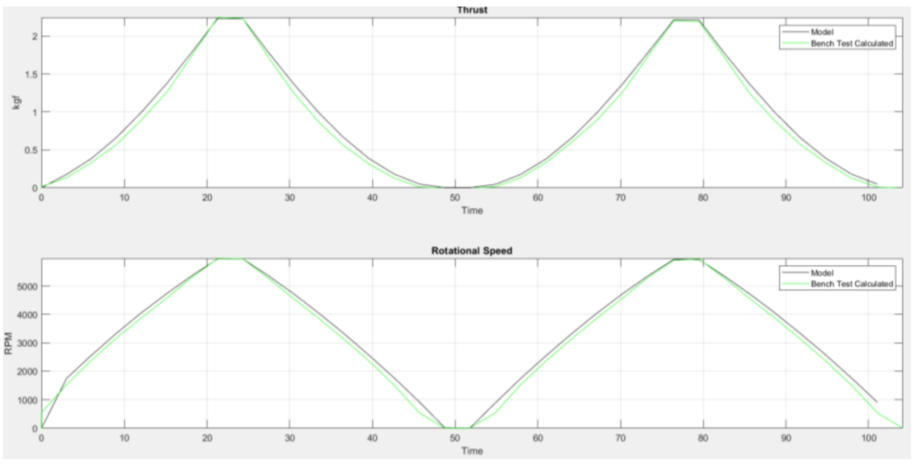


Figure 14: Test number 1

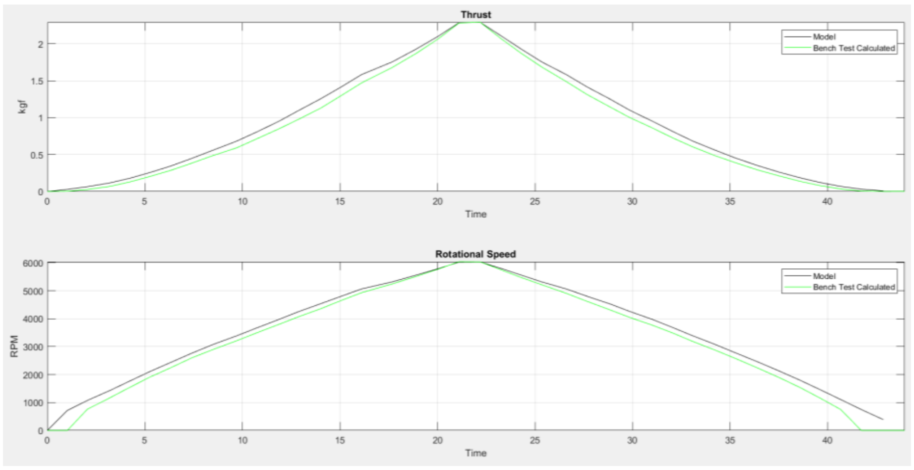


Figure 15: Test number 2

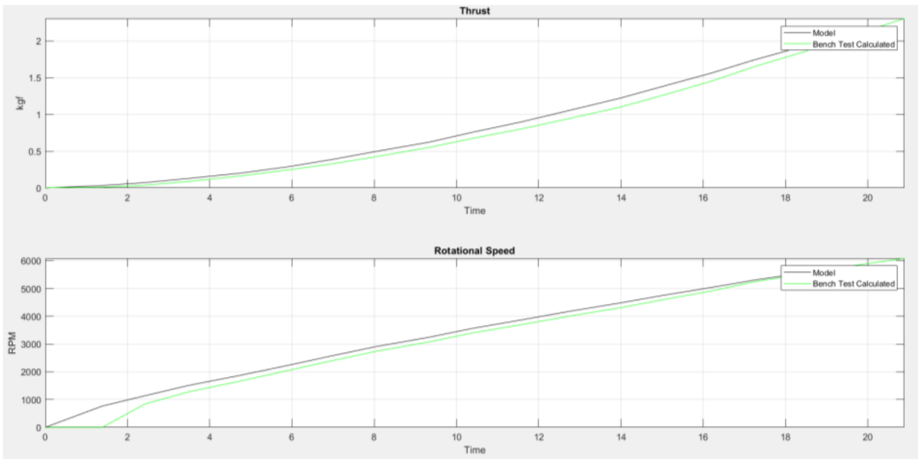


Figure 16: Test number 3

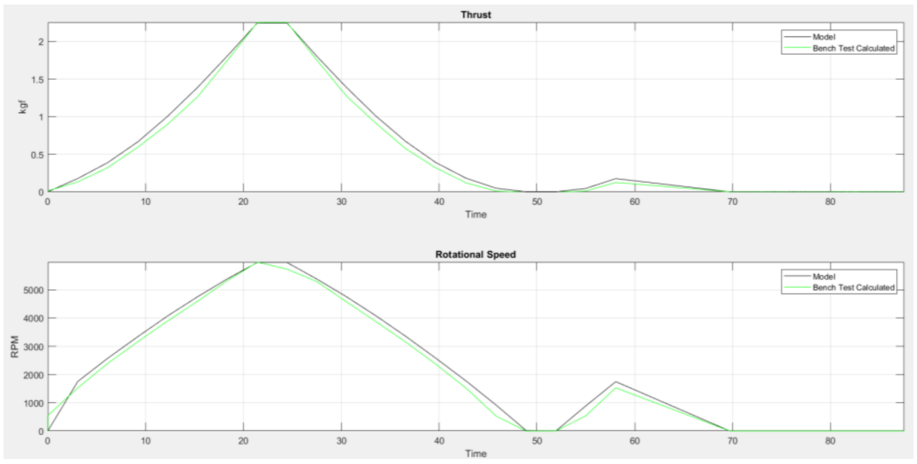


Figure 17: Test number 4

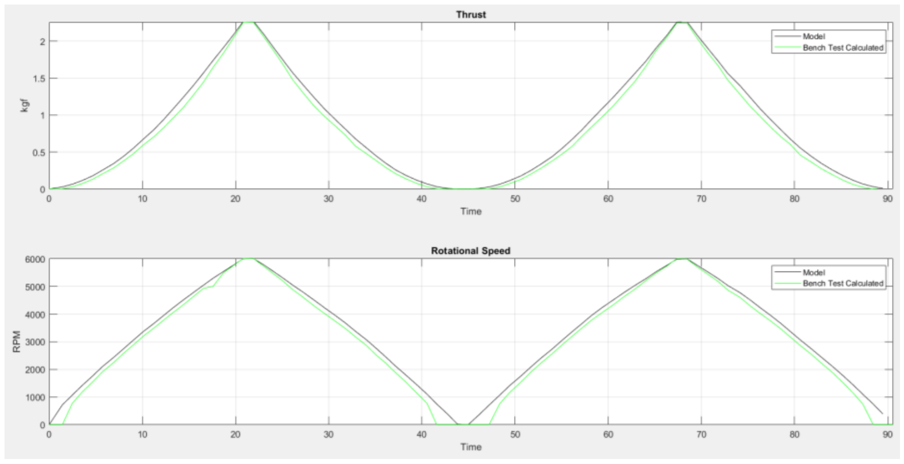


Figure 18: Test number 5

As it can be seen, the model is very precise when the Angular Speed passes the 1000 RPM threshold. The model fails to correctly simulate the starting phase when the propeller is almost still. This is a significant issue, as Strategy's take off phase is very fast and seems to be very stable due to the high thrust to weight ratio.

5.4 Thrust-Speed Interpolation, 1st Attempt

The data from the producing company's datasheet was analysed to perform a Thrust Speed interpolation. As previously said, this data is not precise and often time needs to be corrected. The following figure has been obtained with the data from the datasheet and is a 3D graph of the Thrust:

$$T = f(\omega, V)$$

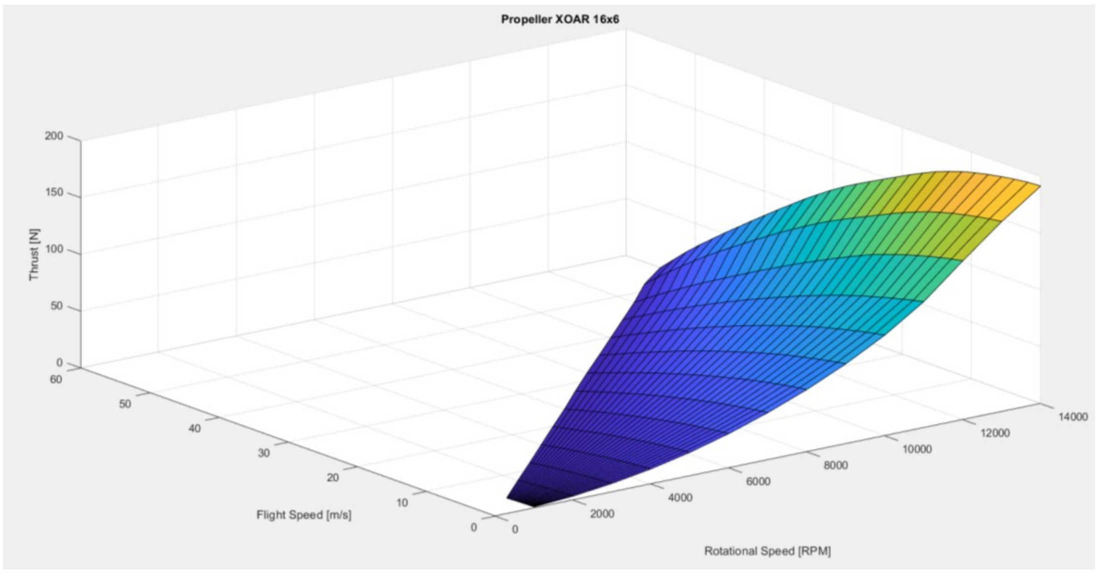


Figure 19: Manufacturer's data interpolation

Despite its apparent consistency, there's a significant variance from what has been measured during the bench test and observed throughout the flight tests.

This interpolation is by far the most complicated one, as no Wind Tunnel was available to study the propeller performance.

5.5 Thrust-Speed Interpolation: 2nd Attempt

The second attempt aimed to use general empirical equation created by drone pilots and engineers all over the world.

The following equation has been created through the data gathered on the internet, uploaded by hundreds of different drone manufacturers. This data was analysed, and the result was a semi-analytical and semi-empirical solution, which is now briefly illustrated.

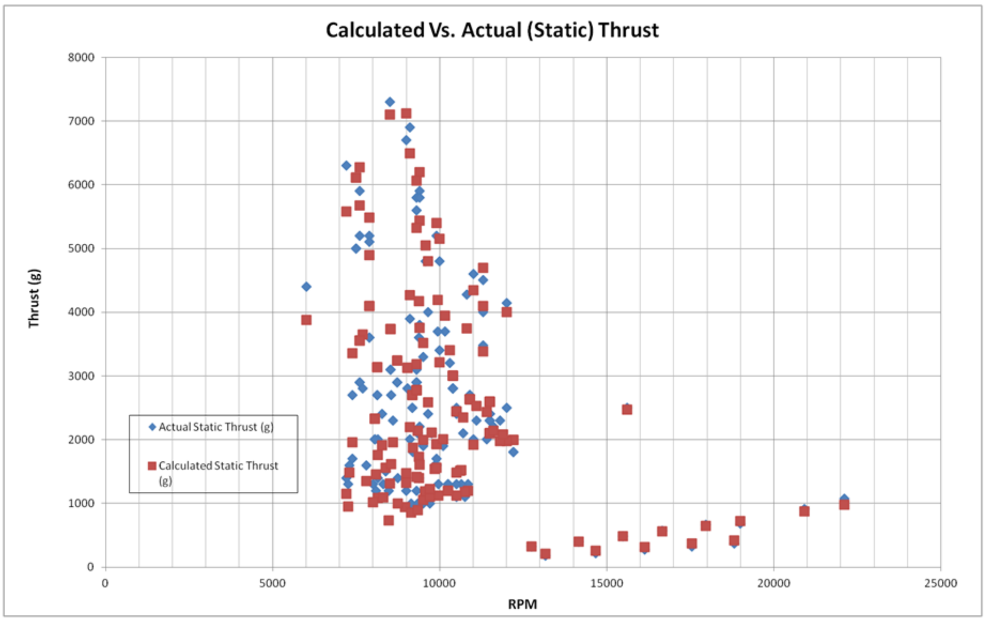


Figure 20: Calculated Thrust Vs Actual Thrust

In the graph above, in blue is the actual thrust versus red, which represents the calculated thrust. The Equation 5.11 is useless for this project since the bench test provided more accurate data. Nevertheless, the equation is necessary, as the subsequent part is a useful step for this work. In fact, it tries to interpolate more data to obtain the dynamic thrust.

The equation is now rapidly obtained. [6]

Newton's second law:

$$F = \frac{d(mV)}{dt} = \frac{dm}{dt}V = \dot{m}\Delta V = \dot{m}(V_{exit} - V_{a/c})$$

Where:

- V_{exit} is the induced velocity after the propeller;
- $V_{a/c}$ is the velocity of the aircraft.

Writing the flow passing through the propeller in terms of constants:

$$\dot{m} = \rho A V_{exit} \rightarrow F = \rho A V_{exit}\left(V_{exit} - V_{\frac{a}{c}}\right) = \rho A V_{exit}^2 - \rho A V_{exit} V_{a/c}$$

Finally:

$$A = \frac{\pi d^2}{4} \rightarrow \qquad F = \rho\frac{\pi d^2}{4}\left(V_{exit}^2 - V_{exit}V_{a/c}\right)$$

This equation lets user take the information usually provided by the manufacturer and plug them in the equation without the need of conversion. Therefore, it uses Miles per hour instead of meter per second, inches instead of meters and revolution per minutes instead of radians per seconds.

Converting all the measurements results in the following equation:

5.11

$$F = 1.225\frac{\pi(0.0254\,d)^2}{4}\left(RPM \cdot 0.0254 \cdot Pitch \cdot \frac{1}{60}\right)^2$$

Until this moment, no empirical correction factor has been introduced.

Afterward, using the gathered data to perform a data optimization, the equation transformed into the following:

5.12

$$F = 1.225\frac{\pi(0.0254\,d)^2}{4}\left[\left(RPM \cdot 0.0254 \cdot Pitch \cdot \frac{1}{60}\right)^2 - \left(RPM \cdot 0.0254 \cdot Pitch \cdot \frac{1}{60}\right)V_0\right]\left(\frac{d}{3.29546\, \cdot Pitch}\right)^{1.5}$$

It can be noted that the following factors have not been considered:

a) Number of propeller's blades.
b) Velocity distribution along the propeller blade.
c) Propeller chord, area and characteristics.

However, despite comparing this equation to real data, the results were very unsatisfactory.

The propeller data has been inserted into the Excel sheet and the following graph represents the results. [6]

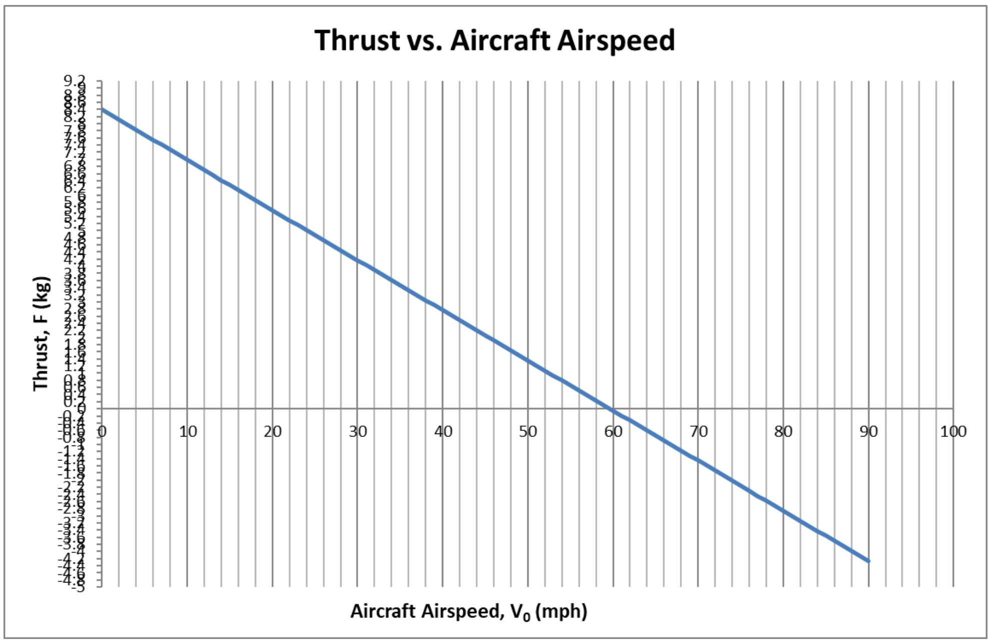


Figure 21: Thrust as a function of Aircraft Airspeed

Already at first glance, the curve cannot be precise as these two quantities are not linearly related. The static thrust is completely out of the range but, as explained by the creator of the previous formula, the most precise value measured by the equation is by far the maximum velocity reachable. Indeed, the maximum speed shown is around 59 mph or about 26-27 m/s.

The actual maximum measured velocity during the flight tests was about 25 m/s. The discrepancy of 3m/s is completely understandable, given that the autopilot itself limit the time at which the maximum PWM can be maintained, and therefore limiting the maximum speed reachable.

5.6 Thrust-Speed Interpolation, 3RD Attempt

The third attempt has been made using the data gathered from the flight campaign.

The objective is to obtain the actual propeller thrust as a function of both aircraft's speed and propeller's rotational velocity.

To start, a first estimate of the Drag and Lift coefficient was needed. Once obtained, the thrust could be calculated from the logs choosing specific sections. To simplify the calculation and decrease the averaging error, two conditions needed to be met for a section of the log to be chosen:

a) Steady State flight;
b) Levelled Flight.

The Steady State condition imposes both a nearly constant regime of the propellers and Drag force. The second condition, instead, imposes the Euler Angle to be zero. This allows to remove terms inside of the equation, which would have brought more approximation to the results.

The equation which needs to be solved now is the following:

$$2T = D = \frac{1}{2}\rho V^2 S C_D$$

The "2" at the left side of the equation implies that the propellers' thrust are being calculated separately, but in this case are equal. The drag coefficient has been estimated using CFD simulations.

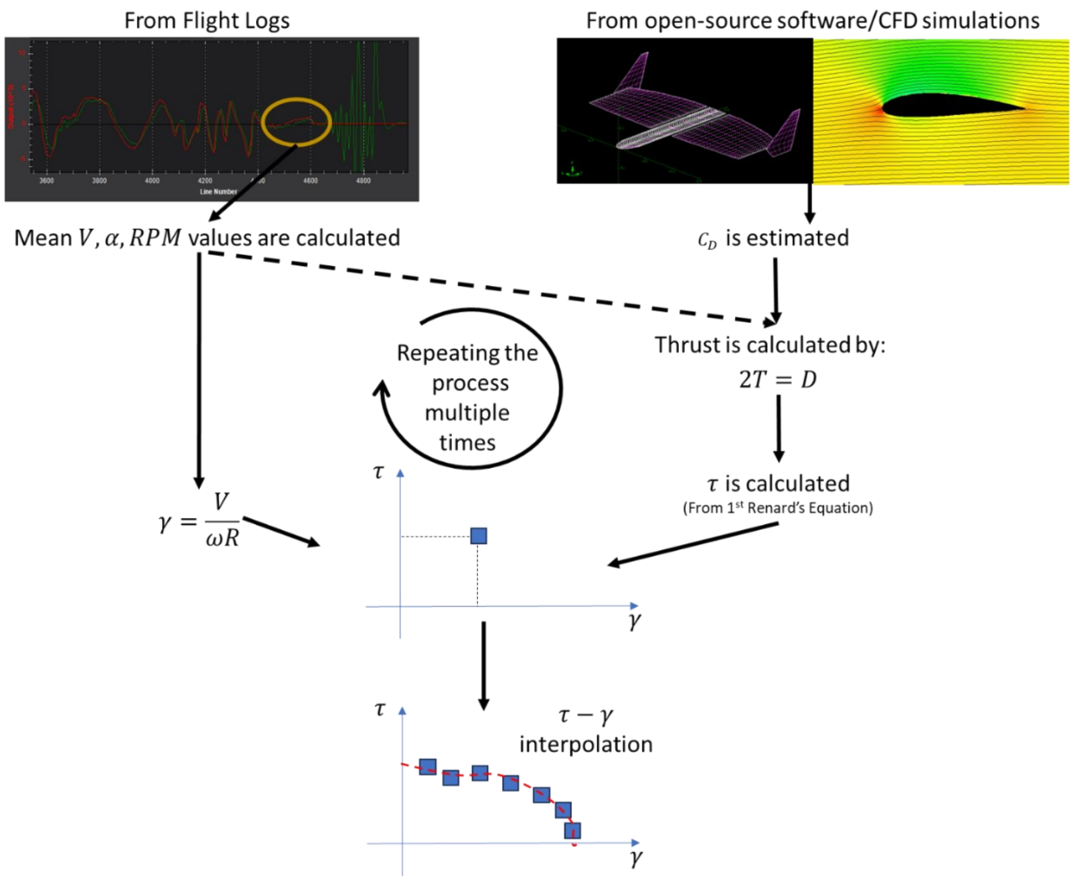
To select the correct portions of the log to be analysed, the two necessary conditions are:

a) Constant Speed;
b) Zero Pitch Angle.

Even when selecting very specific portions of the flight, the calculated variables are extremely noisy. The most disturbed estimation is the Angle of Attack which is measured by the Pitot's Tube in the front of the fuselage. To obtain a constant averaged value, a simple average of the measurement has been taken for each portion. It is now possible to estimate the actual thrust.

To create the interpolation, the rotational speed of the propeller is needed. Once obtained, the advance ratio can be calculated, and therefore the Thrust coefficient of that specific point.

The following figure represents a scheme of the implemented logic.



From Flight Logs
From open-source software/CFD simulations
Mean V, α, RPM values are calculated
C_D is estimated
Thrust is calculated by:
$2T = D$
Repeating the process multiple times
τ is calculated
(From 1st Renard's Equation)
$\gamma = \frac{V}{\omega R}$
τ
γ
$\tau - \gamma$ interpolation

Figure 22: Dynamic Thrust calculation procedure

Once the process has been repeated multiple times, a curve could be interpolated in the $\gamma - \tau$ plane, representing the thrust coefficient as a function of the advance ratio. Unfortunately, the points collected couldn't cover all the curve, since extreme flight conditions would need to be reached to gather the complete data. A third order interpolation has been performed knowing that outside of the advance ratio range is approximating the actual curve.

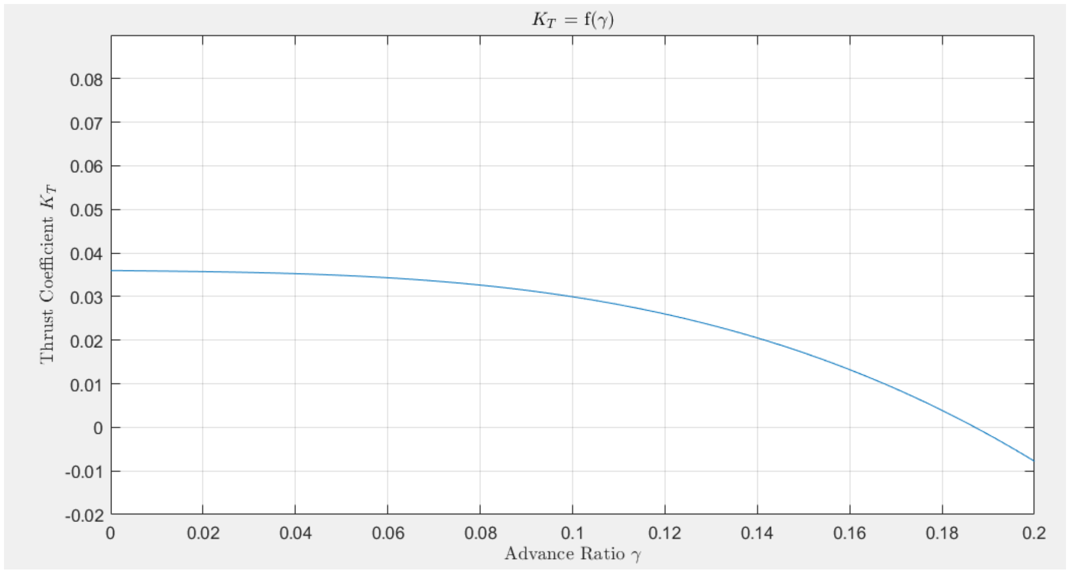


Figure 23. Thrust Coefficient as a function of the Advanced Ratio

Although being incomplete, this approximation gives much more realistic results. In fact, the simulator's results based on this curve, and the maximum velocity reached during flights correspond.

5.7 PROPELLER'S BLADE AIRFOIL

The propeller's blade Airfoil is E224, shown in the Figure below.

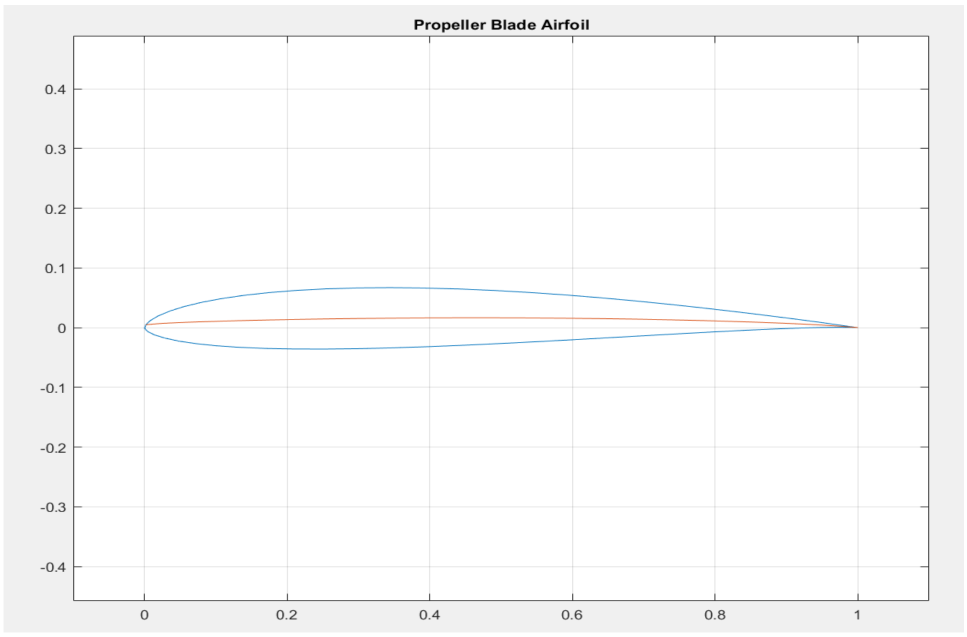


Figure 24: Propeller's Blade Airfoil

The blade's Airfoil is provided by the manufacturer, while its coefficients are obtained from Literature. The chord profile along the blade span is again provided by the manufacturer.

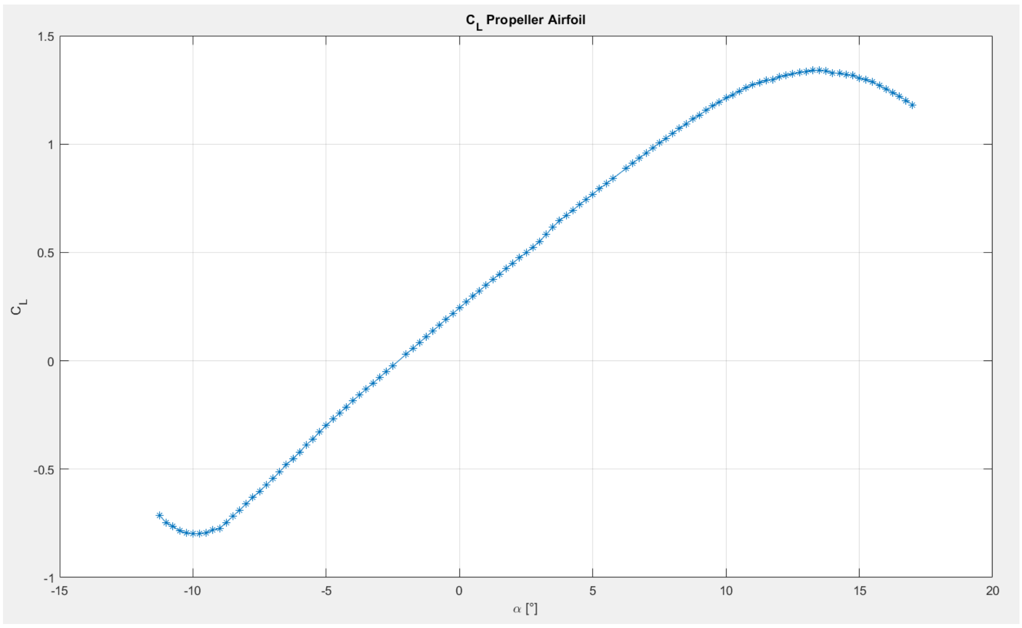


Figure 25: Airfoil's Lift Coefficient

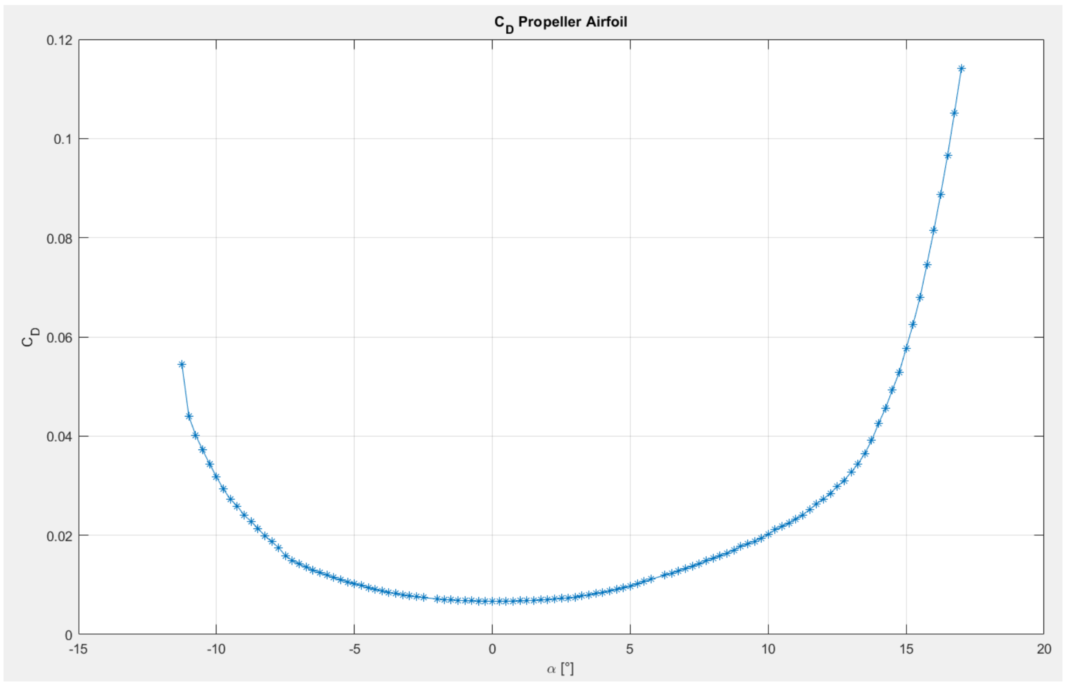


Figure 26: Airfoil's Drag Coefficient

The propeller's zero lift angle is $\alpha_0 = -2.25°$.

