NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

JOÃO MIGUEL FERREIRA DE CARVALHO

Master of Science in Electrical and Computer Engineering

INSTRUMENTATION AND CONTROL OF A TARGET FIXED-WING DRONE FOR LAUNCH AND CAPTURE

MASTER IN ELECTRICAL AND COMPUTER ENGINEERING

NOVA University Lisbon
November, 2022

NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

INSTRUMENTATION AND CONTROL OF A TARGET FIXED-WING DRONE FOR LAUNCH AND CAPTURE

JOÃO MIGUEL FERREIRA DE CARVALHO

Master of Science in Electrical and Computer Engineering

Adviser: Bruno J. N. Guerreiro
Assistant Professor, Department of Electrical and Computer Eng. NOVA SCHOOL OF SCIENCE AND TECHNOLOGY

Examination Committee

Chair: Doutor Paulo da Costa Luís da Fonseca Pinto

Rapporteur: Doutor Rui Alexandre Nunes Neves da Silva

Member: Doutor Bruno João Nogueira Guerreiro

MASTER IN ELECTRICAL AND COMPUTER ENGINEERING

NOVA University Lisbon
November, 2022

Instrumentation and control of a target fixed-wing drone for launch and capture

This document was created with the (pdf/Xe/Lua)LaTeX processor and the NOVAthesis template (v6.8.4) [11].

Acknowledgements

I would first like to thank FCT-UNL for all the opportunities it has given me, as well as the excellent quality of teaching it has provided me. Major thanks to my thesis advisor, professor Bruno Guerreiro, for guiding me and for always being available, with extreme commitment and with a lot of patience. I would also like to thank my family for the fantastic support they have given me, specifically my father, mother and grandfather for always being there. Last but not least, a big thank to all my friends for being very supportive and understanding during this long stage.

This work was partially funded by the FCT projects CAPTURE (PTDC/EEI-AUT /1732/2020) and REPLACE (LISBOA-01-0145-FEDER-032107), which include Lisboa 2020 and PIDDAC funds, and also project CTS (UIDB/EEA/00066/2020).

Abstract

This work was developed within the scope of the CAPTURE project, in which a collaborative network was intended to be built in which a quadcopter drone would help a fixed-wing drone perform landing and takeoff maneuvers. The study of small fixed-wing unmanned aerial vehicles (UAVs) were presented, as well as their attitude control, instrumentation, and trajectory tracking. One of the goals of this dissertation was to model a real vehicle, specifically the Easy Glider 4. All the work was developed based on this vehicle, for which it was necessary to use the XFLR software to obtain its aerodynamic response and thus obtain a more accurate model and, consequently, its control. The main challenges of this dissertation were related to obtaining the full dynamic model (with the aerodynamic coefficients included), the control techniques that would be used to deal with their nonlinearities, and their integration with a path following algorithm. Two types of attitude controllers were developed: a linear controller based on PI and a nonlinear controller based on the backstepping technique. An external loop was then added to make the UAV follow a specific path. Two different techniques were implemented: a path following algorithm that would make the vehicle follow a vector field around the intended trajectory and an adaptive algorithm capable of dealing with uncertainties in the environment, such as wind with unknown direction and intensity.

Keywords: UAV, PID, Backstepping, Path following, Vector field, Adaptive

Resumo

Este trabalho é desenvolvido no âmbito do projecto CAPTURE , em que se pretende construir uma rede colaborativa em que um drone quadricóptero ajude um drone de asa fixa a realizar manobras de aterragem e descolagem.
Será apresentado o estudo e modelação de pequenos veículos não tripulados de asa fixa (UAV), bem como o seu controlo de atitude, instrumentação e seguimento de trajetória. Um dos objectivos desta dissertação é a modelação de um veículo real, mais especificamente o *Easy glider 4*. Todo o trabalho será desenvolvido com base neste veículo, para isso, é necessário utilizar o software XFLR para obter sua resposta aerodinâmica e assim obter uma modelação mais precisa e, consequentemente, o seu controlo. Devido à complexidade da dinâmica do UAV, os principais desafios desta dissertação estão relacionados com a obtenção do modelo dinâmico, às técnicas de controlo que serão utilizadas para lidar com suas não linearidades e a sua integração com um algoritmo de path following. Serão desenvolvidos dois tipos de controladores de atitude: Um controlador linear baseado no PID e um controlador não linear baseado na técnica de backstepping. Um loop externo é então adicionado para que o UAV siga um determinado caminho. Serão implementadas duas ténicas diferentes: Um algoritmo de path following que fará o veículo seguir um campo vectorial em volta da trajetória pretendida e um algoritmo adaptativo capaz de lidar com incertezas do meio ambiente, tais como vento com direção e amplitude desconhecidas.

Palavras-chave: UAV, PID, Backstepping, Path following, Campo vectorial, Adaptativo

Contents

List of Figures

List of Tables

1

Introduction

Drones, also known as unmanned aerial vehicles (UAVs), are aircraft that are either controlled remotely or operate autonomously through the use of computers and sensors. They can be used for a variety of purposes, such as surveillance, mapping, search and rescue, delivery of goods, and even for recreational purposes. One of the main benefits of drones is their ability to operate in environments that may be hazardous or difficult for humans to access, such as disaster-stricken areas or high altitudes.

This chapter will present the context for this dissertation, its motivation, the main problem as well as the main objectives and the proposed solution.

1.1 Motivation

There are two main types of unmanned aircraft, fixed-wing aircraft and rotorcraft. As the name suggests fixed-wing aircraft have wings that do not move, coupled with ailerons and rudders to allow the vehicle to perform movements like roll, pitch and yaw. The thrust force of this type of aircraft, in line with their aerodynamics, generates the lift force, making fixed-wing aircraft extremely energy efficient, due to the fact they only need the energy to move forward. This advantage makes this type of drone ideal for long time flights. The main downside of fixed-wing aircraft lies in the fact they cannot hover in the air, because the propellers are unable to provide upward force and their aerodynamics only generate lift force at relatively high speeds, making them inappropriate for some tasks where low speeds are required. Launching and landing can also be a difficult task, needing a catapult launcher or a runway, depending on its size. On the other side, rotorcraft due to its rotor configuration which generates lift force, can remain static in the air and perform vertical movement, making them appropriate to do some tasks like photographing building facades for anomaly detection, for example. This feature is also what makes it its biggest disadvantage, energy consumption. The fact that he constantly needs the lift force to be actively generated by rotors makes his energy consumption extremely inefficient, making it inappropriate for long-time flights. Due to this feature, fixed-wing aircraft can also fly at relatively higher speeds and higher altitudes. The term

unmanned aerial vehicle does not only include the aircraft but all the systems associated, including sensors, navigation systems, users' interface and communications hardware [1]. In autonomous flight, sensors like magnetometer, accelerometer, gyroscope , static pressure sensor, dynamic pressure sensor and GPS have to be included in the aircraft.

1.2 Objectives and problem statement

The objective of the dissertation is to implement an autopilot and a path following for a UAV, including all the modelling processes and the instrumentation, integrating various systems, like the *Pixhawk 4 mini* with the ground computer, along with the UAV, its actuators and sensors.

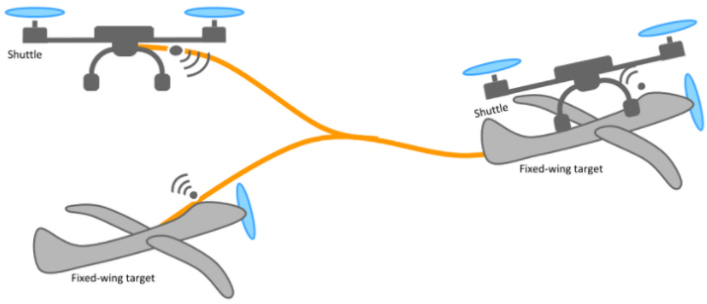


Figure 1.1: CAPTURE project [8].

This dissertation comes within the scope of the CAPTURE project, illustrated in figure 1.1, in which we see a quadcopter drone (shuttle drone) capturing a UAV (target drone) so that it can land in a confined environment. This scenario involves challenges like cooperative planning of trajectories, cooperative, hybrid and distributed control [8].
The problem presented for this dissertation is the modelling of the target drone, its instrumentation, the control of its attitude through non-linear control techniques robust to external disturbances, as well as the elaboration of an autopilot that makes the drone perform manoeuvres with a view to being integrated into a cooperation environment with the shuttle drone.

1.3 Proposed solution

The first stage of this work is to instrument the vehicle, with all the hardware necessary so it can perform a flight. As the objective is to develop controllers for the real UAV, and the added weight for carrying out an autonomous flight slightly modify the dynamics of the system, the instrumentation appears in the first place. Subsequently, the next step is to obtain the dynamic model of the UAV, described in chapter 4, and later implement it using SIMULINK. The dynamic model depends on the aerodynamic coefficients, which are calculated using the XFLR software. The next stage of the dissertation addresses the

problem of non-linear UAV attitude control. With this, it is intended that the angular position of the vehicle follows a certain reference. Linear and non-linear control techniques will be used and compared. Afterwards, a control system based on path following will be developed so that the UAV follows a certain spatial trajectory.

1.3.1 Document Outline

The dissertation is organized as follows:

- Chapter 2 contains a brief review of the coordinate frames that are going to be used in the modelling process and control techniques that were used and some others that are likely to be used in future work.
- Chapter 3 deals with the vehicle instrumentation problem, showing all the materials and methods that were used to assemble all the hardware necessary for the UAV to fly.
- Chapter 4 contains a UAV modelling, using SIMULINK.
- Chapter 5 presents the autopilot and the path following development.
- Chapter 6 presents the experimental results of the work developed in Chapter 5.
- Chapter 7 concludes the dissertation.

2

Background and related work

In this chapter, the theoretical bases used for the work developed are presented. Topics such as coordinate frames and basic control techniques will be exposed. Given those, we have all the fundamentals to understand all the work presented.

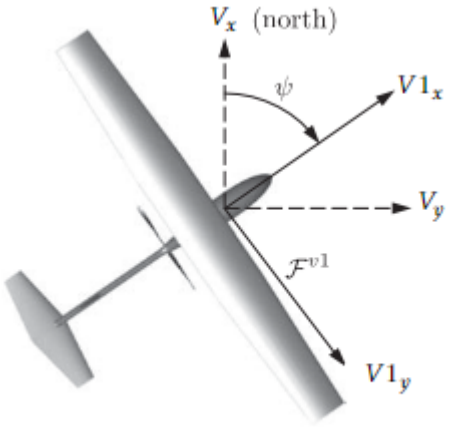
2.1 Coordinate Frame

When we build a dynamic control system that interacts with its environment it's essential to track its position and its velocity in space [23]. In this chapter, several coordinate systems are going to be presented, to represent the position of the aircraft relative to the earth referential, how it is oriented compared to the wind, and the geometrical transformations between these coordinate frames. To build the UAV dynamic model and its subsequent control system, four coordinate frames need to be considered, which are: the inertial frame $F^I = \{I_x, I_y, I_z\}$- also known as a north-east-down (NED) reference frame, considering the inertial x-direction referred to as the north, the inertial y-direction referred to as east, and the inertial z-direction referred to as down-, body frame $F^B = \{B_x, B_y, B_z\}$,- being the origin the centre of mass-, wind frame $F^W = \{W_x, W_y, W_z\}$, stability frame $F^S = \{S_x, S_y, S_z\}$, and the vehicle frame $F^V = \{V_x, V_y, V_z\}$. The vehicle frame is aligned with the inertial frame, but with its origin located at the centre of mass of the aircraft. However, by rotating the vehicle frame about the V_z axis by the yaw ψ angle, we obtain the F^{v1} coordinate frame [5].

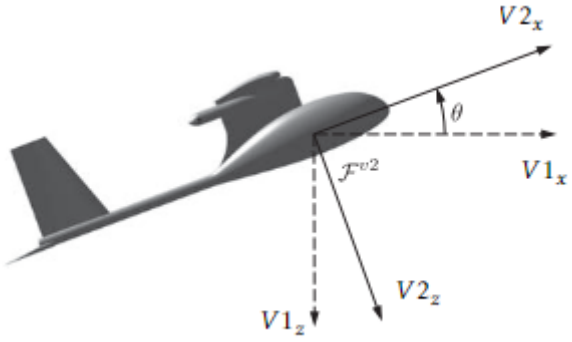
$$R_V^{V1}(\psi) = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.1}$$

By rotating the F^{v1} with the following rotation matrix about the V_y axis by the pitch angle θ, we obtain the F^{v2} coordinate frame

$$R_{V1}^{V2}(\psi) = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}. \tag{2.2}$$



(a) Vehicle-1 frame[5].



(b) Vehicle-2 frame

Figure 2.1: Vehicle coordinate frame[5].

By rotating the F^{v2} frame with the following rotation matrix about the $V2_x$ axis by the roll angle ϕ we obtain the body frame F^b, which also has its origin at the centre of mass of the aircraft [5].

$$R^b_{v2} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \tag{2.3}$$

As the inertial frame has the same orientation as the vehicle frame and considering that we can transform the vehicle frame into a body frame, we can use the Euler angles to define the relative orientations of the inertial frame and the body frame using the Euler Angles roll ϕ, pitch θ and yaw ψ, by applying the rotation matrix

$$R^B_I(\theta,\psi,\phi) = \begin{bmatrix} c_\psi c_\theta - s_\psi s_\phi s_\theta & -s_\psi c_\phi & c_\psi s_\theta + s_\psi s_\phi c_\theta \\ s_\psi c_\theta + c_\psi s_\phi s_\theta & c_\psi c_\theta & s_\psi s_\theta - c_\psi s_\phi c_\theta \\ -c_\phi s_\theta & s_\phi & c_\phi c_\theta \end{bmatrix}. \tag{2.4}$$

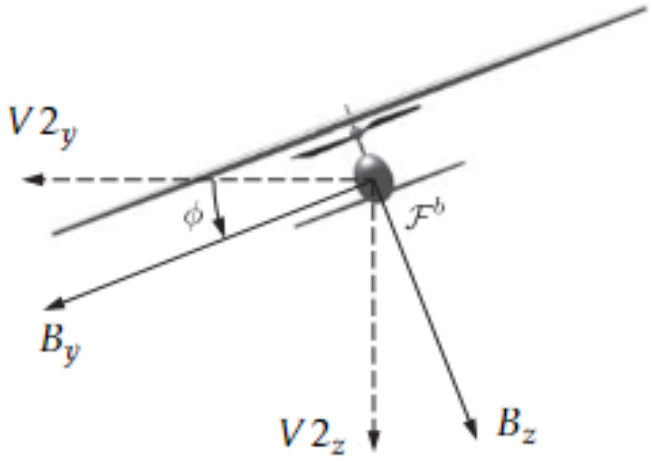


Figure 2.2: Body frame[5].

The stability frame or the aerodynamic frame is determined by the instantaneous orientation of the relative wind vector to the airframe [19]. As the aircraft moves through the air, aerodynamic forces are created. For example, the lift force is generated if the airframe travels towards the air vector direction V_a with an angle α, named the angle of attack [5]. The transformation matrix that makes the relation between the body frame and the stability frame is described by

$$R_b^s(\alpha) = \begin{bmatrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix}. \tag{2.5}$$

The wind frame is formed by rotating the stability frame by an angle β called the side-slip angle [5]. When this angle is zero, the wind frame and the stability frame are coincident [19]. The transformation between the stability frame and the wind frame is given by

$$R_s^w(\beta) = \begin{bmatrix} \cos\beta & \sin\beta & 0 \\ -\sin\beta & \cos\beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.6}$$

and the transformation from the wind frame to the inertial frame is

$$R_w^i(\alpha,\beta,\theta,\phi,\psi) = \begin{bmatrix} c_\alpha c_\beta & s_\beta & s_\alpha s_\beta \\ -c_\alpha s_\beta & c_\beta & -s_\alpha c_\beta \\ -s_\alpha & 0 & c_\alpha \end{bmatrix} \begin{bmatrix} c_\theta c_\psi & c_\theta s_\psi & -s_\theta \\ s_\phi s_\theta c_\psi - c_\phi s_\psi & s_\phi s_\theta s_\psi + c_\phi c_\psi & s_\phi c_\theta \\ c_\phi s_\theta c_\psi + s_\psi s_\psi & c_\phi s_\theta s_\phi - s_\phi c_\psi & c_\phi c_\theta \end{bmatrix}. \tag{2.7}$$

The wind triangle establishes the relationship between the speed at which the UAV travels in relation to the surrounding fluid V_a and the ground V_g, introducing variables

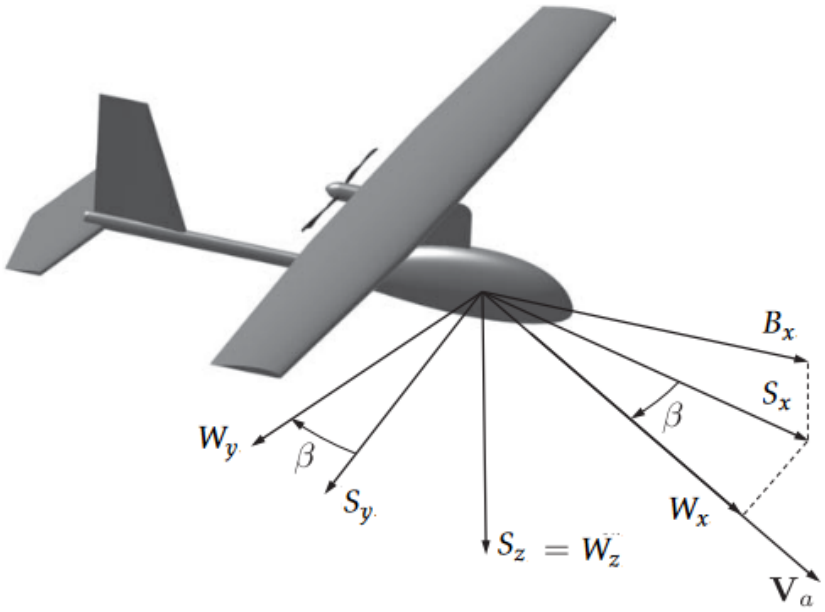


Figure 2.3: The wind frame[5].

such as the flight path angle γ and course angle χ. The figure 2.4 shows us the relationship between these variables. The course angle χ is the angle between the inertial frame and the ground speed V_g and the crab angle χ_c is the angle between the body frame and V_g. The crab angle can also be defined as

$$\chi_c = \chi - \psi. \tag{2.8}$$

In figure 2.5 we can see the vertical projection of the wind triangle, introducing the flight path angle γ and the air-mass-referenced flight path angle γ_a, which is calculated as

$$\gamma_a = \theta - \alpha. \tag{2.9}$$

Given all these references, the ground speed V_g can be obtained by

$$V_g^i = \begin{bmatrix} \cos\chi & -\sin\chi & 0 \\ \cos\gamma & 0 & \sin\gamma \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\gamma & 0 & \sin\gamma \\ 0 & 1 & 0 \\ -\sin\gamma & 0 & \cos\gamma \end{bmatrix} \begin{bmatrix} V_g \\ 0 \\ 0 \end{bmatrix} \tag{2.10}$$

where $V_g = \|V_g\|$. Similarly, the airspeed vector in the inertial frame may be written as

$$V_a^i = V_a \begin{bmatrix} \cos\psi\cos\gamma_a \\ \sin\psi\cos\gamma_a \\ -\sin\gamma_a \end{bmatrix} \tag{2.11}$$

where $V_a = \|V_a\|$. Knowing the values of V_g and V_a we can calculate γ_a as

$$\gamma_a = \arcsin(\frac{V_g \sin\gamma + w_d}{V_a}). \tag{2.12}$$

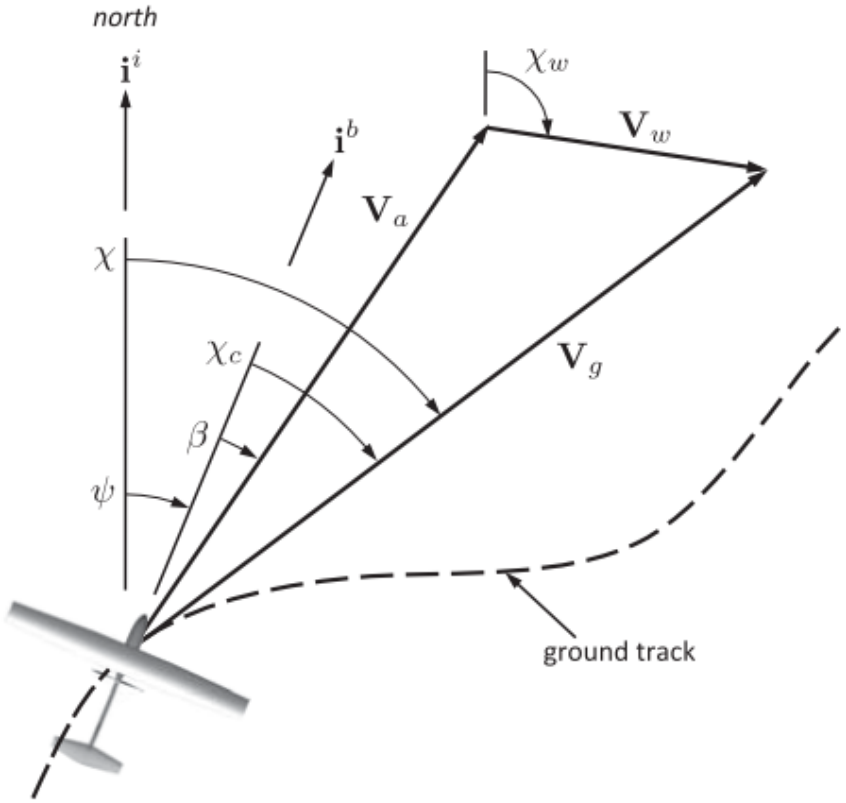


Figure 2.4: Relationship between ground speed, airspeed and inertial frame, where V_w represents the wind velocity relatively to the inertial frame, as well as the angles between them [5].

2.2 Control Techniques

In the face of external disturbances, a flight control system is necessary to guarantee that the UAV shows stable behaviour and delivers desirable performance, such as following a specified trajectory with sufficient precision. This is accomplished by attempting to remove the tracking error between the given reference and the UAV's measured response [7]. There are many various ways and rules to design a controller; in this example, different control techniques will perform better in some jobs than others, depending on the UAV model and its function [18]. Control laws typically can be organized into two

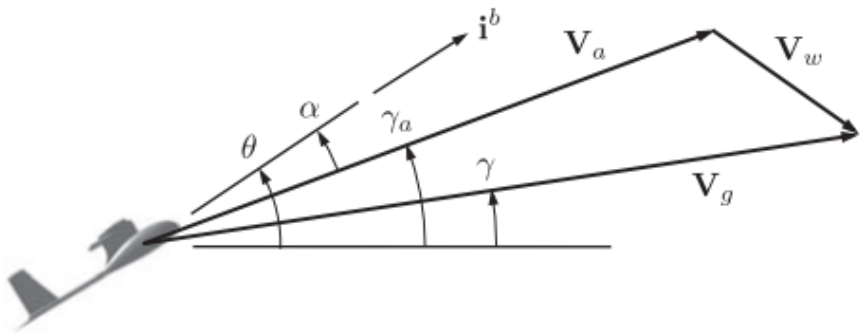


Figure 2.5: Vertical projection of the relationship between ground speed, airspeed and inertial frame, as well as the angles between them [5].

main categories: Linear and non-linear. In this chapter, some examples of each category will be presented.

2.2.1 Linear control

Linear control has the advantage of being simple to implement and can be a good strategy in preliminary work development. However, because it is necessary to linearize the system, the control law may sometimes not be adequate for the real system, in case it has non-linear dynamics.

2.2.1.1 PID

The PID (proportional, integral and derivative) controller as the name implies has proportional, integral and derivative action. The PID controller is the most widely used control algorithm, due to its simplicity in tuning. It is good enough for a lot of control issues, especially when the process dynamics aren't too complicated and the performance requirements aren't too high. The main role of the integral action is to eliminate the steady-state error due to the proportional component [2], as well as add some robustness to the system due to external disturbances. The main function of the derivative action is to improve closed-loop stability, by producing a phase advantage. Along with proportional gain, it can provide an anticipation behaviour [1]. The PID control law can be described as

$$u(t) = K_p\left(e(t) + \frac{1}{T_i}\int_0^t e(\tau)d\tau + T_d\frac{d}{dt}e(t)\right) \tag{2.13}$$

There are many techniques to tune a PID, although the most common methods for optimum settings are the *Ziegler-Nichols* methods, more specifically the ultimate gain method.

