Titre: Title:	Optimal Path Planning and Control of the Descent of a Gondola with Payload
Auteur: Author:	Rémi Pédenon-Orlanducci
Date:	2023
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Pédenon-Orlanducci, R. (2023). Optimal Path Planning and Control of the Descent of a Gondola with Payload [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/56772/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/56772/
Directeurs de recherche: Advisors:	David Saussié, & Sofiane Achiche
Programme: Program:	Génie électrique

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Optimal planning and control of the descent of a gondola with payload

RÉMI PÉDENON-ORLANDUCCI
Département de génie électrique

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Génie aérospatial

Novembre 2023

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

Optimal planning and control of the descent of a gondola with payload

présenté par **Rémi PÉDENON-ORLANDUCCI**
en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
a été dûment accepté par le jury d'examen constitué de :

Richard GOURDEAU, président
David SAUSSIÉ, membre et directeur de recherche
Sofiane ACHICHE, membre et codirecteur de recherche
Mohebbi ABOLFAZL, membre

DEDICATION

“A pessimist sees the difficulty in every opportunity;
an optimist sees the opportunity in every difficulty.”
Winston Churchill

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my research advisors, who have supported me throughout these two years of my master's program. My stay in Quebec has been going very well so far, and you have played a significant role in that. Thank you, David, for your high standards, sometimes strict but always caring, which have helped me grow both personally and scientifically. Thank you, Sofiane, for your cheerfulness and unwavering dedication. Thank you both for giving me the opportunity to experience university teaching as a teaching assistant. Undoubtedly, it is the beginning of a calling.

This project was carried out under a contract with the startup Lux Aerobot, through the Mitacs program. I would like to thank this organization as well as Lux and its employees: they have allowed me to have a first glimpse of the working world abroad, in a stimulating environment and surrounded by wonderful people.

I would also like to express my gratitude to professors Abdolfazl and Gourdeau for agreeing to be part of my jury. Professor Gourdeau, in particular, our discussions throughout my master's program have been extremely helpful. They have given me a lot to think about and have significantly helped me move forward when I lacked clarity. Your way of bridging the beauty of mathematics with the pragmatism of engineering is fascinating.

I also want to warmly thank my friends with whom I have shared so much over the past two years. Olivier, the ups and downs of our master's programs have made you my first friend from another continent, and you are still the person I describe when asked, 'So, what are the Quebecois like?' Ahmad, thank you for always being available to help me whenever I need it, whether it's moving, writing an article, or playing GTA. You have brought a Middle Eastern touch to my master's program, and I am grateful for that. Chris, thank you for your kindness and unwavering support. You have always been there for me, through good times and bad, for important matters and trivial ones. Lucas, what a joy to have a desk neighbour like you! When I am at Poly, I know that I just have to look to my left to have a good laugh; and if not, I just have to call you to benefit from the plans that only you know! Narimane, finally, thank you for your cheerfulness and unwavering positivity. Thankfully for the lab, you have been a top-notch psychologist for us!

Lastly and above all, thank you to Marie for everything. You are both my roots and my future. I am happy to be able to live in Montreal with you. This master’s thesis is also a little bit yours. If I hadn’t used the dedication space for a quote, I would certainly have mentioned my parents and you. My parents, who have always been there for me as well. Thank you for allowing me to become who I am. Thank you dad for your always wise advice, thank you mom for your unwavering moral support, thank you both for your love and your patience. I do not forget Alexis, Nanou and Anaëlle either. Ultimately, the greatest difficulty of this stay in Montreal has been to be far away from all of you.

RÉSUMÉ

L'importance croissante des effets du réchauffement climatique, tels que les feux de forêt, a suscité de nombreuses recherches sur la prévention et l'étude des risques liés à ces incendies, en particulier par télédétection. Pour faciliter la récupération de nacelles d'imagerie une fois au sol et éviter les pertes matérielles, un système de récupération guidée autonome doit être conçu. Ce système doit guider la nacelle vers un point d'atterrissage précis tout en minimisant la consommation d'énergie.

Le but de ce travail de maîtrise est de concevoir un planificateur de trajectoire optimale et un contrôleur pour la descente de la nacelle. Il commence par une revue de littérature expliquant le choix d'un parafoil et présentant l'état de l'art des contrôleurs de vol et des planificateurs de trajectoire pour ce type de système. Ensuite, l'étude dynamique de la nacelle permet de construire un simulateur non linéaire et d'identifier les modes aérodynamiques du système linéarisé autour des vols nominaux, en les comparant avec des systèmes similaires de la littérature.

Une fois le modèle établi, la planification de trajectoires optimales est abordée : le système doit atteindre la zone souhaitée avec un effort de commande minimal. La contribution principale de ce travail réside dans l'approche hybride de cette planification de trajectoires, qui combine la méthode continue et précise du tir non linéaire (*Non Linear Shooting* - NLS) avec la discrétisation de la programmation dynamique (*Dynamic Programming* - DP). Les trajectoires à suivre sont générées sur un modèle simplifié (2D) et le planificateur de trajectoire fonctionne comme un navigateur GPS, recalculant la trajectoire optimale par DP lorsque le système en vol s'éloigne de sa trajectoire nominale générée par NLS.

Une fois sur la zone, le système effectue une descente en spirale pour perdre de l'altitude, puis atterrit en ligne droite, similairement à un avion. Le suivi de trajectoire en phase 1 s'effectue grâce à un asservissement de l'écart entre la trajectoire optimale générée en 2D et la projection de la trajectoire réelle sur le sol. L'algorithme hybride ainsi que le contrôleur d'atterrissage sont testés en simulateur 3D sur différents types de trajectoires et de profils de vent, démontrant l'efficacité de l'algorithme et les économies d'énergie jusqu'à 15% dans la première phase de vol pour des vents très défavorables.

ABSTRACT

The growing importance of the effects of global warming, such as forest fires, has given rise to much research on the prevention and study of the risks linked to these fires, in particular by remote sensing. To facilitate the recovery of the gondola once on the ground and avoid material losses, an autonomous guided recovery system must be designed. This system must guide the nacelle to a precise landing point while minimizing energy consumption.

