

## Master thesis : RC Helicopter Control

**Auteur :** Pirottin, Thomas

**Promoteur(s) :** Redouté, Jean-Michel; 12799

**Faculté :** Faculté des Sciences appliquées

**Diplôme :** Master : ingénieur civil électricien, à finalité spécialisée en "signal processing and intelligent robotics"

**Année académique :** 2021-2022

**URI/URL :** <http://hdl.handle.net/2268.2/14457>

---

### Avertissement à l'attention des usagers :

Tous les documents placés en accès ouvert sur le site le site MatheO sont protégés par le droit d'auteur. Conformément aux principes énoncés par la "Budapest Open Access Initiative"(BOAI, 2002), l'utilisateur du site peut lire, télécharger, copier, transmettre, imprimer, chercher ou faire un lien vers le texte intégral de ces documents, les disséquer pour les indexer, s'en servir de données pour un logiciel, ou s'en servir à toute autre fin légale (ou prévue par la réglementation relative au droit d'auteur). Toute utilisation du document à des fins commerciales est strictement interdite.

Par ailleurs, l'utilisateur s'engage à respecter les droits moraux de l'auteur, principalement le droit à l'intégrité de l'oeuvre et le droit de paternité et ce dans toute utilisation que l'utilisateur entreprend. Ainsi, à titre d'exemple, lorsqu'il reproduira un document par extrait ou dans son intégralité, l'utilisateur citera de manière complète les sources telles que mentionnées ci-dessus. Toute utilisation non explicitement autorisée ci-avant (telle que par exemple, la modification du document ou son résumé) nécessite l'autorisation préalable et expresse des auteurs ou de leurs ayants droit.

---



---

Master thesis  
RC helicopter control

---

Thomas PIROTTIN

June 2022

Master's thesis carried out to obtain the degree of  
Master of Science in Electrical Engineering

## Abstract

Remote controlled helicopters (usually shortened to RC helicopters) are a specific type of rotary-wing drone. Their main advantage relies on their better power efficiency and reactivity than traditional “multicopter” drones (an example of RC helicopter and multicopter is shown on Figure 1).

However, they are unstable in flight and their greater reactivity makes their control quite challenging. In this master thesis, we will explore how to ease RC helicopter control, by adding a flight controller allowing any newcomer to control the helicopter in a much simpler way.

We will explore the hardware and software required to add such a flight controller to an existing RC Helicopter and will discuss our return of experience on adding a Pixhawk 4 mini flight controller running ArduPilot to the Devil 380 RC Helicopter.



(d) RC helicopter.



(e) Multicopter.

Figure 1: RC helicopter and Multicopter examples.

# Acknowledgements

I would like to thank many people without whom I could not have completed this master thesis.

Firstly, I would like to thank Mr. Christophe Greffe who proposed this subject and Pr. Jean-Michel Redouté who accepted to promote it.

I also want to thank Mr. Geoffrey Mormal and the entire ALX-System team who accepted me as an intern to work on the development of this master thesis.

I would particularly like to thank Adrien and Xavier who helped me a lot with the practical aspects.

A big thank you also to all the other people who helped me : Mr. Angel Calderon, Luca and François, Sacha and Colin, Maxime and Gilles.

Finally, I would like to thank my parents, and all my family, for their support and encouragement, without forgetting my friends (and in particular Nicolas).

Thank you all for your help and support!

# Table of Contents

|  |            |
|--|------------|
| <b>Abstract</b>                                    | <b>I</b>   |
| <b>Acknowledgements</b>                            | <b>II</b>  |
| <b>Table of Contents</b>                           | <b>III</b> |
| <b>List of Figures</b>                             | <b>V</b>   |
| <b>List of Tables</b>                              | <b>VI</b>  |
| <b>Introduction</b>                                | <b>1</b>   |
| <b>1 The theory of helicopters</b>                 | <b>4</b>   |
| 1.1 The vertical force (the lift)                  | 4          |
| 1.2 The horizontal force                           | 5          |
| 1.3 The helicopter angles                          | 7          |
| 1.4 How to manage the collective and cyclic pitch? | 7          |
| 1.5 Controlling the helicopter yaw angle           | 9          |
| 1.6 Other side effect forces                       | 9          |
| 1.7 How to move a helicopter summary               | 10         |
| <b>2 Introduction to the flight controller</b>     | <b>12</b>  |
| <b>3 Hardware description</b>                      | <b>14</b>  |
| 3.1 Sensors  | 14         |
| 3.2 Actuators                                      | 14         |
| 3.3 Pilot's communication                          | 16         |
| 3.4 Flight controller board                        | 17         |
| <b>4 Software description</b>                      | <b>19</b>  |
| 4.1 Software overview                              | 19         |
| 4.2 Sensors data standardisation                   | 20         |
| 4.2.1 Role and behaviour                           | 20         |
| 4.2.2 Parameterisation                             | 21         |
| 4.3 Pilot data standardisation                     | 21         |
| 4.3.1 Role and behaviour                           | 21         |
| 4.3.2 Parameterisation                             | 22         |
| 4.4 Core algorithm                                 | 23         |
| 4.4.1 Current state management                     | 23         |
| 4.4.1.1 Role and behaviour                         | 23         |
| 4.4.1.2 Parameterisation                           | 24         |
| 4.4.2 Next state interpolation                     | 24         |
| 4.4.2.1 Role and behaviour                         | 24         |
| 4.4.2.2 Parameterisation                           | 26         |
| 4.4.3 Main control system                          | 26         |
| 4.4.3.1 Role and behaviour                         | 26         |
| 4.4.3.2 Parameterisation                           | 28         |
| 4.5 PWM signal transformation                      | 28         |
| 4.5.1 Role and behaviour                           | 28         |
| 4.5.2 Parameterisation                             | 29         |
| <b>5 Flight tests</b>                              | <b>31</b>  |
| 5.1 Before the flight tests                        | 31         |
| 5.2 First flight tests : ground tests!             | 32         |
| 5.3 First take off                                 | 32         |
| 5.4 Advanced flight tests                          | 33         |

|          |  |           |
|----------|--|-----------|
| <b>6</b> | <b>Attention point : vibrations</b>  | <b>35</b> |
| 6.1      | Blade tracking . . . . .   | 35        |
| 6.2      | Vibration damping . . . . .  | 36        |
| 6.3      | Software filters . . . . .   | 36        |
| <b>7</b> | <b>Limitations and Further Improvements</b>  | <b>38</b> |
|          | <b>Conclusion</b>  | <b>39</b> |
|          | <b>References</b>  | <b>40</b> |
|          | <b>Appendices</b>  | <b>41</b> |
| A        | More advanced description of the flight controller board and its ecosystem . . . . . | 41        |
| B        | Sensors parameters and calibration . . . . .   | 42        |
| B.1      | Board orientation settings . . . . .   | 42        |
| B.2      | IMUs calibration and parameter settings . . . . .                                    | 43        |
| B.3      | Magnetometer calibration and parameter settings . . . . .                            | 44        |
| B.4      | GPS parameter settings . . . . .   | 45        |
| B.5      | Battery monitoring calibration and parameter settings . . . . .                      | 45        |
| C        | Remote control and telemetry parameters and calibration . . . . .                    | 46        |
| C.1      | Remote control channel calibration and parameter settings . . . . .                  | 46        |
| C.2      | Telemetry module calibration . . . . .   | 48        |
| D        | Current state management parameters . . . . .  | 49        |
| E        | Next state interpolation parameters . . . . .  | 49        |
| E.1      | Flight mode selection parameters . . . . .   | 49        |
| E.2      | Stabilised flight mode parameters . . . . .  | 50        |
| E.3      | AltHold flight mode parameters . . . . .   | 50        |
| E.4      | PosHold flight mode parameters . . . . .   | 51        |
| E.5      | Loiter flight mode parameters . . . . .  | 51        |
| E.6      | Auto flight mode parameters . . . . .  | 51        |
| F        | Main control system parameters . . . . .   | 52        |
| G        | PWM signal transformation parameters and calibration . . . . .                       | 54        |

# List of Figures

|      |  |    |
|------|--|----|
| 1    | RC helicopter and Multicopter examples. . . . .  | I  |
| 2    | Fixed-wing drone example. . . . .  | 1  |
| 3    | Quadcopter drone example. . . . .  | 2  |
| 4    | RC helicopter drone example. . . . .   | 2  |
| 1.1  | Freedom of movement of a helicopter in flight. . . . .   | 4  |
| 1.2  | Thrust as the reaction force of the air pushed down. . . . .                                     | 5  |
| 1.3  | Blade pitch angle $\theta$ . . . . .   | 5  |
| 1.4  | Thrust decomposition in its lift and propulsive force component. . . . .                         | 6  |
| 1.5  | Main rotor generates a uniform lift. . . . .   | 6  |
| 1.6  | Main blade revolution angle $\psi$ . . . . .   | 7  |
| 1.7  | Main rotor generates a non-uniform lift. . . . .   | 7  |
| 1.8  | Helicopter frame angles. . . . .   | 8  |
| 1.9  | Schema of a swash plate mechanism. . . . .   | 8  |
| 1.10 | Swash plate of the Devil 380. . . . .  | 8  |
| 1.11 | Tail rotor thrust counteracts the main rotor torque. . . . .                                     | 9  |
| 2.1  | Pilot sends commands to the helicopter. . . . .  | 12 |
| 2.2  | Pilot sends commands to the helicopter and captures information. . . . .                         | 12 |
| 2.3  | Pilot sends command to the helicopter via the flight controller. . . . .                         | 13 |
| 2.4  | Flight controller pilots the helicopter in an automated flight. . . . .                          | 13 |
| 3.1  | Schema of the swash plate configuration of the Devil 380. . . . .                                | 17 |
| 3.2  | Telemetry modules connecting the flight controller to the ground control station. . . . .        | 17 |
| 3.3  | Flight controller ecosystem overview. . . . .  | 18 |
| 4.1  | Basic overview of the flight controller inputs and outputs. . . . .                              | 19 |
| 4.2  | High-level decomposition of the flight controller task. . . . .                                  | 20 |
| 4.3  | Core Algorithm decomposition. . . . .  | 23 |
| 4.4  | Current state management process using a Kalman filter. . . . .                                  | 24 |
| 4.5  | Next state interpolation interactions. . . . .   | 25 |
| 4.6  | Main control system interactions. . . . .  | 26 |
| 4.7  | Main control system block diagram[1]. . . . .  | 27 |
| 4.8  | PWM signal transformation overview. . . . .  | 29 |
| 5.1  | Evolution of the training kit feet length along the flight tests. . . . .                        | 32 |
| 5.2  | Definition of an autonomous mission using Mission Planner. . . . .                               | 34 |
| 6.1  | Main rotor blades of the Devil 380 after matching of their weight and centre of gravity. . . . . | 35 |
| 6.2  | Visual appearance of non-tracking and tracking blades. . . . .                                   | 36 |
| 6.3  | Pitch link of the Devil 380 with adaptable length. . . . .                                       | 36 |
| 6.4  | 3D printed box, 3D printed platform and rubber stoppers used for vibration damping. . . . .      | 37 |
| A.1  | Input and output ports of the Piwhawk 4 mini[2]. . . . .   | 41 |
| B.1  | Level and Accelerometer calibration page on Mission Planner. . . . .                             | 43 |
| B.2  | Magnetometer calibration page on Mission Planner. . . . .  | 44 |
| B.3  | Battery monitoring board selection page on Mission Planner. . . . .                              | 46 |
| C.1  | Remote control calibration page on Mission Planner. . . . .                                      | 47 |
| C.2  | Telemetry module binding page on Mission Planner. . . . .  | 49 |

## List of Tables

|     |   |    |
|-----|---|----|
| 1   | Advantages of Multicopters and RC helicopters. . . . .                            | 2  |
| 2   | List of components used. . . . .  | 3  |
| 1.1 | Summary of the helicopter physics and how to control it. . . . .                  | 10 |
| 3.1 | List of hardware devices containing sensors and their respective sensors. . . . . | 15 |
| 3.2 | Actuators on the Devil 380. . . . .   | 16 |
| 4.1 | Main steps of the main control system. . . . .                                    | 27 |
| B.1 | Board orientation parameters. . . . .   | 43 |
| B.2 | IMU parameters. . . . .   | 44 |
| B.3 | Magnetometer parameters. . . . .  | 45 |
| B.4 | GPS parameters. . . . .   | 45 |
| B.5 | Battery parameters. . . . .   | 46 |
| C.1 | Remote control parameters. . . . .  | 48 |
| D.1 | Extended Kalman filter parameters. . . . .  | 49 |
| E.1 | Flight mode selection parameters. . . . .   | 49 |
| E.2 | Stabilised flight mode parameters. . . . .  | 50 |
| E.3 | AltHold flight mode parameters. . . . .   | 50 |
| E.4 | PosHold flight mode parameters. . . . .   | 51 |
| E.5 | Loiter flight mode parameters. . . . .  | 51 |
| E.6 | Auto flight mode parameters. . . . .  | 52 |
| F.1 | Attitude control on the yaw parameters. . . . .                                   | 52 |
| F.2 | Attitude control on the pitch parameters. . . . .                                 | 53 |
| F.3 | Attitude control on the roll parameters. . . . .                                  | 53 |
| G.1 | Side effect compensation mechanisms parameters. . . . .                           | 54 |
| G.2 | PWM output mapping parameters. . . . .  | 54 |
| G.3 | Main motor configuration parameters. . . . .                                      | 54 |
| G.4 | Swash plate configuration parameters. . . . .                                     | 55 |
| G.5 | Tail rotor configuration parameters. . . . .                                      | 55 |



# Introduction

Unmanned Aerial Vehicles (UAVs), also known as drones, are finding a growing place in the current era of new technologies. From military[3] to personal use, their professional applications also increased in diverse and various domains such as cinematography[4], surveillance[5], delivery[6], farming[7], firefighting[8], and so on.

Drones can be divided into two main categories :

- **Fixed-wing** : The principle of operation of these drones is similar to that of aircraft. They need a certain horizontal velocity in order to generate sufficient lift with their wings to make the drone fly. The Figure 2 shows a typical fixed-wing drone.
- **Rotary-wing** : The most common drones, and the ones that most people are familiar with. They use multiple rotors to generate sufficient thrust, allowing them to fly. Their main advantages over the fixed-wing drones are that they can take off and land vertically, they can hover in a fixed position, and they can move in any direction regardless of their heading orientation.



Figure 2: Fixed-wing drone example.

Both categories have their own advantages and disadvantages. In terms of energy consumption, the fixed-wing drones are unmatched as they only need to maintain a minimal horizontal speed whereas the rotary-wing drones have to carry their own weight[9].

On the other hand, the constraint of keeping a horizontal speed to sustain the lift can be very penalising for some applications. As many real-world applications of drones require the capability to hover in a fixed position, the fixed-wing drones will not be further considered here.

Focusing on the rotary-wing drones, essential aspects that will be used to compare them will be their autonomy<sup>1</sup> and their ease of control.

The most well-known flying drones have multiple rotors, that's why they are often called multicopter. The Figure 3 shows a quadcopter, the most common type of multicopter.

Multicopters are controlled by increasing or decreasing the rotation speed of their rotors. This allows them to optimise their blade airfoil to generate a constant thrust at any point. In other words, the pitch angle of the blades is greater near the rotor shaft, because the relative wind speed is lower there than at the extremities. However, the rotor blade length is limited due to their proximity.

Another type of rotary-wing drone is the remote controlled helicopter (called RC helicopter in the following). The Figure 4 shows a typical one.

RC helicopters are controlled by changing the pitch angles of their main rotor blades which have a constant and symmetric airfoil. As there is a single main rotor, its blade length can be much longer than for multicopters.

Despite the optimised airfoil of multicopters, RC helicopters have a greater power efficiency thanks to the length of their main rotor blades. The greater the length, the more efficient it is and a large single rotor is better than multiple

<sup>1</sup>The autonomy represents the mean time of flight without having to recharge the battery or refill the gas tank of the drone. One does not refer here to the number of things that the drone can do independently of the pilot instructions and that can also be referred to as autonomy.



Figure 3: Quadcopter drone example.



Figure 4: RC helicopter drone example.

smaller rotors. This better energy efficiency allows helicopters to have a better autonomy and payload capacity<sup>2</sup> than multicopters[10].

Another major difference resides in their ease of control. Multicopters are easier to control than helicopters. The difficulty in controlling helicopters lies in their strongly coupled and highly nonlinear dynamics[10, 11].

Finally, one last difference we will discuss here resides in the complexity of the mechanics they require. Indeed, multicopters are much simpler than RC helicopters, due to the complexity of managing the blades pitch angle on the latest. This makes RC helicopters less reliable and more costly to maintain than multicopters[10].

In summary, we have two types of rotary wing drones, with their main advantages illustrated by the Table 1.



| Drone Type  | Advantages  |
|---|---|
|  | Easier to control<br>Better reliability<br>Better maintainability |
|  | Better autonomy<br>Better payload capacity                        |

Table 1: Advantages of Multicopters and RC helicopters.

Increasing the autonomy of a drone is a very complex problem. Indeed, a first idea would be to increase the battery capacity (or any other source of energy). Unfortunately, this comes with a corresponding increase of weight which needs to be compensated by an increased thrust. This leads to an increase of power consumption that will limit

<sup>2</sup>The payload capacity is the additional charge to its own weight that a drone can lift.

the gain in autonomy.

Controlling a drone is not an easy problem either. A solution is to use a flight controller in addition to the drone. The advantage of this approach is that it transfers a hardware problem (linked to the physics of the drone) to a software problem.

The benefits of that is the flexibility introduced by the software and the adaptability of the solution to any drone. Indeed, once this solution has been implemented for a specific drone, it becomes easier to simply adapt it to another one and if a new problem or a new limitation is found, it is easier to perform a software update than to rebuild a full drone.

This solution is applied on any kind of drone. The design and parameterisation of a flight controller are, however, more difficult for RC helicopters than for multicopters.

This brings us to the core of this master thesis : stabilise an RC helicopter with a flight controller to make it flyable by a newcomer.

This study aims to tackle the problem from a very practical point of view and will therefore be performed on an actual RC helicopter with an actual flight controller. This solution is mainly based on components listed in Table 2.



|   |  |
|---|--|
| <p>The RC helicopter that will be used is an <i>ALZRC Devil 380 FAST</i><sup>3</sup>.</p>   |    |
| <p>The flight controller hardware is a <i>Pixhawk 4 mini</i><sup>4</sup>.</p>   |   |
| <p>The flight controller software is the <i>ArduCopter firmware</i><sup>5</sup> using the “Heli” airframe.</p> <p>The software is completed by the <i>Mission Planner</i><sup>6</sup> ground control station application, running on Windows.</p> |  |

Table 2: List of components used.

<sup>3</sup>Link to Devil 380 manufacturer website : <http://www.alzrc.com/search.html?cid=64>

<sup>4</sup>Link to Pixhawk 4 mini manufacturer website : <http://www.holybro.com/product/pixhawk4-mini/>

<sup>5</sup>Link to ArduCopter firmware website : <https://firmware.ardupilot.org/Copter/>

<sup>6</sup>Link to Mission Planner website : <https://ardupilot.org/planner/>

# 1 The theory of helicopters

Before entering the core of the subject, it can be useful to better understand the details of the physics allowing helicopters to fly and navigate in the air.

The word helicopter was used for the first time in 1862 by Ponton d'Amécourt[12], but the first “modern” helicopter flight was performed in 1939[12]. Helicopters have greatly evolved since then, mainly in the military field after helicopters demonstrated their potential advantages during the Vietnam and Korean wars.

Nowadays, the physics of helicopters is well-known and well documented[13, 14, 15, 16]. We give here a basic intuitive foundation of the theory of helicopters to better understand how the flight controller will control the helicopter.

It can also be noted here that Remote Controlled helicopters follow exactly the same principles as the standard manned helicopters.

Starting from the basics, for any object to be able to fly, it has to generate and control a vertical force that will compensate for its own weight and a horizontal force to move it in the desired direction. Helicopters are very flexible flying objects, as they can hover in a stable position, and move in any direction. This is represented on Figure 1.1.

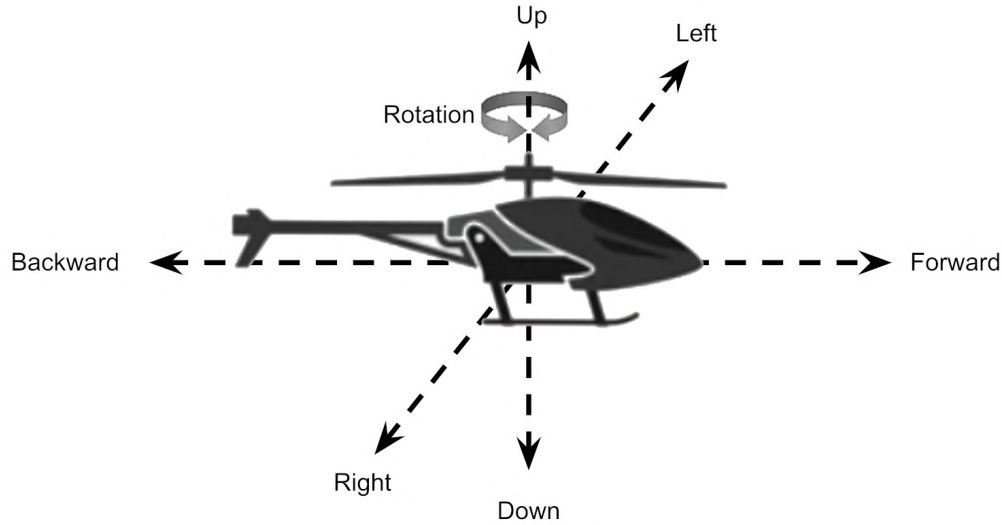


Figure 1.1: Freedom of movement of a helicopter in flight.

But how does the helicopter move in any direction? What are the forces that enter in action to obtain the desired behaviour? How can they be controlled by the pilot?