The airfoil has a maximum thickness of 10.2% at 29.8% of chord. The maximum camber is 1.5% at 54.2% of chord. The coefficients have been found on AirfoilTools Website, [7] and they were obtained for 10^6 Reynolds value.

5.8 Slipstream

The induced velocity generated by the propeller has been at first estimated at the propeller plane. The velocity depends on the radial distance from the propeller axis therefore:

$$V_i = f(r)$$

To determine the axial velocity for each radial distance, the inflow angle must be evaluated. Using the Blade Element Theory equation and the moment theory, the following equation can be written [8]:

5.13

$$C_{l_\alpha}(\theta - \alpha - \varphi) - C_D \tan(\varphi) = 8\pi \sin(\varphi)\left(\frac{\omega r tan(\varphi) - V_X}{\omega N c}\right)$$

Where:

$-C_{L_\alpha}$, C_D are the 2D lift slope and drag coefficient of the blade's airfoil;

$-\ \varphi$ is the inflow angle;

$-\ \alpha_0$ is the zero lift angle;

$-\ c$ is the blade's chord;

$-\ r$: is the radius.

$-\ V_X$is the aircraft speed;

$-\ \omega$ is the angular speed;

$-$ N: is the number of blades.

This equation must be solved iteratively for each radius value, to obtain the inflow angle and, therefore, determine the radius along the near field slipstream, as well as the velocity immediately after the propeller's plane.

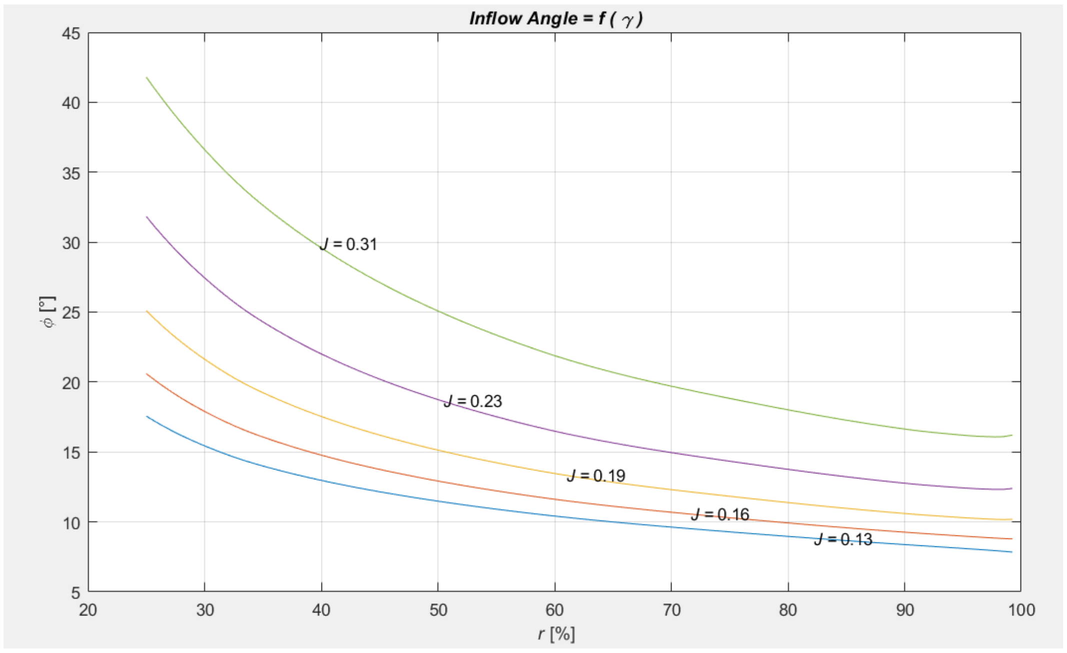


Figure 27: Inflow Angle as a function of the Advanced Ratio and the radius

The graph shows the inflow angles values calculated for different advance ratios γ and across the whole blade span[1].It's possible to notice how the increase of propeller's rotational speed decreases the inflow angle across the whole blade. The inflow angle decreases moving from the hub to the tip of the blade.

The propeller induced flow creates a local speed increase and a local angle of attack difference. To calculate these two quantities, different equations have been taken from literature. Since the wing leading edge is close to the propeller plane, only the Near Field Region of the slipstream has been taken into consideration.

The induced velocity of the propeller can be estimated using the following: [8]

5.14

$$V_{avg}(x) = V_{X_0}\left(\frac{-1+\sqrt{1+\frac{8K_T}{\pi}}}{2}\right)\left(1+\frac{x}{\sqrt{R_P^2+x^2}}\right)$$

[1] Please notice that both "J" and "γ" are used to indicate the Advanced Ratio.

[10]

Where:

$-R_P$ is the propeller radius.

$-K_T$is the thrust coefficient and is calculated as: $K_T = \tau = \frac{T}{\rho R_P^4 \omega^2}$

$-x$ is the distance from the propeller plane.

It can be noted that the velocity at $x = 0$ is equal to:

$$V_{avg}(x) = V_{X_0}$$

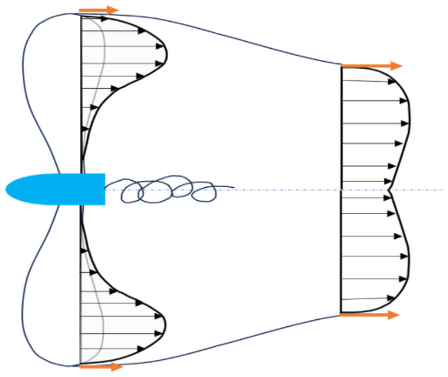


Figure 28: Near Field Flow Development

5.9 Slipstream Fast Calculation

To integrate the slipstream model inside the simulator, a faster way has been selected.

In fact, the standard equation 5.14 described previously, is very computational expensive. It needs to be solved iteratively for each radius value. It is extremely non-linear, and it would make the model run a lot slower.

The method proposed by Shamsheer Singh Chauhan instead provides a valid approximation, while being very light in terms of computational cost. This method also provides a simple method to estimate the propeller tangential velocity. Although it has been calculated, the tangential velocity has not been utilized in the 6 *DoF* Simulator created, but it will be of use in future developments.

The method is now briefly described.

The method uses the distribution of axial force, as an input to calculate the slipstream velocity exiting the propeller plane. This method takes advantage of two approximations:

a) Constant Pitch along the blade span;

b) Aerodynamic Forces are always perpendicular to the chord of the rotating blade.

Fortunately, the first hypothesis true for this propeller type, while the second doesn't represent a problem since, given small the pitch angle of the blade, the forces are actually almost perpendicular.

The first step is to model this distribution based on the solution provided by the iterative method previously described. The distribution of axial velocity therefore follows the same curve calculated previously:

5.15

$$f_x = \tilde{F}\tilde{r}^m \left(1 - \frac{\tilde{r}}{a}\right)^n$$

Where $\tilde{r}$ is the normalized radius calculated by the following:

5.16

$$\tilde{r} = \frac{r - r_{min}}{R_{max} - r_{min}}$$

And a, m, n are shape parameters used to obtain the desired force distribution.

Varying these parameters results in changing the shape of the curve and the load distribution. The parameters used are the same that are suggested which follows "Goldstein" distribution:

a=1; m=1; n=0.2

Decreasing *n* shifts the position of maximum thrust toward the tip.

The last parameter is $\tilde{F}$, which makes the integral across the blade resulting in the actual thrust generated by the propeller. It can be easily calculated by defining the other parameters. Doing this, results in obtaining an analytical equation which can be easily integrated inside the simulator. If the distribution a, m, n parameters happen to change, the linear equation which provide the correct value of $\tilde{F}$ can be recalculated. This can be done only once, at the beginning of the simulation:

5.17

$$K = \int_0^1 \tilde{r}^m \left(1 - \frac{\tilde{r}}{a}\right)^n d\tilde{r}$$

So that:

5.18

$$\tilde{F} = \frac{K \cdot T}{R - r_{min}}$$

Given a specific Thrust generated by the propeller, the parameter $\tilde{F}$ can be easily calculated. The following graphs are obtained at a constant Thrust of $T = 3\ kgf = 29.43\ N$.

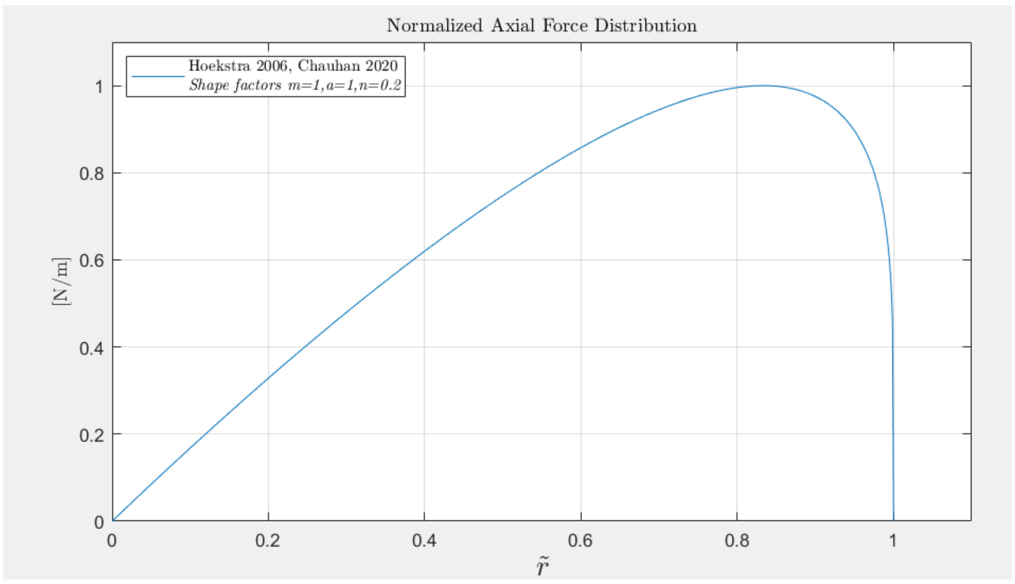


Figure 29: Normalized Axial Force Distribution

As it can be seen, the parameters have been chosen so that the majority of the axis force is produced between 60% to 90% of the total blade length. Most of axial force distributions happen to have the same trait. More advanced studies must be conducted for the adequate choice of parameters. Please notice that in the graph, $\tilde{r}$ has been graphed and not r/R. This means that the graph goes to zero in the X-axis. Once the axial force distribution is known, the induced velocity can be calculated:

5.19

$$V_{i,x} = \sqrt{\frac{V_\infty^2}{4} + \frac{f_X}{4\rho\pi r}} - \frac{V_\infty}{2}$$

Where ρ is the air density, which has been 1.225 kg/m^3 given the altitude the drone can reach.

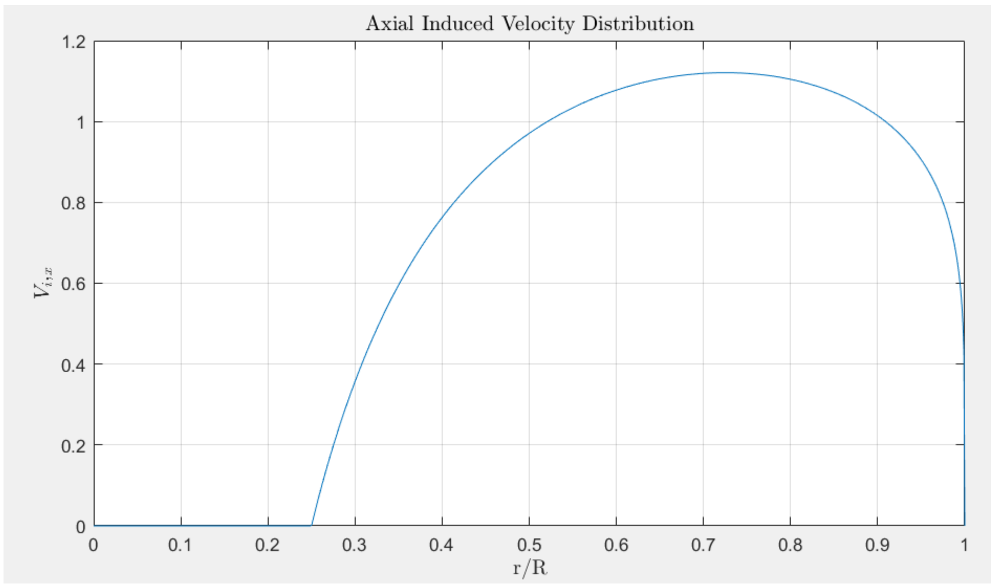


Figure 30: Axial Induced Velocity Distribution

Given the presence of the root square, the induced velocity has smoother top.

Please notice that this time the X axis is the ratio between the radius and the maximum radius. Since the central part of the propeller does not provide any kind of thrust, the induced velocity goes to zero. The same process has been repeated at different airspeeds, as it can be observed in the graph below.

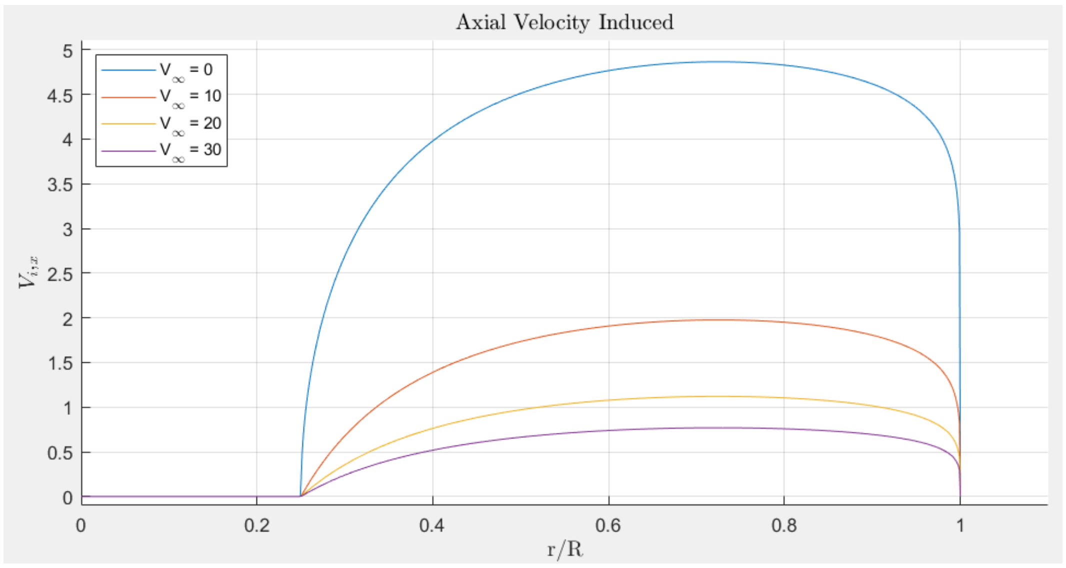


Figure 31: Axial Velocity Induced at different airspeeds

As it can be seen the effect dramatically decrease when the aircraft speed increases from 0 m/s. This data is extremely important for the *Copter* mode as it represent the only velocity washing the wing. During hovering, this speed tends to create a lift which moves the drone horizontally. It also increases, very slightly, the motors' angular velocity required for hovering.

Although the following equations have not been used, they were still implemented inside the model for completeness. Future uses might take advantage of this calculation, such as for propeller-wing-fuselage interaction studies.

The tangential force radial distribution can now be calculated by:

5.20

$$f_\theta = f(r) = \frac{f_X \cdot P/D_\emptyset}{\pi \cdot r/R}$$

And then the tangential velocity can be obtained by:

5.21

$$V_\theta = \frac{f_\theta}{2\pi r\rho\left(V_\infty + V_{i,x}\right)}$$

The tangential force distribution is showed below: [9]

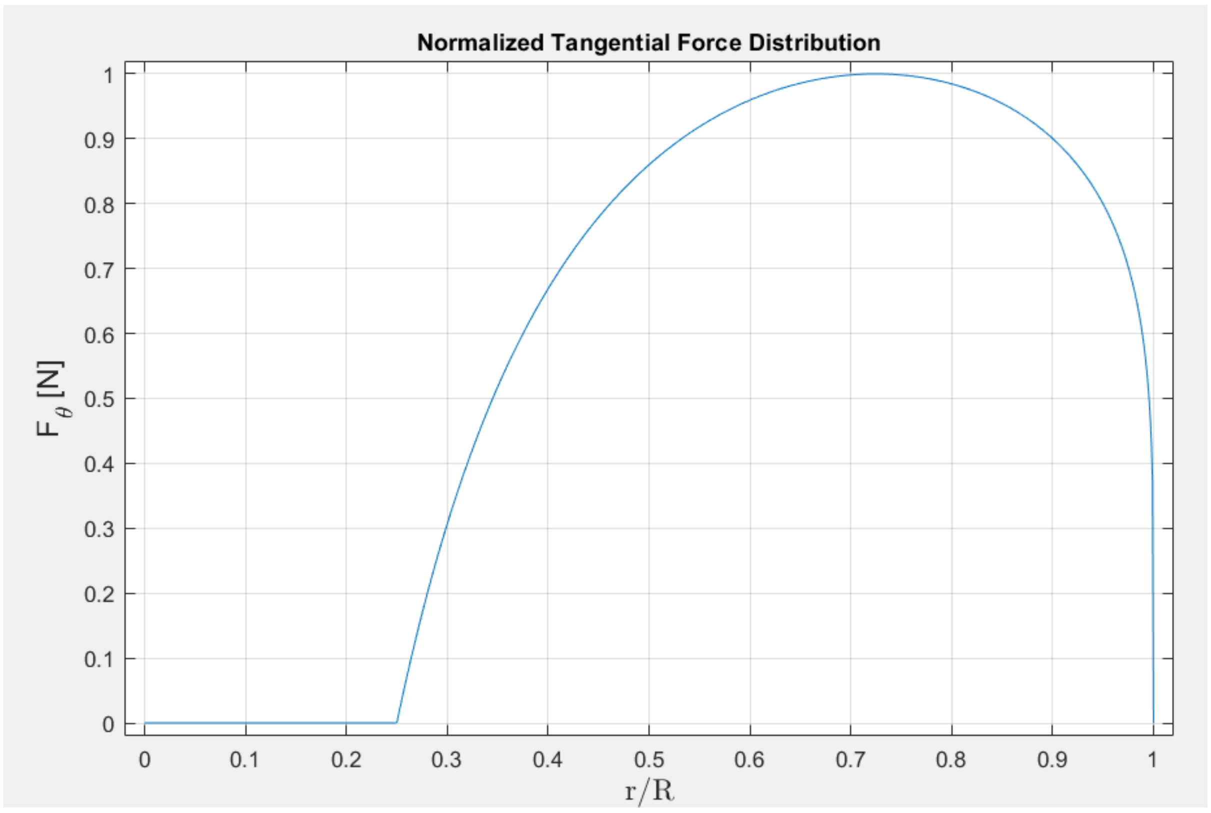


Figure 32: Normalized Tangential Force Distribution

6 ATHENA VORTEX LATTICE

As the previous method lacks consistency, another path has been followed to estimate the aerodynamic coefficients. Athena Vortex Lattice has been selected as a suitable solution to estimate the coefficient in the linear region of the drone's dynamics. [10] [11] [12]

The AVL Vortex-Lattice model is optimal for aerodynamic configurations with thin lifting surfaces at small angles of attack and sideslip.

It utilizes single-layer vortex sheets discretized into horseshoe vortex filaments to represent surfaces and their trailing wakes. Slender bodies like fuselages can be easily modelled.

Unsteady flow is valid only for slow oscillations, adhering to specific angular velocity limits.

Compressibility is addressed through the Prandtl-Glauert transformation, and the validity of linearization depends on the Mach number, expressed by the PG factor.

Since the results outside these limits should be interpreted with caution, Angle of attacks higher than 12 degrees have not been inserted inside the simulator, although they have been calculated. The same goes for the sideslip angle.

The following Figure represents a scheme of AVL works. It needs to be provided with three input files containing respectively:

- **Run**: Specifies the state of the aircraft so variables such as speed, angle of attack, sideslip and others need to be inserted here.
- **Geometry**: Defines the aircraft's geometry dividing it into bodies and sections. In each section an air foil file must be imported. The movable surface is specified here and, by specifying the hinge position, it's possible to create variable chord surfaces.
- **Mass**: Defines each body and/or section mass, inertia and Centre of Gravity.

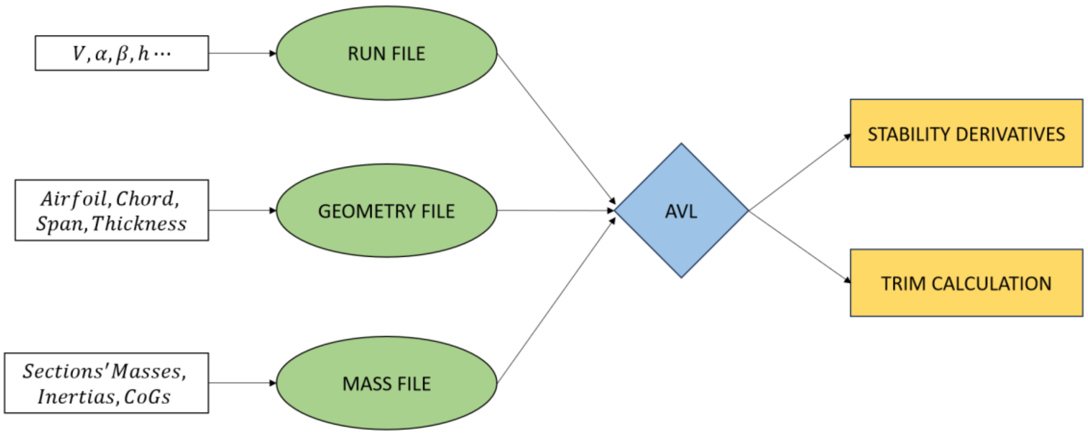

$V, \alpha, \beta, h \cdots$
RUN FILE
$Airfoil, Chord, Span, Thickness$
GEOMETRY FILE
$Sections' Masses, Inertias, CoGs$
MASS FILE
AVL
STABILITY DERIVATIVES
TRIM CALCULATION

Figure 33: AVL scheme

It is important to notice that AVL has broader capabilities, but in this study only stability derivatives and trim calculation were obtained.

This is because the data outputted by AVL was not considered to be extremely reliable and therefore subsequent calculation would have been pointless.

The geometry file and the mass files have been created and used to obtain the coefficients for both aileron and elevator configuration. The movable surface needs to be defined only in one way but by creating two different files, all the coefficients can be estimated.

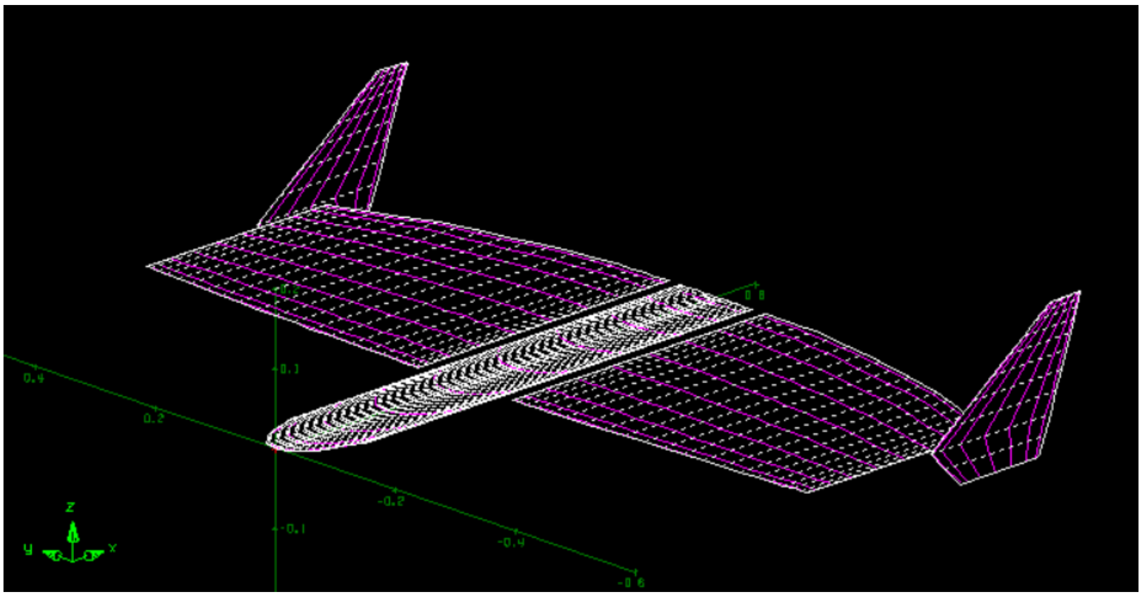


Figure 34 : AVL Geometry

As it can be seen, the aircraft has been divided into fuselage, wings and winglets. The winglets' airfoil is a common symmetrical NACA 0010.

The following table shows the data obtained by AVL.

Table 6: AVL longitudinal results

α		$\hat{q}$		δ	
C_{L_α}	4.63	C_{L_q}	2.29	C_{L_δ}	0.97
C_{Y_α}	0.0	C_{Y_q}	0.0	C_{Y_δ}	0.0
C_{l_α}	0.0	C_{l_q}	0.0	C_{l_δ}	0.0
C_{M_α}	-0.2418	C_{M_q}	-0.953	C_{M_δ}	-0.00773
C_{N_α}	0.0	C_{N_q}	0.0	C_{N_δ}	0.0

Table 7: AVL lateral-directional results

$\hat{p}$		β		$\hat{r}$	
C_{L_p}	0.0	C_{L_β}	0.0	C_{L_r}	0.0
C_{Y_p}	-0.0229	C_{Y_β}	-0.2002	C_{Y_r}	0.02823
C_{l_p}	-0.1646	C_{l_β}	-0.0132	C_{l_r}	0.00785
C_{M_p}	0.0	C_{M_β}	0.0	C_{M_r}	0.0
C_{N_p}	0.0067	C_{N_β}	0.05119	C_{N_r}	-0.03369

7 LOG BASED METHOD

This chapter describes the first method implemented to obtain the aerodynamics coefficients. The data is gathered inside logs through Mission Planner, and it's then imported into MATLAB thanks to the "ArduPilog.m" function. The data inside these logs is filtered, refined and organized.

Afterwards, the equations of motions are found and the raw coefficients are calculated.

A multivariate interpolation is performed thanks to the hypothesis of the superposition of effects. The final output are the linearized coefficients in the standard form.

ArduPilot provide data inside channels. Every channel has a different logging frequency which complicates the matter. Most of the data is obtained from the EKF. Multiple lines are calculated simultaneously so that if one fails, others can compensate. As soon as the failure is detected, the autopilot immediately switches onto another lane. Unfortunately, this passage is extremely risky and can lead to the loss of the drone.

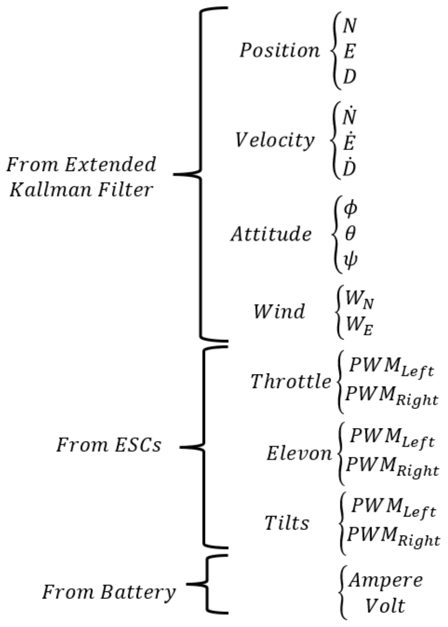


Figure 35: Log inputs channels

Wind Estimations are performed only on the North and East Components. Because of the low height at which these drones fly, vertical wind component is neglected.

