FINAL DEGREE PROJECT

TITLE: Drone trajectory prediction toolset

DEGREE: Doble Grau en Enginyeria de Sistemes de Telecomunicació i Enginyeria d'Aeronavegació

AUTHOR: Daniel Gutierrez Herranz

ADVISOR: Enric Pastor Llorens

DATE: 8 de febrer del 2022

Title: Drone trajectory prediction toolset

Author: Daniel Gutierrez Herranz

Advisor: Enric Pastor Llorens

Date: February 8th 2022

Overview

Nowadays, microprocessor and microcontroller technology is evolving very quickly, and this means that drones can have more features at a competitive cost. This has caused drone operations to evolve from purely military purposes to cargo transportation and delivery tasks. As a consequence, the demand for drone permits has been increased exponentially over the last decade and this sudden growth can cause problems in very low-level air traffic management.

There are models that explain the behaviour of a drone whose computational cost is very high due to the large number of degrees of freedom that are taken into account. The objective of this project is to design and develop a tool capable of accurately estimate and predict the trajectory of a drone. This study is done from a macroscopic point of view in which factors such as the aerodynamics of the vehicle or its shape are not considered.

The first part of the thesis consists of making an exhaustive analysis of the behaviour of the drone in all flight stages of a trajectory. Throughout the development of the thesis, some flight plans have been designed in order for the drone to fly them and thus obtain the telemetry it generates.

The last part of the project consists of extracting a mathematical model from the data analysed that is capable of estimating and predicting the performance of a drone given a defined trajectory. The tool has been developed for a specific model of drone but, can be extrapolated to any other. By doing that, it allows this tool to have applications both in air traffic management, at drone air level, and to continuously monitor the performance of a fleet of drones.

Títol: Eina de predicció de trajectòries de drons

Autor: Daniel Gutierrez Herranz

Director: Enric Pastor Llorens

Data: February 8th 2022

Resum

Avui dia la tecnologia de microprocessadors i microcontroladors està evolucionant a gran velocitat, i això provoca que els drons puguin tenir més funcionalitats, tot això a un cost més competitiu. Aquesta situació ha derivat a que les operacions de drons hagin evolucionat des de fins purament militars a aplicacions en sectors comercials, com el transport i el lliurament de mercaderies. Com a conseqüència, la demanda de permisos per drons s'ha incrementat exponencialment durant la darrera dècada i això por suposar problemes en la gestió del transit aeri a baix nivell.

Hi ha models que expliquen el comportament d'un dron en els quals el seu cost computacional es molt gran a causa del gran nombre de graus de llibertat que tenen en compte. L'objectiu d'aquest projecte és dissenyar i desenvolupar una eina capaç d'estimar i predir amb precisió la trajectòria d'un dron. Aquest estudi s'ha realitzat des d'un punt de vista macroscòpic on no es tenen en compte factors com l'aerodinàmica o la forma física del vehicle.

La primera part del projecte consisteix en fer una anàlisi exhaustiva del comportament d'un dron en totes les fases d'una trajectòria qualsevol. Aquest estudi s'ha fet a partir de dades reals captades per un dron durant el vol dels diferents plans de vol dissenyats al llarg del projecte.

La segona part de la tesi consisteix en extreure un model matemàtic capaç d'estimar i predir el rendiment d'un don donada una trajectòria. Aquest model s'ha desenvolupat per a un model concret de dron, però es pot extrapolar per a qualsevol altre. Això permet que aquesta eina pugui estar present en aplicacions tant en la gestió del trànsit aeri, a nivell de dron, com per fer un seguiment continu del rendiment d'una flota de drons.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

Given the current rise of unmanned aerial vehicles, this project entails the development of an algorithm, to monitor the navigation parameters and internal system elements. This broad idea can be mainly divided into two sub-projects: an analysis of multiple flight plan data, in which, all the flight events are defined, identified and analysed and, a tool development process that predicts the behaviour using curve fitting methods and mathematical algorithms. This section will continue with a brief explanation about what is the motivation behind the project and which is the project structure.

0.1 Motivation

Air traffic management (ATM) research and development activities require modelling and simulation tools capable of replicating real-life operations and aircraft performances as realistic as possible. There are some models and tools that are used to simulate the behaviour of airplanes. For instance, at EUROCONTROL (European Organisation for the Safety of Air Navigation) has develop a tool called BADA (Base of aircraft data) (see [1]) that can model realistically the performance of any aircraft. This tool is based on a kinetic approach to aircraft performance modelling, which models aircraft forces. An additional example is a paper recently published by the University of Naples Federico II (see [2]) presenting a method developed to predict the flight time it takes for a drone to complete a planned route. All of this, by adopting and approach based on machine learning. The process it follows is quite similar to the one used in this thesis, a database is composed by a set of data from real flights and, after processing this information a prediction model is extracted. This article is focused on collision avoidance and air traffic management.

This project intends to develop a drone performance model that should capture the evolution in the operational performance of a vehicle, and in that way understand and predict the potential failure evolution of that particular vehicle. This tool does not take into account as many parameters as BADA does, because the analysis is done from a macroscopic point of view. That means that the aerodynamic and vehicle shape are not considered. As a result of this, the error assumption is higher. Moreover, the environment is not taken into account so, there will be always a surrounding noise that would affect the results. For that reason, it is considered that the conditions for each flight are practically the same. This could be improved by placing sensors along the drone to take information about the environment, such as the pressure or the temperature. Decreasing the number of input parameters increases the number of flights that can be processed and simulated simultaneously and the computational cost need for a simulation is considerately lower. The idea is to be able to process multiple flight plans at the same time.

0.2 Project structure

This project is divided into several chapters containing the information relative to the flight analysis and tool development. The chapters are organized as follows:

- The first chapter is the drone and State of the Art, whose aim is to provide an overview of the drone evolution and the current regulation.

- The next section contains all the information related to the communication protocol, used to exchange information between the drone and the ground station.

- Chapter number 3 describes the different flight stages that define a flight plan. In fact, a trajectory is combination of these events. Moreover, all the analysis and event identification process are explained. This is are the foundations on which the project is based to develop the simulation tool.

- The following four chapters enclose the simulation development tool and its conclusions for all the flight stages. From this interpretation, the flight models are extracted.

- In chapter number eight, the algorithm obtained from the analysis is testes using real flight and then, the error the overall error is computed.

- The last section is used to summarize and conclude, as well as some discussions about future improvements to the project.

- Two appendices are added to provide the flight plans used throughout the thesis and some information about the software developed.

CHAPTER 1. DRONE

1.1 Introduction to drones

This section provides a broad overview of drones and their implications, as well as, the drone evolution from its beginning to nowadays.

The first step to be introduced to this sector is to define the different meanings of drones, then a brief explanation about the evolution of drones from their beginnings to the present. Finally, the current regulation, and how it affect us in the development of the project is explained.

1.1.1 Drones and Unmanned Aircrafts

A drone is defined as an Unmanned Aircraft (UA), which means that it is an aircraft designed in order to operate autonomously or to be controlled remotely pilotless. It is an aircraft with its aircrew replaced by a computer, which instructs the components (i.e. rotors) to carry out a defined trajectory, and a radio link to communicate with the ground station or with other systems. An Unmanned Aircraft (UA) and the equipment to control, it is what is called an Unmanned Aircraft System (UAS). An Unmanned Aircraft System (UAS) is formed by three elements:

- An Unmanned Aerial Vehicle (UAV). It only involves the aircraft and neither the ground station nor the communication systems.
- An autonomous or a pilot-operated control system that is placed outside of the drone (i.e. at a ground station).
- Communication system to link the drone and the control system.

In recent years, the tendency to refer to any UAV as drone has been developed but the term is not universally considered as appropriate.

1.1.2 Evolution and State of Art

Drones, like much of the inventions in aerospace engineering, were devised as military applications. The first idea that can be related to the think nowadays it is known as a drone dates back to 1849, when the Austrian army bombed the city of Venice using balloons remotely activated by a long cable. These balloons flew without a pilot and the trajectory could not be fully controlled due to the air currents. That is why this cannot be considered as a drone but it was the pioneering idea.

Nowadays, the operations of drones have been evolved from purely military purposes to transportation and delivery. Taking into account that the technology of microprocessors and microcontrollers has evolved a lot in the recent years,

drones have great functionalities at a more competitive cost compared to a few years ago. That is why the demand for drone permits has been increased exponentially over the last decade.

The trend predicts that grow will continue over the next few years finding their way into others sectors such as agriculture, construction, mining, media and telecommunications.

1.2 Regulation

The fact that autonomous aircraft appeared, which anyone could buy and fly, has been totally atypical for the different administrations and aeronautical agencies around the world. To fight against this sudden growth in the use of low-level airspace, regulations and permits has been taken shape. Current drone regulation have undergone several changes in the last few years. Many challenges are related to the operative environment in which the drones fly and the employment of several on-board systems that are characterized by different performance.

In order to define the regulations, three categories have been defined (see [3]):

- 'Open' category. The maximum take-off weight of the drone must be lower than 25 kilograms. Autonomous operations is totally prohibited. Moreover, pilot has to maintain the visual line of sight with the vehicle during all the flight, being 120 meters the maximum flight height above the surface.
- 'Specific' category. The differences with respect to the previous class is that now the flight beyond the visual line of sight (BVLOS) is allowed, the maximum take-off weight can be higher than 25 kilograms and there is no ceil defined. In order to flight in these conditions, the drone operator need an operational authorisation from the National Aviation Authority (NAA), AESA (*Agencia Estatal de Seguridad Aerea*) in Spain, unless the operation is covered by a standard scenario. Which means that, there is visual line of sight (VLOS) and the maximum flight height is 120 meters over a controlled ground or, must be beyond visual line of sight (BVLOS) with the drone at a distance lower than 2 kilometres from the remote pilot with the presence of airspace observers and the maximum flight height is 120 meters over a controlled ground.
- 'Certified' category. Evolves the operations with the highest level of risk, as it could be the transportation of people. That is why these drones must be certified, the flight operator must need a pilot license and the approval of the competent authority.

Depending on the drone available and the manoeuvres desired to perform, one regulation or another will apply. The drone used during the project and on which the behaviour analysis is based is the '*dji MAVIC PRO*' (see [4]). The study has been done to analyse the behaviour of the drone when automatic pilot is enabled, that means that, the category that applies is the 'specific' one. Therefore, the operator must have an operational authorisation from the NAA.

CHAPTER 2. COMMUNICATION PROTOCOL. MAVLINK

2.1 Introduction to MAVLink

One of the protocol used to send commands and data between air vehicles and grounds station is the Micro Air Vehicle Link also known as MAVLink (ML) (see [5]). MAVLink is the most widely used messaging protocol for communication with drones (or its components). It is based on publish-subscribe design pattern, where the publisher does not know who will receive the information and the subscriber, who does not know where the information comes from, for the data flow (positioning, velocity messages etc.), and point-to-point to establish the mission and parameters protocols. The protocol only defines the way to share information between the ground station and the air vehicle so, the messages can be transmitted on different frequencies as long as both drone and ground station agree.

MAVLink is a very reliable protocol due to the fact that it provides methods for packet authentication, packet drops and corruption. Allows up to 255 simultaneous systems, including air vehicles, drone hardware (i.e. camera) or ground stations.

In the current version of MAVLink (MAVLink 2.3) each message ID is described by 24 bits. That means that, more than 16 million of different messages can be defined. In addition to the predesigned messages, the operator can define new ones, and this allows the system to be scalable so, new sensors and features can be incorporated.

There are three kind of messages defined (see [6]):

- MAVLink Enumeration. 124 different messages are described, in order to, define the type of firmware release and flags to report failures.
- MAVLink Commands. 155 different messages are described. Commands are used for actions in missions or to ask for actions in missions or for acknowledgment.
- MAVLink Messages. 216 different messages are described. Used to obtain information about the performance or status of the drone.

It is worth mentioning that to be able to read the message, it is required to have the appropriate hardware/peripheral.

2.2 Packet format

This section describes the format of the messages of the latest version (MAVLink 2.3). Each message is made up of several packets. The length of the packets is not a fixed value, it can range between 11 and 279 bytes depending on the information sent. On Figure 1, the packet format is shown:

The packed is composed by eleven fields:

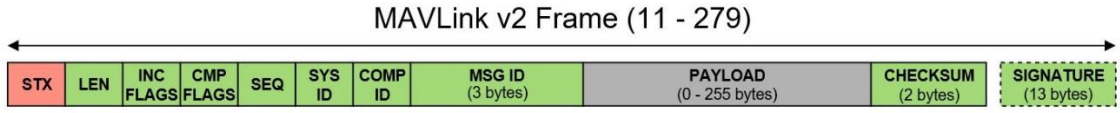


Fig 1 MAVLink packet

1. STX (Start of Text) [1 Byte]: Used to indicate the beginning of a new packet.
2. LEN (Length) [1 Byte]: Indicates the length of the payload.
3. INC FLAGS (Incompatibility Flags) [1 Byte]: Flags needed for MAVLink compatibility.
4. CMP FLAGS (Compatibility Flags) [1 Byte]: Flags to be ignored if not understood.
5. SEQ (Sequence Number) [1 Byte]: Used to detect packet loss. For each message sent, the value increases by one.
6. SYS ID (System ID) [1 Byte]: Identification number of the sender system.
7. COMP ID (Component ID) [1 Byte]: Identification number of the sender component (i.e. camera).
8. MSG ID (Message ID) [3 Bytes]: Identification number of message type. Needed to decode correctly the payload field.
9. Payload [0 - 255 Bytes]: Data field.
10. Checksum [2 Bytes]: Used to verify that the sequence received has no errors. Detects if there are errors introduced during the transmission or storage.
11. Signature [13 Bytes]: Optional field. To ensure that the link is made so that it cannot be interfered with or changed.

The maximum size of a packed is 280 bytes when the whole payload and the signed field are used. On the other hand, the minimum size of a packet is 12 bytes, neither the payload nor the signature fields are filled, which are used as acknowledgment messages.

2.3 Message set

The MAVLink protocol has more than 200 messages defined to obtain information about the performance of the flight. It is necessary to select correctly which set of messages is the most optimal to analyse the flight. In this chapter, the set of messages selected and its functionality are explained.

2.3.1 Global Position Int

The 'Global Position Int' message offers information about the real-time positioning of the drone. This message fuse the information of the multiples sensors of the system, such as the accelerators and the GPS. The velocities transmitted are defined with respect to the NED (North, East, Down) reference coordinate system, where v_x, v_y, v_z are established as positive when pointing north, east, down respectively.

This message is selected to be retransmitted and analysed due to the useful information it contains. It provides information about the positioning (latitude, longitude, altitude) and the ground speed (v_x, v_y, v_z) of the vehicle. It is very convenient having the altitude variable for the climb/descent analysis and the velocity for the acceleration/deceleration analysis.

It takes the information of 'GPS Raw Int' message and processes it in order to make it easier to work with. The fields the message contains are described in Table 1.

Table 1 Global Position Int fields

Field	Type	Units	Description
time_boot_ms	5 bits unsigned integer	ms	Timestamp[1]
lat	5 bits signed integer	degE7	Latitude
lon	5 bits signed integer	degE7	Longitude
alt	5 bits signed integer	mm	Altitude above mean sea level(MSL)
reltive_alt	5 bits signed integer	Mm	Altitude the ground
vx	4 bits signed integer	cm/s	Ground Speed x-axis

[1] Timestamp, also known as UNIX time, is the number of millisecond that have been elapsed since 00:00 UTC on January 1, 1970, regardless of leap seconds.

vy	4 bits signed integer	cm/s	Ground Speed y-axis
vz	4 bits signed integer	cm/s	Ground Speed z-axis
hdg	4 bits signed integer [0.0 – 359.99]	cdeg	Heading (yaw angle)

From these variables two more have been defined:

- Course: Needed to do the turns analysis. It is obtained from the x-axis Ground speed (v_x) and the y-axis Ground speed (v_y). Following this equation:

$$course = \arctan\left({}^{v_y}/_{v_x}\right)\ [rad] \quad \textbf{(1)}$$

- Horizontal speed: Needed to do the speed analysis. Defined as the modulus of the x-axis Ground speed (v_x) and the y-axis Ground speed (v_y).

$$hspeed = \sqrt[2]{{v_x}^2 + {v_y}^2}\ [m/s] \quad \textbf{(2)}$$

2.3.2 GPS Raw Int

The ‘GPS Raw Int’ message offers information about the real-time positioning of the drone. The main difference between this message and the ‘Global Position Int’ message is that this one offers the raw information of the GPS sensor, while the other fuse the information of all sensors. Table 2 shows the message fields and its description.

Table 2 GPS Raw Int fields

Field	Type	Units	Description
time_usec	6 bits unsigned integer	us	Timestamp
fix_type	3 bits unsigned integer	-	GPS fix type. Describes if the GPS is connected and its accuracy.
lat	5 bits signed integer	degE7	Latitude(WGS84)
lon	5 bits signed integer	degE7	Longitude(WGS84)
alt	5 bits signed integer	mm	Altitude above mean sea level(MSL)
eph	5 bits signed integer	-	GPS HDOP(horizontal dilution of position)
epv	4 bits signed integer	-	GPS VDOP(vertical dilution of position)
vel	4 bits signed integer	cm/s	Ground Speed y-axis
cog	4 bits unsigned integer	cdeg	Course over Ground
Satellites_visible	3 bits unsigned integer	-	Number of satellites visible
Alt_ellipsoid	5 bits unsigned integer	mm	Altitude
h_acc	5 bits unsigned integer	mm	Position uncertainty
v_acc	5 bits unsigned integer	mm	Altitude uncertainty
vel_acc	5 bits unsigned integer	mm	Speed uncertainty
hdg_acc	5 bits unsigned integer	degE5	Heading
yaw	4 bits unsigned integer	cdeg	Yaw in earth from north

2.3.3 VFR_HUD

This message provides information about the different speeds of the system and the current heading. The main idea of this message is to have the information fully processed to be used directly by the user (HUD). Table 3 shows the message fields and its description.

Table 3 VFR HUD fields

Field	**Type**	**Units**	**Description**
airspeed	float	m/s	Airspeed. The vector difference between the ground speed and the wind speed
groundspeed	float	m/s	Current ground speed
heading	4 bits signed integer	deg	Current heading [0(North), 360]
throttle	4 bits unsigned integer	%	Current throttle [0 to 100]
alt	float	m	Current altitude (MSL)
climb	float	m/s	Current climb rate

2.4 Message comparison

In this section the accuracy and fidelity of the different messages are studied. This comparison must be done before the analysis, in order to select properly the message and its fields so that, the estimation is done as accurate as possible.

The altitude parameter is compared in Figure 2(a). The information sent by ‘GPS Raw Int’, ‘Global Position Int’ and ‘VFR Hud’ messages has been contrasted. The drone receives the global position by the Global Positioning System (GPS) (‘GPS Raw Int’) then, this information is processed in order to estimate the actual global position of the system (‘Global Position Int’). So that, the user can work with this information in a more comfortable way, a second process is carried out where only the most relevant information is delivered to the operator using the international system of units. Note that, there is an altitude offset between the information sent by the ‘GPS Raw Int’ and the real one. This is because the ‘GPS Raw Int’ altitude is above mean sea level and the other messages provide the mean sea level altitude in addition to the WGS84 altitude.

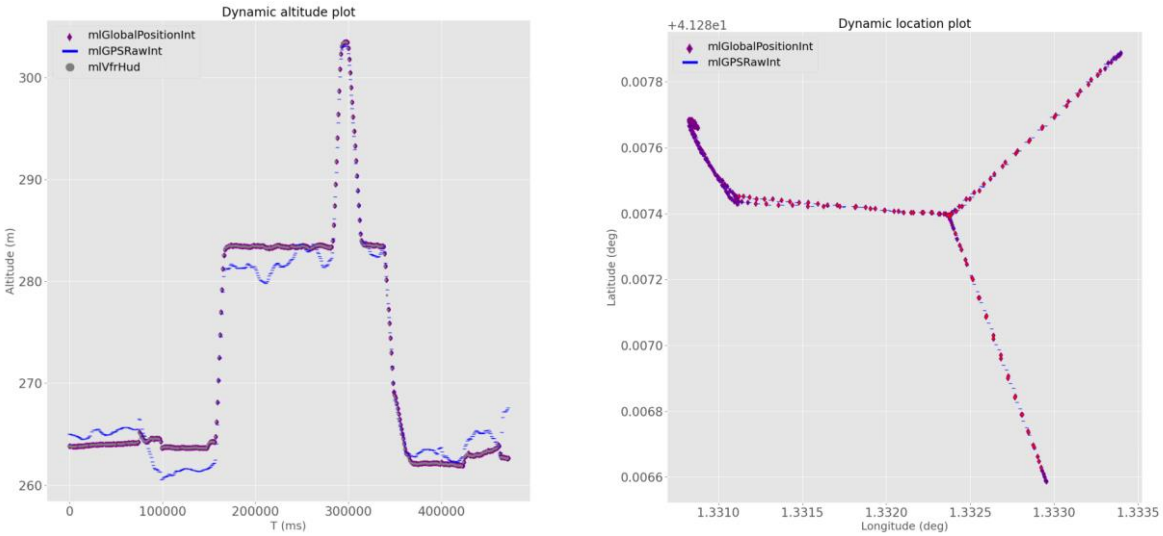


Fig 2 Altitude accuracy

Image 2(b) shows the differences in positioning. The ‘VFR Hud’ message does not contain latitude nor longitude information so, it cannot be included in this comparison. ‘GPS Raw Int’ and ‘GPS Raw Int’ messages send the same latitude and longitude information so the positioning is exactly the same. As the two messages are not sent in the same unit of time, it may seem that there is a difference in position, but this is not the case, it is due to the timestamp. Moreover, Global Position Int’ is the only message that contains velocity information on all three axes. Therefore, this message is the main one considered throughout the development of the project.

CHAPTER 3. DRONE PERFORMANCE TOOL DEVELOPMENT

As mentioned earlier, the purpose of the project is to analyse, monitor and identify the different flight stages, in order to, design a model that predicts the trajectory and behaviour of the drone, as accurate as possible in a short computational time.

This section describes in detail the procedure done to create and develop the drone performance analysis tool starting with an explanation about what a trajectory is and how to analyse it. The first analysis iteration has been done using a series of files with flight data provided by the project manager. During the analysis and the development of the tool, it was noticed that the provided data was not extensive enough for all desired events. Consequently, some flight plans were designed to do a second analysis iteration. In the next chapter, this is explained in detail.

3.1 Flight stages

The trajectories that perform the drones are defined by a consecutive set of waypoints. A waypoint is defined as a geographic localization that defines a point in space (x_o, y_o, z_o). The following flight stages have been defined with the aim of doing a detailed study of each one.

- **Initial stage:** The operator in charge of the flight moves the vehicle, from the point in which the system has been booted and configured, to the initial point of the trajectory. In this period of time, the system is sending information (i.e. positioning). As the movement is done by the operator freehand, some irregularities can be observed in the speed and altitude graphs.

- **Straight segment:** It is defied when the transition between two consecutive waypoints occurs in the horizontal plane, without height variation, and with a rectilinear movement.

- **Climb/Descent:** Encompasses the period in which the vehicle ascends or descends from a defined altitude to another one. This stage can be identified correctly analysing altitude data to detect changes. In another words, it can be calculated when there is a change in height and if it exceeds a defined threshold, the vehicle is considered to be climbing or descending. Regarding how the manoeuvre is performed, it can be divided into two:

    - **Vertical Take-Off/Landing (VTOL):** The system only moves on the vertical axis (z-axis). It is equivalent to a straight segment on the vertical axis.

- **Vertical and transversal Climb/Descent.** The system climbs/descent moving on different axes. The vertical speed depends on the elevation angle alpha (α). There is a maximum vertical speed defined by the capabilities of the system. If the needed vertical speed is lower or equal than the maximum one, the drone can perform successfully the segment. Otherwise, if the needed vertical speed is higher than the maximum one, the system will not be able to reach the desired altitude on the required point so, it will arrive later (delayed).

- **Turn:** This stage encompasses any turn or variation in course of the vehicle. This stage can be easily identified by analysing the course of the drone in each point of its trajectory. In another words, it can be calculated when there is a change in the course of the drone and if it exceeds a defined threshold, the vehicle is considered to be turning until the difference of course of a set of consecutive samples is lower than another defined threshold. This event usually occurs when the projection of a segment does not match with the following. Depending on how the vehicle handles the manoeuvre, two types of turn can be identified:

  - **Fly – by:** The turn occurs in advance, so that means that the vehicle does not cross the waypoint at any point. To do this, the system must calculate the transition from one segment to the following using the speed, position and direction of the drone.