These are the questions we will address in the following. We will decompose the forces to explain them separately but also explain how they have to be combined to compensate for their side effects and obtain the desired resulting force to make the helicopter fly and navigate in the air, while keeping it stable.

## 1.1 The vertical force (the lift)

First of all, the helicopter needs a vertical force to compensate for its weight. The task of generating this vertical force is handled by a single rotor also called the main rotor. The blades attached to this rotor (usually between 2 and 4 blades but it can even go up to 8 blades, the Devil 380 has only two blades) undergo a certain relative wind speed due to their rotation.

A vertical force is generated from the interaction between the main rotor blades and this relative wind speed. This can be explained based on the action-reaction principle. A reaction force is generated when the blades push the air around the helicopter down. This situation is illustrated on Figure 1.2.

The force produced by the main rotor blades rotation is called the thrust and will be perpendicular to the rotation plane. The vertical component of the thrust is called the lift.

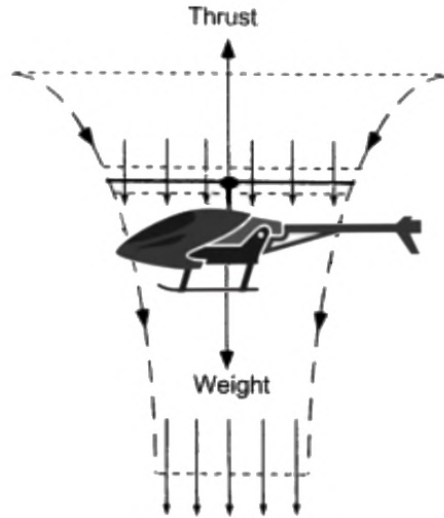


Figure 1.2: Thrust as the reaction force of the air pushed down.

At this point, we assume the rotation plane to be horizontal and hence the thrust will only be composed of the lift.

The produced thrust (and lift) is proportional to the relative airspeed squared and a coefficient depending on the horizontal asymmetry of the blade.

There are two ways to increase the thrust (and lift) :

- Increase the rotational speed of the rotor in order to increase the relative speed of the wind.
- Change the blade pitch angle in order to increase the horizontal asymmetry.

The method used for a typical multicopter drone is a fixed pitch angle of the blades and a variable rotor speed. For helicopters, it is the opposite : the rotor speed stays constant and the blade pitch angle is variable. In addition, the blade airfoil is usually symmetric, meaning that if no pitch angle is given to the blades, the thrust will be null. This position is called the zero thrust position. The pitch angle of the blades  $\theta$  is shown on Figure 1.3.

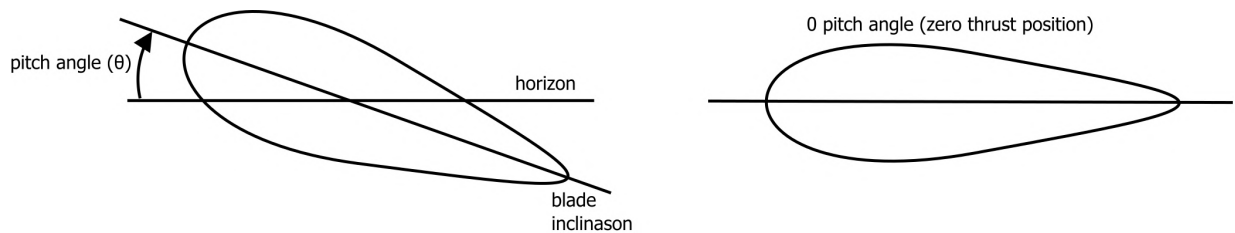


Figure 1.3: Blade pitch angle  $\theta$ .

A negative pitch angle could even be used, to go down faster. This can also be used to fly with the helicopter upside down. This is sometimes used for acrobatic helicopters (remote controlled or not).

## 1.2 The horizontal force

To make the helicopter move in a given direction, a horizontal force is required. This is handled by slightly tilting the helicopter. Indeed, as the force produced by the blades of the main rotor, called the thrust, is perpendicular to its revolutionary plane, if this plane is tilted, the thrust will have a non-zero horizontal component. The Figure 1.4 illustrates this situation.

The question is now how to tilt the helicopter.

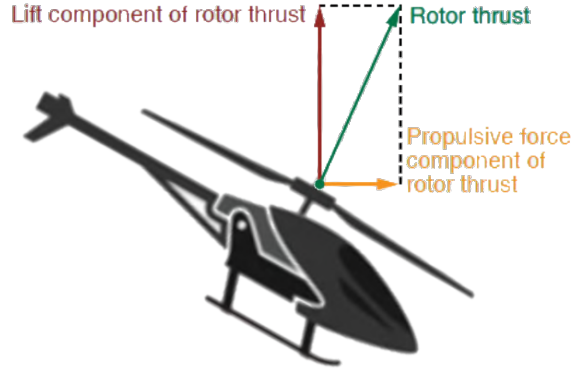


Figure 1.4: Thrust decomposition in its lift and propulsive force component.

The answer is again based on the main rotor blades pitch. We saw that the lift force (i.e. the vertical component of the thrust) can be controlled by varying the blade's pitch. The pitch angle discussed in this context, also called the collective pitch, is constant along the full revolution of the blades. Hence it induces a thrust that will be uniform along the complete revolution circle. Assuming that the plane of rotation is horizontal, the thrust will only have a vertical component (the lift). This uniform thrust (and lift) is shown on Figure 1.5.

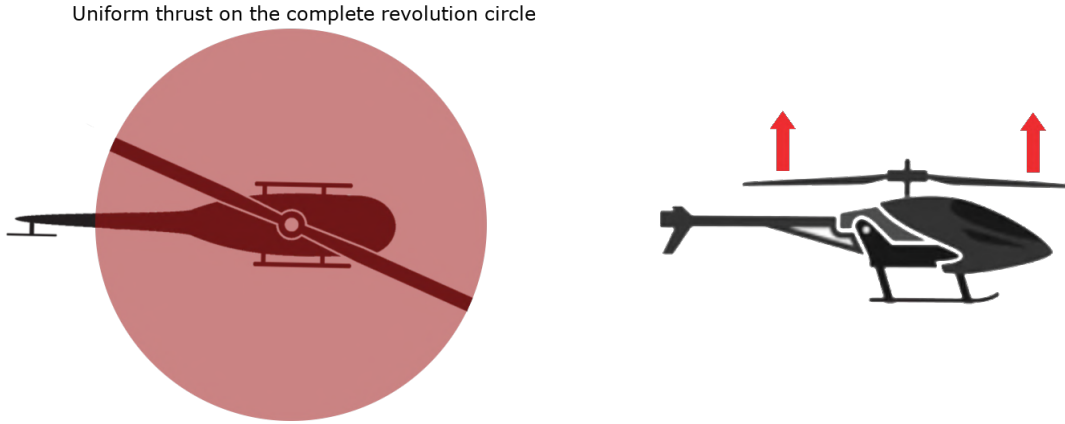


Figure 1.5: Main rotor generates a uniform lift.

In addition to the collective pitch, helicopters manage a cyclic pitch. That means the pitch of a blade will slightly vary during its circle revolution.

The pitch angle of the blades is actually a combination of the collective pitch explained earlier and this cyclic pitch. Given  $\psi$  the angular position of a blade in its revolution from the heading of the helicopter, its pitch angle  $\theta$  can be expressed as

$$\theta(\psi) = \theta_{col} + \theta_{cycl}(\psi) = \theta_{col} + \theta_s \cdot \sin(\psi) + \theta_c \cdot \cos(\psi)$$

where  $\theta_s$  and  $\theta_c$  are the two components of the cyclic pitch. The angle  $\phi$  is illustrated on Figure 1.6.

A positive  $\theta_s$  implies an increase of the pitch angle (and therefore of the thrust) on the right side of the helicopter, with its maximum at  $90^\circ$  and a decrease on the left side, with its minimum at  $180^\circ$ . The blades act like a gyroscope due to their rotation. Hence the force differential induced by the cyclic pitch will attempt to move the gyroscopic axis leading to a precession effect[14] that will deflect the action of the applied forces about  $90^\circ$  later (in the direction of rotation) from where they are applied. That means that, if the rotor turns clockwise as illustrated here, the helicopter will tilt, creating a horizontal component of thrust that will move it forward (see Figure 1.4). This situation is shown on Figure 1.7.

Similarly, a negative  $\theta_s$  will tilt and move the helicopter backward.

Following the same reasoning with  $\theta_c$  leads to a leftward or rightward tilt and move of the helicopter.

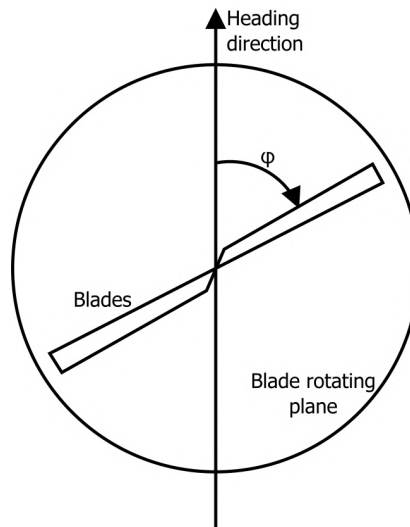


Figure 1.6: Main blade revolution angle  $\psi$ .

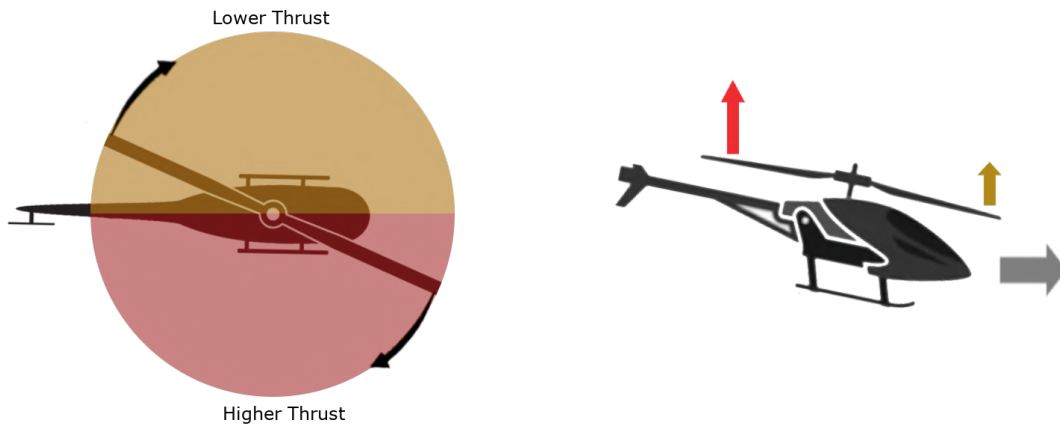


Figure 1.7: Main rotor generates a non-uniform lift.

Of course, if the rotor turns counterclockwise, these effects will be inverted.

In summary, the collective pitch and cyclic pitch will allow the helicopter to tilt, and the rotor thrust produced by the main blades will be able to compensate for its weight (by the lift component of the thrust), making it fly, and giving it a propulsive force allowing it to move in any direction.

### 1.3 The helicopter angles

As we saw in the description of the horizontal force, the helicopter angular orientation is very important in managing its moves. The schema on Figure 1.8 shows the different angles that are used to define its angular orientation and how they are named.

- The **pitch angle** is the tilt angle in the forward and backward direction.
- The **roll angle** is the tilt angle in the rightward and leftward direction.
- The **yaw angle** is the heading angle of the helicopter to a reference direction.

### 1.4 How to manage the collective and cyclic pitch?

The helicopter moves require adjusting the main rotor blades collective and cyclic pitch. This complex task is handled by a clever mechanical solution called the swash plate.



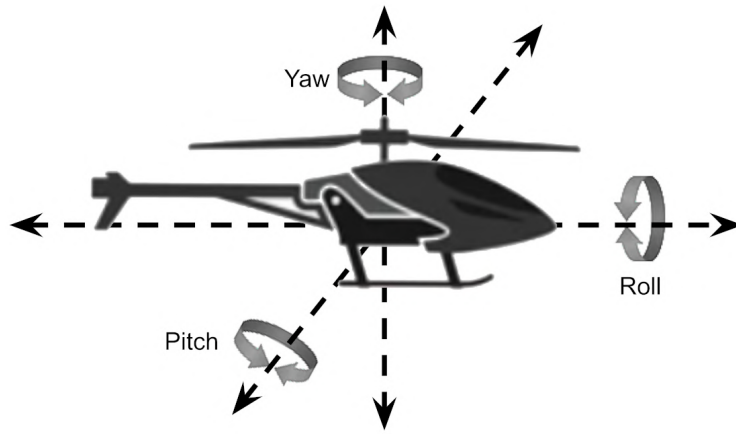


Figure 1.8: Helicopter frame angles.

The swash plate can slide up and down the main rotor shaft and this will modify the collective pitch of the blades, through the pitch links. It can also be tilted in any direction allowing to induce a cyclic pitch that will in its turn tilt the helicopter in the desired direction. The swash plate mechanism is shown on Figure 1.9.

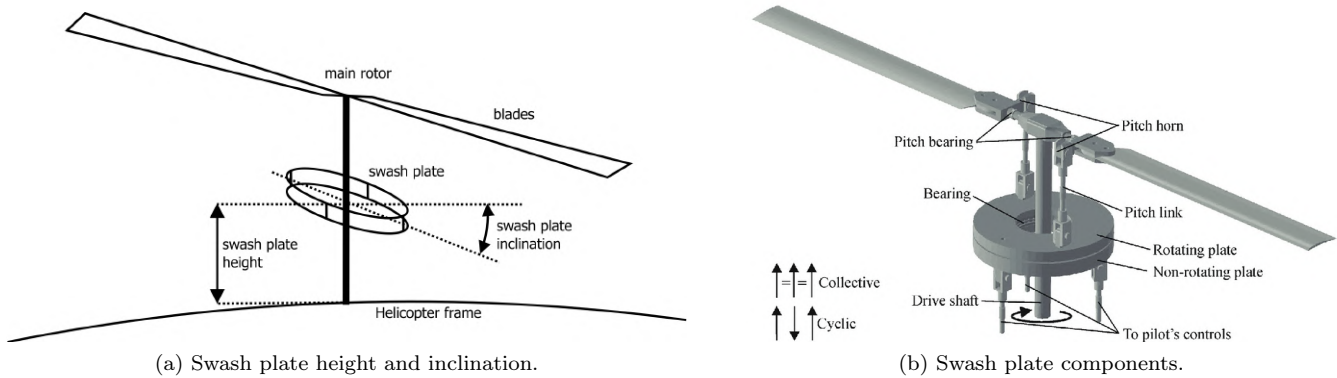


Figure 1.9: Schema of a swash plate mechanism.

The swash plate is the main mechanism used to pilot a helicopter. The Figure 1.10 shows a picture of the swash plate of the helicopter used in this master thesis, the Devil 380.



Figure 1.10: Swash plate of the Devil 380.



## 1.5 Controlling the helicopter yaw angle

In addition to the thrust, the rotation of the main rotor blades produces a torque opposed to this rotation. This torque is an undesired side effect that needs to be compensated. Without this compensation, the helicopter would begin to turn around itself.

This is where the helicopter's secondary rotor, also known as the tail rotor, comes into play. This rotor is designed to generate a horizontal thrust. This thrust times its distance to the centre of gravity of the helicopter generates another torque. This second torque value has to be matched with the torque value of the main rotor in order to compensate for it. The tail rotor thrust action is shown on Figure 1.11.

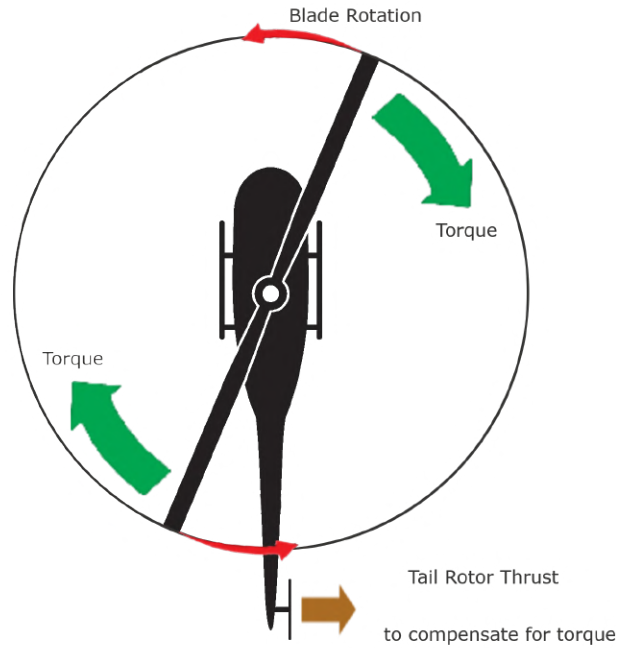


Figure 1.11: Tail rotor thrust counteracts the main rotor torque.

A greater or lower value will result in a change of the helicopter heading direction, called the yaw angle. In other words, to keep a constant yaw angle, the thrust of the tail rotor must perfectly match the torque of the main rotor.

This torque produced by the main rotor depends on the pitch angle of its blades. At a first approximation, one can consider that it only depends on the collective pitch angle, the cyclic pitch angle having only a minor influence.

Again, adjusting the thrust of the tail rotor can be done by varying its rotation speed or by adjusting the pitch of its blades. Either solution is used depending on the helicopter model.

## 1.6 Other side effect forces

To get a more complete picture of all the forces that act on a helicopter, one must also take into account other forces, such as the drag force that resists its movement through the air.

The tail rotor also induces a small translation movement if it is not compensated. It means that the helicopter should have a certain roll angle in order to have a horizontal component of the main rotor thrust of the same amplitude as the tail rotor thrust to keep the helicopter in a stable position.

In addition, the tail rotor, just as the main rotor, creates a torque that tends to induce a rotation of the helicopter. This torque can be easily compensated by tilting the swash plate a bit to create a matching opposite torque.

These 2 side effects of the tail rotor translate into a modification of the cyclic pitch of the swash plate.

## 1.7 How to move a helicopter summary

The summary of the physics principles underlined in the above sections and the resulting way of control of a helicopter flight is summarised in Table 1.1.

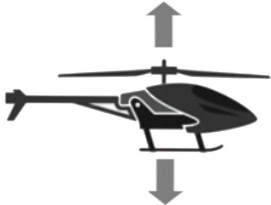
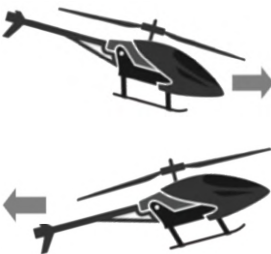
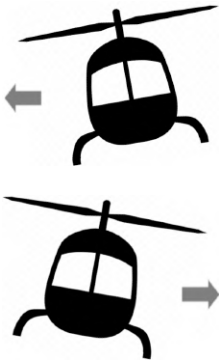
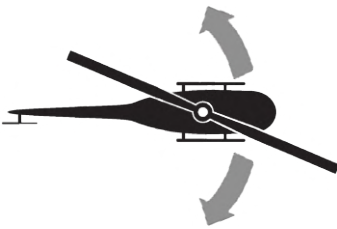
| Helicopter moves  | How does it work?   | How to control it?   |
|---|---|--|
|    | To change its <b>altitude</b> , a helicopter uses the <b>lift</b> corresponding to the vertical component of the main rotor <b>thrust</b> that compensates for its own <b>weight</b> .                            | The <b>lift</b> can be adjusted by controlling the <b>collective pitch</b> of the main rotor blades.<br><br>This is done by moving up or down the <b>swash plate</b> of the main rotor.                |
|    | Helicopter moves <b>forward</b> or <b>backward</b> by tilting around its <b>pitch</b> angle.<br><br>That creates a horizontal component of the main rotor <b>thrust</b> making it move forward or backward.       | Helicopter <b>pitch</b> angle can be adjusted by controlling the <b>cyclic pitch</b> of the main rotor blades ( $\theta_s$ ).<br><br>This is done by tilting the <b>swash plate</b> of the main rotor. |
|  | Helicopter moves <b>leftward</b> or <b>rightward</b> by tilting around its <b>roll</b> angle.<br><br>That creates a horizontal component of the main rotor <b>thrust</b> making it move leftward or rightward.    | Helicopter <b>roll</b> angle can be adjusted by controlling the cyclic pitch of the main rotor blades ( $\theta_c$ ).<br><br>This is done by tilting the <b>swash plate</b> of the main rotor.         |
|  | Helicopter can rotate around its <b>yaw</b> angle.<br><br>This rotation comes from the <b>thrust</b> produced by its <b>tail rotor</b> that compensates for the <b>torque</b> produced by its <b>main rotor</b> . | The <b>thrust</b> of the <b>tail rotor</b> can be adjusted by controlling the (collective) <b>pitch</b> of the tail rotor blades or by controlling the tail rotor blade rotation speed.                |

Table 1.1: Summary of the helicopter physics and how to control it.

As we can see from this high-level overview of the basic principles of helicopter physics, there are many forces implied and they have side effects that need to be compensated. The resulting system is quite complex and strongly coupled, and this tends to make helicopters quite hard to control by a human pilot without assistance.

Using a flight controller helps the pilot by providing assistance to manage this complex system. It allows the pilot to act on one component of the system without having to care about all possible coupling with the other components.

We will discuss in the next sections how such a flight controller works, from a hardware perspective and then from a software perspective.

## 2 Introduction to the flight controller

When a pilot controls an RC helicopter without a flight controller, it will basically send direct commands to the RC helicopter servo motors controlling the swash plate and the tail rotor such as illustrated in the schema on Figure 2.1.