The goal of this master's work is to design an optimal trajectory planner and a controller for the descent of the gondola. It begins with a literature review explaining the choice of a parafoil and presenting the state of the art of flight controllers and trajectory planners for this type of system. Then, the dynamic study of the gondola allows the construction of a nonlinear simulator and the identification of the aerodynamic modes of the linearized system around nominal flights. These modes are compared with the ones from similar systems presented in the literature.

Once the model has been established, optimal path planning is addressed: the system must reach the desired area with minimal control effort. The main contribution of this work lies in the hybrid approach to this trajectory planning, which combines the continuous and precise method of nonlinear shooting (NLS) with the discretization of dynamic programming (DP). The trajectories to follow are generated on a simplified model (2D) and the path planner functions as a GPS navigator, recalculating the optimal trajectory by DP when the system in flight deviates from its nominal trajectory generated by NLS.

Once in the area, the system performs a spiral down movement to lose altitude, then lands in a straight line, similar to an airplane. The trajectory tracking in phase 1 is carried out by controlling the difference between the optimal trajectory generated in 2D and the projection of the real trajectory on the ground. The hybrid algorithm as well as the landing controller are tested in a 3D simulator on different types of trajectories and wind profiles, demonstrating the efficiency of the algorithm and energy savings of up to 15% in the first flight phase for highly unfavourable winds.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ACRONYMS

Acronyms

DOF	Degree Of Freedom
DP	Dynamic Programming
GNC	Guidance Navigation and Control
GR	Glide Ratio
NED	North-East-Down (flat-Earth inertial frame)
PID	Proportional-Integral-Derivative
SO	Sub-Objective
UAV	Unmanned Aerial Vehicle
NLS	Nonlinear Shooting

Flight mechanics

Frames

$\mathcal{F}_{\text{NED}} = \{O, \mathbf{x}_{\text{NED}}, \mathbf{y}_{\text{NED}}, \mathbf{z}_{\text{NED}}\}$	NED (North-East-Down) inertial frame
$\mathcal{F}_{\text{b}} = \{O_{\text{b}}, \mathbf{x}_{\text{b}}, \mathbf{y}_{\text{b}}, \mathbf{z}_{\text{b}}\}$	Body frame
$\mathcal{F}_{\text{s}} = \{O_{\text{s}}, \mathbf{x}_{\text{s}}, \mathbf{y}_{\text{s}}, \mathbf{z}_{\text{s}}\}$	Stability frame
$\mathcal{F}_{\text{w}} = \{O_{\text{w}}, \mathbf{x}_{\text{w}}, \mathbf{y}_{\text{w}}, \mathbf{z}_{\text{w}}\}$	Wind frame

Movement characteristics

$\mathbf{V}_{X/Y}^{F}$	Velocity vector of X relatively to Y expressed in the frame F
ϕ, θ, ψ	Euler angles (roll, pitch, yaw)
p, q, r	Roll, pitch and yaw angular rates
$\mathbf{V}_{\text{b/w}}^{\text{b}} = [u, v, w]^\top$	Velocity of the system relatively to the air, expressed in the body frame
$V_T = \|\|\mathbf{V}_{\text{b/w}}^{\text{b}}\|\|$	Airspeed of the system
α	Angle of attack
β	Sideslip angle
δ_A	Asymmetric control input
δ_B	Symmetric control input

Physical characteristics of the system

b_C	Canopy span
c	Canopy chord
$\mathbf{I}$	Inertia matrix of the gondola
m	Total mass of the gondola
S	Parafoil surface
$R_{\min}$	Minimal turning radius

Forces and torques

X_A, Y_A, Z_A	Aerodynamic forces applied to the system in the body frame
X_T, Y_T, Z_T	Thrust forces applied to the system in the body frame
C_l, C_m, C_n	Aerodynamic torques applied to the system in the body frame

Other physical quantities

ρ	Air density
$\rho_0 = 1.225\,\mathrm{kg.m^{-3}}$	Air density at sea level in standard conditions.
$g_0 = 9.81\,\mathrm{m.s^{-2}}$	Gravity constant

Path planning

(a, b)	North-East coordinates of the desired landing point
R	Radius of the circle targeted in phase 1
t_f	Final time for phase 1
$z_{2,\min}$	Minimum altitude to engage spiral down
V	Projection of airspeed in the North-East plane
$\times^*$	Optimal quantity or function $\times$
J	Cost to optimize with optimal controller
$\mathbf{x}$	State vector in phase 1
$\boldsymbol{\lambda}$	Lagrange functions vector in phase 1
$\boldsymbol{\psi}[\mathbf{x}(t), t]$	Constraints function which must be zero at $t = t_f$
$H(\mathbf{x}(t), \boldsymbol{\lambda}(t), t)$	Hamiltonian of the system in phase 1

LIST OF APPENDICES

CHAPTER 1 INTRODUCTION

1.1 Context

Over the past few years, with the increasing negative consequences of global warming, the damage caused by wildfires across the world, especially in Australia and North America, has greatly increased: the area burned tends to stabilize from one year to the next and around a higher value than the average for previous years (see Fig. 1.1 for Canada).