  - **Fly – over:** The turn starts once the drone crosses the waypoint.

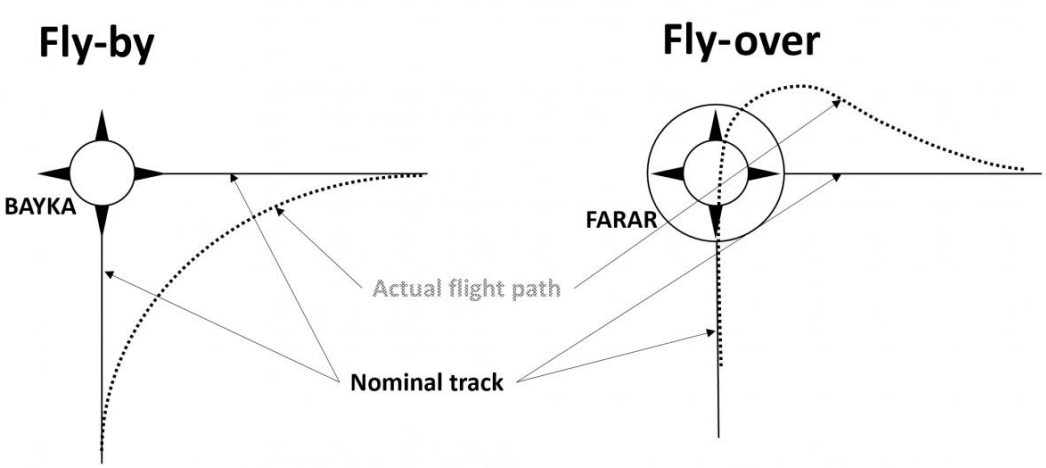


Fig 3 Fly-by and Fly-over

- **Hover:** It is defined as the process in which the system is held in a stable and static way in the air. It can be easily identified looking the variance of the latitude, longitude and altitude of the system. If they not change, or the variation is not big enough (considering perturbations due to the environment, as it could be the wind) the drone is considered to be in a hover stage.

The reason why this division is so important is because any flight can be defined as the combination of these events. Due to this, an independent analysis of each of these flight events has been made, with the exception of the straight segments, whose performance is affected by the events that precede and succeed them. Once this separate analysis has been carried out, the behaviour of a complete flight plan can be explained and computed.

3.2 Physical model

A good starting point when making an analysis is to develop a physical model, in order to get an idea about how the result should be. In flight plans, the trajectory that the drone must follow is defined by establishing waypoints. A trajectory can be defined as the concatenation of the segments, horizontal or vertical, formed by these waypoints, with the exception of the hovers which require to apply a delay. Next, a kinematic model is developed in order to explain the behaviour of an object that travels a defined segment.

Two different scenarios have to be identified. This applies both to horizontal and vertical segments. The first one, is the easiest one and happens when the drone reaches the defined speed. The other possible scenario is the opposite, when the drone cannot reach the defined segment speed. This is caused when the segment length is not long enough for the drone to reach the desired speed. It can also be interpreted as that the acceleration/deceleration that the system offers is not enough. That implies that, for each set of initial/final speed and accelerations, there is a maximum segment speed V_N. It corresponds to Equation 5, obtained from the uniformly accelerated motion formulas.

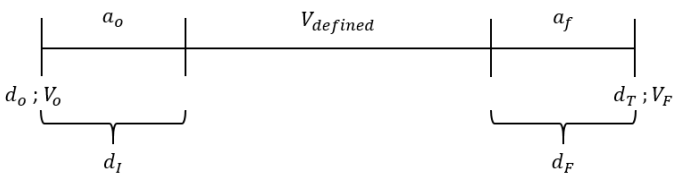


Fig 4 Scenario 1

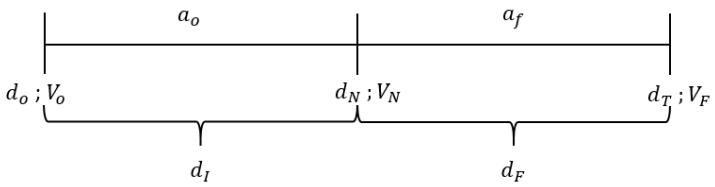


Fig 5 Scenario 2

Figure 4 shows the first scenario. It occurs when the drone has enough distance to accelerate/decelerate its velocity from the initial speed (V_0) to the defined segment speed (V_{def}) and be able to reach the final segment speed (V_F). On the other hand, scenario two, represented in Figure 5, occurs when the distance d_I is not large enough to reach the segment speed (V_{def}) starting from V_0 and taking into account that it will then have to accelerate/decelerate to reach V_F along the distance d_F.

Starting from the equation of a uniform accelerated motion, Equation 3, it is possible to extract the required time as a function if the speed and the distance to go from V_0 to V_{def} and from V_{def} to V_F. The resulting formula is expressed in Equation 7. The acceleration is considered to be constant.

$$d = d_0 + V_0 \cdot t + \frac{1}{2} \cdot a \cdot t^2 \quad \textbf{(3)}$$

$$V_F = V_0 + a \cdot t \quad \textbf{(4)}$$

$$V_F = \sqrt[2]{2 \cdot a \cdot \Delta x + V_0^2} \quad \textbf{(5)}$$

$$\Delta x = d - d_0 = \frac{V_F^2 - V_0^2}{2 \cdot a} \quad \textbf{(6)}$$

$$\Delta t = \frac{2 \cdot \Delta x}{V_F + V_0} \quad \textbf{(7)}$$

In the worst case, the initial and final horizontal speeds (V_0, V_F) would be equal to zero meters per second and the defined segment velocity (V_{def}), the maximum allowed by the system, 10 m/s for horizontal segments, 5 m/s for vertical climbs and 3 m/s for vertical descents. Flights are performed on autopilot so, the drone tries to reach the defined segment speed as soon as possible. Therefore, it is necessary to know what is the maximum acceleration and deceleration that the system can provide. In order to do this, as this information does not appear in the datasheet, the different real flights have been analysed. As a consequence, it is concluded that both the maximum acceleration and deceleration applied by the drone is 1.5 meters per second per second for horizontal segments and 1.0 meter per second per second for the

vertical ones. Knowing a_{max} and V_{max}, the required distance and time can be calculated using Equations 6 and 7. Solving this equations, it is obtained that the required distance to accelerate from zero to the maximum segment speed is 33.33 meters for horizontal segments, 12.5 for climbs and 4.5 for descents, which corresponds to 6.6, 2.5 and 0.9 seconds respectively. For decelerations, the computed values are the same. Note that this only applies when the segment defined speed is the maximum and the initial/end speeds are equal to zero. In Table 4 this information is summarized.

Table 4 Maximum segment speed and length

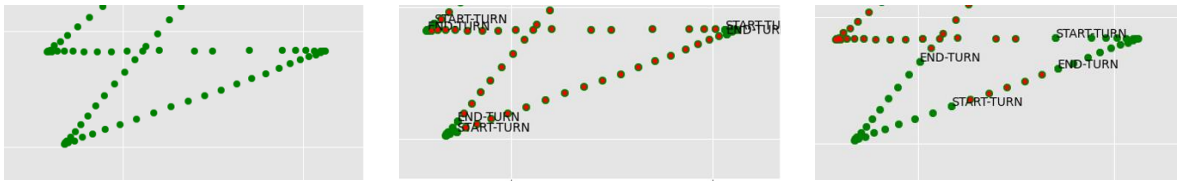
Event	Δx to reach V_{MAX}	Δt if a_{MAX}
Climb	12.5 m	6.6 s
Descent	4.5 m	2.5 s
Horizontal segment	33.3 m	0.9 s

3.3 Analysis of flight stages

In this section, the different functions created to split and identify the different stages of the flight are explained.

First of all, each of the phases of any flight must be correctly identified. This stages have been defined in the previous chapter, which are: initial stage, straight segment, climb/descent, turn and hovers. As there is a large amount of data that the system sends to the ground station, it is necessary to know how to identify which messages are needed and if they are valid for the study.

The analysis of the data is based on having a work window, where the samples enter and leave chronologically and, depending on their parameters and how the vary and interact between then over time, the events are defined. A time limited circular buffer is used in order to process the data obtained. The buffer has no maximum length but, the data contained in it cannot differ more than a predefined time variable. This time threshold cannot be too big because any error fluctuation could be identified as an event and, cannot be too small because some events, as sharp turns, could not be correctly identified. Originally, it was defined to three seconds and a half but, after some analysis iterations it had to be increased to four second to correctly identify all the events. The drone transmits one sample per second so, ideally each second a sample should enter in the window but, due to the fact that the drone can be flying away or for some external interferences, this samples input value can vary. The following set of Images, Figures 6(a), 6(b) and 6(c), shows the event detection variability between setting difference thresholds. It can be seen how when the buffer is very small, no events are detected, while when it is very large, the beginning and end of the events are not accurately detected.



(a) Buffer 2500 ms

(b) Buffer: 3500 ms

(c) Buffer: 9500 ms

Fig 6 Buffer threshold

So, in conclusion, the working window is composed by all the samples the system receives in 3.500 milliseconds. When a start of an event is detected, the oldest sample in the window (the one placed in the head position of the buffer) is considered to be initial time of the event. On the other hand, when an end of an event is detected, the newest sample in the window (the one placed in the tail position of the buffer) is considered to be final time of the event.

3.3.1 Altitude change analysis

The first event studied was the climb/descent. The altitude change study has been done analysing the variation of the altitude in each point of the flight.

As explained previously there are two ways an air vehicle can perform a climb/descent (vertical climb/descent, and vertical and transversal climb/descent). Regardless of which of the two the drone is performing, the identification process is the same.

An altitude change is contemplated when the difference in altitude between the oldest and the newest sample in the buffer is greater than or equal to a defined threshold. A trade-off must be found between a maximum threshold and a minimum one. The maximum value must be large enough so that it does not detect any climb or descent caused by disturbances produced by the surrounding or, by the operator when placing the drone at the starting point. Nevertheless, the minimum value of the threshold must be set in order not to miss any climb or descent performed at low climb rate by the drone. After analysing a several files with flight data, the threshold was established to 3 meters. The first climb event identified in each flight is the take off and the last descent is the landing. During the first analysis iteration, there was a lack of samples so, some new flight plans have been defined in order to obtain a high amount of samples to be able to correctly obtain a behaviour model.

The main purpose of the altitude change analysis is to find the curve/equation that better fits the behaviour of the drone in order to obtain a model that predicts the performance in future flights. First of all, a plot of altitude changes of each flight has been done to clearly identify that the climb/descent events have been

correctly detected. It is called dynamic altitude plot because it is generated at the same time the events are detected, Figure 7.

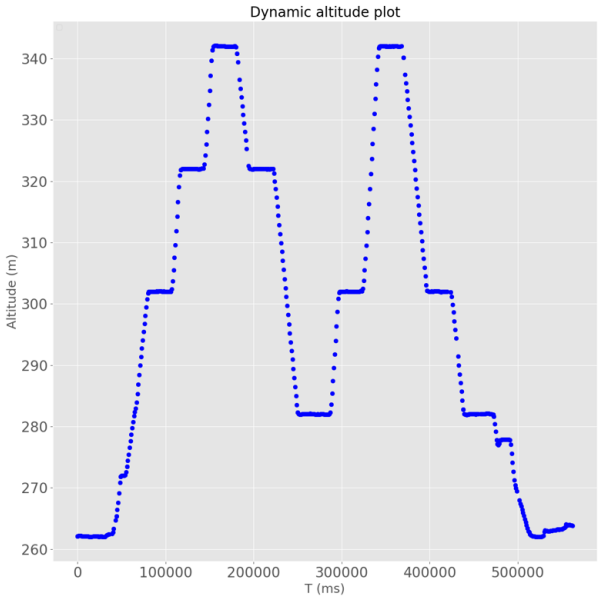


Fig 7 Altitude variation

Image 7 shows that all the altitude changes are accurately detected. Before the take-off, it can be seen that there are some altitude changes, which indeed, are not identified as any kind of event. It corresponds to the manoeuvres done by the operator to place the drone in the desired starting point. It occurs the same at the end of the flight, after the landing. Due to the tuning process of the windows size parameter, the system can identify correctly sudden climbs and descents. Once the events are correctly detected and listed, the process to develop a model, that predicts the behaviour when an altitude variation event occurs, can begin.

3.3.2 Hover analysis

The simplest events to analyse are the hovers. They are defined as the process in which the system is held in a stable and static way in the air. So, the way to identify them is to examine the variation of latitude, longitude and altitude. If they not change, or the variation is not big enough (considering perturbations due to the environment, as it could be the wind) it is considered that the drone is on hover stage. Hovers are processed with priority with respect turns due to the course uncertainty. While the hover is occurring, the course could vary and it cannot be detected as a turn.

A hover is defined when the total speed difference with respect to the previous sample is lower to a threshold speed limit. This threshold must be set low enough to not misidentify any manoeuvre performed at low speed and high

enough to not detect multiple hovers when the vehicle is static due to external conditions as it could be the wind or GPS errors. After a tuning process, it is set to 0.5 meters per second.

3.3.3 Turn analysis

The most complex analysis is the one done for the turns. That's due to the fact that a drone can perform different kinds of turns, fly-by and fly-over, according to the trajectory it follows, and drift and driftless turns, according to the rotation the vehicle follows when taking the turn. A fly-by occurs when the turn occurs in advance, so that means that the drone does not cross the waypoint at any point. Meanwhile, a fly-over occurs when the turn starts once the drone crosses the waypoint. Otherwise, it is considered that the drone is drifting when it takes a bigger turn than expected and it has to rectify the heading. By doing a tuning process, the threshold that the excess turn must exceed to be considered a drift is a 10% of the total turn angle. All this turn variations will be taken into account in Chapter 5, where the turn analysis is done in depth.

A turn is defined when the course difference between the oldest and the newest sample in the buffer is greater than or equal to a defined threshold. A trade-off must be found between a maximum threshold and a minimum one. The maximum value must be large enough so that it does not detect any turn caused by disturbances produced by the surrounding or, by the operator when placing the drone at the starting point. Nevertheless, the minimum value of the threshold must be set in order not to miss any turn performed by the drone. After analysing a several files with flight data, the threshold is established to 15 degrees to identify the start of the turn and 5 for the end. Ideally, a machine learning program would analyse the different turns and, in this way, the threshold would not be a fixed predefined value set by the operator and could be adapted to the different types of turns. Figures 8a, 8b and 8c show two turns of 40 degrees, the flight corresponds to the nonagon polygon, Figures 76 and 77 (Appendix A.2). It can be seen how when the threshold is small, not all turns are detected, while when it is large, no turns events are detected because the turn is smaller than the start turn threshold.

(a)
Start Turn delta: 5 º
End Turn delta: 1 º

(b)
Start Turn delta: 15 º
End Turn delta: 5 º

(c)
Start Turn delta: 45 º
End Turn delta: 15 º

Fig 8 Turn identification

This method do not detect accurately the beginning and the end of the turns. In order to solve this issue and refine the turn event detection, another post-processing analysis has been done. Once the turns are detected, the process consist on check which is the contribution of each sample involved on the turn in the total angle of rotation. If it is lower than five percent of the total turn, the contribution is considered to be negligible. This process is repeated with the consecutive samples of the flight event, until the defined threshold is exceeded and, this is the sample which is considered the start of the turn. The way to obtain the refined end of turn is the same but starting by the end of the event.

In Images 9 and 10 (Longitude vs Latitude), two flight can be observed, whose trajectories are absolutely different.

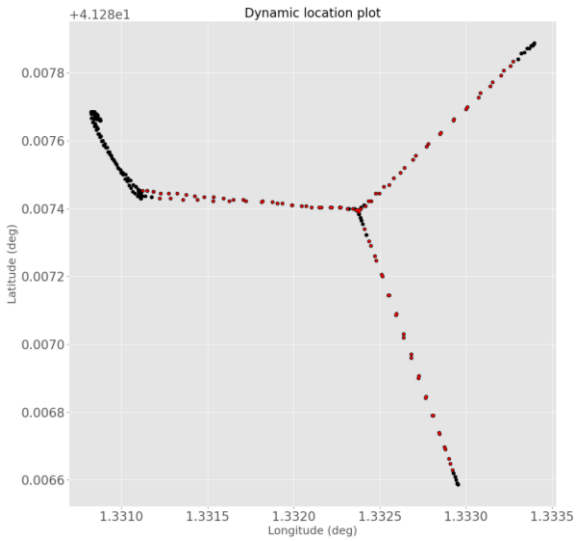


Fig 9 Turn Refinement FP1

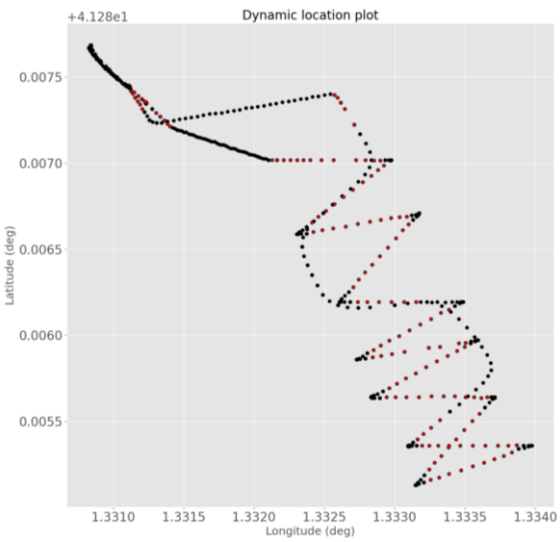


Fig 10 Turn Refinement FP2

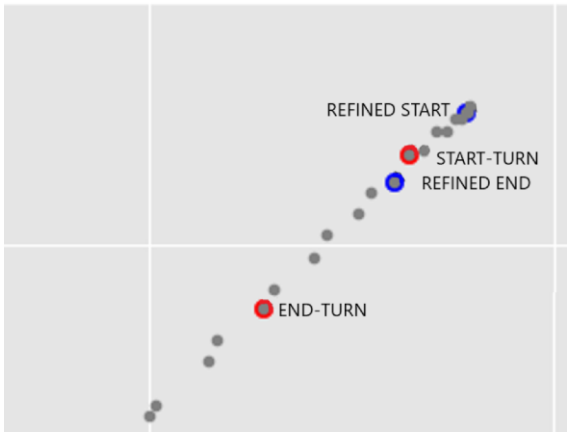


Fig 11 Turn Refinement Zoom 1

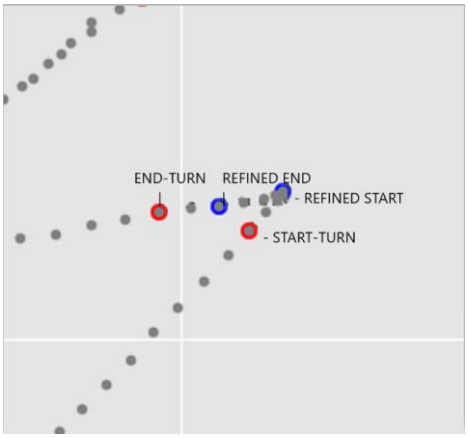


Fig 12 Turn Refinement Zoom 2

b

Refering to Figure 11 and 12, the refinement post-process improves the accuracy at turn event detection. The main purpose of the turn analysis is to find a curve/equation that better fits the behaviour of the drone in order to obtain a model that predicts the performance in future flights. So, the more accurate the detection, the more precise the model obtained will be.

3.3.4 Straight Segment analysis

Finally, the straight segment analysis has been done. Once, the study and the identification of all the events have been done, it can be correctly recognized that there is a dependency with the previous and consecutive events.

So as to achieve a set of consecutive samples to be identified as a straight segment, they have to fulfil certain conditions. The first one is that the altitude must not vary more than a defined threshold (same that the used to the altitude change analysis, 3 meters) along the segment. The beginning of a straight segment can be recognize if the previous event is one of the following:

- End of course change
- End of altitude change, excluding end of landing.
- End of hover.

Otherwise, the end of a straight segment should be one of the following events:

- Start of course change.
- Start of altitude change, climb or descent.
- Start of hover.

This constraint is the reason why the straight segment analysis process is done the last, it depends on the previous events identification. The last constraint that must be taken into account before the straight segment is established is that the total length must be higher or equal than a defined threshold (set to 5 meters). This restriction is imposed in order to avoid misidentifications of straight segments due to the short period between the drone ends a manoeuvre (event) and the start of the following. To give an example, the drone has to do a climb to a certain altitude and just after it must start a turn, if this limitation is not set, it could be considered as a straight segment when in fact there is not.

Once the segments are correctly identified, the analysis can begin. The first approach done in order to classify the different straight segments was to pinpoint the parameters that defines a straight segment. These are the total length, initial, final and segment speed and the type of event that precedes and follows the segment. The speed at which the segment starts defines the time the drone would need to reach the established segment speed. The behaviour at the end of the segment with respect to the final speed is analogous. This confirms the idea of making the division according to which are the flight stages that precedes and succeeds the straight segment.

3.4 Mission Planner simulation tool

With the aim of making a detailed study of all flight stages, it is necessary to have enough data from flights where all possible combinations occurs. The ideal would be to fly the different flight plans designed along the project with a real drone, but due to the expense of resources that this would entail, a mixed method has been implemented, where the use of data from real flights is combined with the use of a flight simulation tool.

The tool used is 'Mission Planner', it is an open-source software designed by ArduPilot Development Team (see [7]). It supports the MAVLink protocol for communication with Ground Stations. That means that the same flight plans used to fly the real drone can be also simulated using the tool. The use of this tool gives the possibility of being able to design a flight plan to study the effect of any type of parameter, such as be capable of iterate the same flight plan multiple times by changing speeds. All designed flight plans are in appendix A.2.

This software allows the user to tune any type of parameter which is useful to optimize the performance of the drone. The parameters that have been modified and tuned during the development of the thesis are the following:

- Waypoint acceleration (WPNAV_ACCEL). Defines the horizontal acceleration in cm/s/s used during missions.

- Waypoint vertical acceleration (WPNAV_ACCEL_Z). Defines the vertical acceleration in cm/s/s used during missions.

- Waypoint horizontal speed target (WPNAV_SPEED). Defines the speed in cm/s which the aircraft will attempt to maintain horizontally during a mission.

- Waypoint descent speed target (WPNAV_SPEED_DN)._Defines the speed in cm/s which the aircraft will attempt to maintain while descending during a mission.

- Waypoint climb speed target (WPNAV_SPEED_UP)._Defines the speed in cm/s which the aircraft will attempt to maintain while climbing during a mission.

The process of tuning the accelerations and speeds allows the model to be extrapolated to other speeds configurations. Moreover, the versatility the software provides during the designing of different events made the tool very useful during the development of the project. With the help of this tool, a series of flight plans have been designed for the study of all flight phases. These flight plans are found in Appendix A.2.

CHAPTER 4. CLIMB/DESCENT ANALYSIS

In the following four chapters, the analysis processes of the different events that make up a flight are explained extensively. The study has been carried out by previously defining a set of parameters. The ascent speed is 2.5 meters per second, the descent speed is 1.5 and, the waypoint speed, the horizontal one, is set to 10 meters per second.

This section explains in detail the analysis that has been made for the climb/descent events. The analysis has been carried out taking into account information sent by the drone in each point of the flight. The performance and therefore the behaviour of the drone at taking off or, at landing is not the same done at climbing or descending. So, they will be analysed separately.

As it is explained previously, there are two different kind of climb/descent. The simplest one is the vertical climb/descent, where drone only moves on the vertical axis, the climb/descent is done in a two dimensional plane. There is also the non-vertical climb/descent, where the horizontal axis must be taken into account.

After doing a first analysis, it was noticed that the amount of data recorded was not high enough to obtain a model that describes the behaviour for all climbing/descending altitude variance regions. The ideal solution would be to design multiple flights with the desired trajectories and events, record the data sent by the drone during the flight and iterate the analysis process. Owing to reasons external to the project, this could not be done. As a workaround, a MAVLink simulation tool is used in conjunction with the real data flight to cover the entire spectrum of analysis. The tool used is ‘Mission Planner’, it is an open-source software designed by ArduPilot Development Team. The focus of this idea is to design different flight plans where the drone performs a trajectory with multiples climb/descent of different altitude variations. To do that, a random flight plan generator was planned. The reason why it has been done by means of a code is so that the flight plan is totally random and contains all types of altitude variations necessary to be able to do a complete study of climb/descents.