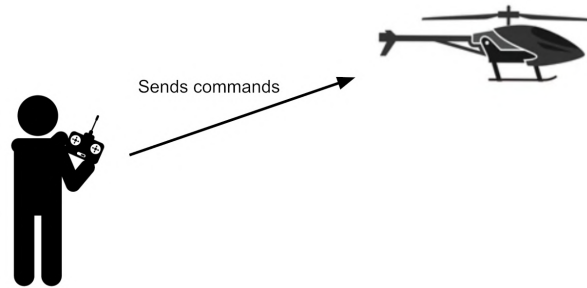


Figure 2.1: Pilot sends commands to the helicopter.

In this situation, the pilot has to actually manage the stability of the helicopter by himself, which makes it quite difficult to control.

This creates a kind of continuous feedback loop, where the pilot plays a very important role :

- The pilot captures information about the helicopter state, such as its position or its velocity.
- He can then interpret this information and combine it with the behaviour he wants to induce to the helicopter, such as the direction he wants it to go, and decide which commands to send to it.
- Then the helicopter reacts to the commands it received (and also to external perturbations like the wind) and changes its state.

This loop continues, with the pilot adjusting the commands based on the reactions of the helicopter as illustrated in the schema on Figure 2.2.

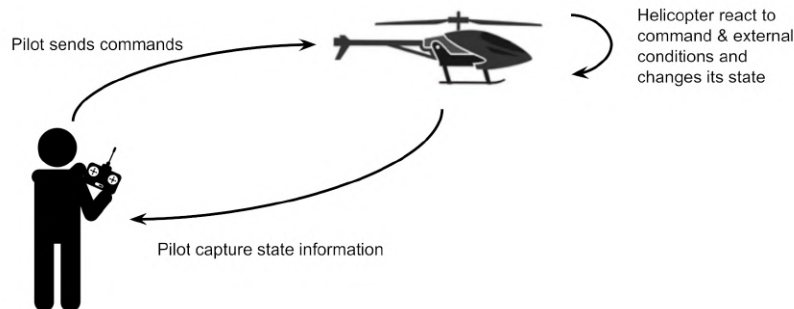


Figure 2.2: Pilot sends commands to the helicopter and captures information.

In this situation, keeping the helicopter on a fixed position can be quite hard and requires an experienced pilot, capable of constantly adapting the commands sent to compensate for any external perturbation and to correct any exaggerated movement.

With the addition of a flight controller[11, 17], the flight controller will act as a second pilot, taking over many aspects of the flight, and greatly easing the task of the real pilot that will just need to send instructions to let the flight controller know how the helicopter should behave (direction, speed, etc.). This situation is illustrated in the schema on Figure 2.3.

The flight controller creates a second feedback loop that interprets information coming from sensors, combines it with the pilot instructions to deduce the commands to be sent to the motors. It automatically takes care of the helicopter stability, taking into account external perturbations and hence greatly reducing the complexity managed by the real pilot.

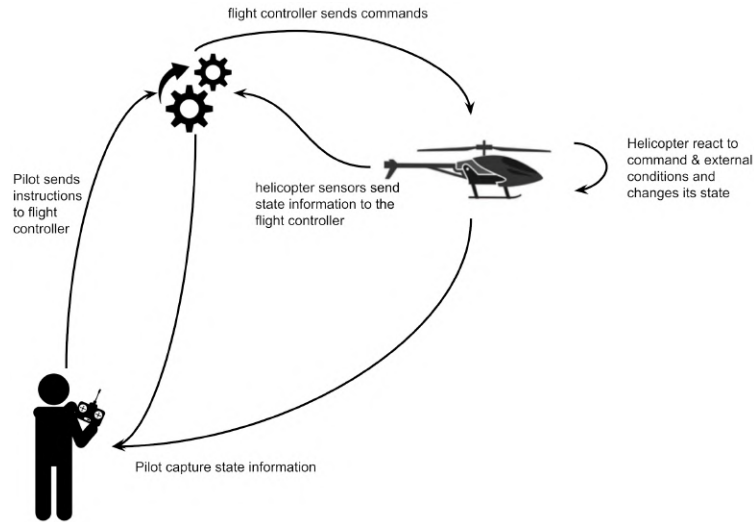


Figure 2.3: Pilot sends command to the helicopter via the flight controller.

In this situation, the helicopter is able to hover in a fixed position without any action from the pilot.

Also note that the flight controller can provide additional state information and flight metrics to the pilot.

The flight controller can be parameterised to give different levels of assistance to the pilot. These flight modes will be detailed in the software part, let's just mention here that this assistance can even go further to implement a so-called automated flight. The pilot can predefine a path composed of different waypoints, send it to the flight controller, and let it autopilot the helicopter to follow the predefined path without any further interaction from the pilot. This situation is illustrated in the schema on Figure 2.4.

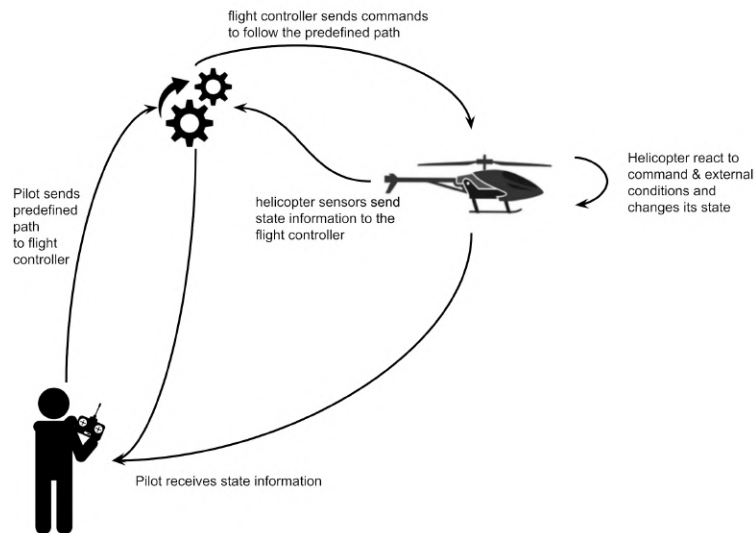


Figure 2.4: Flight controller pilots the helicopter in an automated flight.

## 3 Hardware description

In this section, we will describe the different hardware parts of the flight controller-based system, namely the sensors, actuators, pilot communication and the flight controller board that links them together.

We will discuss the different alternatives for each of these parts and describe the ones used in the context of this master thesis.

### 3.1 Sensors

The sensors can be considered as the eyes of the flight controller. The whole helicopter state should be deduced from these sensors.

The helicopter state is at least composed of :

- Its position in a reference frame (for example, its take-off position).
- Its velocity.
- Its angular orientation, also called attitude.
- Its angular speed, also called angular rate.

*Note that all these measures are given along 3 dimensions.*

All this information can be derived from an IMU (Inertial Measurement Unit). Such a unit is composed of a 3-axis accelerometer and a 3-axis gyroscope returning respectively accelerations and angular rates.

The helicopter angular orientation can be found by computing the integral of the measured angular rate. Similarly, the velocity and the position can be found by integrating the measured acceleration.

However, these integral computations are error-prone and could amplify the measurement errors. To minimise this risk, 2 solutions are implemented in our context :

- The first one is to allow a sensor redundancy that reduces the measurement incertitude.
- The second one is to add different sensors measuring different complementary data, such as a GPS position.

These solutions will be further discussed in section 4.4.1 **Current state management**.

The sensors used in this master thesis are included in different hardware components. These components and their sensors are listed in Table 3.1.

Other sensors might have been used, such as lidar<sup>7</sup> or optical flow sensors, respectively to measure the altitude or ground velocity.

### 3.2 Actuators

As seen in section 1 **The theory of helicopters**, the flight of an RC helicopter is influenced by 3 elements :

- The main rotor speed.
- The swash plate position inducing the main rotor blades collective and cyclic pitch.
- The tail rotor thrust.

However, there are many different ways to control them, depending on the RC helicopter model :

- **The main rotor speed** : The main rotor is set in motion by a motor, which is usually driven by an ESC (Electrical Speed Controller) adapting the power transferred to the motor to keep its rotational speed constant regardless of the torque generated. The rotational speed targeted by the ESC can be modified thanks to a PWM (Pulse Width Modulation) signal. As explained in section 1.1 **The vertical force (the lift)**, RC helicopters use a constant rotor speed in flight. Therefore, if the RC helicopter uses an ESC, there is no need to adapt the signal sent to it during the flight. But if this is not the case, it would require sending an adaptive signal to control the main rotor speed (to maintain it constant).

---

<sup>7</sup>Lidar is an acronym of "laser imaging, detection, and ranging" it measures distance by targeting an object with a laser and measuring the time for the reflected beam to come back.

|  |  |
|--|--|
|   | <p>The flight controller Pixhawk 4 mini module[2] contains the 2 IMUs that will be used in our solution.</p> <p>It also includes a 3-axis magnetometer to measure angular orientation and a barometer to measure altitude, but they will not be used :</p> <ul style="list-style-type: none"> <li>• The magnetometer will not be used because of magnetic field disruptions produced by the carbon fibre in the helicopter frame.</li> <li>• The barometer will not be used because of the air depression produced by the main rotor blades rotation.</li> </ul> |
|   | <p>A GPS module[2] has been added and connected to the flight controller.</p> <p>It contains :</p> <ul style="list-style-type: none"> <li>• A GPS returning the position in 3 dimensions.</li> <li>• A 3-axis magnetometer returning the angular orientation of the helicopter frame.</li> </ul>   |
|  | <p>The power board also contains two sensors, a voltmeter and ammeter that monitor the battery for safety reasons.</p> <p>These sensors are connected to the flight controller that uses them to send alerts to the ground control station for pilot communication purposes.</p> <p>However, the control of the helicopter in flight is not influenced by these sensors.</p>   |

Table 3.1: List of hardware devices containing sensors and their respective sensors.

- **The swash plate position** : The swash plate can only be tilted and slide along the main rotor shaft. Two servo motors would be sufficient to move it as required and keep it in place. However, this configuration cannot withstand strong constraints. This explains why only small toy RC helicopters use only two servo motors. The 3 servo motors configuration is way more popular and exists in different positioning. A 4 servo motors configuration also exists for bigger RC helicopters and also exists in different positioning.
- **The tail rotor thrust** : Some helicopters use the main motor to set the tail rotor in motion. In that configuration, a servo motor controls the pitch of the tail blades to modify the tail rotor thrust. Other helicopters might have tail blades with fixed pitch and rely on a separated motor to have a variable rotational speed. A third category of helicopters have a separated tail motor to ease the task of the main motor and still rely on a servo motor to modify the tail blade pitch angle.

In the case of the Devil 380, these 3 elements are controlled as explained in Table 3.2.

To summarise, the Devil 380 helicopter flight can be controlled by the activation of 5 motors :

- 1 single main motor inducing a rotational speed to the main rotor and the tail rotor.
- 3 servo motors controlling the swash plate position.


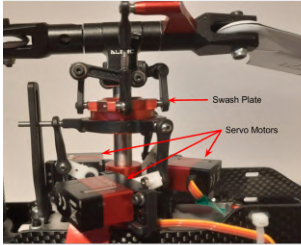
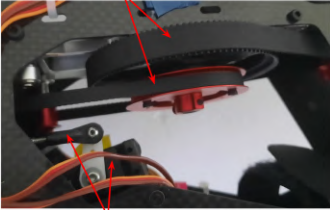

|  |   |
|--|---|
|   | <p><b>The main rotor speed :</b> The Devil 380 uses a constant main rotor speed controlled by the ESC shown on this picture.</p> <p>Hence, there is no need to adapt the ESC signal during the flight.</p>  |
|   | <p><b>The swash plate position :</b> The Devil 380 uses a 3 servo motors configuration, 120° apart from each other and with one of them aligned with the helicopter head, as shown on Figure 3.1.</p>   |
| <p>tail motor drive gears transmitting the main rotor speed times 2</p>  <p>Servo motor actioning a control rod transmitting the pitch position to the tail pitch change mechanism</p> <p>Tail pitch change mechanism</p>  <p>Control rod</p> | <p><b>The tail rotor thrust :</b> The tail rotor of the Devil 380 is set in motion by the same motor as the main rotor. The tail rotor speed is, however, 2 times bigger due to different gear sizes.</p> <p>As the main rotor has a constant speed, this is also the case for the tail rotor and the tail rotor thrust is only controlled by the servo motor controlling the pitch of the tail blades.</p> |

Table 3.2: Actuators on the Devil 380.

- 1 servo motor controlling the tail rotor thrust.

These 5 motors are each controlled by a simple PWM signal.

### 3.3 Pilot's communication

Communication with the pilot is very important. Indeed, the pilot has to send instructions to the helicopter in order to control it, but he might also need to get precise data about the helicopter state. This second communication purpose is called telemetry.

On the ground, the pilot uses a remote control to send its instructions but such a device does not offer a good user interface to display rich information about the current flight. A ground control station application, running on a computer for example, offers a better display and can even allow parameter modifications of the flight controller if needed.



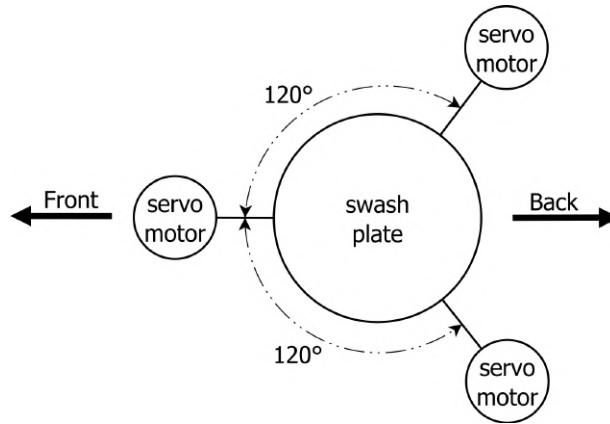


Figure 3.1: Schema of the swash plate configuration of the Devil 380.

Again there can be various solutions here. Communication between the remote control and the helicopter is done through an RC receiver. This one could also manage the telemetry. If this is not the case, another component could send telemetry directly to the remote control. Another kind of component could send telemetry to a specific receiver on the ground connected to a ground control station.

We have used a FRSky XSR<sup>8</sup> RC receiver for the communication between the remote control and the helicopter. This module does not handle telemetry.



The Pixhawk 4 mini only supports a single telemetry signal (in addition to the RC receiver connection). We chose to connect this telemetry signal to the ground control station, using a SIK Telemetry Module as shown on Figure 3.2.

### Ground Control Station



### SIK Telemetry modules

Figure 3.2: Telemetry modules connecting the flight controller to the ground control station.

## 3.4 Flight controller board

Most of the flight controller boards include different sensors directly, such as IMUs and magnetometers. There are many different boards on the market. They differ on many different aspects :

- Computational power.

<sup>8</sup>Link to the FRSky XSR manufacturer website : <https://www.frsky-rc.com/product/xsr/>

- Memory.
- Embarked sensors (and their quality).
- Vibration damping solution for the IMUs.
- Price.
- Weight and size.

The one used in our context is the Pixhawk 4 mini. It provides good capabilities, and computational power for a reasonable price. Its weight and size allow using it on relatively small drones.

It has 2 IMUs, a magnetometer and a barometer embarked on the board. It also has additional input/output ports to connect additional Sensors (such as a GPS module), to control the actuators through PWM output signals, and to manage the Pilot communication (remote control and ground control station).

All the resulting communications are shown on Figure 3.3. This figure ignores the fact that some sensors could be directly embedded on the flight controller board, as this will involve communication between the sensors and the CPU anyway.

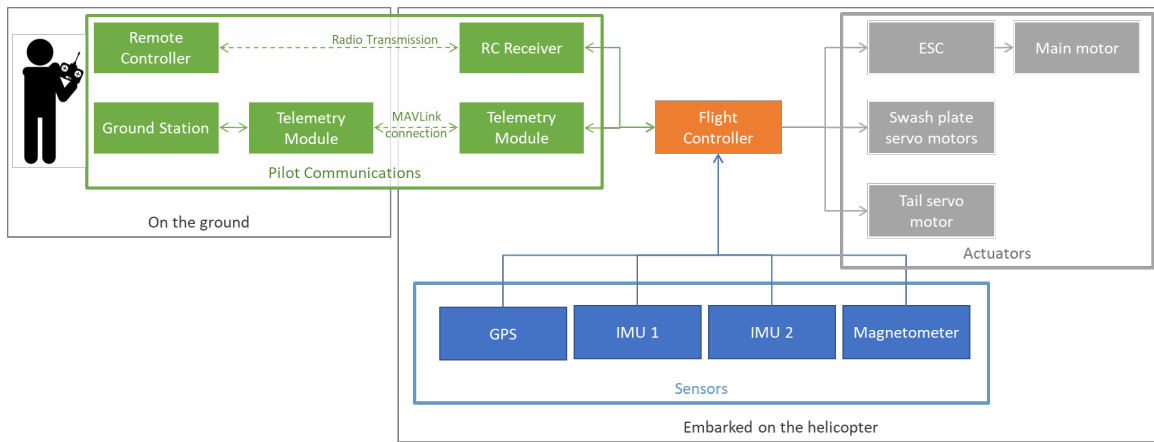


Figure 3.3: Flight controller ecosystem overview.

More information about the flight controller board and the hardware in general are given in Appendix A **More advanced description of the flight controller board and its ecosystem.**

## 4 Software description

As explained in section 2 **Introduction to the flight controller**, the main objective of the flight controller, as its name indicates, is to control the helicopter during the flight. For doing that, it receives information from the sensors and the pilot, does some complex computation with them and then returns PWM signals to the different motors. These interactions are schematised on Figure 4.1. In this section, the focus will be set on the complex computations it does between its input and output signals.

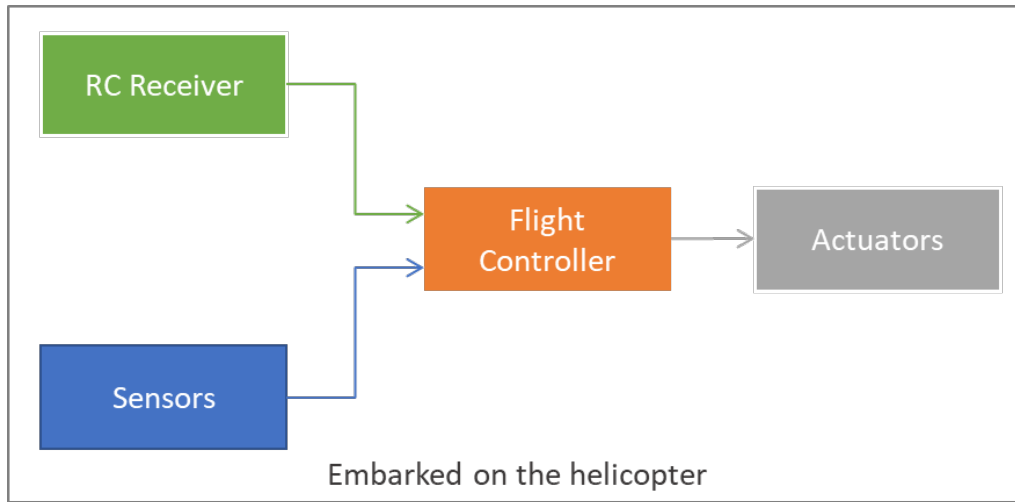


Figure 4.1: Basic overview of the flight controller inputs and outputs.

The software solution we have used to control the Devil 380 helicopter is the one provided by ArduPilot. ArduPilot is an open source software supporting many unmanned vehicle types, such as multicopters and RC helicopters, but also other kinds of unmanned vehicle systems : fixed-wing aircraft, boats, submarines, etc.

ArduPilot provides different firmwares for the different families of unmanned vehicles. For the Devil 380, we have used ArduCopter which supports multicopters and RC helicopters.

The version used is 4.1.5, the latest stable release at the time we made our experimentations. This is quite a mature solution as the ArduPilot project’s earliest roots date back to 2008 when Jordi Munoz wrote an Arduino program (which he called “ArduCopter”) to stabilise an RC Helicopter[18]. This solution later on became open source and greatly evolved during the years, thanks to a large community of developers and users.

ArduCopter is still constantly evolving, as there is already a new version released at the time of writing, and it is well documented. It supports many flight controller boards, the Pixhawk 4 mini being one of them. The specific documentation on the helicopter parameterisation is quite exhaustive.

Other alternatives exist on the market. Its main competitor is the PX4 software, another open source solution. PX4 and ArduPilot are very similar. However it seems that ArduPilot provides more sophisticated algorithms with more advanced parameterisation, allowing better flight behaviour. Moreover, PX4 for RC helicopters has relatively poor documentation and is still in early stages.

### 4.1 Software overview

As the software is a generic software able to pilot many different multicopter or RC helicopter drones, it has to support many different sensors, RC receivers or actuators. The core algorithm should be decoupled from these technical specificities to be easily adapted to any hardware.

In addition, most sensors have to be calibrated, because they are never perfectly positioned or oriented, and their measures must be corrected before being used.

For that reason, there will be some components to transform the raw sensor data into standardised data that can be used by the core algorithm and to transform the output commands into specific PWM signals. These transforma-

tion steps are based on parameters that need to be defined beforehand.

The overall software structure is depicted on Figure 4.2.