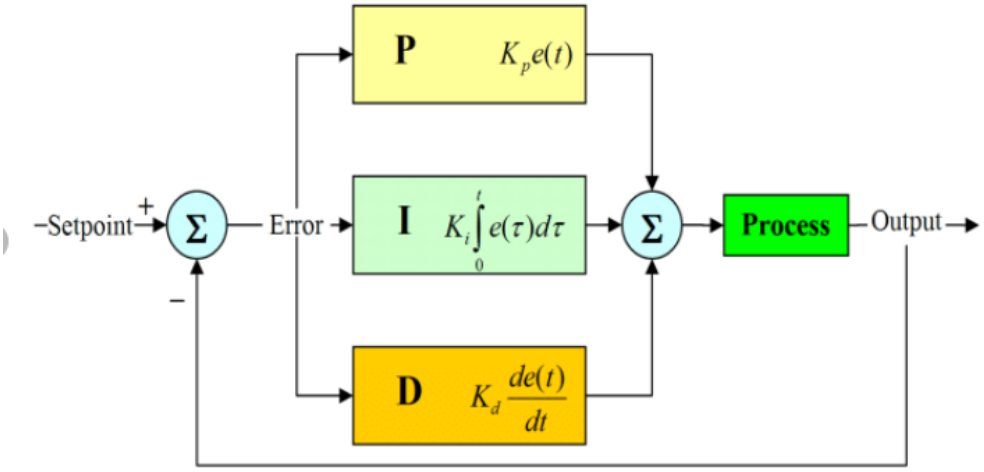


Figure 2.6: PID block diagram [15].

1. Eliminate the actions of the integral and derivative effect of the PID controller, placing T_i and T_d in their maximum and minimum values, respectively;
2. Tune K_p to a low value and run the control system;
3. Increase the value of k_p until there is a permanent oscillation of constant amplitude in the controlled variable;
4. Measure the value of k_p and assign it to the critical stability gain variable, K_{cr};
5. Measure the value of the critical oscillation period, P_{cr};
6. Use the empirical relationships proposed by Ziegler-Nichols and determine the PID controller parameters: K_p, T_i, and T_d;
7. Tune the controller gain value, K_p, to the recommended value;
8. Tune the value of the constant of integration, T_i, to the recommended value;
9. Tune the value of the constant of derivation, T_d, to the recommended value;
10. Observe the performance of the control system and, if necessary, delicate tune the parameters;

2.2.1.2 Linear MPC

MPC makes predictions about the future behaviour of a system using its model. MPC solves an optimization method to determine the best control action for driving the expected output to the reference. MPC can manage MIMO systems that may have interactions between them. It also has the ability to deal with input and output restrictions [13].

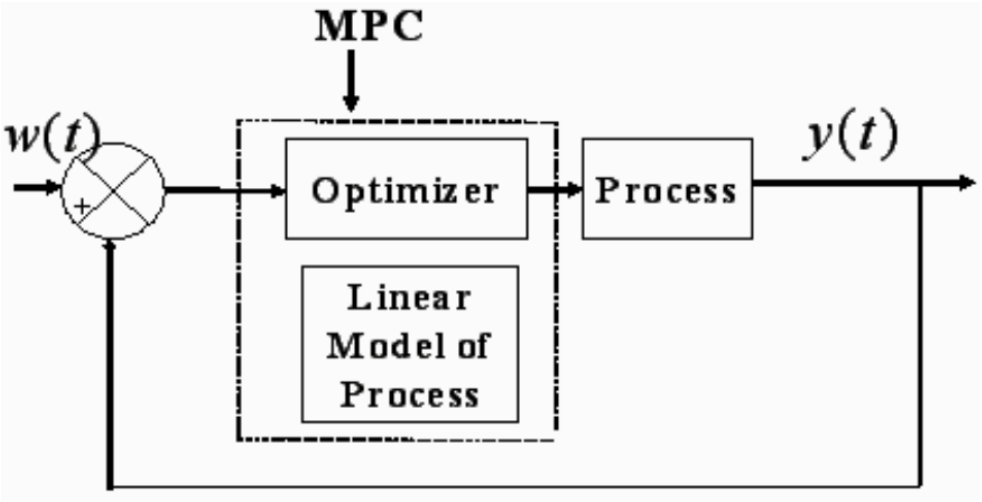


Figure 2.7: Linear MPC block diagram [4].

The MPC problem consists of an optimization problem, that aims to find an optimal solution for a control sequence u_k^* while minimizing the performance function J, according to its system dynamics and within input and state restrictions. For example, considering the quadratic problem, with the current state $x(t_k)$ as initial condition x_0

$$\min_U \quad J = (x_0, U) = x_N^T P x_N + \sum_{k=0}^{N-1} x_k^T Q x_k + u_k^T R u_k \tag{2.14}$$

constrained by

$$x_k \in \chi, u_k \in U, \forall_{k=0,\dots,N-1}, x_N \in \chi_f. \tag{2.15}$$

If the optimization is feasible, from the optimal sequence U^* apply the first one to the system, $u(t_k) = u_0^*$.

2.2.2 Non-linear control

The importance of studying nonlinear systems is related to the fact that the general real systems, from physical, biological or economic systems, among others, show a non-linear behaviour and its representation through a linear approximation is not, in many situations, desirable or admissible. When systems show notably non-linear behaviour, using linear controllers tuned to a specific operating point might cause a reduction in the closed-loop system's performance when the desired output needs to be changed, which can lead to control system instability in some cases, [16]. therefore, in most cases, non-linear control is the most viable option.

2.2.3 Exact linearization of non-linear systems

As its name suggests, the exact linearization method is based on obtaining a linear model through an algebraic transformation (see figure 2.8), from which controllers are designed

using linear techniques, such as using the feedback of state variables [16].

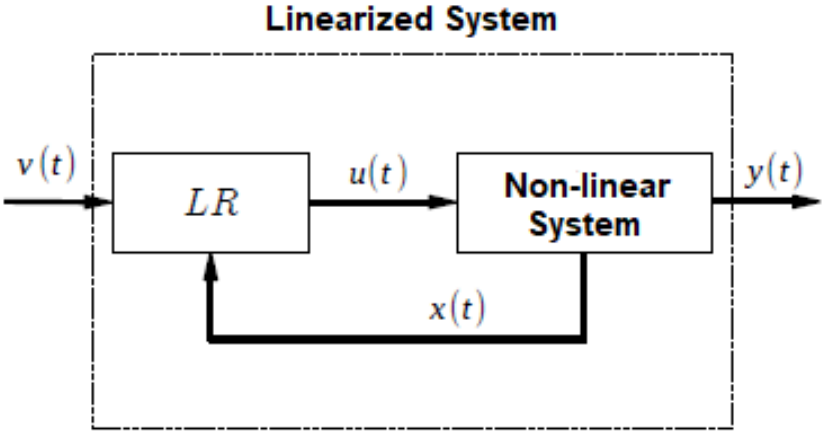


Figure 2.8: Exact linearization method [16].

Considering the non-linear system

$$\dot{x}(t) = f(x(t), u(t)) \tag{2.16}$$

for which we want to find a state transformation defined generically by $z(t) = \varphi(x(t))$ and an input transformation such that $u(t) = \zeta(x(t), v(t))$. Through those algebric transformations, the non-linear system is transformed into a linear system, described by

$$\dot{z} = Az(t) + Bv(t) \tag{2.17}$$

from which it is possible to build a controller by eigenvalue placement.

2.2.4 Integrator Backstepping

Backstepping control design is a recursive design technique in which a design problem on a higher-order system is broken down into a series of sub-problems on lower-order systems, which are designed with Control-Lyapunov functions [12]. Considering the model

$$\begin{cases} \dot{\eta} = f(\eta) + g(\eta)\xi \\ \dot{\xi} = u \end{cases} \tag{2.18}$$

such that $\eta \in \mathfrak{Re}^n$, $\xi \in \mathfrak{Re}$, $f : D \to \mathfrak{Re}^n$, $g : D \to \mathfrak{Re}^n$, $D \in \mathfrak{Re}^n$, $f(0) = 0$.
The goal is to design u such that the system is asymptotically stable at the origin $\eta = 0$ and $\xi = 0$.

Assuming we know a state feedback control law $\varphi(\eta)$ such that $\dot{\eta} = f(\eta) + g(\eta)\varphi(\eta)$ is asymptotically stable at the origin.That means we know some Lyapunov function $V(\eta)$ where $\dot{V}(\eta) = \frac{\partial V}{\partial \eta}[f(\eta) + g(\eta)\varphi(\eta)] \leq -W(\eta) < 0$. This means we know how to stabilize the

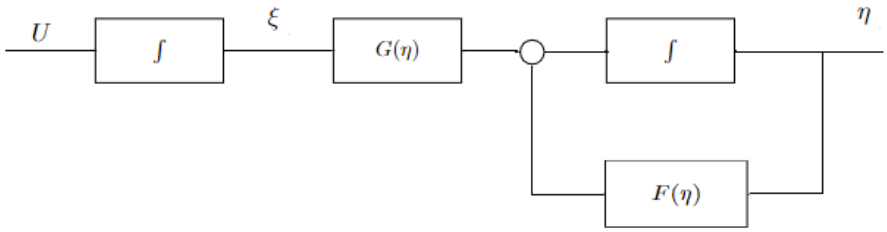


Figure 2.9: Block diagram of the system [12].

system. The next step is to find the control law u that stabilizes the system including the integrator. Let

$$\dot{\eta} = f(\eta) + g(\eta)\eta + g(\eta)\varphi(\eta) - g(\eta)\varphi(\eta)$$
$$\dot{\eta} = f(\eta) + g(\eta)\varphi(\eta) + g(\eta)[\eta - \varphi(\eta)]$$

and

$$z = \eta - \varphi(\eta)$$
$$\dot{z} = \dot{\eta} - \dot{\varphi}(\eta)$$
$$\dot{z} = u - \dot{\varphi}(\eta).$$

Redesigning the system, we see that the signal $-\varphi(\eta)$ was set back in Figure 2.10. This is why this configuration is called Backstepping. Defining $v = \dot{z}$ we can write V_a as a candidate to Lyapunov function, described as

$$V_a = V(\eta) + \frac{1}{2}z^2$$
$$\dot{V}_a = \frac{\partial V}{\partial \eta}\dot{\eta} + z\dot{z} \Leftrightarrow$$
$$\dot{V}_a = \frac{\partial V}{\partial \eta}[f(\eta) + g(\eta)\varphi(\eta)z] + zV$$
$$\frac{\partial V}{\partial \eta}g(\eta) + v = -kz$$
$$v = -kz - \frac{\partial V}{\partial \eta}g(\eta)$$
$$v = u - \dot{\varphi}(\eta)$$
$$u = v + \dot{\varphi}(\eta).$$

And finally, the backstepping control law is given by

$$u = -k(\xi - \varphi(\eta)) - \frac{\partial V}{\partial \eta}g(\eta) + f(\eta) + g(\eta)\xi. \tag{2.19}$$

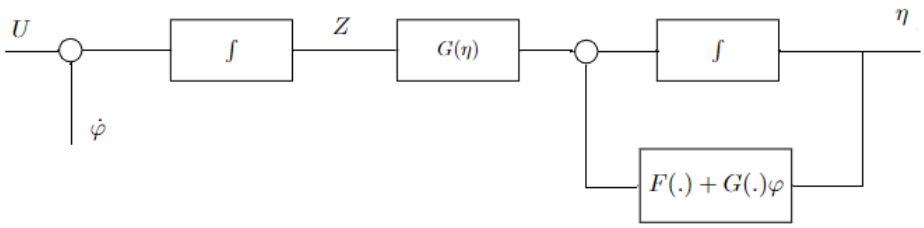


Figure 2.10: Backstepping applied [12].

2.2.5 Sliding mode

The sliding mode technique- SMC, consists of reducing the control problem of a system generic, described by a nonlinear equation of "n"order, for one of 1st order, with uncertainties in its parameters and/or in its mathematical modelling . The SMC methodology consists of designing a control law that converges all trajectories of that system to a defined surface in state space, called the sliding surface $S(t)$ [22]. The dynamics of this surface are chosen by the designer so that all trajectories within the surface S(t) will converge to their desired value. The sliding mode technique consists of two steps: the first is to define the sliding surface that makes the dynamic system stable, and the second is to define a control law that guarantees that all trajectories converge to the sliding surface. Consider the "n"order nonlinear system

$$x^n = f(X,t) + b(X,t)u + d(t) \tag{2.20}$$

where X is the state vector, $f(X,t)$ and $b(X,t)$ are nonlinear functions, and $d(t)$ is a perturbation. We can define the tracking error by $\tilde{x} = x - x_d$ and the sliding surface

$$S(\tilde{x},t) = (\frac{d}{dt} + \lambda)\tilde{x} \tag{2.21}$$

where λ is a positive constant. The figure 2.11 illustrates the sliding surface.
The control law u is designed so that reaches the surface $S(\tilde{x},t) = 0$ in a finite time interval. Once reached, stay sliding on it indefinitely. To obtain the control law of the system, one must differentiate the equation 2.2.5 with respect to time, which results in

$$\dot{S} = (\ddot{x} - \ddot{x}_d) + \lambda\tilde{\dot{x}} = 0. \tag{2.22}$$

The control law is given by

$$u = k\text{sign}(S) \tag{2.23}$$

where k is a positive constant and $sign$ is

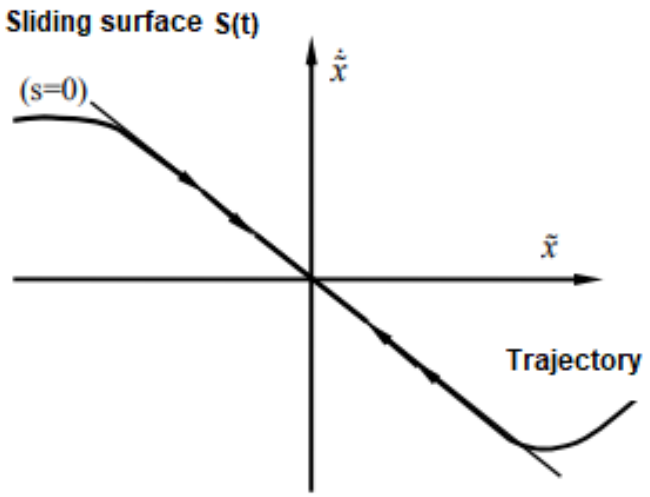


Figure 2.11: Sliding surface [22].

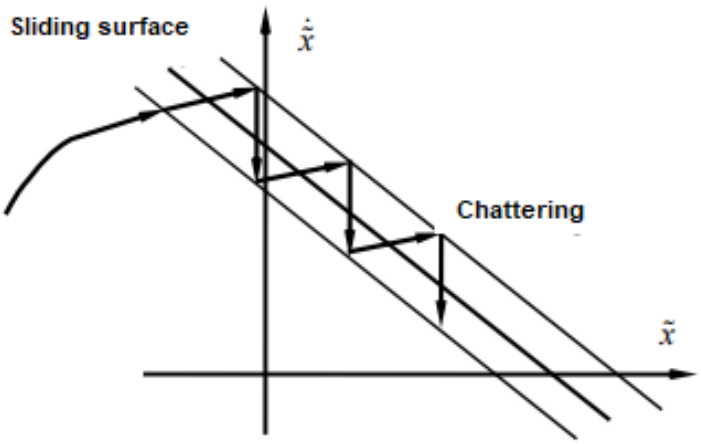


Figure 2.12: Chattering phenomena [22].

$$\text{sign}(S) = \begin{cases} +1, & S \geq 0 \\ -1, & S \leq 0. \end{cases} \tag{2.24}$$

This can introduce a phenomenon called "chattering", as the controller can commute at infinite frequency, illustrated in figure 2.12. The smoothed control law is therefore given by

$$\text{sat}(S) = \begin{cases} \text{sign}(S), & |S| > w \\ S, & |S| \leq w \end{cases}. \tag{2.25}$$

3

Instrumentation

This chapter will describe the part of the configuration and assembly of the hardware necessary for the UAV to perform an autonomous flight as well as a manual flight, this includes wiring, electronic components selection, flight modes configuration and sensor calibration. The UAV employed to carry out the real flight and used to project the controllers is the *Easy Glider 4* illustrated in figure 3.1.

Figure 3.1: Easy glider 4.

3.1 Hardware assembly

For this flight test, the PX4 will be used, which is an open source open hardware project, developed by Pixhawk, aimed at providing a high-end autopilot for small unmanned vehicles. It is available at a low cost and is highly versatile. The easy glider has a wingspan of 1800 mm, an overall length of 1080 mm and a wing area of 40 dm^2. With the hardware mounting configuration in Figure 3.4, the mass on the easy glider goes from 1100g to 1470g, and its center of mass is shifted from 70mm (measured at the root leading edge)

to 57 mm. The center of mass was estimated by replicating the easy glider 4 model and its mass distribution in the XFOIL software, illustrated in 3.2.

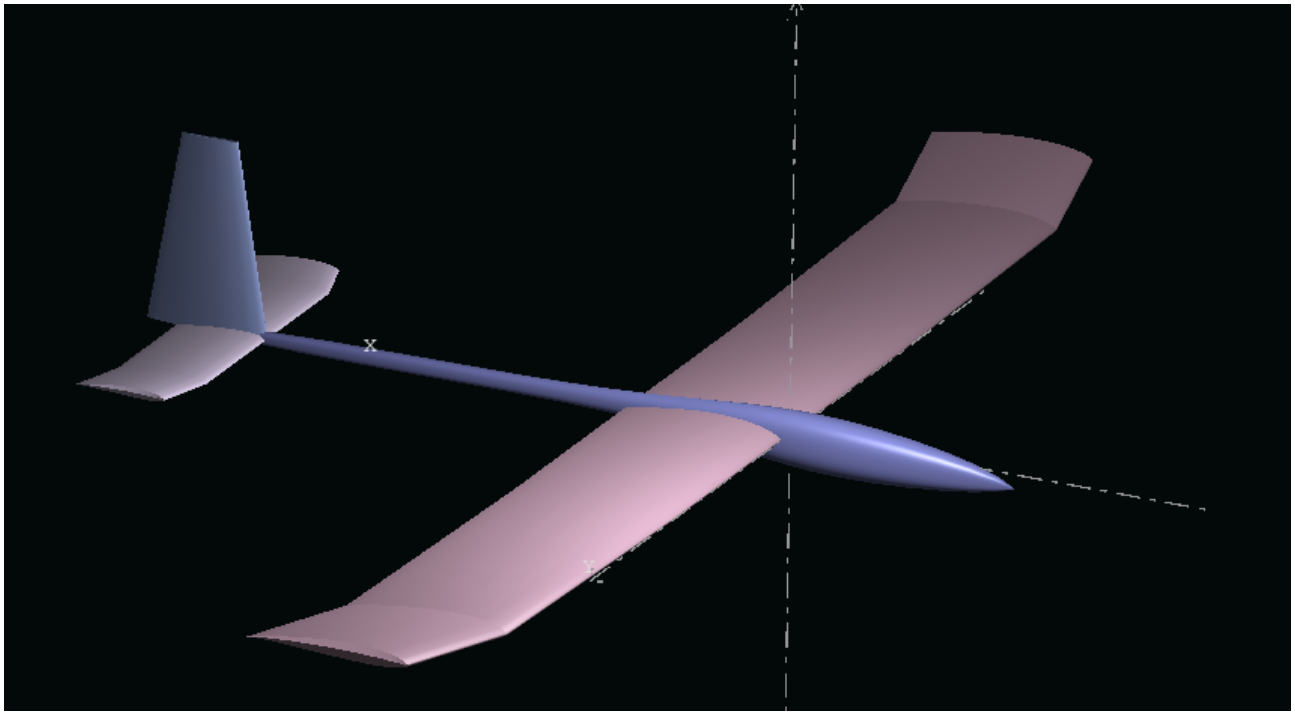
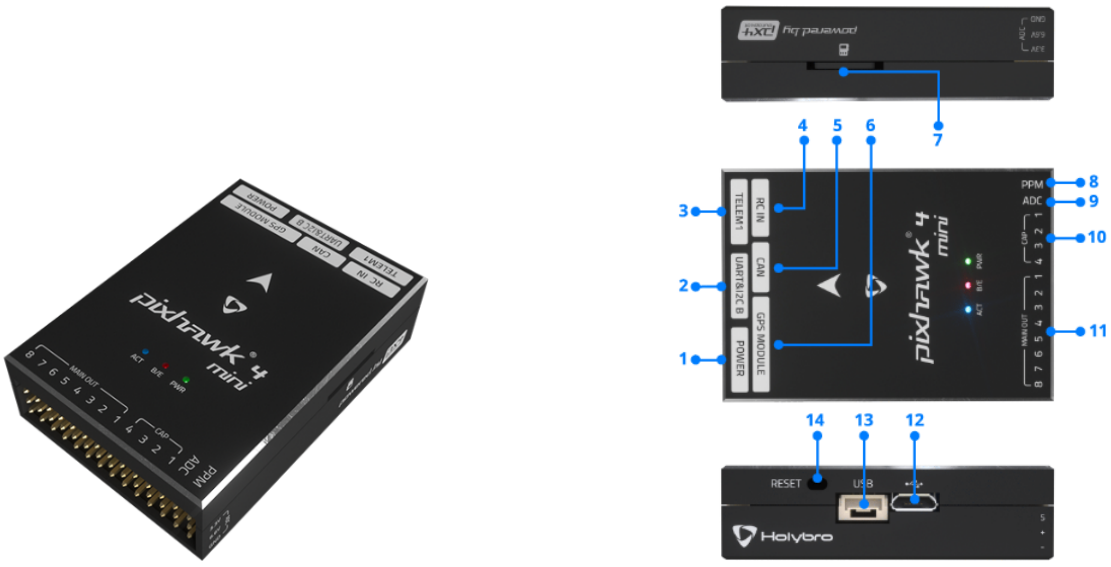


Figure 3.2: Easy glider 4 designed in XFOIL.



(a) (b)

Figure 3.3: (a) *Pixhawk 4 mini* flight controller. (b) Interfaces of *Pixhawk 4 mini*. 1-Power Module; 2-UART and I2C (for additional GPS); 3-Telemetry; 4-Radio Control 5-Receiver Input; 6-CAN BusM; 7-SD Card; 8-Radio Control Receiver Input; 9-ADC In; 10-PWM Input Capture; 11-Main PWM Outputs; 12-Micro-USB Outputs; 13-USB; 14-Reset Button [20].

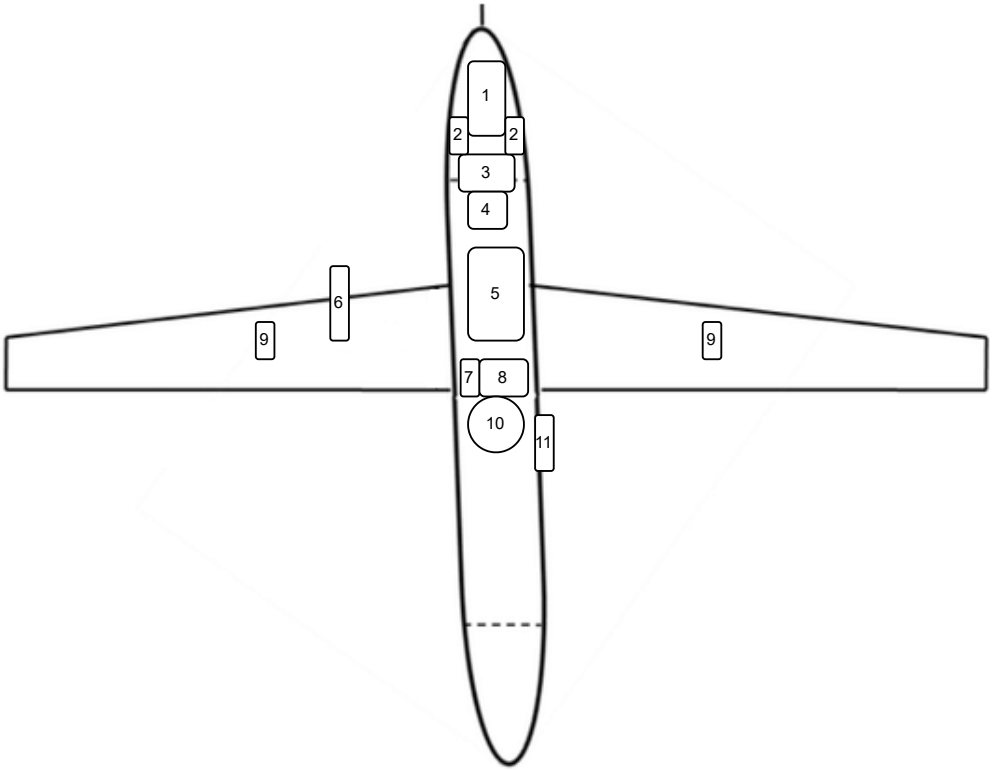


Figure 3.4: Hardware mount schematic. 1-Lippo battery, 2-Elevator and rudder servos, 3- 5V BEC, 4- Pixhawk power management board, 5-Pixhawk 4 mini, 6- Pitot tube, 7-Airspeed sensor module, 8-X8R radio module, 9- Nano servos for aileron control, 10-Pixhawk GPS module, 11-Pixhawk telemetry module.

As our UAV is small in size, we used the *Pixhawk 4 mini*, which includes the *Pixhawk 4* processor and its memory capacity, while removing the interfaces that are usually not used, thus decreasing its unnecessary size. The figure 3.3 shows us the *Pixhawk 4 mini* flight controller and its interface. The easy glider came initially equipped with an ESC (Electronic Speed Controller) called *Roxxy BL Control 720 S-BEC*, with an integrated 5V BEC, and four 5V servos to move the control surfaces, which will be harnessed and connected to the Pixhawk.