The main idea behind this action, is to have two data sets. One of them used solely and exclusively for the analysis and development of the trajectory’s model, composed by real and simulated data. However, the other set of data, is only used to check and verify that the extracted model represents the real behaviour. This provide more information and represents the real performance of the drone.

This flight plans generated for the analysis are designed following a climb/descent sequence, in which the only parameter that vary is the altitude interval. Figure 13 shows an altitude variation plot of a flight plan analysed in this section.

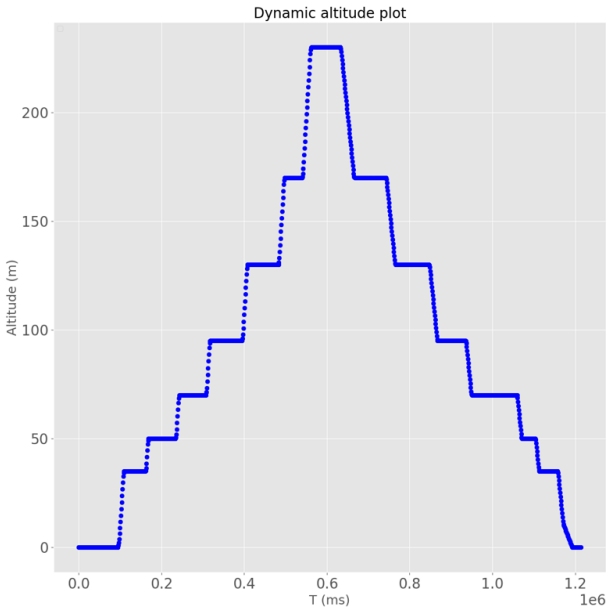


Fig 13 Altitude variation flight plan

In addition, an altitude change event refinement has been done. This process consist in making the beginning and end of the event more accurate, as has also been done with course events. To do that, the contribution of each sample to the climb/descent has been taking into account. If the impact of the sample to the total altitude variance is lower than a defined threshold, five percent of the total height variation for climbs and ten percent for descents, it is considered that the sample does not contribute to the event. This process is repeated for the following samples until the point, where the contribution is higher than the threshold, is found.

4.1 Vertical Climb

The parameter that defined the performance of a vertical climb is vertical speed. Therefore, the focus has to be on how the vertical speed evolves throughout the climb as a function of the altitude to ascend.

Analysing Figure 14, each point represent a climb, where the x-axis is the climb altitude variance (Δh) and, the y-axis is the time it took to climb the defined altitude (Δh). It can be seen that there is a linear relation between this two variables and it follows Equation 8. It can be predicted how much time the drone would take to climb a certain height using the equation.

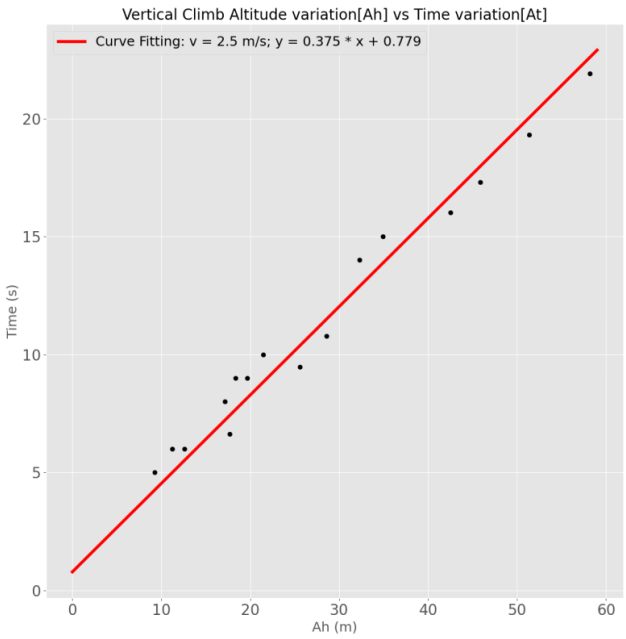


Fig 14 Vertical climb Δh vs Δt

$$\Delta t = 0.375 \cdot \Delta h + 0.779 \quad \textbf{(8)}$$

Analysing Equation 8, it can be seen that there is a linear relationship between the distance it travels and the time elapsed. The variable that relates these two parameters is vertical speed. The behaviour is as if the drone climbed at 2.666 meters per second (0.375^{-1}), but considering that there is a time lag. This value is very close to the one at which the study has been made (2.5 m/s). The cause of the observed time offset is due to the fact that the movement is not performed at constant speed, there is an acceleration and braking stage, which adds extra time.

Doing a deeper analysis of the evolution of vertical speed during a vertical climb, it can be observed that in any climb it tries to achieve a certain speed. In Figure 15, it is shown an example where the drone reaches this certain speed (2.43 m/s), it remains constant until it needs to decelerate because the altitude change hits the desired altitude. This altitude variation is defined in the simulator and that, verifies that the drone reaches the defined climbing speed. Image 15 represents the horizontal distance travelled (x-axis) versus the altitude distance travelled (y-axis) and, the number at the right of the samples is the vertical speed, in meters per second.

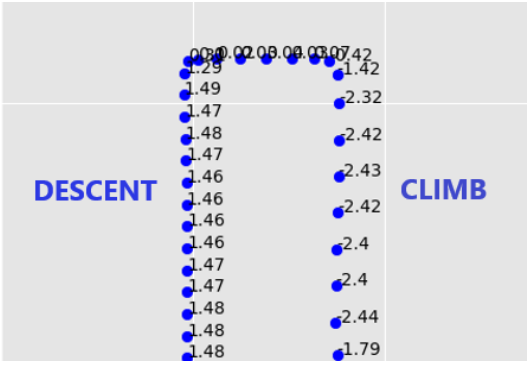


Fig 15 Vertical speed evolution

Once the equation have been extracted, it is needed to verify if it describes the behaviour of the drone during a vertical climb using the second data set. It can be observed, in Figure 16, that the slope of the Δt-Δh equation is the same but, an offset of 2.0 seconds appears. The reason of adding this offset may have several root causes, for instance the influence of external factors or the fact that the horizontal and vertical speeds changes along the flight. This analysis is done taking into account that the horizontal and vertical speed does not change. As the process is repeated for several speeds, an equation is obtained for each speed so, if this is the root cause it will disappear after the model extraction.

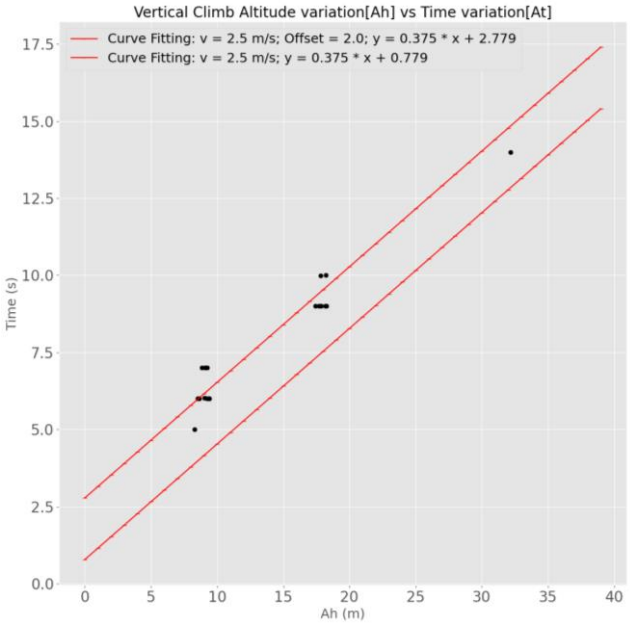


Fig 16 Vertical Climb real Offset

4.2 Vertical Descent

As is happens in the vertical climb event, the main parameter to study is the evolution of the vertical speed so, the process followed to obtain the mathematical expression is the same.

Figure 17 shows how much time (Δt) it takes a drone to descent vertically a determined altitude height (Δh). There is a linear relation between this two variables (Δh- Δt) and its behaviour is described by Equation 9.

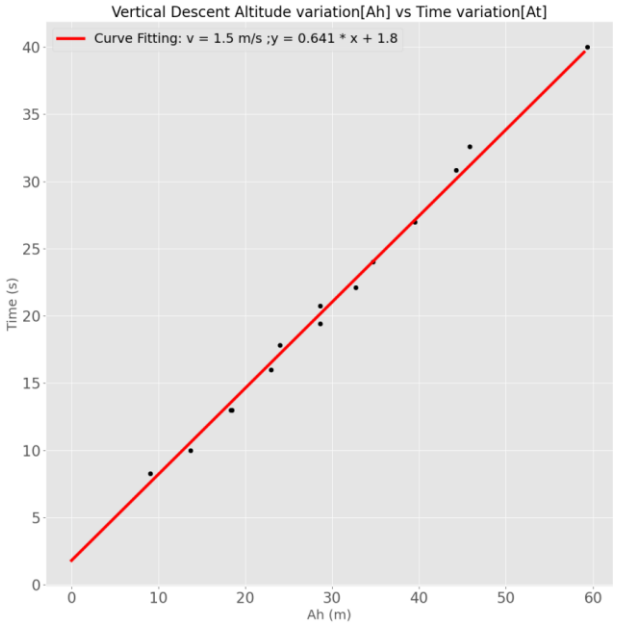


Fig 17 Vertical descent Δh vs Δt

$$\Delta t = 0.641 \cdot \Delta h + 1.80 \quad \textbf{(9)}$$

The evolution of the vertical speed along the descent is quite similar to the vertical climb. It starts the descent with a determined vertical speed, it accelerates/decelerates to reach the descent speed and then, it accelerates/decelerates to hit the vertical speed set for the following event. As it is defined in the flight plan, the vertical speed the vehicle tries to reach is 1.5 meters per second.

While analysing Equation 9, it can be seen that there is a linear relationship between the distance travelled and the time elapsed. The variable that relates these two parameters is vertical speed. The behaviour is as if the drone descended at 1.560 meters per second (0.641^{-1}), but considering that there is a time lag. This equivalent speed is very close to the one at which the study has

been made (1.5 m/s). The cause of the observed time offset is due to the fact that the movement is not performed at constant speed, there is an acceleration and braking stage, which adds extra time.

4.3 Non-Vertical Climb/Descent

The reason why the analysis of not fully vertical altitude changes has been more complex is due to the influence of more parameters than the vertical one. These variables are the altitude variance, longitudinal variance, vertical and horizontal speed. That's why, a three dimensional plot has been done to represent the influence of these variables on each other. Figure 18 shows the behaviour of the drone at doing a non-vertical climb, while Figure 20 describes how the drone performs a non-vertical descent.

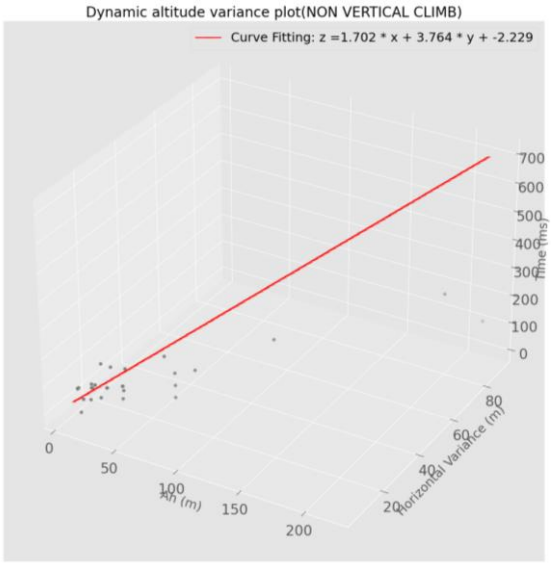


Fig 18 Non-Vertical Climb

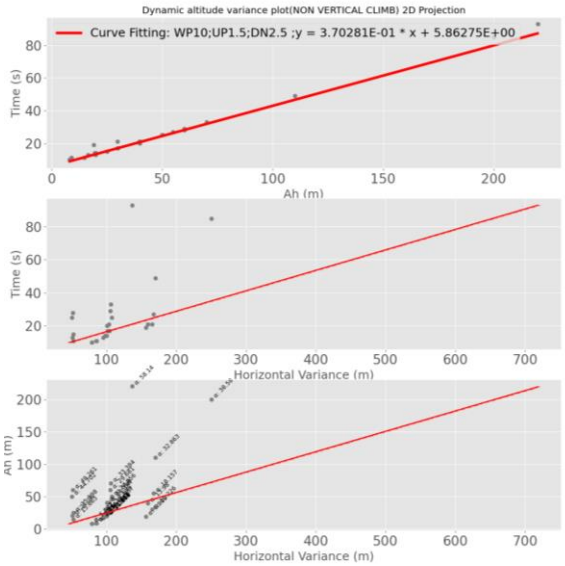


Fig 19 Non-Vertical Climb 2d projection

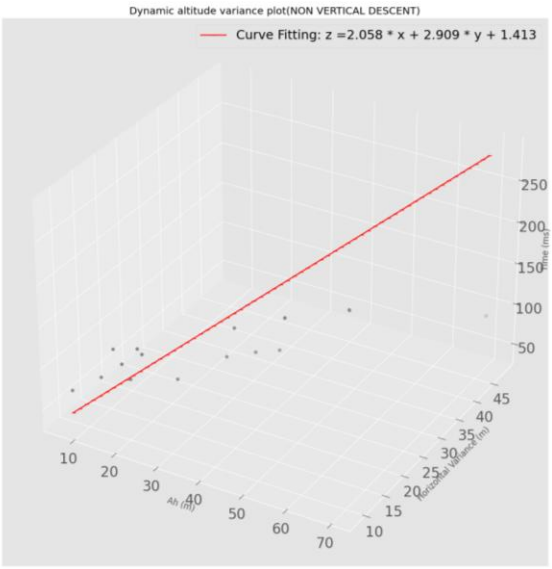


Fig 20 Non-Vertical Descent

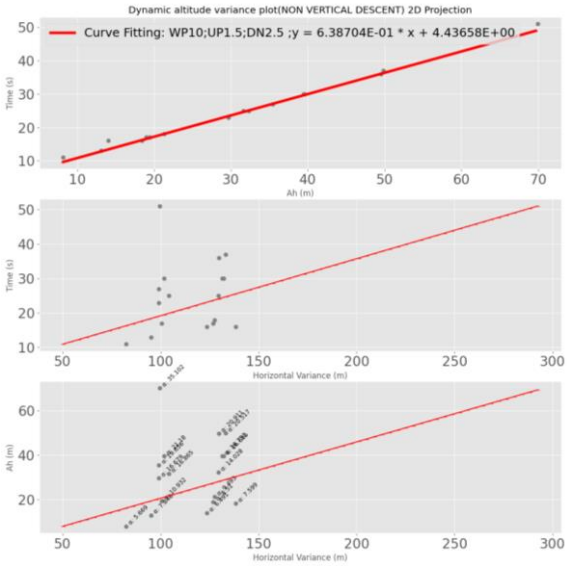


Fig 21 Non-Vertical Descent 2d projection

In the 3d representations (altitude variance vs. horizontal variance vs. delta time), it can be observed that there is a linearity between the vertical distance to climb/descent and the elapsed time. This follows the Equations 10 for climbing and 11 for descending extracted from the 2d projection, Figures 19 and 21.

$$\Delta t = 3.70281 \cdot 10^{-1} \cdot \Delta h + 5.8627 \quad \textbf{(10)}$$

$$\Delta t = 6.38704 \cdot 10^{-1} \cdot \Delta h + 4.43658 \quad \textbf{(11)}$$

After having been simulating and studying this behaviour, it can be stated that: when the time needed to ascend/descent the altitude is five times lower than the required to travel the horizontal length, the event is considered as a straight segment, because the altitude variation with respect to the horizontal is negligible. That means that, the delta time is computed applying the corresponding formula, described in Chapter 7, taking into account which are the previous and subsequent type of event. If the time needed to ascend/descent the altitude is eight times higher than the required to travel the horizontal length, the event is considered as a vertical climb/descent. If horizontal and vertical speeds are the same, time conditions can be translated to angle conditions. When the angle of climb is higher than 76 degrees it is considered to be a vertical climb/descent and, when it is lower than 7 degrees, it is considered to be a straight segment. Otherwise, the previous equations (10, 11) are applied. So, since the scope of this analysis is to obtain the delta time as a function of some parameters, Equations 10 and 11 are the used ones to

explain and describe the behaviour of the drone in non-vertical altitude changes events. Image 22 corresponds to one of the flight plans designed to study this kind of event. The solid continuous line corresponds to the simulation tool and the dashed to the statements explained.

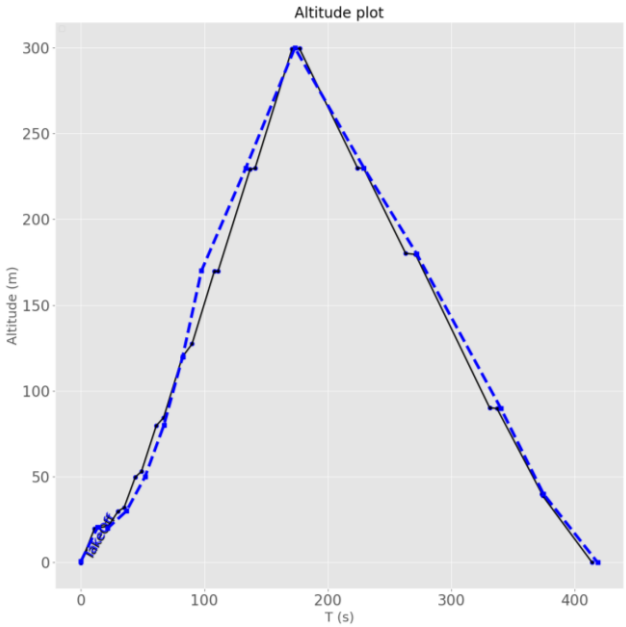


Fig 22 Non-Vertical Altitude Changes

4.4 Take-Off

The performance of a take-off is broadly similar to a vertical climb. The behaviour is exactly the same but the equation that describes the velocity is not the same. For take-offs, the curve that better fits the vertical speed evolution is shown in Figure 23. As occurs with the last events analysed, there is a linear relationship between the take-off altitude and the time required to perform this manoeuvre. Using a curve fitting method, Equation 12 describes the curve that explains its behaviour.

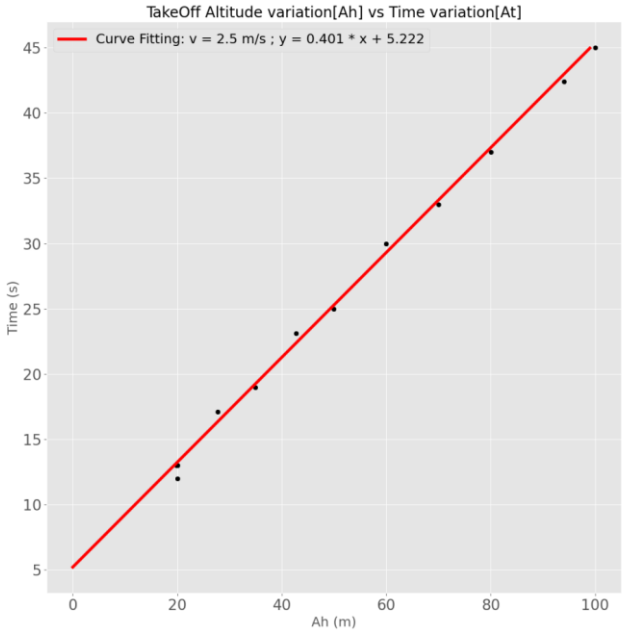


Fig 23 Take Off Δh vs Δt

$$\Delta t = 4.01400 \cdot 10^{-1} \cdot \Delta h + 5.22203 \quad (\mathbf{12})$$

While analysing Equation 12, it can be seen that there is a linear relationship between the vertical distance and the time elapsed. The variable that relates these two parameters is vertical speed. The behaviour is as if the drone climbed at 2.491 meters per second (0.4014^{-1}), but considering that there is a time offset of 5.22 seconds. This equivalent speed is very close to the one at which the study has been made (2.5 m/s). The cause of the observed time offset is due to the fact that the movement is not performed at constant speed, there is an acceleration and braking stage, which adds extra time.

4.5 Landing

A landing is indeed a vertical descent. Owing to the big similarities between them, it can be erroneously affirmed that the performance is the same. The model extracted from the analysis is the one represented in the Figure 24 and Equation 13.

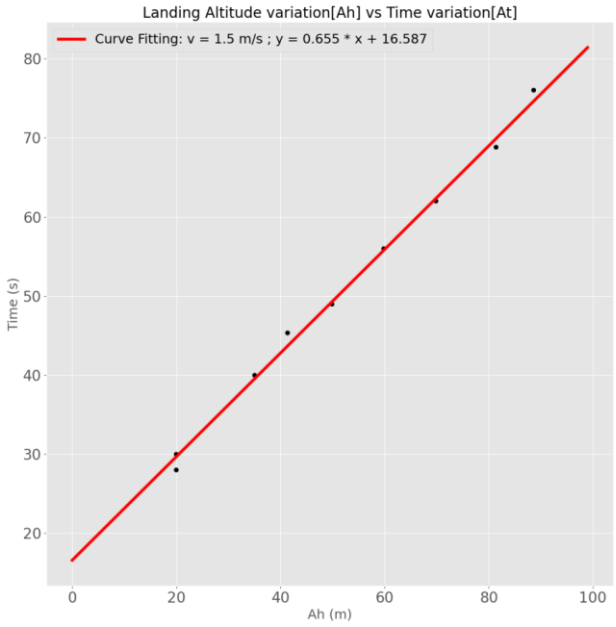


Fig 24 Landing Δh vs Δt

$$\Delta t = 6.54630 \cdot 10^{-1} \cdot \Delta h + 16.587 \quad (\mathbf{13})$$

After an extensive research, two different stages can be identified. The final segment, which corresponds to the last ten meters of descent, is performed at a constant vertical speed of 0.49 meters per second. While, the first one, which is defined from the initial altitude to ten meters above the ground is performed at 1.48 meters per second, value defined in flight plan. In Figure 25 the ascent profile of a studied trajectory is shown, where the x-axis corresponds to the altitude and, the y-axis is the time it took to climb the defined altitude. This is the reason why in the ΔhvsΔt figure some samples are shifted from the model equation. So, the best approximation that can be done is to consider the descent to an altitude of 10 meters at a constant vertical speed of 1.50 meters per second and then, the last approach at 0.49 meters per second. Despite the landing can be divide in this two stages, Equation 13 is used to calculate the time because the precision obtained is great enough, as long as, the landing altitude is greater than twenty meters. If the altitude variance is lower, this sub stage is taken into account.

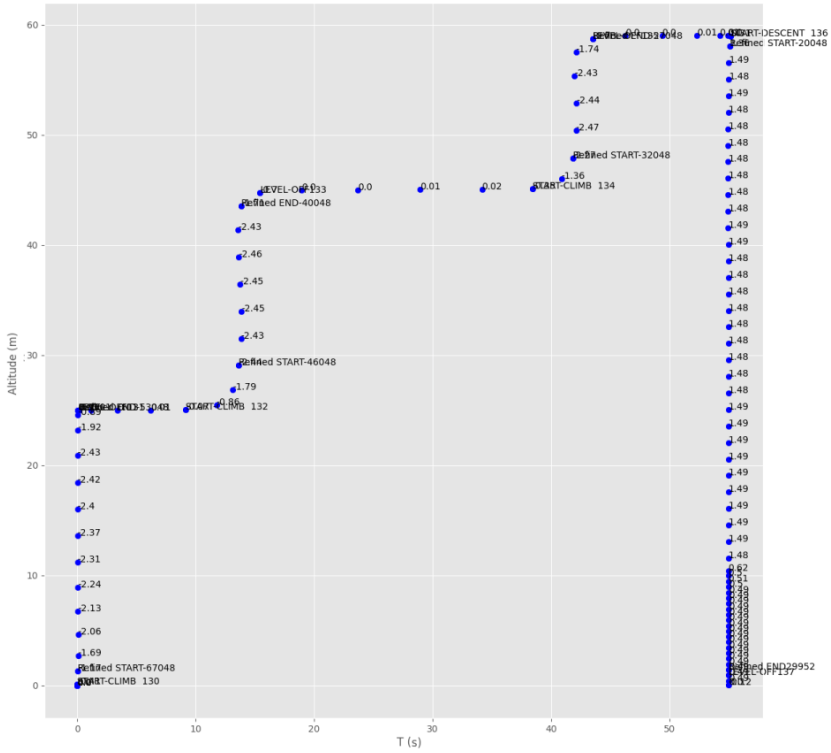


Fig 25 Altitude variation

While analysing Equation 13, it can be seen that there is a linear relationship between the distance descended and the time elapsed. The variable that relates these two parameters is vertical speed. The behaviour is as if the drone descended at 1.528 meters per second (0.065463^{-1}), but considering that there is a time lag of 16.58 seconds. This equivalent speed is very close to the one at which the study has been made (1.5 m/s). The cause of the observed time offset is due to the fact that the movement is not performed at constant speed, there is an acceleration and braking stage, which adds extra time or, produced by the huge difference speed in the second landing stage.