Through this approximation the Wind Vector is calculated as follows:

7.1

$$\vec{V}_{Airspeed} - \vec{V}_{GroundSpeed} = \vec{W}_{wind}$$

7.1 General Description

The multivariate problem can be written in the following form:

7.2

$$z = VP + v$$

Where:

- z is the measurement;
- V is the matrix variable;
- P is the vector of parameters;
- v are the residuals.

The dimension of the matrix Z is defined by the N measurements considered to be valid.

The independent regressor variables chosen are $V = \{\alpha, \alpha^2, \beta, \delta_{ele}, \delta_{ail}, \tilde{p}, \tilde{q}, \tilde{r}\}$.

The velocity has not been included for two reasons:

a) The coefficients are non-dimensional therefore the velocity is not needed;

b) The Mach number is extremely low and no effects would be observed anyway.

The 21 model parameters, which will be discussed in depth later, are:

$$P = \{C_{D_0}, C_{L_0}, C_{M_0}, C_{L_\alpha}, C_{M_\alpha}, C_{D_\alpha}, C_{L_q}, C_{M_q}, C_{L_\delta}, C_{M_\delta} \dots\}$$

The cost function is defined in the following way:

7.3

$$J(\boldsymbol{P}) = \frac{1}{2}(z - \boldsymbol{VP})^T(z - \boldsymbol{VP})$$

The equation which minimizes the cost is given by:

7.4

$$\boldsymbol{P} = (\boldsymbol{V}^T\boldsymbol{V})^{-1}(\boldsymbol{V}^T z)$$

In the end the coefficient of determination $R^2$is calculated:

7.5

$$R^2 = 1 - \frac{RSS}{TSS}$$

Where RSS is the sum of the squares of the residuals υ and TSS is the total sum of squares.

At a specific timeframe, let's name the modelled value, f_i, the actual value y_i and the error between the two e_i:

7.6

$$RSS = \sum_N \upsilon_i^2 = \sum_N (y_i - f_i)^2$$

Also defining the mean of the calculated data $\bar{y}_i$:

7.7

$$\bar{y}_i = \frac{1}{N}\sum_{i=1}^{N} y_i$$

It's possible to define the total sum of square:

7.8

$$TSS = \sum_{i=1}^{N} (y_i - \bar{y}_i)^2$$

7.2 METHOD'S LIMITS

Given the fact that this type of drone has very fast dynamics, the method chosen has insurmountable limit created by the sensors. As described previously, they all suffer from disturbances, noise, biases or other problems. Even after multiple filters are applied, the data is still very noisy, especially pitot's data. This issue directly translates into having noisy coefficients with large errors. As will be described at the end of the next chapter, the results are not completely reliable and more sources are needed. The other limit of the method comes from the mathematics involved: "There are two important problems with using ordinary least squares to estimate a generic aerodynamic model for aircraft. (..) the model structure must be known. Sometimes prior knowledge can be used, other times step-wise regression can determine model structures using a variety of statistical metrics. However, the model structure problem must be determined iteratively until a solution is deemed sufficient. Another problem is that regressors typically have some level of correlation, either from feedback control, small ranges in which variables are similar, or simply the motion of the aircraft. In these cases, the least-squares estimator cannot correctly attribute variation to the correct regressor and the XT X matrix becomes ill-conditioned. These problems make using ordinary least squares difficult for identifying and estimating generic aircraft models from data." [13]

7.3 FLIGHT CAMPAIGN

The first thing to do was to gather data regarding the flight behaviour of the drone. Four days of flight tests have been conducted aiming to gather data regarding every possible state of the drone. Unfortunately, most of the days were windy and one was too dangerous to fly. Even though, more than 20 valid logs have been collected and all the results obtained later come from this data.

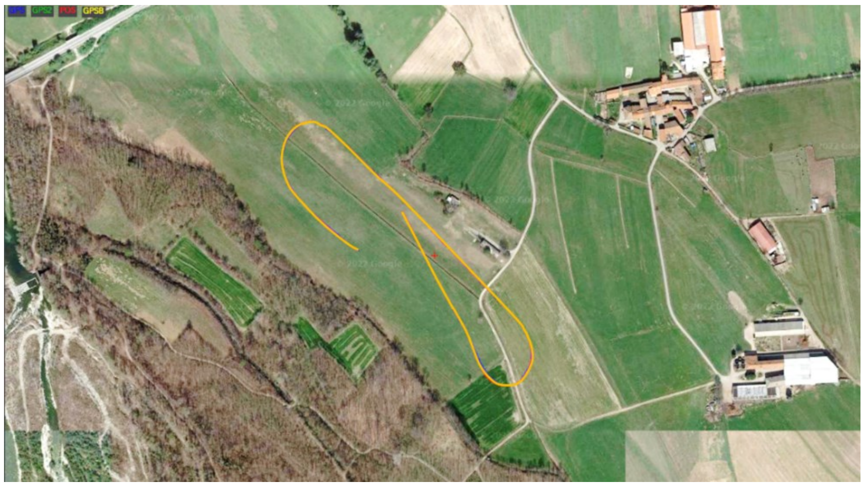


Figure 36: Log sample

The figure above represents a sample of what has been obtained.
The pilot had been instructed to fly the drone in extreme behaviours which aren't possible to achieve for the standard autopilot's modes. This, of course, includes the fact of having not perfect trajectories and height oscillations.
After, the filters, which will be described later, have been applied. The number of time frames considered to be valid is roughly:

$$N \approx 3000$$

Given the fact that the time frames have been interpolated at a frequency of 0.1s, this number of frames results in roughly 5 minutes.
This means that around 10% of the total flight time has been analysed.
There are multiple reasons for this:

1) For a timeframe to be valid, all the channels needed to not to skip or lag in that particular instant. Logging is a low priority activity for the autopilot;
2) ArduPilot's start to log as soon as the drone is armed creating many useless timeframes;
3) The timeframes where the drone's is landing have been skipped;
4) The timeframes where the drone's is flying very close to the ground have been skipped;
5) The timeframes where the drone's accelerometer clip have been skipped.

7.4 INERTIAL AND GEOMETRIC DATA

The centroid of the drone is an extremely point that defines multiple aspects of the aircraft's dynamics. At the time that the drone was first provided, no mass related data was provided.
Therefore, the first thing done was to correctly setup its CAD mass properties. Each component of the drone was weighted, and its own barycentre estimated.
Different ways to estimate both the barycentre and inertias were used based on the type of data available. Datasheet were the first option whenever the mass properties of the component were provided. In most cases they had to be estimated.
The path chosen was to give homogeneous density to many components since no extreme mass distribution was observed. From the CAD, multiples data have been obtained.
Various distances between different part of the structure which will be needed later in the method or in the simulation to correctly estimate of model the drone's behaviour.
The mass was, of course, obtained as the sum of the component's mass. Two different weights were used because of the different masses of the components: the more accurate one had 0.1 g sensitivity and 200 g maximum load, the second one with 1.0 g sensitivity and 500 g maximum load. The most difficult part to calculate were the mass distribution in the wings and the

fuselage, since they were 3D printed and the thickness of the print depends on multiple factors such as: the slicing, printing pattern, infill, material used and others.

This problem was sorted taking apart both parts of the wings from the fuselage, weighting them and calculating their centre of gravity. To calculate the centre of gravity the simplest possible method, yet extremely effective, was used: balancing the object on a thin line. Doing this operation twice, on both directions of the wing-plane, gave a more accurate result than balancing the airframe parts on a single point both because of the ease and because of the consistency of the results.

The hardest thing to calculate was the Z-position of the centre of gravity. Given its strong dependence on the 3D printing technique used (which was not an available information) and given the impossibility of balancing the components in the correct way, a completely different path has been selected. The Z-Position of the centre of gravity was estimated giving to the components a linear thickness distribution depending on the overall thickness. This was especially important for the wings since the fuselage was almost completely printed using the same thickness. The X and Y positions of the centre of gravity of these airframe parts were ignored since more accurate results were obtained through the balance process although they were used to check the consistency of the previous process.

This process was the least accurate one and therefore the Z position of the centre of gravity has the biggest relative error. Fortunately, given the small ratio between wing chord and fuselage thickness, the error has been estimated to be only +/- 5mm.

The most important effect of having an inaccurate estimation of the Z position of the centre of gravity is on the torque created by the Thrust given by the propellers. Since there's a distance along the Z-axis between the Thrust and the centre of gravity, the overall pitching moment of the aircraft is dependent on it. Even a small error such as the one found can be extremely important when the Throttle is raised. Notice that the distance of the C.o.G is intended to by the distance between the C.o.G. and the bow.

The results obtained are listed below.

Table 8: Inertial Properties

INERTIAL PROPERTIES	
Mass $[kg]$	4.35
$C.o\,G.[m]$	0.36
$I_X\ [kg \cdot m^2]$	0.02
$I_Y\ [kg \cdot m^2]$	0.036

$I_Z\ [kg \cdot m^2]$	0.053
$Blades'Inertia\ [kg \cdot m^2]$	10^3
$Wingspan\ [m^2]$	1.26
$Wing\ chord\ [m]$	0.37

7.5 Rotational Matrices

Multiple rotational matrices are needed in order to perform this calculation. Since the equation of motion will be calculated in the body frame, every force and torque need to be rotated inside it. The forces that need to be rotated from the wind axis to the body axis are lift, drag and deviance. The following matrix rotates from body to wind:

$$R_{wb} = \begin{bmatrix} \cos(\alpha)\cos(\beta) & \sin(\beta) & \sin(\alpha)\cos(\beta) \\ -\cos(\alpha)\sin(\beta) & \cos(\beta) & -\sin(\alpha)\sin(\beta) \\ -\sin(\alpha) & 0 & \cos(\alpha) \end{bmatrix}$$

Since every rotational matrix is orthogonal, the inverse corresponds to the transposed matrix:

$$R_{bw} = R_{wb}^{-1} = R_{wb}^{T} = \begin{bmatrix} \cos(\alpha)\cos(\beta) & -\cos(\alpha)\sin(\beta) & -\sin(\alpha) \\ \sin(\beta) & \cos(\beta) & 0 \\ \sin(\alpha)\cos(\beta) & -\sin(\alpha)\sin(\beta) & \cos(\alpha) \end{bmatrix}$$

This last is the matrix that has been used.

The aerodynamic angles α and β are taken from the log. The coefficients C_L, C_D and C_Y are the variables which must be calculated and will be therefore isolated inside the equation.

The second matrix needed is from NED to body frame:

$$\vec{V}_{NED} = R_{b\to v}\begin{pmatrix} u \\ v \\ w \end{pmatrix}$$

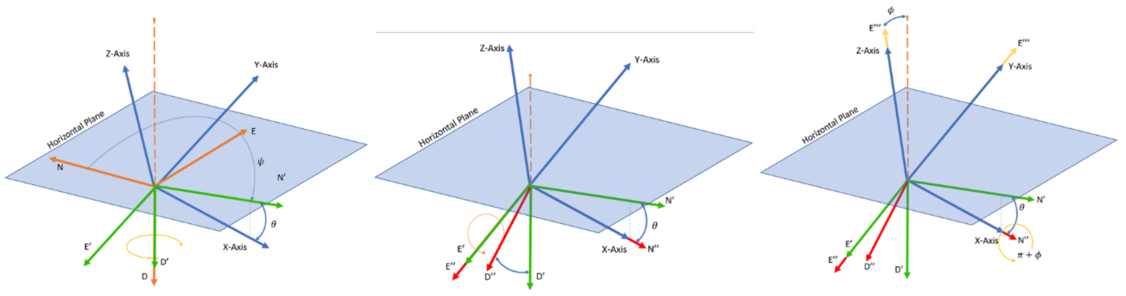


Figure 37: Rotation Matrix

The image shows the three rotations performed which result in the rotational matrix: [14]

Equation 9

$$R_{bn} = \begin{bmatrix} \cos(\theta)\cos(\psi) & \sin(\phi)\sin(\theta)\cos(\psi) - \cos(\phi)\sin(\psi) & \cos(\phi)\sin(\theta)\cos(\psi) + \sin(\phi)\sin(\psi) \\ \cos(\theta)\sin(\psi) & \sin(\phi)\sin(\theta)\sin(\psi) + \cos(\phi)\cos(\psi) & \cos(\phi)\sin(\theta)\sin(\psi) - \sin(\phi)\cos(\psi) \\ -\sin(\theta) & \sin(\phi)\cos(\theta) & \cos(\phi)\cos(\theta) \end{bmatrix}$$

The last matrix needs to perform the rotation between the tilt motor angle and the body frame. Since Strategy's motors can rotate around the Y_{Body}axis, only one rotation is needed:

Equation 10

$$R_{b \to P} = \begin{bmatrix} \cos(\delta_{Prop}) & 0 & \sin(\delta_{Prop}) \\ 0 & 1 & 0 \\ -\sin(\delta_{Prop}) & 0 & \cos(\delta_{Prop}) \end{bmatrix}$$

Where δ_{Prop} is the tilt angle which can be different from the left and right motors.

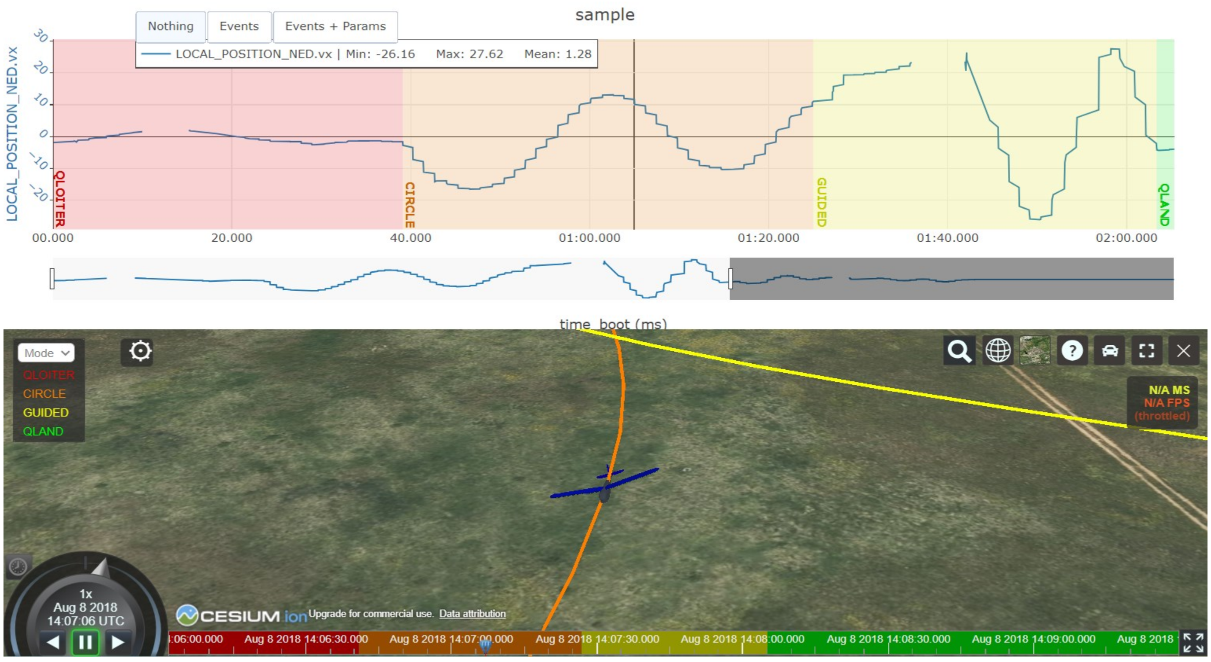


Figure 38: UAV log viewer [15]

These rotational matrices have been validated thanks to the usage of an online log viewer "UAV LOG Viewer":

To avoid gimbal lock, the method implemented by David Hosier was followed:

"Finally, the actual working method of avoiding gimbal lock is to form a direction cosine matrix using a position pointing vector instead of gravity. The pointing vector is simply the projectile's current position subtracted from a target's position. For scenarios that apply a lift force in a constant direction, the target should be a position in space in the direction of desired travel, preferably at a distance that is unattainably far away in order to avoid causing the velocity vector to be coincident with the position pointing vector." [16]

7.6 Equations of Motion: Forces

The first thing was to find the correct equation of motions that would describe the drone's behaviour. Vanity has only two types of controls which are Thrust and Elevon Surfaces, meanwhile Strategy has also the tiltable motors.

Therefore, the Strategy's equations of motion must include the possibility of the Thrust being directed in different directions. This can be easily achieved with a 2D Directional Cosine Matrix. The equations of motions have been calculated in the body axis.

Starting from a levelled, stationary flight, the main forces acting on the Drone are the following:

1. Aerodynamic Forces: Drag, Lift and Deviance.
2. Generated Forces: Thrust.
3. Inertial Forces.

Decomposing the Thrust force along the X_{Body} axis and the Z_{Body} axis can be done by the knowledge of the Tilt Servo Angle.

The aerodynamic forces, in wind axis are modelled as follows:

- Drag: $D = \frac{1}{2}\rho V^2 S C_D$
- Lift: $L = \frac{1}{2}\rho V^2 S C_L$

Finding the correct coefficients is the aim of this part of the work, but some coefficients can already be estimated as some parameters are known. Subsequently, the gravitational force has been decomposed into its components along the X, Y and Z_{Body} axis using the three Euler angles.

7.11

$$\vec{W} = R_{bn}\begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix}$$

The weight only acts on the third component and is positive as the D axis of the NED frame of reference is positive downward.

The Thrust has also been decomposed into three components where only one is non-zero.

7.12

$$\vec{T} = \begin{bmatrix} T \\ 0 \\ 0 \end{bmatrix}$$

The Matrix used to perform these calculations depends on the Tilts' Angles.

$$\vec{T}_{Body} = R_{bP}(\delta_{prop})\,\vec{T}$$

The matrix needs to be multiplied by both Thrusts, left and right, because they might differ, therefore:

7.13

$$\vec{T} = \vec{T}_{Body_{Left}} + \vec{T}_{Body_{Right}}$$

As previously explained, differential thrust is used to perform different Roll and Yaw manoeuvres, especially in *Copter* mode.

The model has been later refined to include the normal force generated when the aircraft is rotating. Note that the sign of the force is given by the sign of both pitching rate q and angle of attack α. [17]

7.14

$$P_N = \frac{q\sigma A}{2}\left\{\bar{C_l} + \frac{aJ}{2\pi}\cdot \ln\left[1+\left(\frac{\pi}{J}\right)^2\right] + \frac{\pi}{J}C_d\right\}\alpha$$

Where:

- A: is the propeller disc area: $A = \pi r_{blade}^2$;
- σ: is the propeller solidity, calculated as: $\frac{S_{blade}}{A}$;
- C_l, C_d: are the 2D lift and drag coefficient of the blade's airfoil;
- J: is the advance ratio, calculated with the propeller's perpendicular velocity $V_\perp$: $J = \frac{V_\perp}{\omega \cdot R_{blade}}$;
- a: is the derivative of the lift coefficient: $a = \frac{\partial C_l}{\partial \alpha}$

As the rest of the forces generated by the propeller, also this force needs to be rotated onto the body frame of reference therefore:

7.15

$$\vec{P}_{N\,body} = R_{bP}\begin{bmatrix} 0 \\ 0 \\ P_N \end{bmatrix}$$

The last component of the force equations are the inertial forces, as follows:

7.16

$$\vec{F}_{in} = m\left(\vec{w}\times\vec{V}\right) = m\begin{vmatrix} i & j & k \\ p & q & r \\ u & v & w \end{vmatrix} = m\begin{bmatrix} qw - rv \\ ur - pw \\ pv - qu \end{bmatrix}$$

Now the complete equations can be written:

7.17

$$m\vec{\ddot{X}} = \vec{F}_{aero} + \overrightarrow{W} + \vec{T} + \vec{F}_{in} + \vec{P}_N$$

7.7 Equations of Motion: Torques

The rotational equations can be written following an analogous procedure, but more terms compare inside them as there are more phenomena involved.

Again, the aerodynamic forces are modelled in the same way:

7.18

$$\vec{Q}_{aero} = \begin{bmatrix} \mathcal{L} \\ \mathcal{M} \\ \mathbb{N} \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\rho V^2 SbC_l \\ \frac{1}{2}\rho V^2 ScC_M \\ \frac{1}{2}\rho V^2 SbC_N \end{bmatrix}$$

Next, the torque caused by the Propellers' thrust is found. This torques are born because of the Thrust's acting axis does not correspond to the barycentre. The torques along the X and Y axis are caused by both differential Thrust and different tilt angles between the two propellers. The distances between the Thrust axis and the centre of gravity have been calculated in the previous chapters:

7.19

$$\vec{X}_{Th_{R/L}} = \begin{Bmatrix} X_T \\ \pm b \\ Z_T \end{Bmatrix}$$

The vector for both propellers is the same except for the Y component. Now the whole equation can be written:

7.20

$$\vec{Q}_{Thr} = \vec{X}_{Th_L} \times \vec{T}_{Body_{Left}} - \vec{X}_{Th_R} \times \left(\vec{T}_{Body_{Right}}\right)$$

The complexity of this equation derives from the tilts otherwise only differential thrust could create torques. In a nominal case, no torques are generated by the propellers apart from the Y axis since both propellers have a positive lever along the Z axis and a positive force resulting in a negative torque in the pitching axis.

The next contribution to the torque equation is the inertial torques:

7.21

$$\vec{Q}_{In} = \vec{w} \times (I\,\vec{w}) = \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \left(\begin{bmatrix} I_X & 0 & 0 \\ 0 & I_Y & 0 \\ 0 & 0 & I_Z \end{bmatrix} \begin{vmatrix} i & j & k \\ p & q & r \\ u & v & w \end{vmatrix} \right)$$

Since the Inertial Matrix is diagonal (due to Plane of Symmetry X-Z), all the contributions to the torque involving these quantities can be set to 0. This results in the following equation:

7.22

$$\vec{Q}_{In} = \begin{bmatrix} qr(I_Z - I_Y) \\ pr(I_Z - I_X) \\ qp(I_Y - I_X) \end{bmatrix}$$

The next contribution is caused by the rotation of the propellers. The drag created by the blades' rotation is unloaded onto the airframe. The left propellers rotates clockwise and the right one counterclockwise. This means that if the propellers rotational speed is the same, no torque is generated as they are opposite in directions and equal in magnitude.
This requirement is often not fulfilled, and a torque is present.
To calculate the torque generated by the propellers, the Blade Element Theory is used.
The torque generated by the propellers are the following:

7.23

$$\vec{Q}_r = K\rho w_r^2 r_b^5 \hat{J_r} \qquad \vec{Q}_l = -K\rho w_l^2 r_b^5 \hat{J_l}$$

Where:

- $w \left[\frac{rad}{s}\right]$ is the rotational speed of the propellers;
- r is the propellers' blades' length;
- K coefficient is calculated in the following chapters as many tests are required to find the right value;
- $\hat{J_\#}$ are the unit vectors indicating thrust's direction with respect to the body frame;
- $\rho \left[\frac{kg}{m^3}\right]$ is the air density which is measured using the following equation:

$$\rho = \rho_0 \left(1 - \frac{hz}{T_0}\right)^{m-1}$$

Where ρ_0 is the air density at sea level which is $1.225 \frac{kg}{m^3}$, z is the height, $h = 0.0065 \frac{K}{m}$ is the thermal gradient across the Troposphere and, finally, $m = 4.2561$ is an adimensional value that depends on the air molecular mass.
The Torque equations have opposite signs since the rotational speed is opposite. They both need to be rotated onto the body frame of reference:

7.24

$$\vec{Q}_{l/r_{body}} = R_{bP}\left(\delta_{prop_{\frac{l}{r}}}\right) \vec{Q}_{l/r}$$

The resulting Torques acting on the airframe is the difference between the two in case they are both rotating at the same speed. This contribution is extremely important as it determines, alone, the yaw capability of all the multirotor. For this specific drone is important as well in the *Copter* flight mode, although other control forces can be created by the elevons.
Later the model has been refined to include gyroscopic torques:

7.25

$$N_{gyro} = I_{prop} \cdot \dot{\omega} \cdot q$$

7.26

$$M_{gyro} = -I_{prop} \cdot \dot{\omega} \cdot r$$

Of course, these gyroscopic torques tend to be zero whenever the propellers are rotating at the same speed and in the same axis since the two rotations are opposite.
Torque due to the acceleration of the propeller:

7.27

$$Q_{accel} = I_{prop} \cdot \dot{\omega}$$

Please notice that all of these torques are calculated considering ω positive for the right-handed propeller, the signs are opposite for the left propeller.
A moment is generated by the propeller when the flow is not axial and therefore has a non/zero angle of attack:

7.28

$$N_P = -\frac{\sigma \cdot q \cdot A \cdot R_{Prop}}{2} \left\{ \frac{2 \cdot \pi}{3 \cdot J} \overline{C_l} + \frac{a}{2} \left[1 - \left(\frac{J}{\pi} \right)^2 \ln \left(1 + \left(\frac{\pi}{J} \right)^2 \right) \right] - \frac{\pi}{J} \cdot C_d \right\} \alpha$$

Moreover, two more torques are generated when the aircraft is rotating about the Pitch and Yaw axis. These two torques tend to dampen the rotation easing the pilot's effort to control the drone. They are generated because, due to the rotation, different angle of attack is found along the propeller's blade span resulting in a counter acting torque:

7.29

$$M = \frac{k_d \rho}{2} \omega^2 R^5 \arctan\left(\frac{q}{\omega}\right)$$

7.30

$$M = \frac{k_d \rho}{2} \omega^2 R^5 \arctan\left(\frac{r}{\omega}\right)$$

Where the parameter k_d can be approximated as follows:

$$k_d \approx 2\pi^2 \sigma$$

These two torques are synthesized in the following image:

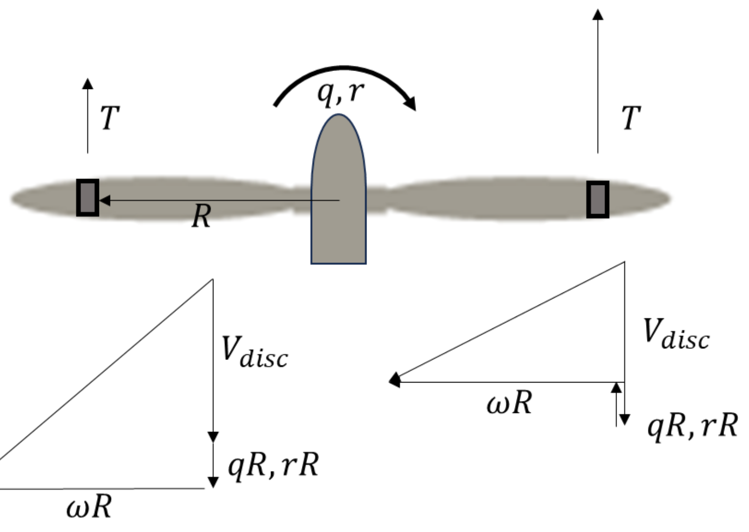


Figure 39: Velocity Triangles

As it can be seen, the velocity triangles change, due to the overall motion of the propeller. This results in a local change of speed which in turn creates local differences of advance ratio. Given the thrust's dependency to the advance ratio, a torque is generated.

Finally, the model has not considered the torques generated by the motion of the propeller's servo $\dot{\delta}_{Prop} = \frac{\partial \delta_{Prop}}{\partial t}$. The reason behind this choice is that δ_{Prop} is constant during plane mode and the tilt move very slowly during *Copter* mode resulting in:

$$\dot{\delta}_{Prop_{Plane}} = 0 \qquad \dot{\delta}_{Prop_{Copter}} \approx 0$$

7.8 Commands' Data Manipulation

The 6 aircraft's commands are transformed from PWM signals into their respective dimensions:

7.31

$$\begin{Bmatrix} RC_{out}\ 1\ [PWM] \\ RC_{out}\ 2\ [PWM] \\ RC_{out}\ 3\ [PWM] \\ RC_{out}\ 4\ [PWM] \\ RC_{out}\ 5\ [PWM] \\ RC_{out}\ 6\ [PWM] \end{Bmatrix} \rightarrow \begin{Bmatrix} rad \\ rad \\ rad \\ rad \\ \% \\ \% \end{Bmatrix} \rightarrow \begin{Bmatrix} \delta_{Elevon_{Right}} \\ \delta_{Elevon_{Left}} \\ \delta_{Tilt_{Right}} \\ \delta_{Tilt_{Left}} \\ \delta_{Motor_{Right}} \\ \delta_{Motor_{Left}} \end{Bmatrix} = \vec{u}$$

The two elevon deflections and the two Tilt Servo Angles have been measured at different PWM levels with a Protractor (from 1000 to 2000 with 100 PWM steps). Then, the input PWM from the log has been interpolated to achieve the correct angle value.

The throttle value is also present inside the log, but it has not been utilized as ArduPilot's manipulates the throttle with the usage of the parameter "THRUST_EXPO" which complicates the calculation to obtain the RPM. [18]

Unfortunately, Strategy's ESCs do not log RPM by themselves like many do. The only solution found is to interpolate the PWM value inside the log to obtain the corresponding RPM. The interpolating equation has been found during the bench tests.