3.1.1 Manual control mode

This mode is exclusively used for safety and basic testing purposes, as it allows us to verify the basic connections and power supply to the motors and servos. For this flight mode, we only need the radio and telemetry system connected to the Pixhawk, the ESC, and the servos. We will need a radio transmitter and receiver, and remote control. For this specific project, the *FrSky* radio control system was used, which will have the function of

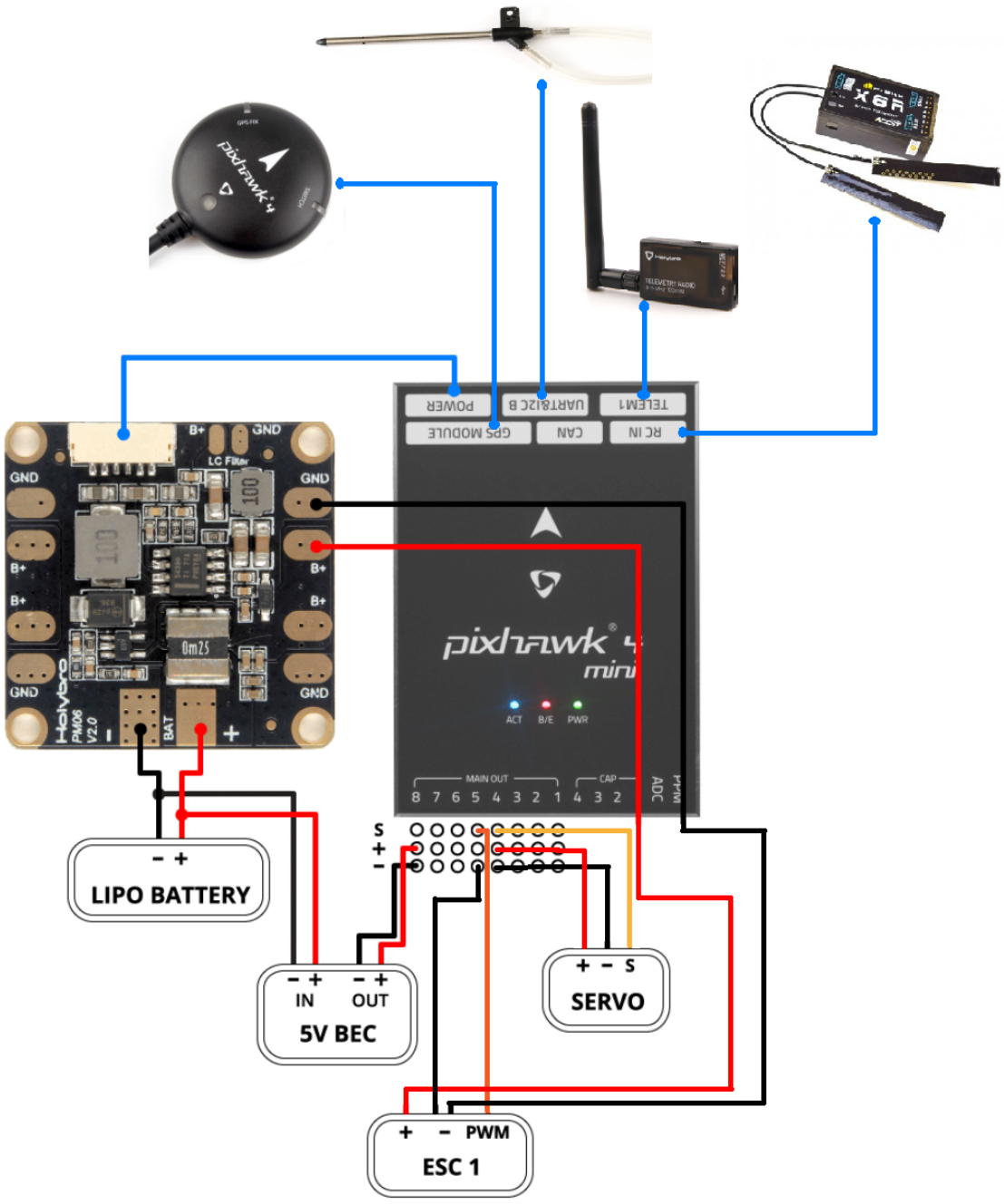


Figure 3.5: Pixhawk high level wiring schematic [20].

specifying the vehicle's movement, such as throttle, yaw, pitch and roll. This module will be connected to the **RC IN** Pixhawk pin. The flight controller interprets the commands and controls the vehicle motors and actuators accordingly. For this specific flight mode, the telemetry module, connected to the **TELEM1** pin, will be exclusively used to monitor the vehicle and arm it to start the flight, in QGroundControl software. As we can see in Figure 3.5, we need to connect the Pixhawk to the power management board (PMB),

which functions as both a power module and a power distribution board, giving power to the ESC. The PMB is powered with an 11V Lipo battery.

3.1.2 Autonomous flight mode

To carry out an autonomous flight, Pixhawk will have to receive information about the path it will have to follow, as well as the settings it will have to follow, through the Qground Control software. The UAV can also operate in Off-Board control mode, which allows us to control the drone using an external controller, running on a computer. In both cases, Pixhawk will communicate with the ground computer through the telemetry module, rep, using the MAVLink protocol. Vehicle stabilization and navigation control will be done using peripheral modules, as shown in figure 3.5. The *Pixhawk 4 mini* is integrated with a gyroscope/accelerometer, a magnetometer and a barometer, therefore it is extremely important that the controller is aligned with the body frame axis. Then it is necessary to connect the GPS module, integrated with a compass, to the **GPS MODULE** input, to provide information about the spatial location of the UAV. For fixed-wing drones, airspeed sensors are strongly recommended. They are critical since the autopilot has no other way of detecting stall condition. For fixed-wing flying, airspeed, not ground speed, ensures lift. The pitot tube-based airspeed, communicates with the Pixhawk via the I2C protocol, and is therefore connected to the Pixhawk's **UART I2C** input. The servos and the PWM signal from the ESC are connected to the *MAIN OUT* pins in the following order:

- **MAIN1**: aileron
- **MAIN2**: aileron
- **MAIN3**: elevator
- **MAIN4**: rudder
- **MAIN5**: throttle
- **MAIN8**: 5V BEC

In order to provide a power supply to the servos, it is necessary to connect a 5V BEC, which is integrated into the ESC, to the **MAIN OUT** pins. This module operates in 2 modes: 5V output and 6V output. In this application, 5V is sufficient.

Figure 3.6: Up view of the easy glider with all the hardware assembled.

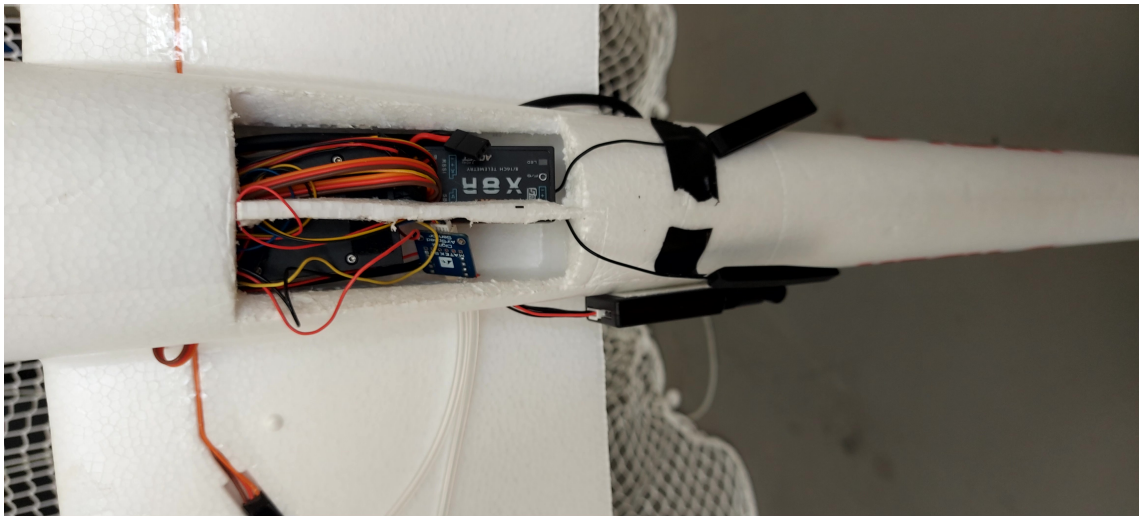


Figure 3.7: Down view of the easy glider with all the hardware assembled.

3.2 Tests

3.2.1 Manual flight

In order to validate the instrumentation effort, a basic remotely piloted flight test was performed. The goal was to validate the general assembly , namely the connection of the servos, the propeller and the telemetry and radio communication mechanisms, through the acquisition of data and control, respectively. In Figure 3.9 we can see the GPS signal referring to the trajectory that the UAV carried out in manual mode and in Figure 3.10 the temporal evolution of the airspeed, thus validating the correct connection of the GPS and the wind sensor. In Figures 3.11 and 3.12 are shown the temporal evolution of the attitude and the control signals, respectively. Both by analyzing the flight data and by the pilot's input, the vehicle was considered ready for the next stages of development, namely,

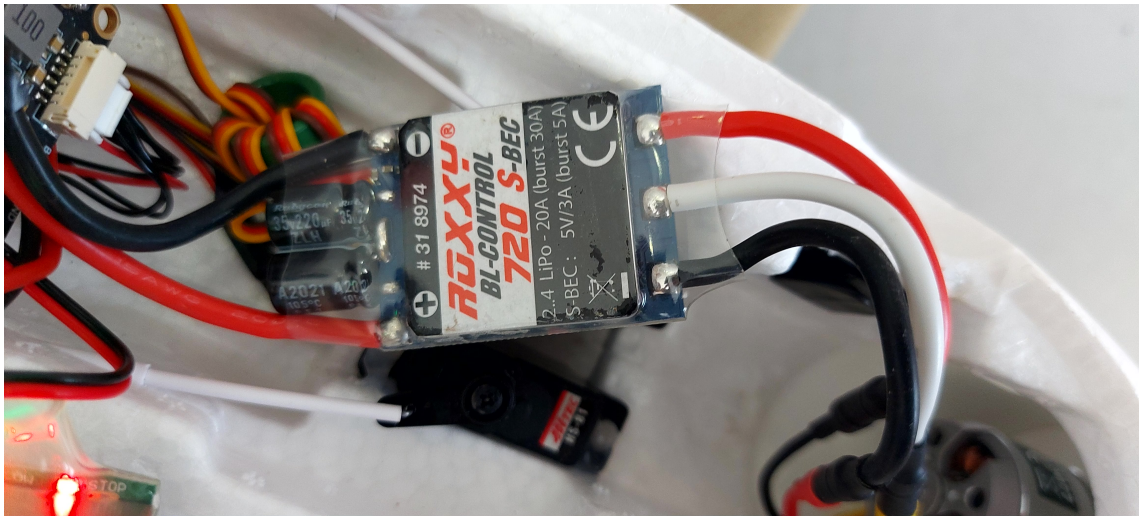


Figure 3.8: ROXXY BL-Control 720 S-BEC Model aircraft brushless motor controller Load (max.): 30 A.

Figure 3.9: GPS track signal.

the execution of pre-programmed missions and external control of other techniques.

3.2.2 Autonomous flight

Once it has been confirmed that the vehicle is well assembled through manual flight, an autonomous flight is tested. Mission mode instructs the vehicle to carry out a specified autonomous mission (flight plan) that was uploaded to the flight controller. A Ground Control Station is used to create and upload the mission. For security reasons, the PX4

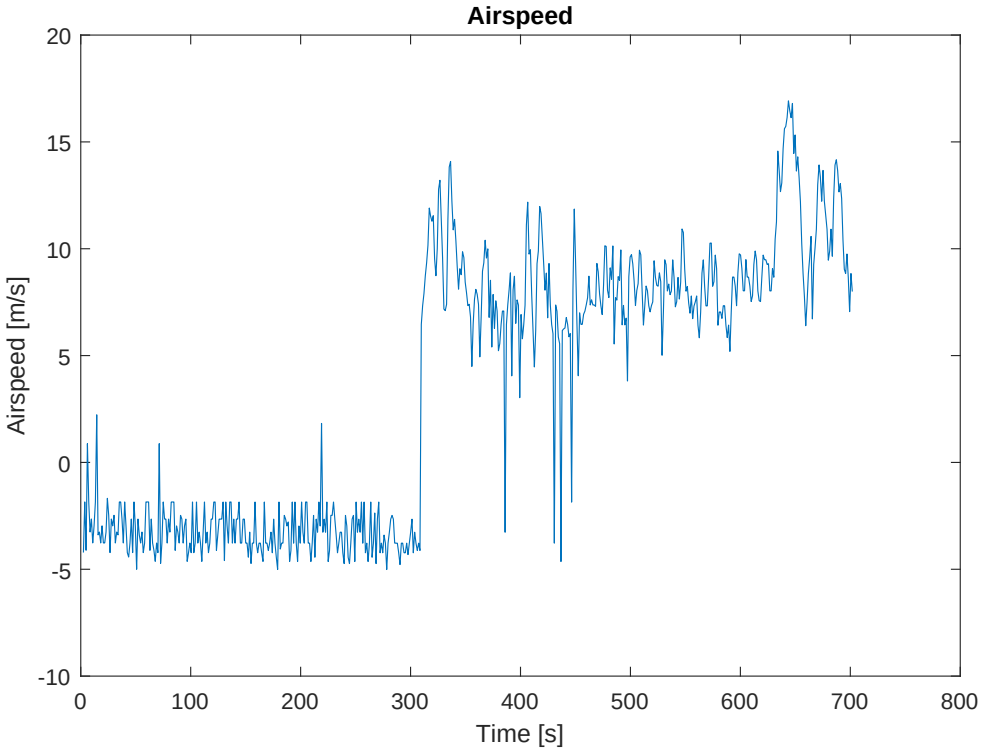


Figure 3.10: Temporal evolution of airspeed in m/s in manual mode.

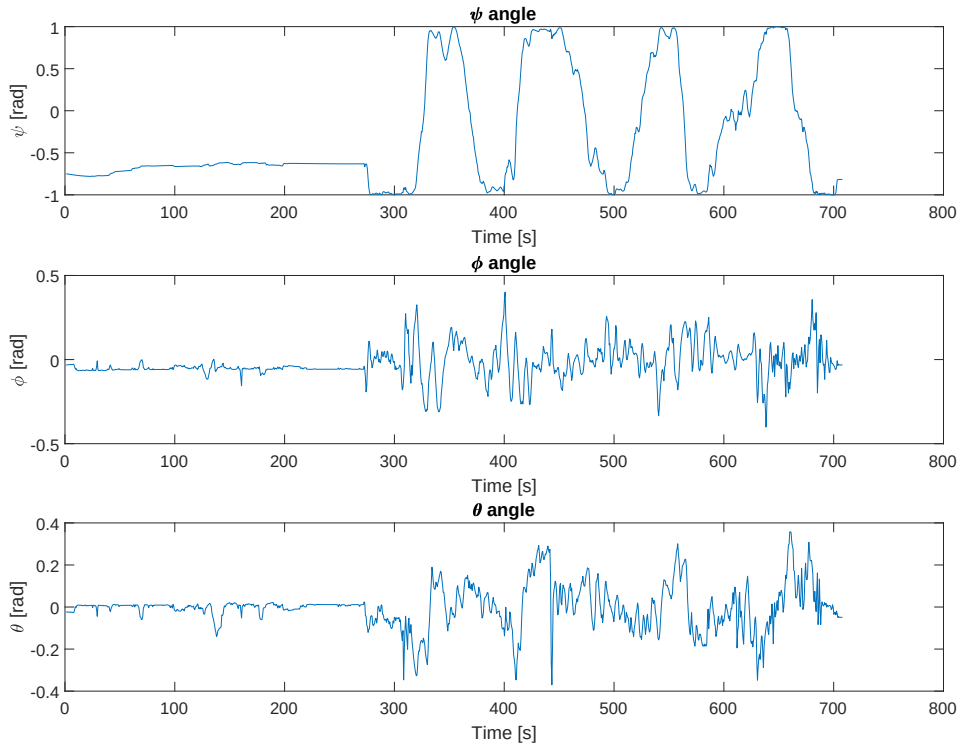


Figure 3.11: Temporal evolution of attitude in manual mode.

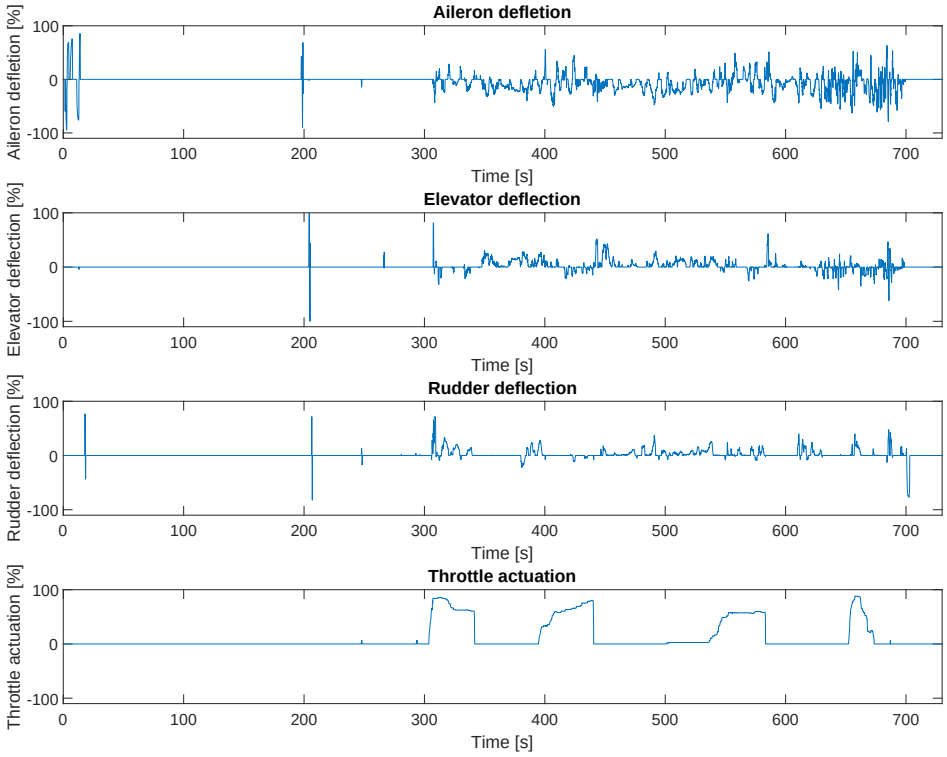


Figure 3.12: Temporal evolution of actuator deflection in manual mode.

was configured to be able to operate in two different modes: in manual mode and in mission mode. The mission is monitored and commanded through the QGround Control, which communicates with the PX4 using the telemetry module, using the MAVLINK protocol. In Figure 3.16 we can see the mission that was planned and the UAV tracking the desired path. The UAV does the launch and the take off in manual mode, then switched to mission mode. To make the landing, it is necessary to first move to the position marked with the number 10, in order to position itself for the landing area marked as orange, where the UAV loses altitude, and green, where the UAV lands.

In conclusion, this chapter has described the hardware configuration and assembly required for the UAV to perform both autonomous and manual flights. The PX4 open source hardware project was used as the autopilot for the Easy Glider 4 UA. The manual control mode was used for safety and basic testing purposes and required a radio transmitter and receiver, telemetry module, and power management board. The autonomous flight mode required communication with the Qground Control software, peripheral modules for stabilization and navigation, and a GPS module with compass and airspeed sensors. All of these components were necessary for the successful completion of the flight test.

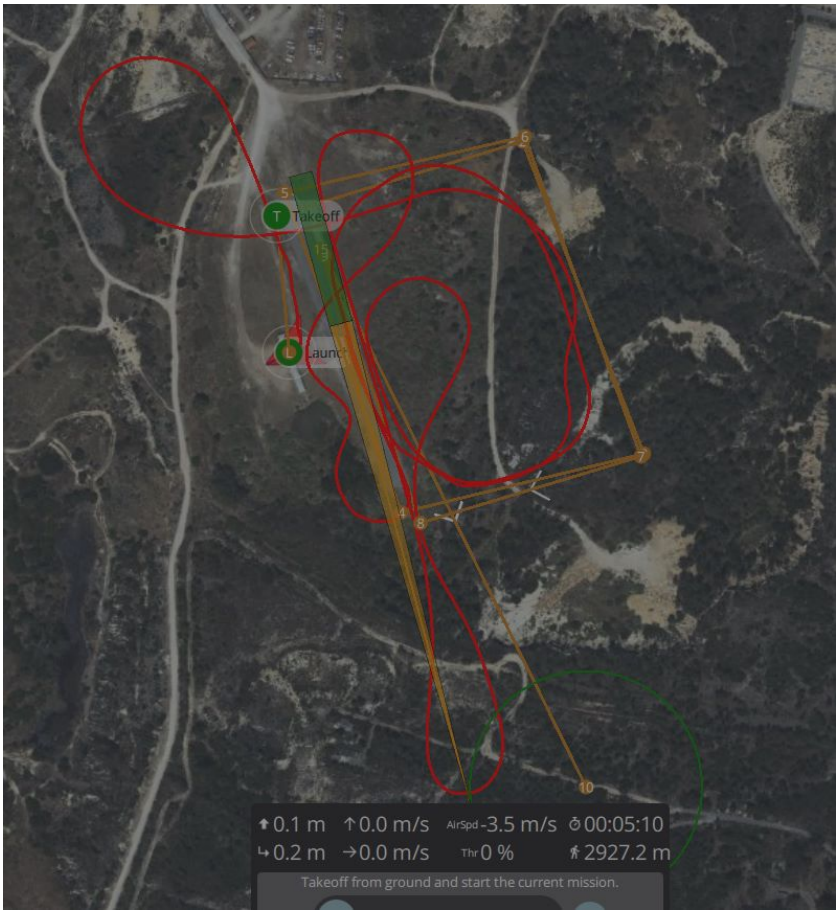


Figure 3.13: UAV performing the planned mission in QGround Control.

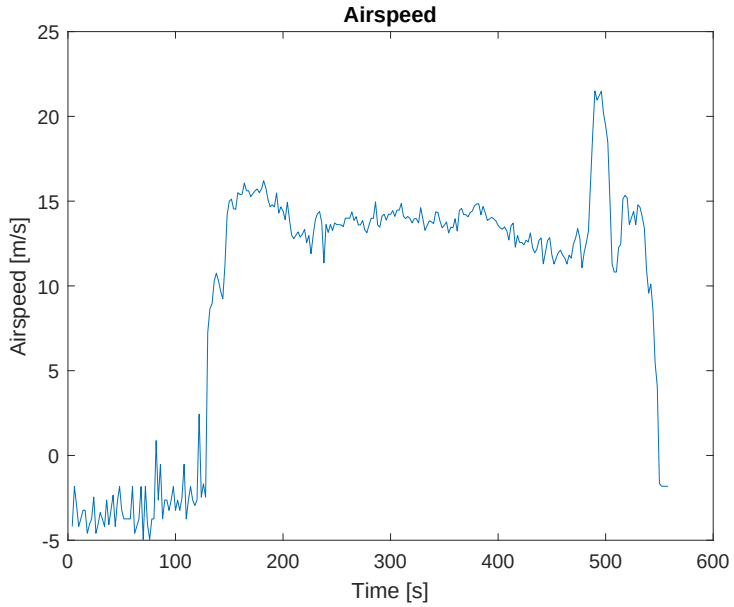


Figure 3.14: Temporal evolution of airspeed in m/s in mission mode.

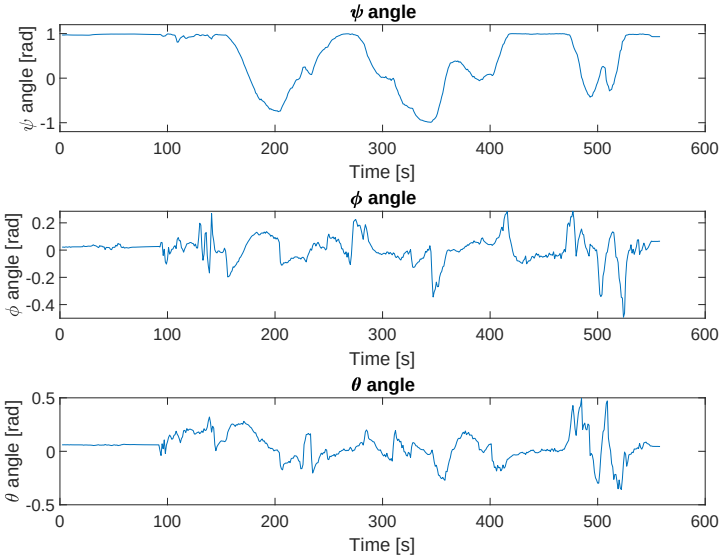


Figure 3.15: Temporal evolution of attitude in mission mode.