All in all, an equation has been obtained for each type of altitude change event. This equations shows a linear relationship between the vertical distance ascended/descended and the time elapsed. The variable that relates these two parameters is vertical speed. The inverse of the slope of these curves defined an equivalent vertical speed. It is as if the drone performed the segments at this speed but adding an extra time (offset). The value of the equivalent speed is in all cases around the defined one, 1.5 m/s for descends and 2.5 for ascends, Table 5. The offset is different between all cases due to the fact that speed is not constant.

Table 5 Altitude Changes Equivalent Speed

	Vertical Climb	Vertical Descent	Not fully Vertical Climb	Not fully Vertical Descent	Take-Off	Landing
Equivalent Speed (m/s)	2.666	1.560	2.700	1.565	2.491	1.527
Offset (s)	0.779	1.800	5.863	4.436	5.222	16.587

4.6 Speed variation

The model extracted is absolutely valid as long as the descent speed is 1.5 meters per second and the ascent speed is 2.5 meters per second. If the flight plan that is going to be studied has to be flown using other speeds, this model will not be accurate and the results obtained will be incorrect. The solution is to know how the equations obtained vary with the speed. To achieve this, the same flight plans used to extract the previous model have been simulated with several speeds. As the equations obtained are of the first order, it is necessary to see how the progression of the slope of the line (m) is and its offset (a).

$$y = m \cdot x + a \tag{14}$$

In the following set of images, Figures from 26 to 37, it can be shown how in the ΔhvsΔt curves, both the slope and the offset vary when the vertical speed is modified. Dashed lines corresponds to vertical speed of 2.5 meters per second for ascents/take-offs and 1.5 for descends. When the speed is increased, dash-dotted line for 5 meters per second for ascents/take-offs and 3 for descends, the time needed to do the vertical segment is lower. To know the real influence of the speed over time, a second set of images is plotted for each type of event, Images 27, 29, 31, 33, 35 and 37. These figures represents the evolution of the slope and the offset over the speed.

One last constraint must be added when computing timings. It is related to the specifications of the system. Each drone model has defined maximum speeds, which cannot be exceeded. The study has been done with a drone model ‘dji MAVIC PRO’ (see [4]). The datasheet specifies that the maximum ascent speed is five meters per second and maximum descent speed is three meters per second. So, knowing that, there is a speed that the system cannot exceed and, this is considered when computing the timings.

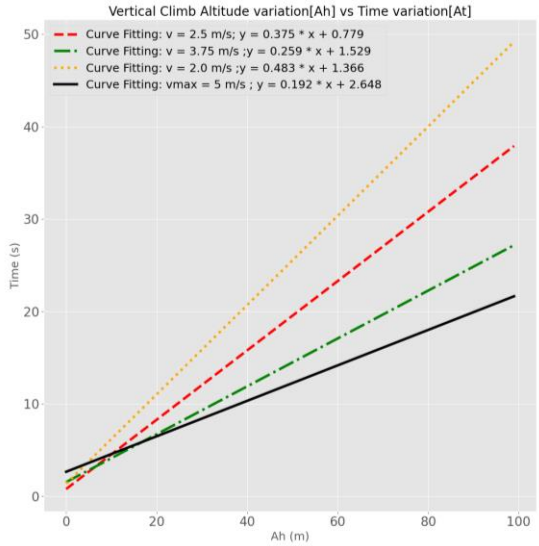


Fig 26 Vertical climb Δh vs Δt

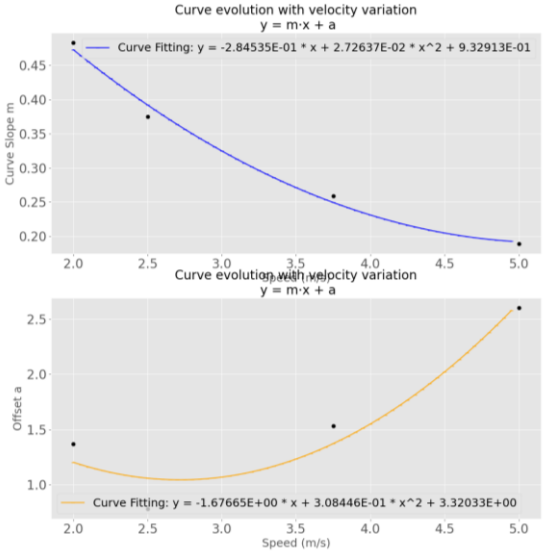


Fig 27 Vertical climb Speed influence

$$m = -2.84535 \cdot 10^{-1} \cdot v + 2.72637 \cdot 10^{-1} \cdot v^2 + 9.32913 \cdot 10^{-1}$$

$$a = -1.67665 \cdot v + 3.08446 \cdot 10^{-1} \cdot v^2 + 3.32033 \quad \textbf{(15)}$$

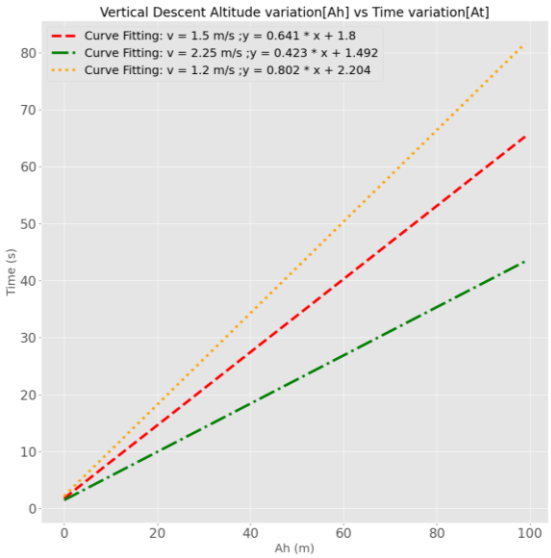


Fig 28 Vertical descent Δh vs Δt

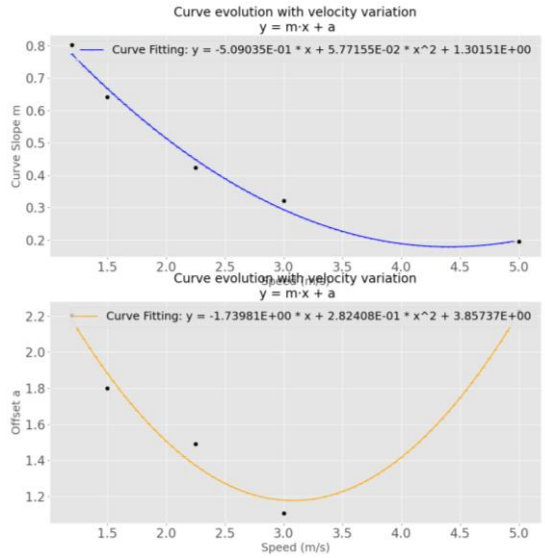


Fig 29 Vertical descent Speed influence

$$m = -5.09035 \cdot 10^{-1} \cdot v + 5.77155 \cdot 10^{-2} \cdot v^2 + 1.30151$$

$$a = -1.73981 \cdot v + 2.82408 \cdot 10^{-1} \cdot v^2 + 3.85737 \quad \textbf{(16)}$$

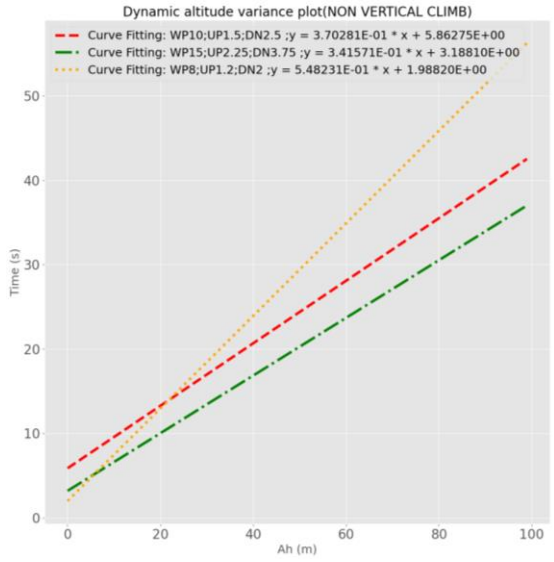


Fig 30 Non-Vertical climb Δh vs Δt

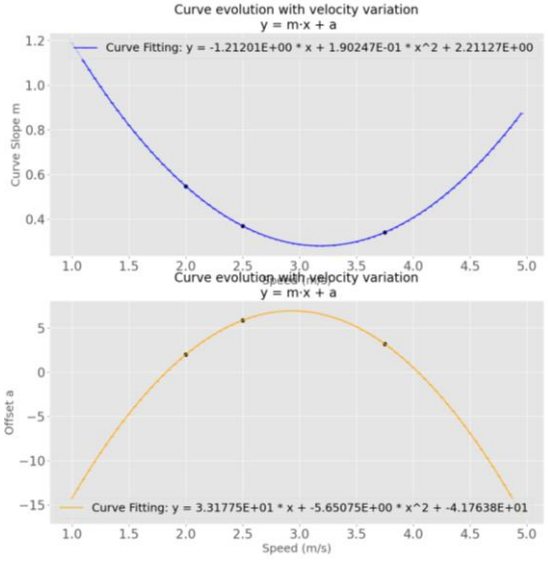


Fig 31 Non-Vertical climb Speed influence

$$m = -1.21201 \cdot v + 1.90247 \cdot 10^{-1} \cdot v^2 + 2.21127$$

$$a = 33.1775 \cdot v - 5.65075 \cdot v^2 - 41.7638 \quad \textbf{(17)}$$

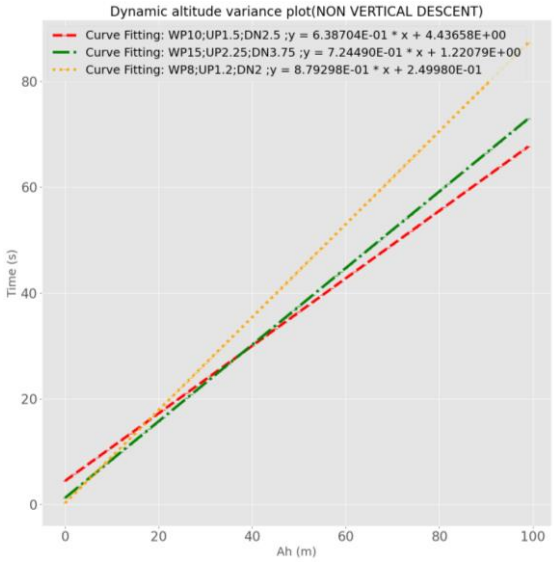


Fig 32 Non-Vertical descent Δh vs Δt

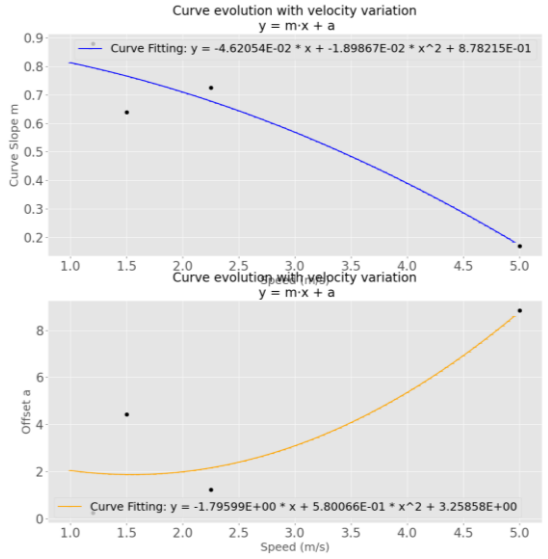


Fig 33 Non-Vertical descent Speed influence

$$m = -4.62054 \cdot 10^{-2} \cdot v - 1.89867 \cdot 10^{-2} \cdot v^2 + 8.78215 \cdot 10^{-1}$$

$$a = -1.79599 \cdot v + 5.80066 \cdot 10^{-1} \cdot v^2 + 3.25858 \quad \textbf{(18)}$$

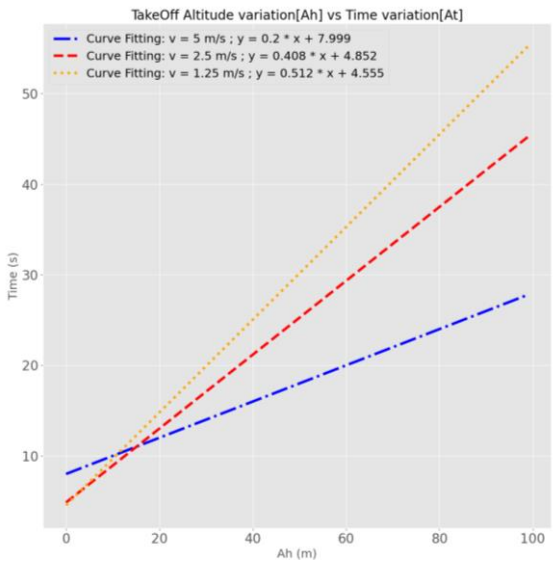


Fig 34 Take Off Δh vs Δt

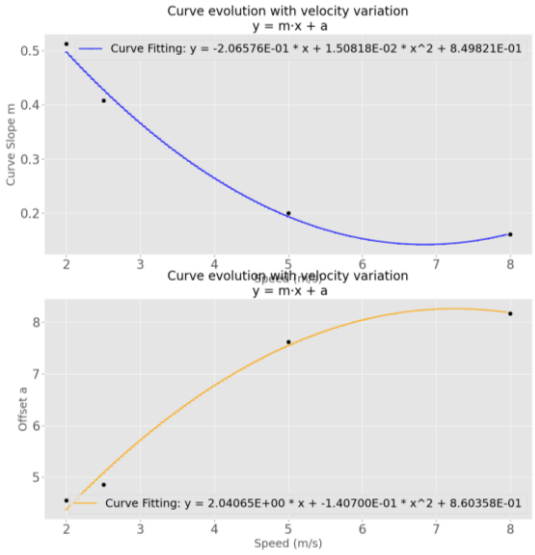


Fig 35 Take Off Speed Influence

$$m = -2.06576 \cdot 10^{-1} \cdot v + 1.50818 \cdot 10^{-2} \cdot v^2 + 8.49821 \cdot 10^{-1}$$

$$a = 2.04065 \cdot v - 1.40700 \cdot 10^{-1} \cdot v^2 + 8.60358 \cdot 10^{-1} \quad \textbf{(19)}$$

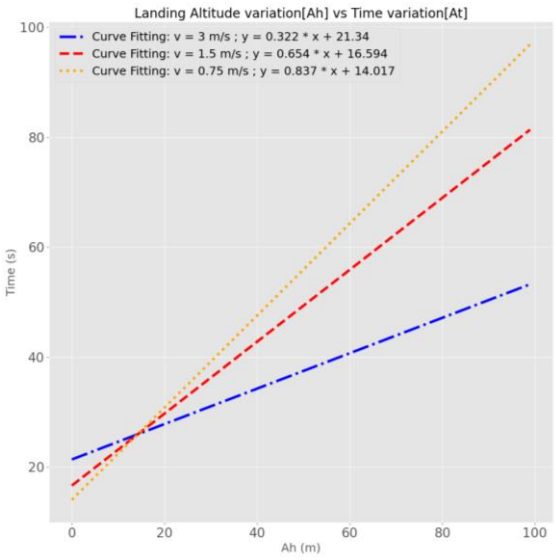


Fig 36 Landing Δh vs Δt

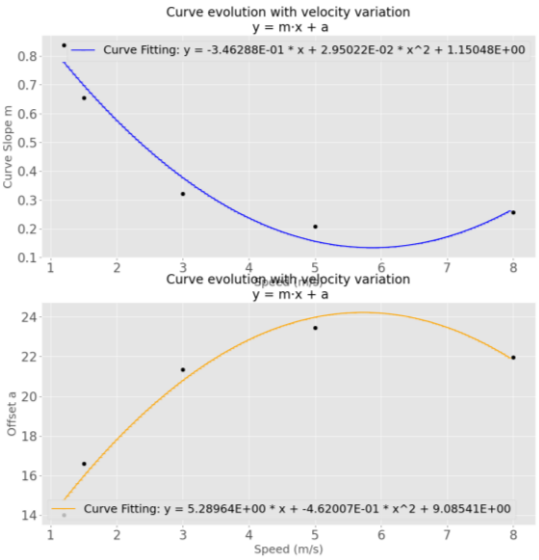


Fig 37 Landing Speed Influence

$$m = -3.46288 \cdot 10^{-1} \cdot v + 2.95022 \cdot 10^{-2} \cdot v^2 + 1.15048$$

$$a = 5.28964 \cdot v - 4.62007 \cdot 10^{-1} \cdot v^2 + 9.08541 \quad \textbf{(20)}$$

CHAPTER 5. TURN ANALYSIS

All the analysis due to a course variation is explained in this chapter. A turn is defined as a course variation along time. As it is explained previously, there are two different kind of turns, fly-by and fly-over. It is observed that, in all the real flights and the simulated ones, the drone performs a flyby waypoint. It is due to the fact that a fly-by turn is simpler than a fly-over. It has three advantages over flyover waypoints:

- The transition distance from one segment to the other is shorter.
- The transition from one segment to another is smoother and it only has to make one turn.
- Theoretically, the transition area is smaller.

Once the turning path is known, the influencing variables must be identified. There are different variables that define a turn and these are: the radius of turn and the angle to turn, which indicates how sharp the turn is. The first approach was to monitor the evolution of these parameters along the event. As a consequence, an in-depth study has been done and, no linear relationship has been found that relates the radius and the angle of turn with the time required to perform it. Since this approach do not explain the behaviour, another analysis plan had to be considered.

Previously, in Section 3.3.3, the refinement process carried out to improve the precision of the start and end point of the turns has been explained. In addition to this process, once the analysis had started, it was noticed that some turns were incorrectly detected. This is due to the fact that sometimes consecutive samples are received with very disparate course values. The software detected this as a turn, when in reality what is happening is that due to external effects, such as turbulence in the air, the vehicle suddenly changes its course until it is compensate by the autopilot. The post processing has the function of removing these sudden course changes which are not considered in the defined trajectory.

Since the interval between samples is one second and, in many cases, the turn time is less than one second, it is necessary to find another way to study the behaviour if the course changes. The attack plan that has been considered and implemented consists of designing a set of flight plans with the shape of polygons. The idea is to know the real time that the drone requires to carry out the flight plan and subtract the time it would take to fly the polygon segments, using the model extracted from the straight segment analysis and, taking into account that these segments are well-defined and the total length is known with accuracy. If the segments were concatenated in a straight way, there should be no difference between the flight value and the calculated one. So, the resulting value, divided by the number of polygon’s sides, corresponds to the time the vehicle spends to do a turn. If this process is repeated with different polygons, a curve can be obtained that relates the time of turn per degree rotated.

Moreover, to have a pool with a larger number of samples, some irregular polygons have been simulated. A widely explanation of the flight plan generation can be found in the Appendix A.

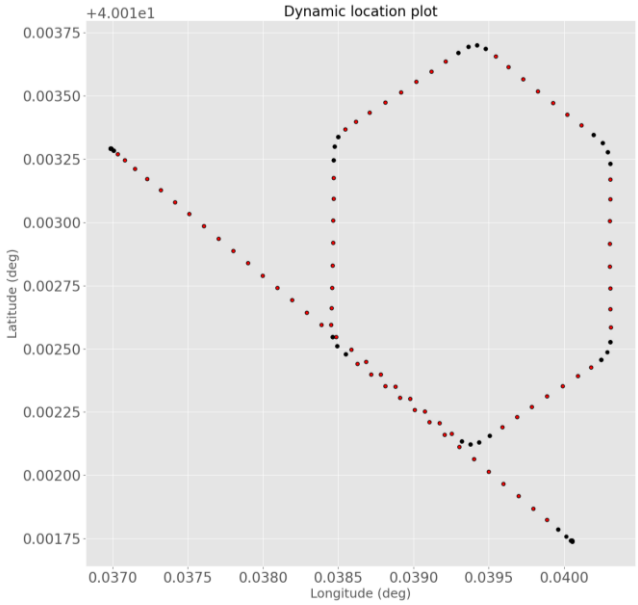


Fig 38 Hexagon flight plan

Image 38 corresponds to the flight plan of a hexagon. There is one side of the polygon that is flown over twice. That is the beginning and the end of the polygon flight. If the value of the timestamp from the two times it passes through that point is subtracted, the total polygon flight is obtained. It takes 66 second to do that flight. Then, the next step is to compute how much time the drone would need to perform the six sides as it were straight segments without turns. Using Equation 23, the required time would be 60.74 seconds. Knowing the total flight time, the number of turns (six) and, the time it would take to perform the straight segments as if there is no turns, the time spent at performing a single turn of 60 degrees is 0.876 seconds.

Figure 39 corresponds to the angle vs. time curve and, clearly, there is a linear relationship between these two variables which are described by Equation 21. The sample distribution fits a second-order curve that has as input parameter the angle of rotation ($\Delta\theta$), in degrees, and as output the time required to perform the rotation.

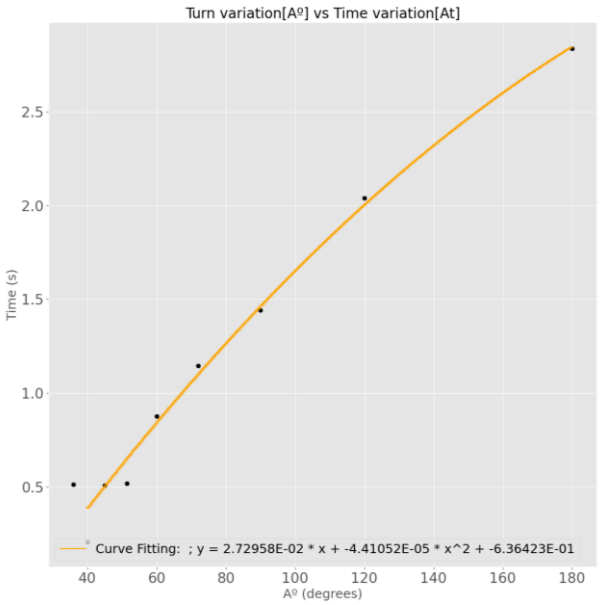


Fig 39 Turn variation Δº vs Δt

$$\Delta t = 2.72958 \cdot 10^{-2} \cdot \Delta\theta - 4.41052 \cdot 10^{-5} \cdot \Delta\theta^2 - 6.36423 \cdot 10^{-1} \quad \textbf{(21)}$$

As it has been made with the other events, a study has been carried out on how the influence of speed affects a turn. The variation of the curve fitting with respect with the one obtained previously is so small that it has been considered negligible. So, regardless of the speed of the system, the curve used to calculate the time spent in a turn is the corresponding to Equation 21.

CHAPTER 6. HOVER ANALYSIS

The main aim of this chapter is to explain how the hover analysis has been done. A hover in fact is a time delay between two navigation commands. During this time delay, the system is held in a stable and static way in the air.

Conceptually, hovers are the simplest event to study, due to the few variables involved in the manoeuvre. So, the way to identify them is to examine how much latitude, longitude and altitude vary over time. If they do not change or the variation is not enough the drone is performing a hover stage. There always has to be a minimum threshold of variation in which, below this, despite the drone is moving, it is considered as a hover. The reason of this defined threshold is to counter the suddenly perturbations caused by the environment, as it could be the wind. If the environment is pushing the drone steadily, as it could be a continuously wind, the drone by itself does the necessary to stay stable, since it is in autopilot mode.

The way a hover is defined in a MAVLink flight plan is by using the command '*MAV_CMD_NAV_DELAY'.* It allows the operator to define the time delay to wait between two navigation commands or define a specified UTC time at which the drone will have to wait to perform the following instruction.

Hovers are the only type of event that is not influenced by speed, because the system stays in the air with zero relative speed. So all things considered, once a start of hover is identified, a time increment must be added to the current starting time.