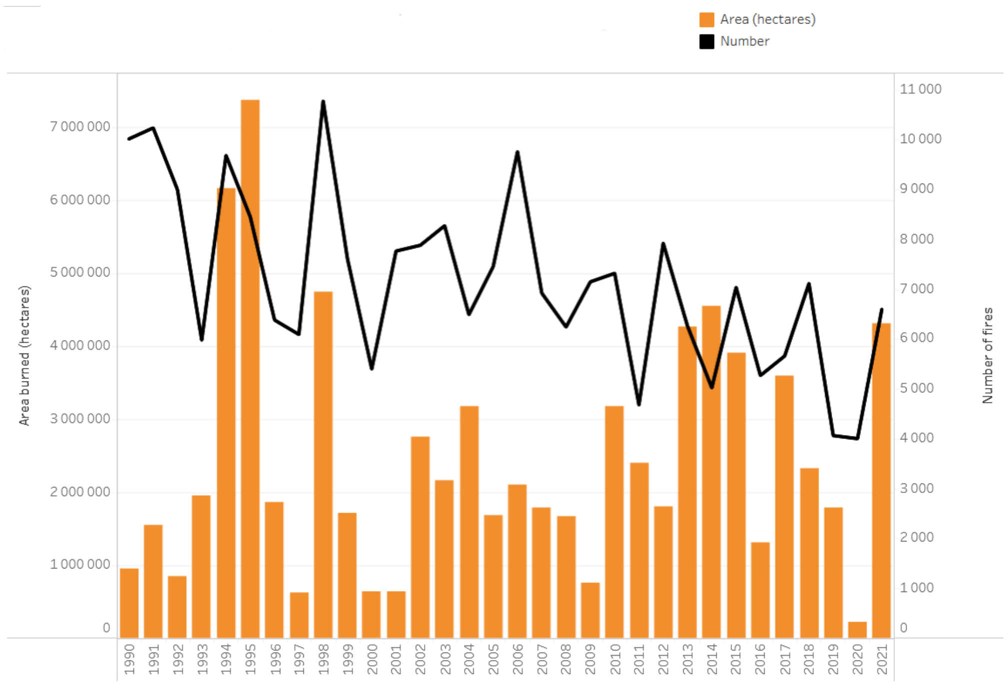

Figure 1.1 Forest area burned and number of forest fires in Canada from 1990 to 2020. Reprinted from [2].

Faced with these risks of devastation, the company Lux Aerobot was created in 2017 to help in the fight against the spread of an incipient fire by taking photos and imagery of the Earth. Lux is a space robotics company specializing in the design, manufacture and operation of atmospheric satellites for Earth observation. Their atmospheric satellites are high-altitude balloons equipped with a stratospheric gondola of hardware containing a wide range of scientific instruments such as remote sensing equipment. The use of a high-altitude balloon and the associated gondola consists in three main stages: Launch (stage 1), Data acquisition (stage 2), and Recovery (stage 3). These three stages are presented in Fig. 1.2.

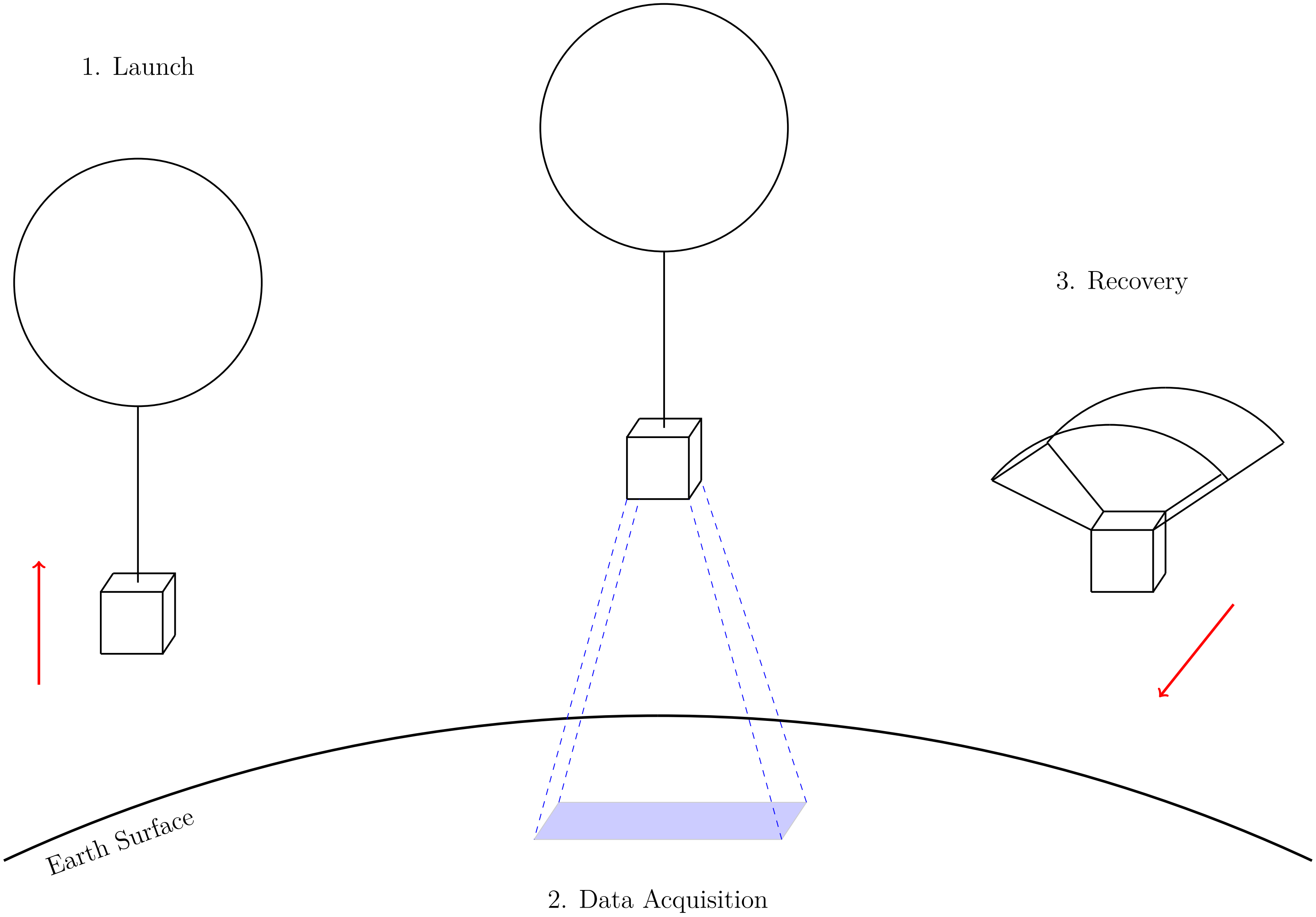

Figure 1.2 Stages of flight

This project focuses on the third stage, which involves a controlled descent and landing in order to find the gondola and the acquired data. The main research objective will focus on the design and the development of an autonomous guided descent vehicle that will allow better control of the descent and landing of the gondola, while considering the atmospheric conditions, and from a nominal altitude ranging from 20 to 25 km. It is then possible to ensure the safe retrieval and reuse of the gondola, and to favour a landing zone far from populated areas and on favourable terrain.