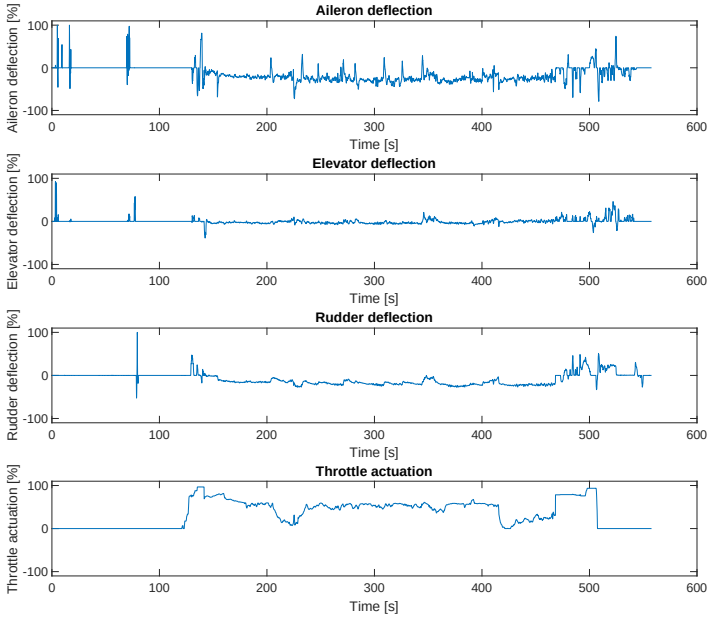


Figure 3.16: Temporal evolution of actuator deflection in mission mode.

4

Modeling

In this chapter, we will discuss the modeling of aerodynamic forces and moments in UAVs as well as the methods for obtaining the aerodynamic coefficients.

4.1 Kinematics and non-linear dynamics

Before we start developing the aircraft dynamic model and its subsequent control, we need to define the state variables, described in Table 4.1. The position of the aircraft relative to the inertial frame is defined as p_n, p_e, and p_d. The p_d axis is pointing downwards, and thus, we use $h = -p_d$ to refer to the altitude [5]. The inertial linear velocities measured along the body frame are u, v and w, which can be calculated in function of the aircraft position in the inertial frame with the expression

$$\begin{bmatrix} \dot{p}_n \\ \dot{p}_e \\ \dot{p}_d \end{bmatrix} = \begin{bmatrix} c_\theta c_\psi & s_\phi s_\theta c_\psi - c_\phi s_\psi & c_\phi s_\theta c_\psi \\ c_\theta s_\psi & s_\phi s_\theta s_\psi + c_\phi c_\psi & c_\phi s_\theta s_\psi - s_\phi c_\psi \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix}. \tag{4.1}$$

We cannot write the angular velocity of the body frame p, q, and r as direct derivatives of the Euler angles, because they are not described in the same coordinate frames. Euler angles are rotations from the vehicle frame to the body frame and the angular velocities p, q, and r are relative to the body frame. Those two vectors are related through the transformation matrix

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & -\cos\phi\sin\theta \\ 0 & 1 & \sin\theta \\ \sin\theta & 0 & \cos\phi\cos\theta \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{4.2}$$

And by inverting the equation 4.1 we have

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}. \tag{4.3}$$

Table 4.1: Stable variables

Variable	Description
p_n	Inertial north position of the UAV along I_x in inertial frame
p_e	Inertial east position of the UAV along I_y in inertial frame
p_d	Inertial down position (negative of altitude) of the UAV along I_z in inertial frame
u	Body frame velocity measured along B_x
v	Body frame velocity measured along B_y
w	Body frame velocity measured along B_z
ϕ	Roll angle defined with respect to velocity frame
θ	Pitch angle defined with respect to velocity frame
φ	Yaw angle defined with respect to velocity frame
p	Roll rate measured along B_x in body frame
q	Pitch rate measured along B_y in body frame
r	Yaw rate measured along B_z in body frame

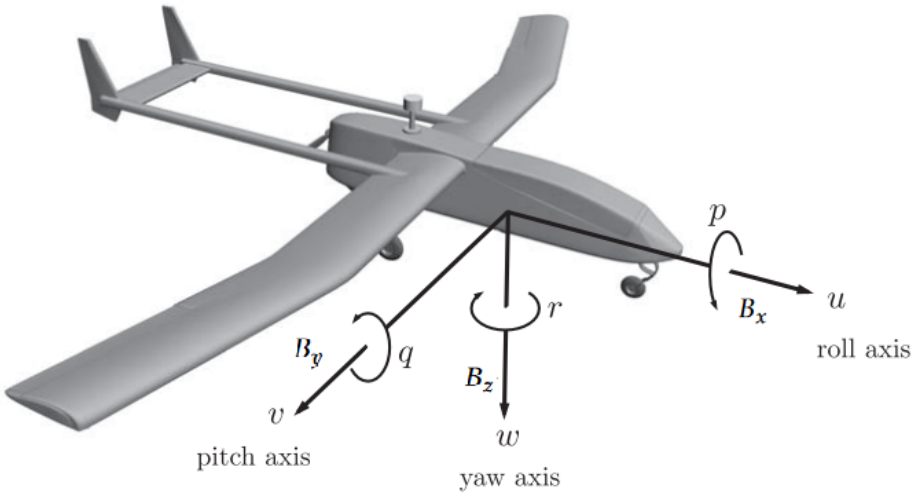


Figure 4.1: Angular rates p, q, and r are represented in the body frame [5].

We will use Newton's second law to obtain the dynamic equations of motion for the UAV. Newton's laws apply in inertial reference frames, which means that the motion of the aircraft must be referenced to the inertial frame [5]. The Newton's law applied to a body that is moving in an inertial frame is

$$m\frac{dV_g}{dt} = f. \tag{4.4}$$

The derivative of a vector $\mathbf{p}$ that is moving in a referential that is rotating relative to the inertial frame is given by

$$\frac{d}{dt_i}\mathbf{p} = \frac{d}{dt_b}\mathbf{p} + \omega_{b/i} \times \mathbf{p} \tag{4.5}$$

which $\omega_{b/i}$ represents the angular velocity of frame F^b in the inertial frame F^i. By applying the equation 4.1 in results

$$m(\frac{dV_g^b}{dt_b} + \omega_{b/i}^b \times V_g^b) = f^b. \tag{4.6}$$

By knowing that $V_g^b = (u, v, w)^T$, $\omega_{b/i}^b = (p, q, r)^T$, f^b represents all the external forces applied in the UAV, and expanding all the cross products, we finally get

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \begin{bmatrix} rv - qw \\ pw - ru \\ qu - pv \end{bmatrix} + \frac{1}{m} \begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix}. \tag{4.7}$$

The Newton's law applied to a body that is rotating in an inertial frame is

$$\frac{da}{dt_i} = m \tag{4.8}$$

where **a** represents the angular moment and **m** is the sum of all external forces applied to the body's center of mass [5]. By expanding the derivative term with the equation 4.1, results

$$\frac{da^b}{dt_b} + \omega_{b/i}^b \times a^b = m^b \tag{4.9}$$

where a^b is the angular moment, obtained by multiplying the inertia matrix **J** by the angular velocity vector $\omega_{b/i}^b$. For a symmetrical body around the x-axis, like an aircraft is, the matrix **J** is given by

$$J = \begin{bmatrix} \int y^2 z^2 \, dm & 0 & \int xz \, dm \\ 0 & \int x^2 z^2 \, dm & 0 \\ -\int xz \, dm & 0 & \int z^2 y^2 \, dm \end{bmatrix}. \tag{4.10}$$

Moments of inertia are measurements of an aircraft's tendency to resist acceleration about a given axis of rotation. As an illustration, The greater the value of J_x, the greater the aircraft's resistance to angular acceleration about the x axis [5]. Once again, assuming $a^b = J\omega_{b/i}^b$ and deviate the expression 4.1, the angular acceleration of the body frame results in

$$\dot{\omega}_{b/i}^b = J^{-1}[-\omega_{b/i}^b \times (J\omega_{b/i} b + m^b)]. \tag{4.11}$$

The inverse matrix of J is given by

$$J^{-1} = \begin{bmatrix} \frac{J_z}{\Gamma} & 0 & \frac{J_{xz}}{\Gamma} \\ 0 & \frac{1}{J_y} & 0 \\ \frac{J_{xz}}{\Gamma} & 0 & \frac{J_x}{\Gamma} \end{bmatrix} \tag{4.12}$$

where $\Gamma = J_x J_z - J_{xz}^2$ [5]. Putting m^b in a vector form $m^b = (l, m, n)^T$ as all moment components that act on each axis of the body frame, results

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \Gamma_1 pq - \Gamma_2 qr + \Gamma_3 l + \Gamma_4 n \\ \Gamma_5 pr - \Gamma_6 (p^2 - r^2) + \frac{1}{J_y} m \\ \Gamma_7 pq - \Gamma_1 qr + \Gamma_4 l + \Gamma_8 n \end{bmatrix} \tag{4.13}$$

where

$$\begin{aligned} \Gamma_1 &= \frac{J_{xz}(J_x - J_y) + Jz}{\Gamma} \\ \Gamma_2 &= \frac{J_z(J_z - J_y) + J_{xz}^2}{\Gamma} \\ \Gamma_3 &= \frac{J_z}{\Gamma} \\ \Gamma_4 &= \frac{J_{xz}}{\Gamma} \\ \Gamma_5 &= \frac{J_z - J_x}{J_y} \\ \Gamma_6 &= \frac{J_{xz}}{J_y} \\ \Gamma_7 &= \frac{(J_x - J_y)J_x + J_{xz}^2}{\Gamma} \\ \Gamma_8 &= \frac{J_x}{\Gamma}. \end{aligned}$$

4.2 Forces and Moments

This topic's goal is to explain the forces and moments that act on a UAV. We'll assume that the forces and moments (pitch moment m, roll moment l and yaw moment n) are predominantly attributable to three sources: gravity F_g, aerodynamics F_a, and propulsion F_p. The aerodynamic forces are Lift Force F_{lift}, Drag Force F_{drag} and Lateral Force F_Y. The gravitational force acts in the I_z axis of the aircraft. Applying Newton's law in the frame F^v we have

$$F_g^v = \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix}. \tag{4.14}$$

Resolving this equation in the body frame, we apply the transformation

$$F_g^v = R_v^b \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} = \begin{bmatrix} -mgsin\theta \\ mgcos\theta sin\phi \\ mgcos\theta cos\phi \end{bmatrix}. \tag{4.15}$$

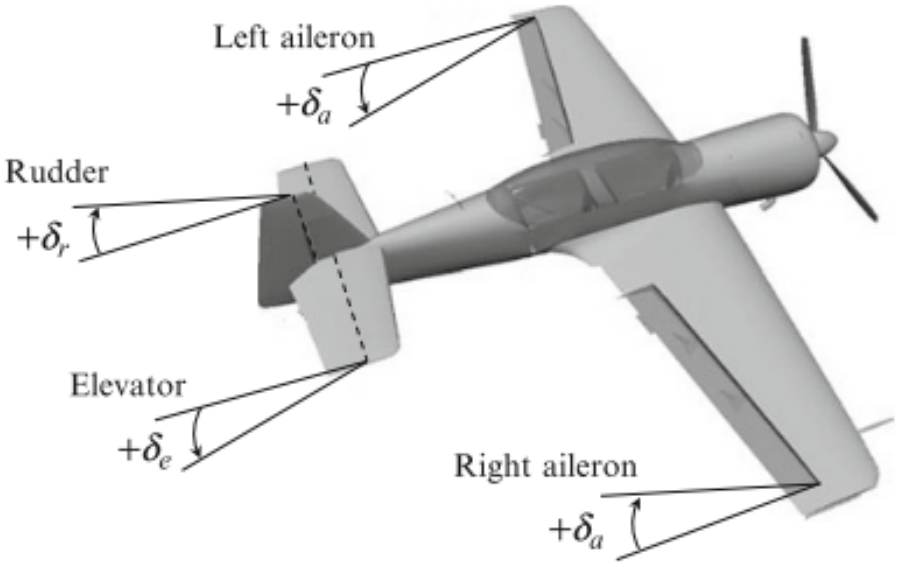


Figure 4.2: Control surfaces of an aircraft [7].

Aerodynamic forces and moments are determined by the aircraft's contact with the airflow, which is a function of the shape of its surface, and its attitude and also depends on the velocity relative to the surrounding air [7]. The surfaces shape can be modified by control surfaces, which usually are aileron, rudder and elevator. The aileron's deflection δ_a controls the roll angle ϕ, the rudder deflection δ_r controls the yaw angle ψ and the elevator's deflection δ_e controls the pitch angle θ. Although these angles cannot be controlled independently, which means, for instance, that a rudder deflection can affect the roll angle. The right-hand rule is applied to the rotation axis of a control surface to determine its positive deflection. Figure 4.2 also shows the positive direction of aileron, elevator, and rudder deflections.

The UAV dynamics can be divided into two parts: longitudinal and lateral. The lateral dynamics are those which are affected by the sides-slip angle β variations, along the B_y axis causing changes in the yaw and roll velocities and subsequent moving the symmetry axis of the aircraft. When an aircraft is disturbed in a roll the lift tilts to the side with the aircraft. The tilted lift vector causes the aircraft to sideslip in direction of the roll, making it subjected to a sideway component of the relative airflow. As the aircraft sideslips, the lower swept back wing generates more lift than the upper wing. This is because, in a side slip, the lower wing presents more wingspan to the relative airflow than the upper wing. As de lower wing generates more lift it recovers the aircraft back to the wing's level altitude. We say that an aircraft is lateral stable if it rolls back beyond its wings level altitude in a sequence of decreasing oscillations, and directional stable if an aircraft has a natural ability to recover from a disturbance in the yawing plane. Sometimes, lateral and directional stability act in conflict with each other. A dutch roll happens when lateral stability is strong and directional is weak. For example, if an aircraft rolls and sideslips to

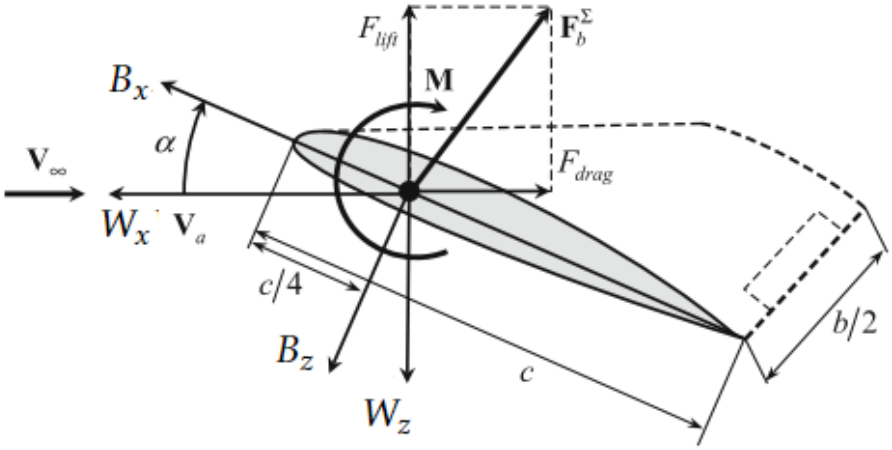


Figure 4.3: Lift and Drag forces and Pitching moment represented in the wind frame [7].

the right is subsequently begins to recover thanks to the strong lateral stability. However, the lower wing now generates more drag because it generates more lift. More drag on the right wing causes yaw to the right. The weaker directional stability attempts to restore the aircraft back onto its path as the aircraft overshoots in a roll to the left. The restoring forces of lateral and directional stability are out of phase with each other [21].

On the other side, longitudinal dynamics don't move the symmetry axis, causing pitch velocity in B_x and B_z by the angle of attack α variations, also known as pitch plane [24]. The lift F_{lift} and drag F_{drag} forces act in the wind frame and are applied at the aerodynamic center of the lifting surface, which is positioned at the quarter-chord point, as illustrated in 4.3 (c is the length of the mean aerodynamic chord) and the pitching moment M acts around the aerodynamic center [7]. The lift, drag and pitching moment can be expressed as

$$\begin{aligned} F_{lift} &= \frac{1}{2}\rho V_a^2 S C_L \\ F_{drag} &= \frac{1}{2}\rho V_a^2 S C_D \\ m &= \frac{1}{2}\rho V_a^2 S c C_m \end{aligned} \tag{4.16}$$

being S the wing's area, b the wing pan and C_L, C_D and C_m nondimensional aerodynamic coefficients which are nonlinear functions of the angle of attack α[5]. These aerodynamic coefficients can be obtained by interpolation of data for a small range of α around 0 using XFOIL and semi-empirical methods for higher angles of attack, like wind tunnel [10]. The nonlinear set of equations 4.2 can be linearized with Taylor Series and rewritten as follows:

$$
\begin{aligned}
F_{drag} &= \frac{1}{2}\rho V_a^2 S(C_{D_0}(\alpha) + C_D^{\alpha}\alpha + C_D^{q}\frac{c}{2V_a}q + C_D^{\delta_e}\delta_e) \\
F_{lift} &= \frac{1}{2}\rho V_a^2 S(C_{L_0}(\alpha) + C_L^{\alpha}\alpha + C_L^{q}\frac{c}{2V_a}q + C_L^{\delta_e}\delta_e) \\
M &= \frac{1}{2}\rho V_a^2 Sc(C_{M_0}(\alpha) + C_M^{\alpha}\alpha + C_M^{q}\frac{c}{2V_a}q + C_M^{\delta_e}\delta_e).
\end{aligned} \tag{4.17}
$$

The same can be done for the lateral force and moments, as follows

$$
\begin{aligned}
F_{side} &= \frac{1}{2}\rho V_a^2 S(C_{Y_0}(\alpha) + C_Y^{\beta}\beta + C_Y^{p}\frac{b}{2V_a}p + C_Y^{r}\frac{b}{2V_a}r + C_Y^{\delta_r}\delta_r + C_Y^{\delta_a}\delta_a) \\
L &= \frac{1}{2}\rho V_a^2 Sb(C_{l_0}(\alpha) + C_l^{\beta}\beta + C_l^{p}\frac{b}{2V_a}p + C_l^{r}\frac{b}{2V_a}r + C_l^{\delta_r}\delta_r + C_l^{\delta_a}\delta_a) \\
N &= \frac{1}{2}\rho V_a^2 Sb(C_{n_0}(\alpha) + C_n^{\beta}\beta + C_n^{p}\frac{b}{2V_a}p + C_n^{r}\frac{b}{2V_a}r + C_n^{\delta_r}\delta_r + C_n^{\delta_a}\delta_a).
\end{aligned} \tag{4.18}
$$

The forces and moments calculated when all states, including the control surface deflection, are zero are denoted by the coefficients with zero in the subscript. The stability derivatives are defined with respect to states $[\alpha, \beta, p, q, r]$, whereas the control derivatives are specified with regard to controls $[\delta_a, \delta_r, \delta_e]$, according to a standard naming system. Finally, yet importantly, we have to have into consideration the forces and moments produced by the propeller. The thrust created by a propeller can be modeled by using Bernoulli's principle [5]. The thrust force produced by the propeller is then

$$
T = 2\rho A(V_a + V_i)V_i \tag{4.19}
$$

where V_i is the induced velocity by the propeller [9]

$$
V_i = \frac{1}{2}(nD_p\sqrt{C_T}) \tag{4.20}
$$

being V_a the propeller forward speed, n the rotational speed, D_p the propeller's diameter and C_T the thrust coefficient, determined by experience. The moment due to the propulsion system is

$$
m_p = \begin{bmatrix} 2\rho A(V_a + V_i)V_i \\ 0 \\ 0 \end{bmatrix}. \tag{4.21}
$$

4.3 Aerodynamic analysis

The next step is to obtain the aerodynamic coefficients as a function of the wind frame angles α and β and the deflections of the rudder δ_r, the elevator δ_e and aileron δ_a, using XFLR software. The XFLR uses the XFOIL engine, which can calculate the pressure distribution on a 2D airfoil given the coordinates specifying the shape, Reynolds and Mach numbers. Two types of analysis were performed: the first to obtain the coefficients associated with the longitudinal dynamics, varying the angle of attack α, and the other

Figure 4.4: Coefficient of lift in function of alpha. Grey line- negative elevator deflection, Blue line- neutral configuration , Green line - positive elevator deflection.

associated with the lateral dynamics, varying the β angle, both at a fixed speed with all combinations of elevator, rudder and aileron deflections.

Looking at the figures 4.4, 4.5 and 4.6 we can see that in each graph we have 3 curves that represent the aerodynamic response for all the configurations of the elevator deflection. If we examine the CL-alpha plot, we can see the lift force increased as we deflect the elevator positively and the pitching moment Cm decreases, as expected. Next, the same procedure was done, but this time varying the beta angle in order to obtain the lateral dynamic response, for all rudder deflection combinations, which are again: neutral, positive deflection and negative deflection.
The lateral aerodynamic response is graphically represented in figures 4.7 to 4.12, as lateral, yaw and roll, for rudder and aileron reflections, respectively. As we can see, the roll moment coefficient is relatively small when compared to the yaw moment, however it is not zero, confirming that the control of surfaces is not exclusive to its corresponding movement. The graphs are expected to be linear, as they are computed at low angles, however, as weight was added to the UAV in a manner not programmed by the manufacturer, its aerodynamic response is quite abnormal.

To conclude, understanding the forces and moments acting on an aircraft is crucial for aircraft design and control. The mathematical models presented in this chapter provide a way to analyze these forces and moments, and their effect on the aircraft's stability and control.

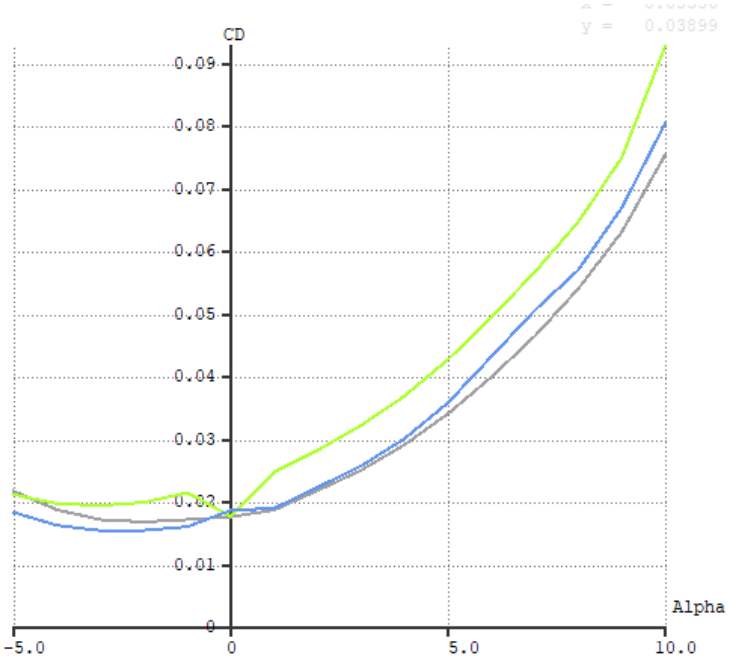


Figure 4.5: Coefficient of drag in function of alpha. Grey line- negative elevator deflection, Blue line- neutral configuration, Green line - positive elevator deflection.

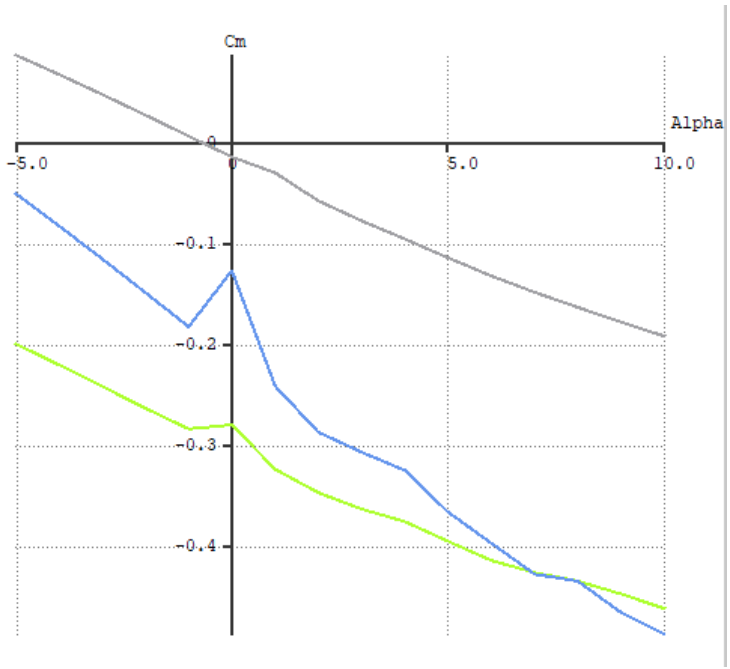


Figure 4.6: Coefficient of pitching moment in function of alpha. Grey line- negative elevator deflection, Blue line- neutral configuration, Green line - positive elevator deflection.

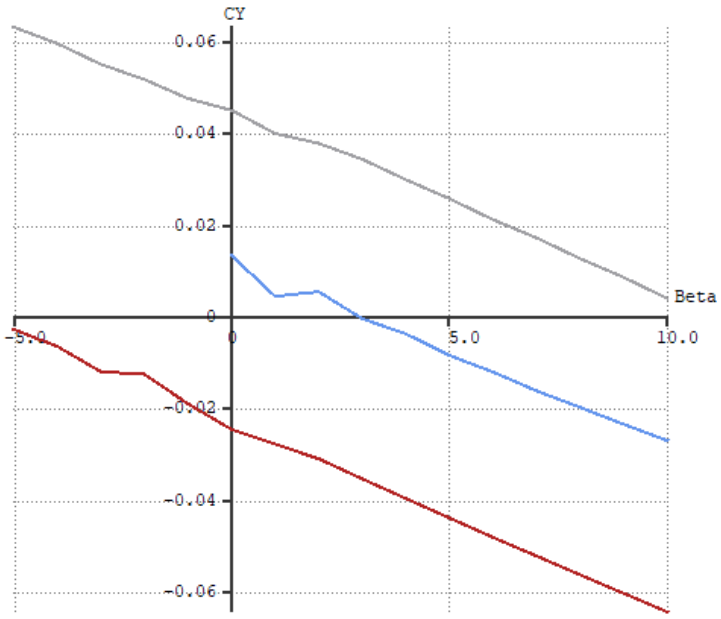


Figure 4.7: Lateral coefficient in function of beta. Red line- negative rudder deflection, Blue line- neutral configuration, Grey line - positive rudder deflection.