CHAPTER 7. STRAIGHT SEGMENT ANALYSIS

The main aim of this chapter is to explain the last event analysed. It consists on the study of the straight segments. The most relevant parameters in this research are the velocity (vx, vy, vz) and the altitude variance, if there is an altitude variation along the segment means that the event is indeed a non-vertical climb/descent. This information is sent by the drone to the ground station using the MAVLink protocol and more specifically, using the 'Global Position Int' message.

As it is explained in Section 3.3.4, the previous step is to correctly identify all other events because there is a high dependency between them. The reason for this consideration is because the speed at which the drone starts and ends the straight segment is not the same for a vertical climb/descent, end of turn or for any other event. For this reason, all possible combinations have been considered and an independent study has been done for each of them.

7.1 Equation fitting

The analysis has been done using the set of data that is formed by a group of real data obtained from a real flight and, data obtained from simulating flight plans. The key is to study the evolution of the horizontal speed along the segment and consequently, the acceleration/deceleration at the beginning and at the end of the event. To do that, the set of files with simulated flight plans consists of data from the ones used for the other events.

The first approach was to do the analysis without doing any kind of subdivision and no speed variation, setting the straight segment speed to ten meters per second. Each point of Figure 40 represent a straight segment, where the x-axis is the horizontal longitude variance (ΔL) and, the y-axis is the time it took to perform the defined trajectory (Δt). It can be seen that there is a linear relation between this two variables and it is represented by Equation 22. From it, it can be predicted how much time the drone would take to accomplish the segment.

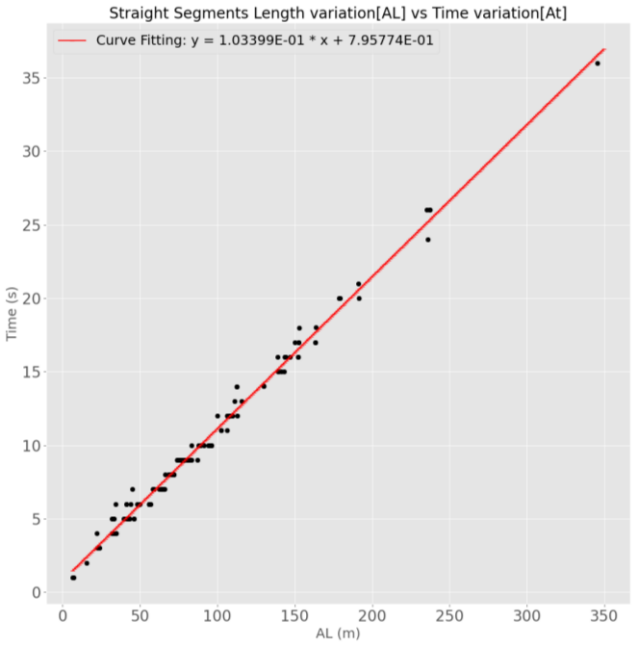
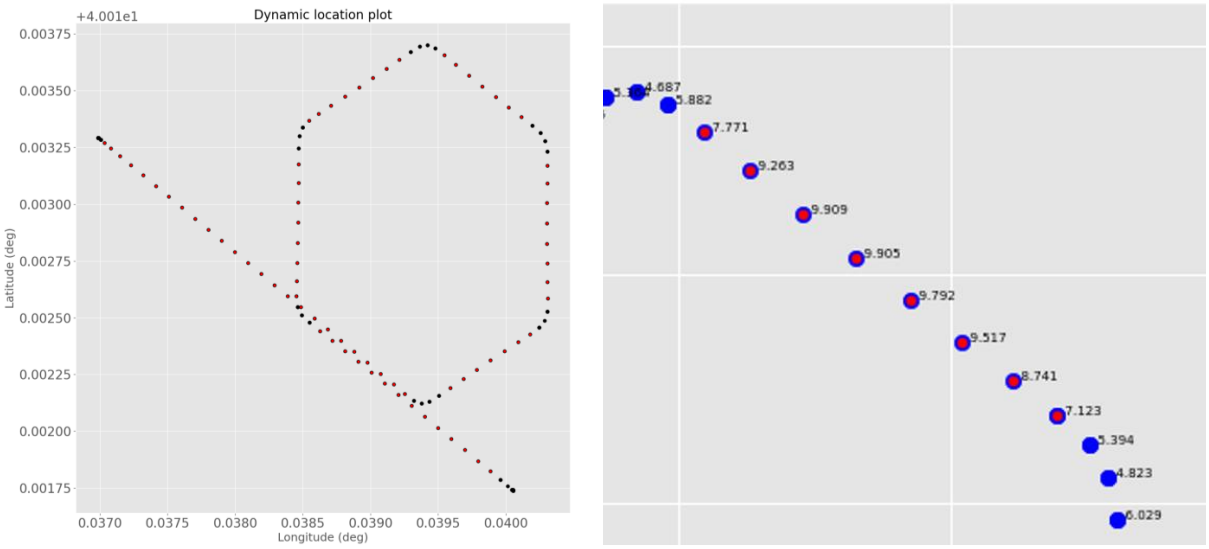


Fig 40 Straight Segment ΔL vs Δt

$$\Delta t = 1.03399 \cdot 10^{-1} \cdot \Delta l + 7.95774 \cdot 10^{-1} \quad \textbf{(22)}$$

In the region where the segments are shorter, it can be observed how the samples deviates a little from the curve. This is due to the fact that when the segment is too short, the influence of the speeds of the previous and posterior segment is greater that with long segments. In other words, every event has a defined speed. If the drone enters into a straight segment with a large speed, the drone has to decelerate in order to reach this speed and, this causes that the shortest segments have a more uneven behaviour. To reduce this effect, the idea is to identify the previous/posterior speed and the acceleration it performs to reach this defined velocity. To do that, a straight segment division has been done.

Analysing Equation 22, it can be seen that there is a linear relationship between the distance it travels and the time elapsed. The variable that relates these two parameters is the horizontal speed. The behaviour is as if the drone performed the segment at 9.671 meters per second (0.103399^{-1}), but considering that there is a time lag. This value is very close to the one at which the study has been made (10.0 m/s). The cause of the observed time offset is due to the fact that the movement is not performed at constant speed, there is an acceleration and braking stage, which adds extra time. Moreover, in Image 41(a), which is a longitude-latitude plot, it can be seen how the drone tries to reach the defined segment speed. Furthermore, it can be seen how it has to decelerate at the end to reach the new event speed.



(a) **(b)**

Fig 41 Hexagon segment analysis

Straight segments have been divided into five different type based on the events that precede and succeed them. These events are start/end of course change event, start/end of altitude change and hovers, besides the consideration of all possible combinations between them. These cases are explained in more detail in the following subsections. The division done to do the study is as follows:

- Case 1: End of course change + Straight Segment + Start of course change
- Case 2: End of altitude change + Straight Segment + Start of altitude change
- Case 3: End of altitude change + Straight Segment + Start of course change
- Case 4: End of course change + Straight Segment + Start of altitude change
- Case 5: Combinations with hover events.

7.1.1 Case 1: End of course change + Straight Segment + Start of course change.

The first type of straight segment corresponds to the case in which the segment is preceded by an end of course change and followed by a start of course change. To execute this analysis, the flight plans used are the same as for the study of turns, for the reason that the side of polygons are formed by segments preceded and succeeded by a course variation event.

Once the event detection has been done, the relationship between the length travelled by the drone and the elapsed time can be obtained. This linear dependence can be shown in Figure 42 where the x-axis is the horizontal longitude variance (ΔL) and, the y-axis is the time it took to perform the defined trajectory (Δt). The curve that better fits the samples plotted is a first order curve that linearly relate the two variables is described by Equation 23.

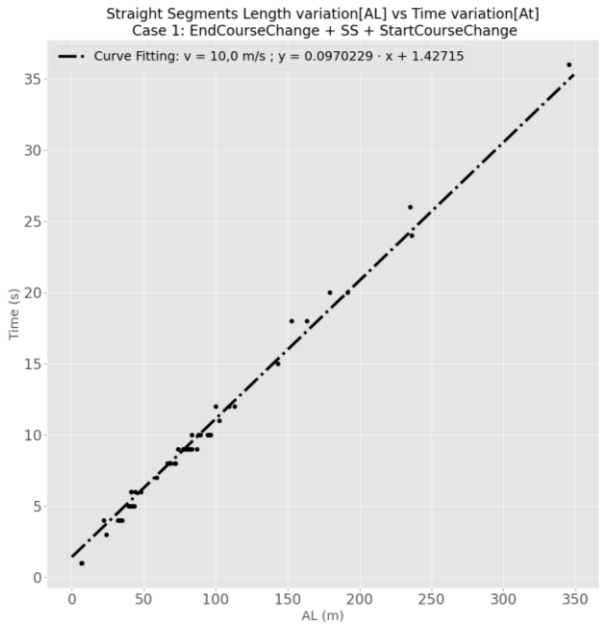


Fig 42 Straight Segment ΔL vs Δt, Case 1

$$\Delta t = 9.70229 \cdot 10^{-2} \cdot \Delta L + 1.42715 \quad \textbf{(23)}$$

The inverse of the slope of Equation 23 is the equivalent speed at which the segment is performed, which is equal to 11.08 meters per second. This equivalent speed is higher that the defined segment speed (10 m/s) but, there is a time offset to considerate. The cause of the observed time offset is due to the fact that the movement is not performed at constant speed, there is an acceleration and braking stage, which adds extra time.

7.1.2 Case 2: End of altitude change + Straight Segment + Start of altitude change.

Case number two corresponds to the ensuing sequence of events: End of altitude change, straight segment and start of altitude variation. It has to be noted that there are two types of altitude changes, vertical and not fully vertical. It is important to clarify this difference because for the vertical climb/descends, both vertical and horizontal initial/final speeds, are equal to zero. On the other hand, for the non-vertical climb/descends, the initial/final horizontal speed could be higher than zero. To effectuate this analysis, a set of flight plans, that contain this type of straight segments, have been designed.

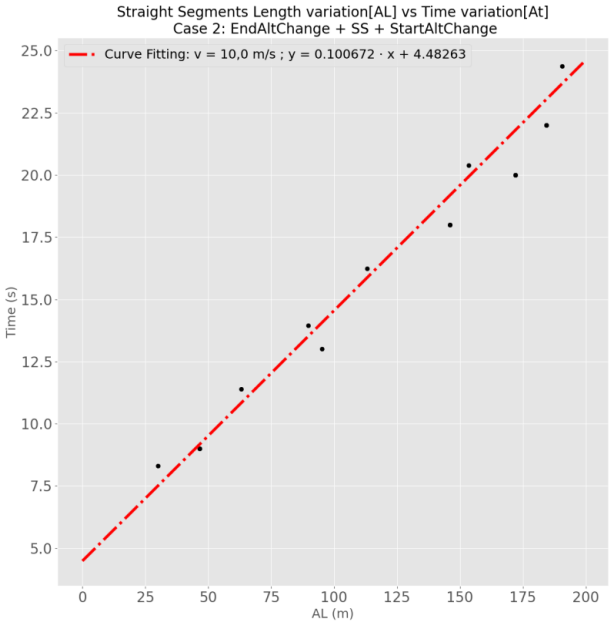


Fig 43 Straight Segment ΔL vs Δt, Case 2

$$\Delta t = 1.00672 \cdot 10^{-1} \cdot \Delta L + 4.48263 \quad \textbf{(24)}$$

Starting from Equation 24, it is possible to obtain the equivalent speed of the system when performing a straight segment of case 2. This equivalent speed is the inverse of the slope of the curve of Image 43 and it is equal to 9.933 m/s, value very close to the real segment speed (10m/s).

7.1.3 Case 3 and Case 4

Both cases 3 and 4, which are in fact combinations of cases 1 and 2, are explained in this subsection. The straight segment of case 3 is preceded by a change in altitude and followed by a turn, while, case 4 is the other way around, a turn followed by a straight segment and ended by a climb/descent.

In order to have data to fulfil the analysis, a flight plan had to be designed in which, several straight segments belonging to this cases are incorporated. Next, two graphs are shown that describe how the drone behaves in the face of this type of event. It can clearly be seen that there is a linear relationship between the two studied variables and it can be expressed following Equations 25 and 26.

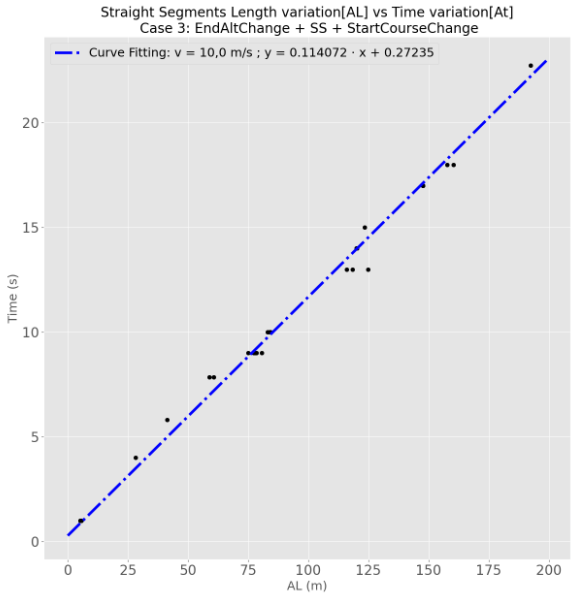


Fig 44 Straight Segment ΔL vs Δt, Case 3

$$\Delta t = 1.14072 \cdot 10^{-1} \cdot \Delta L + 2.72350 \cdot 10^{-1} \quad \textbf{(25)}$$

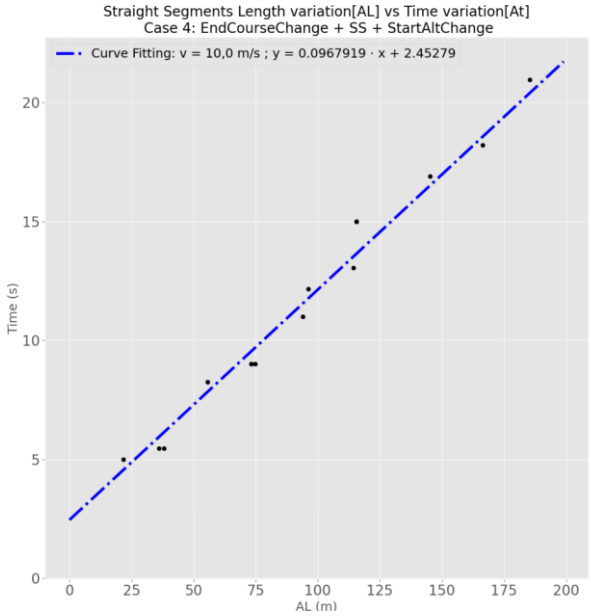


Fig 45 Straight Segment ΔL vs Δt, Case 4

$$\Delta t = 9.6791 \cdot 10^{-2} \cdot \Delta L + 2.45279 \quad \textbf{(26)}$$

7.1.4 Case 5: Combination with hover events.

The last case is formed by the combination of the previous events, altitude change, course variation, and a hover. From the speed point of view, it must be equal to zero at the beginning and end of the hover and therefore, the behaviour should be the same as a vertical climb/descent. Initially, this case had been analysed separately from the others to confirm this statement, and seeing that it is true, it has been decided that when a hover is detected, the corresponding equation, obtained from the previous cases, will be applied.

Table 6 Straight segment equivalent speeds

	Case 1	Case 2	Case 3	Case 4
Equivalent Speed (m/s)	10.306	9.933	8.766	10.331
Time Offset (s)	1.427	4.482	0.272	2.453

All in all, an equation has been obtained for each type of straight segment. This equations shows a linear relationship between the horizontal distance travelled and the time elapsed. The variable that relates these two parameters is speed. The inverse of the slope of these curves defined an equivalent speed. It is as if the drone performed the segments at this speed but adding an extra time (offset). The value of the equivalent speed is in all cases around 10 m/s, it has

sense because this is the speed at which the flight have been flown, Table 6. The offset is different between all cases due to the fact that speed is not constant. As the initial and final speeds are not the same in five cases, the time it takes to accelerate/decelerate to reach the defined speed would be also different. That is the reason why the offsets are not equal or even close for all five cases.

7.2 Speed variation

The aim of this subsection is to explain how the segment defined speed affects and how the shape of the curves changes.

First of all, a brief explanation should be made on how a segment is performed. As it is explained is previous sections, there are different events that can precede a straight segment. The reason for this consideration is because the speed at which the drone starts the straight segment is not the same for a vertical climb/descent, end of turn or for any other event. The time the drone spends to reach the defined segment speed depends on the starting speed. So, there are three parameters to take into account at predicting the behaviour: the type of event that precedes and succeeds the segment, the acceleration at which the drone passes from the starting speed to the defined one and to the ending speed and the longitude of the segment.

The same flight plans used to extract the previous model, have been simulated with several speeds. As the equations obtained are of the first order, equation 27, it is necessary to see how the progression of the slope of the line (m) is and its offset (a). The following graphs, 51 - 58, show how the equations, obtained in the previous subsections, evolves when horizontal speed is modified. The slope of theses curves decreases exponentially as speed increases. This makes absolutely sense since the higher the speed, the shorted the time it needs to travel the segment. The more the speed increases, the smaller the value of the slope of the equation. In the datasheet of the studied drone, ‘dji MAV PRO’ [4], a maximum value of horizontal speed is defined, 65 kilometres per hour (18.05 meters per second). It must be taken into account when defining the flight plan and setting the drone.

$$y = m \cdot x + a \tag{27}$$

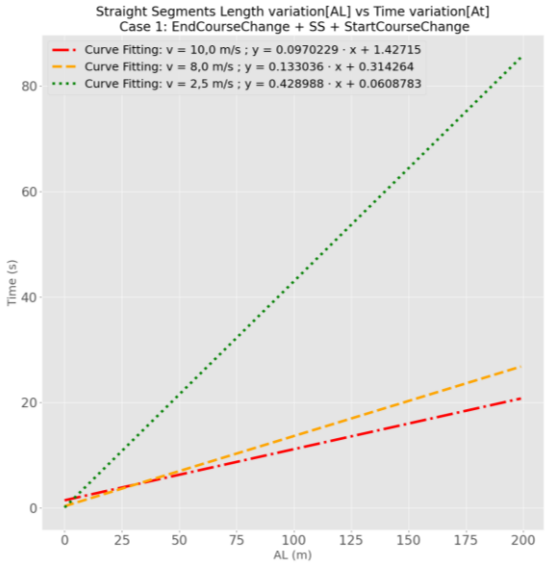


Fig 46 Straight Segment Case 1 multiple speeds

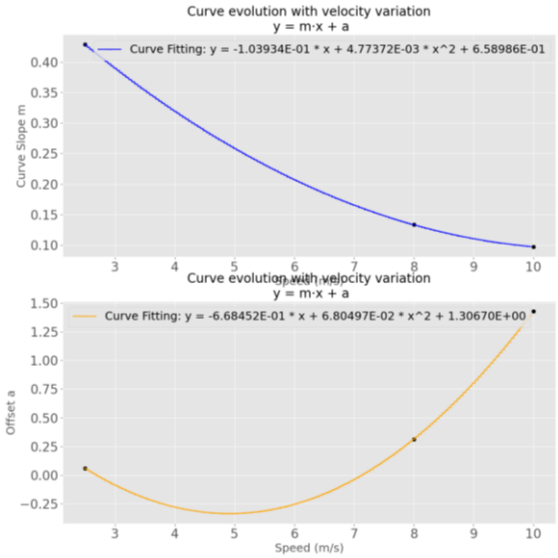


Fig 47 Straight Segment Case 1 speed influence

$$m = -1.03934 \cdot 10^{-1} \cdot v + 4.77372 \cdot 10^{-3} \cdot v^2 + 6.58986 \cdot 10^{-1}$$

$$a = -6.68452 \cdot 10^{-1} \cdot v + 6.80497 \cdot 10^{-2} \cdot v^2 + 1.30670 \quad \textbf{(28)}$$

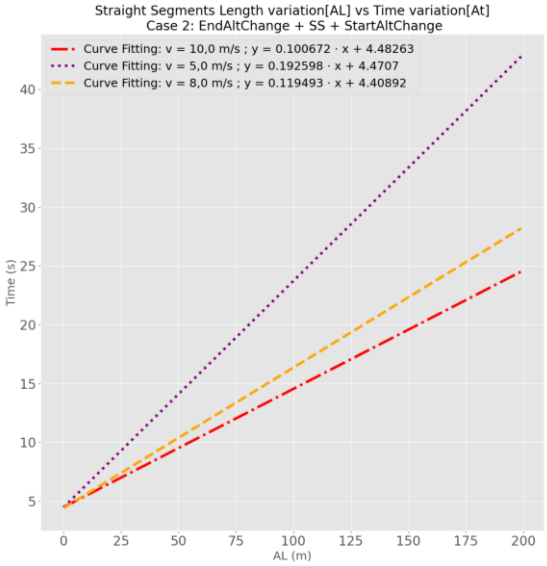


Fig 48 Straight Segment Case 2 multiple speeds

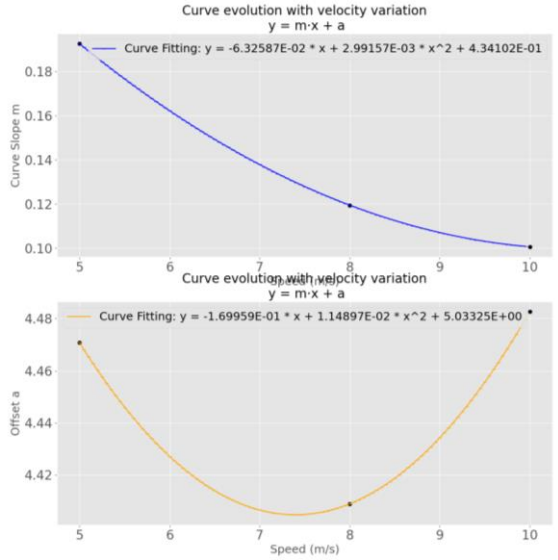


Fig 49 Straight Segment Case 2 speed influence

$$m = -6.32587 \cdot 10^{-2} \cdot v + 2.99157 \cdot 10^{-3} \cdot v^2 + 4.34102 \cdot 10^{-1}$$

$$a = -1.69959 \cdot 10^{-1} \cdot v + 1.14897 \cdot 10^{-2} \cdot v^2 + 5.03325 \quad \textbf{(29)}$$

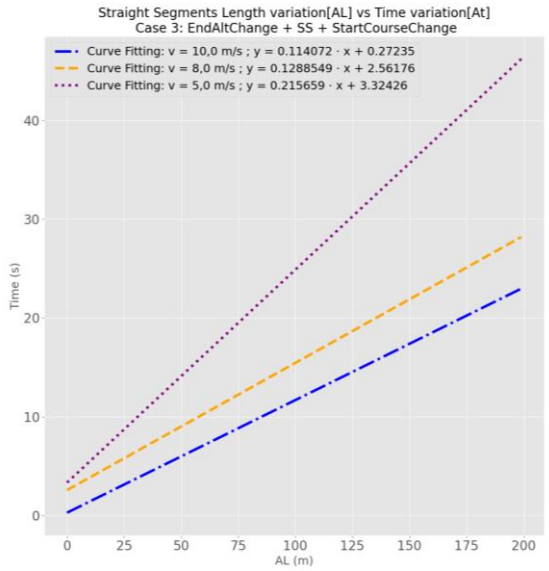


Fig 50 Straight Segment Case 3 multiple speeds

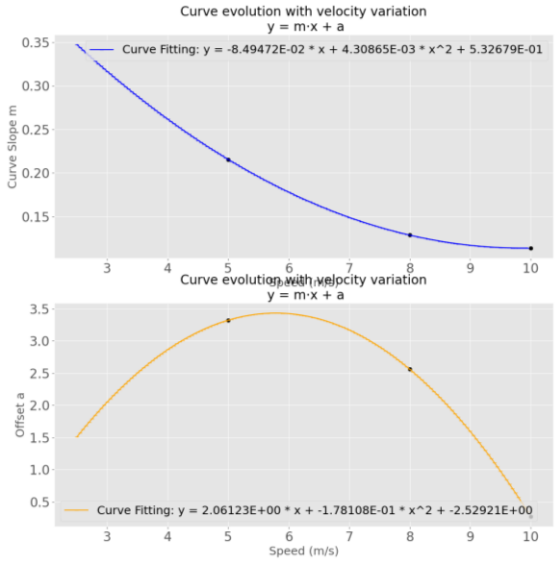


Fig 51 Straight Segment Case 3 speed influence

$$m = -8.49472 \cdot 10^{-2} \cdot v + 4.30865 \cdot 10^{-3} \cdot v^2 + 5.32679 \cdot 10^{-1}$$

$$a = 2.06123 \cdot v - 1.78108 \cdot 10^{-1} \cdot v^2 - 2.52921 \quad \textbf{(30)}$$

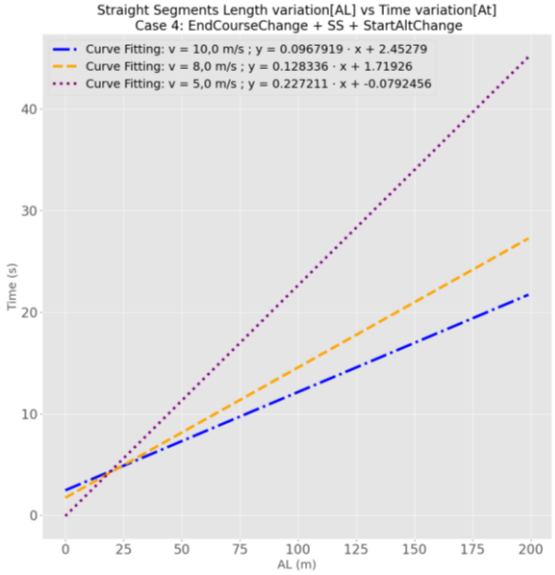


Fig 52 Straight Segment Case 4 multiple speeds

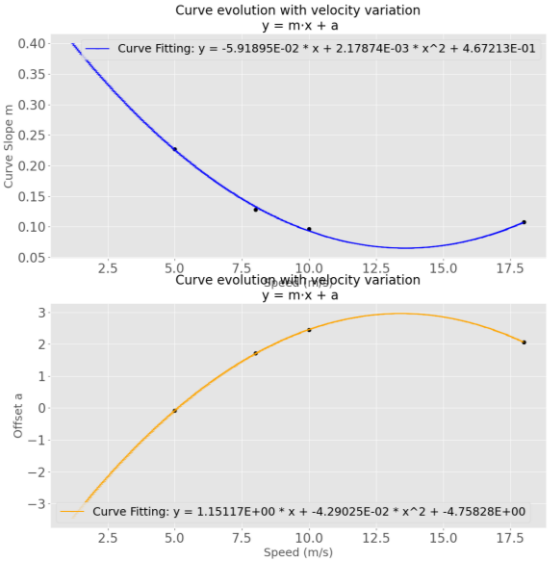


Fig 53 Straight Segment Case 4 speed influence

$$m = -5.91895 \cdot 10^{-2} \cdot v + 2.17874 \cdot 10^{-3} \cdot v^2 + 4.67213 \cdot 10^{-1}$$

$$a = 1.15117 \cdot v - 4.29025 \cdot 10^{-2} \cdot v^2 - 4.75828 \quad \textbf{(31)}$$

CHAPTER 8. FLIGHT PLAN SIMULATION

The main aim of this chapter is to verify that the algorithm extracted in the previous sections is correct and accurate enough. In order to achieve this, a flight plan has been designed to compare the data obtained when flying it in real life with the one obtained from the ArduPilot simulator 'Mission Planner' and with the one obtained from the prediction model extracted during the project.