The recovery phase begins when balloons are unhooked from the gondola, and the flight hardware deployed, and ends when the gondola touches the ground. This flight is made of three phases :

1. Gliding, from the deployment point to a targeted circular area;
2. Spiral down, i.e., losing altitude while staying close to the targeted landing point, by rotating around the point;
3. Flare, which consists in landing in straight line, like a plane.

1.2 Research objectives

The main research objective of this work can be defined as:

Design an optimal path planner and a controller for the descent of a stratospheric gondola.

This objective is composed of four different sub-objectives (SO), which are:

SO1 Identify the most suitable technological solution to fly the gondola to a desired location.

SO2 Characterize the 3D dynamic behaviour for the identified system.

SO3 Build the control architecture for optimal path planning and trajectory tracking.

SO4 Validate the performance of the control strategy by simulation for different atmospheric conditions.

1.3 Master thesis outline

In Chapter 2, a state of art concerning aerial guided recovery systems, aircraft control methods and UAV path planning will be established. The choice of a parafoil will be explained, as well as its advantages and disadvantages compared with other existing solutions. Then, controllers from the literature will be presented, from high level (global structure) to low level (actuators and control surfaces). Finally, path planning for flying systems such as planes, fixed-wing and parafoils will be explored. In particular, references on trajectory optimisation and existing algorithms will be given. This chapter will address **SO1**.

In Chapter 3, a dynamical model of a gondola with parafoil will be presented. The dynamical equations as well as the aerodynamic forces and moments, will be introduced, as well as the different frames used (NED, body, air); a nonlinear Simulink model will be built based upon these equations. Then, equilibrium points will be calculated and, from the linearization of the dynamical equations around those, flight modes will be deduced and compared with ones from the literature. A quantitative study of the correlation between system mass, parafoil surface and steady-state velocity will be established. This chapter, by building the 3D simulator, and thus addressing **SO2**, is also a first step towards **SO3** (comparing NLS and PD path planners) and **SO4** (testing the hybrid controller in real-conditions).

Chapter 4 will focus on path planning and trajectography. In particular, for the first phase of flight (gliding), an optimal trajectory for a simplified problem will be found via the deterministic and continuous-time method of Pontryagin minimum for unknown final time, implemented thanks to nonlinear shooting. This method shall give an open-loop control law, which will then be used as the nominal trajectory for simulated flights. In order to be able to implement the controller onboard, another method will be implemented for the first phase, using dynamic programming. Its performance will be compared with that of the nonlinear shooting generator over five use-case trajectories, representing a set of different initial conditions (position, velocity), and different final conditions (location of the targeted area), in the 3D simulator previously built and without wind. The complementarity of the two methods will be highlighted, in order to emphasize the advantages of their combination to build an optimal and wind-robust controller, without the need to incorporate a wind model in the deterministic dynamical equations. This will be completed by an analysis of the subsequent phases (spiral down) and (landing) as well, and in particular an approach to the transition between spiralling down and landing using Bézier curves will be introduced. Therefore, **SO3** will be addressed.

In Chapter 5, the influence of the wind and the robustness of the algorithm will be tested in the 3D simulator. From the Dryden turbulence and wind model, different wind profiles will be generated and the effectiveness of the hybrid algorithm with onboard recomputing will be emphasized. This chapter addresses **SO4** and hence concludes the design and validation of the complete guidance, navigation and control (GNC) algorithm.

Finally, Chapter 6 is the conclusion of this research work, and presents a summary of the results, and the limitations and future possible improvements.

CHAPTER 2 LITERATURE REVIEW

The structure of the present literature review will follow the order of the research sub-objectives: firstly, the choices which have led to choose a parafoil system will be exposed. Then, general aircraft control architectures, and especially those of parafoil systems are to be presented; and finally, GNC algorithms, as well as optimal path planning algorithms, will be studied.

2.1 Selection of the most adequate mechanical system

The first step in this project was to choose the mechanical solution that would allow to bring down the gondola. Here we set out the selection criteria, as well as some of the existing solutions that were examined. We then explain why the current solution was chosen.

2.1.1 Selection criterion

The paramount quantity is the glide ratio GR, defined as the quotient between the horizontal and vertical inertial velocities:

$$\mathrm{GR} = \frac{\text{Horizontal mean speed}}{\text{Vertical mean speed}} \tag{2.1}$$

The higher the glide ratio of a solution is, the better. Assuming that the system does not gain altitude, the glide ratio will always be positive. A solution is considered useless if it does not achieve a glide ratio greater than 1, and efficient if it achieves a glide ratio greater than 2.

Other secondary criteria will also be taken into account: if possible, the system selected should not be too heavy (limited flight weight) or too difficult to implement, as Lux currently has a prototype gondola that cannot be completely redesigned for the sole use of guided recovery. Finally, a system that is as inexpensive as possible to implement would be a plus.

2.1.2 Existing solutions

This first subsection is dedicated to giving an overview of the existing solutions that may be used in our case.