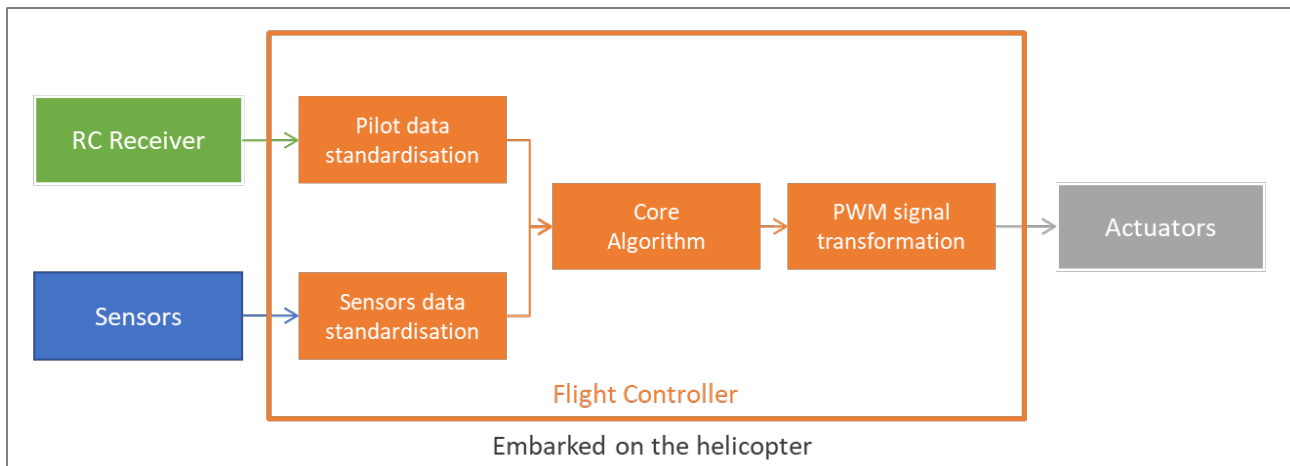


Figure 4.2: High-level decomposition of the flight controller task.

One could think of it as if the core algorithm is actually managing a kind of “virtual” helicopter, only using standard sensors perfectly aligned, and standardised pilot instructions. The core algorithm will analyse all the data it receives and deduce commands to be applied to the virtual helicopter. The mapping between a real helicopter and the virtual one is done by the other components, translating and correcting the real data collected by sensors or translating the virtual commands to real ones adapted to the actual helicopter model. These mapping operations are based on a large number of parameters.

ArduPilot is therefore a highly configurable piece of software, with hundreds of parameters.

The parameterisation does not only cover the sensors and actuators, it also covers the core algorithm itself, allowing to define different flight modes or to fine-tune different aspects.

Most of the parameters have to be defined beforehand, and will not change during the flight. Some others might, however, be changed during the flight, like the chosen flight mode, for example.

All the parameters of the flight controller are accessible via the ground control station. The ground control station used in this master thesis is Mission Planner<sup>9</sup> which acts as a user interface for managing ArduPilot parameters and calibration data. We used the Windows version, but it could also run on Linux (using MONO<sup>10</sup>) and a version for Android OS is currently under development.

In the next sections, each of the software components will be described in terms of its role and behaviour during the flight control and in terms of parameterisation.

## 4.2 Sensors data standardisation

### 4.2.1 Role and behaviour

The sensors data standardisation step is in charge of collecting the configured sensors raw data and transforming it into data that can be used by the core algorithm.

The data transformation applied will make it agnostic of the specific sensor used and will correct the measurements by attempting to convert them to measurements that a perfectly aligned sensor would have measured.

These computations can be more or less complex, depending on the sensor. It often requires mathematical operations such as scaling, offset removal or change of basis. We will not dive into the implementation details here.

<sup>9</sup>Link to the download page of Mission Planner : <https://ardupilot.org/planner/docs/mission-planner-installation.html>

<sup>10</sup>More info about MONO : <https://www.mono-project.com/>

### 4.2.2 Parameterisation

Many parameters must be defined to specify the sensors that will be used by the core algorithm and to provide calibration data. These parameters need to be defined beforehand.

The parameterisation can be done manually via the ground control station. However, defining all required parameters by hand could quickly become a tedious task. Hopefully, the ground control station also provides access to different calibration protocols that will simplify the definition of some of these parameters.

Here are a few important examples of sensor parameters and calibration that can be defined :

- **IMU** (embedded on the Pixhawk 4 mini) :
  - **Accelerometer** : It needs to be calibrated, as it should ideally be perfectly aligned with the helicopter main axis. A calibration phase is manually initiated on ArduPilot. Additional parameters can be fine-tuned by hand to correct the accelerometer measures if it is far from the helicopter centre of gravity.
  - **Gyroscope** : It will automatically be calibrated on each boot of the flight controller.
- **GPS** : The GPS does not require any calibration. Some specific parameters can be defined based on the GPS, such as the minimum number of satellites to allow any flight requiring the GPS position.
- **Magnetometer** : It requires a complex calibration phase. This calibration tries to detect any interference with the earth magnetic field that could be induced by other components of the RC helicopter. In some cases, the calibration can fail due to such interferences. This was the case in our configuration with the magnetometer embedded in the Pixhawk 4 mini due to its location between two carbon fibre plates. That's why we only use the magnetometer included in the GPS module and why we placed it as far as possible from the main motor, servo motors and the carbon fibre structure.
- **Battery Monitor** : Battery monitoring does not influence the flight control and is only used for alerting the pilot, through telemetry. Its parameterisation just requires specifying the power board type used. ArduPilot will deduce all the related parameter values.
- **Others** :
  - For some sensors there is a parameter to define the orientation of the sensor. We used this parameter to indicate that the Pixhawk 4 mini and GPS module are mounted backward.
  - The “level” calibration allows detecting the Pixhawk 4 mini tilt angle when the main rotor axis is perfectly vertical.

More detailed information about the sensors calibration and parameters is available in Appendix B **Sensors parameters and calibration**.

## 4.3 Pilot data standardisation

### 4.3.1 Role and behaviour

The pilot data used to control the helicopter during the flight comes from the remote control and is transmitted to the flight controller through the RC receiver.

The data transferred by the RC receiver are interpreted as a series of PWM signals, each of them corresponding to an RC channel.

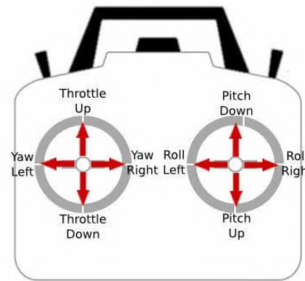
The flight controller can handle up to 16 RC channels and at least 6 are mandatory to flight an RC helicopter :

- 4 channels are needed for the transmission of the remote control stick positions.
- 1 channel is used for selecting the flight mode (flight modes will be further discussed in section 4.4.2 **Next state interpolation**).
- The last mandatory channel is used by the motor interlock.

Note that these 6 channels are required by the ArduPilot software. It will not allow the helicopter to fly without them.

The first 4 channels are the ones that will be used to control the helicopter movements during the flight. They are used to transmit the position of the two joysticks present on the remote control.

- The throttle controls the main rotor blades collective pitch, allowing the helicopter to go up or down.
- The yaw controls the helicopter yaw angle, allowing it to pivot its head left or right.
- The pitch controls the helicopter pitch angle, allowing it to move forward or backward.
- The roll controls the helicopter roll angle, allowing it to move leftward or rightward.



*Note that the exact effect on the helicopter will also depend on the flight mode, as it will be explained later on.*

The motor interlock is a safety button on the remote control. It acts as an on/off button on the main motor of the helicopter when this one is armed, i.e. when it is ready for take-off, after a series of safety checks.

To summarise, the pilot data standardisation step aims at receiving the PWM signals through the RC receiver channels and transforming them to different usable data given to the core algorithm :

- The throttle, yaw, pitch and roll input signals are translated into numeric values as a percentage representing the corresponding remote control stick position.
- The flight mode signal is translated to a numeric value representing the selected flight mode.
- The motor interlock signal is translated into a numeric value basically representing the ON/OFF position.

#### 4.3.2 Parameterisation

To arm the helicopter, a series of safety checks should be validated such as the battery level or the number of satellites connected to the GPS. A parameter can be used to enable or disable some of these safety checks. Note that some safety checks cannot be disabled, this is the case for the motor interlock that always needs to be off before arming the helicopter.

In other words, turning on the main motor and setting the rotors in motion needs a 2 phases protocol : arming the helicopter with the main motor disabled and then, after all safety checks have been successfully validated, starting the main motor.

The remote control must be calibrated, using the ground control station. This calibration consists in defining the number of RC channels that will be used by the pilot as well as the minimum and maximum PWM pulse width for each of them.

In addition to this calibration, a parameter can be set for each RC channel to reverse the input signal. This means that when the signal reaches its maximum pulse width, the flight controller will interpret this as a minimum value and when it reaches its minimum pulse width, the flight controller will interpret this as a maximum value.

Once the channels have been calibrated, they can be assigned their role through a set of parameters.

Also note here that the telemetry modules need a binding between the embarked one and the one that stays on the ground. This corresponds to a basic calibration phase. We do not give more detail about telemetry here as we focus on the flight control part of the software.

More detailed information about the calibration and parameters linked to the pilot data standardisation is available in [Appendix C Pilot data standardisation](#).

## 4.4 Core algorithm

The core algorithm is the most complex part and can be decomposed in three main components as illustrated on Figure 4.3.

- **Current State Management** : to keep track of the current helicopter state.
- **Next State Interpolation** : to derive its desired next state.
- **Main Control System** : to control the helicopter to move it towards the desired state.

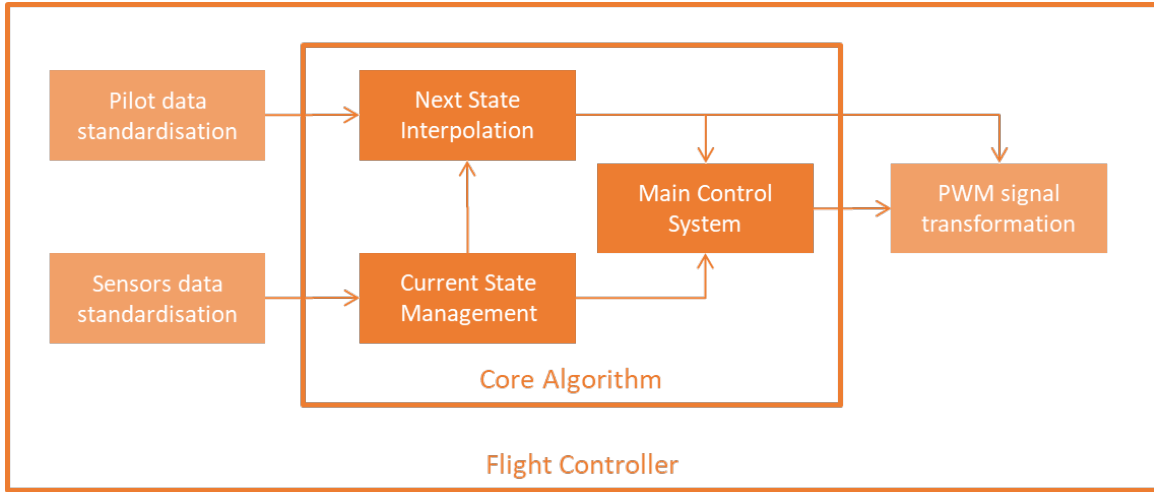


Figure 4.3: Core Algorithm decomposition.

Each of these roles is described in more detail hereafter.

### 4.4.1 Current state management

#### 4.4.1.1 Role and behaviour

In order to control the flight, the flight controller must know the current state of the helicopter at every moment. However, this essential task is not an easy one! This is the role of the current state management component, which outputs a detailed state composed of many data. Here are a few important ones :

- GPS position.
- Relative position to the take-off.
- Speed.
- Angular orientation also called attitude.
- Angular speed also called angular rate.

For doing that it uses an Extended Kalman Filter[19] (EKF) able to deduce the current state of the helicopter based on its knowledge of the previous state and data coming from sensors.

It is also able to give a confidence score evaluating how confident it is on its current state evaluation.

As explained in section 3.1 **Sensors**, sensors are error-prone and their measures can be improved by using multiple sensors of the same type or different sensors measuring complementary data. The advantage of using Kalman filter is that it can merge data coming from different sensors or create multiple filter instances and select the one with the best confidence score.

The Kalman filter evaluates the whole helicopter state and some useful data such as different sensor biases or wind speed. The current state management is illustrated on Figure 4.4.

From version to version, the available Kalman filter handles more and more features. As it may be quite resource consuming, previous versions are still available. We used, however, the last released version (i.e. EKF3) and did not encounter any problem with it. A 4<sup>th</sup> version is currently under development.

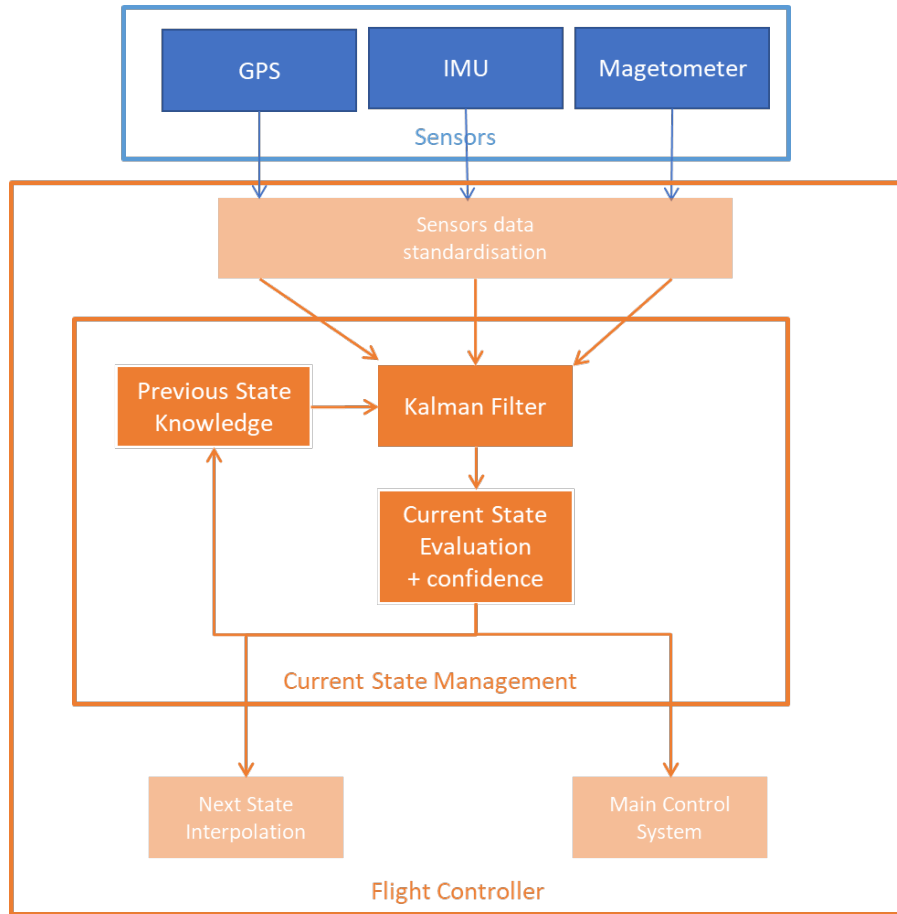


Figure 4.4: Current state management process using a Kalman filter.

#### 4.4.1.2 Parameterisation

The EKF3 has a lot of parameters to define its data sources and to customise the update weights of different inputs. We mostly used the default parameters. Only the source of altitude measurement had to be changed in order to use the GPS only and ignore the barometer data. Indeed, as the barometer lies in the main rotor wind, its altitude estimate suffers from a variable offset and a lot of noise.

More detailed information about the current state management parameters is available in Appendix D **Current state management parameters**.

### 4.4.2 Next state interpolation

#### 4.4.2.1 Role and behaviour

This component is in charge of computing the target expected state. For that it will use the current state, coming from the current state management and the pilot instructions coming from the pilot data standardisation.

In addition to these inputs, it also strongly relies on the current flight mode. The flight modes define different interpretations of the pilot instructions and different levels of autonomy<sup>11</sup> of the RC helicopter to find a desired pitch, roll and yaw angles as well as a throttle command.

The desired pitch, roll and yaw form the desired attitude of the RC helicopter. This attitude is transmitted to the control system. The throttle command, on the other hand, is directly transmitted to the PWM signal transformation component to be interpreted and applied to the real helicopter.

The next state interpolation is pictured on Figure 4.5.

<sup>11</sup>The autonomy, here, refers to the number of things that the drone can do independently of the pilot instructions.



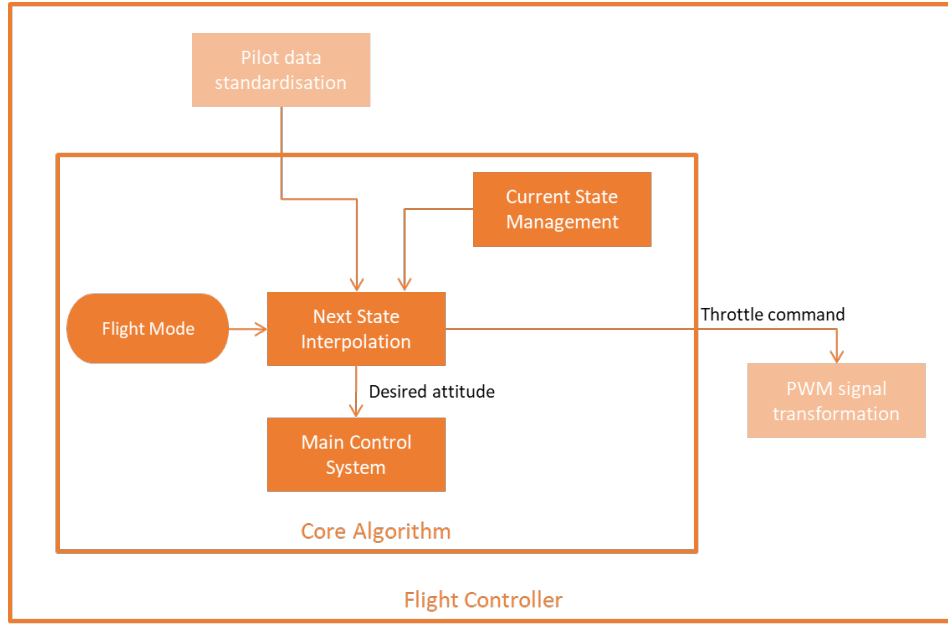


Figure 4.5: Next state interpolation interactions.

This section underlines a strong advantage of the flight controller. Once the flight controller has been set to fit the helicopter used, a large variety of flight modes is available without any other requirements.

The flight modes tested in the context of this master thesis are :

- **Stabilised** : The desired attitude and the throttle value are directly derived from the pilot instructions<sup>12</sup>. This is the flight mode with the least possible help of the flight controller as it relies only on the helicopter state and the control system. However, the flight controller (or more specifically the control system) helps the pilot by compensating external perturbations such as the wind impact on the helicopter attitude, this is why this mode is called stabilised.
- **AltHold** : It has the same behaviour as the stabilised flight mode except for the throttle value. This one follows a specific control loop that maintains the helicopter at a fixed altitude if the throttle stick is centred. The descent and climb of the helicopter are controlled in speed based on the throttle stick position.
- **PosHold** : It has the same behaviour as in the AltHold flight mode except for the pitch and roll angles. These angles are still controlled by the pilot based on the pitch/roll stick position. However, a correction on the pilot instruction is added by the flight controller. This correction is such that the helicopter keeps a fixed position when the pitch/roll stick is centred, while in AltHold or Stabilised modes it would drift with the wind.
- **Loiter** : It has the same behaviour as in the AltHold flight mode except for the pitch and roll angles. The pitch/roll stick position is now translated to a desired speed and the desired pitch and roll angle are set in order to track this speed. This mode is very similar to PosHold, as the speed and the corresponding angle are strongly correlated, but it leads to a slightly different behaviour. For example, with the PosHold mode, if the wind is varying, the helicopter will keep the angle determined by the stick position, leading to a change of speed. While in the Loiter mode, in the same windy conditions, it will keep the speed determined by the stick position, automatically adapting its angular orientation.
- **Auto** : In this flight mode, the flight controller has full control over the helicopter. The pilot gives it a mission that consists of a list of instructions such as a path composed of different waypoints to reach and the flight controller leads the helicopter through this mission. This flight mode uses the loiter flight mode, with a desired speed vector directed to the next waypoint.

Several other flight modes exist, such as a land or an acrobatic flight mode, but their uses fall out of the scope of this master thesis.

<sup>12</sup>The yaw is controlled in a slightly different way as the stick position corresponds to a delta to be applied to the yaw angle, why for the pitch and roll, the stick position indicates the corresponding absolute angular value.

#### 4.4.2.2 Parameterisation

The main parameters used in this context are related to the flight mode.

The flight modes can be changed in flight. For this purpose, a channel of the remote control allows the pilot to select the flight mode out of 6 predefined flight modes.

These predefined flight modes need to be parameterised beforehand. The flight modes can be ordered by the degree of autonomy they provide with each relying on the previous one, and adding a supplementary level of autonomy.

The Stabilised flight mode is the most basic one, but it already provides the basic stabilisation features, coming from the main control system, that all other modes will use. Then the AltHold flight mode uses the Stabilised one and adds a level of autonomy by controlling the throttle command. Then the PosHold and Loiter flight modes both use the AltHold one and add another level of autonomy, this time on the pitch and roll angle regulations. Finally, the Auto flight mode takes the loiter one and adds a final level of autonomy reaching a fully autonomous flight mode.

Each flight mode comes with its own set of parameters. However, most of these parameters only represent specific limitations that can be set for safety reasons. The flight mode parameters related to their behaviour itself have good default values and it is recommended to leave them as such.

More detailed information about the current state management parameters is available in Appendix E **Next state interpolation parameters**.

#### 4.4.3 Main control system

##### 4.4.3.1 Role and behaviour

The control system is at the core of the flight controller. This is the main reason why it is used. The control system of ArduPilot controls the helicopter based on its angular orientation (also called attitude). This means that given a desired pitch, roll and yaw angles, and a current state (including the current pitch roll and yaw angles and angular speeds) it will deduce commands to send to the helicopter for it to reach this attitude and to keep it. The main control system interactions are shown on Figure 4.6.