The flight time employed by the drone to complete each trajectory is computed from the telemetry data, using the software developed, and compared with the flight-time forecast by the algorithm extracted from the data analysis process. The final flight time is computed summing the partial results of each event.

8.1 Algorithm developed vs. Mission Planner

First of all, a comparison has been made between a simulated flight plan using the 'Mission Planner' tool, and the data obtained from applying the model extracted throughout the thesis. This flight plan contains all kinds of events. Image 54 shows the flight plan from an overhead point of view besides the flight plan commands.

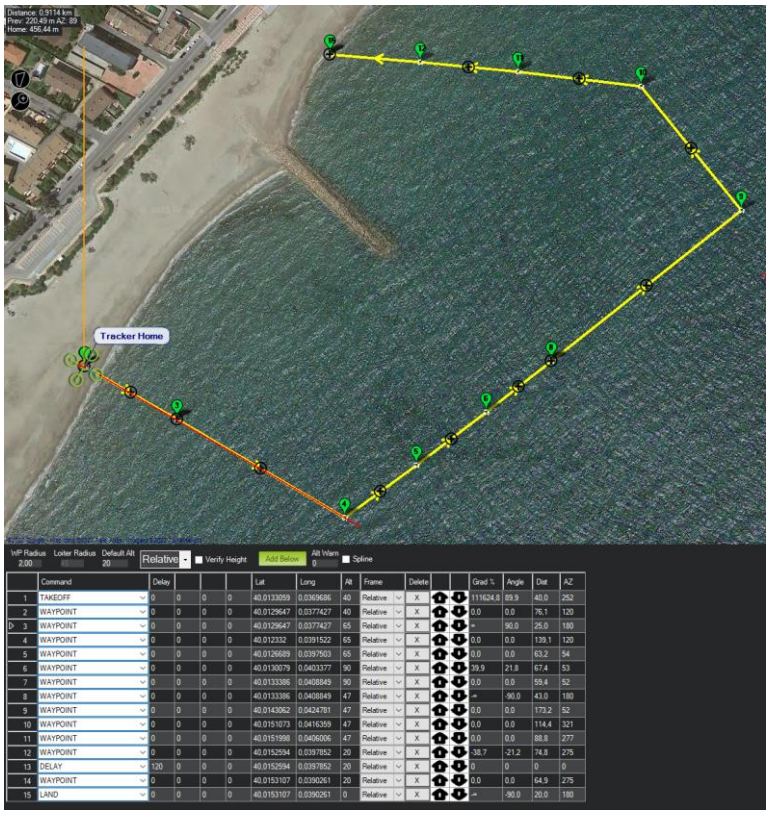


Fig 54 Flight Plan view using Mission Planner

The flight plan has been designed with the focus on containing events of all kinds. In this way, it is confirmed that the developed algorithm accurately predicts any type pf circumstance. The flight begins with a vertical take-off of 40 meters, followed by a straight horizontal segment (case 2) of length: 76.1 meters. Once it reaches waypoint 2 (latitude 40.0129647, longitude 0.077427 and altitude 40 m), it begins a vertical ascent of 25 meters. Next, there is a straight horizontal segment of 139.1 meters (Case 3), which ends in waypoint4, where it makes a 65 degree turn to start another straight segment of 63.2 meters (Case 4). At waypoint 5 a non-vertical ascent that ends at waypoint6, located at 62.6 meters on the horizontal axis and 25 on the vertical axis. This event is followed by a straight segment (Case 2) of 59.4 meters. At waypoint7, a vertical descent of 43 meters begins followed by a straight horizontal segment (Case 3) of 173 meters long that connects with another straight segment (Case 1) through a 90 degree turn. At waypoint 9, the previous segment ends and after a 45 degree turn there is another segment (Case 4) of 88.8 meters long. At waypoint 11 begins a non-vertical descent of 22 meters along 69.8 meters of horizontal length that ends with a 120 seconds hover. Finally, after the hover, there is another segment (Case 5), which leads to a 20 meter landing. It can be seen that this flight plan contains all possible event types.

In order to measure the precision between the computed and real times, the error between the data obtained from the simulator and those obtained from the model has been calculated. Tables 7 and 8 contrast this information. The software that performs the simulation analysis detects the events by grouping samples sent by the system. That means that the time duration of these events are defined in intervals of 1 seconds so, the event detection tolerance is ±1 second. This can be clearly seen on turns, whose real time is less that a second but the analysis tool detect them as if they took at least one second. Flying at high speed makes the distance between samples higher, so the detection of events is done with less accuracy. In order to minimize this error, the rate of telemetry message generation should be increased.

The study of the flight plan has been made at different speeds. For the first simulation, the horizontal speed defined is 10 meters per second, the ascent one is 2.5 meters per second and the descent speed is 1.5 meters per second. In Table 7, it can be seen that there is a dissimilarity between the time obtained from the simulator software (column two) and the one computed by using the model (column three). The larger the events in time, the smaller the error between both models. This is because the detection of the exact starting and ending point of an event can differ and introduces error. The fact that the start of an event is defined in a sample or in its consecutive one defines a different error value.

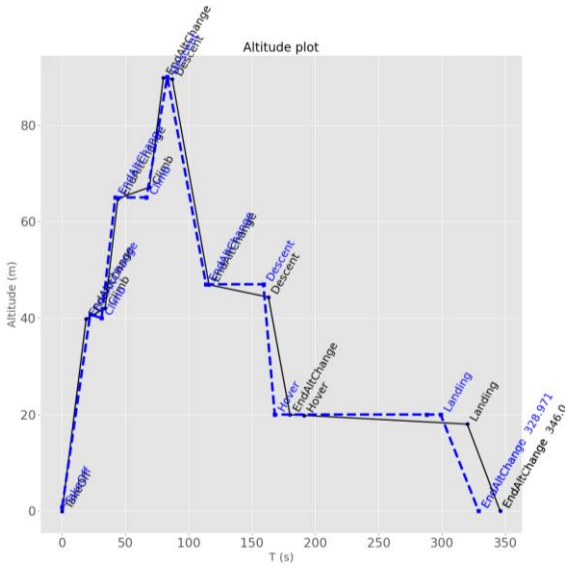


Fig 55 Altitude Plot v1

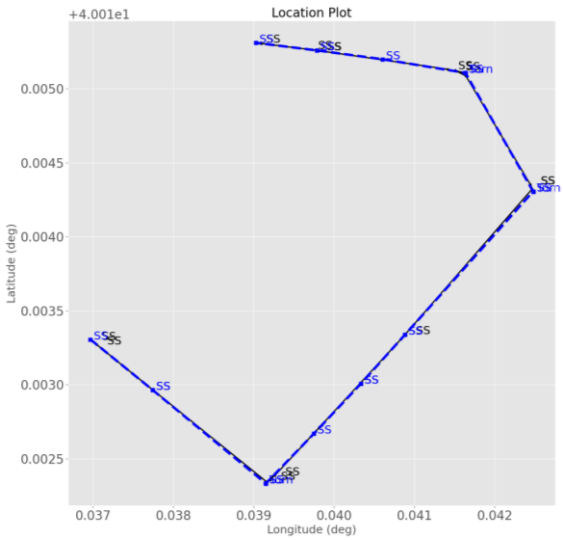


Fig 56 Location Plot v1

Both accuracy and error have been calculated using the data contained in Figures 55, 56 and Table 7. As with the model, what is computed is the starting and ending point of each event, the comparison is made between the variables of these events, such as longitude, latitude, altitude and timestamp. Figures 55 and 56 correspond to the altitude plot (Time vs Δh) and the location plot (Longitude vs Latitude) of the flight. The continuous solid line represents the event succession corresponding to the simulator tool, while the dashed line is for the one obtained with the model developed along the thesis. In the location plot, it can be seen that the starting point of straight segments and turns are the same for both tools. Altitude plot shows the starting and ending points of altitude changes events and the time offset between them.

There are two events that introduce a large part of the total error, these are the first straight segment and the segment before the hover. The time error in the first straight segment is 6.041 seconds. In image 55, it can be seen how the starting point of this segment is not exactly the same, this mismatch may be due to the fact that the start/end of the event is identified in a different way in both models. Since the rate of generating data is one sample per second, the fact of choosing a sample or its consecutive as the start of an event cause that the absolute error differs one second. Note that there is a three second difference in take-off times, part of this errors could have the same root cause (the starting point of the event is detected with a few samples of difference). The other noteworthy error occurs in the non-vertical descent previous to the hover. It can be observed in altitude plot that the starting altitude of this event is not exactly the same. At this point the error is introduced and it is propagated to the end of the trajectory.

The absolute error of the entire flight is 13.029 seconds, while the relative error is 4.90 % and the mean absolute error of the flight stages is 1.820 seconds.

Table 7 Mission Planner vs Model

Event	Time simulator	Time model	Error
Take off Δh 40 m	19.0 s	22.188 s	3.188 s
Straight Segment Case 2 Δl 76.1 m	15.0 s	8.959 s	6.041 s
Vertical Climb Δh 25 m	10.0 s	10.856 s	0.856 s
Straight Segment Case 3 Δl 139.1 m	16.0 s	16.168 s	0.168 s
Turn Δθ 65º	1.0 s	0.973 s	0.027 s
Straight Segment Case 4 63.2 m	8.0 s	7.552 s	1.552 s
Non-vertical Climb Δh 62.6 m Δl 25 m	11.0 s	8.522 s	2.478 s
Straight Segment Case 2 Δl 59.4 m	5.0 s	7.873 s	2.873 s
Vertical Descent Δh 43 m	31.001 s	30.599 s	0.402 s
Straight Segment Case 3 Δl 173 m	20.999 s	20.045 s	0.954 s
Turn Δθ 90º	1.0 s	1.482 s	0.482 s
Straight Segment Case 1 Δl 114.4 m	12.0 s	12.524 s	0.524 s
Turn Δθ 45º	2.0 s	0.479 s	1.521 s
Straight Segment Case 4 Δl 88.8 m	11.0 s	10.756 s	0.244 s
Non-vertical Descent Δl 69.8 m Δh 27m	17.0 s	8.98 s	8.02 s
Hover Δt 120s	120.0 s	120.0 s	0
Straight Segment Case 5 Δl 64.9 m	12.0 s	11.085 s	0.915 s
Landing Δh 20m	34.0 s	29.929 s	4.071 s
Total Time	346 s	328.971s	17.029 s

Next, the same flight plan has been simulated and computed again but changing the speeds. The horizontal speed defined is 8 meters per second, both the ascent and descent are 2 meters per second.

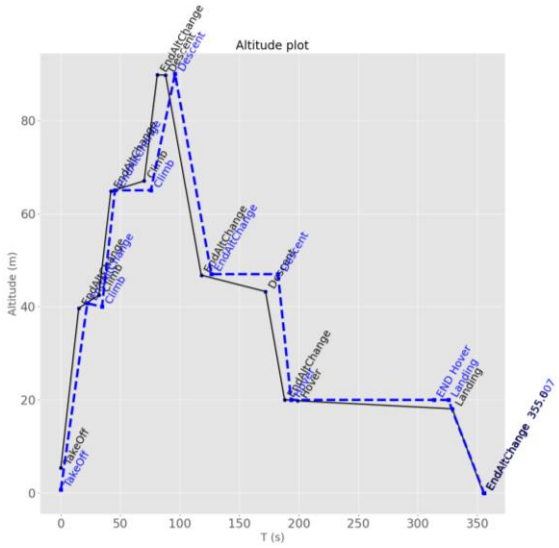


Fig 57 Altitude Plot v2

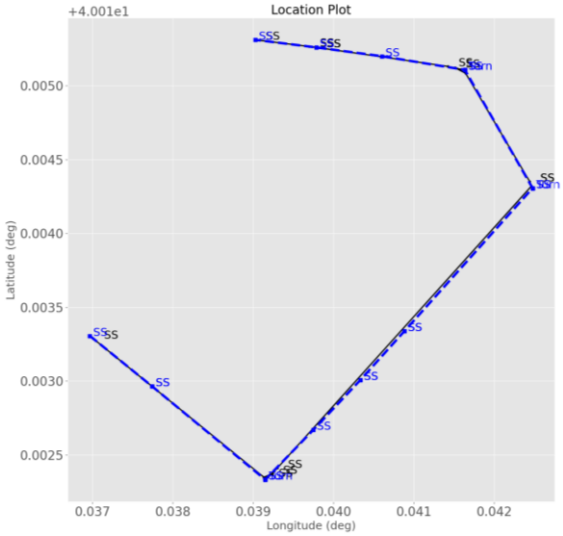


Fig 58 Location Plot v2

Both accuracy and error have been calculated using the data contained in Figures 57, 58 and Table 8. The highest absolute error occurs in the last non-vertical descent as it occurs with the previous speed configuration. The straight segment previous to landing also has a high error value. As it can be contrasted in the table, the software detected the hover as it took more seconds. The other flight stages have an absolute error of less than three seconds, which can be caused by the fact that both models does not identify the start/end of the event in the same point. The absolute error of the entire flight is 3.393 seconds, while the relative error is 0.945 %.

Table 8 Mission Planner vs Model. Second speed configuration

Event	Time simulator	Time model	Error
Take off Δh 40 m	19.0 s	22.188 s	3.188 s
Straight Segment Case 2 Δl 76.1 m	12.0 s	12.375 s	0.375 s
Vertical Climb Δh 25 m	10.0 s	10.856	0.856 s
Straight Segment Case 3 Δl 139.1 m	18.002 s	20.517 s	2.515 s
Turn Δθ 65º	0.998 s	0.973 s	0.025 s
Straight Segment Case 4 63.2 m	9.000 s	8.735 s	0.265 s
Non-vertical Climb Δh 62.6 m Δl 25 m	11.002 s	10.054 s	0.946 s
Straight Segment Case 2 Δl 59.4 m	6.998 s	10.218 s	3.22 s
Vertical Descent Δh 43 m	30.002 s	30.599 s	0.597 s
Straight Segment Case 3 Δl 173 m	23.998 s	24.897 s	0.899 s
Turn Δθ 90º	1.002 s	1.482 s	0.480 s
Straight Segment Case 1 Δl 114.4 m	15.998 s	15.530 s	0.468 s
Turn Δθ 45º	2.0 s	0.479	1.521 s
Straight Segment Case 4 Δl 88.8 m	13.998 s	13.552 s	0.446 s
Non-vertical Descent Δl 69.8 m Δh 27m	15.0 s	11.015 s	3.985 s
Hover Δt 120s	125.0 s	120.0 s	5.0 s
Straight Segment Case 5 Δl 64.9 m	15.998 s	12.207 s	3.791 s
Landing Δh 20m	29.0 s	29.929 s	0.929 s
Total Time	359.0 s	355.607 s	3.393 s

All in all, the accuracy of the data obtained by the model is high enough. In the next section, the same process will be done with real data, where external factors have higher impact and can introduce some noise.

8.2 Algorithm developed vs. Real flight

In this section, the data obtained from the model has been compared with the data obtained from real flights. In order to achieve this, three flights performed at different speeds and presenting all possible events have been chosen.

The first one, is made up of 90x20 m rectangles in which vertical ascents/descents of 10 and 20 meters take place. Therefore, the flight follows the sequence: horizontal straight segment of 90 meters, turn of 90 degrees, horizontal straight segment of 20 meters and a vertical climb/descent.

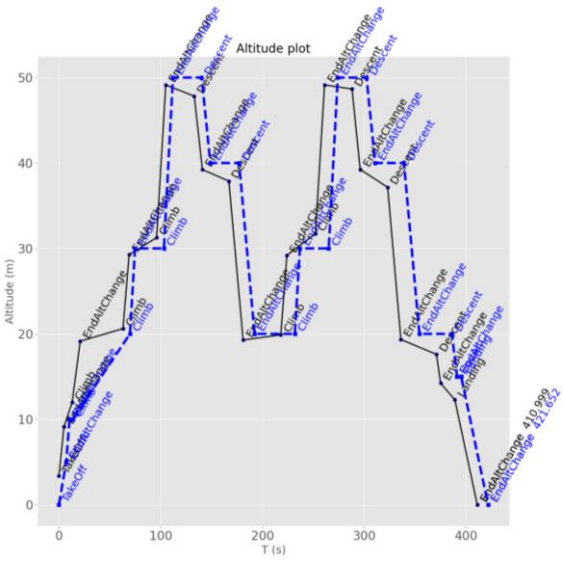


Fig 59 Altitude Plot Real Flight 1

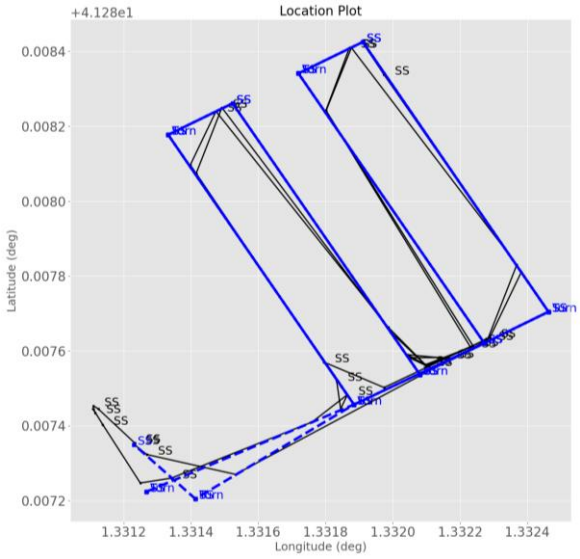


Fig 60 Location Plot Real Flight 1

As in the previous section, a comparison has been made between the data obtained from the drone telemetry and the one corresponding to the model. This flight plan contains commands which are used to change speed: ‘MAV_CMD_DO_CHANGE_SPEED (178)’. Although it contains this type of command, the initial speeds must be defined. The horizontal speed defined is 4.5 meters per second, the ascent one is 2.5 meters per second and the descent speed is 1.5 meters per second, data obtained from observing the messages sent by the system. The continuous solid line represents the event succession corresponding to the real data, while the dashed line is for the one obtained with the model developed along the thesis. Comparing both altitude

and location plots, it can be observed that the most noticeable error is found at the beginning of the flight plan. The flight plan starting point is not really the same. The starting point of the tool exactly matches with the defined in the flight plan (longitude 1.33110893, latitude 41.28744125, altitude 0 m) but, the flight started at longitude 1.3311228, latitude 41.287483. Between this two points there is a distance of 4.8 meters, this bad positioning can be caused a by a human error. This shift error is maintained during the whole flight and each straight segment increases it. The real flight took 410.999 seconds while the tool estimated it at 421.652. The overall absolute time error is 10.653 seconds which corresponds to a relative error of 2.591 %.

The second flight analysed in this section is made up of two 190x90 m rectangles. There are no altitudes variations along the rectangles. The horizontal speed defined is 8 meters per second, both the ascent and descent speeds are 1.4 meters per second.

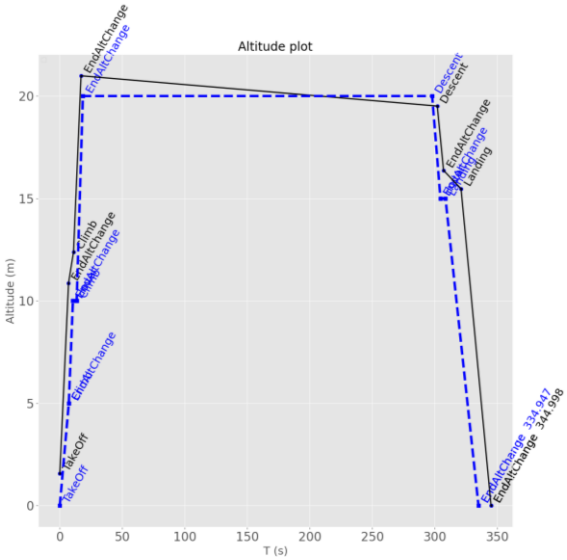


Fig 61 Altitude Plot Real Flight 2

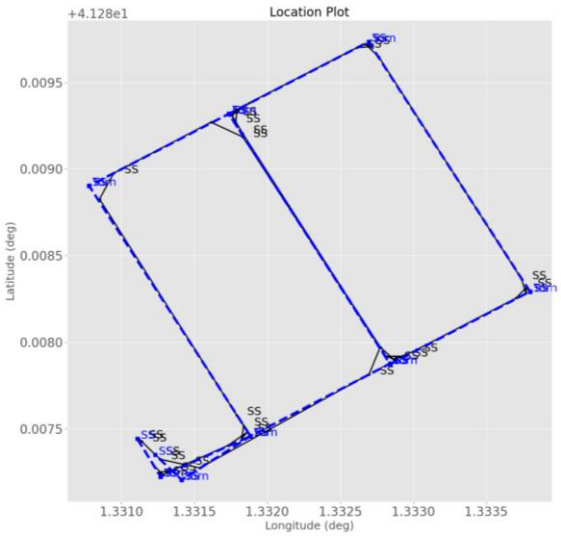


Fig 62 Location Plot Real Flight 2

This is a simpler flight, where only there are two altitude variations which corresponds to the take-off and landing. Observing the altitude plot, Figure 61, it can be noted that there is an altitude variation, of 4.61 meters, between the final point of the take-off and the starting point of the landing. The reason for this slow variation could be related to some kind of error when generating the flight telemetry. The interesting thing about this flight is that there are speed changes in each straight segment. The 90 meter segments are performed at 4 meters per second while the 190 meters segments at 7 meters per second. The drone takes 30.0 seconds to travel the 190-meter segments and 21.0 in the 90 meter segments, while the algorithm estimates it to be 30.7 and 23.769, respectively. The overall absolute time error is 10.051 seconds which corresponds to a relative error of 2.913 %.