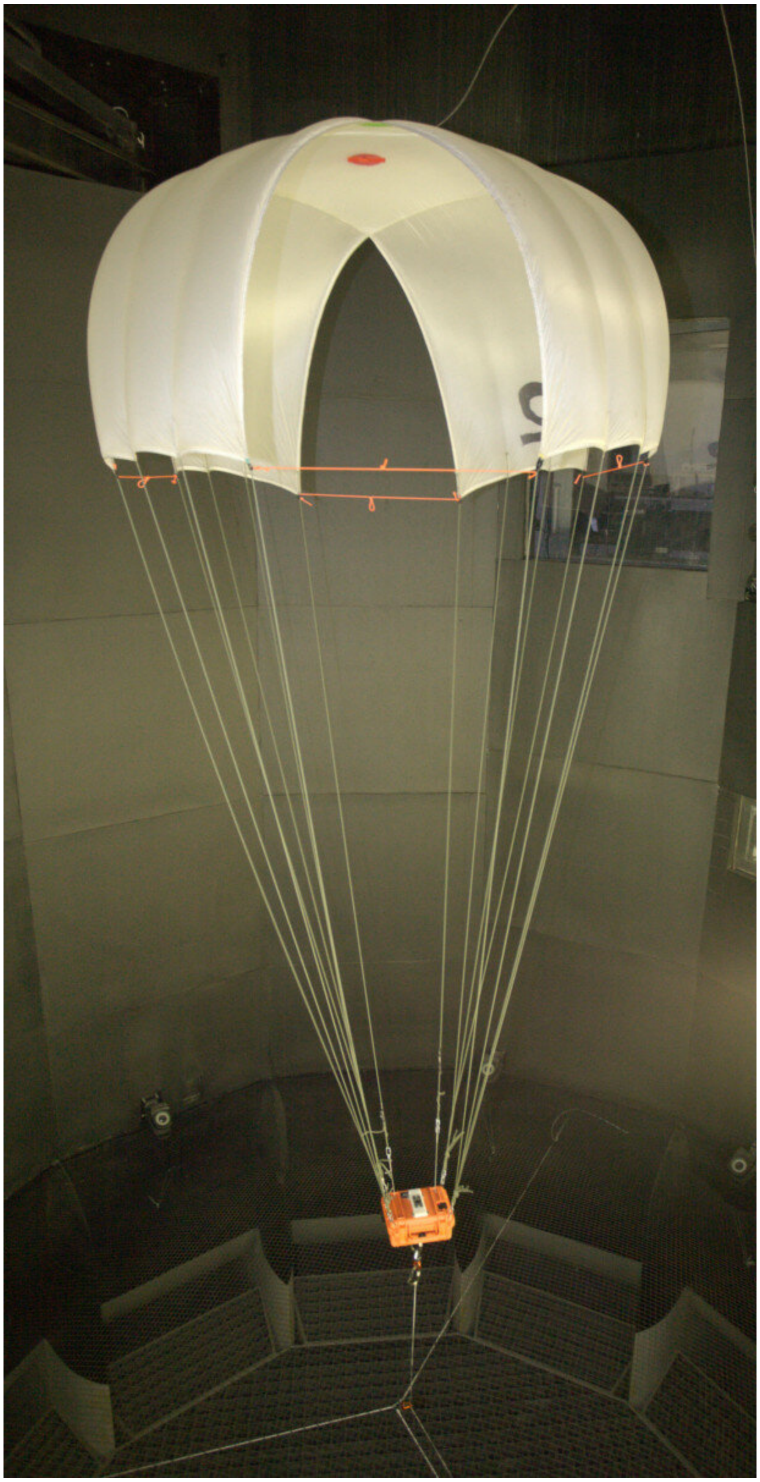

Figure 2.1 Cruciform parachute. Reprinted from [3]

Cruciform parachute [3, 12]

The forward glide ratio of a cruciform parachute (Fig. 2.1) is achieved by shortening the risers on one side, and steering is achieved by varying the lengths of the adjacent risers.

The advantages of this solution are its easy integration to the system already built by Lux, and that it needs a single actuator to steer it. The disadvantages are its low glide ratio (from 0.25 to 1 while gliding, almost null while turning), and the high sensitivity to wind. As the glide ratio is the most important variable to consider, and the controllability is almost made impossible because of the wind, it has been decided to exclude this solution.

Round parachute with variable drag [4]

The descent rate of a round parachute with variable drag (Fig. 2.2) is variable and can be modified by varying the geometry by opening the hole at the top, which will increase the drag. Steering is achieved through riding winds on the way down.

The main advantage is the ease with which it can be integrated into Lux's current system. The disadvantages are that the glide ratio is close to 0, and that this solution is very dependant on favourable wind profiles. For this reason, the solution is not considered viable in our case.

Round parachute with steering [5]

A round parachute with steering (Fig. 2.3) is quite similar to a round parachute with variable drag, but in this particular case, the system uses variable length pneumatic muscle actuators (PMA) to vary the lengths of its risers. Moreover, they differ in the method of control : the size of the hole at the top of the parachute is constant, and the drag cannot be changed. But the actuators allow the system to move horizontally, which increases glide ratio compared with a variable-drag solution.

The main advantages of this solution are the easy integration to Lux's current system, and the fact that there is no forward orientation: the system is therefore capable of moving in four independent directions. Yet, there are a few drawbacks to this solution: even though the glide ratio is better than in the previous solutions, it is still too low (from 0.7 to 1). In addition, using this solution requires accurate wind predictions and four separate actuators (one per riser).

Since the glide ratio is still below 1, the number of actuators is high, and the mass and energy of the gondola devoted to guided recovery is very limited in flight, this solution was not considered.

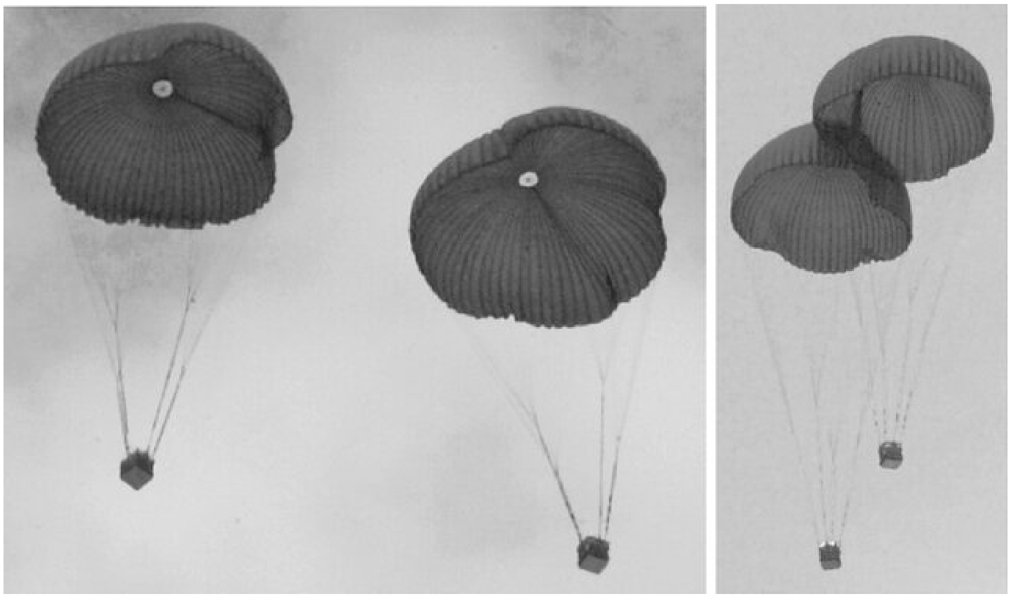

Figure 2.2 Round parachute with variable drag. Reprinted from [4].