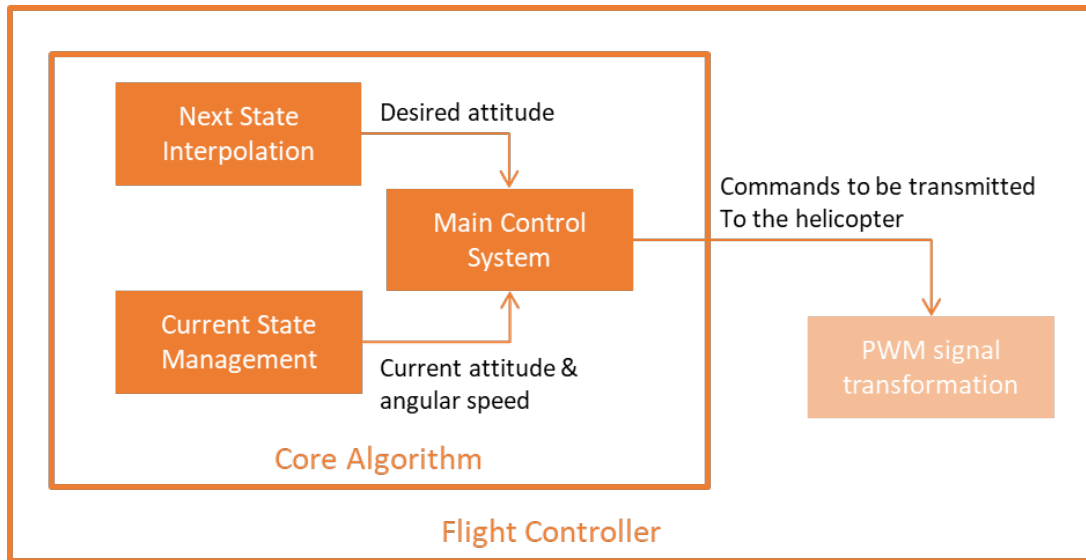


Figure 4.6: Main control system interactions.

An intuitive explanation of the control system behaviour is given in Table 4.1. It is composed of three main steps, and the pitch angle has been taken as an example.

The output of the control system is represented as a command that can be sent to the helicopter (via the PWM signal transformation we will discuss afterward) to move the swash plate (for the pitch and roll angle) or the tail rotor pitch change mechanism (for the yaw angle).

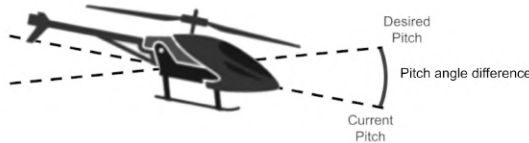


|   |  |
|---|--|
|  <p>Desired Pitch<br/>Pitch angle difference<br/>Current Pitch</p> | <p>The first step consists in comparing the current and desired angles, deducing a difference that needs to be corrected.</p>  |
|  <p>Desired Angular Speed<br/>Current Angular Speed</p>            | <p>From this difference of angles to be corrected, the control system will estimate an angular speed that should be used and it will compare it with the current angular speed it received from the helicopter state.</p> <p>This step will provide a difference of angular speeds.</p>  |
|  <p>Torque to be applied</p>                                       | <p>From the previous step, the control system will evaluate the command it needs to send to the helicopter to change its angular speed.</p> <p>One can intuitively interpret this command as the angular acceleration, or the torque that should be applied on the helicopter, in order to modify its angular speed, to finally make the angle tend towards the desired value.</p> |

Table 4.1: Main steps of the main control system.

Of course, this computation is done at every iteration and the control system needs to constantly adapt the commands sent to correct the angle to make it reach the desired value without exceeding it and to obtain a movement as fluid as possible.

The block diagram visible on Figure 4.7, taken from the ArduPilot documentation[1], shows a more detailed and technical explanation of the control system implementation.

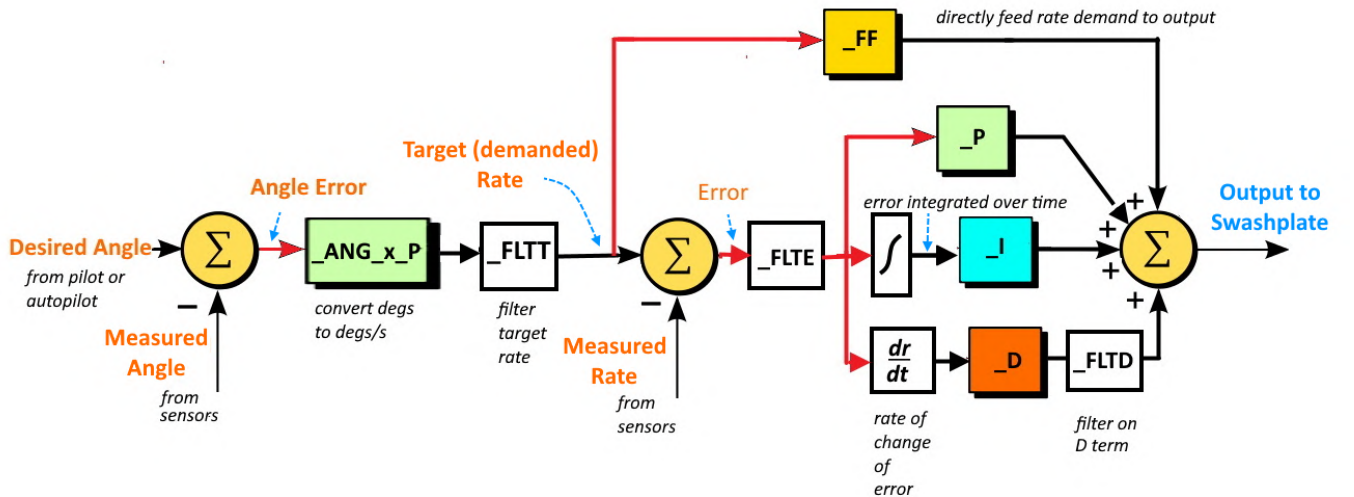


Figure 4.7: Main control system block diagram[1].

The system has 3 inputs for each angle, namely the desired angle, the current (measured) angle and the current (measured) angular rate. It generates a single output command for each angle.

- The control system receives a desired angle. It computes the current angle error between the desired one and the

helicopter one (obtained from the sensors out of the EKF algorithm, more information in section 4.4.1 **Current state management**).

- This angle error is transformed into a desired angle rate through a proportional gain.
- A feedforward and a feedback loop try then to control the helicopter according to this desired rate.
  - The feedforward is simply proportional to the desired rate (open loop).
  - The feedback loop is a PID operating on the error between the desired rate and the measured rate. A PID is basically the sum of 3 actions :
    - \* An action proportional to the error (**P**).
    - \* An action proportional to the integral of the error (**I**).
    - \* An action proportional to the derivative of the error (**D**).

Several safety mechanisms are also present such as low-pass filters, limitations on the rate error integration terms or limitations on the angular rates and accelerations.

#### 4.4.3.2 Parameterisation

Again, ArduPilot allows fine-tuning and constraining the control system through a number of parameters.

A maximum desired pitch and roll angle can be set. This defines the range of action of the flight controller. Some flight modes will limit themselves to lower angles such as the Auto flight mode.

The angle to rate gain, the feedforward gain and the 3 gains of the PID should be tuned. An AutoTune[20] flight mode exists and makes several manoeuvres with the RC helicopter in flight in order to determine the best gain corresponding to the actual RC helicopter. This flight mode is, however, a feature introduced in the 4.2.0 version of the ArduPilot software. This version was still in beta at the time we made our experimentations and the risk of using an unstable version of ArduPilot has not been taken.

An alternative exists to tune these gains manually[21]. A procedure to tune them in flight using the remote control to modify a specific gain at a time is described in the ArduPilot documentation. Tuning the gains manually is, however, not recommended as it needs to reach the limits of stability in flight to determine the best gains and an experienced pilot is recommended. This illustrates the difference in the ease of control between RC helicopters and multicopters mentioned in the introduction. The procedure to manually tune a multicopter control system is less dangerous and easier to apply.

A default value of these parameters is given such that any helicopter could at least take off and flight even if it is not perfectly smooth.

The other parameters related to the control system such as the low pass filter cut-off frequency or maximum angular rate usually have good default values and do not require further tuning.

More detailed information about the main control parameters is available in Appendix F **Main control system parameters**.

## 4.5 PWM signal transformation

### 4.5.1 Role and behaviour

This last component makes the link back from the software world to the physical world. As discussed in section 4.1 **Software overview**, this component is in charge of translating the core algorithm output commands on a kind of virtual helicopter to real commands adapted to the real RC helicopter under use.

As a reminder from section 3.2 **Actuators**, the Devil 380 has 5 motors, each controlled by a single PWM signal. These 5 signals are generated here from 4 commands returned by the Core Algorithm : throttle, pitch, roll and yaw commands. An overview of the PWM signal transformation is shown on Figure 4.8.

The throttle command can be intuitively assimilated to the wanted thrust of the main rotor. Similarly, the pitch, roll and yaw commands can be assimilated to the wanted torque on their respective axis.

Making reference to section 1 **The theory of helicopters**, we can make the following mappings :

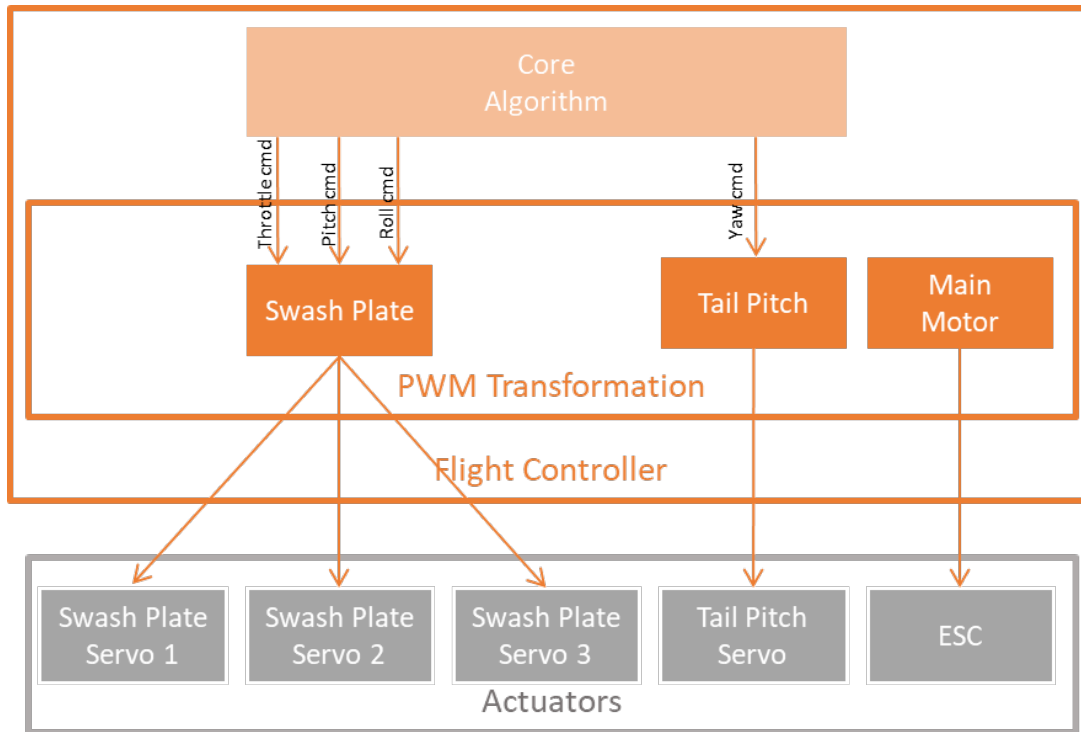


Figure 4.8: PWM signal transformation overview.

- The throttle command corresponds to a collective pitch.
- The pitch command corresponds to a cyclic pitch  $\theta_s$ .
- The roll command corresponds to a cyclic pitch  $\theta_c$ .
- The yaw command corresponds to a tail rotor blade pitch angle.

The signal of the main motor is sent to the ESC. The PWM pulse width should be constant and is therefore not influenced during the flight.

The 3 signals to the swash plate servo motors are entirely determined from the throttle, pitch and roll commands. As explained in section 1.4 **How to manage the collective and cyclic pitch?**, the throttle that corresponds to a collective pitch will define the height of the swash plate and the pitch and roll that correspond to a cyclic pitch (respectively  $\theta_s$  and  $\theta_c$ ) will define its inclination. This means that the combination of the throttle, pitch and roll information can be used to compute the exact position of the 3 servo motors to give the swash plate the correct height and orientation.

For the tail rotor pitch angle, the yaw is directly translated in a PWM pulse width to send to the tail servo motor.

This represents a simplified overview of the PWM signal transformation. Actually, a series of compensations is also implemented here to prevent some side effects. For example, if the requested command increases the throttle, this will lead to an increase of the main rotor torque that can be compensated by an automatic increase of the tail rotor thrust to prevent the helicopter from rotating around its yaw axis.

If not compensated here, the control system would correct it, but only after the perturbation due to the side effect has been detected. Doing the compensation here is an optimisation used to prevent side effects rather than react to them. This will ease the control system task and smooth the movement.

#### 4.5.2 Parameterisation

As explained in section 3.2 **Actuators**, several configurations exist to handle the main motor, the swash plate and the tail thrust of a helicopter. A parameter for each allows specifying which configuration is used. Additional

parameters exist for each of these configurations.

For the main motor, as its speed is handled by the ESC, a single parameter describing the constant PWM pulse width is sufficient.

For the swash plate, a minimum and maximum output signal corresponding to a maximum and minimum collective thrust must be set. The same applies for the zero thrust position. Depending on the swash plate servo motors configuration, it might also be necessary to inverse the collective thrust influence on the signals.

For the tail rotor, only the configuration with a specific motor and servo motor requires additional parameters, which was not the case for the Devil 380, so nothing was needed here.

Finally, the 5 output signals have to be matched to an output port of the Pixhawk 4 mini. All of these signals can be reversed (exchange long pulse width situation with small pulse width) for a greater flexibility of hardware design.

The parameterisation also includes a series of parameters to fine-tune some optimisations that try to compensate for side effects. One can think about the roll angle of the helicopter in hover to compensate for the tail rotor thrust as mentioned in section 1.6 **Other side effect forces** or the automatic adaptation of the tail rotor thrust based on the applied collective pitch.

More detailed information about the calibration and parameters linked to the PWM signal transformation is available in Appendix G **PWM signal transformation parameters and calibration**.

## 5 Flight tests

Now that we understand the physics of RC helicopters, have mounted all the necessary hardware for the flight controller to work properly, know how the flight controller works and have performed all required parameterisation and calibration beforehand, we can move on to flight testing.

### 5.1 Before the flight tests

Before starting the flight tests, it is important to keep in mind that the RC helicopter control system is not tuned yet. Indeed, the tuning of the gains of the control system has to be performed in flight, but the Devil 380 has never left the ground yet.

If the RC helicopter should be flyable by any newcomer after the full parameterisation of the flight controller, this is certainly not the case during the parameterisation phase! The responsiveness of the RC helicopter can make it difficult to control and accidents happen quickly.

We experienced this at the very beginning of our tests, the first time we started the main motor. One of the parameters was wrong and the Devil 380 started spinning, amplifying the torque generated by the main rotor instead of compensating for it. We were not able to stop it, the tail of the helicopter hit the ground and one of its blades broke off. This bad adventure delayed our flight tests because we had to find new tail blades. In addition, the shipping delays were too long, so we had to replace them with those from another RC helicopter and adapt them slightly.

Here are two pieces of advice to avoid this kind of bad surprise. They were applied after the tail blade incident.

#### Training on a simulator :

When the RC helicopter is in flight, it is important to have some reflexes. If it has an unwanted behaviour due to any perturbation, the pilot should be able to send the right instructions to correct its trajectory in a split second.

To train its reflexes in a safe environment, the pilot can use a flight simulator. The flight simulator used in our case is the free demo version of the HELI-X<sup>13</sup> flight simulator. It provides a good physics simulation and a large number of drone models (mainly RC helicopters, but also multicopters and fixed-wing drones).

The model used for training was a Goblin Kraken RC helicopter, whose performances are comparable to the Devil 380.

Moreover, we were able to use the remote control used for the flight tests to interface between the pilot and the simulator, so we could become familiar with its control.



#### Using training kit :

A training kit is a device directly attached to the RC helicopter. It can be seen as very large feet increasing the RC helicopter support surface.

By increasing the ground surface of the helicopter, it makes it easier to land. The risk of the helicopter flipping onto its side is significantly reduced as well as the risk of the tail rotor blades hitting the ground.

The training kit has, however, an important drawback. The feet stand in the air column of the main rotor blades. This air column is pushed down by the main rotor but it also has a rotational movement and is overall quite turbulent. The training kit feet are perturbing the flight of the helicopter, making it less stable and increasing the vibrations. Actually, the longer the feet, the easier the helicopter will be to land but the more disturbed it will be.

We used a homemade training kit, with a 3D printed support, and an initial foot length of 50 *cm* each. The foot length has been progressively reduced until the training kit has been entirely removed once the pilot had gained confidence. This evolution is visible on Figure 5.1

<sup>13</sup>Link to HELI-X website : <https://www.heli-x.info/cms/>





Figure 5.1: Evolution of the training kit feet length along the flight tests.

## 5.2 First flight tests : ground tests!

Before the first take-off, several safety checks must be performed to avoid any big issue during the flight.

ArduPilot has a safety test mode where it drives commands to the servo motors but leaves the main motor still. In this mode, one can check the actions resulting from the pilot instructions. For example, if we increase the throttle by moving the throttle stick up, the swash plate should rise up to increase the collective pitch angle of the main rotor blades. It is important to check that variations of the throttle, yaw, pitch and roll are handled correctly in this safety test mode before doing any test with the main rotor in motion.

Using the Stabilised flight mode in this safety test mode, it is also possible to see the corrections applied by the flight controller. Leaning the helicopter in its pitch and roll axis should result in a specific inclination of the swash plate to compensate for this “unwanted” change in the helicopter’s attitude. The effects on the yaw axis are visible on the pitch angle of the tail rotor blades. Here, the flight controller’s reaction counteracts an “unwanted” angular speed and not a simple lean angle, as for the pitch and roll.

Once ensured that the pilot instructions and the control system have a good synergy on the helicopter actuators, one can safely start the main rotor. However, it is still too early for the first take-off.

ArduPilot allows some time for the motor to start. During this time the helicopter is still on the ground and the main blades produce a torque. A slight rotation of the helicopter might appear before the control system takes over and adapts the tail rotor pitch angle. This problem can be solved by changing the default output to the tail servo motor in order to have a default tail pitch angle that compensates for the start-up torque.

As this is the first time the main rotor blades rotate, it is also time to check if they are tracking. The blades are tracking if they rotate in the same plane. This is important to reduce the vibrations of the RC helicopter in flight. This problem is further detailed in section 6.1 **Blade tracking**.

## 5.3 First take off

The RC helicopter is finally ready for its first take off! It understands the pilot’s instructions, is assisted by the flight controller and its main motor starts without any problems.

The main potential source of problems from this point resides in the control system. The default values given in the ArduPilot documentation[22] should allow the pilot to control the RC helicopter in flight even if they are not perfectly adapted to the specific RC helicopter used. To check if these gains do the job, a first flight in Stabilised flight mode is necessary. Indeed, the Stabilised flight mode relies solely on the control system and the source of potential problems is therefore limited to it.

The Stabilised flight mode is, however, hard to control by a novice pilot and requires some training beforehand. If no major issues are encountered during this first flight test, it might be interesting to switch to the AltHold flight mode.

Be prepared to switch back to Stabilised flight mode the first time you switch to AltHold mode. Indeed, the AltHold flight mode adds a level of autonomy to the flight controller but also a potential source of failure. This should not normally be the case when using the default parameters for the AltHold, which should be trouble-free.



If everything works fine, the control of the RC helicopter in AltHold is already way easier as the pilot does not have to care about the altitude any more. At this point, in order to hover in a fixed position, the pilot still has to give some pitch and roll instructions, even in the absence of wind.

The pilot correction on the pitch results from a bad level calibration. The level calibration should be done when the main rotor shaft is perfectly vertical. However it is usually done when the RC helicopter lies on the ground. This results in a bad calibration of the pitch axis. The calibration can be improved manually after this flight, based on information taken from the flight log.

For the pilot correction on the roll axis, it actually corresponds to the necessary roll of the helicopter frame to compensate for the tail rotor thrust as explained in section 1.6 **Other side effect forces**. Again, this can be manually parameterised afterward, based on information from the flight log.

As we discussed in section 4.5 **PWM signal transformation**, an optimisation can be made to preventively compensate for the increase of torque when the main rotor thrust is increased by increasing the tail rotor thrust as well. This compensation can also be fine-tuned here, based on information from the flight log.

## 5.4 Advanced flight tests

Once all the previous steps have been successfully passed, one can finally tune the RC helicopter control system gains.

As already explained in section 4.4.3 **Main control system**, the manual tuning of the gains is quite challenging but also time consuming. It can be done in the AltHold flight mode but still requires reaching the limits of stability in flight which is not recommended, especially for inexperienced pilots. We only scratched the surface of the control system manual fine-tuning, due to our lack of experience as a pilot and the lack of time, but the Devil 380 already had a very good flight behaviour.

After achieving satisfactory flight behaviour in AltHold, more advanced flight modes giving greater autonomy to the flight controller can be tested. This is the case of the PosHold and Loiter flight modes. Piloting the RC helicopter in either of these two modes is within the reach of anyone with a very short learning curve, as the helicopter keeps its position when no instructions are sent to it. There is really very little difference between these two modes, as the only difference is that the Loiter mode translates the pitch and roll positions into a speed and will maintain a constant speed instead of a constant pitch/roll angle. This gives the pilot a slightly different flight sensation.

The latest mode we tested is the Auto flight mode which introduces fully autonomous flights. Mission Planner is able to create a mission as a list of commands to be executed such as waypoints to reach. A complete list of mission commands can be found in the ArduPilot documentation<sup>14</sup>. This list of commands is then sent to the flight controller and, when the flight mode is switched to Auto, it starts executing them one after the other, without any further interaction from the pilot. A mission definition example using Mission Planner is shown on Figure 5.2.

A short video showing the main steps of the flight tests realised is available here : <https://youtu.be/LDdtr4r-VKI>.