Taking all results into account, it is noticeable that the accuracy of the tool, around 1 and 4 of relative error, is good enough to predict the performance of the drone given a defined trajectory. The events that introduce a larger error are the not fully vertical altitude changes due the complexity of its behaviour. The main objective of the project was to develop a tool capable of accurately compute the trajectory of a drone while being efficient in terms of resource management. The results of this section confirm that the desired accuracy of the tool has been successfully achieved.

CHAPTER 9. CONCLUSIONS AND FUTURE WORK

The presented thesis exploits data analysis techniques to predict the flight time employed by a drone to complete a defined trajectory. The idea was to divide a complex trajectory into a sequence of standard events. From a database composed of simulated and real flight data, a mathematical algorithm has been developed that models the behaviour of a drone given a trajectory. At the end, three possible flight plans were analysed, computing all the flight stages timings. The error in the prediction resulted on the order of 1 - 4 % of the real total flight. Thus, the presented algorithm can support an accurate estimation of the flight plan time.

The proposed algorithm was developed considering a selected drone model but it can be extrapolated to any kind of drone.

There are several options to expand the project. The first one could be the increase of input parameters. As it is explained during the project, there are many variables that have not been taken into account. For instance, all the related with the surrounding of the system, as it could be the density, the temperature of the environment or the wind acting over the drone. In order to increase the accuracy of the obtained models, some sensors should be placed all over the drone to obtain more data and develop a more accurate tool.

The most transcendent improvement would be the use of a machine learning program. This is a kind of predictive model, which use a statistical technique, using the aid of historical and existing data, to predict and forecast the futures outcomes. During the development of this thesis, there are several parameters, such as the climb detection threshold, which has been manually set based on a tuning process. The accuracy of this defined variables must be increased by the use of machine learning. Moreover, predictive models are validated and revised in every iteration. This is a very interesting fact because if the drone performance changes due to some external factor, the calculated data must be also changed or the forecast data would not be close to the real one. In other words, machine learning models make assumptions based on what has happened in the past and what is happening in the present.

BIBLIOGRAPHY

[1] Nuic, Angela & Poles, Damir & Mouillet, Vincent. (2010). BADA: An advanced aircraft performance model for present and future ATM systems. International Journal of Adaptive Control and Signal Processing. 24. 850 - 866. 10.1002/acs.1176.

[2] Conte, Claudia & de Alteriis, Giorgio & Schiano Lo Moriello, Rosario & Accardo, Domenico & Rufino, Giancarlo. (2021). Drone Trajectory Segmentation for Real-Time and Adaptive Time-Of-Flight Prediction. Drones. 5. 62. 10.3390/drones5030062.

[3] European Aviation Safety Agency (EASA). Introduction of a regulatory framework for the operation of drones. First Edition (2017).

[4] DJI Sciences and Technologies Ltd. Mavic Pro datasheet, [Online] Available: https://www.dji.com/es/mavic/info [Accessed: 06-Feb-2022]

[5] MAVLink, [Online] Available: https://mavlink.io/en/ [Accessed: 06-Feb-2022]

[6] MAVLink, [Online] Available: https://mavlink.io/en/messages/common.html [Accessed: 06-Feb-2022]

[7] ArduPilot. Mission Planner, [Online] Available: https://ardupilot.org/planner/ [Accessed: 06-Feb-2022]

[8] Bitbucket. Icarus UPC. [Online] https://bitbucket.org/ICARUSGenericUser/cafe-analyzer/src/master/ [Accessed: 06-Feb-2022]

[9] International Civil Aviation Organization (ICAO) RNAV Procedure Design (2017)

APPENDICES

TITLE: Drone trajectory prediction toolset

DEGREE: Doble Grau en Enginyeria de Sistemes de Telecomunicació i Enginyeria d'Aeronavegació

AUTHOR: Daniel Gutierrez Herranz

ADVISOR: Enric Pastor Llorens

DATE: 8 de febrer del 2022

APPENDIX A. DRONE FLIGHT PLAN DESIGN

This appendix gathers all the flight plans mentioned throughout the thesis document, in addition to explaining how the design of the flight plans have been made.

A.1 Flight plan design

The file format must be correct so that when the software reads it there is no error when interpreting and computing the desired trajectory. To achieve this, every flight plan must follow the format of Image 63. It is extracted from the MAVLink guide (see [5]).

```
QGC WPL 110
0       1       0       16      0       0       0       0       40.013292       0.036989        0.700000        1
1       0       3       22      0.00000000      0.00000000      0.00000000      0.00000000      40.0132924      0.0369887       20.000000       1
2       0       3       16      0.00000000      0.00000000      0.00000000      0.00000000      40.01265660     0.03820540      20.000000       1
3       0       3       16      0.00000000      0.00000000      0.00000000      0.00000000      40.01265660     0.03820540      40.000000       1
4       0       3       16      0.00000000      0.00000000      0.00000000      0.00000000      40.01204440     0.03936140      40.000000       1
5       0       3       16      0.00000000      0.00000000      0.00000000      0.00000000      40.01227450     0.04103510      40.000000       1
6       0       3       16      0.00000000      0.00000000      0.00000000      0.00000000      40.01249020     0.04253180      60.000000       1
7       0       3       16      0.00000000      0.00000000      0.00000000      0.00000000      40.01274290     0.04421090      60.000000       1
8       0       3       21      0.00000000      0.00000000      0.00000000      0.00000000      0.00000000      0.00000000      0.000000        1
```

Fig 63 Flight Plan Template

The first line contains the file format and version information, while subsequent the lines are mission items. The first field correspond to the item index. Field number four corresponds to the command that the drone must carry out. The list of commands supported by the software are as follows:

- Command #16: Navigate to waypoint. Fields nine, ten and eleven correspond to latitude, longitude and altitude respectively. This command is used both to indicate climbs/descents and horizontal segments. The first #16 indicates a take-off event.

- Command #19: Loiter at specified latitude, longitude and altitude for a certain amount of time. Indeed, it is a ‘Navigate to waypoint’ command ended with a delay. Filed number five indicates the time to loiter, whereas, fields number nine, ten and eleven correspond to latitude, longitude and altitude.

- Command #21: Land at a location. Fields nine, ten and eleven correspond to latitude, longitude and altitude where drone must land.

- Command #93: Delay the next navigation command a number of seconds or until a specified time. Field number five is used to set the number of seconds to hold on the air. Fields six, seven and eight

correspond to hour, minute, second until the next command must be delayed.

- Command #178: Change speed and/or throttle set points. Field number five indicates the kind of speed to change (0=Airspeed, 1=Ground Speed, 2=Climb Speed, 3=Descent Speed) while, filed number six is for the new speed.

When the angle between two consecutive straight segments and/or non-vertical climb/descents is greater that a defined threshold (22 degrees obtained from a tuning process), the transition between this two events must be considered as a turn.

A.2 Designed flight plans

In this section of the appendix, all used flight plans during the development of the thesis are listed. These flight plans have been designed during the elaboration of the project so, together with the real data, be able to obtain a model that explains the behaviour of the drone given a well-defined trajectory.