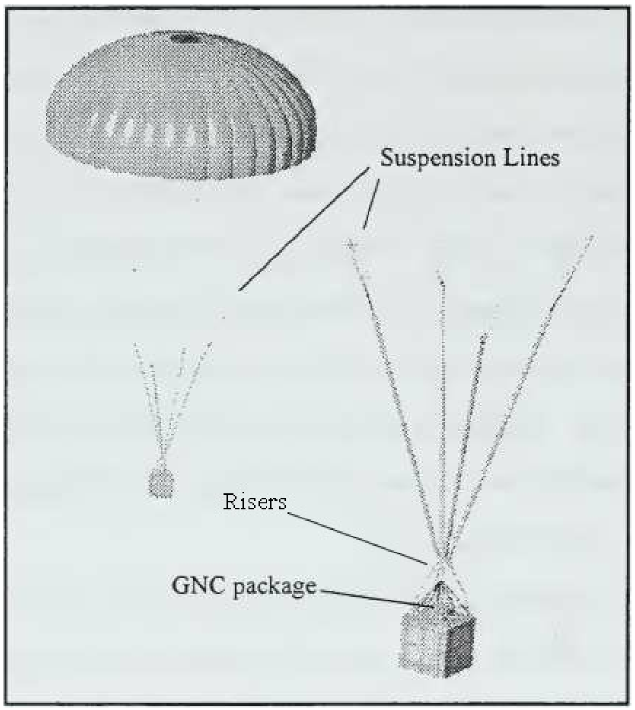

Figure 2.3 Round parachute with steering. Adapted from [5]

Fixed-wing glider [6]

The fixed-wing glider (Fig. 2.4) is a very different solution from the previous ones. Here, avionics and payload are integrated in a glider chassis which is then attached to the balloon.

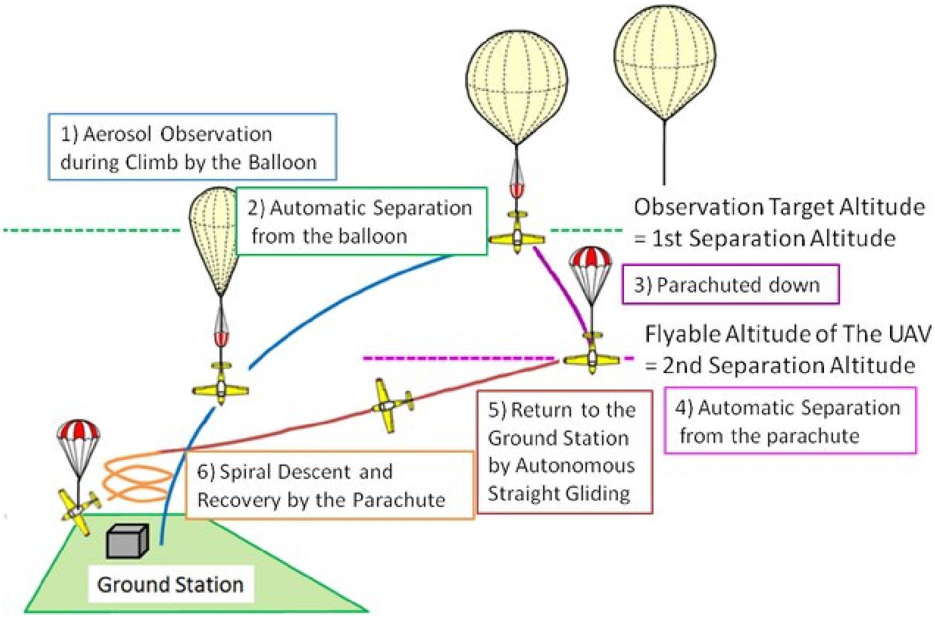

Figure 2.4 Fixed-wing glider from [6].

The main advantages are a high glide ratio (around 4, or even 5), and the possibility of including propellers for thrust, thereby increasing the glide ratio. However, this would require a complete redesign of the gondola in order to integrate them. A more complex payload integration than current gondola designs would also be needed.

This solution has shown that reaching high glide ratio is possible, and that powering an existing solution to further improve it would be interesting. However, this solution was not selected as it would be too restrictive for the other teams working on the project: the complete design cannot exclusively depend on the guided recovery team.

Comparative table: parafoil, rotorcraft, fixed wing [1]

In Lee's PhD thesis [1], several other solutions have been explored for return-to-point vehicle concepts. The final table of the bibliography of that PhD thesis can be adapted to the present case. Indeed, some of the disadvantages, such as long flight duration, are actually advantages in this study. A summary of the results is presented in Table 2.1.

Solution	Advantages	Disadvantages
Parafoil	Glide Slope Manoeuvrability Slow descent rate Long flight duration	Susceptibility to wind
Rotorcraft	Slow descent rate	Cost and Complexity Heavy weight
Fixed wing	Manoeuvrability Long range	Cost and Complexity Heavy weight

Table 2.1 Return-to-point vehicle concepts. Adapted from [1].

It is worth noting that opting for a heavy solution is not feasible, as a substantial portion of the gondola's weight cannot be allocated solely to the final phase of the flight. Furthermore, the solution needs to be cost-effective and straightforward to ensure reproducibility for multiple launches in Australia, Quebec, and Western Canada.

All these considerations make the parafoil the best solution. The next step of this literature review is devoted to collecting more details about the properties of a parafoil system.