---

<sup>14</sup>Link to the mission command list page : <https://ardupilot.org/copter/docs/mission-command-list.html>

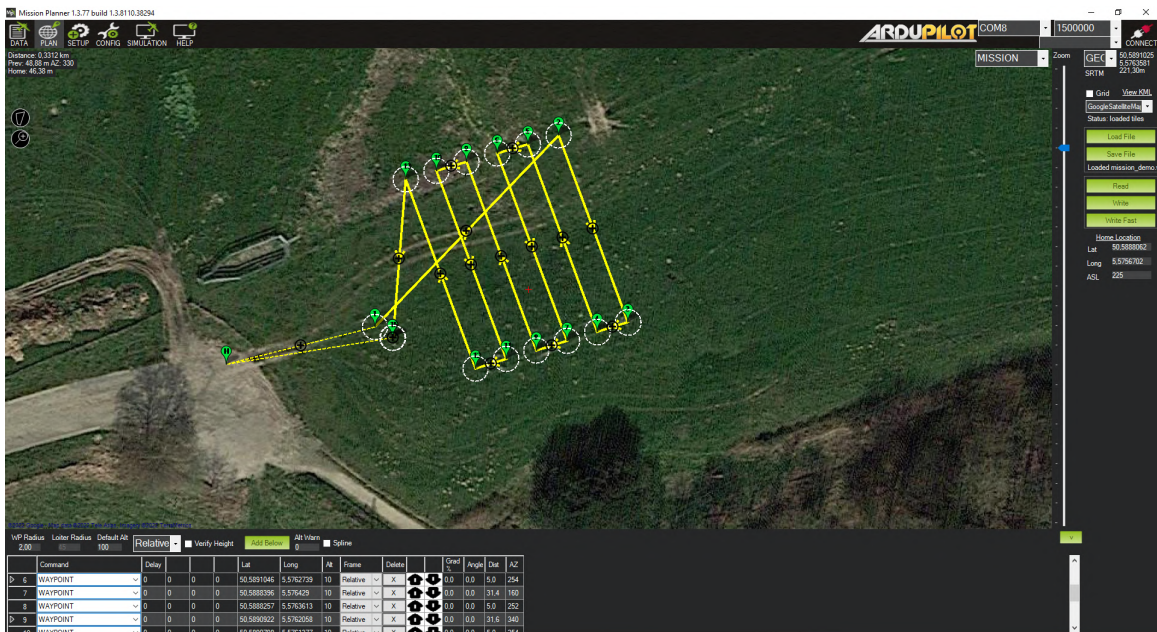


Figure 5.2: Definition of an autonomous mission using Mission Planner.

## 6 Attention point : vibrations

The motors of a drone tends to produce vibrations. The bigger the motor, the greater the vibrations. As RC helicopters have a single main motor, they undergo stronger vibrations than multicopters that have multiple smaller motors. Vibration at the main rotor frequency and some harmonics have to be expected and can lead to flight issues if they are not minimised. It can for example disturb the flight controller through noise in the IMU sensors data. Several solutions to reduce the vibration problem exist and are listed here by order of importance :

- Limit the real vibrations generated. This is the case when the blades are tracking (i.e. they rotate in the same plane).
- Isolate the flight controller sensors from the helicopter frame.
- Use a software solution (filter the sensor data).

These 3 solutions are explained in more details in the following sections.

### 6.1 Blade tracking

This is a first step to reach before the first flight test. If the blades of the helicopter do not rotate in the same plane, they will generate a lot of vibrations and it can jeopardise the flight.

For a same pitch angle, the blades rotate in the same plane if they have the exact same weight, the exact same centre of gravity and are fixed at the exact same distance from the rotor shaft. In other words, this is almost a mission impossible. One can, however, get as close to these conditions as possible.

This is done by checking the weight and centre of gravity of the blades. Some tape can be added on one or both to try to have 2 matching blades. The result on the blades of the Devil 380 is shown on Figure 6.1.

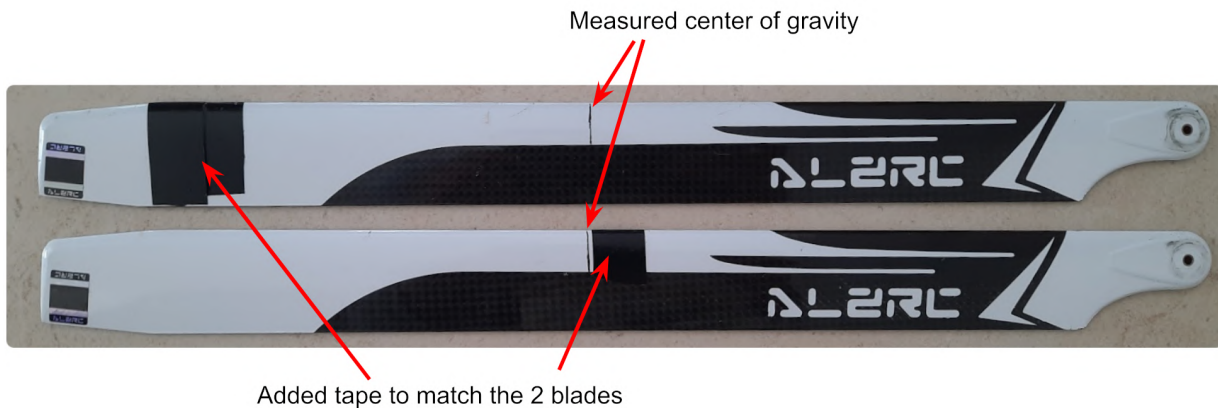


Figure 6.1: Main rotor blades of the Devil 380 after matching of their weight and centre of gravity.

From this point, the pitch link length of the 2 blades has to be adapted in an iterative manner in order to compensate for the remaining little imperfections. With a way to differentiate the blades once in motion, one can check if they are tracking by looking along their revolution plane. This is represented on Figure 6.2.

If the blades are not tracking, 2 shady blades will appear, while when they are tracking only one will be visible. In the example on Figure 6.2, the upper blade has a dark mark close to the tip. Its pitch link length should be slightly changed in order to decrease the blade pitch angle while the other pitch length should be changed in order to increase its blade pitch angle. A pitch link of the Devil 380 is shown on Figure 6.3.

After this little adjustment, one can check again if the blades are tracking and adapt the pitch link length until a satisfying result is obtained.

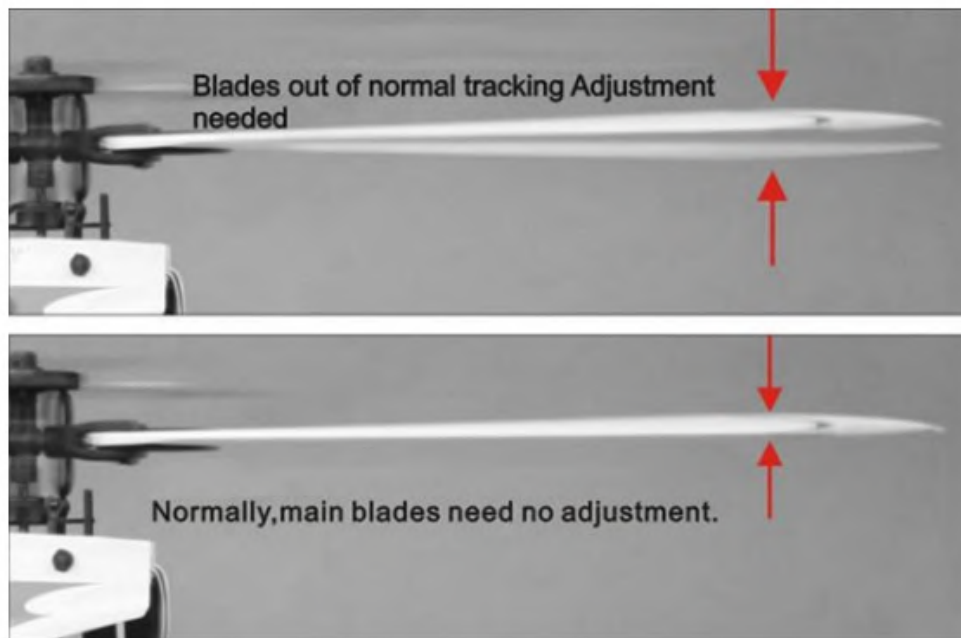


Figure 6.2: Visual appearance of non-tracking and tracking blades.



Figure 6.3: Pitch link of the Devil 380 with adaptable length.

## 6.2 Vibration damping

This solution consists in isolating the flight controller board from the RC helicopter frame with a vibration damping material.

The solution used on the Devil 380 is a little 3D printed platform mounted on rubber stoppers. This little platform is fixed in a 3D printed box attached to the RC helicopter as close as possible to its centre of gravity. The Figure 6.4 shows a picture of these elements.

## 6.3 Software filters

ArduPilot can apply a frequency filter on the data returned by the sensors in order to reduce the effects of the vibrations. For this purpose, a fast Fourier transform analysis of the log of a flight can reveal the critical frequency

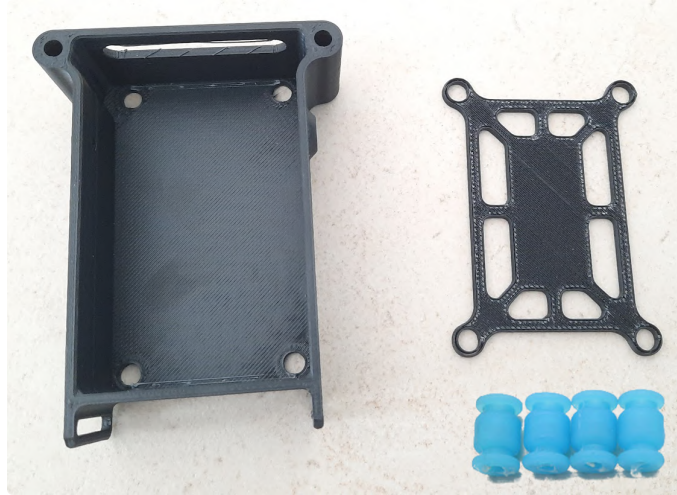


Figure 6.4: 3D printed box, 3D printed platform and rubber stoppers used for vibration damping.

to be filtered out. If a single spike of frequency is problematic (usually corresponding to the main rotor frequency), a notch filter on the specific frequency band can be set. If several spikes are problematic due to harmonic frequencies, a harmonic notch filter can be used on these specific harmonics. This solution is, however, to be used as a last resort as it consumes computing power on an already limited board.

## 7 Limitations and Further Improvements

When doing this master thesis, we have been faced with some constraints leading to limitations.

The major constraint is, of course, the limited time. As we had no experience at all with drones or helicopters, there were a lot of new theories and concepts to assimilate. We also lost some time due to technical issues (the tail rotor blades have been broken) and due to our first choice to use the PX4 for the software part. Indeed the PX4 turned out to be poorly documented for its usage for RC helicopters and not mature enough. That's why we decided to switch to the ArduPilot solution.

As we saw in this master thesis, the usage of a flight controller is a real game changer for drones as it allows non-experienced pilots to easily control them instead of requiring experienced pilots. This help given to a real pilot could, however, go even further as one can imagine many real life usages of drones where the drone could become totally autonomous and would not require any pilot any more.

With this idea in mind, we can see some limitations with what has been done in the context of this master thesis.

Firstly, the control system should be fine-tuned to provide a smoother flight. The behaviour of the helicopter in flight was already much improved by the flight controller, but it could still be improved by tuning further the gain parameters of the control system.

The tuning of the gains has not been performed very accurately for 3 main reasons :

- The gains used were close enough to make the helicopter flight without risk.
- Tuning the gain manually is a very complex task and a trained pilot is highly recommended.
- The time required to carefully perform the tuning manually was lacking.

The resulting behaviour of the helicopter is a bit coarse, showing little overshoot and ringing of the actual attitude around the desired one.

A further improvement would be to update the software version of ArduPilot to the version 4.2.0 or later and perform an autotune of the control system, as this is a new feature that appeared just after we did our flight tests.

The autotune protocol makes some manoeuvres in flight in order to deduce the ideal gains of the control system. It is able to fine-tune the gains way more accurately than with a manual tuning and it limits the risk of failures coming from the pilot unable to control a destabilised RC helicopter.

Another current limitation of the helicopter relies on the limited number of sensors used and their poor diversity. Adding more sensors would help the flight controller to be more confident on the helicopter state, but it could also broaden its scope of action.

If we think of real-life autonomous missions, without any human supervision, a critical issue will come from the possible obstacles the helicopter could meet on its predefined way. This could be birds, electrical cables, trees, or even other drones. A detection of obstacles and the possibility to avoid them is an essential feature for this kind of autonomous mission.

ArduPilot is able to correct the helicopter behaviour to avoid obstacles on its flight trajectory. However, additional sensors able to detect these obstacles are, of course, required to unlock this feature. So this would be a very interesting further improvement to add such sensors and test this capability in a real situation.

Finally, several additional flight modes could be tested. The Land and RTL flight modes can be taken as examples. The Land flight mode performs an automatic landing of the RC helicopter on its current position while the RTL flight mode drives the RC helicopter back to its take-off position before performing the automatic landing. These 2 flight modes can be triggered automatically when a problem occurs such as a battery level beyond the critical threshold.

# Conclusion

The objective of this master thesis was to demonstrate the benefits of adding a flight controller to an RC helicopter in terms of its ease of control.

This objective has been addressed through a practical approach, taking an RC Helicopter, the Devil 380, and adding a flight controller.

The main steps to carry out this approach have been :

- Acquire understanding of helicopter physics.
- Mount the various hardware components required for the flight controller, based on the Pixhawk 4 mini as the main board, some sensors to collect data during the flight, RC receiver and telemetry modules.
- Understand the operating principles of the flight controller software ArduPilot which will provide different flight modes, helping the pilot to control the helicopter.
- Parameterise the flight controller through several calibration and manual parameter settings. This was done using Mission Planner, the ground control station provided by ArduPilot.
- Become familiar with piloting an RC helicopter.
- Perform flight tests and fine-tune the parameterisation. Various flight modes were tested, including the Auto mode, a fully autonomous mode that does not require a pilot at all during the flight.

Our flight tests confirmed that the hardware and software work as expected. It also confirmed that the flight controller is a real game changer here, as it allows inexperienced pilots to easily control the RC helicopter using flight modes such as AltHold (maintain altitude), PosHold (maintain angular position) or Loiter (maintain speed). And of course, the Auto flight mode which takes the support provided further to achieve fully autonomous flight.

We finally discussed the limitations of the work done and proposed further improvements, in particular with the idea of improving the current solution to be able to plan fully autonomous missions in a real-world context without any human supervision, which requires adding more sensor capabilities like obstacle detection.

Flight controllers for helicopters and UAVs (drones) in general will certainly continue to evolve in the near future, as real-world applications are very numerous and will require more and more autonomous missions as well as longer flight duration.

This work has been very rewarding, as we have learned a lot about helicopters, drones and flight controllers. It was also instructive to see how the combination of hardware and software can solve difficult physical problems in the real world, such as helicopter flight.

Finally, this was a great human experience. Nothing is more fulfilling from a human perspective than listening to passionate people sharing their knowledge.



## References

- [1] “General arducopter attitude controller description.” <https://ardupilot.org/copter/docs/traditional-helicopter-control-system.html>. ArduPilot Copter Wiki.
- [2] “Pixhawk 4 mini.” [https://docs.px4.io/en/flight\\_controller/pixhawk4\\_mini.html](https://docs.px4.io/en/flight_controller/pixhawk4_mini.html). PX4 User Guide.
- [3] P. Mátyás and N. Máté, “Brief history of uav development,” *Repüléstudományi Közlemények*, vol. 31, no. 1, pp. 155–166, 2019.
- [4] I. Mademlis, V. Mygdalis, N. Nikolaidis, and I. Pitas, “Challenges in autonomous uav cinematography: An overview,” in *2018 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6, 2018.
- [5] E. Semsch, M. Jakob, D. Pavlicek, and M. Pechoucek, “Autonomous uav surveillance in complex urban environments,” in *2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, vol. 2, pp. 82–85, IEEE, 2009.
- [6] A. Goodchild and J. Toy, “Delivery by drone: An evaluation of unmanned aerial vehicle technology in reducing co2 emissions in the delivery service industry,” *Transportation Research Part D: Transport and Environment*, vol. 61, pp. 58–67, 2018. Innovative Approaches to Improve the Environmental Performance of Supply Chains and Freight Transportation Systems.
- [7] P. Lottes, R. Khanna, J. Pfeifer, R. Siegwart, and C. Stachniss, “Uav-based crop and weed classification for smart farming,” in *2017 IEEE international conference on robotics and automation (ICRA)*, pp. 3024–3031, IEEE, 2017.
- [8] A. Ollero and L. Merino, “Unmanned aerial vehicles as tools for forest-fire fighting,” *Forest Ecology and Management*, vol. 234, no. 1, p. S263, 2006.
- [9] P. Ramesh and J. M. L. Jeyan, “Comparative analysis of fixed-wing, rotary-wing and hybrid mini unmanned aircraft systems (uas) from the applications perspective,” *INCAS Bulletin*, vol. 14, no. 1, pp. 137–151, 2022.
- [10] J. Whidborne, “Introduction to multicopter design and control q. quan springer. 2017. xvii; 384pp. illustrated.£ 79.99. isbn 978-981-10-3381-0.,” *The Aeronautical Journal*, vol. 122, no. 1258, pp. 2044–2046, 2018.
- [11] I. A. Raptis and K. P. Valavanis, *Linear and nonlinear control of small-scale unmanned helicopters*. Springer, 2011.
- [12] G. de Ponton d’Amécourt, *La Conquête de l’Air par l’Hélice; exposé d’un nouveau système d’aviation..* 1863.
- [13] R. Raletz, *Théorie élémentaire de l’hélicoptère*. Cépaduès, 1990.
- [14] O. Dellicour, “Les bases de l’hélicoptère radiocommandé,” 2008.
- [15] G. Dimitriadis, “Lecture 9 : Helicopters,” in *Flight Dynamics and Aircraft Performance*, ULiège, Aerospace & Mechanical engineering.
- [16] P. Bruyere, B. Gombert, and E. Vence, “La dynamique de vol de l’hélicoptère.” Institution Esme Sudria.
- [17] E. Ebeid, M. Skriver, K. H. Terkildsen, K. Jensen, and U. P. Schultz, “A survey of open-source uav flight controllers and flight simulators,” *Microprocessors and Microsystems*, vol. 61, pp. 11–20, 2018.
- [18] “History of ardupilot.” <https://ardupilot.org/dev/docs/common-history-of-ardupilot.html>. ArduPilot Copter Wiki.
- [19] G. Welch, G. Bishop, *et al.*, “An introduction to the kalman filter,” 1995.
- [20] “Autotune.” <https://ardupilot.org/copter/docs/traditional-helicopter-autotune.html>. ArduPilot Copter Wiki.
- [21] “Manual tuning instructions.” <https://ardupilot.org/copter/docs/traditional-helicopter-manual-tuning.html>. ArduPilot Copter Wiki.
- [22] “Preparing for tuning - initial setup of parameters.” <https://ardupilot.org/copter/docs/traditional-helicopter-tuning-preparing.html#initial-setup-of-parameters>. ArduPilot Copter Wiki.
- [23] “Hobbywing platinum v4 escs speed-governing function.” <https://www.hobbywing.com/products/pdf/PlatinumV4.pdf>. Platinum 60A V4 - Application Guidance.



## Appendices

### A More advanced description of the flight controller board and its ecosystem

The flight controller board used is the Pixhawk 4 mini. It contains :

- A processor STM32F765<sup>15</sup> to execute the flight controller software.
- An SD Card to save flight logs, mission information and other data.
- Several sensors :
  - An IMU ICM-20689<sup>16</sup>.
  - An IMU BMI055<sup>17</sup>.
  - A magnetometer IST8310<sup>18</sup>.
  - A barometer MS5611<sup>19</sup>.

Pixhawk 4 mini also has several input/output ports to link other components to the flight controller ecosystem. Its ports are visible on Figure A.1.

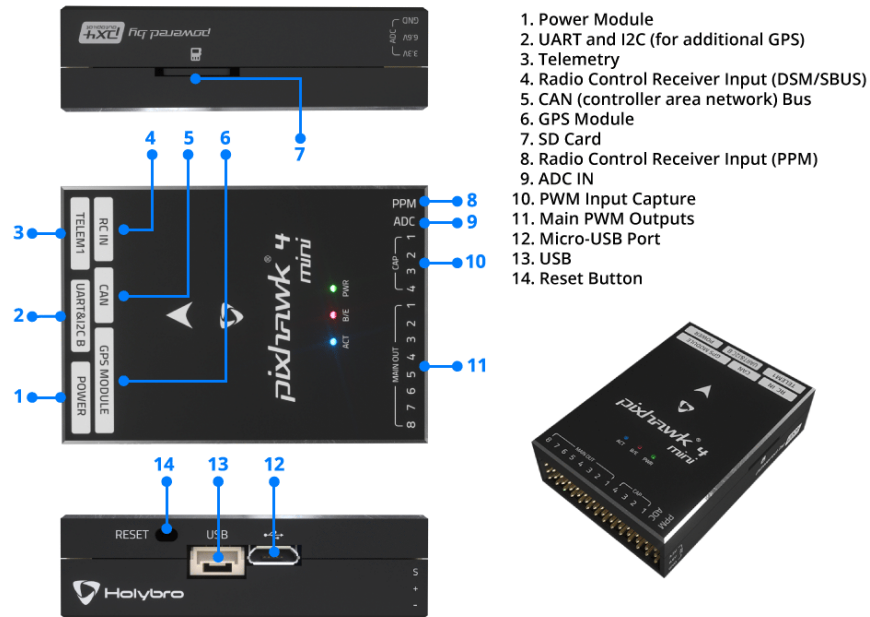


Figure A.1: Input and output ports of the Piwhawk 4 mini[2].