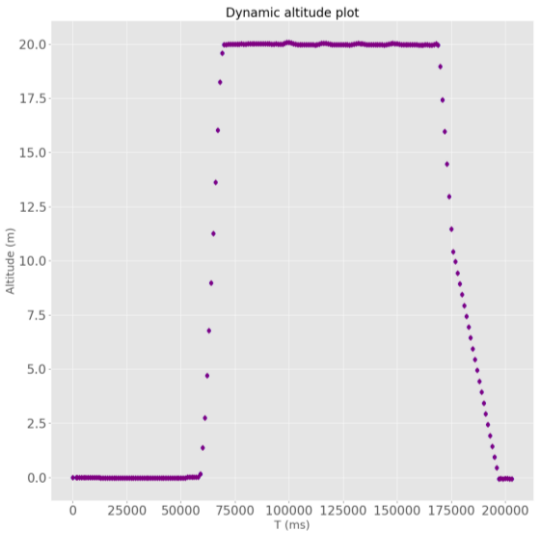


Fig 64 Altitude Plot Triangle

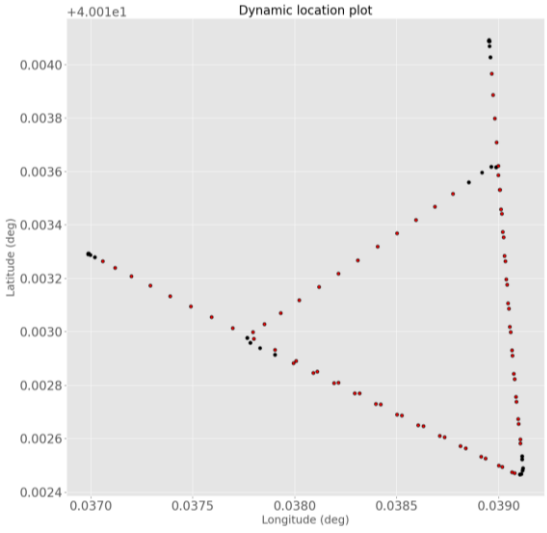


Fig 65 Location Plot Triangle

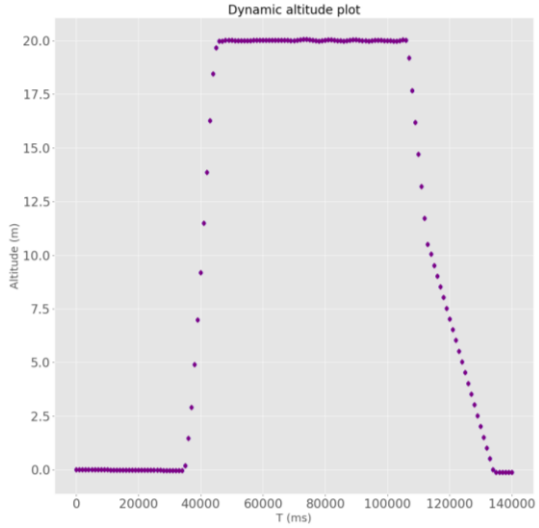


Fig 66 Altitude Plot Tetragon

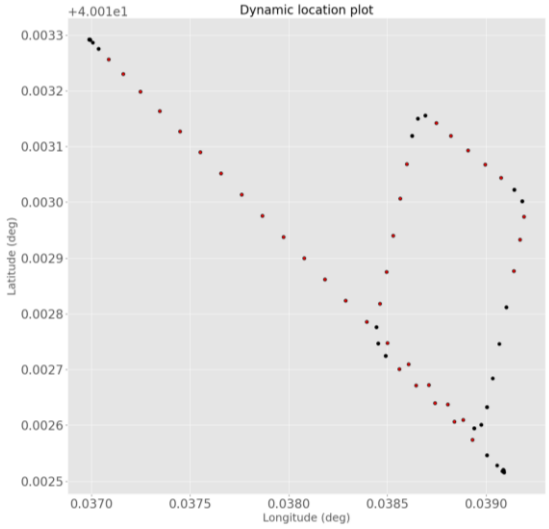


Fig 67 Location Plot Tetragon

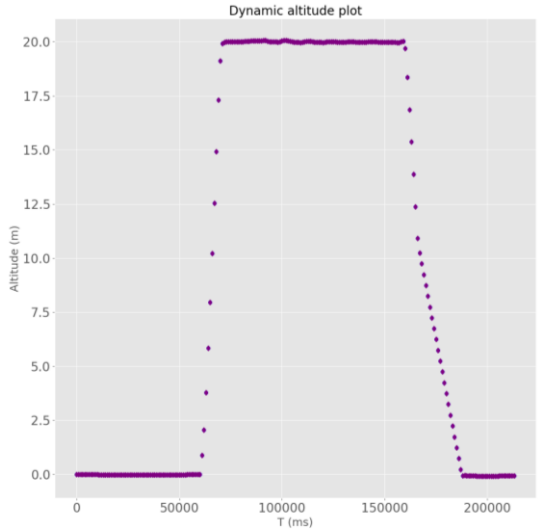


Fig 68 Altitude Plot Pentagon

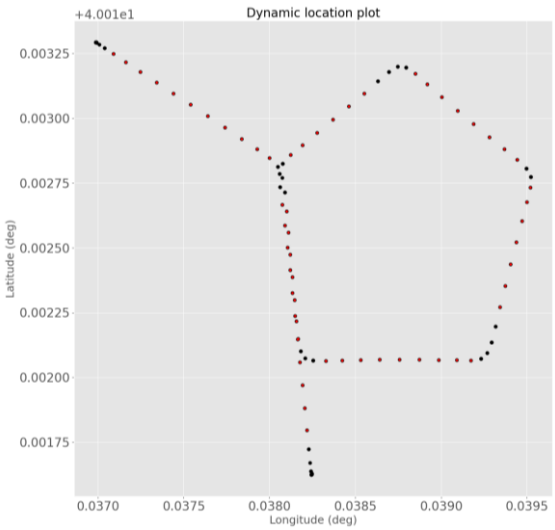


Fig 69 Location Plot Pentagon

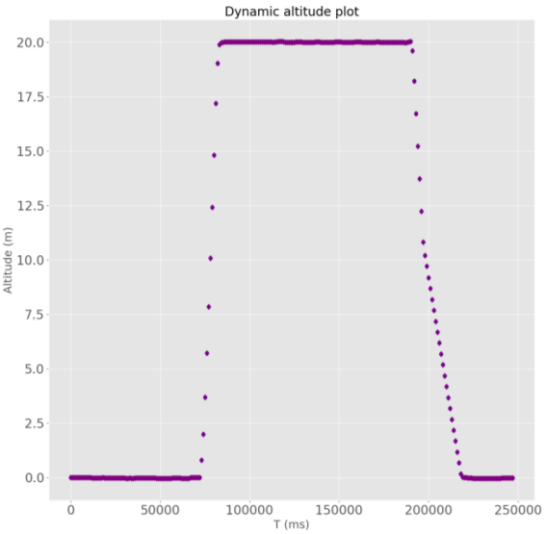


Fig 70 Altitude Plot Hexagon

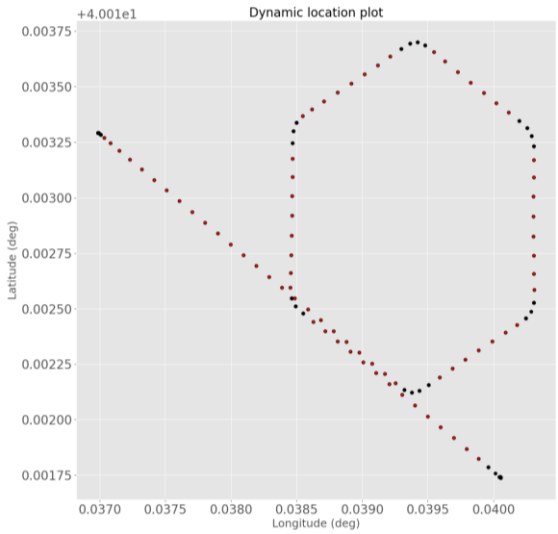


Fig 71 Location Plot Hexagon

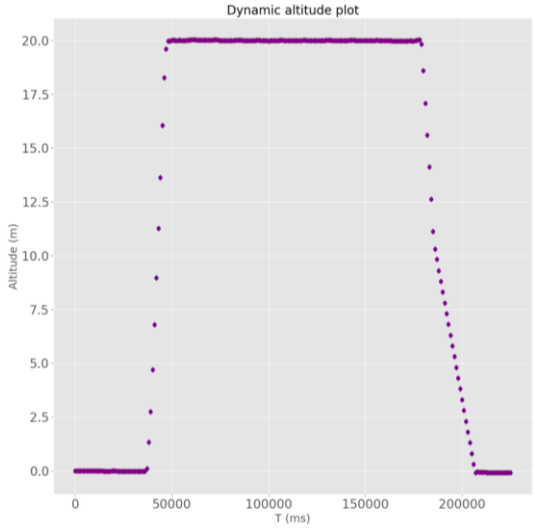


Fig 72 Altitude Plot Heptagon

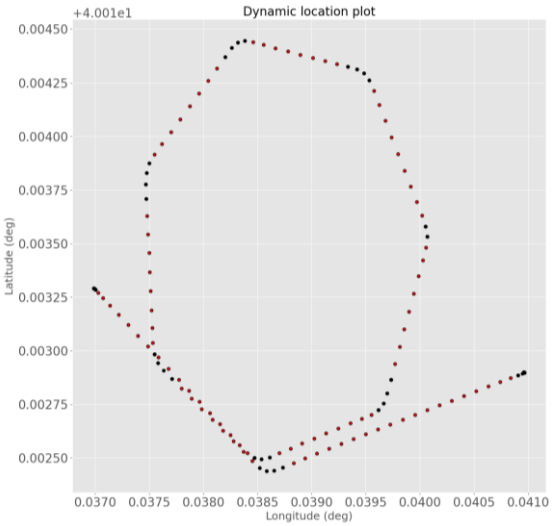


Fig 73 Location Plot Heptagon

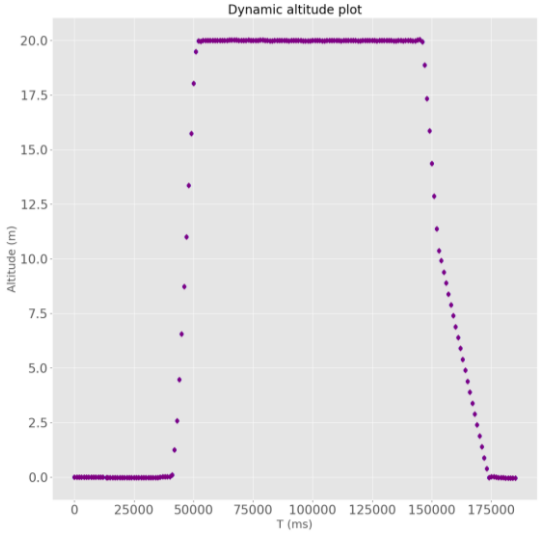


Fig 74 Altitude Plot Octagon

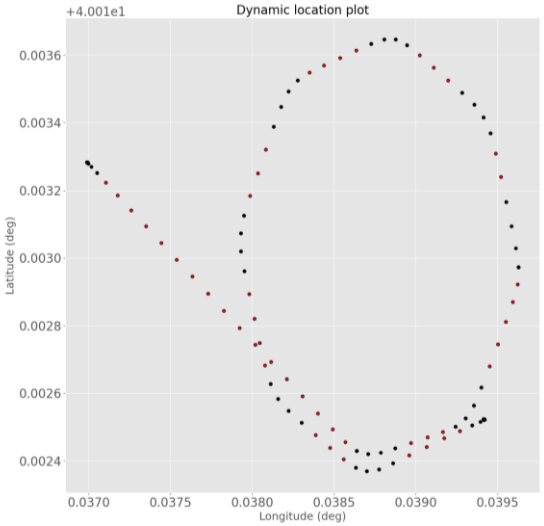


Fig 75 Location Plot Octagon

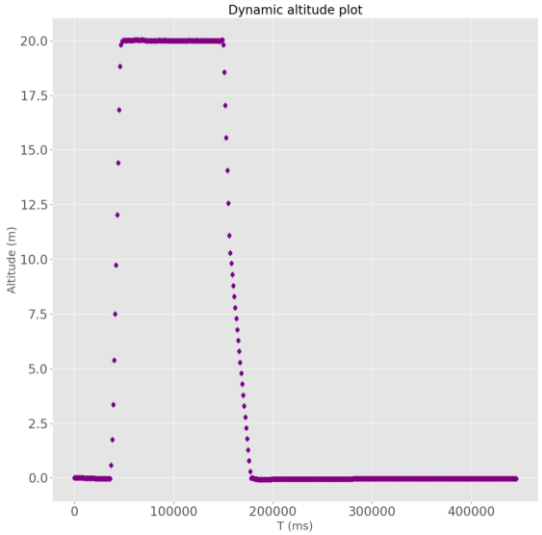


Fig 76 Altitude Plot Nonagon

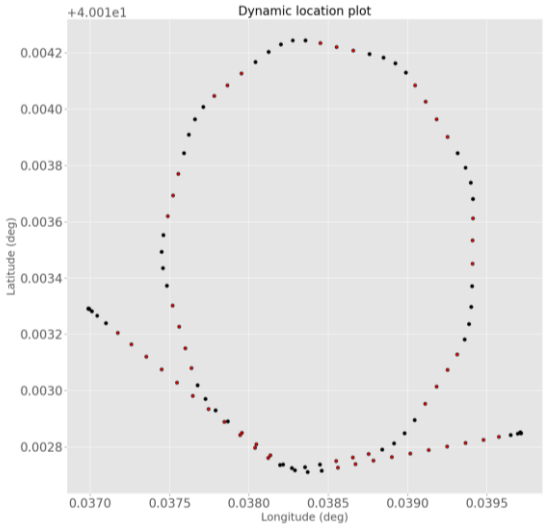


Fig 77 Location Plot Nonagon

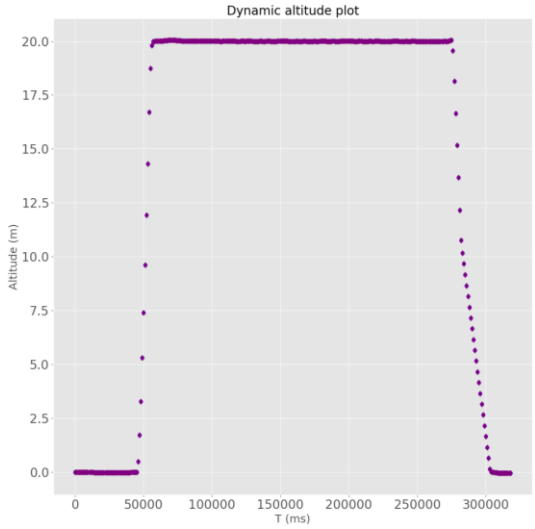


Fig 78 Altitude Plot Decagon

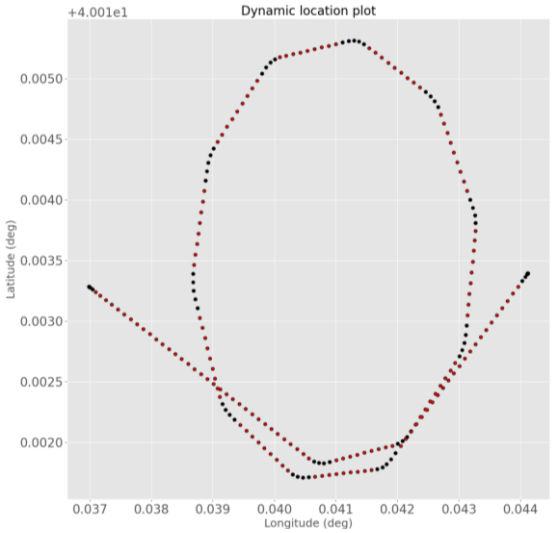


Fig 79 Location Plot Decagon

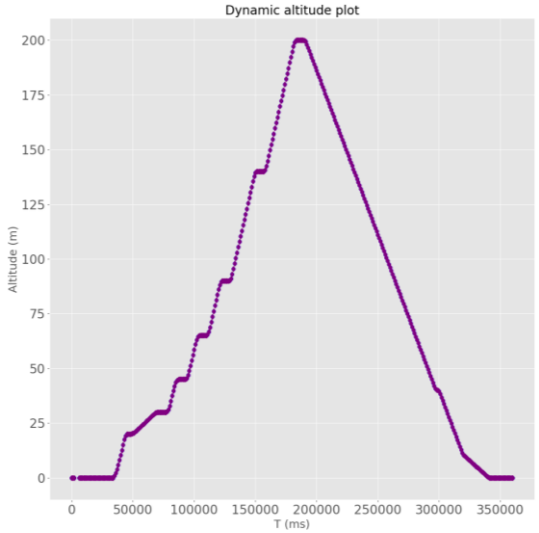


Fig 80 Altitude Plot Hippodrome Tetragon

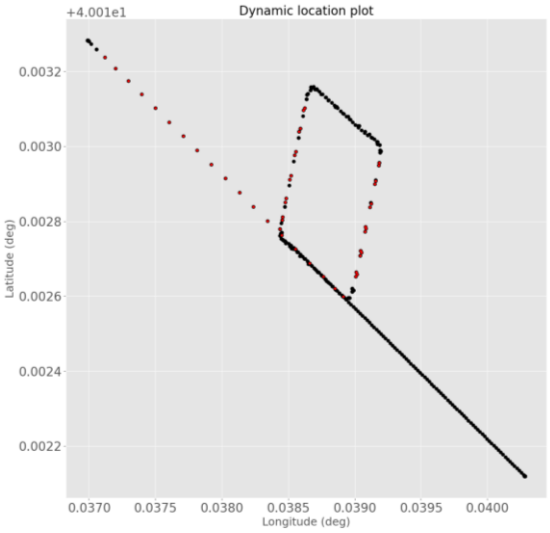


Fig 81 Location Plot Hippodrome Tetragon

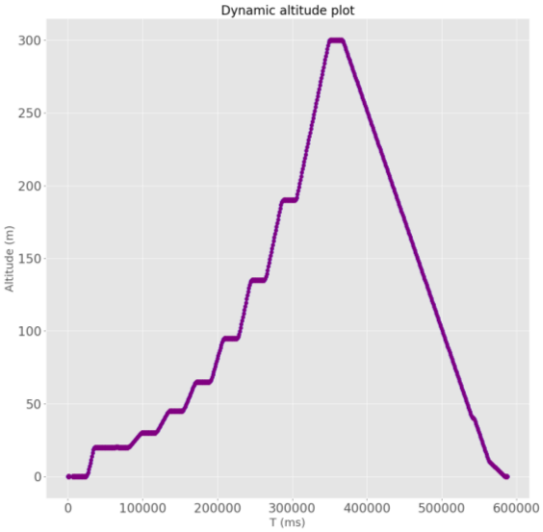


Fig 82 Altitude Plot Hippodrome Pentagon

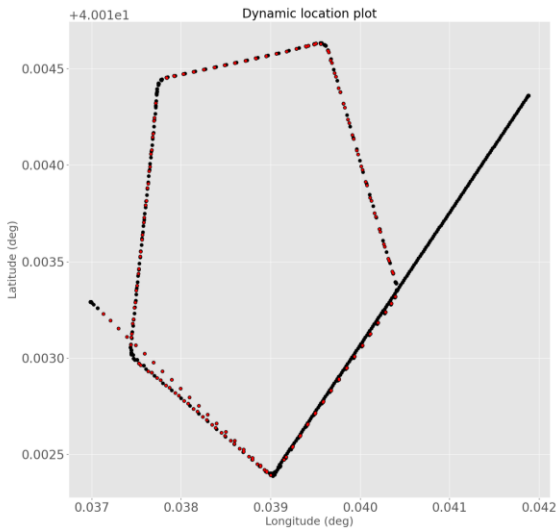


Fig 83 Location Plot Hippodrome Pentagon

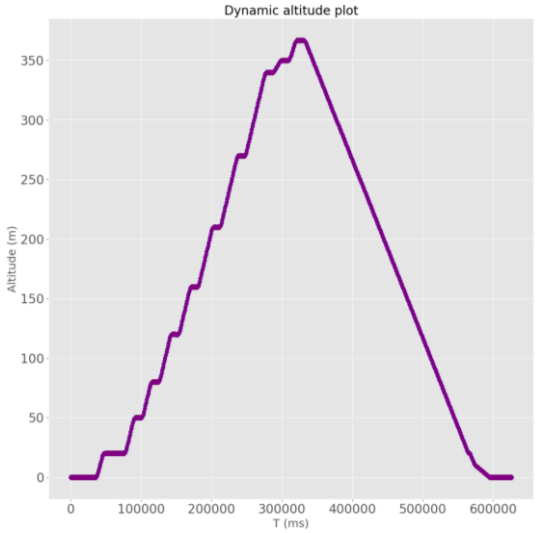


Fig 84 Altitude Plot Hippodrome Hexagon

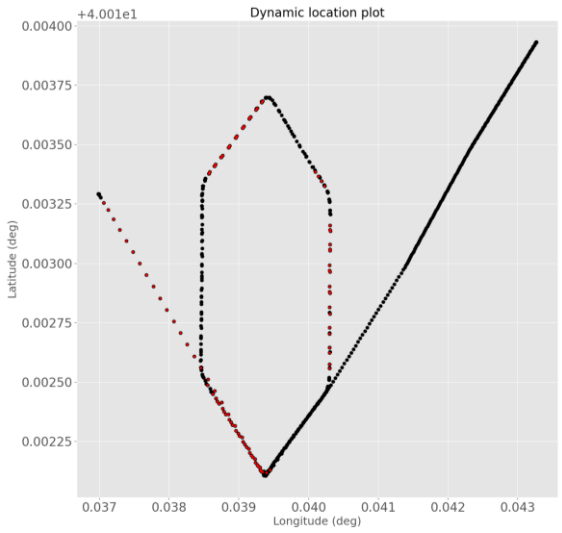


Fig 85 Location Plot Hippodrome Hexagon

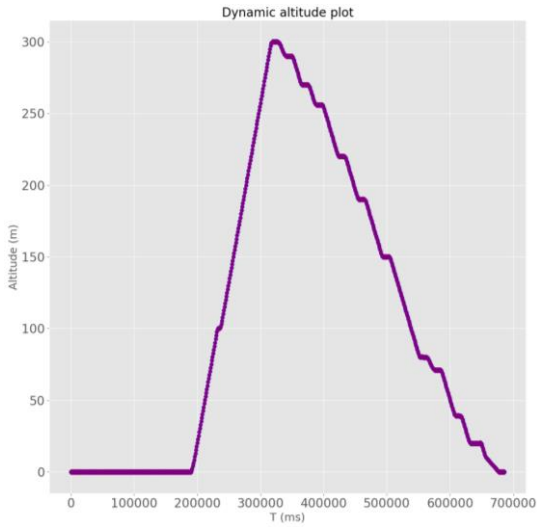


Fig 86 Altitude Plot Hippodrome Hexagon II

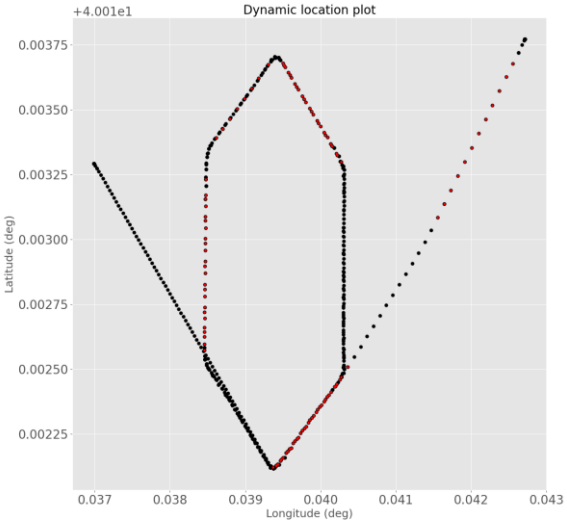


Fig 87 Location Plot Hippodrome Hexagon II

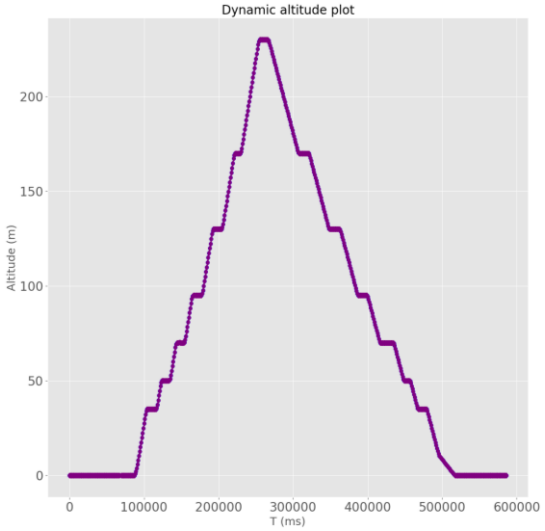


Fig 88 Altitude Plot Vertical Changes

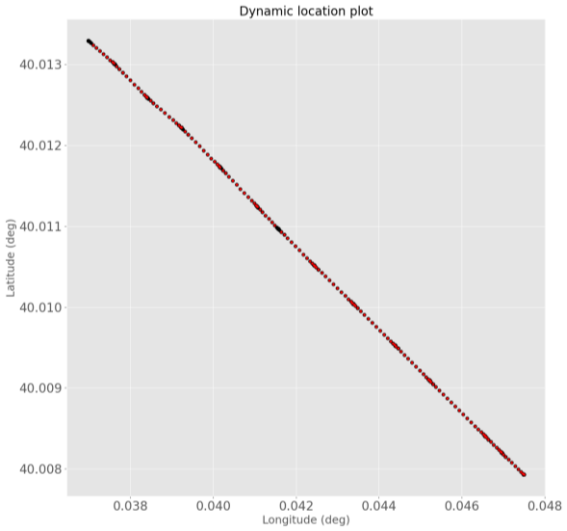


Fig 89 Location Plot Hippodrome Vertical Changes

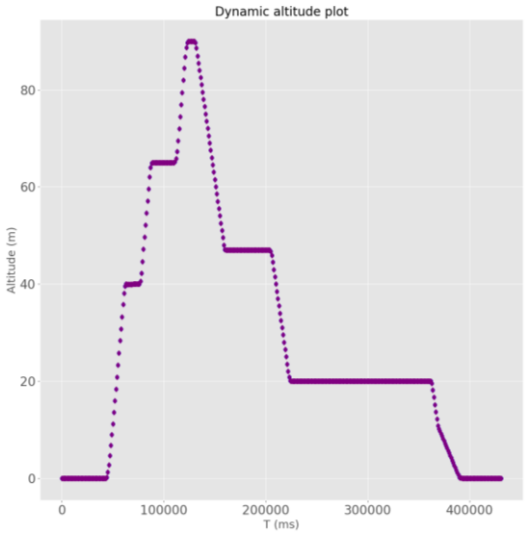


Fig 90 Altitude Plot Test

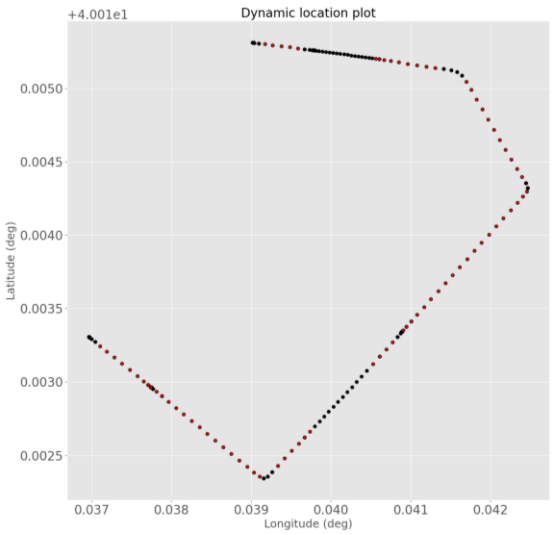


Fig 91 Location Plot Test

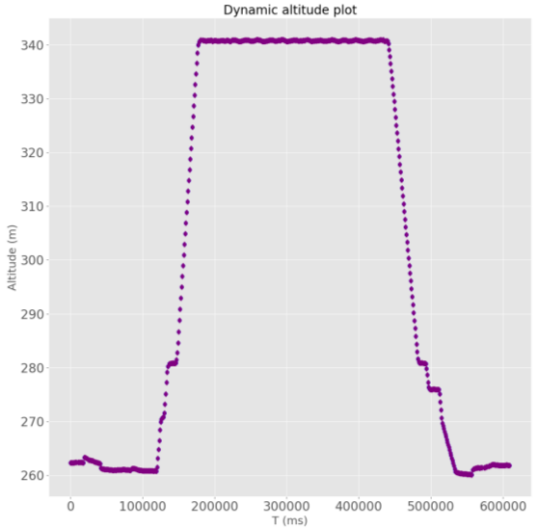


Fig 92 Altitude Plot FP1

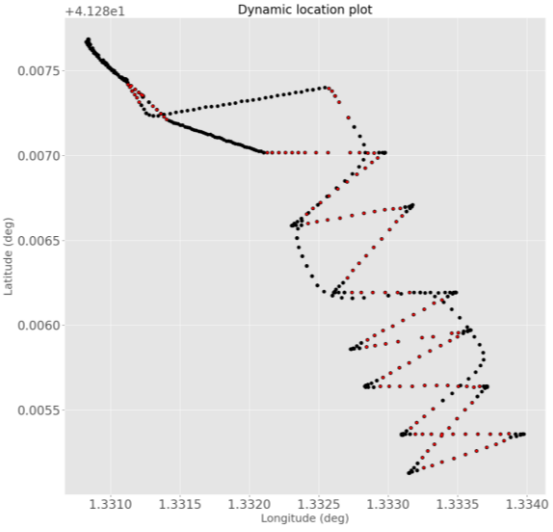


Fig 93 Location Plot FP1

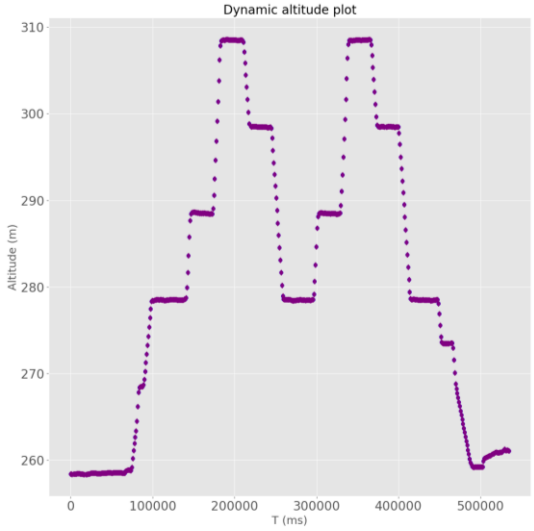


Fig 94 Altitude Plot FP2

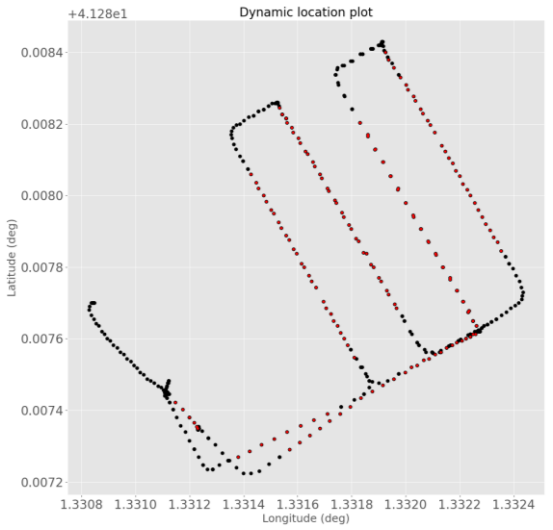


Fig 95 Location Plot FP2

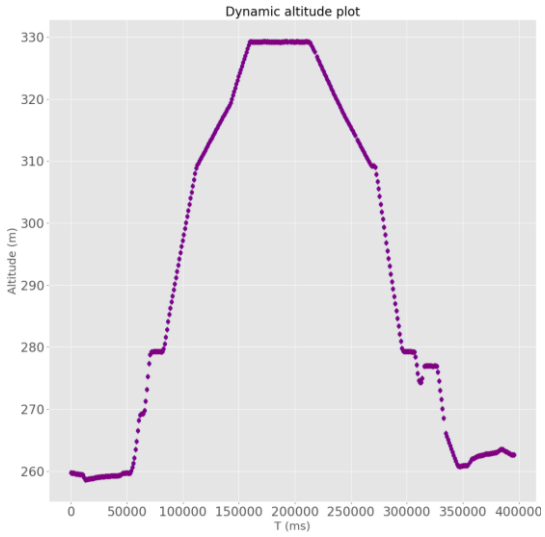


Fig 96 Altitude Plot FP3

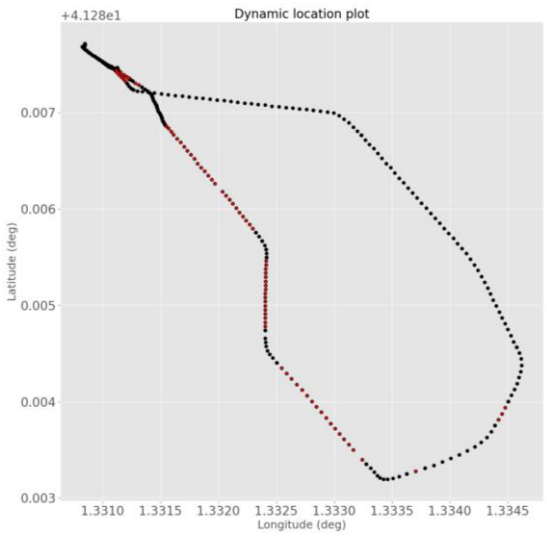


Fig 97 Location Plot FP3

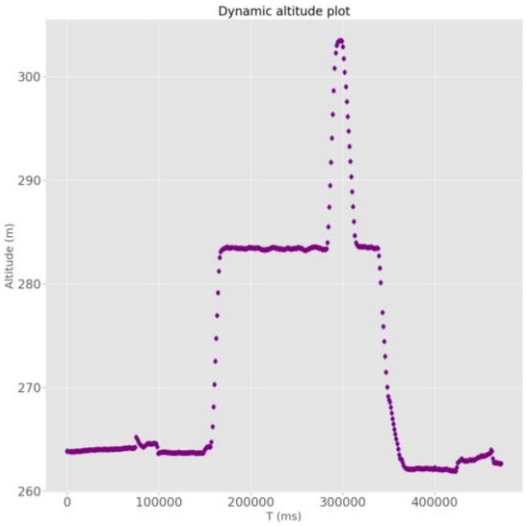


Fig 98 Altitude Plot FP4

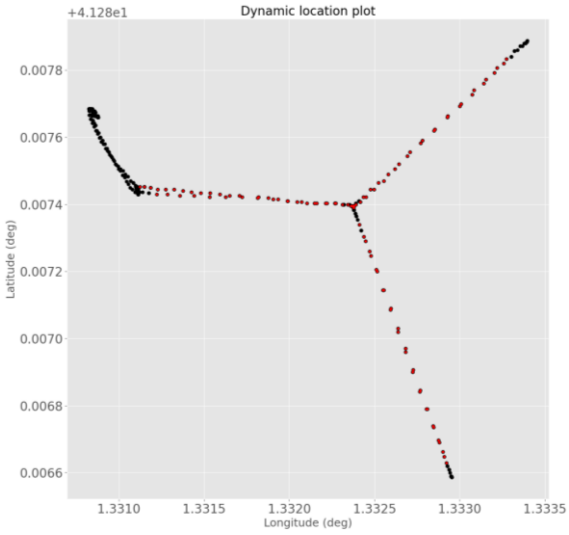


Fig 99 Location Plot FP4

APPENDIX B. USE OF SOFTWARE APPLICATION

The objective of this appendix is to act as a tutorial to the application, programmed in Python. Explanations about the main methods and functionalities used in the application are listed here. All the Python program files used in the application can be found in the GitHub repo (see [8]).

B.1 Folder distribution

The files are organized in folders according to their functionality. The folder called '*Mocca'* contains everything related to the decryption of MAVLink messages. Together with the reading of the MAVLink telemetry, a first processing of the data is carried out. This allow the data to be ready for use, such as adjusting units or computing intermediate parameters.

Folder called '*Performance_Analyzer*' collects everything concerning to the identification, analysis, processiong and refinement of the different events that define a flight. As it is explained before, in Chapter 3, the identification process is done by studying the samples sent by the drone using a temporary buffer. For this reason, the circular buffer is also located in this section. When a figure is plotted, the files located in the '*Plots*' folder are imported and these are responsible for processing the data and saving the corresponding figure.

The '*TrajectoryGenerator*' folder contains the files that allow the randomly generation of flight plans. Through a *main()*, the function generates N number of flight plans with multiples altitude changes from *hmin* to *hmax* and it has as input parameters the following ones:

- nOfFP: number of flight plans to generate.
- hTakeOff, hLanding: define the altitude at which the flight plan must start and end.
- hmax, hmin: defines the upper and lower altitude boundary that cannot be crossed by the flight.
- intervalax, intervalmin: restrict the minimum and maximum altitude change.
- nOfSamples: defines the number of altitude changes.

B.2 Input/output files

There are two types of files that the software application needs to work properly. If what is contemplate is to analyse and/or verify that the behaviour of the drone is as expected, the required file corresponds to the one generated by the drone itself that contains the flight data. This file with extension *.dat* contains the information generated at each moment by the system. Before starting the flight, the operator sets the rate of message generation and selects which messages the drone will send during the flight to the ground station. The software decodes takes theses files in order to obtain the data by decoding them. Image 100 corresponds to an extract of one of this data files. Each line of the file corresponds to a message generated and sent by the drone. Several types of messages can be observed in the image, among them there is the so-called 'mlGlobalPositionInt', which is the one that has been used for the development of the thesis.

```
mlSysStatus; 1580295141379; 1580295141379; 56753199; 23198767; 24222767; 686; 0; -1; -1; 0; 0; 0
mlAttitude; 1580295142364; 1580295142364; -0.025848448; -0.03345882; -0.5550332; -0.011082497; -0.03591863; 0.012517409
mlGlobalPositionInt; 1580295142367; 1580295142367; 41.28806; 1.3314269; 278.53; 278.53; 3.91; -2.34; -0.01; 328.16
mlScaledIMU; 1580295142379; 1580295142379; 0; 0.045; 0.068; -1.071; 0.053; -0.015; 0.004; 0.302; -0.01; 0.425
mlScaledPressure; 1580295142379; 1580295142379; 0; 989.3187; 0.0; 37.02
mlScaledPressure; 1580295142382; 1580295142382; 0; 989.97144; 0.0; 28.36
mlNavControllerOut; 1580295142387; 1580295142387; -0.92000514; -1.5362115; -31.0; 30.0; 21.0; 0.018304443; 0.0; 10.466297
mlSysStatus; 1580295142387; 1580295142387; 56753199; 23198767; 24222767; 642; 0; -1; -1; 0; 0; 0
mlGlobalPositionInt; 1580295143364; 1580295143364; 41.288097; 1.3313982; 278.52; 278.52; 3.89; -2.4; -0.02; 327.69
mlAttitude; 1580295143364; 1580295143364; -0.011457173; -0.03215035; -0.56398106; -0.024224842; 0.053194817; -0.0058382796
mlScaledIMU; 1580295143384; 1580295143384; 0; -0.006; -0.048; -0.866; -0.01; 0.092; 0.006; 0.299; 0.0; 0.423
mlScaledPressure; 1580295143384; 1580295143384; 0; 989.2659; 0.0; 37.05
mlScaledPressure; 1580295143387; 1580295143387; 0; 989.9658; 0.0; 28.38
mlNavControllerOut; 1580295143389; 1580295143389; 1.6710421; 5.5331907; -27.0; 41.0; 19.0; 0.023811035; 0.0; 6.0099335
```

Fig 100 Telemetry

If what is desired is to simulate a flight using the model extracted in this project, the software expects as input parameter a file with extension *.waypoints*. This file indeed is a flight plan with the commands the drone must perform to do the defined trajectory. All commands supported by the software have been explained in depth in appendix A.1.

```
QGC WPL 110
0       1       0       16      0       0       0       0       41.28744125     1.33110893      0       1
0       0       3       22      0.000000        0.000000        0.000000        0.000000        41.28744125     1.33110893      5       1
1       0       3       178     1.000000        2       0.000000        0.000000        41.28744125     1.33110893      5       1
2       0       3       16      1.00000 0.000000        0.000000        0.000000        41.28744125     1.33110893      10      1
3       0       3       19      3.000000        0.000000        0.000000        0.000000        41.28744125     1.33110893      10      1
4       0       3       16      1.000000        0.000000        0.000000        0.000000        41.28744125     1.33110893      10      1
5       0       3       178     1.000000        3       0.000000        0.000000        41.28744125     1.33110893      20      1
6       0       3       16      0.000000        0.000000        0.000000        0.000000        41.28722382     1.33126831      20      1
7       0       3       178     1.000000        5       0.000000        0.000000        41.28722382     1.33126831      20      1
8       0       3       16      0.000000        0.000000        0.000000        0.000000        41.28745651     1.33188462      20      1
9       0       3       16      0.000000        0.000000        0.000000        0.000000        41.28817749     1.33133233      20      1
10      0       3       16      0.000000        0.000000        0.000000        0.000000        41.28826141     1.3315239       20      1
```

Fig 101 Flight Plan simulated

One of the main processes when extracting the model has been to look for relationships and dependencies between the variables that conform a flight. Each time the study was done to find any kind of reliance between several parameters, it was represented graphically and the help of the curve fitting process, it was analysed whether or not there was some kind of function that relates this parameters. Throughout the document, some of these images can be found.