Aileron and elevator deflections have been calculated in the following way:

7.32

$$\delta_{ail} = \delta_{Elevon_{Right}} - \delta_{Elevon_{Left}}$$

7.33

$$\delta_{ele} = \frac{\delta_{Elevon_{Right}} + \delta_{Elevon_{Left}}}{2}$$

7.9 TIME INTERPOLATION, LAGS AND CUTS

The data logging is not a high priority task inside the ArduPilot's firmware. During flight, it occurs that various instances are skipped to run higher priority tasks. This means that the logged data has holes. Also, the different channels that provide the data have different rates because they are calculated at frequencies depending on their importance. For example, gyroscopes provide data at much higher frequencies than the Kallman Filter does.

To correctly solve the equations of motion, all the variables need to be synched therefore an interpolation in time is performed. The data is interpolated every 0.1 seconds, and the interpolation depends on the channel. The initial time is set to zero so that it's easier to understand the time reference.

As previously mentioned, the log presents many lags that must be cut. To do that, the time difference between each value is calculated and when the period passes a certain threshold, the data is not considered in the subsequent calculations.

The logic implemented is that every time a lag is observed, the whole period related to the lag is discarded. This procedure is repeated separately for each channel since lags are observed at different times and frequencies.

7.10 COEFFICIENT CALCULATION

As previously explained, the equations of motion have been reversed at each time step.

The three translational equations respectively output the lift C_L, the drag C_D and the side force C_Y coefficients. Meanwhile, the three rotational equations along X, Y and Z output respectively C_L, C_M and C_N. Once these coefficients are calculated using the Principle of Superposition of Effects, the different contribution to each coefficient can be calculated.
Here's the logic implemented: [13]

7.34

$$\begin{bmatrix} m\dot{u} \\ m\dot{v} \\ m\dot{w} \\ I_X\dot{p} \\ I_Y\dot{q} \\ I_Z\dot{r} \end{bmatrix}_i = \begin{bmatrix} \sum F_X \\ \sum F_Y \\ \sum F_Z \\ \sum Q_X \\ \sum Q_Y \\ \sum Q_Z \end{bmatrix}_i \quad i \in [0, T_{final}]$$

Next the equations are inverted to isolate each aerodynamic coefficient inside each equation. For example, the first equation, at a specific time frame "i", becomes:

7.35

$$m\dot{u}_i = \vec{F}_{aero_i} \cdot \hat{\imath}_{body} + \vec{T}_\iota \cdot \hat{\imath}_{body} + \vec{F}_{in_i} \cdot \hat{\imath}_{body}$$

Where $\hat{\imath}_{body}$is the unit vector indicating that only the component along X_{Body} is taken.
Given the fact that:

7.36

$$\vec{F}_{aero_i} = f(C_{L_i})$$

This equation can be transformed into the following form, where the lift coefficient is the only unknown variable and can be therefore calculated.

7.37

$$C_{L_i} = f(\vec{T_\iota}, \vec{F}_{in_i}, m\dot{u}_i, \cdots)$$

7.38

$$\begin{bmatrix} C_D \\ C_Y \\ C_L \end{bmatrix} = \frac{1}{Q_i S} \begin{bmatrix} -\cos(\alpha) & 0 & -\sin(\alpha) \\ 0 & 1 & 0 \\ \sin(\alpha) & 0 & \cos(\alpha) \end{bmatrix} \begin{bmatrix} ma_X - (T_{X_L} + T_{X_R}) \\ ma_y \\ ma_Z - (T_{Z_L} + T_{Z_R}) \end{bmatrix}$$

Where:

- Q_i is the dynamic pressure modified considering the induced velocity;
- a_X, a_Y, a_Z: are the linear accelerations;

- $T_{X_{L/R}}$: is the thrust along X_{Body};
- $T_{Z_{L/R}}$: is the thrust along Z_{Body}.

7.39

$$\begin{bmatrix} C_l \\ C_m \\ C_n \end{bmatrix} = \frac{1}{Q_i S} \begin{bmatrix} \frac{1}{b} & 0 & 0 \\ 0 & \frac{1}{\bar{c}} & 0 \\ 0 & 0 & \frac{1}{b} \end{bmatrix} \left(\begin{bmatrix} I_X \dot{p} - I_y q r \\ I_y \dot{q} + (I_x - I_Z) p r \\ I_Z \dot{r} + \left(I_y - I_x\right) p q \end{bmatrix} + \begin{bmatrix} Q_{T_X} \\ Q_{T_Y} \\ Q_{T_Z} \end{bmatrix} \right)$$

Where:

- $\dot{p}, \dot{q}, \dot{r}$: are the rotational accelerations;
- p, q, r: are the rotational velocities;
- I_x, I_y, I_Z: are the inertias along the three main body axis;
- Q_T: is a vector containing all of the torques generated by the propeller (due to rotation, due to pich/yaw rate, gyroscopic, forces misalignment).

This process is then repeated for each time frame:

7.40

$$\begin{bmatrix} C_{L_1} \\ C_{L_2} \\ C_{L_3} \\ \vdots \\ C_{L_N} \end{bmatrix} = \begin{bmatrix} f\left(\overrightarrow{T_1}, \vec{F}_{in_1}, m\dot{u}_1, \cdots\right) \\ f\left(\overrightarrow{T_2}, \vec{F}_{in_2}, m\dot{u}_2, \cdots\right) \\ f\left(\overrightarrow{T_3}, \vec{F}_{in_3}, m\dot{u}_3, \cdots\right) \\ \vdots \\ f\left(\overrightarrow{T_N}, \vec{F}_{in_N}, m\dot{u}_N, \cdots\right) \end{bmatrix} = \begin{bmatrix} f_{A_1} \\ f_{A_2} \\ f_{A_3} \\ \vdots \\ f_{A_N} \end{bmatrix}$$

Where N is the number of time frame considered to be valid after the filters applied and “A” is to distinguish the X_{Body} equation from Y_{Body} (“B”), Z_{Body}(“B”), …

The same procedure is repeated for each of the six equations of motion. These results still need to be translated into common aerodynamic theory.

These coefficients can be seen as the sum of the effects of:

7.41

$$\begin{bmatrix} C_D \\ C_Y \\ C_L \\ C_l \\ C_M \\ C_N \end{bmatrix} = \begin{bmatrix} f(\alpha, \delta_{ele}) \\ f(\alpha, \beta) \\ f(\alpha, \tilde{q}, \delta_{ele}) \\ f(\alpha, \beta, \delta_{ail}, \tilde{p}, \tilde{r}) \\ f(\alpha, \delta_{ele}, \tilde{q}) \\ f(\beta, \delta_{ail}, \tilde{p}, \tilde{r}) \end{bmatrix}$$

From the previous section, the right-hand side of the equation is a known value for each timeframe. Applying the principle of superposition of effects:

7.42

$$\begin{bmatrix} C_D \\ C_Y \\ C_L \\ C_l \\ C_M \\ C_N \end{bmatrix} = \begin{bmatrix} C_{D_0} + C_{D_\alpha}\alpha^2 + C_{D_\delta}\delta_{ele} \\ C_{Y_\beta}\beta + C_{Y_p}\tilde{p} + C_{Y_r}\tilde{r} \\ C_{L_\alpha}\alpha + C_{L_q}\tilde{q} + C_{L_\delta}\delta_{ele} \\ C_{l_p}\tilde{p} + C_{l_\beta}\beta + C_{l_r}\tilde{r} + C_{l_{\delta_a}}\delta_{ail} \\ C_{M_0} + C_{M_q}\tilde{q} + C_{M_\alpha}\alpha + C_{M_\delta}\delta_{ele} \\ C_{N_\beta}\beta + C_{N_p}\tilde{p} + C_{N_r}\tilde{r} + C_N\delta_{ail} \end{bmatrix}$$

For each coefficient, the corresponding variables have been gathered inside a vector to perform the minimum square interpolation. A total of 21 terms have been inserted inside the nonlinear model. Future improvements can be made increasing the number of this coefficients to capture aircraft's non linearities such as stall. [13]

Here's an example of the calculation performed for the first equation of motion:

7.43

$$A = \begin{array}{c} \begin{bmatrix} f_{A_{1_1}} & f_{A_{2_1}} & f_{A_{3_1}} & \cdots & f_{A_{M_1}} \\ f_{A_{1_2}} & f_{A_{2_2}} & f_{A_{3_2}} & \cdots & f_{A_{M_2}} \\ f_{A_{1_3}} & f_{A_{2_3}} & f_{A_{3_3}} & \cdots & f_{A_{M_3}} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ f_{A_{1_N}} & f_{A_{2_N}} & f_{A_{2_N}} & \cdots & f_{A_{M_N}} \end{bmatrix} \updownarrow N\ timeframes \\ \longleftrightarrow \\ M\ variables \end{array}$$

Each component of the matrix has three subscripts which indicate the following:

$$f_{A_{1_2}}$$

- f: stands for the value of the equation of motion;
- A: stands for the first equation of motion ($X_{Body}, X_{Body}, X_{Body}$ translational and the subsequent 3 rotational);
- 1: stands for the first of the M superposed effects for that coefficient;
- 2: stands for the second of the N timeframes.

Given the fact that C_D can be written as: $C_{D_0} + C_{D_\alpha}\alpha^2 + C_{D_\delta}\delta_{ele}$ it has $m = 3$ and the A matrix is populated as follows:

7.44

$$A = \begin{bmatrix} 1 & \alpha_1^2 & \delta_{ele_1} \\ 1 & \alpha_2^2 & \delta_{ele_2} \\ 1 & \alpha_3^2 & \delta_{ele_3} \\ \vdots & \vdots & \vdots \\ 1 & \alpha_N^2 & \delta_{ele_N} \end{bmatrix}$$

Where α_1^2 is the square of the angle of attack at the first considered timeframe, α_2^2 is the same at the second timeframe considered and so forth.

Naming the vector of calculated drag coefficients at each timeframe $\overrightarrow{C_D}$:

7.45

$$\overrightarrow{C_D} = \begin{bmatrix} C_{D_1} \\ C_{D_2} \\ C_{D_3} \\ \vdots \\ C_{D_N} \end{bmatrix}$$

The following equation has been calculated, which minimizes the cost function:

7.46

$$\begin{bmatrix} C_{D_0} \\ C_{D_\alpha} \\ C_{D_\delta} \end{bmatrix} = (A^T A)^{-1}\left(A^T\, \overrightarrow{C_D}\right)$$

The dimension of each matrix is shown:

7.47

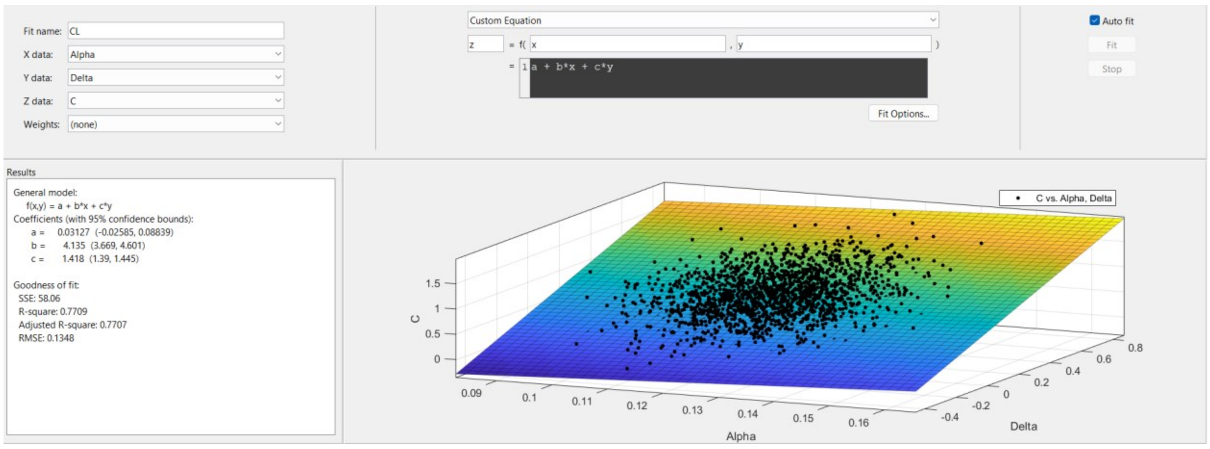
$$\underbrace{(\overset{M\times N}{A^T}\,\overset{N\times M}{A})}_{M\times M}{}^{-1}\overbrace{\left(\underset{M\times N}{A^T}\,\underset{N\times 1}{\overrightarrow{C_D}}\right)}^{M\times 1}$$

The same procedure is repeated for each of the six equations of motion.

This process corresponds to a multivariate interpolation and it can be extremely effective if all of the forces and torques acting on the system can be correctly modelled.

For each coefficient, the coefficient of determination R^2 has been calculated.

The following image represents a sample of the obtained data:

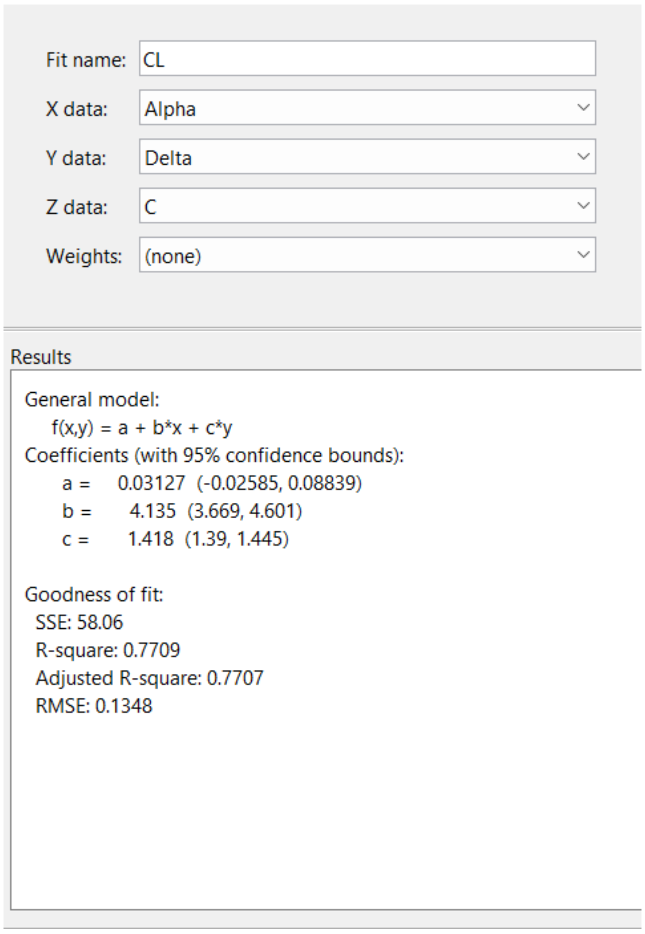

Fit name: CL
X data: Alpha
Y data: Delta
Z data: C
Weights: (none)

Custom Equation
z = f(x , y)
= a + b*x + c*y

Results

General model:
  f(x,y) = a + b*x + c*y
Coefficients (with 95% confidence bounds):
  a = 0.03127 (-0.02585, 0.08839)
  b = 4.135 (3.669, 4.601)
  c = 1.418 (1.39, 1.445)

Goodness of fit:
  SSE: 58.06
  R-square: 0.7709
  Adjusted R-square: 0.7707
  RMSE: 0.1348


Fit name: CL
X data: Alpha
Y data: Delta
Z data: C
Weights: (none)

Results

General model:
  f(x,y) = a + b*x + c*y
Coefficients (with 95% confidence bounds):
  a = 0.03127 (-0.02585, 0.08839)
  b = 4.135 (3.669, 4.601)
  c = 1.418 (1.39, 1.445)

Goodness of fit:
  SSE: 58.06
  R-square: 0.7709
  Adjusted R-square: 0.7707
  RMSE: 0.1348

Figure 40: Curve Fitting Toolbox results

As it's possible to see, it was used the curve fitting toolbox of MATLAB to obtain the image. The R^2 value is 0.7709 whereas the $R^2_{adj} = 0.7707$. The toolbox does not allow user to perform interpolation with more than two variables therefore only the angle of attack and the elevon deflection were used inside the graph. Also, for the same reason, R^2 value is lower than the actual result obtained interpolating also with $\hat{q}$. Again, for the same reason the coefficients are slightly different than the final result. [19]

7.11 Results

As previously mentioned, the data was filtered because of lags inside the logs. Additional filters were implemented to achieve a better result and discard unreliable instants.

The logic implemented was the following: for a specific instant to be considered valid, the derivative taken across 0.3 s, with at the centre the wanted data point, can't be more than a certain value.

Also, the oscillation of all of the variables involved must not pass the $\pm 25\%$ threshold. The following coefficients are the results:

$$C_L = C_{L_0} + C_{L_\alpha}\alpha + C_{L_q}\hat{q} + C_{L_\delta}\delta_{ele}$$

$$C_{L_0} = -0.0013 \quad C_{L_\alpha} = 4.3 \quad C_{L_q} = 3.55 \quad C_{L_\delta} = 1.45$$

$$C_D = C_{D_0} + C_{D_\alpha}\alpha^2$$

$$C_{D_0} = 0.027 \qquad C_{D_\alpha} = 0.00126$$

$$C_M = C_{M_0} + C_{M_\alpha}\alpha + C_{M_q}\hat{q} + C_{M_\delta}\delta_{ele}$$

$$C_{M_0} = 0.0208 \quad C_{M_\alpha} = -10^{-5} \quad C_{M_q} = -1.0484 \quad C_{M_\delta} = -0.012$$

Most of these coefficients seemed reasonable, compared to AVL e Datcom's results although the square error was very high implying that many points suffered from some kind of modelling error or sensor noise. A lot of data points were discarded because of the high oscillation speed of the elevon PWM signals.

In the following are reported the graph of the lift coefficients obtained so far.

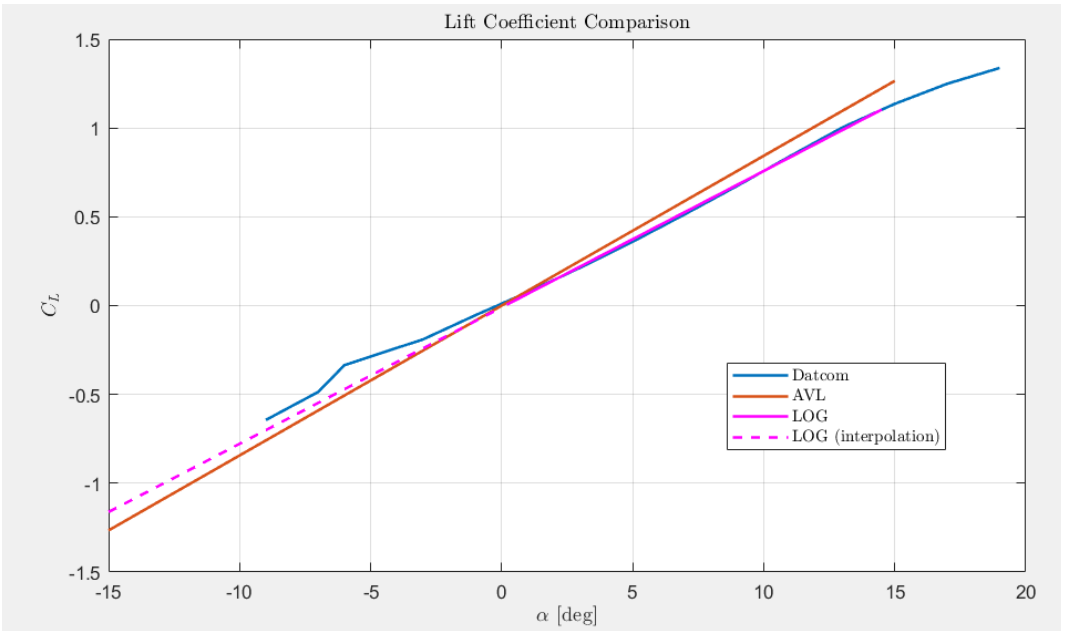


Figure 41: Lift Coefficient Comparison

As it can be seen, DATCOM and LOG-based C_L are very similar to each other. AVL estimates a lift slope slightly higher.

After refining the calculation with the slipstream model, the same script gave different results for the drag coefficient: the drag coefficient at 0° AOA went from 0.045 to 0.027. The other results decrease by less than 10% of their original values. The slipstream addition modified the average velocity. Dividing the wing surface in two separate regions in the following way:

7.48

$$S = S_c + 2 \cdot S_{\frac{w}{2}} = S_c + S_w \qquad V_{AVG} = \frac{V_\infty \cdot S_c + V_{Slip} \cdot S_w}{S}$$

Where:

V_∞: is the aircraft velocity; V_{Slip}: is the average velocity inside the slipstream;

S_c: is the surface unaffected; S_w: is the surface affected by the slipstream;

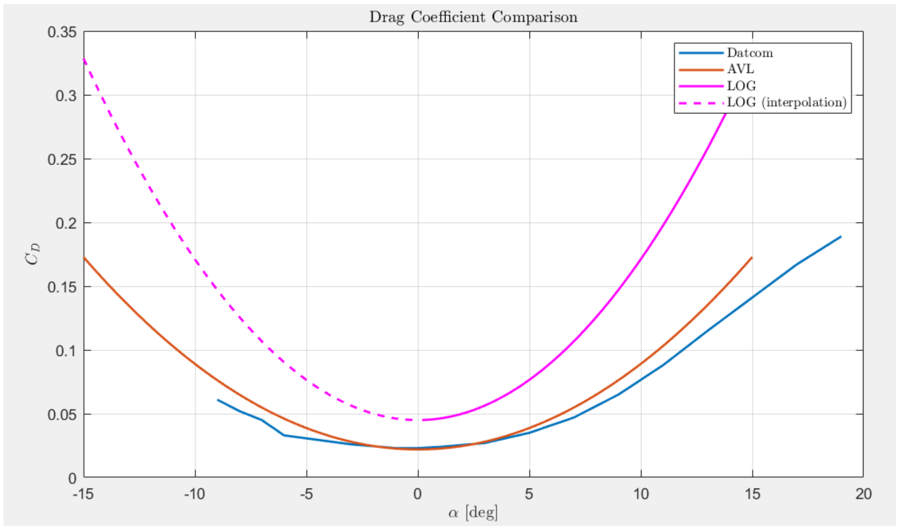


Figure 42: Drag Coefficient Comparison

As it can be observed, AVL and Datcom give very close results for the drag coefficient at 0 AOA. LOG values are still very unreliable since the mean square error didn't lower much after the model refinement.

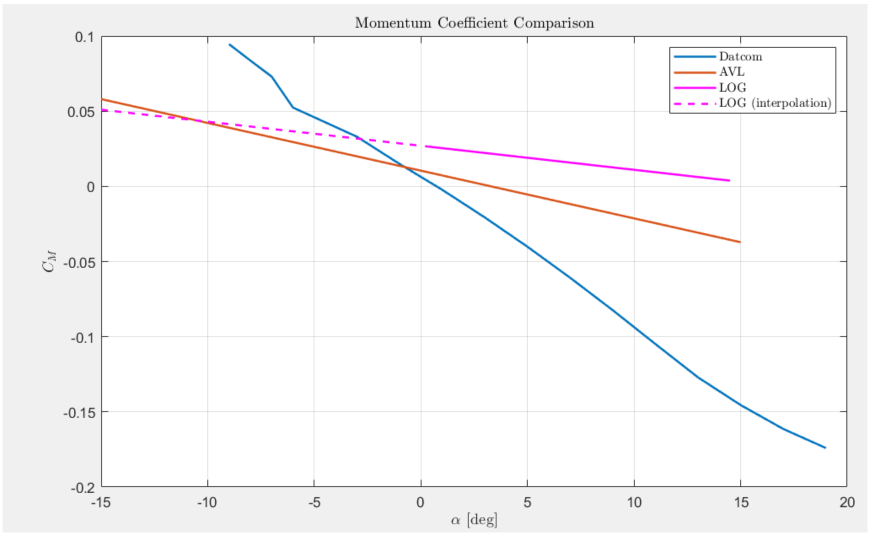


Figure 43: Momentum Coefficient Comparison

The graph shows the last main longitudinal coefficient which unfortunately gave very poor results. The data obtained was very noisy all the three methods gave very different results.
The log-based method although gave surprisingly consistent results for the rest of the longitudinal coefficients since the values didn't change after the model refinement.
Here's a summary of the results:

Table 9: Comparison between used methods, Lift Coefficient

C_L	C_{L_0}	C_{L_α}	C_{L_q}	C_{L_δ}	R^2	R^2_{adj}
DATCOM	0.011	4.06	2.82	1.38	0.822	0.818
AVL	-0.006	4.63	2.3	0.97		
LOG	0.0013	4.3	3.15	1.45		
2D	0.0477	5.92	N/C	N/C		

Table 10: Comparison between used methods, Drag Coefficient

C_D	C_{D_0}	C_{D_α}	R^2	R^2_{adj}
DATCOM	0.02295	0.000491	0.763	0.774
AVL	0.0217	0.00067		
LOG	0.045	0.00126		
2D	0.00765	0.000623		

CM	CM0	CMalpha	CMq	CMdelta	MSE
DATCOM	0.0063	-0.4411	-0.8456	-0.0089	
AVL	0.01043	-0.24186	-0.9531	-0.00774	
LOG	0.027	-0.0016	-1.0484	-0.012	
2D	0.009	-0.0031	N/C	N/C	

Where N/C stands for "Not Calculated".
MSE and R^2 values refer only to the raw data obtained from the logs of that coefficient.
MSE is calculated before performing the multivariate interpolation.
This method must be analysed further since the number of parameters can be increased and R^2 depends on it but also must be studied whether the correct regression is used, the set of variables is appropriate, the collinearity and so on. [20]

8 2D Fluid Dynamics Method

As the first attempt to obtain aerodynamic coefficients across the whole region of available angle of attack failed because of the highly disturbed data available, the focus shifted onto CFD simulation. As 3D simulations of the whole aircraft would take too much computational effort as well as a great deal of time to prepare the correct mesh, another way has been followed to obtain them. The software used to perform this calculation is Ansys as it provides a free student version with some limitations. Since last year, the number of cells available inside a simulation increased from 500.000 to 1 million. [21]

8.1 Proposed Method

The proposed method follows a simple though effective procedure to evaluate some of the main aerodynamic coefficients. Given that the purpose of this thesis is to create a simulator to study the behaviour of the fixed wing *VTOL* during its most important manoeuvres, only the longitudinal coefficients where investigated.

To correctly simulate the transition between horizontal and vertical flight (and vice versa), a wide range of angles of attack are attainable during it. At the same time, the *Copter* flight mode must be simulated and therefore extremely high angles of attack (basically blunt body) coefficients must be evaluated.

The method simplifies the complexity of the 3D problem into an integration of the 2D airfoil data across the wingspan. In this aircraft, this method is particularly effective as the fuselage is itself shaped as the airfoil and therefore the same principles can be applied.

The longitudinal coefficients needed for the transition simulation are calculated in the following ways:

$$C_L = \frac{2\int_0^{\frac{b}{2}} C_l(y)c(y)dy}{S}$$

$$C_D = \frac{2\int_0^{\frac{b}{2}} C_d(y)c(y)dy}{S}$$

$$C_M = \frac{\int_0^{\frac{b}{2}} C_m(y)c(y)dy + \int_{-\frac{b}{2}}^{0} C_m(y)c(y)dy}{S}$$

These equations lead to necessity of calculating each of these coefficients for every distance value from the symmetry plane (y). Fortunately, this calculation can be simplified further dividing the wing into different parts. Each part has its own domain and therefore is subjected to different laws.

The aircraft has been divided into the following sections:

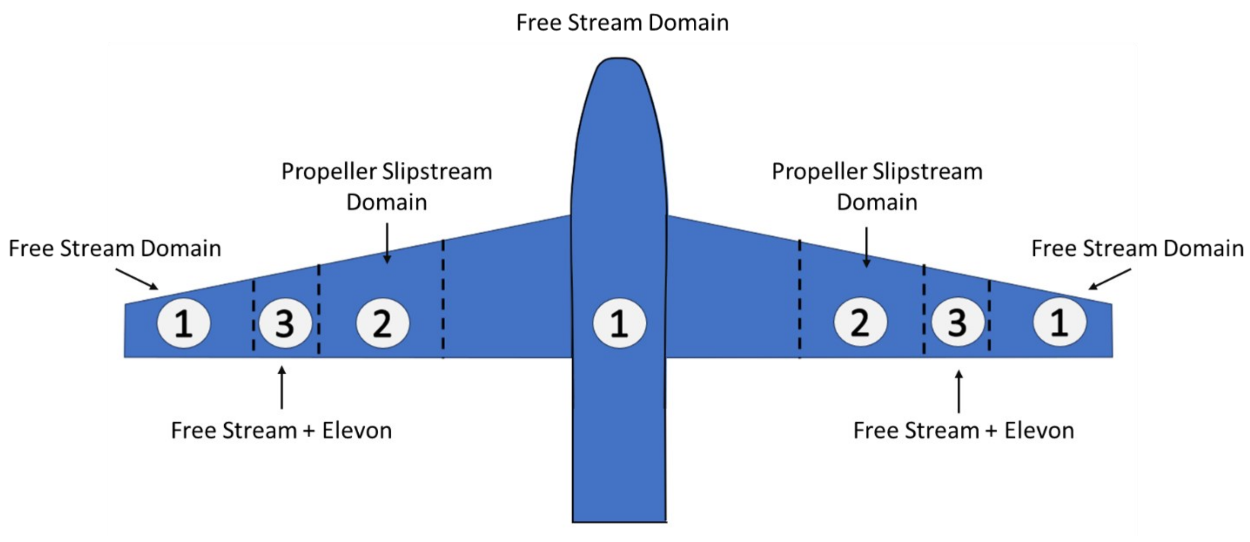


Figure 44: Aerodynamic Surface Domains

The domains will now be described separately.