The ports used are :

- The power module port (port 1 on Figure A.1) to connect the power board.

The power board used is a PM06 V2.0<sup>20</sup> power board. The battery is directly connected to it, it drives the battery voltage to the main motor ESC and generates a 5 V tension to power the flight controller board. It also sends information about the battery to the flight controller thanks to a voltmeter and an ammeter.

The battery used is a lipo battery 6s with a capacity of 3300 mAh and nominal tension of 22.2 V. It can drive a current of about 190 A but a fuse on the power board limit the current to 120 A and the ESC does not consume more than 60 A. As nearly all the power consumption passes through the ESC, one can assume that the output current of the battery does not exceed 60 A.

<sup>15</sup>Link to the STM32F765 data sheet : <https://www.st.com/resource/en/datasheet/stm32f765bi.pdf>

<sup>16</sup>Link to the ICM-20689 manufacturer : <https://invensense.tdk.com/products/motion-tracking/6-axis/icm-20689/>

<sup>17</sup>Link to the BMI055 manufacturer : <https://www.bosch-sensortec.com/products/motion-sensors/imus/bmi055/>

<sup>18</sup>Link to the IST8310 manufacturer : [http://isintek.com/en/the\\_product.php?pid=4](http://isintek.com/en/the_product.php?pid=4)

<sup>19</sup>Link to the MS5611 manufacturer : [http://isintek.com/en/the\\_product.php?pid=4](http://isintek.com/en/the_product.php?pid=4)

<sup>20</sup>Link to the PM06 V2.0 manufacturer : <http://www.holybro.com/product/micro-power-module-pm06/>

- The telemetry port (port 3 on Figure A.1) to connect the SIK telemetry module.
- The radio control receiver input port (port 4 on Figure A.1) to connect the RC receiver.

The RC receiver used is an FRSky XSR<sup>21</sup> as explained in section 3.3 **Pilots communication**. This RC receiver is a very light and small component that makes it suitable for “micro” drones. To achieve such a small size and weight, some functionalities had to be sacrificed, such as telemetry. The RC receiver has 2 antennas. It is recommended to keep them straight with an angle of about 90° between them. This is to prevent connection losses due to the blind spot of the antenna along their main axis.

- The GPS module port (port 6 on Figure A.1) to connect the GPS module.

The GPS module used is a Pixhawk 4 GPS module, later renamed Holybro M8N GPS<sup>22</sup>. Its role is simply to add some sensors to the flight controller ecosystem. It is composed of 2 sensors :

- \* A GPS u-blox Neo-M8N<sup>23</sup> that can connect to 3 different satellite constellations : Galileo, GLONASS and BeiDou constellations. The more satellite connections, the better the precision.
- \* A magnetometer IST8310<sup>24</sup> (the same as the one embarked in the Pixhawk 4 mini).

It should be placed on the exterior of the RC helicopter frame for a better GPS connection and as far as possible from any motor and any carbon fibre structure to prevent magnetic field disruption for its magnetometer.

- The output 1, 2, 3, 7 and 8 of the main PWM outputs ports (port 11 on Figure A.1) to connect the ESC and servo motors.

The actuators on the Devil 380 are :

- \* An RCM-BL3120<sup>25</sup> brushless motor for the main motor.
- \* A Platinum 60A V4<sup>26</sup> ESC to control the main motor speed.
- \* 3 DS452MG<sup>27</sup> servo motors to control the swash plate position.
- \* A DS501MG<sup>28</sup> servo motor to control the tail blade pitch angle.

The ESC has 2 roles. It drives the main motor controlling its speed and it generates a fixed output tension of around 6 V to power up the servo motors. Indeed, the Pixhawk 4 mini creates instruction output signals but it does not handle power management on its own.

It can also be noted that the tail servo motor holds its position when it loses the PWM instruction signal. This is not the case of the 3 servo motors of the swash plate. This functionality is not used in our context.

- The micro-USB port (port 12 on Figure A.1) to connect the ground control station via USB cable.

This connection with the ground control station is faster than the connection with the telemetry modules and does not require to plug the battery in. However, it requires a cable connection between the RC helicopter and the ground station. It is useful to perform calibrations, parameter settings and to retrieve flight logs but not to establish the link with the ground control station in flight.

## B Sensors parameters and calibration

### B.1 Board orientation settings

The flight controller board main orientation can be defined. Its lean angles are then found through a level calibration. The lean angle of the board on the pitch axis, however, has to be tuned by hand after the first take-off test (see section 5.3 **First take off**).

The level calibration can be started from Mission Planner as shown on Figure B.1.

<sup>21</sup>Link to the FRSky XSR manufacturer : <https://www.frsky-rc.com/product/xsr/>

<sup>22</sup>Link to the Holybro M8N GPS manufacturer : [https://shop.holybro.com/pixhawk-4-gps-module\\_p1094.html](https://shop.holybro.com/pixhawk-4-gps-module_p1094.html)

<sup>23</sup>Link to the u-blox Neo-M8N manufacturer : <https://www.u-blox.com/en/product/neo-m8-series>

<sup>24</sup>Link to the IST8310 manufacturer : [http://isintek.com/en/the\\_product.php?pid=4](http://isintek.com/en/the_product.php?pid=4)

<sup>25</sup>Link to the RCM-BL3120 manufacturer : <http://www.alzrc.com/product/612.html>

<sup>26</sup>Link to the RCM-BL3120 manufacturer : <http://www.alzrc.com/product/618.html>

<sup>27</sup>Link to the DS452MG manufacturer : <http://www.alzrc.com/product/564.html>

<sup>28</sup>Link to the DS501MG manufacturer : <http://www.alzrc.com/product/567.html>

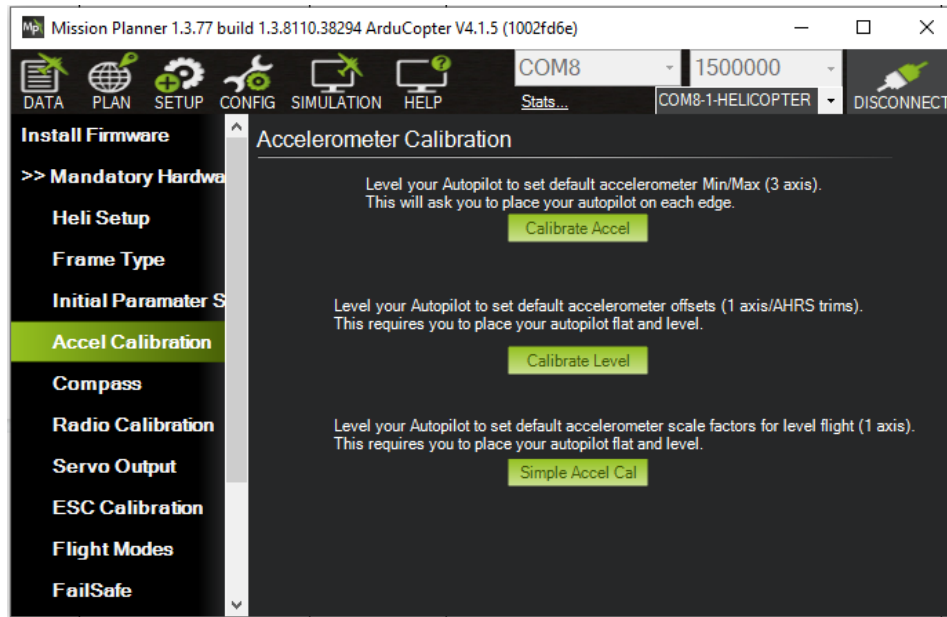


Figure B.1: Level and Accelerometer calibration page on Mission Planner.

The parameters relative to the board orientation are listed in Table B.1.

| Parameter        | Value            | Description   |
|------------------|------------------|---|
| AHRS_ORIENTATION | Yaw : 180°       | Global orientation of the flight controller board.  |
| AHRS_TRIM_X      | 0.022 <i>rad</i> | Lean angle of the flight controller board on the roll axis (level calibration).                           |
| AHRS_TRIM_Y      | 0.224 <i>rad</i> | Lean angle of the flight controller board on the pitch axis (level calibration with handmade correction). |

Table B.1: Board orientation parameters.

## B.2 IMUs calibration and parameter settings

As a reminder, 2 IMUs are used and they are both composed of a 3-axis gyroscope and a 3-axis accelerometer.

The gyroscopes are calibrated automatically, and the accelerometers have a manual calibration. The calibration of the accelerometer requires to hold still the RC helicopter in several orientations. It can be started from Mission Planner as shown on Figure B.1.

The parameters relative to the calibrations are not shown here as they are not really relevant.

Some parameters exist to correct the IMU data if needed. They are listed in Table B.2, even if their related mechanisms have not been used.

| Parameter        | Value    | Description   |
|------------------|----------|---|
| INS_POS1_X       | 0 m      | Distance to the RC helicopter centre of gravity along the helicopter heading direction for the first IMU.             |
| INS_POS1_Y       | 0 m      | Distance to the RC helicopter centre of gravity perpendicular to the helicopter heading direction for the first IMU.  |
| INS_POS1_Z       | 0 m      | Distance to the RC helicopter centre of gravity along the vertical axis for the first IMU.                            |
| INS_POS2_X       | 0 m      | Distance to the RC helicopter centre of gravity along the helicopter heading direction for the second IMU.            |
| INS_POS2_Y       | 0 m      | Distance to the RC helicopter centre of gravity perpendicular to the helicopter heading direction for the second IMU. |
| INS_POS2_Z       | 0 m      | Distance to the RC helicopter centre of gravity along the vertical axis for the second IMU.                           |
| INS_NOTCH_ENABLE | Disabled | Enable notch filtering on IMU data to reduce vibrations impact (more parameters appear if enabled).                   |
| INS_HNTCH_ENABLE | Disabled | Enable harmonic notch filtering on IMU data to reduce vibrations impact (more parameters appear if enabled).          |

Table B.2: IMU parameters.

### B.3 Magnetometer calibration and parameter settings

The calibration of the magnetometer requires to hold the RC helicopter in several orientations (as the calibration of the accelerometers) and to perform rotations. It can be started from Mission Planner as shown on Figure B.2. Once again, a lot of parameters are defined during this calibration but it would not be relevant to list them here.

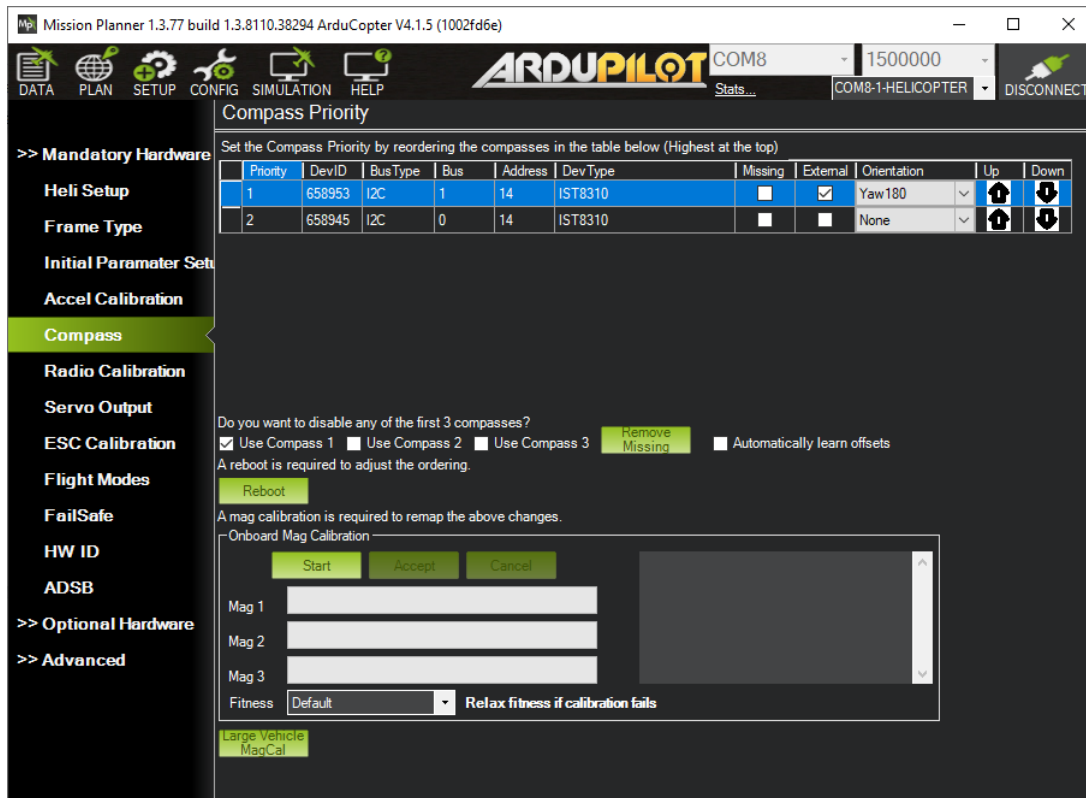


Figure B.2: Magnetometer calibration page on Mission Planner.

Some other parameters define the orientation of the magnetometer and whether or not the software should use them. The relevant ones are listed in Table B.3.

| Parameter        | Value                | Description   |
|------------------|----------------------|---|
| AHRS_ORIENTATION | Yaw : 180°           | Global orientation of the flight controller board and its magnetometer. |
| COMPASS_PRIO2_ID | Board sensor ID      | Id of the magnetometer with the second order priority.                  |
| COMPASS_USE2     | True                 | Enable the use of the magnetometer with second priority order.          |
| COMPASS_ORIENT   | Yaw : 180°           | Global orientation of the first external magnetometer.                  |
| COMPASS_PRIO1_ID | GPS module sensor ID | Id of the magnetometer with the first priority order.                   |
| COMPASS_USE      | True                 | Enable the use of the magnetometer with first priority order.           |

Table B.3: Magnetometer parameters.

#### B.4 GPS parameter settings

A lot of parameters are related to the GPS. Most of them are automatically set up to fit the precise GPS sensor used. Neither calibration, neither manual parameter setting is required.

Apart from the context of proper sensor operation, some parameters are still related to the GPS for its use by different parts of the algorithm. The ones that get our attention are listed in Table B.4.

| Parameter        | Value                          | Description  |
|------------------|--------------------------------|--|
| AHRS_GPS_MINSATS | 6                              | Minimum number of satellites to enable GPS data utilisation. Under this value, the GPS data are considered unreliable.   |
| AHRS_GPS_USE     | Horizontal position and height | Use GPS for its fast attitude and height evaluation (this evaluation is not performed by the EKF, it is a more basic state evaluation that works at a higher frequency). |
| EK3_SRC1_POSZ    | GPS                            | Source of altitude for the first EKF instance.   |
| EK3_SRC2_POSZ    | GPS                            | Source of altitude for the second EKF instance.  |
| EK3_SRC3_POSZ    | GPS                            | Source of altitude for the third EKF instance.   |

Table B.4: GPS parameters.

#### B.5 Battery monitoring calibration and parameter settings

The battery monitoring calibration consist in selecting the power board used on the RC helicopter. The role of the power board is to transmit the battery power to all components on the RC helicopter and to check its voltage and amperage. ArduPilot knows already a lot of power boards and their specific conversion parameters. The Mission Planner interface that allows the configuration of the battery monitoring system is shown in Figure B.3.

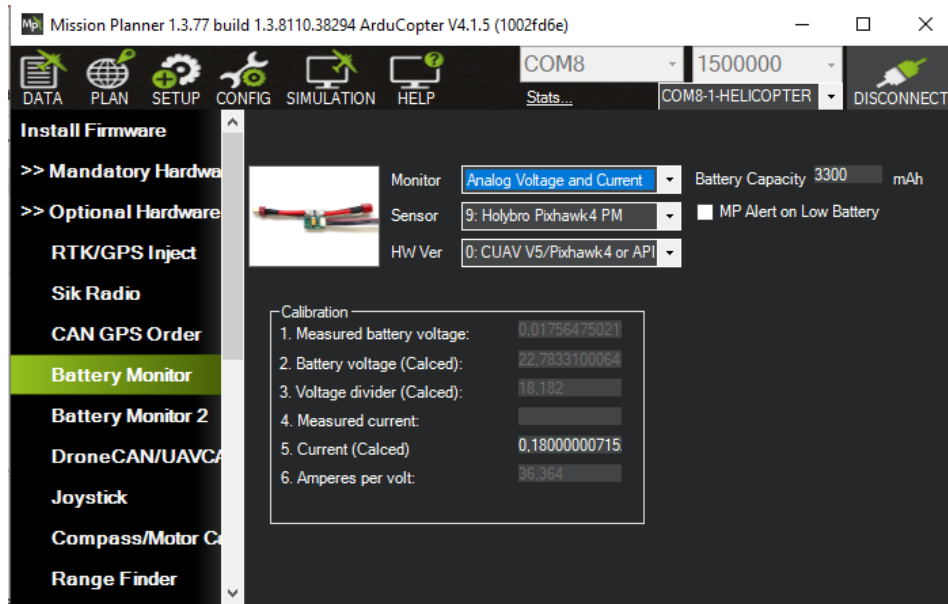


Figure B.3: Battery monitoring board selection page on Mission Planner.

The battery level is monitored in order to trigger some alerts. The thresholds and action to take are defined by a list of parameters listed in Table B.5.

| Parameter       | Value           | Description  |
|-----------------|-----------------|--|
| BATT_CAPACITY   | 3300 <i>mAh</i> | Battery capacity.  |
| BATT_LOW_MAH    | 0 <i>mAh</i>    | Battery low level threshold on the remaining capacity (0 = unused).                    |
| BATT_CRT_MAH    | 0 <i>mAh</i>    | Battery critical level threshold on the remaining capacity (0 = unused).               |
| BATT_LOW_VOLT   | 21 <i>V</i>     | Battery low level threshold on the battery voltage.                                    |
| BATT_CRT_VOLT   | 20 <i>V</i>     | Battery critical level threshold on the battery voltage.                               |
| BATT_FS_LOW_ACT | None            | Action to take on battery low level alert (e.g. start to land, return to launch).      |
| BATT_FS_CRT_ACT | None            | Action to take on battery critical level alert (e.g. start to land, return to launch). |

Table B.5: Battery parameters.

## C Remote control and telemetry parameters and calibration

### C.1 Remote control channel calibration and parameter settings

The calibration of the remote control sets the minimum, middle and maximum PWM pulse width of each RC channel used by the pilot. The related calibration page on Mission Planner is shown on Figure C.1.

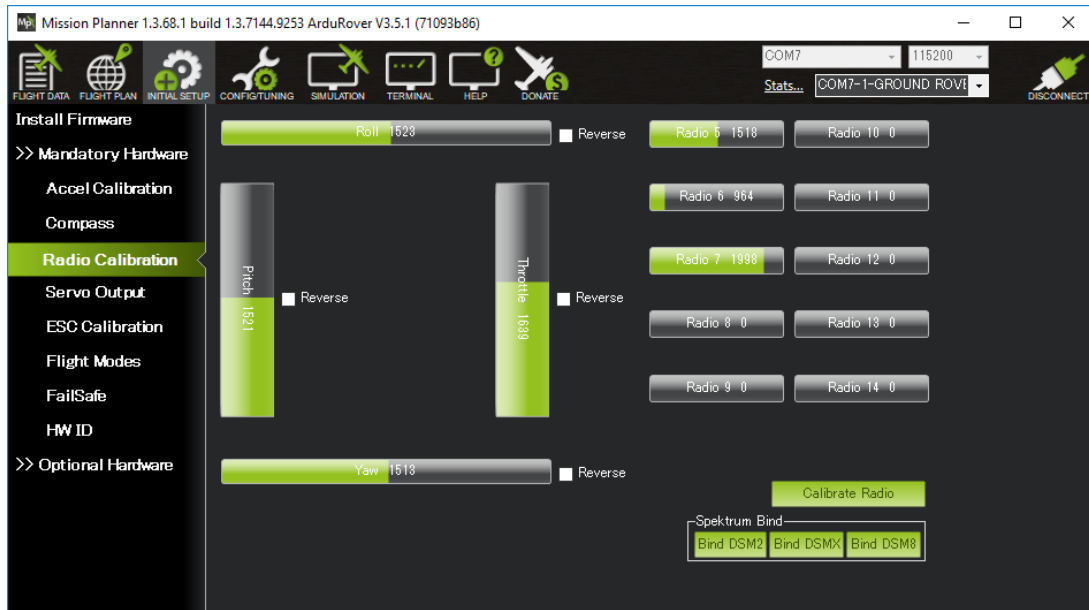


Figure C.1: Remote control calibration page on Mission Planner.

The additional parameters related to the remote control channels and their role are listed in Table C.1.

| Parameter      | Value            | Description   |
|----------------|------------------|---|
| RCMAP_THROTTLE | 3                | Throttle channel of the remote control.   |
| RC3_REVERSED   | Normal           | Reverse mode of the third RC channel input.   |
| RC3_OPTION     | None             | Specific function assigned to the third RC channel.   |
| RC3_DZ         | 10 $\mu s$       | PWM dead zone to increase the “stick at bottom” zone of the third RC channel.   |
| THR_DZ         | 100 $\mu s$      | PWM dead zone to increase the “stick at the centre” zone of the throttle RC channel (used in AltHold, PosHold and Loiter flight modes). |
| RCMAP_YAW      | 4                | Yaw channel of the remote control.  |
| RC4_REVERSED   | Normal           | Reverse mode of the fourth RC channel input.  |
| RC4_OPTION     | None             | Specific function assigned to the fourth RC channel.  |
| RC4_DZ         | 15 $\mu s$       | PWM dead zone to increase the “stick at the centre” zone of the fourth RC channel.  |
| RCMAP_PITCH    | 2                | Pitch channel of the remote control.  |
| RC2_REVERSED   | Normal           | Reverse mode of the second RC channel input.  |
| RC2_OPTION     | None             | Specific function assigned to the second RC channel.  |
| RC2_DZ         | 20 $\mu s$       | PWM dead zone to increase the “stick at the centre” zone of the second RC channel.  |
| RCMAP_ROLL     | 1                | Roll channel of the remote control.   |
| RC1_REVERSED   | Normal           | Reverse mode of the first RC channel input.   |
| RC1_OPTION     | None             | Specific function assigned to the first RC channel.   |
| RC1_DZ         | 20 $\mu s$       | PWM dead zone to increase the “stick at the centre” zone of the first RC channel.   |
| FLTMODE_CH     | 5                | Flight mode selection channel of the remote control.  |
| RC5_REVERSED   | Normal           | Reverse mode of the fifth RC channel input.   |
| RC5_OPTION     | None             | Specific function assigned to the fifth RC channel.   |
| RC5_DZ         | 0 $\mu s$        | PWM dead zone of the fifth RC channel (not useful here).  |
| RC6_REVERSED   | Normal           | Reverse mode of the sixth RC channel input.   |
| RC6_OPTION     | Motor Inter-lock | Specific function assigned to the sixth RC channel.   |
| RC6_DZ         | 0 $\mu s$        | PWM dead zone of the sixth RC channel (not useful here).  |

Table C.1: Remote control parameters.