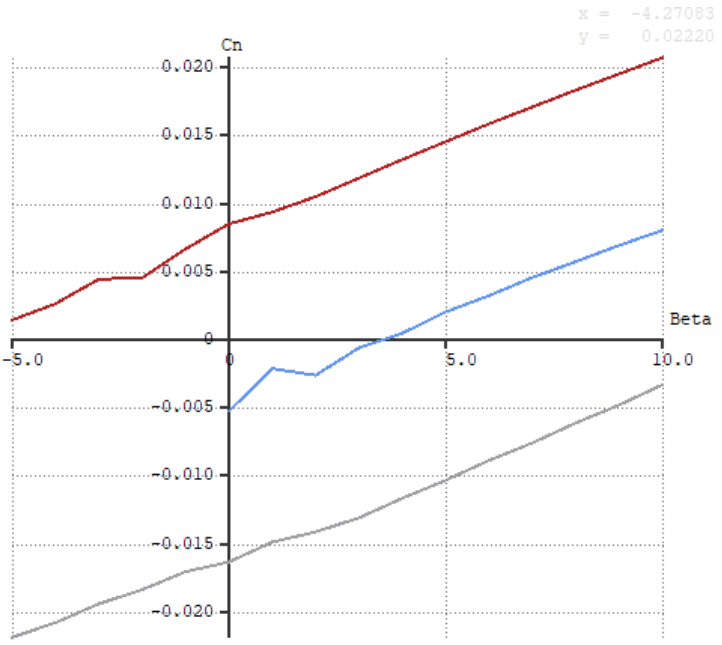


Figure 4.8: Yaw moment coefficient in function of beta. Grey line- negative rudder deflection, Blue line- neutral configuration, Red line - positive rudder deflection.

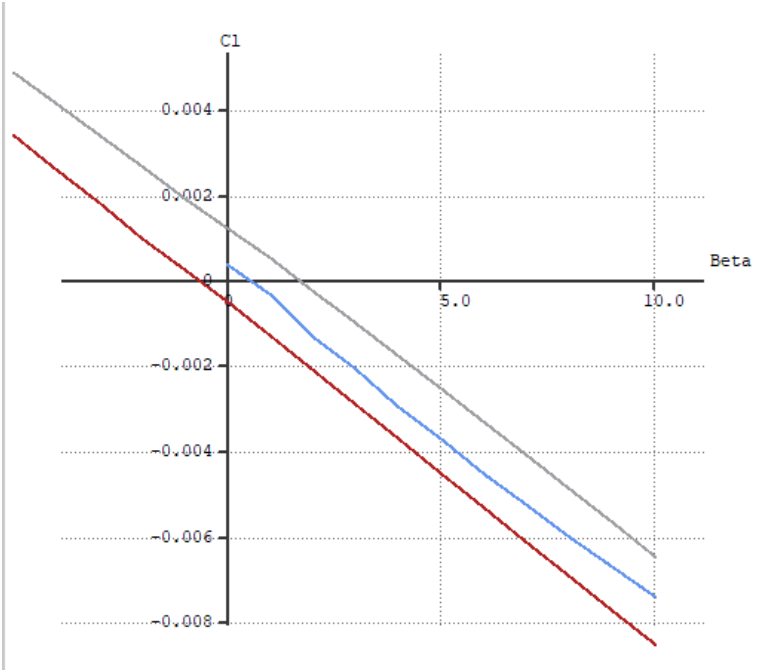


Figure 4.9: Roll moment coefficient in function of beta. Grey line- negative rudder deflection, Blue line- neutral configuration, Red line - positive rudder deflection.

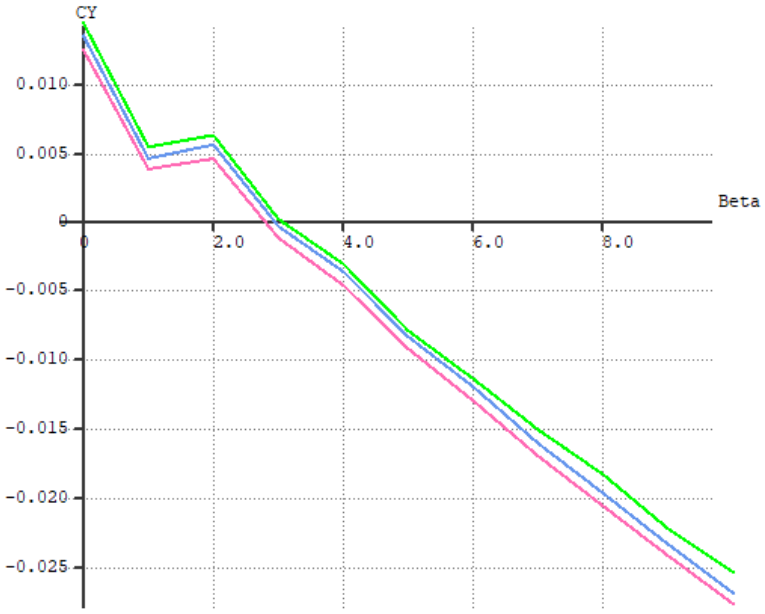


Figure 4.10: Lateral coefficient in function of beta. Pink line- negative aileron deflection, Blue line- neutral configuration, Green line - positive aileron deflection.

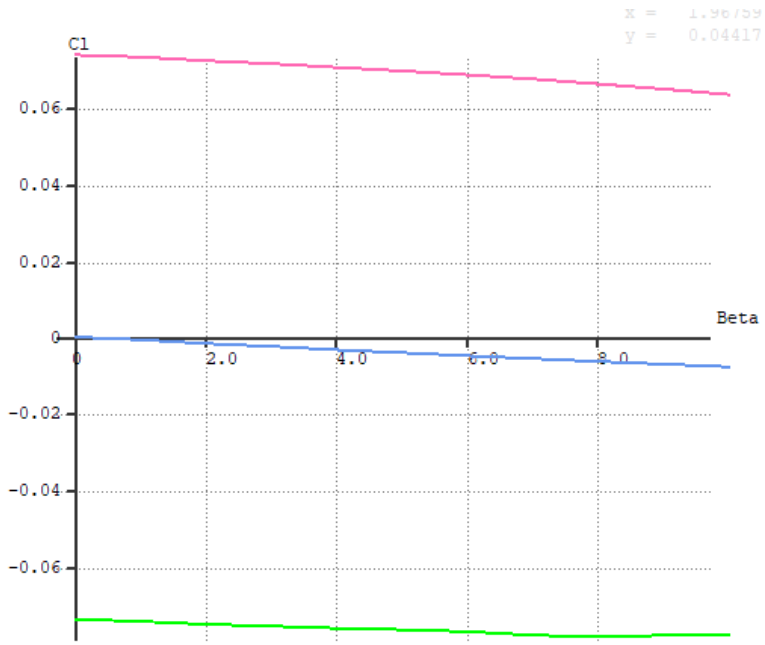


Figure 4.11: Roll moment coefficient in function of beta. Green line- negative aileron deflection, Blue line- neutral configuration, Pink line - positive aileron deflection.

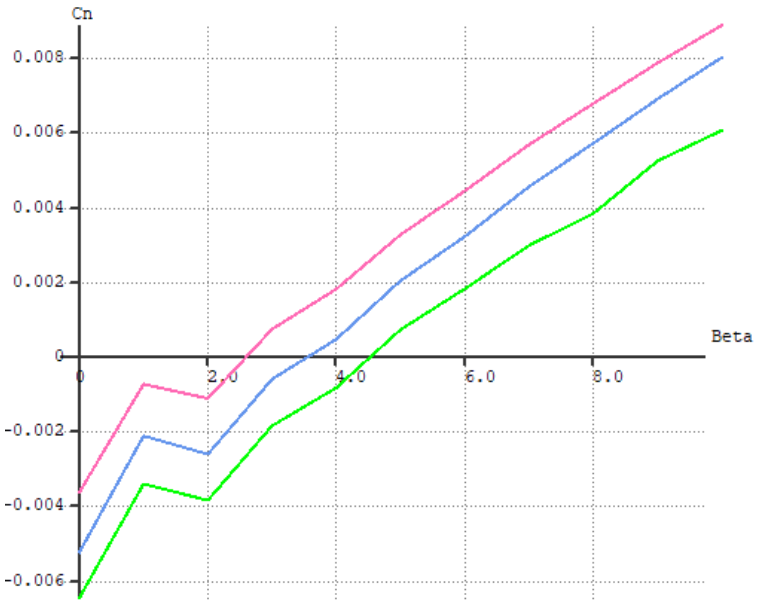


Figure 4.12: Yaw moment coefficient in function of beta. Green line- negative aileron deflection, Blue line- neutral configuration, Pink line - positive rudder deflection.

5

Control Design

This chapter aims at designing a path planning and a path following controller as well as the low-level controller, in order to control the attitude of the UAV. As shown in figure 5.1, we can see that in the outer loop we have the controller responsible for guiding the vehicle along a trajectory (path planning and path following) and in the inner loop we have the low-level attitude controller.

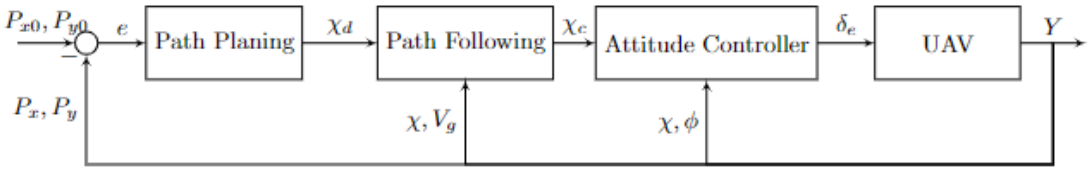


Figure 5.1: Full control system design.

5.1 Attitude Controller

The attitude controller controls the angular rates p, q and r, attitude angles ϕ, θ and ψ, by sending angular references to the control surfaces of the UAV. Two methods will be developed: a linear one (PI) and a non-linear one (Backstepping), and in chapter 6 they will be compared.

5.1.1 Backstepping

The expressions of the elevator, rudder and aileron deflections were obtained by using the backstepping control method presented in[12] with more detail. This controller can be also divided into two parts: Longitudinal controller, which is responsible to control the pitch angle θ through the elevator deflection δ_e; Lateral controller which controls the roll angle ϕ and yaw angle ψ, with the aileron deflection δ_a and rudder deflection δ_r,

respectively. With this method, the attitude angles are controlled by means of the angular rates, and the angular rates are controlled by control surfaces deflection.

5.1.1.1 Longitudinal Control

The longitudinal controller is responsible to stabilize the pitch angle θ. Starting by defining the error:

$$e_\theta = \theta - \bar{\theta} \tag{5.1}$$

with $\bar{\theta}$ being the pitch angle desired value. Our goal is to drive the pitch angle error to zero. Using Lyapunov stability method, the expression that stabilizes the first sub-system presented in figure 5.2 is

$$\bar{q} = (-\mu_\theta e_\theta + r\sin\phi)/\cos\phi \tag{5.2}$$

being μ_θ a positive constant. Next, we define the pitch rate error by :

$$e_q = q - \bar{q} \tag{5.3}$$

and repeat the Backstepping method in order to get the control law, which is given by:

$$\delta_e = \frac{I_y}{P_d S \bar{c} C_{m_{\delta_e}}}(\delta_1 - \frac{P_d S \bar{c}}{Iy}\delta_2 + \delta_3) \tag{5.4}$$

with

$$\begin{aligned}
\delta_1 &= -e_\theta cos(\phi) - \mu_q e_q \\
\delta_2 &= C_{m_0} + C_{m_\alpha}\frac{\theta - \gamma - \beta\sin\phi}{\cos\phi} + \frac{\bar{c}}{2V}C_{m_q}q \\
\delta_3 &= \frac{I_x - I_z}{I_y}pr + \frac{I_{xz}}{I_y}(p^2 - r^2) + \dot{\bar{q}}
\end{aligned} \tag{5.5}$$

being μ_q a positive constant.

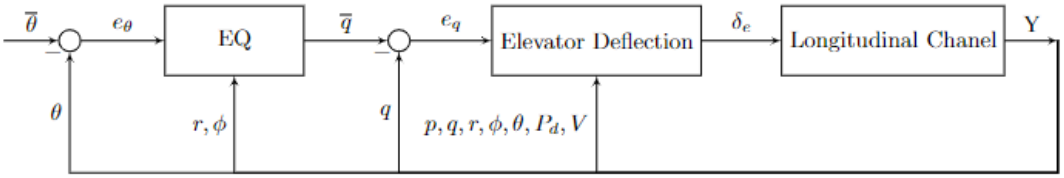


Figure 5.2: Longitudinal channel block diagram.

5.1.1.2 Lateral control

The process to project the lateral channel controller is similar to the longitudinal controller. Starting by defining the attitude errors

$$\begin{aligned} e_\phi &= \phi - \bar{\phi} \\ e_\psi &= \psi - \bar{\psi} \end{aligned} \tag{5.6}$$

Again, applying the Lyapunov stability method, the expressions that drive the errors to zero are

$$\begin{aligned} \bar{p} &= -\mu_\phi e_\phi - q\tan\theta\sin\phi - r\tan\theta\cos\phi \\ \bar{r} &= -\mu_\psi \frac{\cos\theta}{\cos\phi} e_\psi - \frac{\sin\phi}{\cos\phi} q \end{aligned} \tag{5.7}$$

being μ_ϕ and μ_ψ positive constants. Considering the second sub-system error variables

$$\begin{aligned} e_p &= p - \bar{p} \\ e_r &= r - \bar{r} \end{aligned} \tag{5.8}$$

and applying the Backstepping method, the final control law for the lateral channel, which controls the rudder and aileron deflections is:

$$\begin{aligned} \delta_r &= \frac{\tilde{A}_1 C_{n_{\delta a}} - \tilde{A}_2 C_{l_{\delta a}}}{C_{l_{\delta r}} C_{n_{\delta a}} - C_{l_{\delta a}} C_{n_{\delta r}}} \\ \delta_a &= \frac{-\tilde{A}_1 C_{n_{\delta r}} + \tilde{A}_2 C_{l_{\delta r}}}{C_{l_{\delta r}} C_{n_{\delta a}} - C_{l_{\delta a}} C_{n_{\delta r}}} \end{aligned} \tag{5.9}$$

with:

$$\begin{aligned} \tilde{A}_1 &= \frac{I_x}{P_d S b}\left(-e_\phi - \mu_p e_p + \frac{I_z - I_y}{I_x} qr - \frac{I_{xz}}{I_x} pq + \dot{\bar{p}}\right) - \left[C_{l_\beta}\beta + \frac{b}{2V}\left(C_{l_p} p + C_{l_r} r\right)\right] \\ \tilde{A}_2 &= \frac{I_z}{P_d S b}\left(-e_\psi - \mu_r e_r + \frac{I_y - I_x}{I_z} pq + \frac{I_{xz}}{I_z} qr + \dot{\bar{r}}\right) - \left[C_{n_\beta}\beta + \frac{b}{2V}\left(C_{n_p} p + C_{n_r} r\right)\right] \end{aligned} \tag{5.10}$$

being μ_p and μ_r positive constants.

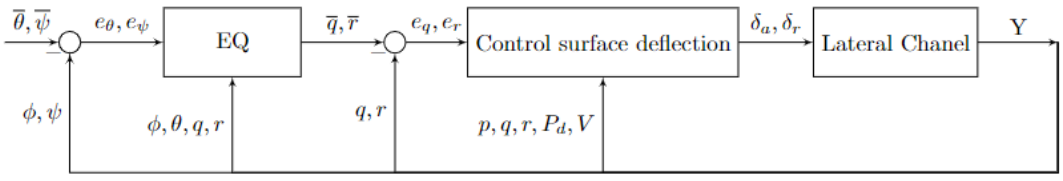
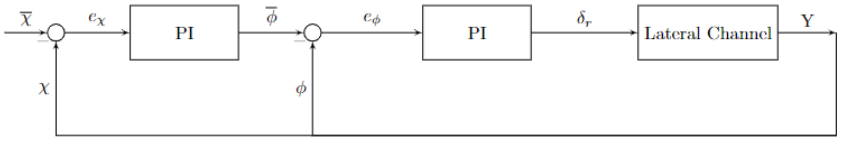


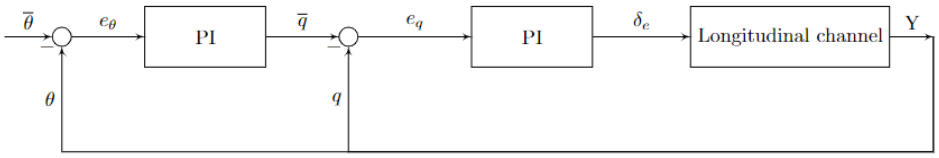
Figure 5.3: Lateral channel block diagram.

5.1.2 PI Controller

The transfer function models needed to create the flight controller system are introduced in this subsection. Fixed-wing aircraft dynamics can be roughly divided into lateral motion, as presented previously in subsection 5.1.1, which includes roll angles, and longitudinal motion, such as airspeed, pitch angle, and altitude. Using the cascaded control strategy, which divides the entire fixed-wing UAV control problem into cascaded loops, the flight controller structure can be divided into smaller control subsystems. By managing the actuator, the inner loop control's primary job is to maintain the UAV's attitude. The outer control loop sends reference signals to the inner loop control [14].



(a)



(b)

Figure 5.4: PI attitude control structure. (a) Pitch angle control. (b) Coursing angle control.

5.1.2.1 Lateral control loop design

The course angle error will be sent to a PI controller, which in turn will send a roll angle reference, whose error will be sent to another PI controller, which will send a control signal to the plant. Much of the work developed indicates that to control the heading angle through cascade control with the roll angle in the inner loop is done through the aileron. However, [3] shows us that it is possible to do it with rudder deflection. The transfer function for the Roll angle ϕ, controlled by the rudder deflection δ_r is

$$\phi(s) = \left(\frac{a_{\phi 2}}{s + (s + a_{\phi 1}}\right)\delta_r(s) \tag{5.11}$$

where,

$$\begin{aligned} a_{\phi 1} &= -\frac{1}{2}\rho V_a^2 S b C_p \frac{b}{2V_a} \\ a_{\phi 2} &= -\frac{1}{2}\rho V_a^2 S b C_{p\delta_r} \end{aligned} \tag{5.12}$$

From equation 4.3 and considering the absence of wind, we can write that

$$\dot{\chi} = \dot{\psi} = \frac{g}{V_g} \tan \phi \tag{5.13}$$

and applying Laplace transformation, we can conclude

$$\chi(s) = \frac{g}{V_a s} \phi(s). \tag{5.14}$$

5.1.2.2 Longitudinal control loop design

In order to create a longitudinal transfer function for longitudinal dynamics, variables like pitch angle θ, pitch rate q, and elevator deflection δ_e were used.

The pitch angle transfer function is given by

$$\theta(s) = \frac{a_{\theta 3}}{s^2 + a_{\theta 1} s + a_{\theta_2}} \delta_e(s) \tag{5.15}$$

where,

$$\begin{aligned} a_{\theta 1} &= -\frac{\rho V_a^2 c S}{2 J_y} C_{m_q} \frac{c}{2 V_a} \\ a_{\theta 2} &= -\frac{\rho V_a^2 c S}{2 J_y} C_{m_\alpha} \\ a_{\theta 3} &= - - \frac{\rho V_a^2 c S}{2 J_y} C_{m_{\delta_e}} \end{aligned} \tag{5.16}$$

The pitch rate expression comes as

$$q(s) = \left(\frac{a_{\theta 3} s}{s^2 + a_{\theta 1} s + a_{\theta 2}}\right) \delta_e(s). \tag{5.17}$$

5.2 Path Following

This chapter aims to develop a path-following algorithm so that the UAV follows a certain path. Unlike the trajectory tracking problem, path following makes the UAV converge on a certain trajectory without any time specification[6]. In the trajectory tracking problem, the UAV has to be at a specific point, at a certain moment, which in the presence of wind of variable amplitude or orientation, can lead the system to instability. The developed controller is based on a 2D vector field, which allows the UAV to follow lines, circles and arcs, calculating for each point in space, a vector with the desired orientation. First, the controller will be tested in the presence of constant known wind, and then an adaptive version will be developed, which will be able to operate in the presence of wind with a non-measurable orientation and amplitude. This approach considers that the altitude of the UAV and the airspeed V_a are constants.

5.2.1 Vector field

The first approach will be a simple vector field controller, used in normal wind conditions, that is, with known amplitude and direction, starting with a straight line follower and then an orbit follower. Path following laws based on vector fields are often developed from Lyapunov stability analysis, which ensures globally stable convergence to the desired path. A more detailed implementation can be found in [17].

5.2.1.1 Straight Line

First, as shown in Figure 5.1, a path planning block will be developed, which is responsible for building a vector field with the desired course angle, for each position, y. The expression for the desired course is

$$\chi^d(y) = -\chi^\infty \frac{2}{\pi} \arctan(ky) \tag{5.18}$$

where k is a positive constant that determines the rate at which the field vectors become collinear with the path as the UAV reaches the straight line. Due to the fact tan^1 is odd, we use the Lyapunov function $W_1(y) = (1/2)y^2$ to guarantee that χ^d goes to zero as y goes to zero. Given that, and assuming that $\dot{y}$ can be substituted with the dynamic equation of a generic UAV model $\dot{y} = V_g sin(\chi)$, we have:

$$\dot{W}_1 = -V_g y \sin(\chi^\infty \frac{2}{\pi} \arctan(ky)) \tag{5.19}$$

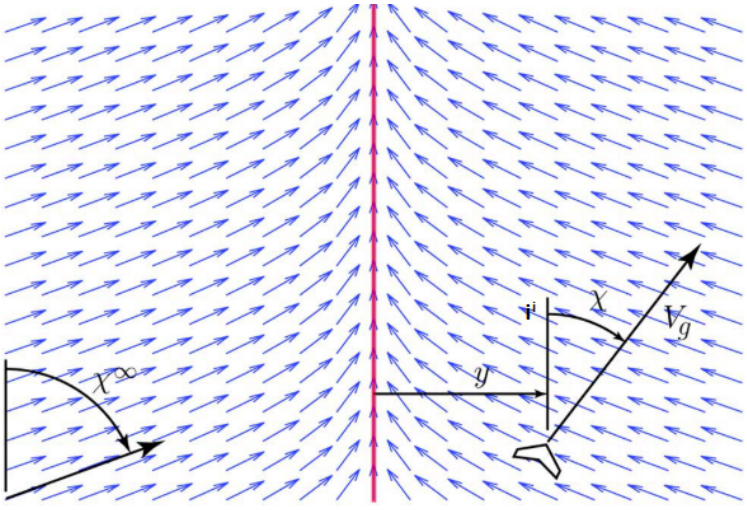


Figure 5.5: Vector Field constructed by the path planning for a straight line following. The χ^∞ angle represents the direction of the vector field when it is very far from the path [17].

Using Sliding mode, the set $S = (y, \chi) : \chi = \chi^d(y)$ is constructed. As presented in Fig.5.5, the navigation will be done in terms of the course angle χ, which is the angle between i^i and the ground velocity vector V_g. Next, we define the course angle error as

$$\tilde{\chi} = \chi - \chi^d \tag{5.20}$$

The role of the Path following is to drive this error to zero. Using Lyapunov stability we have that $W_2 = (1/2)\tilde{\chi}^2$, therefore

$$\dot{W}_2 = \tilde{\chi}(\alpha(\chi^c - \chi) + \chi^\infty \frac{2}{\pi} \frac{k}{1+(ky)^2} V_g \sin\chi) \tag{5.21}$$

If we chose the control law

$$\chi^c = \chi - \frac{1}{\alpha}\chi^\infty \frac{2}{\pi} \frac{k}{1+(ky)^2} V_g \sin\chi - \frac{k}{\alpha}\text{sign}(\frac{\tilde{\chi}}{\epsilon}) \tag{5.22}$$

and $k > 0$, then

$$\dot{W}_2 \leq -k|\tilde{\chi}|. \tag{5.23}$$

Allowing us to conclude that $\tilde{\chi}$ goes to zero in finite time [17]. As we know, the Sliding mode can introduce oscillation to the system, called chattering. In this case, is the sign(t) function that does that. To avoid this effect, we substitute sign(t) for

$$sat(t) = \begin{cases} t & |t| \leq 1 \\ \text{sign}(t) & otherwise \end{cases} \tag{5.24}$$

Hence, the final control law results in

$$\chi^c = \chi - \frac{1}{\alpha}\chi^\infty \frac{2}{\pi} \frac{k}{1+(ky)^2} V_g \sin\chi - \frac{k}{\alpha}\text{sign}(\frac{\tilde{\chi}}{\epsilon}) \tag{5.25}$$

where ϵ and α are positive constants.

5.2.1.2 Orbit

The structure of the orbit following is very similar to the previous one. First, we need to create a vector field for every point of the space, but this time around a circle as shown in figure 5.6.