2.1.3 The parafoil system

This subsection aims at giving a deeper explanation of the recovery capabilities of parafoil system, as shown in Fig. 2.5.

Figure 2.5 Visualization of parafoil systems. Reprinted from [7] and [8].

Advantages

There are a whole range of positive features that make a parafoil the ideal solution:

- It is the most widespread solution found in the literature.
- It enables longer duration flights, which gives room for further decision making on the landing location.
- Its feasibility in high altitude has already been proven [8].
- The control laws depend only marginally on the geometry of the gondola, the main factor being its mass.
- The system has a predetermined velocity vector direction, facilitating landing gear design.
- The glide ratio is very good, generally between 1 and 4, according to [7] and [8].
- This glide ratio can even be improved up to 6 by powering of the system. Such a solution has been tested in [9], and the authors demonstrated that a 50% increase in glide ratio, compared with an unpowered parafoil system, can be achieved. Their system is illustrated in Fig.2.6.

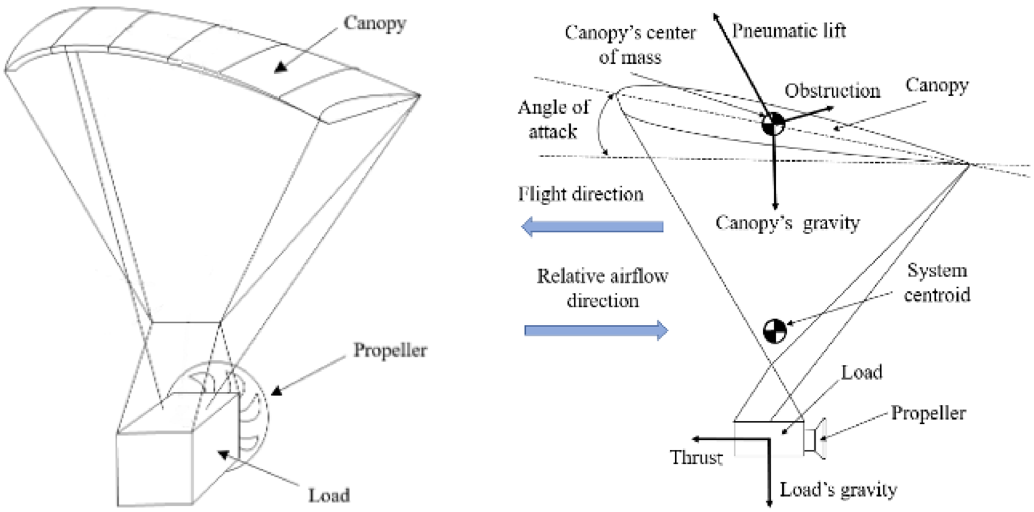

Figure 2.6 Representation of a powered-parafoil system. Adapted from [9]

Disadvantages

The parafoil solution has shown a number of advantages, but there are still some negative characteristics listed below:

- The parafoil is less stable at high altitudes, due to the much lower density of the air, but it improves with descent.
- One forward direction is privileged, and a change of direction requires a shift in orientation.
- It is highly vulnerable to wind.

Possible solutions to mitigate the disadvantages

The above mentioned problems can be resolved thanks to engineering choices. For instance, the lack of stability at high altitudes can be mitigated with a two-phase descent. The first phase would be a vented descent, enabled by a variable-radius hole in one of the balloons, so that the gondola loses altitude rapidly and almost vertically. Once the altitude is low enough, the parafoil can be deployed. The fact that one forward direction is privileged can also be seen as an advantage if the wind is favourable for taking the gondola to its planned final point. Otherwise, it will depend on the robustness of the controller and the weather conditions, which are the main elements that cannot be controlled. If the wind is too strong, the launch will have to be aborted.

2.2 Modelling

2.2.1 The question of the number of degrees of freedom.

Several articles in the literature have investigated the modelling of parafoil systems, and in particular the number of degrees of freedom (DOF) required to obtain an accurate model. Dynamic models with 9 DOF are studied in [13] and [14] and are generally considered to be comprehensive for rigid modelling; they assume a total decoupling between the gondola and the canopy, as shown in Fig. 2.7. The canopy can rotate in any direction relative to the parafoil, as the system is not non-deformable but point C introduces 3 relative DOF; the figure shows a relative pitching, but relative yawing and rolling can happen as well.

A comprehensive summary of these different models is presented in [15], where it is concluded that a 6-DOF model (*i.e.*, considering the system as a single rigid body) is sufficient if the

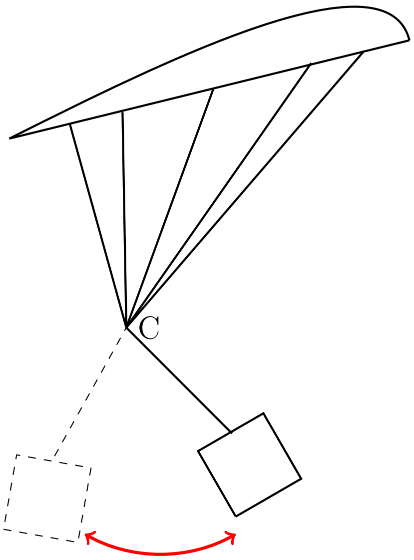

Figure 2.7 Relative pitching between canopy and parafoil in models with more than 6 DOF

GNC algorithm uses only position and velocity feedback. Otherwise, a 7-DOF is required if the control algorithm requires sensor information about the payload orientation.

Since the better compromise for modelling the parafoil system is to consider it as a rigid body, which means that the set {Canopy + Ropes + Parafoil} is non-deformable and that there is no relative movement between any component whatsoever, this assumption will be maintained throughout the whole study; in particular, the relative movement or difference in orientation between the canopy and the parafoil will be neglected.

2.2.2 Dynamic characterization

The control laws are based on nonlinear dynamic equations, known as the flight equations, which characterize the behaviour of the moving body through air. They remain broadly similar regardless of the body under consideration. The force and moment equations are given in [16] and more completely in [17]. They involve aerodynamic terms which vary from one system to another. In the specific case of a parafoil system, the aerodynamic terms are given in [18], which will be the reference model for the current research project. The method for obtaining these coefficient by experimental identification of the system is developed in [19]. The present Ward's model also uses empirical data to identify the aerodynamic coefficients. A more detailed model, using the concept of apparent mass, was introduced in 2012 in [20], and further detailed in [21]. This more elaborate modelling is widely accepted in the literature:

[1] and [22] base their works on this paradigm, which is however not the core of this work.