## C.2 Telemetry module calibration

The ground telemetry module should be bound to the embedded telemetry module. This is simply done through a rapid calibration that can be performed via Mission Planner as shown on Figure C.2.





Figure C.2: Telemetry module binding page on Mission Planner.

## D Current state management parameters

The parameters relative to the current state management are the one related to the Extended Kalman Filter (EKF). These parameters are numerous as the EKF can handle a lot of different sensors and even merge data from different sensors.

The EKF parameters do not need to be changed except in rare cases. This was our case for the source of altitude measurement that uses the barometer data as default (barometers are usually more accurate than GPS for altitude measurement). The list of parameters relative to the EKF are listed in Table D.1. However, the parameters that kept their default value are not listed as they are not really relevant here.

| Parameter     | Value | Description   |
|---------------|-------|---|
| AHRS_EKF_TYPE | 3     | Extended Kalman filter version used.  |
| EK2_ENABLE    | False | Enable version 2 of the extended Kalman filter (gives access to EKF2 parameters when true). |
| EK3_ENABLE    | True  | Enable version 3 of the extended Kalman filter (gives access to EKF3 parameters when true). |
| EK3_SRC1_POSZ | GPS   | Source of altitude for the first EKF instance.  |
| EK3_SRC2_POSZ | GPS   | Source of altitude for the second EKF instance.   |
| EK3_SRC3_POSZ | GPS   | Source of altitude for the third EKF instance.  |

Table D.1: Extended Kalman filter parameters.

## E Next state interpolation parameters

### E.1 Flight mode selection parameters

The selection of the flight mode in flight is deduced from a specific RC channel value and a list of predefined flight modes. The related parameters used in this master thesis are listed in Table E.1.

| Parameter  | Value      | Description   |
|------------|------------|---|
| FLTMODE_CH | 5          | Remote control channel selecting the flight mode in flight. |
| FLTMODE1   | Stabilised | Flight mode 1.  |
| FLTMODE2   | AltHold    | Flight mode 2.  |
| FLTMODE3   | PosHold    | Flight mode 3.  |
| FLTMODE4   | Loiter     | Flight mode 4.  |
| FLTMODE5   | Auto       | Flight mode 5.  |
| FLTMODE6   | Stabilised | Flight mode 6.  |

Table E.1: Flight mode selection parameters.

## E.2 Stabilised flight mode parameters

In Stabilised mode, the next state management unit has very little to do. There are nearly no parameters related specifically to this flight mode. There is only a maximum angle for the pitch and roll axes and a throttle curve.

The throttle command sent to the PWM signal transformation is directly deduced from the pilot throttle instruction. The throttle command allows for a large range of the main blades collective pitch angle. However, such a large range is not required in Stabilised mode, especially for negative angle values.

To ease the control of the throttle by the pilot, a curve linking the throttle instruction to the throttle command for the Stabilised flight mode can be defined.

The parameters related to the Stabilised flight mode are listed in Table E.2.

| Parameter    | Value | Description   |
|--------------|-------|---|
| ANGLE_MAX    | 30°   | Maximum lean angle in pitch and roll axis of the RC helicopter frame.   |
| IM_STB_COL_1 | 42%   | First point of the throttle instruction to throttle command curve : throttle command for a throttle instruction input of 0%.    |
| IM_STB_COL_2 | 65%   | Second point of the throttle instruction to throttle command curve : throttle command for a throttle instruction input of 40%.  |
| IM_STB_COL_3 | 76%   | Third point of the throttle instruction to throttle command curve : throttle command for a throttle instruction input of 60%.   |
| IM_STB_COL_4 | 100%  | Fourth point of the throttle instruction to throttle command curve : throttle command for a throttle instruction input of 100%. |

Table E.2: Stabilised flight mode parameters.

## E.3 AltHold flight mode parameters

This flight mode adds a control loop on the throttle. This control loop receives a target altitude, a measured altitude, a measured velocity and a measured acceleration to return the throttle command. It uses 3 control steps :

- A P feedback control step from altitude signals to desired velocity.
- A PID feedback with feedforward control step from velocity signals to desired acceleration.
- A PID feedback with feedforward control step from acceleration signals to throttle command.

As for the main control system (see section 4.4.3 **Main control system**), several safety mechanisms exist to limit the reaction of this control loop. It does not need specific tuning, but the different gain and safety mechanism parameters are available. If the altitude control behaviour is not satisfactory, the parameters used to define the target altitude should be tuned first.

As the default value of the parameters of this control loop have not been changed, they are not listed here.

The next state interpolation still has to find a desired altitude to feed this control loop. The throttle instruction is understood as desired vertical velocity and some parameters allow tuning the desired behaviour. They are listed in Table E.3

| Parameter      | Value       | Description   |
|----------------|-------------|---|
| THR_DZ         | 100 $\mu s$ | PWM dead zone around the throttle trim to increase the “stick at the centre” zone in AltHold.       |
| PILOT_SPEED_UP | 2.5 $m/s$   | Maximum vertical speed used to climb in AltHold.  |
| PILOT_SPEED_DN | 0 $m/s$     | Maximum vertical speed used to descend in AltHold. If set to 0 $m/s$ , uses PILOT_SPEED_UP instead. |
| PILOT_ACCEL_Z  | 2.5 $m/s^2$ | Acceleration used to match the desired vertical velocity in AltHold.                                |

Table E.3: AltHold flight mode parameters.

## E.4 PosHold flight mode parameters

The PosHold flight mode has just a few parameters. It introduces a notion of braking where it takes over the pilot instructions to keep the drone in place. The drone “brakes” when the pitch and roll instructions are in the centred zone. In this situation, the flight controller tries to keep the RC helicopter on a fixed position and its deviations are compensated.

The PosHold parameters are listed in Table E.4.

| Parameter        | Value | Description   |
|------------------|-------|---|
| ANGLE_MAX        | 30°   | Maximum lean angle in pitch and roll axis of the RC helicopter frame (all flight mode). |
| PHLD_BRAKE_ANGLE | 20°   | Max lean angle while braking in PosHold.  |
| PHLD_BRAKE_RATE  | 4°/s  | Angular rate used while braking in PosHold.   |

Table E.4: PosHold flight mode parameters.

## E.5 Loiter flight mode parameters

The Loiter flight mode understand the pitch and roll instructions from the pilot as velocities. It has a control loop on the horizontal position of the RC helicopter whose output is a target pitch and roll angle to feed to the main control system.

The control loop on the horizontal position receives a target position, a measured position and a measured velocity. It has a P feedback control on the position and a PID feedback with feedforward control on the velocity with safety mechanisms. As for the AltHold control loop on the throttle, the different gain and safety mechanism parameters are available. If the resulting behaviour is not satisfactory, the parameters used to define the target position should be tuned first.

As the default value of the parameters of this control loop have not been changed, they are not listed here.

The next state interpolation still has to find a desired horizontal position to feed this control loop. The pitch and roll instructions are understood as desired velocities and some parameters allow tuning the desired behaviour. They are listed in Table E.5.

| Parameter      | Value                 | Description   |
|----------------|-----------------------|---|
| LOIT_ANG_MAX   | 0°                    | Maximum pitch and roll angles in Loiter. If set to 0°, uses ANGLE_MAX · 2/3 instead.    |
| ANGLE_MAX      | 30°                   | Maximum lean angle in pitch and roll axis of the RC helicopter frame (all flight mode). |
| LOIT_SPEED     | 20 m/s                | Maximum speed in Loiter.  |
| LOIT_ACC_MAX   | 5 m/s <sup>2</sup>    | Maximum acceleration in Loiter.   |
| LOIT_BRK_DELAY | 0 s                   | Delay after pitch/roll stick is centred before beginning to brake in Loiter.            |
| LOIT_BRK_ACCEL | 1.25 m/s <sup>2</sup> | Maximum braking acceleration in Loiter.   |
| LOIT_BRK_JERK  | 5 m/s <sup>3</sup>    | Maximum change in acceleration while breaking in Loiter.                                |

Table E.5: Loiter flight mode parameters.

## E.6 Auto flight mode parameters

The Auto flight mode uses the horizontal position control loop of the Loiter flight mode and the altitude control loop of AltHold. It defines a target altitude and horizontal position to feed these control loops based on its mission instructions and the parameters listed in Table E.6.

| Parameter       | Value                | Description  |
|-----------------|----------------------|--|
| AUTO_OPTIONS    | None                 | Enables some Auto flight mode options : Allow arming; Allow take off without raising throttle; Ignore pilot yaw.     |
| LOIT_ANG_MAX    | 0°                   | Maximum pitch and roll angles in Auto. If set to 0°, uses ANGLE_MAX · 2/3 instead.                                   |
| ANGLE_MAX       | 30°                  | Maximum lean angle in pitch and roll axis of the RC helicopter frame (all flight mode).                              |
| WPNAV_RADIUS    | 2 m                  | Distance to a waypoint to consider it as reached (a waypoint instruction is a 3D position).                          |
| WP_YAW_BEHAVIOR | Follows GPS course   | Determine how the yaw should be controlled by default in Auto (some mission instructions can change this behaviour). |
| WPNAV_SPEED     | 10 m/s               | Target horizontal speed in Auto.   |
| WPNAV_ACCEL     | 2.5 m/s <sup>2</sup> | Horizontal acceleration used in Auto.  |
| WPNAV_SPEED_DN  | 1.5 m/s              | Target vertical speed when descending in Auto.   |
| WPNAV_SPEED_UP  | 2.5 m/s              | Target vertical speed when climbing in Auto.   |
| WPNAV_ACCEL_Z   | 1 m/s <sup>2</sup>   | Vertical acceleration used in Auto.  |

Table E.6: Auto flight mode parameters.

## F Main control system parameters

The control system has a lot of parameters. The parameters for the yaw control are listed in Table F.1, the parameters for the pitch control in Table F.2 and the parameters for the roll control in Table F.3.

| Parameter        | Value               | Description  |
|------------------|---------------------|--|
| ATC_RATE_FF_ENAB | True                | Enable the control system feedforward paths on the angular rates.  |
| ATC_ANG_YAW_P    | 4                   | Feedback loop P gain on the yaw angle.   |
| ATC_RAT_YAW_VFF  | 0                   | Feedforward gain of the yaw angular rate.  |
| ATC_RAT_YAW_P    | 0.2                 | Feedback loop P gain on the yaw angular rate.  |
| ATC_RAT_YAW_I    | 0.12                | Feedback loop I gain on the yaw angular rate.  |
| ATC_RAT_YAW_D    | 0.003               | Feedback loop D gain on the yaw angular rate.  |
| ATC_RATE_Y_MAX   | 0°/s                | Maximum angular rate on the yaw axis. If set to 0°/s, ignore this limit.   |
| ATC_ACCEL_Y_MAX  | 225°/s <sup>2</sup> | Maximum angular acceleration on the yaw axis.  |
| ATC_RAT_YAW_SMAX | 0                   | Constraints the maximum value of the combined P and D paths of the PID on the yaw angular rate. If set to 0°/s, ignore this limit.         |
| ATC_RAT_YAW_ILMI | 0                   | Point over which the integration term on the yaw angular rate error will start to “leak”.  |
| ATC_RAT_YAW_IMAX | 0.33%               | Constraints the maximum value of the I path of the PID on the yaw angular rate.  |
| ATC_RAT_YAW_FLTT | 20 Hz               | Cut-off frequency of the low-pass filter on the target angular rate for the yaw axis. If set to 0 Hz, ignore this filter.                  |
| ATC_RAT_YAW_FLTE | 20 Hz               | Low pass filter frequency on the angular rate error for the yaw axis. If set to 0 Hz, ignore this filter.                                  |
| ATC_RAT_YAW_FLTD | 0 Hz                | Cut-off frequency of the low-pass filter on the derivative of the angular rate error for the yaw axis. If set to 0 Hz, ignore this filter. |

Table F.1: Attitude control on the yaw parameters.

| Parameter        | Value            | Description  |
|------------------|------------------|--|
| ATC_RATE_FF_ENAB | True             | Enable the control system feedforward paths on the angular rates.  |
| ATC_ANG_PIT_P    | 4.5              | Feedback loop P gain on the pitch angle.   |
| ATC_RAT_PIT_VFF  | 0.15             | Feedforward gain of the pitch angular rate.  |
| ATC_RAT_PIT_P    | 0                | Feedback loop P gain on the pitch angular rate.  |
| ATC_RAT_PIT_I    | 0.1              | Feedback loop I gain on the pitch angular rate.  |
| ATC_RAT_PIT_D    | 0                | Feedback loop D gain on the pitch angular rate.  |
| ATC_RATE_P_MAX   | $0^\circ/s$      | Maximum angular rate on the pitch axis. If set to $0^\circ/s$ , ignore this limit.   |
| ATC_ACCEL_P_MAX  | $1100^\circ/s^2$ | Maximum angular acceleration on the pitch axis.  |
| ATC_RAT_PIT_SMAX | 0                | Constraints the maximum value of the combined P and D paths of the PID on the pitch angular rate. If set to $0^\circ/s$ , ignore this limit. |
| ATC_RAT_PIT_ILMI | 0.05             | Point over which the integration term on the pitch angular rate error will start to “leak”.  |
| ATC_RAT_PIT_IMAX | 0.4%             | Constraints the maximum value of the I path of the PID on the pitch angular rate.  |
| ATC_RAT_PIT_FLTT | 20 Hz            | Cut-off frequency of the low-pass filter on the target angular rate for the pitch axis. If set to 0 Hz, ignore this filter.                  |
| ATC_RAT_PIT_FLTE | 20 Hz            | Low pass filter frequency on the angular rate error for the pitch axis. If set to 0 Hz, ignore this filter.                                  |
| ATC_RAT_PIT_FLTD | 0 Hz             | Cut-off frequency of the low-pass filter on the derivative of the angular rate error for the pitch axis. If set to 0 Hz, ignore this filter. |

Table F.2: Attitude control on the pitch parameters.

| Parameter        | Value            | Description   |
|------------------|------------------|---|
| ATC_RATE_FF_ENAB | True             | Enable the control system feedforward paths on the angular rates.   |
| ATC_ANG_RLL_P    | 4.5              | Feedback loop P gain on the roll angle.   |
| ATC_RAT_RLL_VFF  | 0.15             | Feedforward gain of the roll angular rate.  |
| ATC_RAT_RLL_P    | 0                | Feedback loop P gain on the roll angular rate.  |
| ATC_RAT_RLL_I    | 0.1              | Feedback loop I gain on the roll angular rate.  |
| ATC_RAT_RLL_D    | 0                | Feedback loop D gain on the roll angular rate.  |
| ATC_RATE_R_MAX   | $0^\circ/s$      | Maximum angular rate on the roll axis. If set to $0^\circ/s$ , ignore this limit.   |
| ATC_ACCEL_R_MAX  | $1100^\circ/s^2$ | Maximum angular acceleration on the roll axis.  |
| ATC_RAT_RLL_SMAX | 0                | Constraints the maximum value of the combined P and D paths of the PID on the roll angular rate. If set to $0^\circ/s$ , ignore this limit. |
| ATC_RAT_RLL_ILMI | 0.05             | Point over which the integration term on the roll angular rate error will start to “leak”.  |
| ATC_RAT_RLL_IMAX | 0.4%             | Constraints the maximum value of the I path of the PID on the roll angular rate.  |
| ATC_RAT_RLL_FLTT | 20 Hz            | Cut-off frequency of the low-pass filter on the target angular rate for the roll axis. If set to 0 Hz, ignore this filter.                  |
| ATC_RAT_RLL_FLTE | 20 Hz            | Low pass filter frequency on the angular rate error for the roll axis. If set to 0 Hz, ignore this filter.                                  |
| ATC_RAT_RLL_FLTD | 0 Hz             | Cut-off frequency of the low-pass filter on the derivative of the angular rate error for the roll axis. If set to 0 Hz, ignore this filter. |

Table F.3: Attitude control on the roll parameters.

## G PWM signal transformation parameters and calibration

The PWM signal transformation performs some optimisation to prevent side effect forces due to the physics of RC helicopters. The parameters linked to these optimisations are listed in Table G.1.

| Parameter        | Value | Description   |
|------------------|-------|---|
| ATC_HOVR_ROL_TRM | 3.5°  | Roll angle to compensate tail rotor thrust in hover.  |
| H_COLYAW         | 1.229 | Scale factor for the automatic adaptation of the tail thrust based on the collective pitch angle.                     |
| ATC_ANGLE_BOOST  | True  | Increases automatically the output throttle when enabled to reduce the loss of altitude when the RC helicopter leans. |

Table G.1: Side effect compensation mechanisms parameters.

The PWM signal transformation also makes the link between the software part of the flight controller and the hardware part (the ESC and the servo motors). For this purpose, it needs to map its output PWM signals to the Pixhawk 4 mini output ports and to know the actuators configuration.

The parameters related to the output mapping are listed in Table G.2.

| Parameter       | Value                     | Description   |
|-----------------|---------------------------|---|
| SERVO1_FUNCTION | Main motor                | First PWM output port function.                       |
| SERVO1_REVERSED | Normal                    | Reverse the output signal of the first output port.   |
| SERVO2_FUNCTION | Swash plate servo motor 2 | Second PWM output port function.                      |
| SERVO2_REVERSED | Normal                    | Reverse the output signal of the second output port.  |
| SERVO3_FUNCTION | Tail servo motor          | Second PWM output port function.                      |
| SERVO3_REVERSED | Reversed                  | Reverse the output signal of the third output port.   |
| SERVO7_FUNCTION | Swash plate servo motor 3 | Seventh PWM output port function.                     |
| SERVO3_REVERSED | Reversed                  | Reverse the output signal of the seventh output port. |
| SERVO8_FUNCTION | Swash plate servo motor 1 | Eighth PWM output port function.                      |
| SERVO8_REVERSED | Reversed                  | Reverse the output signal of the eighth output port.  |

Table G.2: PWM output mapping parameters.

The parameters related to the main motor configuration are listed in Table G.3. Note that the signal sent to the ESC in flight is not at 100%. This is done on purpose based on the ESC Application Guidance[23] to give it an upper-operating range.

| Parameter        | Value | Description   |
|------------------|-------|---|
| H_RSC_MODE       | ESC   | Main motor speed control configuration.   |
| H_RSC_IDLE       | 40%   | Signal sent to the ESC when the RC helicopter is armed but motor interlock is off. The ESC does not start the motor at this PWM output value. |
| H_RSC_SETPOINT   | 80%   | Signal sent to the ESC when the RC helicopter is armed and motor interlock is on.   |
| H_RSC_RAMP_TIME  | 3 s   | Time used to increase progressively the signal sent to the ESC.   |
| H_RSC_RUNUP_TIME | 10 s  | Time given to the main motor to reach is full speed.  |

Table G.3: Main motor configuration parameters.

The parameters related to the swash plate are listed in Table G.4. The zero thrust, minimum and maximum collective pitch angle PWM signals have to be tuned during a manual calibration phase. This is possible thanks to the manual override parameter.

| Parameter    | Value                  | Description  |
|--------------|------------------------|--|
| H_SW_TYPE    | 3 servo motors at 120° | Swash plate servo motors configuration.  |
| H_SV_MAN     | Disabled               | Manual override of the servo PWM signal for calibration. It can drive the maximum, minimum or zero thrust collective pitch angle PWM signal. |
| H_COL_MIN    | 1225 $\mu s$           | PWM pulse width corresponding to the minimum collective pitch angle.   |
| H_COL_MID    | 1500 $\mu s$           | PWM pulse width corresponding to the zero thrust collective pitch angle.   |
| H_COL_MAX    | 1775 $\mu s$           | PWM pulse width corresponding to the maximum collective pitch angle.   |
| H_CYC_MAX    | 2500                   | Maximum cyclic pitch angle (no unit).  |
| H_SW_COL_DIR | Reversed               | Influence of the collective pitch on the swash plate servo motors signals.   |

Table G.4: Swash plate configuration parameters.

The tail parameters related to the tail rotor are listed in Table G.5. The minimum and maximum values of the PWM pulse width of the corresponding output port have been modified to reduce the range because the tail servo motor range of action was too wide (full range is between 1000  $\mu s$  and 2000  $\mu s$ ). The trim value has been fine-tuned in order to compensate for the start-up torque of the main rotor.

| Parameter   | Value        | Description  |
|-------------|--------------|--|
| H_TAIL_TYPE | Servo only   | Tail rotor configuration.  |
| SERVO3_MIN  | 1100 $\mu s$ | Minimum PWM pulse width of the third output port.  |
| SERVO3_TRIM | 1590 $\mu s$ | Middle PWM pulse width of the third output port (output to the tail servo motor when the main rotor is starting up). |
| SERVO3_MAX  | 1900 $\mu s$ | Maximum PWM pulse width of the third output port.  |

Table G.5: Tail rotor configuration parameters.