It is assumed that the desired orbit has a known center (c_x, c_y) and radius r. The desired course for the orbit following can be written as:

$$\chi^d(d) = \gamma - \frac{\pi}{2} - \arctan(k(d-r)) \tag{5.26}$$

where k is a positive constant and γ and d are, respectively, the angle and the module of the UAV position in polar coordinates. Once the vector field is built, we need now a path following algorithm that will make the UAV follow this field. Let´s define the position error as $\tilde{d} = d - r$, and by using the Lyapunov function $W_1 = (1/2)\tilde{d}^2$ we can say that $\tilde{d}$ goes to zero as χ goes to χ^d. By deriving W_1

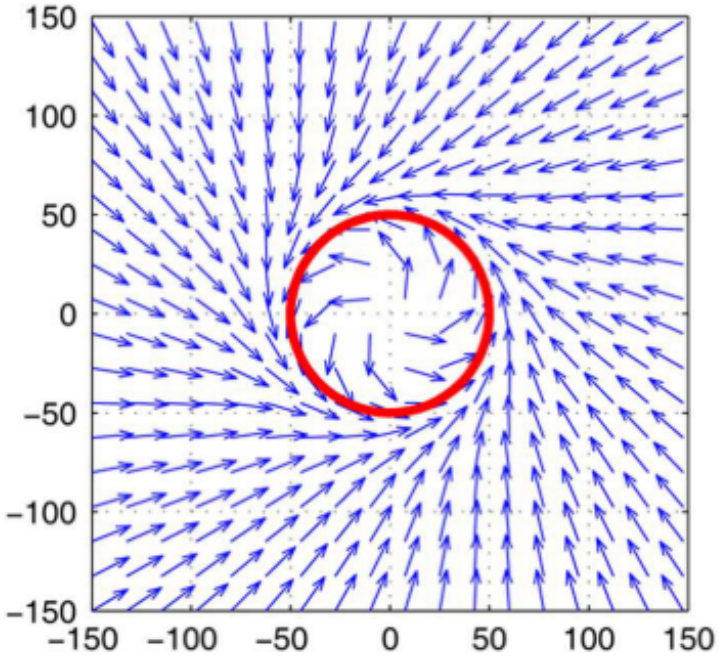


Figure 5.6: Vector Field constructed by the Path planning for an orbit following [17].

$$\dot{W}_1 = -V_g \tilde{d} \sin(\arctan(k\tilde{d})) \tag{5.27}$$

we observe that it is < 0 for $\tilde{d} \neq 0$, which allows us to conclude, using Lyapunov stability, that goes to zero asymptotically. As in the case of the straight line path following, we will use sliding mode to build the set $S = (\tilde{d}, \chi) : \chi = \chi^d(d)$. Again, we differentiate the course error

$$\dot{\tilde{\chi}} = \alpha(\chi^d - \chi) - \frac{V_g}{d}\sin(\chi - \gamma) + \frac{k}{1 + (k\tilde{d}^2)} V_g \cos(\chi - \gamma) \tag{5.28}$$

and use the Lyapunov function $w_2 = (1/2)\tilde{\chi}^2$ to argue that χ goes to χ^d as d goes to zero. By differentiating W_2 we obtain:

$$\dot{W}_2 = \tilde{\chi}(\alpha(\chi^c - \chi) - \frac{V_g}{d}\sin(\chi - \gamma) + \frac{k}{1 + (k\tilde{d})^2} V_g \cos(\chi - \gamma)) \tag{5.29}$$

Considering the control law as

$$\chi^c = \chi + \frac{V_g}{\alpha d}\sin(\chi - \gamma) - \frac{k}{1 + (\alpha(k\tilde{d})^2)} V_g \cos(\chi - \gamma) - \frac{k}{\alpha}\text{sign}(\tilde{\chi}) \tag{5.30}$$

We can say that

$$\dot{W}_2 = -k|\tilde{\chi}| \tag{5.31}$$

which can make us conclude that goes to zero in finite time. As before, the function $sign(t)$ can introduce oscillation to the system, therefore we replace 5.30 with

$$\chi^c = \chi + \frac{V_g}{\alpha d}\sin(\chi - \gamma) - \frac{k}{\alpha(1+(k\tilde{d})^2)}V_g\cos(\chi-\gamma) - \frac{k}{\alpha}\text{sat}(\frac{\tilde{\chi}}{\epsilon}). \tag{5.32}$$

5.2.2 Adaptive path following

This control technique combines the path following law based on the vector field method with an adaptive term that mitigates the effect of the unknown wind component. In order to be able to design a controller, we must first emulate some environmental conditions appropriate to the problem. Let us, therefore, consider the wind as having a constant component and a time-varying in a co-sinusoidal form. The designer is unaware of all the parameters relating to the time-varying element of the wind. The designer only knows the wind's constant component.

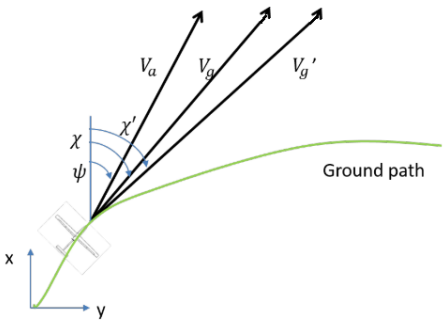


Figure 5.7: Relation between the airspeed velocity V_a, the ground velocity without the unknown wind components V_g and the ground velocity with all the disturbances V_g'. Since the time-varying wind is unknown, V_g' is not measurable [25].

The projection of this controller will be very similar to the one presented in 5.2.1, but as this time we have unknown wind components, we will have to use an estimator. The figure 5.8 presents us the block diagram of the system, with the path planning not included in the loop.

5.2.2.1 Straight line

As in 5.2.1 the goal is to make the UAV converge to a straight line along a vector field. Let's start by defining the position error as

$$e = y - (ax + b) \tag{5.33}$$

where a and b are the slope and the y-intercept, respectively. The desired course is then given by

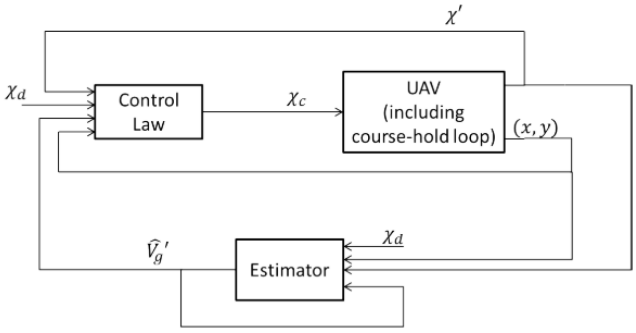


Figure 5.8: Adaptive path following block diagram [25].

$$\chi^d = \chi^\infty \frac{2}{\pi} \arctan(ke) + \arctan(a) \tag{5.34}$$

keeping the meaning of the constants from 5.2.1. Using Lyapunov stability, we will choose the function $V_1 = \frac{1}{2}e^2$ to argue that e will converge to zero as χ' goes to χ^d. Taking the derivative of V_1

$$\dot{V}_1 = eV_g' \frac{\chi^\infty \frac{2}{\pi} \arctan(ke)}{\cos(\arctan(a))} \tag{5.35}$$

we can find out that $\dot{V}_1$ is < 0, which means that e will converge to zero asymptotically. Let $V_2 = \frac{1}{2}\tilde{\chi}'^2$, where $\tilde{\chi}' = \chi' - \chi_d$ is the course error. The derivative of V_2 is

$$\dot{V}_2 = \tilde{\chi}'(\alpha(\chi_c - \chi') - \chi^\infty \frac{2}{\pi} \frac{k}{1 + (ke)^2} V_g'(sin\chi' - \chi')) \tag{5.36}$$

If we chose the control signal

$$\chi_c = \chi' + \frac{1}{\alpha}\chi^\infty \frac{2}{\pi} \frac{k}{1 + (ke)^2} V_g'(\sin\chi' - \chi') - \frac{k}{\alpha}\text{sat}(\frac{\tilde{\chi}'}{\epsilon}) \tag{5.37}$$

then, the derivative of V_2 is

$$\dot{V}_2 \begin{cases} \frac{-k}{\epsilon}\tilde{\chi}'^2 & |\frac{\tilde{\chi}'}{\epsilon}| < 1 \\ -k|\tilde{\chi}'| & otherwise \end{cases}. \tag{5.38}$$

Because $\dot{V}_2$ is <0, we can conclude that χ' will converge to the desired course in a finite time. As the wind has unknown components, in the equation 5.2.2.1 we cannot use the value of V_g', therefore, we have to use an estimated value, whose expression is given by

$$\dot{\hat{V}}_g' = -\Gamma\rho\tilde{\chi}'\chi^\infty \frac{2}{\pi} \frac{k}{1 + (ke)^2}(\sin\chi' - \chi') - \sigma\Gamma\hat{V}_g' \tag{5.39}$$

where Γ is a positive estimator gain, σ is the σ-modification value, which allows us to project an adaptive control without the need of knowing the boundaries of the disturbances, and ρ is a positive scale factor used to weight the terms so that y and chi are similar in magnitude.

5.2.2.2 Orbit

The method for orbit path following is similar to the straight line following in that the target orbits are surrounded by a vector field. The UAV's position is given in polar coordinates, with the orbit's center (c_x, c_y) serving as the origin. The distance between the orbit center and the UAV is represented by d and the angular location of the UAV is represented by γ. Starting by defining the position error as

$$\tilde{d} = d - r \tag{5.40}$$

where r is the orbit radius, the desired course for each point in the space is given by

$$\chi_d = \gamma + \frac{\pi}{2} + \arctan(k\tilde{d}). \tag{5.41}$$

As for the case of straight line path following, we will use Lyapunov theory to determine the control law. Consider the Lyapunov function

$$V_3 = \frac{1}{2}\tilde{d}^2 + \frac{1}{2}\rho\tilde{\chi}'^2 \tag{5.42}$$

whose derivative is

$$\dot{V}_3 = -V_g'\tilde{d}\sin(\arctan(k\tilde{d})) + \rho\tilde{\chi}'[\alpha(\chi_c - \chi') - \frac{V_g'}{d}\sin(\chi - \gamma) - {}_g'\cos(\chi' - \gamma)] \tag{5.43}$$

where

$$\beta = \frac{k}{1 + (k\tilde{d})^2}. \tag{5.44}$$

If we choose the control signal

$$\chi_c = \chi' + \frac{V_g'}{\alpha d}sin(\chi - \gamma) + \frac{\beta}{\alpha}V_g'\cos(\chi' - \gamma) - \frac{k}{\alpha}\text{sat}(\frac{\tilde{\chi}'}{\epsilon}) \tag{5.45}$$

we can observe that V_3 is negative, from which we can conclude that $\tilde{d}$ and $\tilde{\chi}'$ will converge to zero in finite time. As we said before, V_g' has unknown components, which means that it has to be estimated. The expression of the wind estimator is

$$\dot{\hat{V}}_g' = -\Gamma\rho\tilde{\chi}'\frac{\sin(\chi' - \gamma)}{d} + (\chi' - \gamma)) - \sigma\Gamma\hat{V}_g'. \tag{5.46}$$

Substituting V_g in expression 5.2.2.2 by the estimation, the control law is

$$\chi_c = \chi' + \frac{\hat{V}_g'}{\alpha d}\sin(\chi - \gamma) + \frac{\beta}{\alpha}\hat{V}_g'\cos(\chi' - \gamma) - \frac{k}{\alpha}\text{sat}(\frac{\tilde{\chi}'}{\epsilon}) \tag{5.47}$$

We have now defined the control signal χ_c for both straight and orbit follow algorithms, which will be sent to the autopilot. We will then test this algorithm with the attitude controller and with the model developed, in a closed loop.

6

Matlab Simulation

This chapter aims to validate and show the performance of the methods developed in Chapter 5, using Matlab simulations. First, we will test the attitude controller, giving it angular references, and then the path following algorithm, with spatial coordinates as references.

6.1 Backstepping attitude controller

The backstepping-based attitude controller was tested using the UAV model presented in chapter 3, closing the loop with the equations presented in 5.1. In figures 6.1 we have the representation of what was the implementation in Simulink. The trim conditions for the simulation are described as follows:

$u = 20m/s$	$v = 2m/s$	$w = 0m/s$	$\phi = 0.01rad$	$\theta = 0.01rad$
$\omega_e = 600rad/s$	$\alpha = 0.02rad$	$\beta = 0.1rad$	$\psi = 0.01rad$	
$p = 0$	$q = 0$	$r = 0rad/s$		

The choice for the parameters μ_p, μ_q and μ_r was made based on experience. First, they were all set to 1 and changed so that the control effort was as smooth as possible while ensuring that the response was fast enough so that the UAV was set to the desired attitude in a reasonable time. The same logic was applied to tune the values of μ_ϕ, μ_θ and μ_ψ. In figures 6.2, 6.3 and 6.4, we can observe the temporal evolution of the variables associated with the Roll channel, the pitch channel and the yaw channel, respectively. We can see that the higher the values of μ_ϕ, μ_θ and μ_ψ the better the control performance is, however, the control surfaces exert a higher effort, which some applications may be undesirable. For the UAV to be able to be integrated with a path following controller, it is desired that its response be fast.

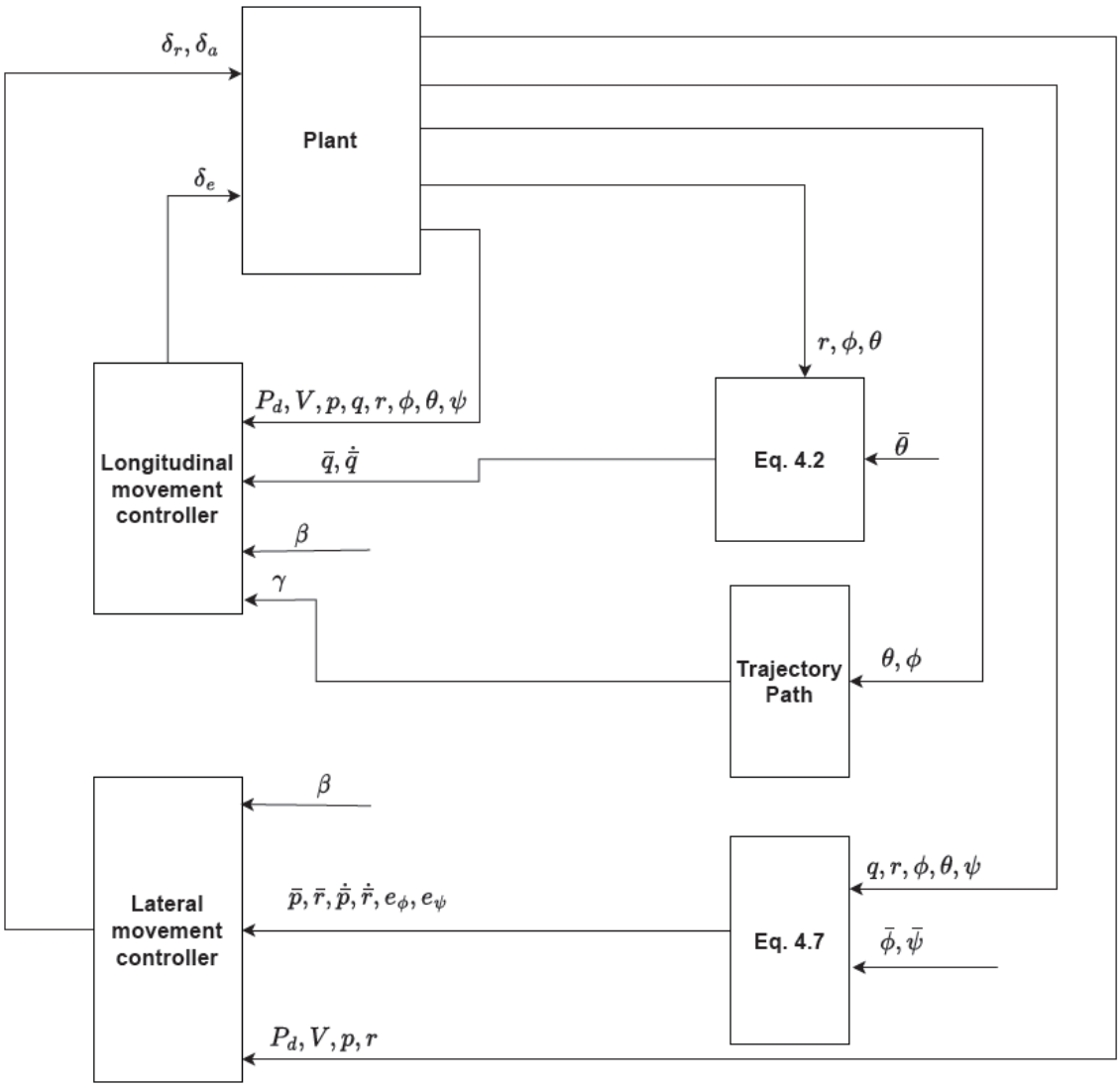


Figure 6.1: High level diagram of the Backstepping based attitude controller implementation.

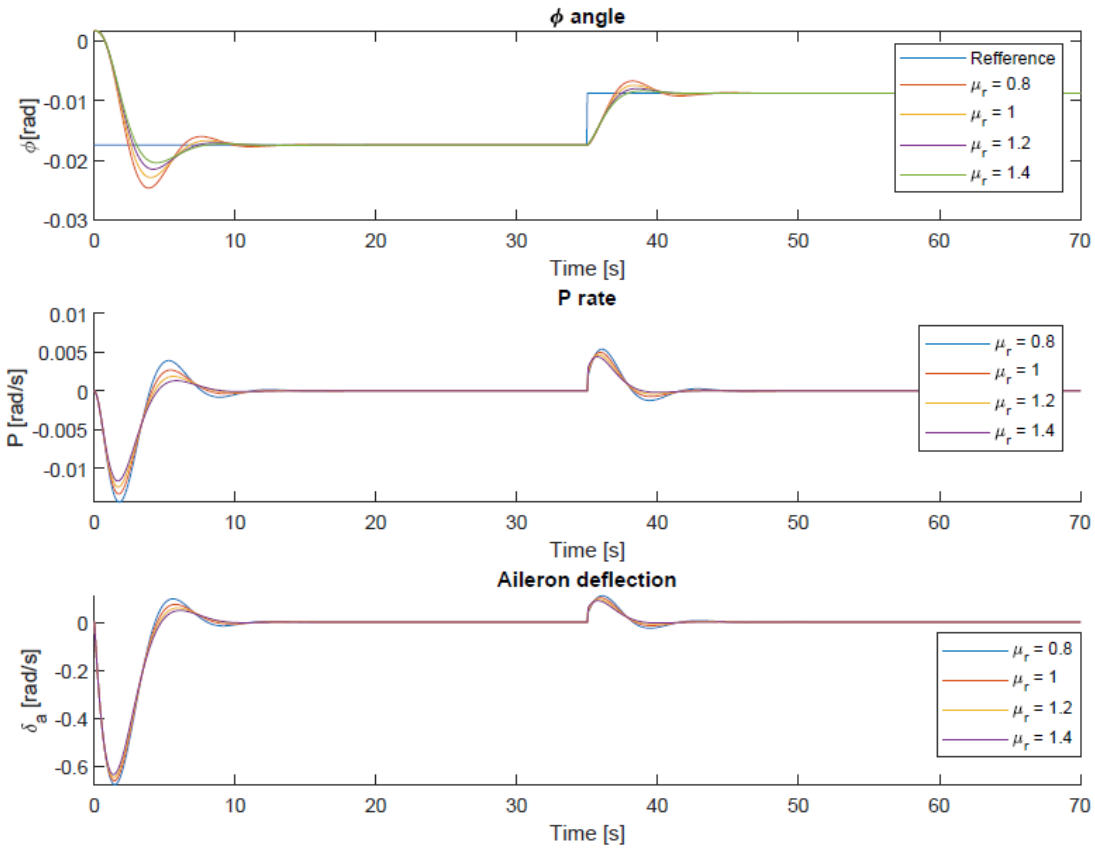


Figure 6.2: Temporal evolution of ϕ, p and δ_a, associated with the roll channel, simulated with different values of μ_ϕ, resulted from the Backstepping-based attitude controller in Simulink.

6.2 PI attitude controller

The implementation of the PI attitude controller was done using the Matlab toolbox called *PID tunner*. This toolbox offers an efficient and broadly applicable single-loop PID tuning approach. This technique allows you to adjust the PID controller's parameters to produce a reliable design with the required reaction time. First, the software computes the linear model, since the PI is a linear controller, then adjusting the control criteria, the *PID tunner* automatically computes the controller gains. In figure 6.5 is presented the Simulink implementation, based on what was projected in 5.1.2. For this design, it's important to have an integral effect, in order to eliminate the static error, although, as we can observe in Figure 6.7, the integral effect can sometimes degrade the response, by introducing a phase lag, reducing the relative stability and resulting in a degraded response. Even though the longitudinal channel response is quite satisfactory, we can see that for more complex dynamics such as the lateral ones, the PID has an insufficient performance when

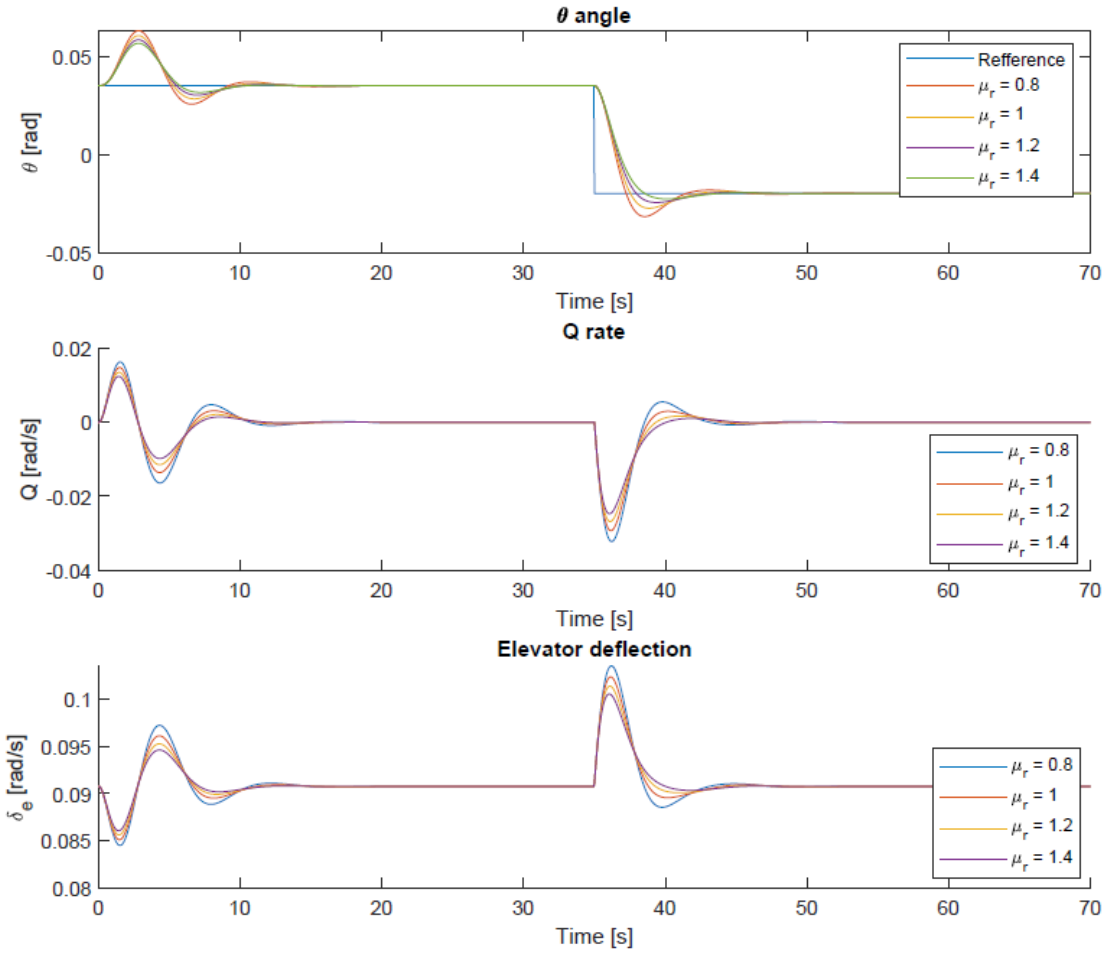


Figure 6.3: Temporal evolution of θ, q and δ_e, associated with the pitch channel, simulated with different values of μ_θ, resulted from the Backstepping-based attitude controller in Simulink.

compared to that obtained with the controller based on the Backstepping technique. As shown in tables 6.2 to 6.2 and by observing figures 6.6 and 6.7 we can see that there is practically no damping in the system response, therefore, there is no need to include a derivative component in the controllers, thus helping to reduce the complexity of the system.