8.1.1 Free Stream Domain

The fuselage, the wing hub and the wing tips are not subjected to the propeller slipstream and no movable surface is present. Given the components of the free stream:

$$\vec{V}_\infty = (U_\infty, V_\infty, W_\infty)$$

The equations governing this section are:

$$\alpha = \operatorname{atan}\left(\frac{w}{u}\right) \qquad u = U_\infty \qquad w = W_\infty$$

The coefficients are calculated as a simple integral of the surface present and only depend on the angle of attack:

$$C_L = f(\alpha) \qquad C_D = f(\alpha) \qquad C_M = f(\alpha)$$

The distribution of the chord across the span is obtained from the CAD model.

8.1.2 Free Stream and movable surface

In this section, the wing has a movable surface which affects the overall coefficients.

In this section the coefficients can be calculated as a function of both Angle of attack and deflection of the elevon. Furthermore, the percentage of the elevon's chords $\left(\frac{c_{ele}}{c(y)}\right)$ with respect to the overall chord, changes along the span and the resulting coefficients change as well. It can be written that:

$$C_L = f\left(\alpha, \delta, \frac{c_{ele}}{c(y)}\right) \qquad C_D = f\left(\alpha, \delta, \frac{c_{ele}}{c(y)}\right) \qquad C_M = f\left(\alpha, \delta, \frac{c_{ele}}{c(y)}\right)$$

Where δ is the elevon deflection and $\frac{c_{ele}}{c(y)}$ is the ratio between the movable chord and the overall chord. The data has been gathered without taking into consideration the effect of the elevon deflection therefore it can be said that: $\frac{c_{ele}}{c(y)} = 0 \ \forall y$.

8.1.3 Slipstream Domain

The slipstream domain requires much more equations to be represented properly as multiple physics phenomena take place in this sector. Given the extreme complexity of this sector, some assumptions were made:

1) The wing has no effect on the propeller Slipstream.
2) The Slipstream is axisymmetric.
3) The Slipstream cease to evolve as the leading edge of the wing is encountered.
4) The Slipstream is a laminar flow.

These assumptions were made to simplify the gathering of CFD data. Now, each of the main propeller's effects will be modelled.

8.1.3.1 Near Field Slipstream

Firstly, the increase in speed in taken into consideration.

From the equation in chapter 6, the Slipstream Radius Evolution can be found:

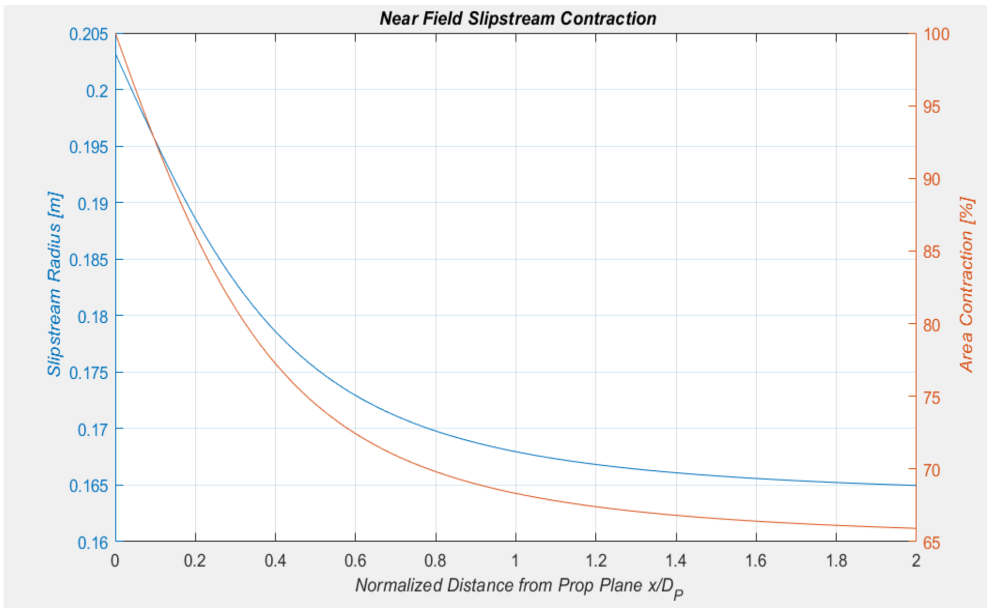


Figure 45: Slipstream Radius Evolution

The diameter of the propeller is 0.205m whereas the distance between the propeller plane and the wing's leading edge corresponds to roughly 30% of the diameter. As can be seen from the graph the slipstream evolution occurs mainly in the first diameter. The cross section of the slipstream contracts to around 80% of the original section. This contraction is accompanied by an acceleration of the flow which is graphed below:

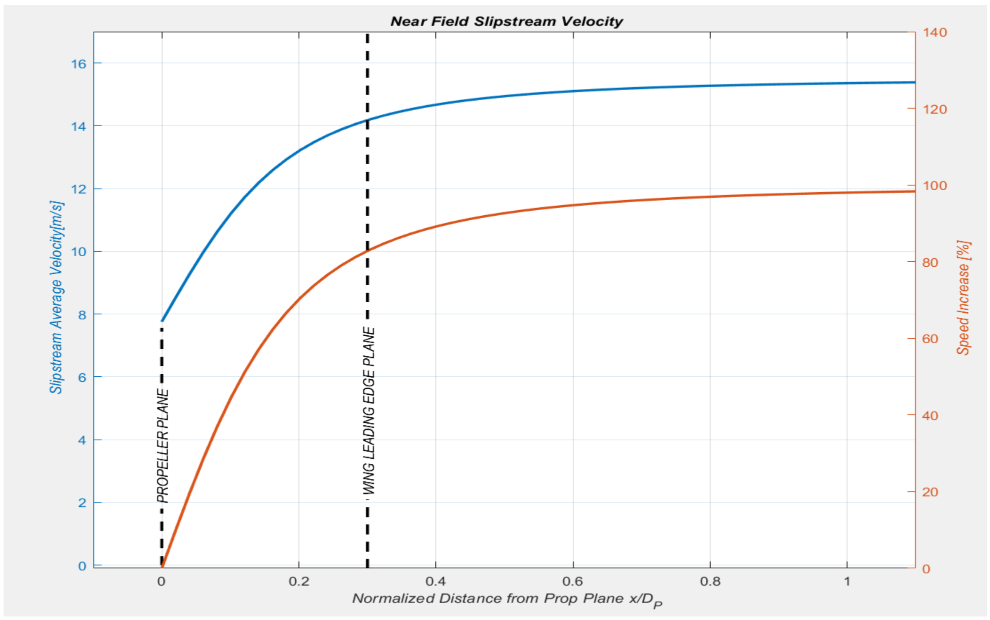


Figure 46: Slipstrem Axial Velocity Evolution

The graphs are taken into a particular flight condition which corresponds to 20 m/s and 6000 RPM. As it can be seen, a lot of the speed gain happens before encountering the leading edge of the propeller. The Far field region is not taken into consideration as the flow, after encountering the wing, is governed by the airfoil geometry. This increase in speed is although distributed across the minimum and maximum radius of the blade and therefore the speed changes greatly with the span. Since the acceleration occurs in the same manner behind the propeller, only the average speed is taken into consideration. This simplification brings a small error in the calculation of the Yaw and Roll Aileron's effects but since the scope of this thesis is the longitudinal behaviour, this small approximation is negligible. The effect of the servo motor tilting the propeller's plane pitch up and down is treated with the following hypothesis:

a) The flow always encounters the leading edge first and then stops following the near field or far field laws and starts being governed by the airfoil geometry.
b) The distance between the propeller's plane and the leading edge doesn't change.

Here's a schematization of the consequences of the approach chosen. The flow region affected by the propwash doesn't change in size if the angle of the propeller changes. This is of course a big approximation but taking into consideration this change might increase the complexity of the problem several times.

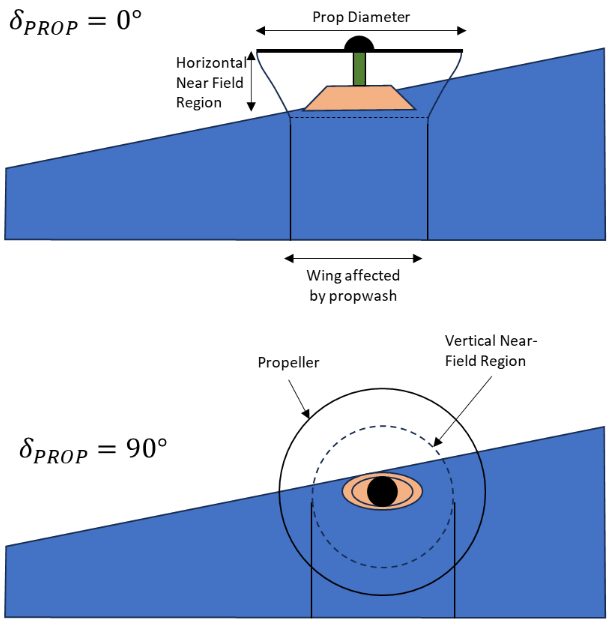

$\delta_{PROP} = 0°$
Prop Diameter
Horizontal Near Field Region
Wing affected by propwash
Propeller
Vertical Near-Field Region
$\delta_{PROP} = 90°$

Figure 47: Propeller Tilt

Only the average speed will be considered. For this reason, the governing equations are the following:

Equation 49

$$|V| = \sqrt{\left(u + V_{avg}\cos\left(\delta_{Prop}\right)\right)^2 + \left(w + V_{avg}\sin\left(\delta_{Prop}\right)\right)^2}$$

Equation 50

$$\alpha = \operatorname{atan}\left(\frac{w + V_{avg}\, sin\left(\delta_{Prop}\right)}{u + V_{avg}\cos\left(\delta_{Prop}\right)}\right)$$

8.2 CFD: Mesh Creation

To validate the structure of the mesh, three different meshes have been created for verification of the results.

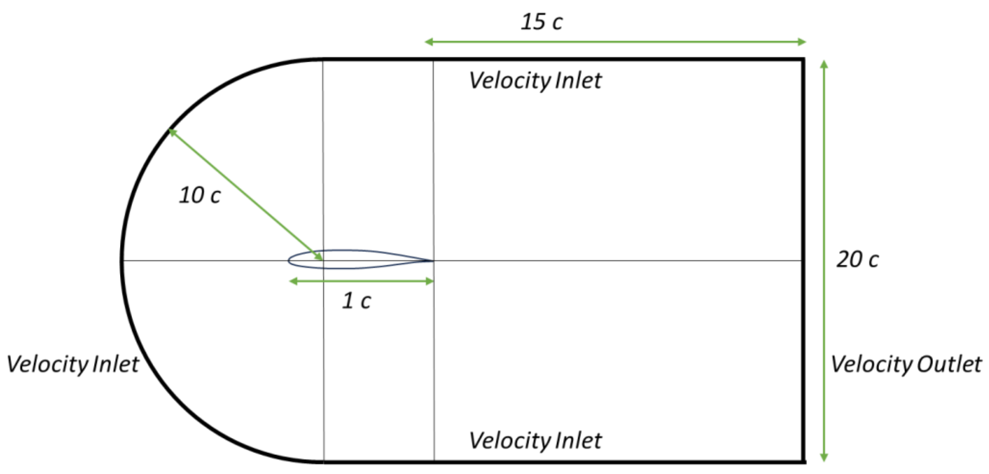


Figure 48: CFD Domain dimensions

As can be seen the centre of the semi-circle is not on the leading edge but rather on the first quarter of the chords. This modification brings better results since the radial lines arrive perpendicular to the profile and also increase in density. The mesh geometry is the same, the only thing changing is the number of subdivisions inside the domain, ordered by increasing number of cells: Coarse, Normal, Refined.

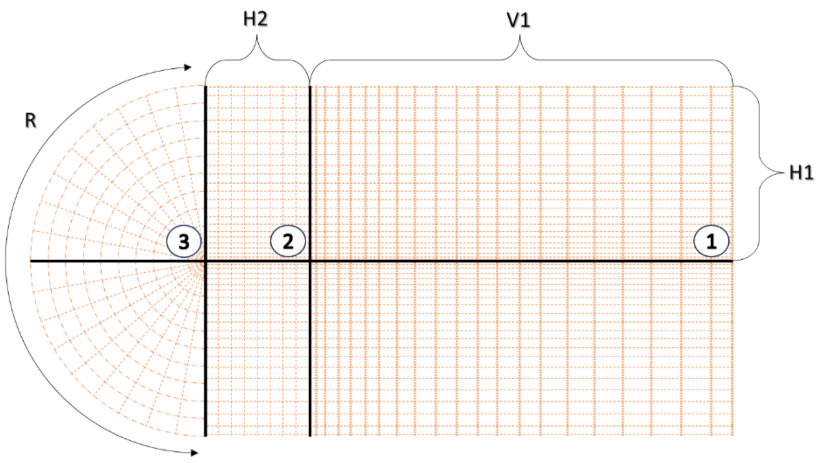


Figure 49: CFD Edges Subdivisions

The mesh is divided into 3 sections which are symmetrical with respect to the X axis. Each length shown in the image is divided into a defined number with a particular growth rate defined by the bias factor. The bias factors are equal for all meshes. Although the size of the first cell grows when decreasing the number of divisions, it has been used a conservative value to maintain the Y+ value below or equal to one. All of them are structured with biases in order to obtain the correct size of the first cell. The meshes have been created with the following subdivisions:

Table 11: Meshes' edges subdivisions

SUBDIVISIONS	Coarse	Intermediate	Refined	Bias Factor
Horizontal	280	400	570	1.20E+05
Vertical 1	300	420	600	3.00E+02
Vertical 2	275	390	550	0E+00
Radial	300	420	600	0E+00

The following table shows the total amount of cells present in each sector of the three meshes:

Table 12: Meshes' number of cells

TOTAL CELLS	Coarse	Intermediate	Refined
Section 1	84 K	168 K	342 K
Section 2	77 K	156 K	314 K
Section 3	84 K	168 K	342 K
TOTAL	245 K	492 K	998 K

As it's possible to see, the three meshes have a ratio between the total cells number of roughly 2 and consequentially their subdivisions have a ratio of $\sqrt{2}$.

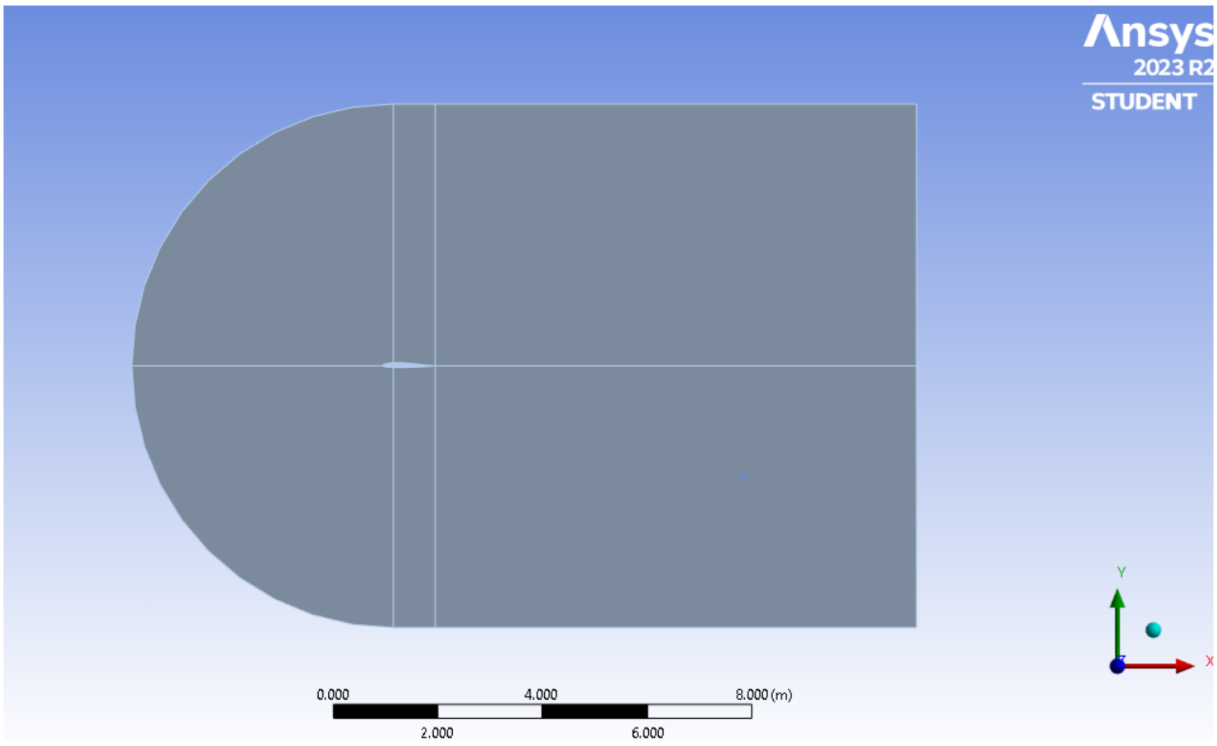


Figure 50: CFD Domain Frontal View

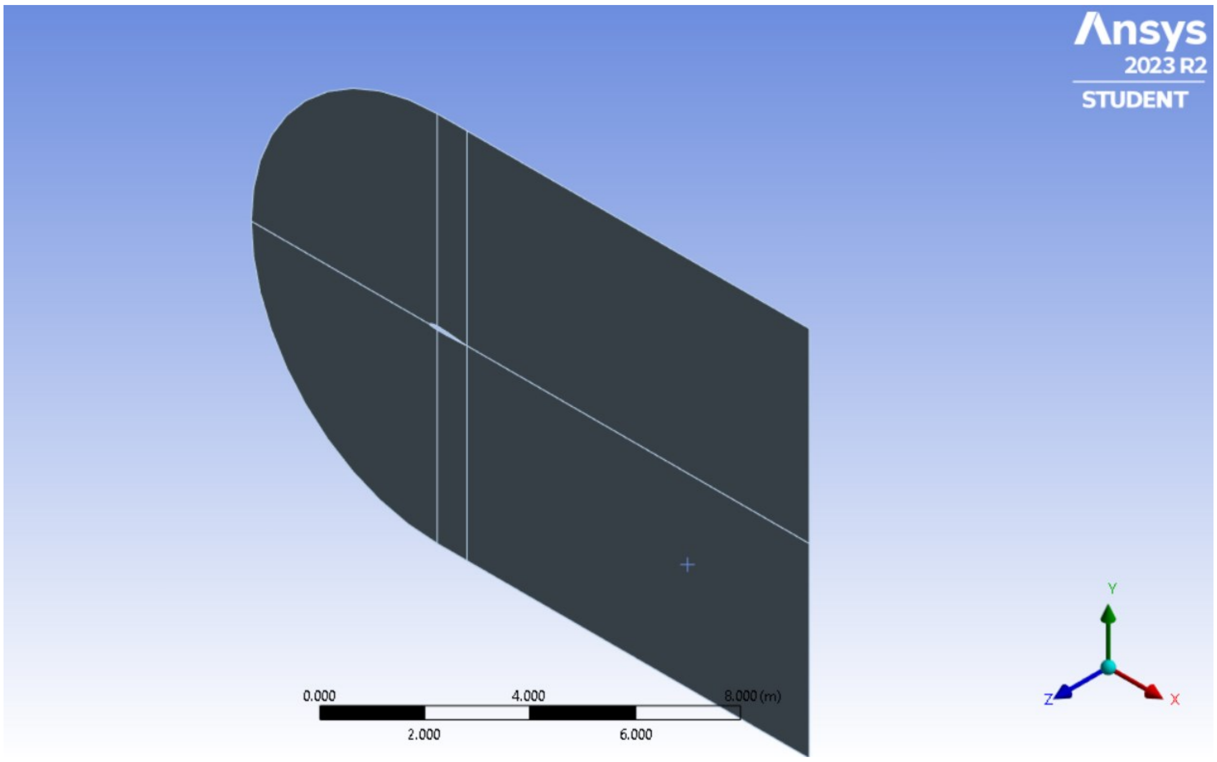


Figure 51: CFD Isometric View

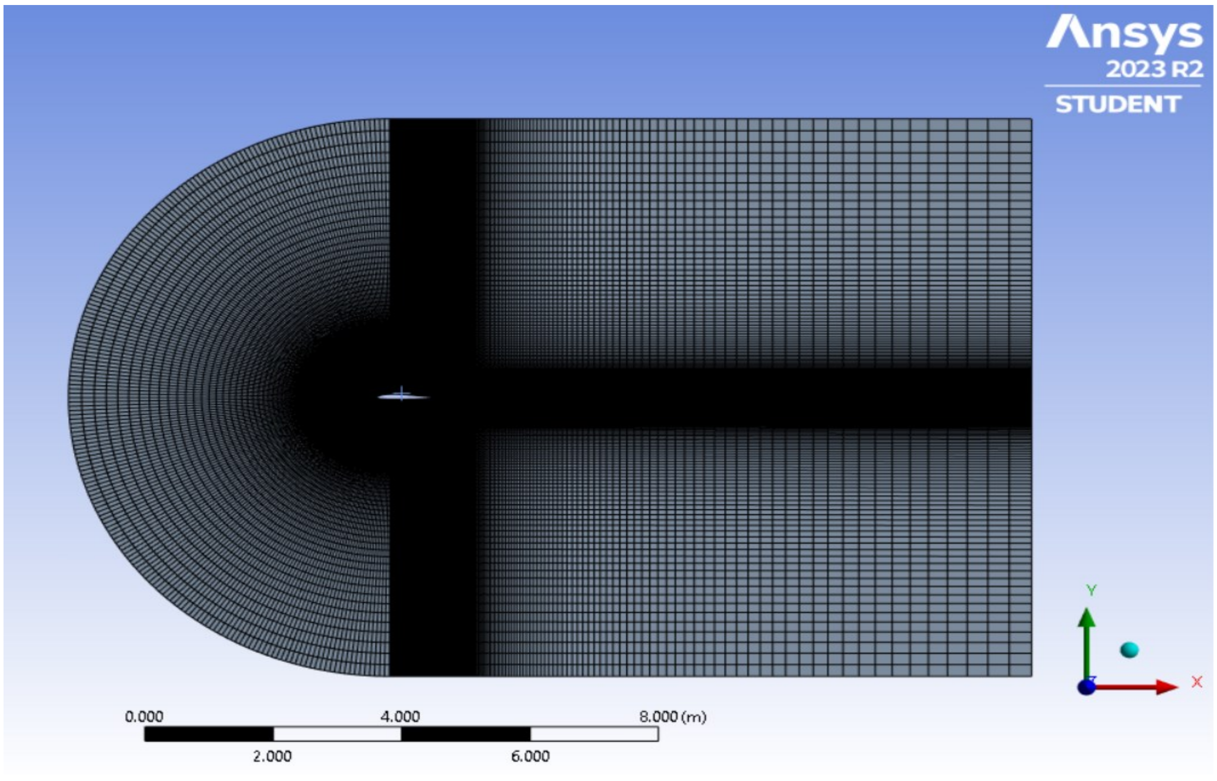


Figure 52: Intermediate Mesh Frontal View

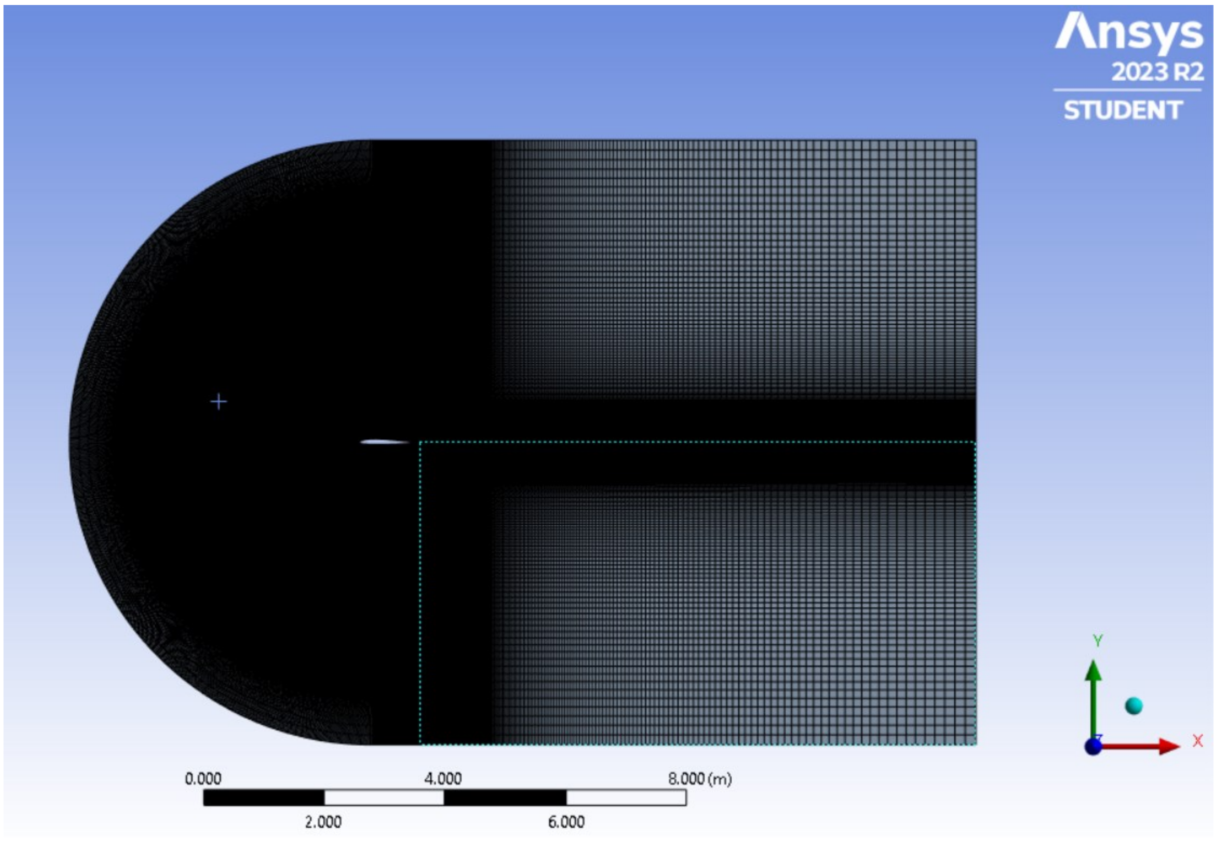


Figure 53: Refined Mesh Frontal View

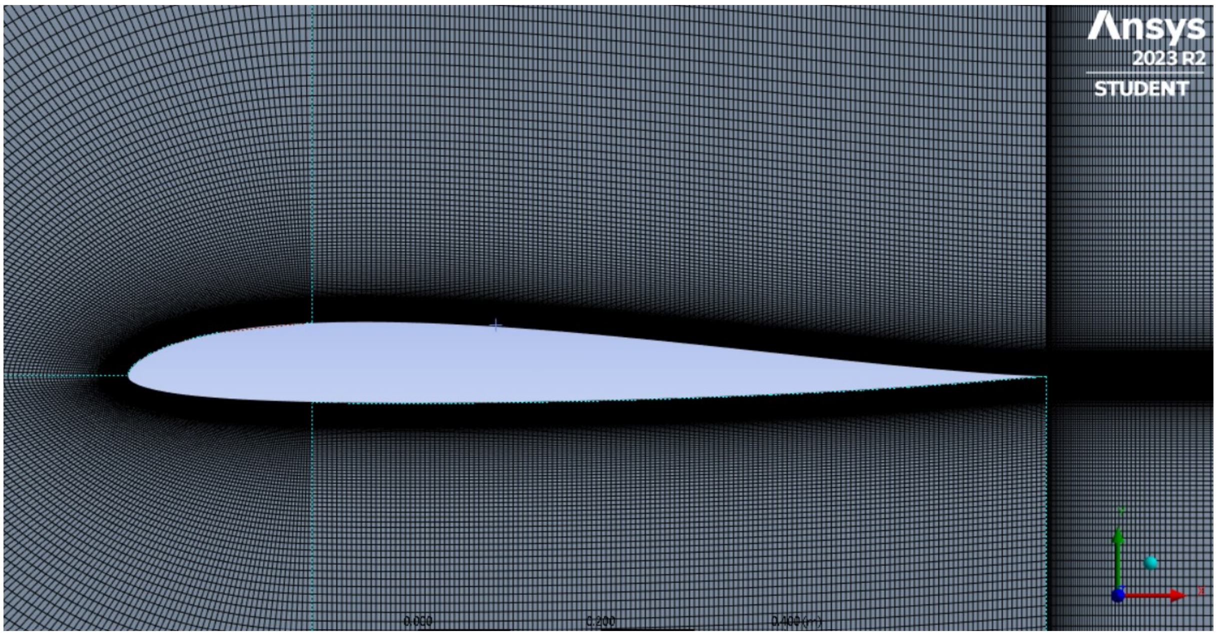


Figure 54: Intermediate Mesh Airfoil

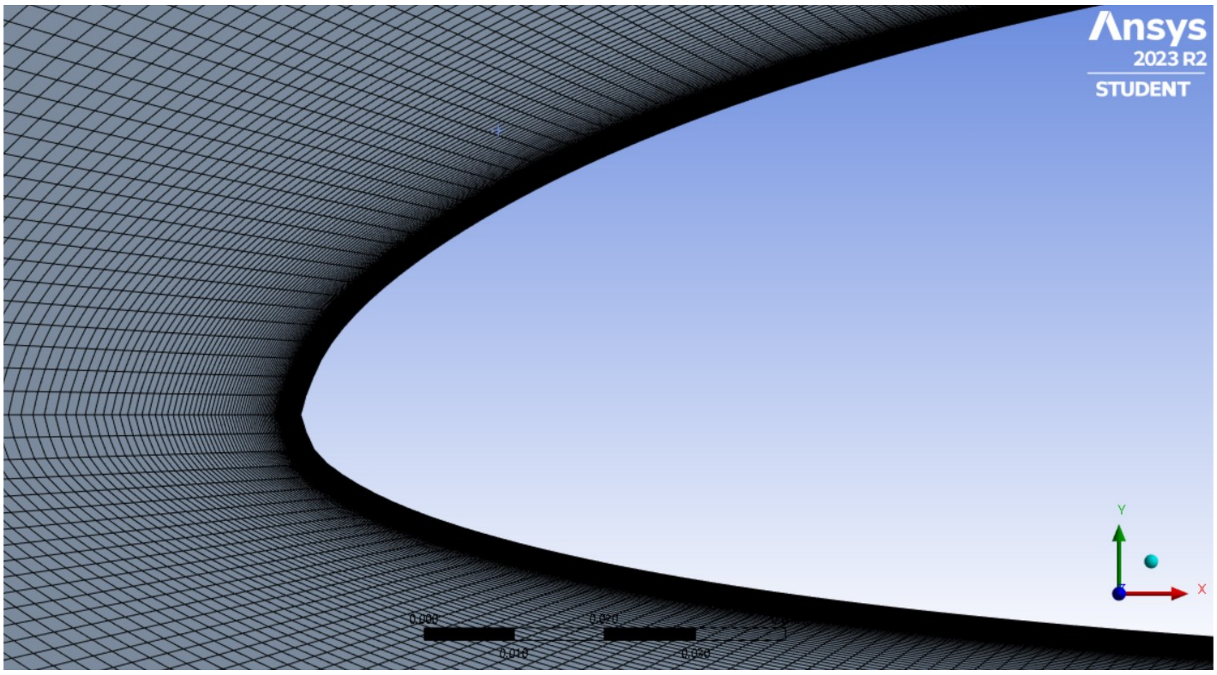


Figure 55: Intermediate mesh leading edge

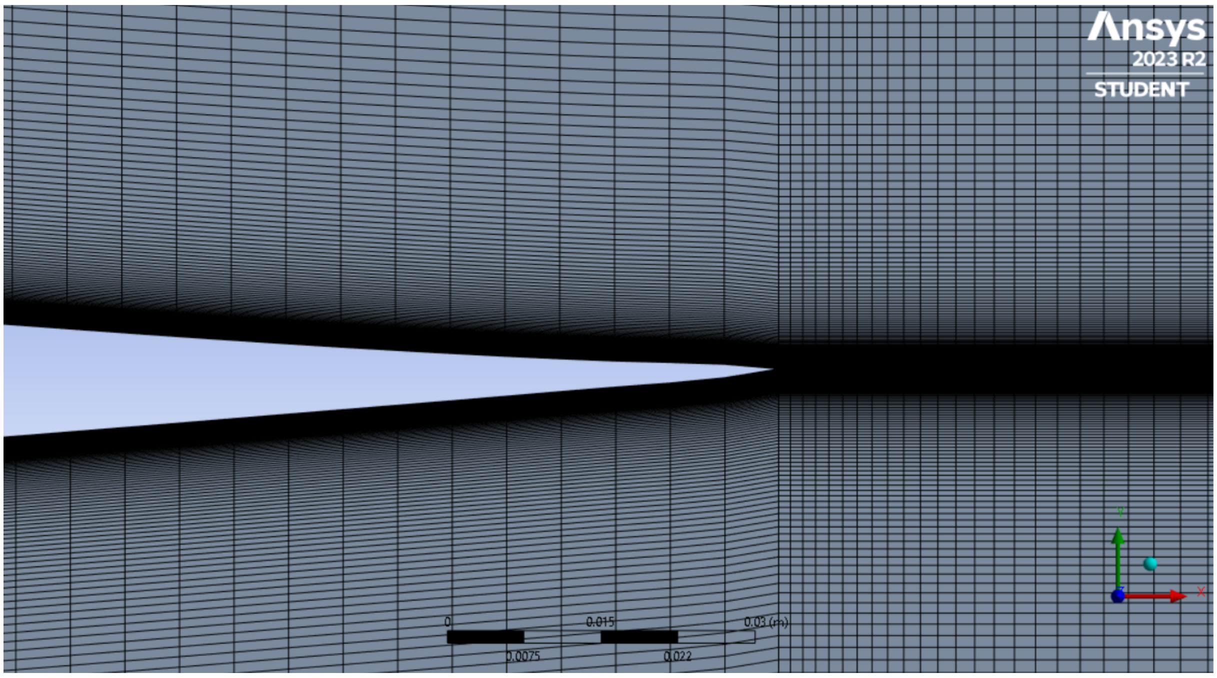


Figure 56: Intermediate mesh Airfoil Trailing edge

8.2.1 Y^+

Given that the aircraft flies at 20 m/s, has a mean aerodynamic chord of 0.37m and flies at low altitudes (<200m) results in: $T = 293K, \mu = 1.82 \cdot 10^{-5}\ [kg/ms]$, it's possible to conclude that:

Equation 51

$$Re_{Cruise} = \frac{\rho V_{Cruise} L}{\mu} = 4.5 \cdot 10^5$$

As the aircraft pitches up, the speed must slow down reaching Reynolds values of almost zero. The Y^+ desired has been set to 1 and has been calculated with the following websites. [22] [23]

Equation 52

$$Y^+ = 1.757 \cdot 10^{-5}$$

Since the fluid, is fully viscous, a great margin was introduced resulting in actual Y+ values listed below:

Table 13: Y+ and first cell height

Mesh	$1^{st} Cell\ Height$	Y^+
Coarse	$1.53 \cdot 10^{-6}$	0.872
Intermediate	$6.4 \cdot 10^{-6}$	0.365
Refined	$2.7 \cdot 10^{-6}$	0.154

As can be seen, all the meshes respect this condition (the image just shows the intermediate mesh).

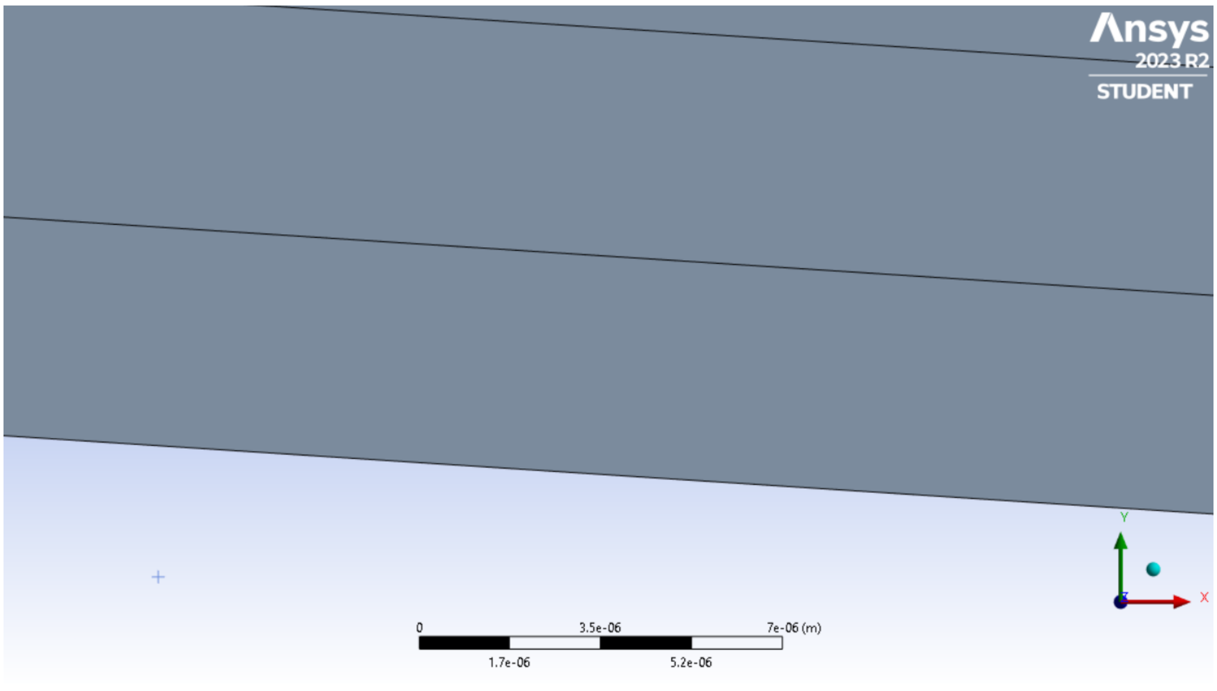


Figure 57: Intermediate Mesh First Cell Height

8.2.2 Mesh rotation

The mesh overall structure has not been changed for higher angles of attack, the only change has been to rotate the airfoil to 45° and 90° to better capture the dynamics in the stream section.

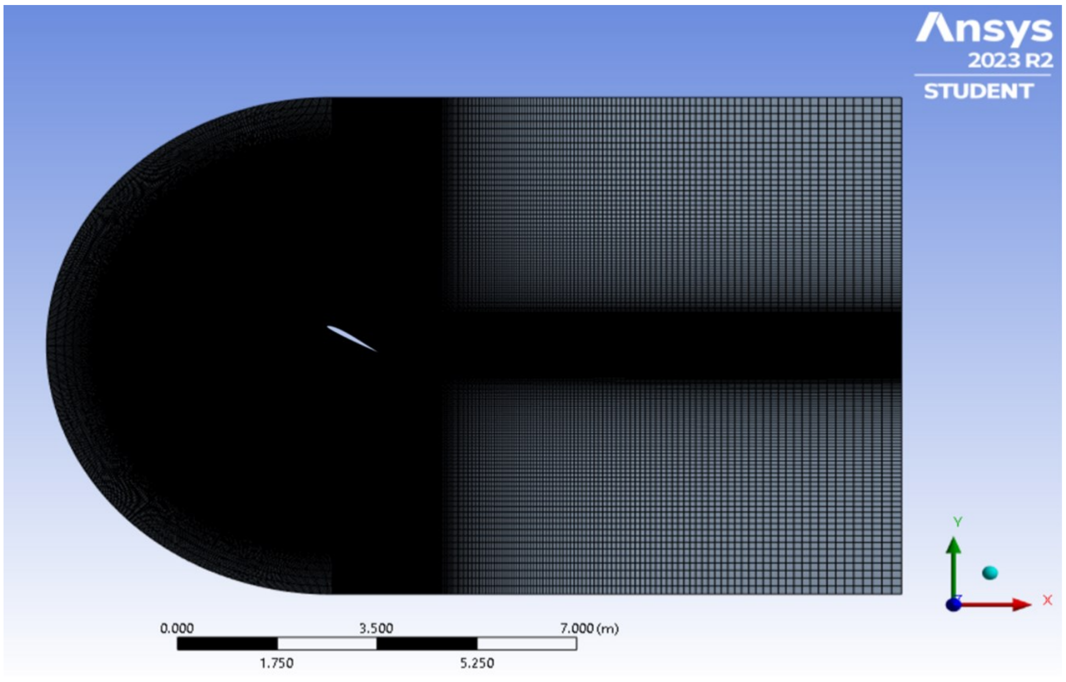


Figure 58: Mesh at 45°

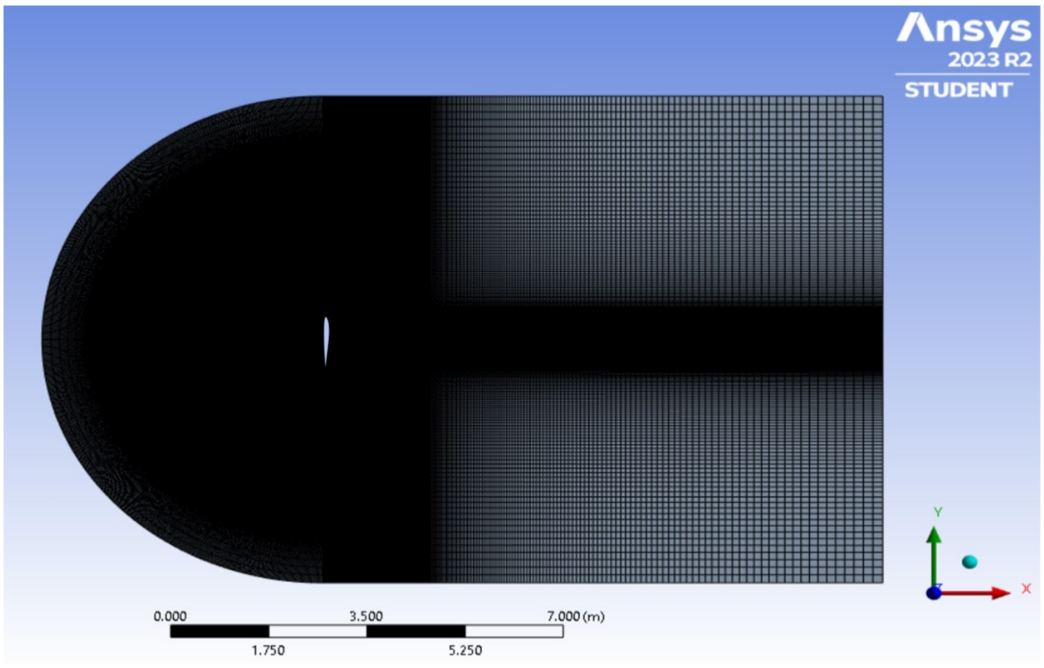


Figure 59: Mesh at 90°

The tool inside Ansys Fluent, dedicated to calculating the coefficients has been reset for each inlet angle variation but also every time the mesh had been switched between the three (0°, 45° and 90°). The ranges of the angles of attack are the following:

- Airfoil Horizontal for angles of attack between -10° and +30°
- Airfoil at 45° for angles of attack between +30° and + 70°
- Airfoil at 90° for angles of attack between +70° and 110°

No differences in the coefficients have been found at the interface between these meshes. This may be because of the tool inside fluent which allows to rotate part of the mesh.

8.3 Mesh Verification and Validation

The three meshes have been used to verify whether the results were different between them.

The validation data was provided by Airfoil Tools since no wind tunnels test has ever been done on this airfoil. The following calculations assume the validation data to be exact.

The available data was from -20° to +20°, therefore this range was the one compared to. The same angles were used for every mesh. The steps are refined near the stall region to better capture the results in the stall region. Spalart-Allmaras vorticity method was used.

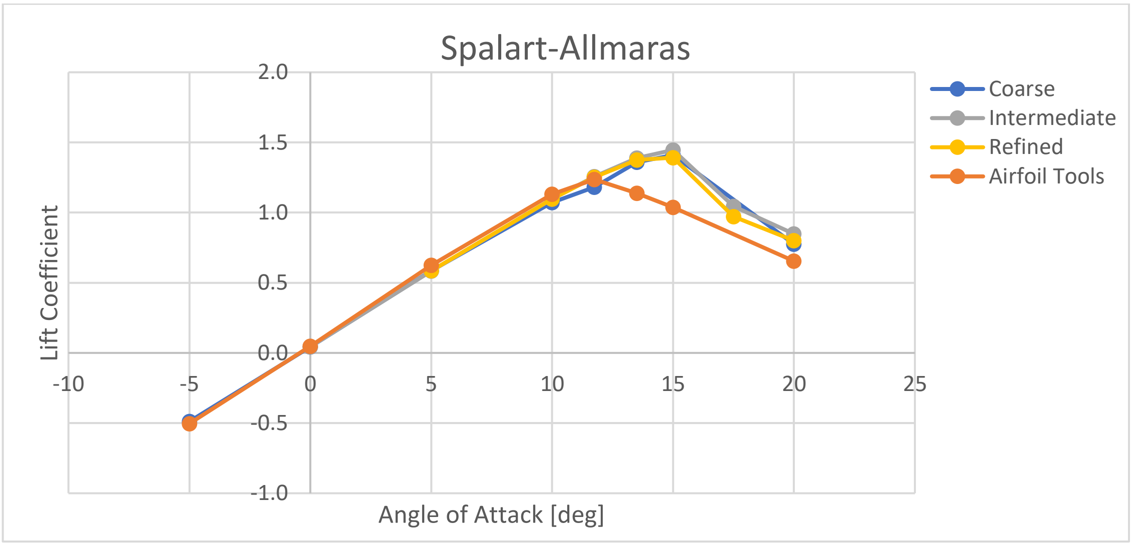


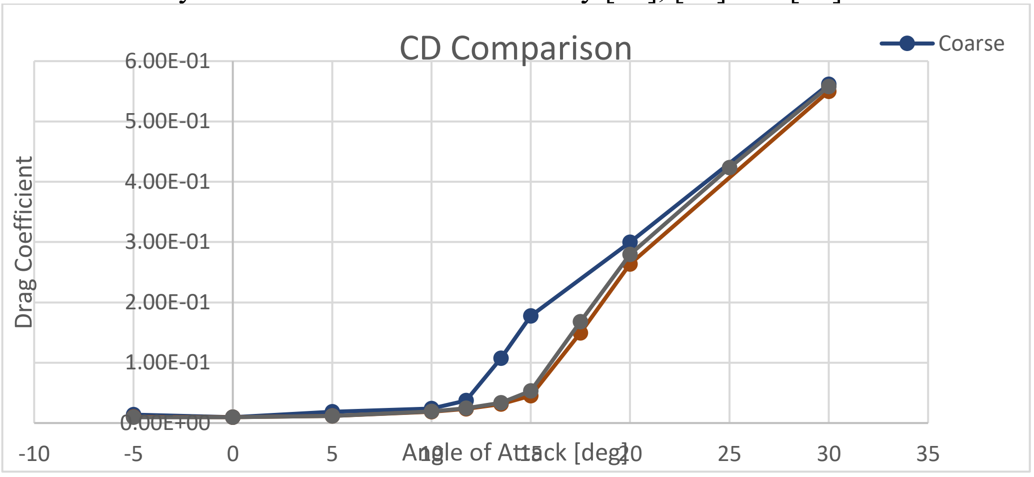
Figure 60: Lift Coefficient vs Airfoil Tools

As can be seen by the graph, the linear behaviour is captured well by all the meshes though all three meshes fail to capture the stall which occurs at 13.5° degrees and instead calculate it to be around 15°. Also, the maximum lift coefficients calculated are:

Table 14: Maximum Lift Coefficient

Mesh	$C_{L_{Max}}$
Coarse	1.43
Intermediate	1.41
Refined	1.39
Airfoil Tools	1.24

These results are very similar to the ones obtained by [24], [25] and [26].



The graph shows the results of the drag coefficients obtained across the three meshes. As it's possible to see the intermediate and the refined mesh give very similar results.

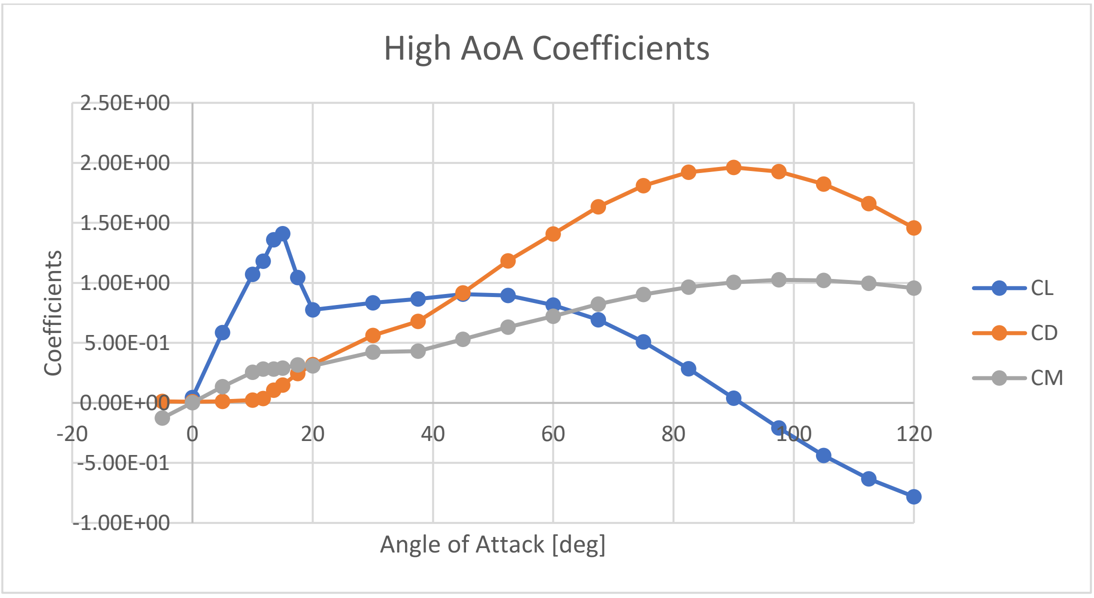


Figure 61: High AoA Coefficients

Now all three coefficients are shown in the graph. It's possible to notice how the recovery lift coefficient remains at 0.8 until almost 60° meaning that this airfoil provides a relatively gentle stall.

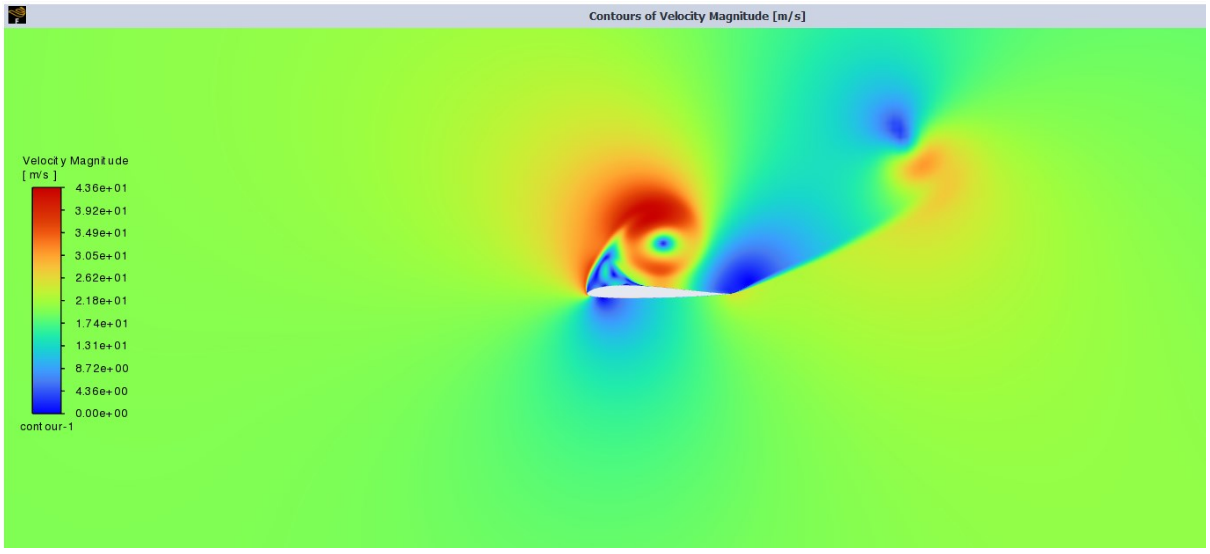


Figure 62: Velocity Field at 30° AoA

To obtain the image, instead of Spalmart-Allmaras, a LES transient has been simulated.

9 SIMULATOR

The 6 *DoF* Simulator is now briefly presented:

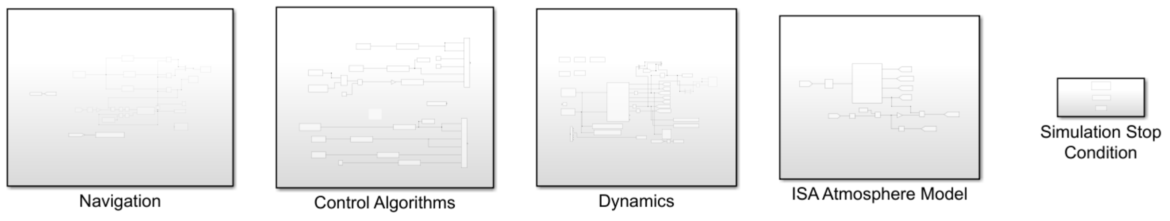


Figure 63: Simulator Main Subsystems

It is composed of 5 main subsystems:

1) Navigation: Provide the input data to the control algorithms for example the desired altitude
2) Control Algorithms: Contains the control loops calculating the required throttle, elevator, aileron and tilt angles.
3) Dynamics: Contains the physics of the simulation.
4) ISA Atmosphere Model: Provides atmospheric data to the Dynamics Block such as: Density, Temperature, Viscosity, Gravitational Acceleration.
5) Simulation Stop Condition: Stops the simulation whenever some conditions are met for example the required altitude or stops in case some variables pass certain thresholds.

The Dynamics Subsystem is as follows:

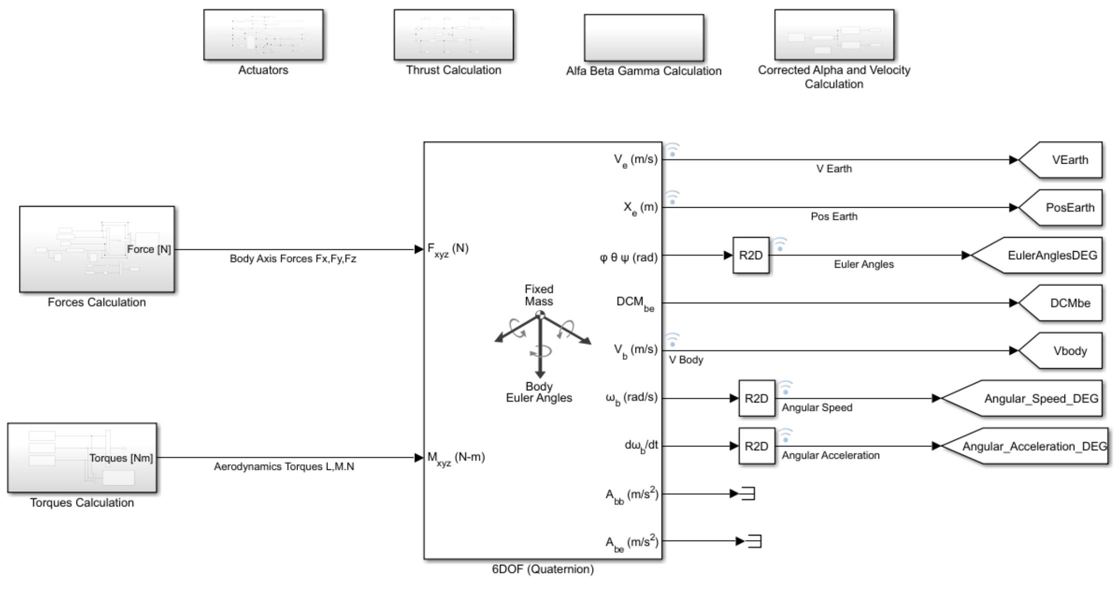


Figure 64: Dynamics Subsystem expanded

a) Forces Calculation: this subsystem calculates the aerodynamic forces, thrust forces and ground reaction forces in body axis frame of reference.

b) Torques Calculation: this subsystem calculates the aerodynamic torques and thrust torques.
c) 6DoF Quaternion: implements the equation of motion in body axis frame of reference outputting all state variables: $N, E, D, \dot{N}, \dot{E}, \dot{D}, u, v, w, \theta, \varphi, \psi, p, q, r$.
d) Actuators: contains the transfer functions of the four servos (two elevons + two tilts).
e) Thrust Calculation: contains the motor model, static and dynamic thrust calculation and slipstream model.
f) Alpha, Beta, Gamma calculation.
g) Corrected Alpha and Velocity: receive as input the velocity and calculates the corrected Angle of Attack and velocity using the slipstream data.