6.3 Path following

This section presents the results for the real-time implementation in Matlab/Simulink of the path-following control algorithm with the vector field path following and the adaptive path following vector field based, for straight line following, orbit following and then show the capacity of the algorithm in performing more complex paths, comparing both

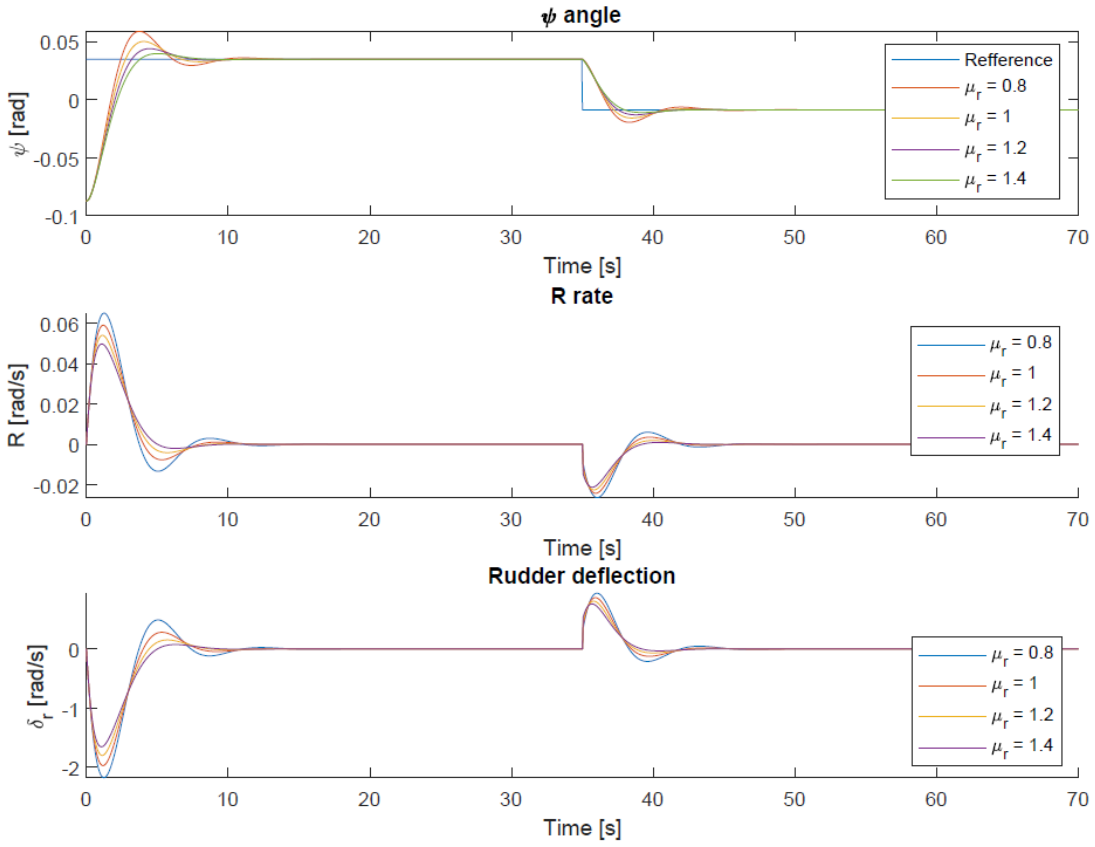


Figure 6.4: Temporal evolution of ψ, r and δ_r, associated with the yaw channel, simulated with different values of μ_ψ, resulted from the Backstepping-based attitude controller in Simulink.

Table 6.1: Pitch rate PI controller

Parameters	Value
P	3.8426
I	0.2481
Rise time	0.421 s
Settling time	0.638 s
Overshoot	1.38%
Margin phase	71.7 deg

path following algorithms. First, will be presented the integration of the UAV model developed, using the PID controller as autopilot and the path following algorithms presented in chapter 5. The PI autopilot was used instead of the Backstepping autopilot, because the integration between the backstepping controller and the path following is not trivial, due to its complexity. In figure 6.8 we can see the Simulink implementation of the full model. The system was simulated with constant altitude, controlling only the course angle, sent from the path following to the autopilot, which in turn will try to set

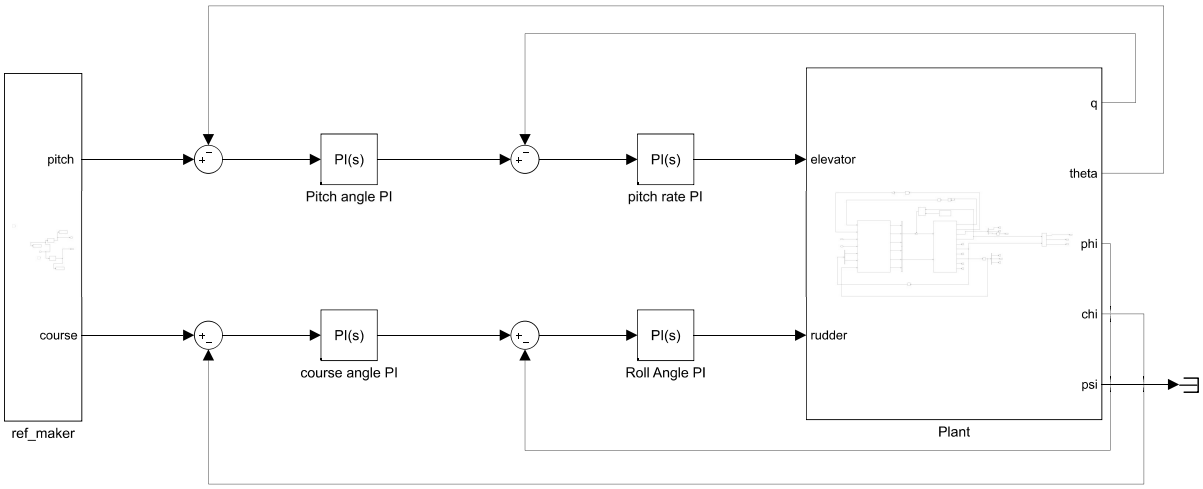


Figure 6.5: Simulink implementation of the PID-based attitude controller.

Table 6.2: Pitch angle PI controller

Parameters	Value
P	-0.0537
I	-1.4801
Rise time	0.121 s
Settling time	0.762 s
Overshoot	13.5%
Margin phase	80.6 deg

the plant with the desired course angle through the deflection of the rudder. In chapter 5 the autopilot was projected under the condition of the absence of wind, thus making the course angle equal to the heading angle.

Table 6.3: Roll angle PI controller

Parameters	Value
P	0.8036
Rise time	1.280 s
Settling time	4.910 s
Overshoot	0%
Margin phase	72.9 deg

Table 6.4: Course angle PI controller

Parameters	Value
P	-1.0189
I	-0.9612
Rise time	0.248 s
Settling time	4 s
Overshoot	1.2%
Margin phase	45 deg

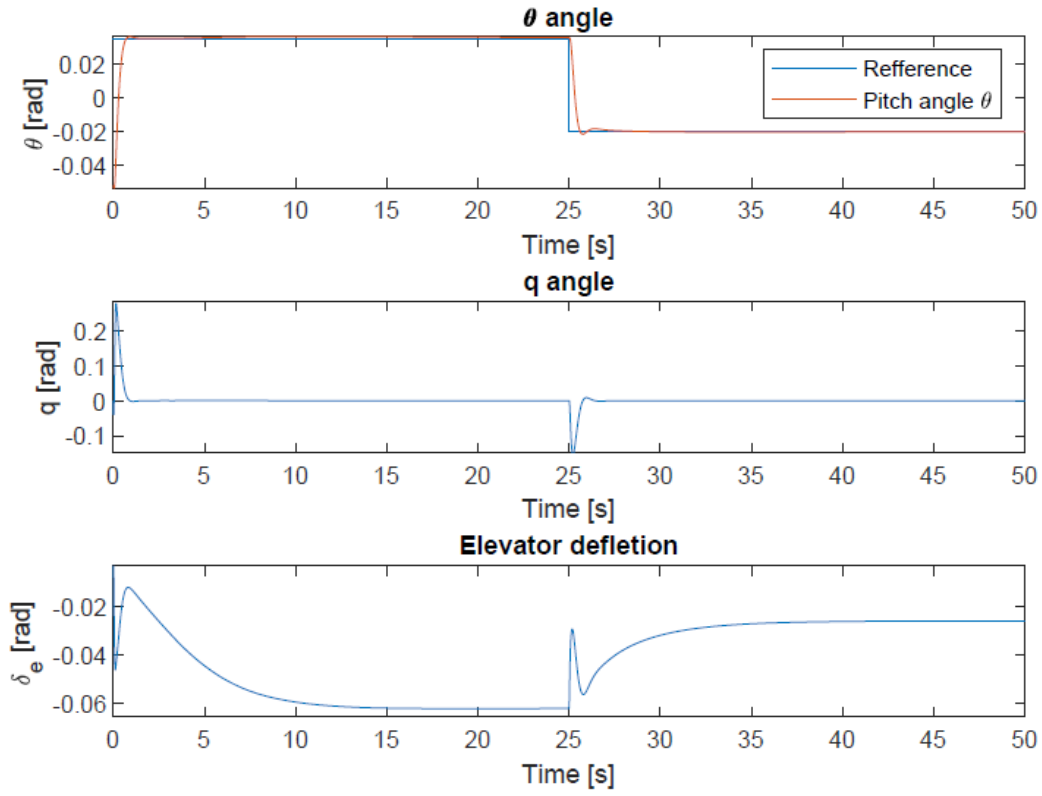


Figure 6.6: Temporal evolution of θ, q and δ_e, associated with the longitudinal dynamics.

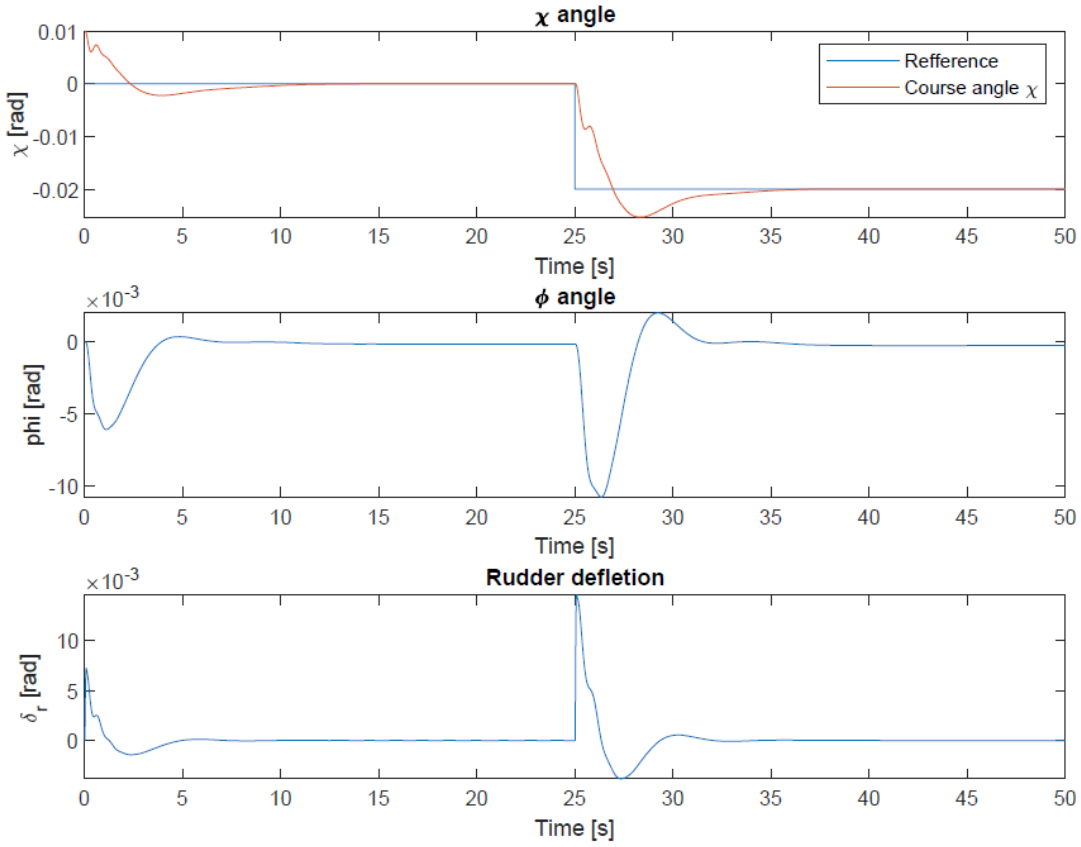


Figure 6.7: Temporal evolution of χ, ϕ and δ_r, associated with the lateral dynamics.

6.3.1 Straight line following

In the simulation of the Straight line following vector field the initial conditions are the following:

$$x = 0m \qquad y = 100m \qquad z = 30m \qquad V_0 = 9m/s$$
$$\alpha = 2° \qquad \beta = 0° \qquad \chi = 45°$$

In Figure 6.9 we can see the experimental results of the straight line path following algorithm and the temporal evolution of the course angle χ, in Figure 6.11. The temporal evolution of the course angle tends towards a value, which is expected for a straight line follow. We can see that the vector field, represented the χ_d approaches the direction of the line as the position error, represented in Figure 6.10, goes to zero, as expected accordingly with what was projected in section 5. The control parameters used in this simulation are:

$$\alpha = 15 \qquad k = 0.05 \qquad \chi_\infty = \pi/2 \qquad \epsilon = 0.5 \qquad \kappa = \pi/2$$

By changing the value of k, we can see in figure 6.12 the different performances of the UAV following the straight line. It goes without saying that the smaller the value of k, the

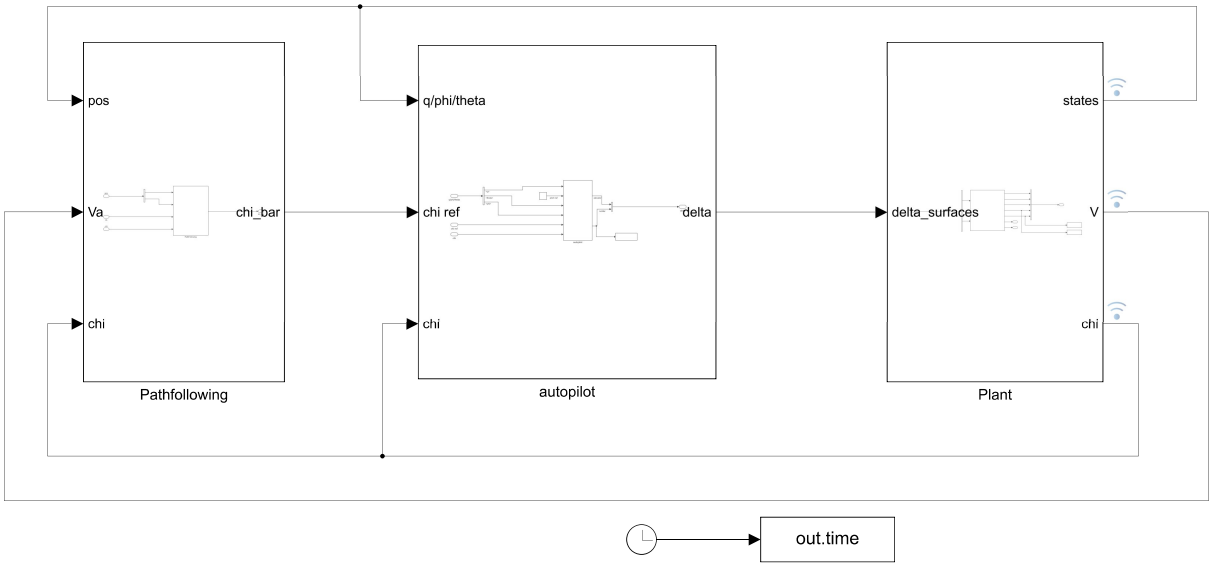


Figure 6.8: Simulink implementation of the full control system, incorporating the PID-based attitude controller and the vector field path following.

smoother the UAV approaches the path. We can justify that by observing figure 6.13, and see that, the larger the value of k, the faster the direction of the vector field approaches the direction of the line.

6.3.2 Orbit following

In the simulation of the Straight line following vector field the initial conditions are the following:

$$x = 40m \qquad y = 40m \qquad z = 30m \qquad V_0 = 9m/s$$
$$\alpha = 2^\circ \qquad \beta = 0^\circ \qquad \chi = -90^\circ$$

In figure 6.14 we can see the experimental results of the orbit follow algorithm and the temporal evolution of the course angle χ, in figure 6.11. As expected, the course angle follows a straight line as the UAV is orbiting. As Matlab limits the course angle between $[-\pi, \pi]$, the angle reference, when passing this range, returns to the initial value,

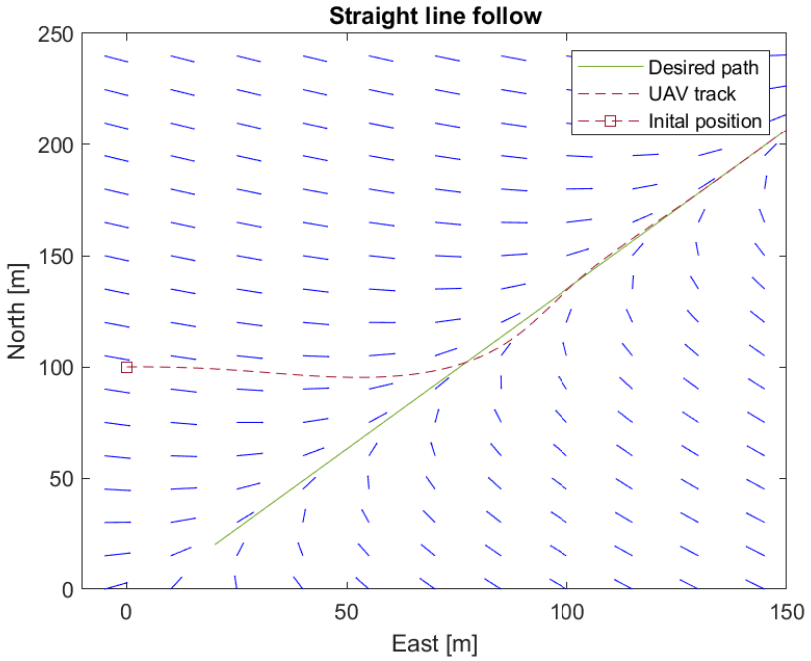


Figure 6.9: Straight line following performance.

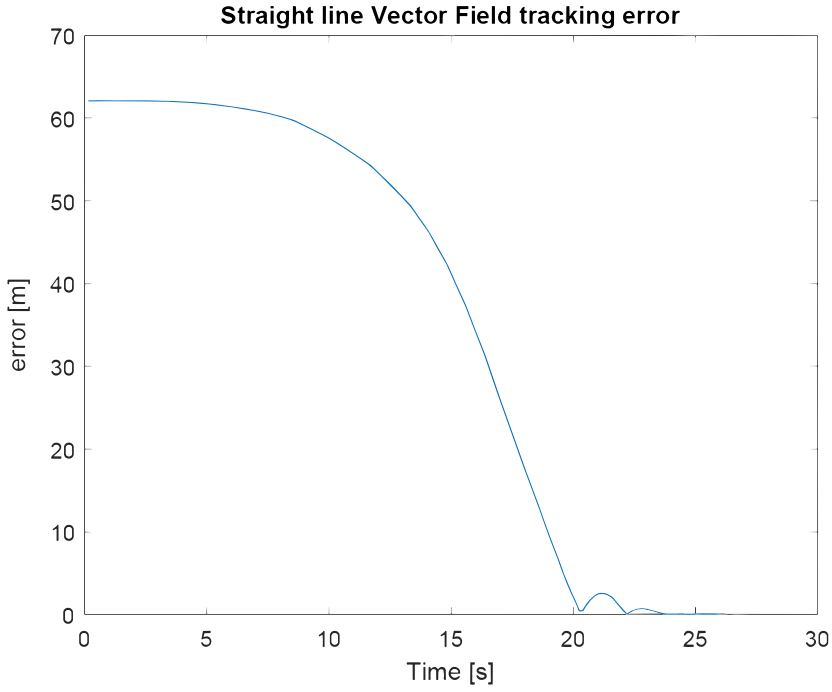


Figure 6.10: Tracking error of the Straight line following algorithm.

resulting in a discontinuity of the reference. To overcome this effect, multiples of 2π were added until the jump was less than π. In this case, the tracking error does not stand at zero, as we can see from the Figure 6.15. As the algorithm is in a loop with the model and with the autopilot, it is expected that the tracking is not perfect. As in the previous

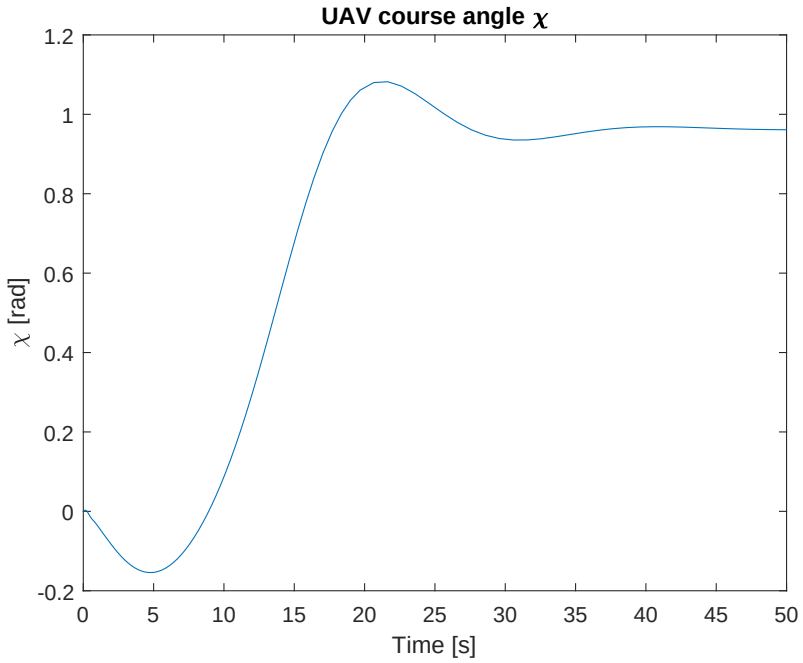


Figure 6.11: Temporal evolution of χ in straight line following simulation.

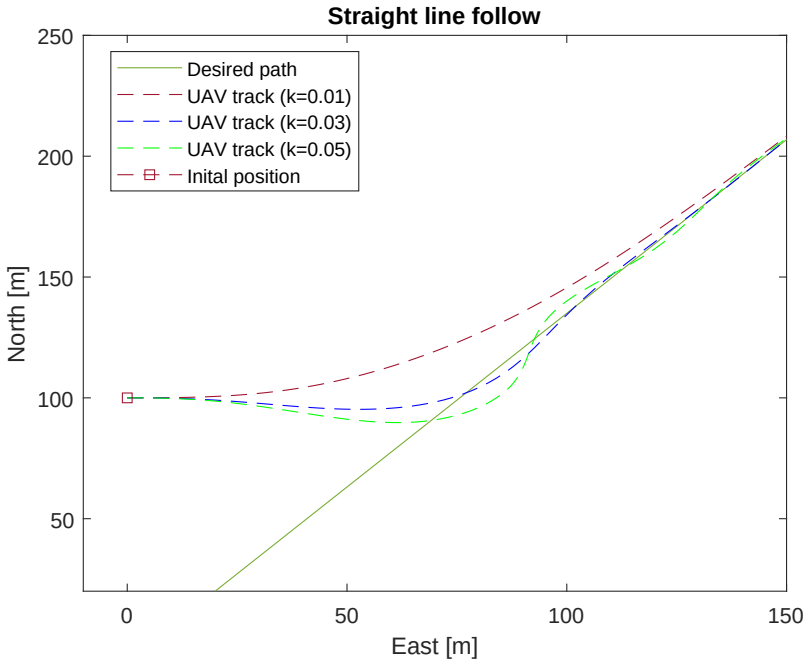


Figure 6.12: Comparison of the performance of the straight line path following, making the value of k vary.

algorithm, by varying the value of k, we can see in figure 6.17 that the higher the value of k, the more attracted the UAV is to the desired path. In figure 6.18 we can see that the smaller the value of k, the less the field vectors point toward the center of the trajectory, causing the UAV to scatter from its path.

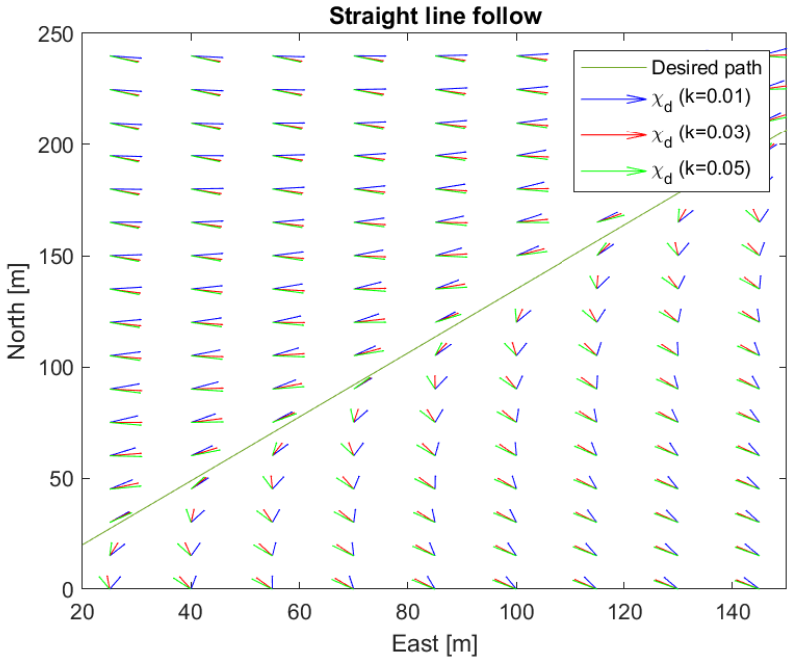


Figure 6.13: Variation of the vector field depending on the value of k, for the straight line path following.