Here's a sample of the obtained data starting with the following initial conditions:

$$X_0 = \{18\,\frac{m}{s}, 6\ \ deg, 0\ deg, 0\,\frac{deg}{s}, -100\ m\}$$

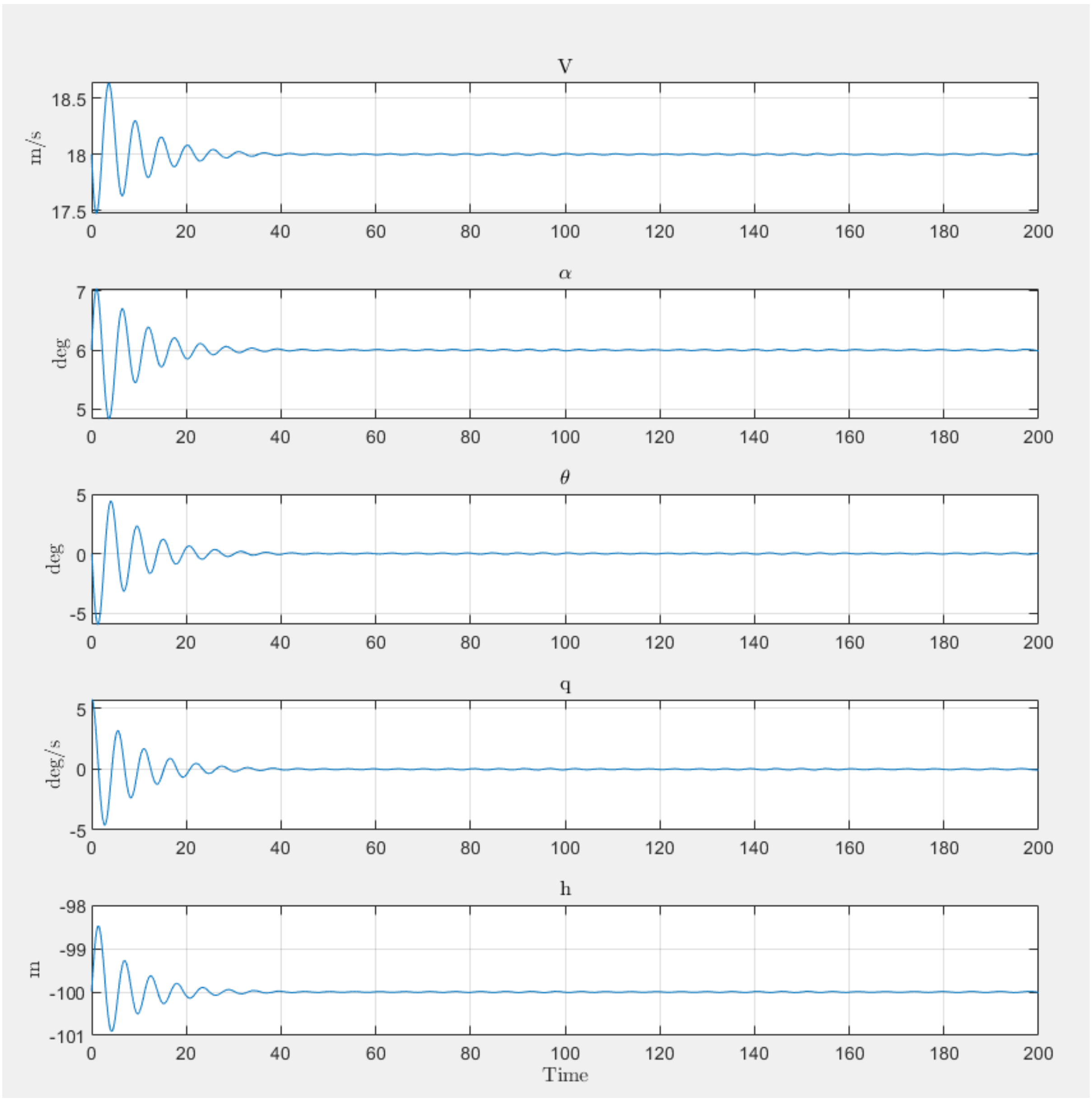


Figure 65: System Response to step

In this graph it's possible to see the aircraft response to a step increase of.

The following instead represents the response to an elevator deflection of 0.2 rad, happening at 100s of simulation:

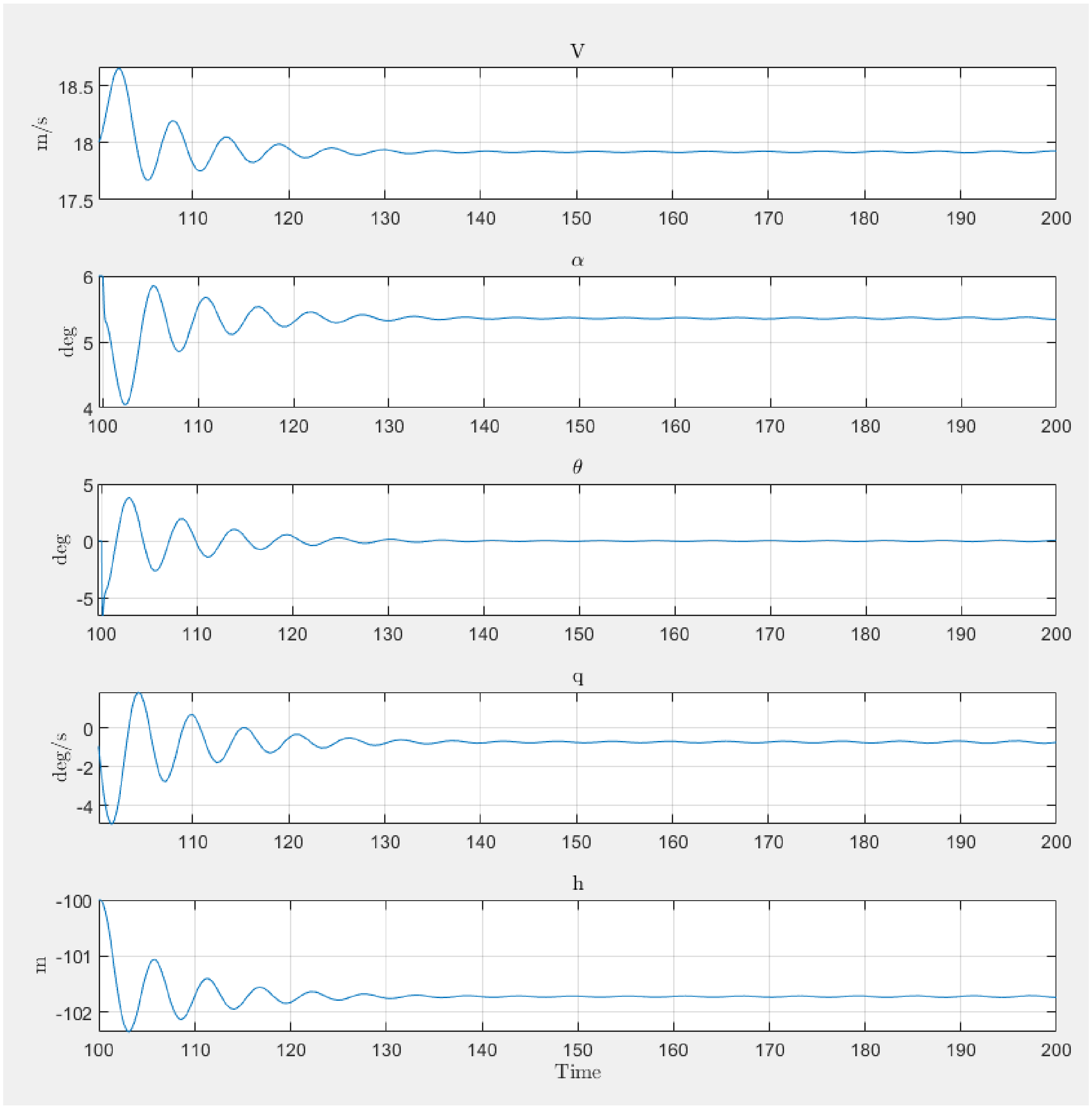


Figure 66: System Response to step increase in elevator angle

Follows the lateral directional response to a sudden increase in the sideslip angle:

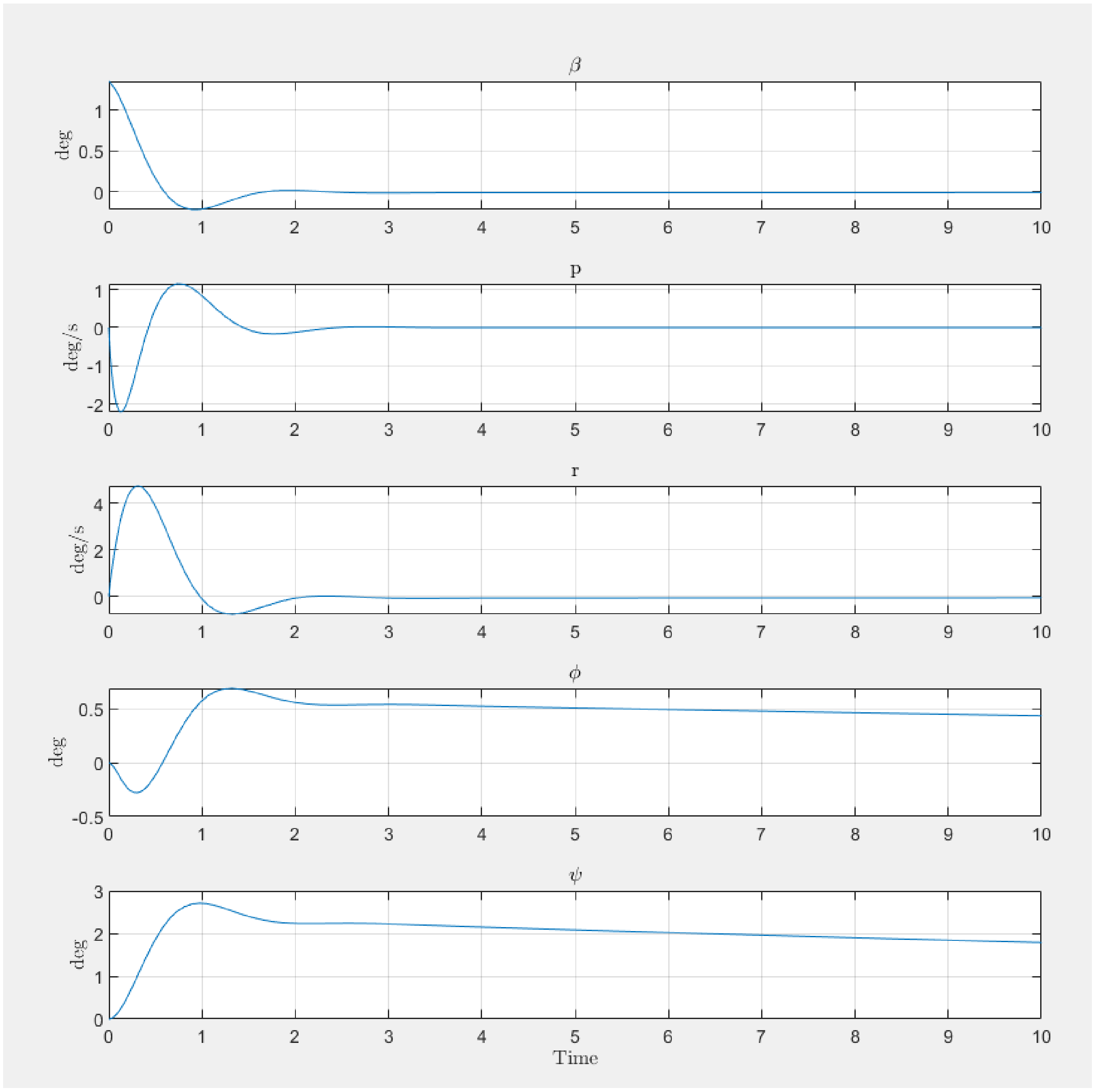


Figure 67: System Response to step increase in sideslip angle

The following shows the response of a singlet aileron deflection of 6 degrees at 100s of simulation:

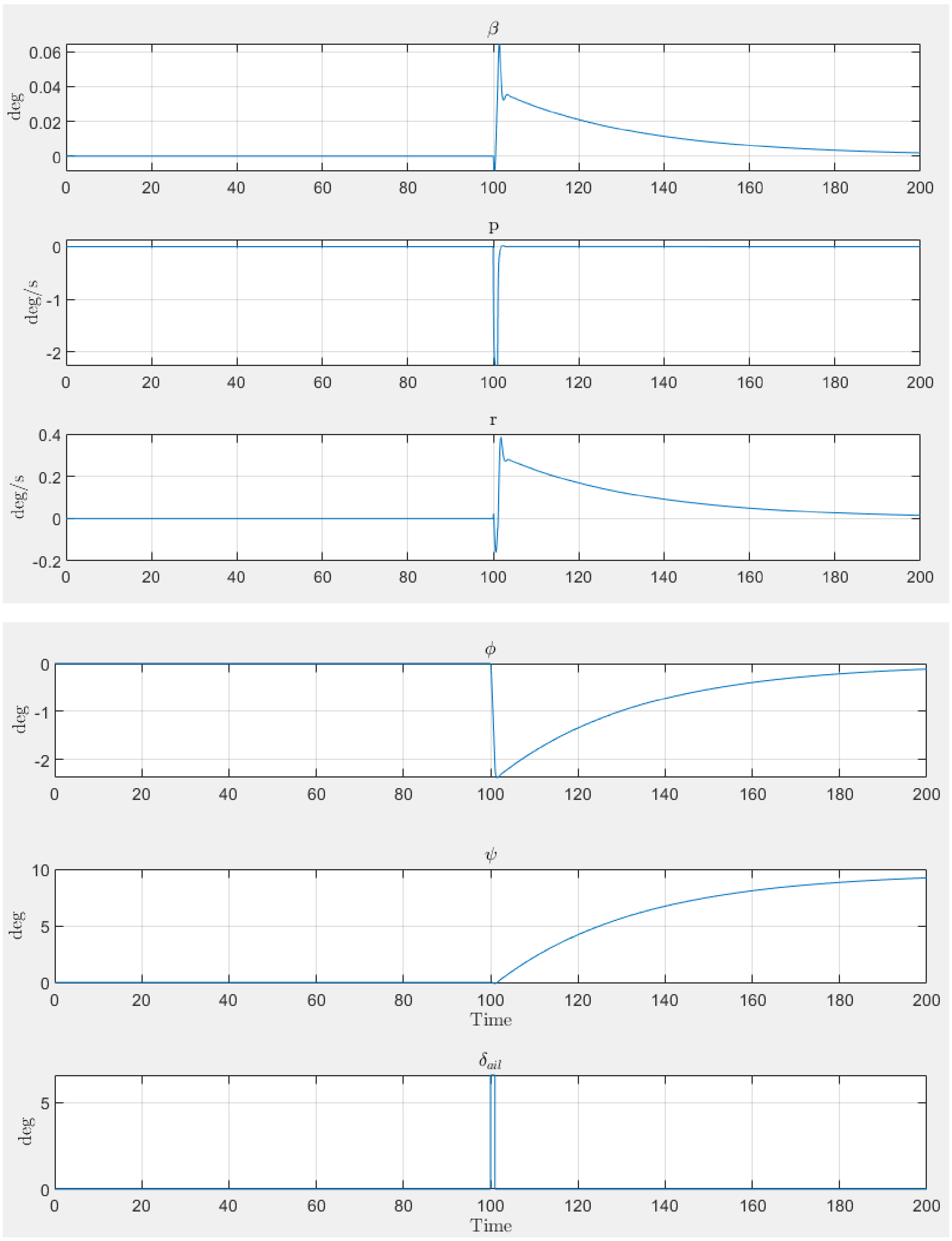


Figure 68: System Response to step increase in aileron deflection

10 TAILSITTER LANDING

The landing of a Bellysitter *VTOL* is a critical phase. The drone first, has to perform a descent while keeping a zero-pitch angle (with respect to *Copter* Frame of Reference or 90° with *Plane*'s). This part can become more difficult because of wind, ground effect and low sensors' accuracy or delays along the calculation paths. The autopilot must realize that it has successfully touched ground. To do that, multiple subroutines are performed while hovering in a position of possible landing which corresponds to a low altitude. A Boolean variable is what tells the drone to start the landing manoeuvre. To set this Boolean TRUE, multiple checks must be completed multiple times. These checks make smart use of IMU, barometer and various other sensors inside the UAV. Of course, the GPS is of vital importance. After making contact with the ground, the drone must complete the rotation around the point of contact. This phase can be extremely dangerous as the propellers are still rotating at high speed while being very close to the ground. The objective of this phase is to complete the rotation and to not move the airplane both horizontally and vertically. The tilting of the motor provides the necessary degree of freedom to control the "fall" and to no to move. The current ArduPilot's procedure is to shut down the motors completely once the Boolean has been set to true. This is accompanied by a tilting of the motor to 90° up so that the blades don't crash against the landing area. Strategy is therefore just turning the propellers off after making contact and rapidly rotate around its point of contact to finally harshly crash its belly on the ground. This can of course damage the structure and the delicate internal electric components. This was not optimal for multiple reasons; the drone needed a large and heavy shock absorber under its belly or otherwise the fuselage would have broken every time. The shock absorber, made of rubber, increases aerodynamic drag and mass simultaneously changing the neutral point. This of course shortens the drone's endurance and ruins its overall aesthetic and flight design. The biggest problem though, is that the rotational energy is not entirely absorbed by the rubber. Once the rubber absorber has hit the ground, the remaining angular momentum makes the aircraft rotate around the rubber itself making it become the instantaneous rotational point. This of course happens because of the high speed gained during the fall and can be disastrous because the blades might touch the ground if the angle gets too high. The blade hitting the ground cause many components to break: the motors, the blades, and the wing-motor interfaces. They usually shatter in multiple pieces projected in different directions which can be extremely dangerous.

This is unfortunately accompanied by ArduPilot's bug which causes the tilts to rotate a little bit when the plane has reached zero-pitch position (*Plane*'s Frame of Reference). ArduPilot has a hard time dealing with the changes of frames of references, especially when they happen as fast and as wide-ranged as during the landing phase. Multiple solutions have been evaluated, both mechanically and software based: a Lua Script was found to be the best solution as it can solve all the previous problems in one take. The bug that was making the propellers' tilts to rotate in the direction of their horizontal position has been found to be caused by the Euler Angles' Singularity. In fact, during this phase the frame of reference used is *Copter*'s therefore the horizontal position is tilted 90° of pitch, the exact requirement for the singularity to occur.

The singularity was making the drone "think" to be upside down as the roll would suddenly increase to 180°. This erroneous attitude estimation caused the tilts to move in the opposite direction to the desired one. The obvious solution to Euler Angles' Singularity is Quaternions.

Quaternions are a different set of parameters that can effectively describe the attitude of any object without any singularity. They have the advantage of being computational inexpensive but not intuitive as the Euler Angles. The relation between Euler Angles and Quaternions is one to one which means that only one combination of quaternions describes a particular attitude and vice versa. Currently the ArduPilot landing is managed through the usage of Euler's Angles estimated by the Extended Kallman Filter. A specific custom script has therefore been created with the objective to safely land the drone.

10.1 System Constraints

The script's targets were:

1) Land the drone delicately: during the descent a maximum speed (Pitch Rate) must be respected.

$$q = f(\theta)$$

2) Zero-Rate at Zero-Pitch: when the aircraft is horizontal, the rotational velocity q must be zero:

$$q = f(\theta = 0°) = 0\frac{rad}{s}$$

3) To reach the belly position in shortest amount of time to minimize the power consumed.

These are the three objectives to pursue. Now the system constraints will be presented:

a) The aircraft doesn't have to move in any direction during the descent:

$$\sum F_x = \sum F_y = \sum F_z = 0$$

b) The propeller's plane must be horizontal during the last part of the descent:

$$u_2(20° > \theta > 0°) = 90°$$

c) The elevons don't move during this phase (otherwise it'd be damaged by the ground):

$$u_3 = 0° \ \forall t$$

d) No torques along the X_{Body} and Z_{Body}axii are generated.

$$\sum Q_X = 0 \quad \sum Q_Z = 0$$

e) This implies the motors' thrust to always be equal, therefore:

$$u_{1_{SX}} = u_{1_{DX}} \ \forall t$$

f) The servo angle of the motors' tilts must be limited to not break the wing structure:

$$-35° < u_2 < 90°$$

The multiple constraints are all extremely important. Even breaking one of these limits can easily cause an incident.

10.2 Script's Constraints

The script must be interfaced with the remaining autopilot's firmware. During this custom landing phase, the Lua script has completely control over the drone behaviour although multiple aspects need to be taken into consideration.

First, the Lua script isn't a part of the original firmware and therefore runs at a low hierarchy level. This is extremely important as it implies the number of times the script is run per second. Multiple tests have been performed before even starting to code and given the high computational power of the autopilot, it was possible to run the Lua code between 50 and 80 times per second. To be conservative, the control algorithm implemented later will be run at 50 Hz although in real flight it might be higher.

Secondly, the script needs to smooth the initial transition between the descent phase and the custom landing phase. The control loops can't be switched abruptly. To solve this problem, some conditions must be met for the custom control loops to take command.

To describe why some conditions are chosen and no other, the following paragraph has been taken from the official ArduPilot's website regarding this matter:

"*Copter will recognise that it has landed if the motors are being commanded to be at low level by the vertical position controller, its climb rate remains between -20cm/s and +20cm/s and is not accelerating for one second. It does not use the altitude to decide whether to shut off the motors except that the Copter must also be below 10m above the home altitude, unless a rangefinder is being used, in which case it must be within 2m of the ground.*" [27]

It must be remembered that the *VTOL* lands in *Copter* mode and follows *Copter*'s set of commands. Once these checks have been positively completed the Lua script takes over. To not cause abrupt accelerations during the transition, the initial values of the motors and servos will be inherited for 100 ms and for at least 10 script runs, whichever comes last. After this check the actual control loop will be activated.

10.3 SETUP

A bench has been created to perform the landing tests. It has the objective to maintain the drone in a fixed position though allowing rotation around the Y axis. The tests have been performed both in lab and then on track.

The setup is composed in the following manner:

a) Platform holding the drone;
b) The drone;
c) Operator controlling both the Ground Control Station and the RC Transmitter in case of immediate shutdown;
d) Operator holding the drone with a dedicated beam mechanically attached to the drone fuselage;

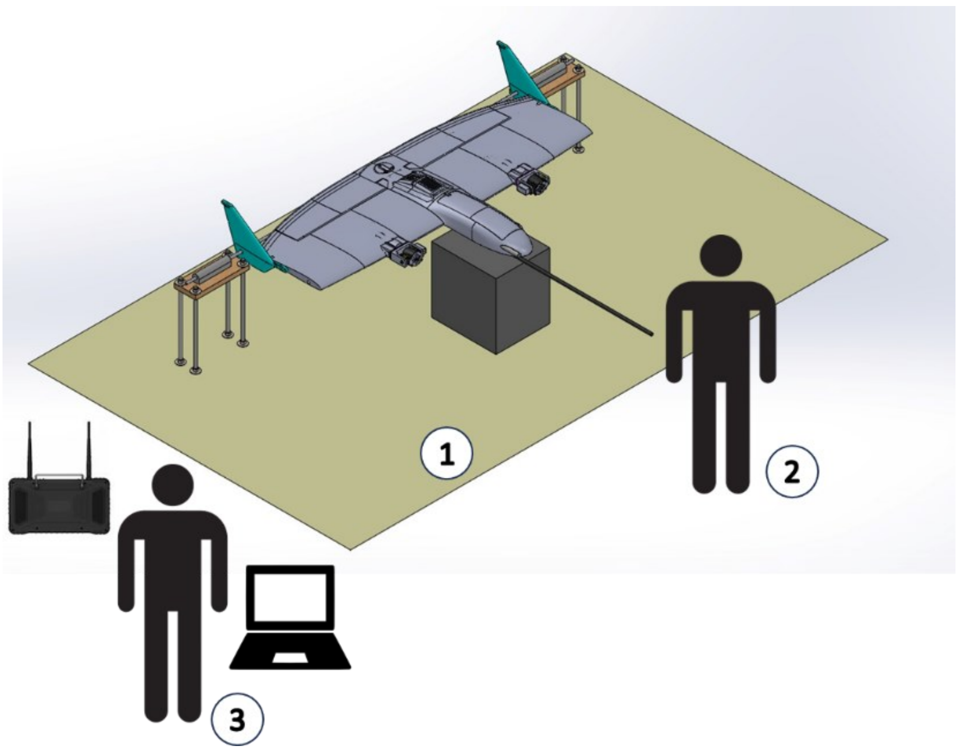


Figure 69: Landing Tests Setup

The platform is shown in the image. It has the objective of not allowing the drone to translate and rotate around the X and Z axis. There's also some foam on the flat surface allowing the drone to fall without breaking the fuselage.

The drone is mechanically attached to the platform through the usage of a beam passing through the winglet. The beam can rotate around its main axis allowing the drone to pitch.

The first operator monitors the signals outputted by the script and is ready to shut down the drone. The second operator holds the beam and mechanically stops the drone in case of sudden loss of control.

10.4 Simulator Details

A simulator has been created in MATLAB/SIMULINK to estimate the necessary gains and better understand the behaviour and dynamics of this phase. During the controlled fall, the drone changes its centre of rotation as different vertices of the winglet, acting as a landing gear, become pivot points. As can be seen in the image, these different points are reached at known angular positions. The lever of the different forces acting on the system change suddenly. Although this detail not being particularly effective over the whole dynamics, if neglected, can create problems in the transition between these points.

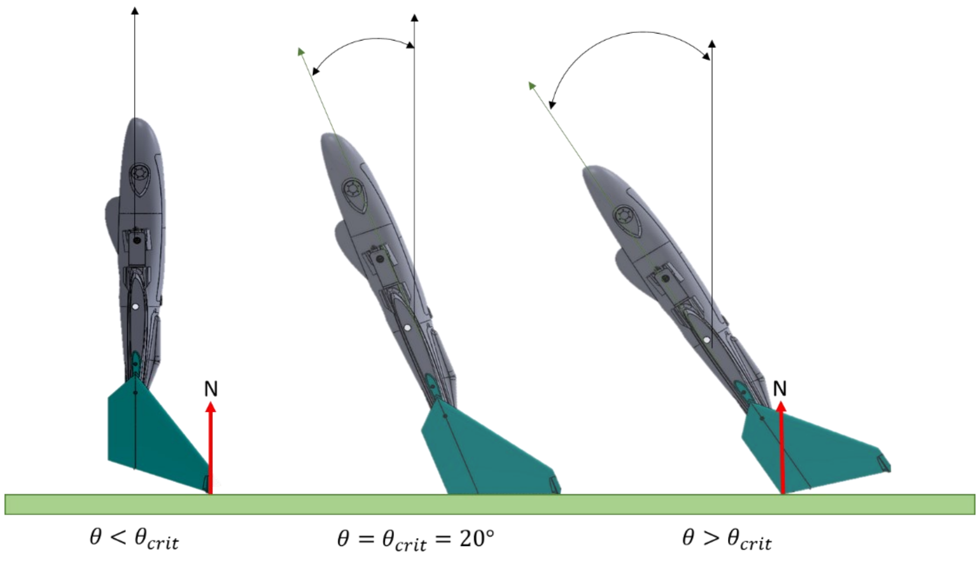


Figure 70: Landing Centres of Rotation

It has been created a dedicated subsystem to calculate the instantaneous point of rotation so future changes in the winglet structure can be reflected easily in the simulator.

The sudden change of the centre of rotation results in a change in angular speed due to a different value of inertia along the new axis. The conservation of angular momentum is used during the simulation.

The force applied to the front of the drone, by the ground is calculated with the following equation:

Equation 53

$$N_{Ground} = -K \cdot \Delta z = -K \cdot (Z_{Front} - Z_{Ground})$$

Where Z_{Ground} is equal to 0 and Z_{Front} is the vertical position in NED frame of reference of the bottom of the rubber component place under the front of the fuselage. The force is only created when the drone touches the ground during the last part of descent. To calculate the reaction force at the centre of rotation instead, another equation is used.

Since the ground takes effect when the sum of the forces is non-zero and the drone's contacting the ground, the resulting force is what makes the Z-axis acceleration zero.

The motor tilt servo is modelled with the following transfer function:

$$\delta = \frac{K_1}{s + K_1}$$

Where $K_1 = 18$ and has been evaluated so that it fits test data obtained by an encoder and gathered by an Arduino.

10.5 Tests

The tests have been conducted at increasing stages of difficulty and drone's autonomy.

The first stage had the objective to validate the capability of the loop to control the system at predefined Pitch Angles. For each test, a specific value has been entered and checked. At the same time, the second operator was holding the beam.

The second stage had the same objective as the first one but without the presence of the second operator. After 10 seconds of correct position hold, the drone has been shut down. The uncontrolled fall has been stopped by the black foam shown in the image above. This stage of testing is particularly important as the Pitch Degree of Freedom is almost completely free of constraints. The only constraint still present inside the system is created by the friction between the beam positioned along the Y_{Body} axis and the two holding components.

The third stage consisted in a controlled fall. The drone has been set at 90° angle with the motors shut down. The second operator is again holding the beam.

The next stage instead aimed to repeat the process without the need of the second operator.

During all the processes, the forces applied on the platform were measured and no non-zero force along the X and Y axis was measured. In the first stage, a small downward force was observed at different times implying that the drone would have taken off is no constraints were present. Indeed, the PWM saturator's upper bound was set too high and therefore lowered.

To complete the next and final stage of testing, a fully interfaced script was needed. The script was then submitted to ArduPilot's forum and is waiting to be validated by others for it to be integrated into the original firmware. An official programmer is needed to refine the script so that other manufacturers can use it.

Table 15: Test Stages

	1st STAGE	2nd STAGE	3rd STAGE	4th STAGE
SECOND OPERATOR	YES	NO	YES	NO
TARGET	PITCH HOLD	PITCH HOLD	FULL DESCENT	FULL DESCENT
PLATFORM HOLD	YES	YES	YES	NO

10.6 Landing Algorithms: PID

The PID is used as first attempt since it’s very easily implemented. It must be noticed that the lever, during the descent changes drastically, increasing the effect of the propellers’ torque over the Y axis. The following scheme provides the basic information on the control loop implemented.

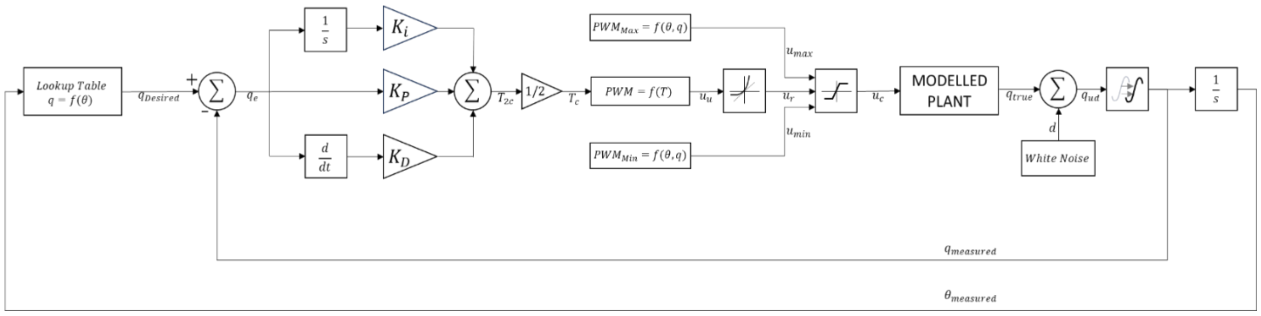


Figure 71: Landing Control Loop

The loop is based on the angular position and angular rate feedbacks. The lookup table at the beginning functions as a desired pitch rate output. As shown in the next graph, it follows a simple function: Once the desired pitch rate has been calculated, the error rate q_e is calculated and fed into the PID controller. The output Thrust is divided by two to consider the two propellers. Afterwards the commanded thrust T_C is transformed into a PWM value. This value hasn’t been limited yet and for this reason has been named u_u (“u” stands for unlimited).

The two subsequent saturations are:

a) Static Rate Saturator: forces the PWM value to not change too drastically. A fast change in PWM value creates oscillations. The oscillations are created by the very low blades’ inertia which, without any input torque, tend to slow down very fast. Since the relation between the propeller angular rate ω and the thrust is very quadratic-like, a light change in ω creates a strong change in thrust which induces oscillations.
b) Dynamic Saturator: forces the PWM value to remain inside a minimum and maximum value. It solves the translational acceleration problem as it makes impossible for the drone to take off. The values change dynamically as more thrust is needed during the last phase of descent. Both the Pitch Angle and the Pitch Rate are fed as in case of fast descent, a higher-than-nominal maximum is required.

Finally, the corrected command u_c is fed to the motors. The script stops at this point, but the simulator tries to model what happens in the feedback branch of the loop. First, a White (Gaussian) Noise is added to the true angular rate q_{true} and later a delay of 10 ms is created. The delay is an estimation of the time it takes the autopilot to run the Kallman filter and provide the data to the Lua script. The delay time can be very different, and it depends on multiple factors which are very hard to model and won't be considered during this work. The delay is a pure estimation and needs to be calculated precisely while the autopilot is working normally.

10.7 Simulation Results

In this section are shown the results of the simulation. Different curves of the desired pitch rate with respect to the pitch angle ($q_D = f(\theta)$)have been created and tested. The following images show only one of the many attempts which is:

Equation 54

$$q_D = a \cdot \sqrt{\left|\frac{pi}{2} - \theta\right|} \qquad \forall\, 0 < \theta < 70$$

And the last part which has been chosen to have a derivative of 0 in the end:

Equation 55

$$q_D = \frac{90 - \theta}{20} b \qquad \forall\, \theta \geq 70$$

The parameters a and b are chosen so that at $\theta = 70°$ the two functions are continuous even though they have different derivatives. The first image shows the Pitch angle during the descent. As can be seen the fall is very slow to lower the damages created by the impact. As can be seen, at 5.2 s the curve has a slight deviation caused by the change in the centre of rotation.

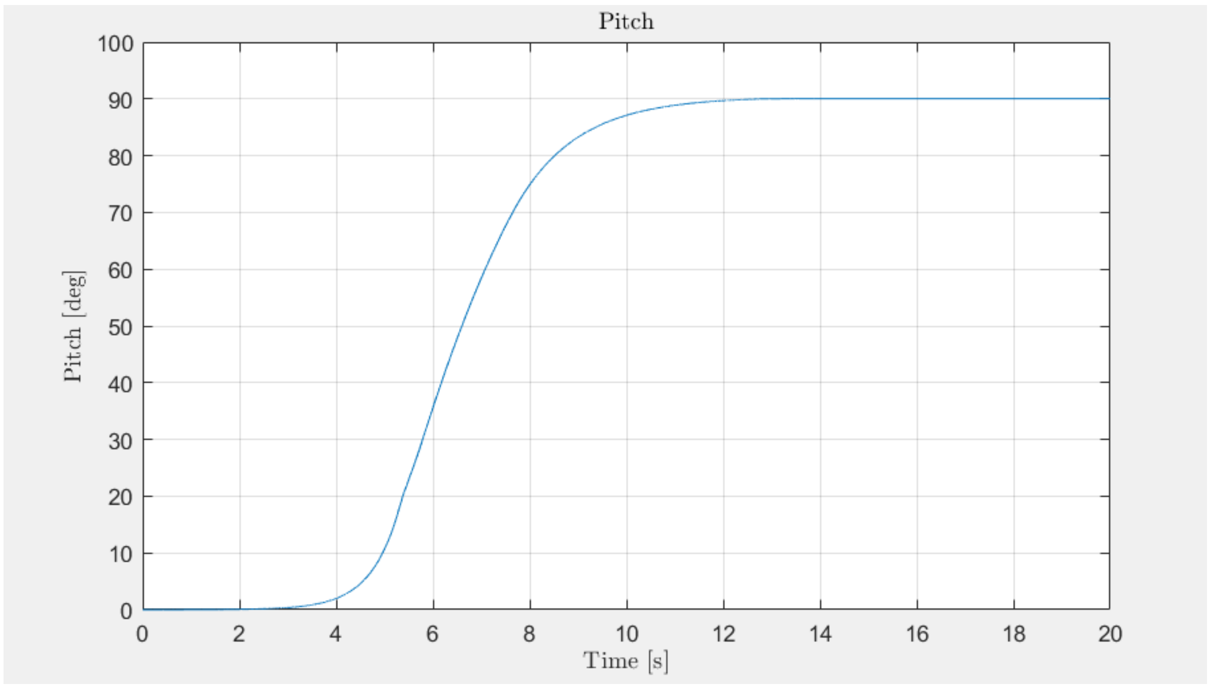


Figure 72: Pitch Angle during landing

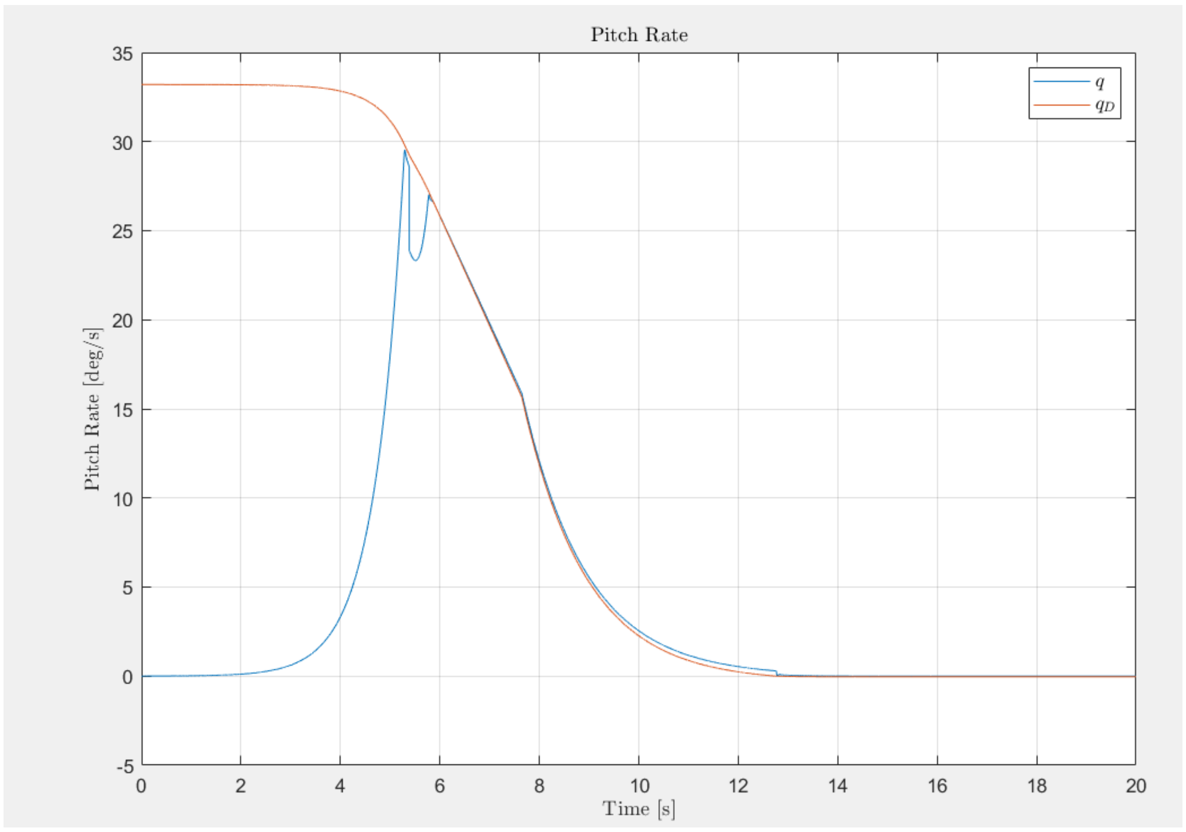


Figure 73: Pitch Rate during landing

The red curve represents the desired pitch rate as the blue one is the actual pitch rate. In the beginning the two are very different as the drone is only beginning to tilt. As can be seen the initial condition is $q(t=0)=0\frac{deg}{s}$. As the drone starts to fall, the motors, which are kept at a reasonable speed, activate to slow down the fall.

At 5 seconds of simulation, the pitch angle reaches 20.29° and the centre of rotation changes to the lower part of the winglet. This causes a sudden change in the inertia and therefore, due to conservation of angular momentum, a change in the pitch rate. The image below is a detail of the previous graph.

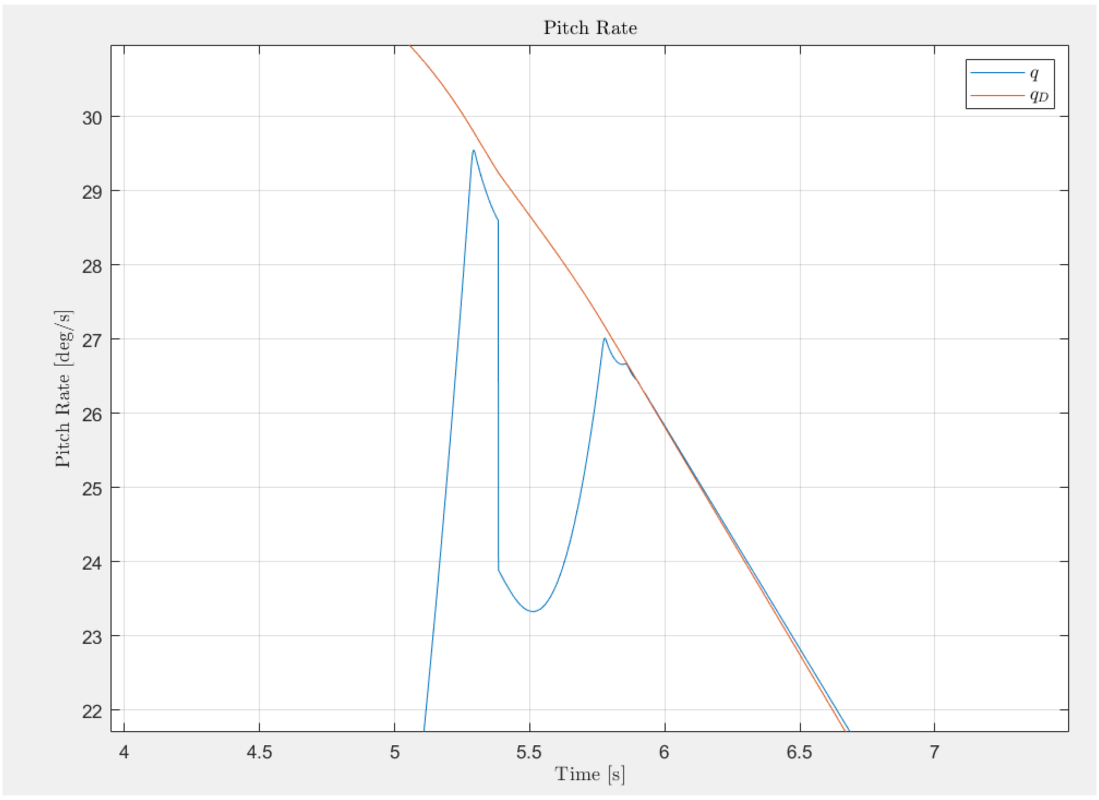


Figure 74: Pitch Rate during change of centre of rotation

After this event, the drone follows the desired descent rate. At time 12.76s the lower part of the rubber shield impacts on the ground creating an upward acceleration which stops the rotation of the drone.

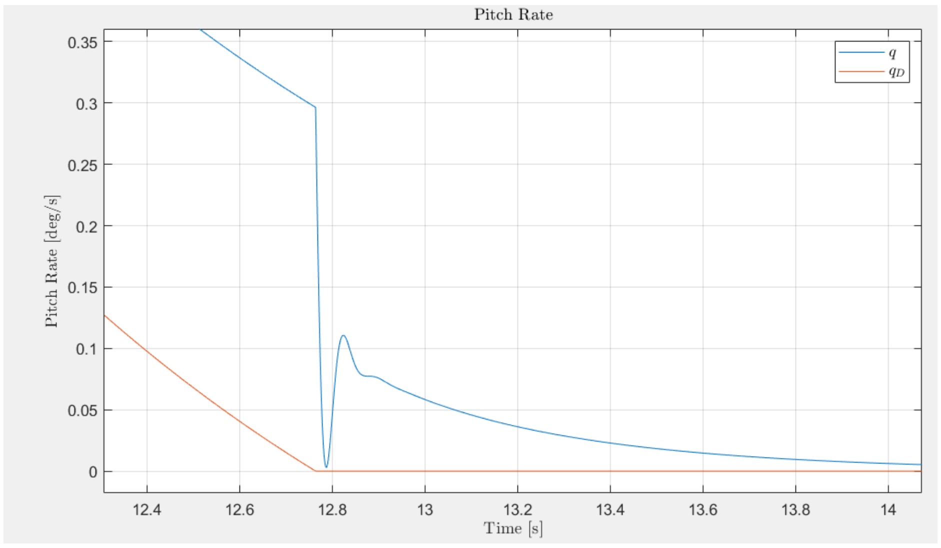


Figure 75: Pitch Rate during ground contact

The average force applied during the impact can be calculated:

$$\alpha_{avg} = \frac{\Delta q}{\Delta t} = \frac{0.3\,\frac{deg}{s}}{12.785s - 12.764s} \approx 15\,\frac{deg}{s^2}$$

And therefore, the force:

$$I_{C.o.R.}\Delta\omega = N_{avg} b_{Front} \Delta t \rightarrow N_{avg} = \frac{I_{C.o.R.}\Delta\omega}{b_{Front}\Delta t} < 1N$$

As can be seen the impact is extremely slow whereas during a normal, uncontrolled fall, the resulting force is much greater:

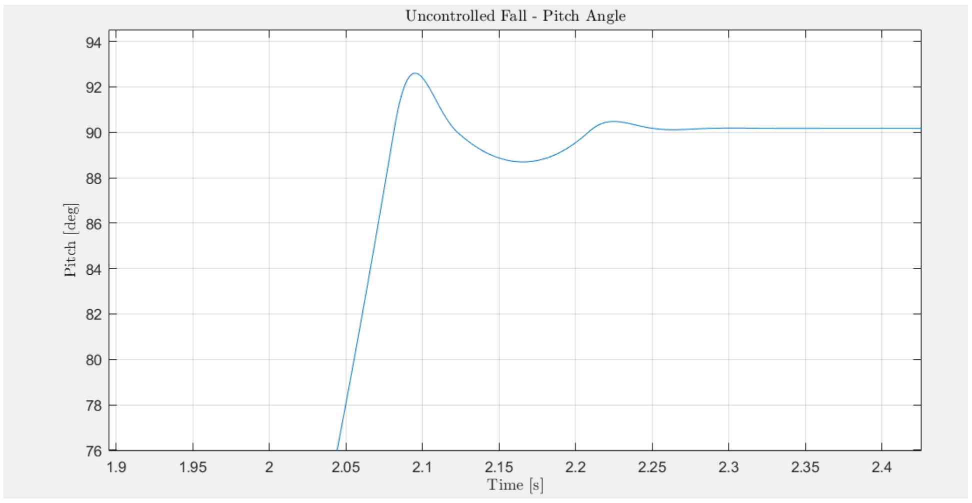


Figure 76: Pitch Angle during uncontrolled fall

In this case the force is equal to ≈150 N. This amount of force can damage the structure as well as risking the propeller. Next are shown the two inputs of the problem, the thrust and the servo angle:

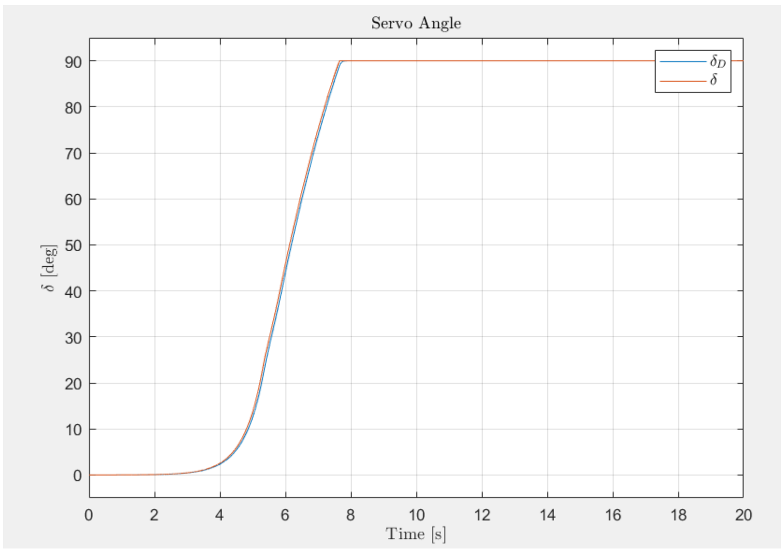


Figure 77: Tilts' Servo Angle during landing

The red curve represents the actual servo angle whereas the blue one is the desired angle which is calculated with the following equation:

$$\delta_D = \theta \cdot K_\delta$$

Where K_δ is calculated based on at what pitch angle, is desired to have the servo completely tilted vertically. Here's chosen $K_\delta = 7/9$.

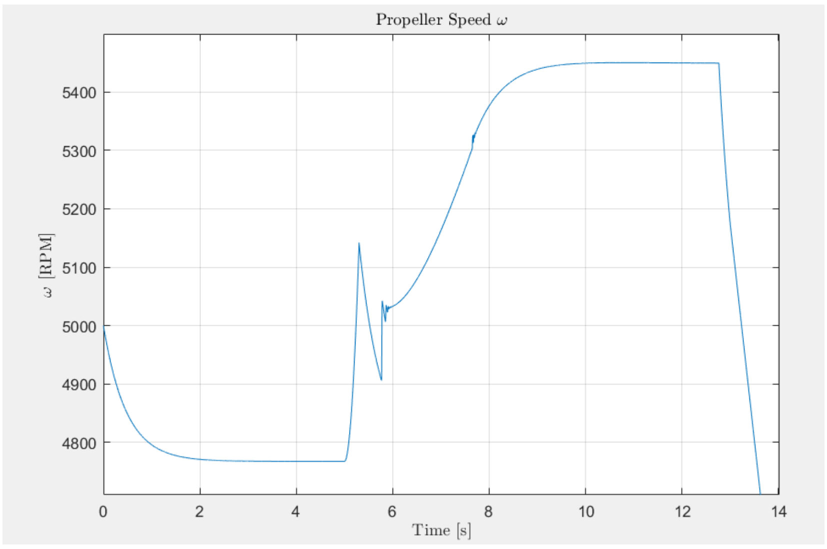


Figure 78: Propeller Speed during landing

As can be seen, the range of RPM increases during the fall because of the increase in torque required. In the last part of the descent, the motors generate around 1.75 kgf each, or 34.3 N in total. The minimum PWM is required because otherwise the RPM would decrease too much in the initial phase, when the drone is gaining rotational speed. Once the drone is completely flat on the ground, a Boolean deactivates the propellers completely.

11 TOOLS

This chapter presents a comprehensive list of tools utilized in this thesis. They are briefly described, highlighting their specific purposes in the study.

- **MATLAB**: it served as the primary platform for computation and simulation in the thesis. It played a crucial role in various aspects:
    - It conducted a detailed examination of log data to gain insights into the aircraft's behavior during tests or simulations;
    - It provided a preliminary estimation of aerodynamic coefficients, including the calculation of propeller slipstream;
    - It analyzed forces, torques, and overall performance of the propeller;
    - It acted as a tool for generating input data used in the development of simulators.
- **SIMULINK**: it was employed for simulating the dynamics of the system and to study and tuning control loops. It played a key role in understanding and refining the control mechanisms governing the aircraft.
- **Cygwin**: it facilitated the integration of the simulator with created scripts, ensuring a seamless interface. It played a vital role in:
    - Enhancing compatibility with Mission Planner;
    - Incorporating scripts into the simulator environment to facilitate testing of interfaces and overall system behavior.
- **Ansys**: it conducted 2D CFD simulations of the PW-75 Airfoil. Ansys was utilized for estimating aerodynamic coefficients of the aircraft, particularly in scenarios involving high angles of attack. The software provided detailed insights into the aerodynamic behavior of the airfoil under such conditions.

- **Mission Planner**: it acted as a data collection tool during flight tests, generating logs for further analysis. Enabled real-time monitoring of the aircraft, providing crucial insights into its performance. It played a central role in mission planning, creating trajectories to achieve target data.
- **AVL (Athena Vortex Lattice)**: it was employed for estimating longitudinal and lateral-directional aerodynamic coefficients at low angles of attack. The software was instrumental in providing estimates crucial for creating an aerodynamic database usable in simulators, particularly for low angles of attack.
- **Excel**: it played a pivotal role in organizing and analyzing data:
    - It sorted and formatted generated databases, ensuring clarity and ease of analysis;
    - It evaluated alternative solutions and semi-empirical equations sourced from literature and the web, contributing to a comprehensive understanding of the subject matter.
- **ArduPilot**: given its importance in this work, the tool is described more in details in the following paragraph.

11.1 ArduPilot

ArduPilot is the firmware installed in the autopilots. It is an open-source code developed by hobbyists to control different types of aircraft but also Helicopters, Rovers, Boats, Submarines and others.

ARDUPILOT

Figure 79: ArduPilot's Logo

It is now being used by most drones' companies since it provides many features adaptable to most of the payload and it's easily configurable.

It provides communication with the Ground Control Station and manages the communication entirely by itself.

Another useful feature is the possibility of integrating other computers in case of need. This is especially important when more complex – computation expensive types of control algorithms have to implemented, such as MPC or Artificial Intelligence.

ArduPilot software is downloadable on GitHub and it's easily buildable on a Linux operating system. It needs to be connected to a Ground Station control software such as Mission Planner,

QGround, MavProxy and others. Its community is constantly expanding, and the code improves systematically thanks to the community's feedback on online platforms.
The customizability is created through parametrization. More than 500 parameters can be set to modify the drone's behaviour in all situations. These can easily be changed on a Ground Control Software such as Mission Planner.
Mission Planner provides users with an intuitive interface of what's happening in real time, it signals warnings, errors, state variables, commands, status and others. It can easily be planned through the "Plan" window with a great variety of options available and can be later analysed as of all of the Mission Data are saved as .bin files.
These .bin log files were the ones used in this works to obtain the data needed to study the dynamics of both drones. They can easily be downloaded from the autopilot through Mission Planner and later be analysed with different software. The programs used to analyse the logs were Mission Planner and 3D. The second one provides a 3D representation of what happened during the mission. It was used to check different behavioural situations and to have an overall better understanding of what happened. Probably the most important aspect of Mission Planner are the parameters which can be modified through the "Config" window. ArduPilot provides the users with important tools to calibrate on-board instruments and to automatically tune the multiple PID present in the various control algorithms.

12 CONCLUSIONS

A physical model of Strategy's dynamics has been developed.

Geometrical and Inertial data has been estimated in lab. Propeller performances have been thoroughly modelled by both bench tests and flight tests Three different approaches were used to estimate the aerodynamic coefficients: log based, open-source software and CFD.

All three methods have advantages and disadvantages although, given the low cost of flying this kind of drone, the log-method is preferred. To increase this method's estimations, more accurate sensors can be implemented as well as more powerful autopilot's CPU. The coefficients of determination R^2 were over 0.7 which can be considered as the starting point for more accurate studies. To improve the results also more regressors can be inserted inside the method as well as more a bigger flight database.

The Plane mode has been described completely. Modelling the propeller slipstream and dividing the wing into sections seems like the way to improve the accuracy, as done by others. *Copter* mode remains to be fully analysed although the modelling is complete. The transition between the two requires a high-fidelity dynamic longitudinal coefficients representation which can be achieved with more expensive transient 3D CFD simulations. These simulations should also consider the aileron and/or elevator deflections across a wide span of angle of attack.

A 6 *DoF* simulator has been created to analyse the dynamics of Strategy during *Plane, Copter* and landing mode. Future improvements include creating a robust control system to obtain a levelled Cobra manoeuvre. The simulator could also be used to model the *VTOL's* dynamics in the presence of external wind with the purpose of creating a wind-resistant control system to obtain better descending phases in *Copter* mode.

The log method can be used to without too much data for many different fixed wing drones having ArduPilot.

All these results could also show how to correctly setup the several Mission Planner parameters related to the transition maybe demonstrating that new control loops are not necessary.

13 Bibliography

[1] "Commercial UAV News June 2023".

[2] S. Mahan, T. Johnston, J. Lidel, R. Riedel and L.-. Tusch, "Drone delivery: More lift than you think," *McKinsey & Company,* 15 March 2022.

[3] https://ardupilot.org/plane/docs/parameters.html.

[4] https://ardupilot.org/copter/docs/parameters.html.

[5] C. Mazza, "Analisi delle caratteristiche aero-meccaniche di un," 2023.

[6] G. Staples, "Propeller Static & Dynamic Thrust Calculation - Part 2 of 2 - How Did I Come Up With This Equation?," *Electricr RC aircraft Guy, LLC,* 12 April 2014.

[7] "Airfoil Tools," [Online]. Available: http://www.airfoiltools.com/.

[8] W. Khan, R. Caverly and M. Nahon, "McGill University".

[9] S. S. Chauhan, "A simple method for estimating propeller-induced tangential velocities," *Medium,* 13 August 2021.

[10] P. Okonkwo and P. Jemitola, "Integration of the athena vortex lattice aerodynamic analysis software into the multivariate design synthesis of a blended wing body aircraft," *Heliyon,* 2023.

[11] M. Drela and H. Youngren, "AVL 3.36".

[12] "TOOLS FOR THE CONCEPTUAL DESIGN AND ENGINEERING ANALYSIS OF MICRO AIR VEHICLES," *AIR FORCE INSTITUTE OF TECHNOLOGY,* March 2009.

[13] G. J.A. and E. Morelli, "A Generic Nonlinear Aerodynamic Model for Aircraft," *NASA Langley Research Center.*

[14] B. Y. University, "UAV Coordinate F dinate Frames and Rigid Body Dynamics ames and Rigid Body Dynamics," 2004.

[15] https://plot.ardupilot.org/#/.

[16] D. Hosier, "AVOIDING GIMBAL LOCK IN A TRAJECTORY SIMULATION," *Technical Report ARMET-TR-17051,* July 2018.

[17] M. S. Selig, "University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA".

[18] https://ardupilot.org/copter/docs/motor-thrust-scaling.html.

[19] A. Chugh, "MAE, MSE, RMSE, Coefficient of Determination, Adjusted R Squared — Which Metric is Better?," *Medium,* 8 December 2020.

[20] "Coefficient of determination," *Wikipedia.*

[21] https://wes.copernicus.org/preprints/wes-2021-45/wes-2021-45.pdf, “Copernicus,” wes, 2021. [Online].

[22] CFD-online, “yplus,” [Online].

[23] cadence. [Online]. Available: https://www.cadence.com/en_US/home/tools/system-analysis/computational-fluid-dynamics/y-plus.html.

[24] J. A. R. J. a. M. S. Andrew Shelton, 2012. [Online]. Available: https://arc.aiaa.org/doi/abs/10.2514/6.2005-1227.

[25] K. J. B. S. G. S. M. W. F. S. L. K. J. T. M. a. J. E. L. O. J. Boelens, 2012. [Online]. Available: https://arc.aiaa.org/doi/abs/10.2514/1.34852?journalCode=ja.

[26] R. C. P. A. G. a. N. T. F. Justin L. Petrilli, 2013. [Online]. Available: https://arc.aiaa.org/doi/abs/10.2514/6.2013-2916.

[27] “Ardupilot Copter,” [Online]. Available: SITE: https://ardupilot.org/copter/docs/land-mode.html.

[28] https://www.electricrcaircraftguy.com/2014/04/propeller-static-dynamic-thrust-equation-background.html.

[29] N. Shelil, “2D Numerical Simulation Study of Airfoil Performance,” *Mechanical Engineering Department, College of Applied Engineering, King Saud University, S.A. & Port Said University,,* May 2021.

[30] “Ardupilot Copter,” [Online]. Available: SITE: https://ardupilot.org/copter/docs/land-mode.html.