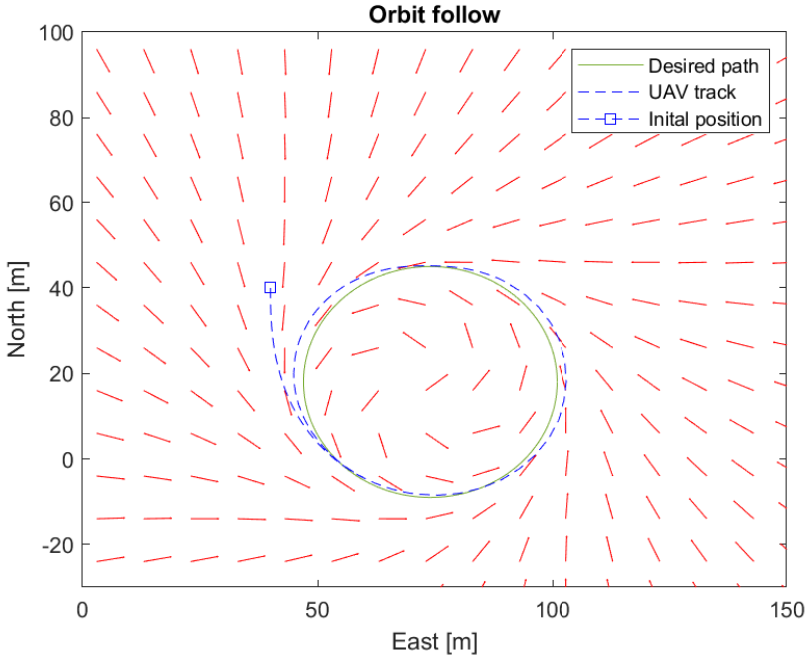


Figure 6.14: Orbit follow performance.

6.3.3 Following a Dubins Path

In this subsection we will test the algorithm's ability to perform more elaborate maneuvers, as in the case of Dubins Path with Right-Straight-Right (RSR) configuration, ending

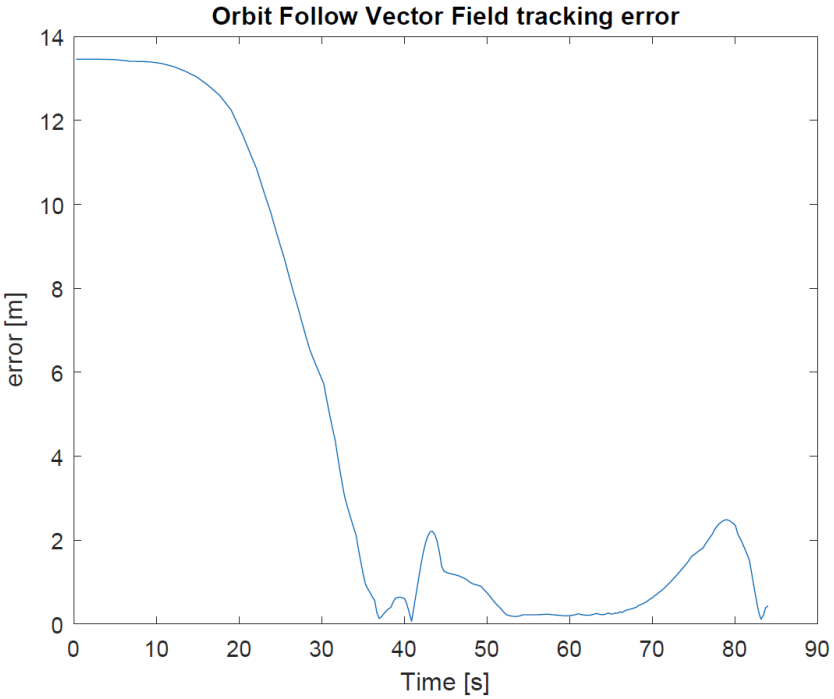


Figure 6.15: Tracking error of the Orbit following algorithm.

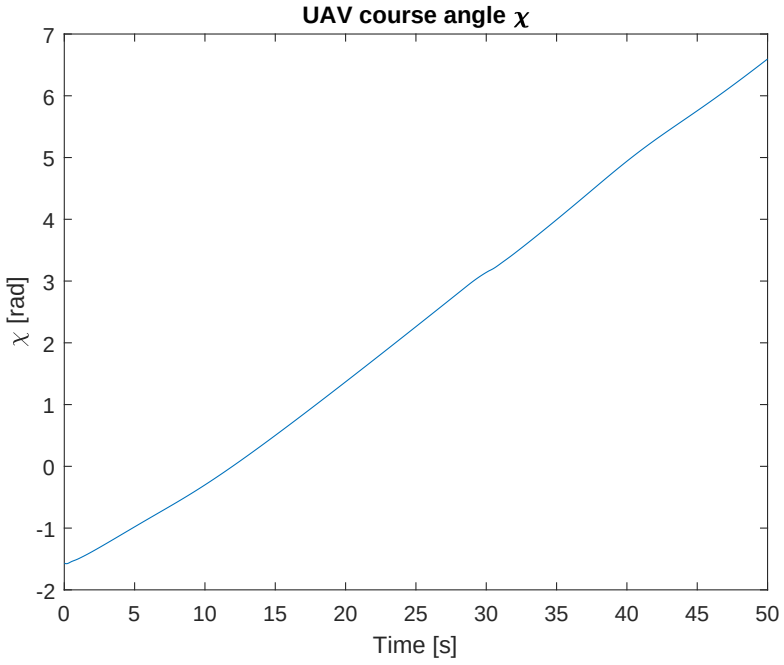


Figure 6.16: Temporal evolution of χ in orbit follow simulation.

with a straight line to close the path. Dubins paths are well known for their excellent coverage for almost all real cases.

As the idea is just to test the path following algorithm, let's consider a general UAV model, in order to eliminate the autopilot dynamics, to visualize with greater precision what

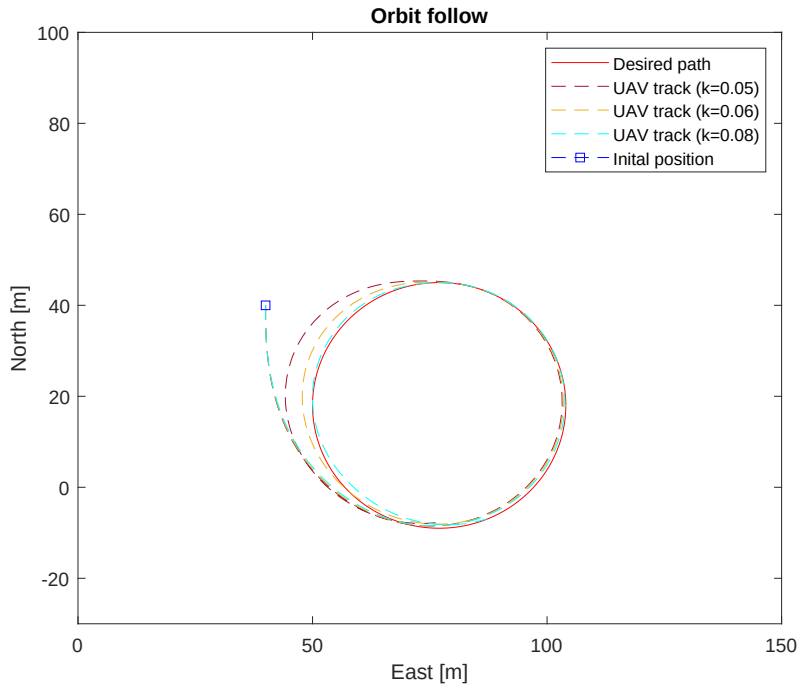


Figure 6.17: Comparison of the performance of the orbit following, making the value of k vary.

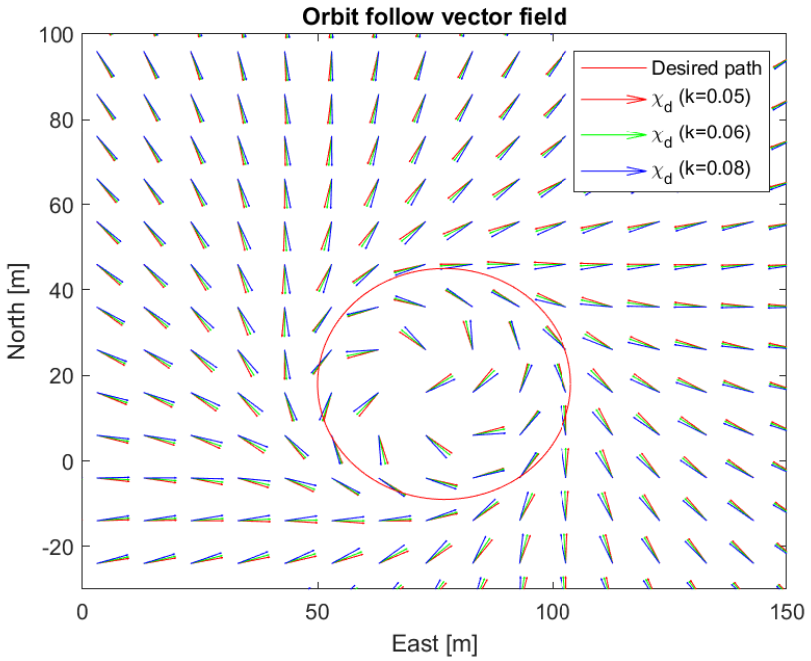


Figure 6.18: Variation of the vector field depending on the value of k, for the orbit following.

the algorithm is capable of doing and simplify the introduction of wind dynamics. The general UAV dynamics are given as

$$\begin{aligned} \dot{x} &= V_g \cos\chi \\ \dot{y} &= V_g \sin\chi \\ \dot{\chi} &= \alpha(\chi^c - \chi) \end{aligned} \tag{6.1}$$

being α a positive constant characterizing the speed at which the autopilot makes the UAV reach the desired χ. In order to test the two path following algorithms and compare their performance, we will introduce wind of variable amplitude and direction to the system, as well as another wind component with known amplitude and direction. Consider the following conditions:

$k = 0.05$	$\alpha = 13$	$\epsilon = 0.5$
$\Gamma = 74$	$W = 5km/h$	$\phi_w = -100°$
$A = 3\cos 0.1t$	$\phi_A = 2.7\cos 0.1t$	

with A being the amplitude of the variable wind component, ϕ_A the variable wind's direction, W the constant wind direction and ϕ_w the direction of the constant wind. In figure 6.19 we can see the comparison between the two algorithms. We can see that the spatial response of the adaptive algorithm is more satisfactory, presenting fewer oscillations, and approaching the reference faster. It is noteworthy that the vector field algorithm, when it passes from the first left curve to the straight line, presents an undesirable oscillation, being evident in Figure 6.20 that its tracking error is greater than that of the adaptive after starting to follow the line (approximately at t=9s). In Figures 6.21 and 6.22 we can see the temporal evolution of position and temporal evolution of the course angle χ, respectively.

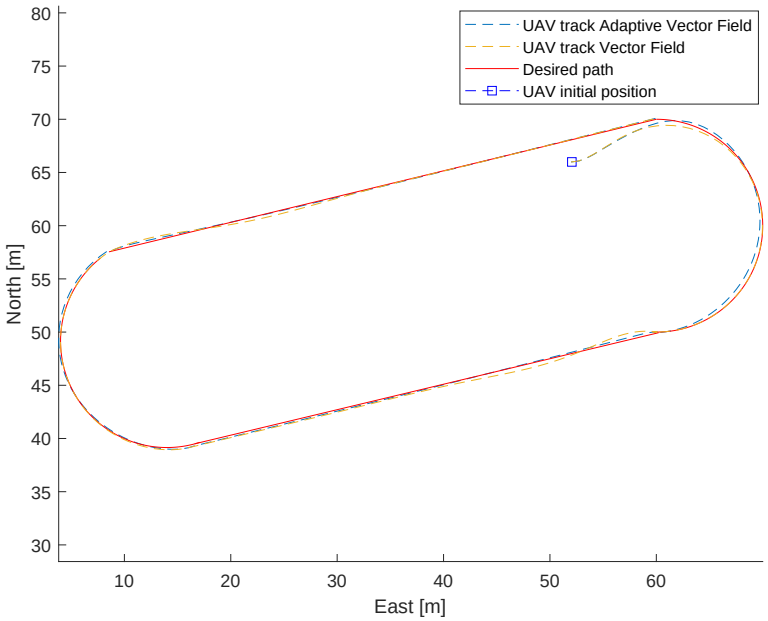


Figure 6.19: Following of a Dubins Path with RSR configuration, ending with a straight line.

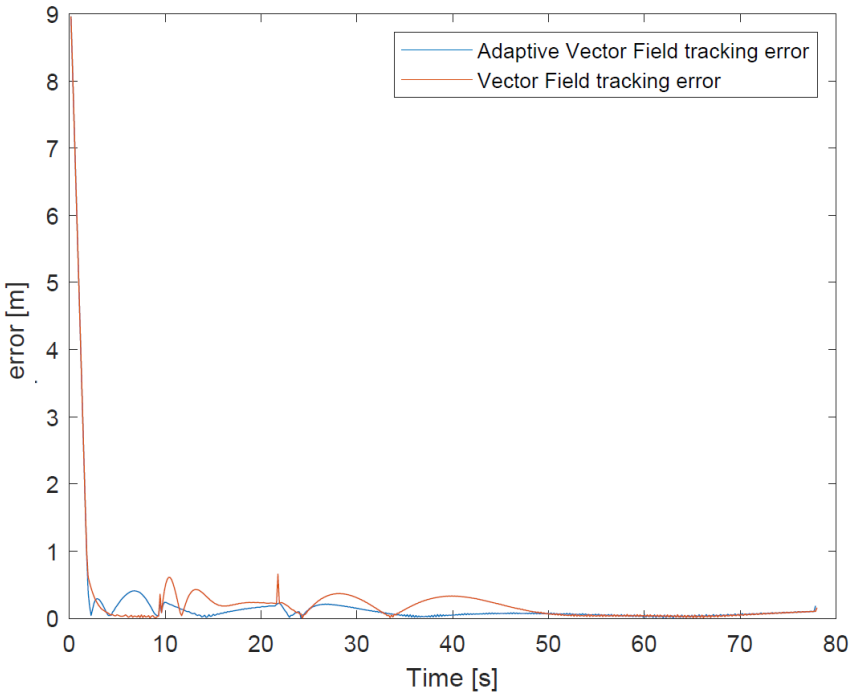


Figure 6.20: Tracking error of the Dubins Path UAV track.

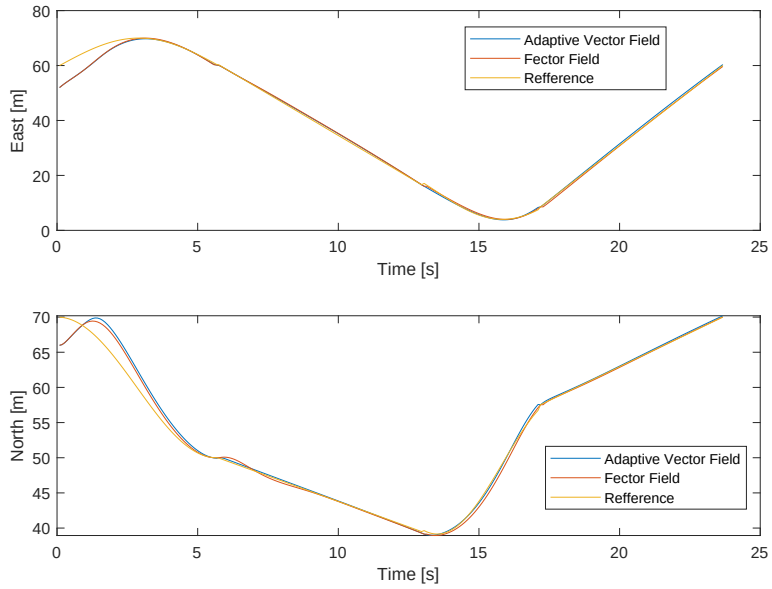


Figure 6.21: Position over time.

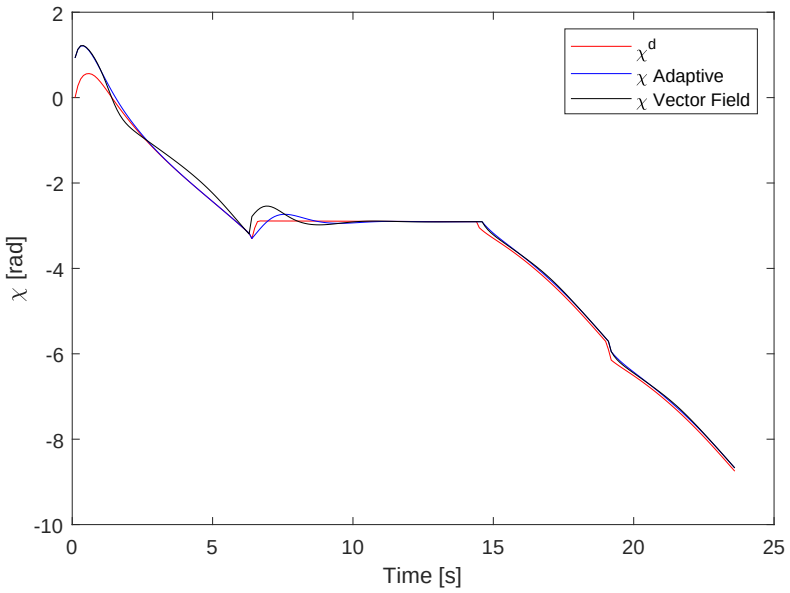


Figure 6.22: Course angle over time.

7

Conclusion

From a broad perspective, the main objective of this dissertation was to instrument and control a UAV. This was made possible by merging methods from various controller classes, which led to autonomous flight.

The first step of this dissertation is the instrumentation of the vehicle. The integration of sensors, communication with the ground computer for monitoring and control purposes, and actuators were carried out. Through small tests, it was concluded that the UAV was ready to fly, although it was not possible to perform an autonomous flight through off-board control, implementing the controllers developed in Matlab into the real system.

A dynamic model of the UAV was developed by considering the kinematics and non-linear dynamics of the aircraft. The state variables, such as position and velocities, were defined, and the aerodynamic forces and moments that act on the UAV were taken into account.

An attitude controller was designed using two different methods: a linear PI controller and a non-linear backstepping controller. The performance of these controllers was tested and compared through simulations using Matlab. It was found that the backstepping controller has a better performance, particularly in the lateral dynamics of the UAV. Finally, a path-planning and path-following controller for the UAV was developed, which allows the aircraft to follow a certain trajectory without any time specification. The controller was tested in simulations with both known and unknown wind conditions. It was found that the controller is able to follow a desired path effectively.

In conclusion, this work provides a comprehensive study on the modeling, control design, and path following algorithms for UAVs. The proposed dynamic model, attitude controller, and path-following controller can be used in the design and control of UAVs for various applications. The main contributions of this work are the development of an attitude controller using both linear and non-linear methods, the implementation of a path-planning and path-following controller and the UAV intrumentation.

7.1 Future work

The proposed controllers and path following algorithms have been successfully simulated in Matlab, but further validation is needed through real-world flight testing. The next step in this work is to implement the controllers and path following algorithm on an autonomous UAV using off-board control instead of the PX4 algorithm. Additionally, the backstepping controller can be integrated with the path following algorithm in order to achieve a more robust control system. Additionally, the work presented in this dissertation will be integrated with the CAPTURE.

Bibliography

[1] A.J.S.B. "Editor's Preface". In: *Centralized and Automatic Controls in Ships*. Ed. by D. GRAY. Pergamon, 1966, p. ix. ISBN: 978-1-4832-1355-2. DOI: https://doi.org/10.1016/B978-1-4832-1355-2.50005-4 (cit. on p. 9).

[2] R. M. Abarca. *Flight Dynamics Principles*. 2021, pp. 2013–2015. ISBN: 9780750669276 (cit. on p. 9).

[3] M. Ahsan, H. Rafique, and Z. Abbas. "Heading control of a fixed wing UAV using alternate control surfaces". In: *2012 15th International Multitopic Conference, INMIC 2012* (2012), pp. 125–129. DOI: 10.1109/INMIC.2012.6511475 (cit. on p. 42).

[4] D. Balaji and E. Maheswari. "MODEL PREDICTIVE CONTROL TECHNIQUES FOR CSTR USING MATLAB". In: 3 (Oct. 2012) (cit. on p. 11).

[5] R. W. Beard and T. W. McLain. *Small Unmanned Aircraft Theory and Practice – Princeton University Pres* 2012, p. 317. ISBN: 9780691149219. URL: http://uavbook.byu.edu/doku.php (cit. on pp. 4–9, 27–30, 32, 33).

[6] V. Cichella et al. "Geometric 3D path-following control for a fixed-wing UAV on SO(3)". In: *AIAA Guidance, Navigation, and Control Conference 2011* August (2011), pp. 1–15. DOI: 10.2514/6.2011-6415 (cit. on p. 43).

[7] K. Goebel and B. Saha. *Prognostics applied to electric propulsion uav*. 2015, pp. 1063–1070. ISBN: 9789048197071. DOI: 10.1007/978-90-481-9707-1_47 (cit. on pp. 8, 31, 32).

[8] B. Guerreiro. *Project capture*. URL: http://capture.isr.tecnico.ulisboa.pt/ (cit. on p. 2).

[9] W. Khan and M. Nahon. "Development and validation of a propeller slipstream model for unmanned aerial vehicles". In: *Journal of Aircraft* 52.6 (2015), pp. 1985–1994. ISSN: 15333868. DOI: 10.2514/1.C033118 (cit. on p. 33).

[10] W. Khan and M. Nahon. "Real-time modeling of agile fixed-wing UAV aerodynamics". In: *2015 International Conference on Unmanned Aircraft Systems, ICUAS 2015* (2015), pp. 1188–1195. DOI: 10.1109/ICUAS.2015.7152411 (cit. on p. 32).

[11] J. M. Lourenço. *The NOVAthesis LaTeX Template User's Manual*. NOVA University Lisbon. 2021. URL: https://github.com/joaomlourenco/novathesis/raw/master/template.pdf (cit. on p. ii).

[12] M. Lungu. *Adaptive Control Signal - 2012 - Lungu - Adaptive backstepping flight control for a mini-UAV.pdf* (cit. on pp. 12–14, 39).

[13] MATLAB. *Model predictive control*. URL: https://www.mathworks.com/videos/series/understanding-model-predictive-control.html (cit. on p. 10).

[14] N. E. A. Mazlan et al. "The Design of an Automatic Flight Control System and Dynamic Simulation for Fixed-Wing Unmanned Aerial Vehicle (UAV) using X-Plane and LabVIEW". In: *Progress in Aerospace and Aviation Technology* (2021), pp. 56–69. DOI: 10.30880/paat.2021.01.01.007 (cit. on p. 42).

[15] N. Mehta et al. "Design of HMI Based on PID Control of Temperature". In: *International Journal of Engineering Research and* V6 (May 2017). DOI: 10.17577/IJERTV6IS050074 (cit. on p. 10).

[16] A. P. Melo. "Sistemas de Controlo Lineares". In: 2015 (2008) (cit. on pp. 11, 12).

[17] D. R. Nelson et al. "Vector field path following for miniature air vehicles". In: *IEEE Transactions on Robotics* 23.3 (2007), pp. 519–529. ISSN: 15523098. DOI: 10.1109/TRO.2007.898976 (cit. on pp. 44–46).

[18] B. Neves and B. Guerreiro. "Flight control of hybrid drones towards enabling parcel relay manoeuvres". In: (2021), pp. 13–19. DOI: 10.1109/yef-ece52297.2021.9505161 (cit. on p. 8).

[19] T. Peden. *Frames of Reference*. URL: https://rb.gy/fll1jf (cit. on p. 6).

[20] Pixhawk. *PX4 autopilot user guide (main)*. URL: https://docs.px4.io/main/en/ (cit. on pp. 17, 19).

[21] J. Roskam. *Airplane Flight Dynamics and Automatic Flight Controls - Jan Roskam-2.pdf*. 2011 (cit. on p. 32).

[22] J.-J. Slotine and W. Li. *Applied nolinear optimal control*. 1991, p. 461. ISBN: 0-13-040890-5 (cit. on pp. 14, 15).

[23] N. Sprague. "Coordinate Frames". In: *Classical Mechanics* (2006), pp. 68–82. DOI: 10.1007/0-387-21851-3_10 (cit. on p. 4).

[24] R. Vepa. *Flight dynamics, simulation, and control: For rigid and flexible aircraft*. 2014, pp. 1–652. ISBN: 9781466573369. DOI: 10.1201/b17346 (cit. on p. 32).

[25] B. Zhou, H. Satyavada, and S. Baldi. "Adaptive path following for Unmanned Aerial Vehicles in time-varying unknown wind environments". In: *Proceedings of the American Control Conference* (2017), pp. 1127–1132. ISSN: 07431619. DOI: 10.23919/ACC.2017.7963104 (cit. on pp. 47, 48).