Nonlinear dynamic models of flying systems can be linearized around a certain flight equilibrium, allowing the modes of the system to be determined. References for this methodology are given in [23–25].

2.3 Parafoil systems control methods

This section presents a brief review of the most common control methods for aircraft and, in particular, for parafoil systems. It begins with a general review of control laws in the first subsection, and then presents the optimal aspects in terms of trajectory planning.

2.3.1 Control laws

The literature has extensively discussed the planning of trajectories for parafoil systems. In all cases, unpropelled systems must always land facing into the wind. The approach chosen to construct a trajectory consists of three distinct phases:

- The first phase consists in gliding to a circular area centred on the desired landing point.
- The second phase is a spiral down, aiming at decreasing the vertical velocity without moving away from the desired landing point.
- The last phase is a plane-like landing, facing the wind.

This type of trajectory is introduced in [10] and explored in details in [11]. They both rely on simple geometric figures, as shown in Figs 2.8 and 2.9. Both figures are based on the same principles: landing must be into the wind, and gliding aims to reduce the total energy of the system; kinetic energy by loosing speed, and potential energy by losing altitude. The acronym EMC refers to Energy Management Circle, which can be covered several times if necessary. The latter figure shows the different parts of the trajectory: from an initial point A (blue) to a target point G (red), and taking into account its initial direction, the system must follow a path generated by several points, B to F (green). These are constructed using only circles and straight lines, according to the Dubins path principle.

Other types of trajectories can be found in [18] and [8]. Consisting of final turn and a loss of altitude in a straight line, this last stage is not the most critical of the flight in terms of path planning.

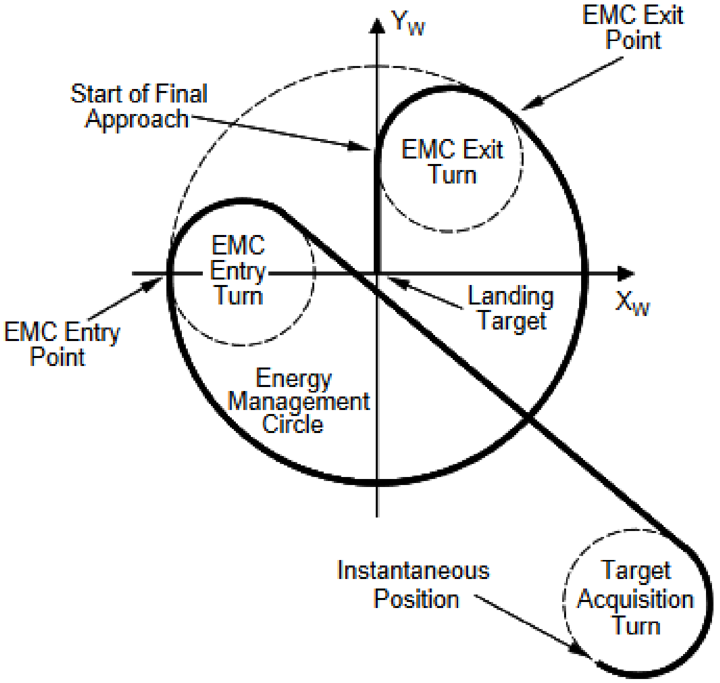

Figure 2.8 Construction of trajectories from [10]

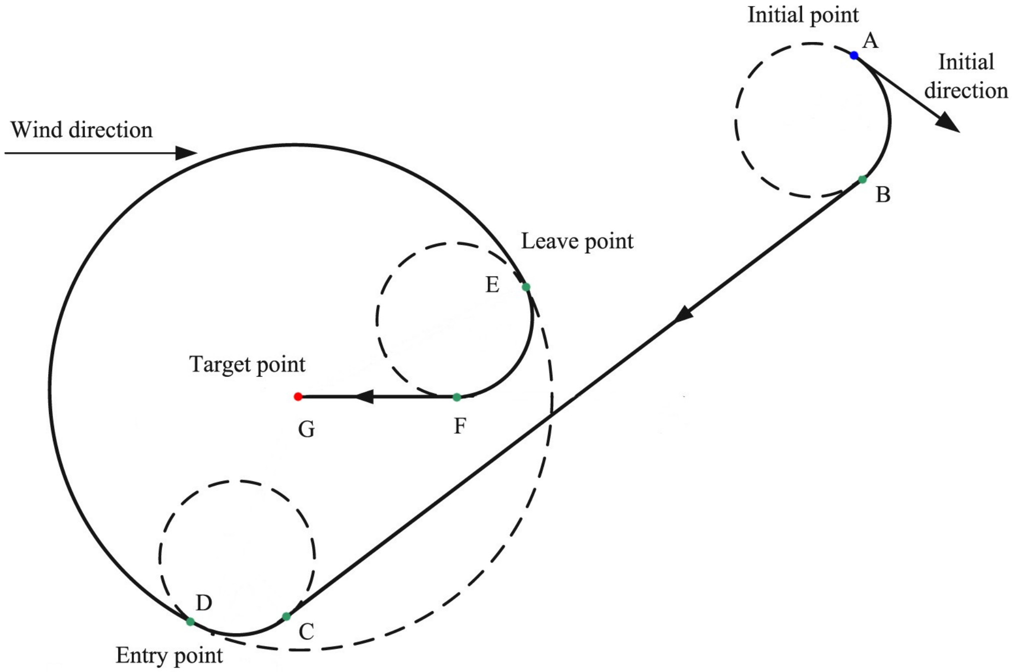

Figure 2.9 Construction of trajectories adapted from [11]

Parafoil systems usually have two control inputs, which are symmetric and asymmetric brake deflection. Symmetric deflection causes the system to gain or lose altitude and therefore increase or decrease its speed, while asymmetric deflection changes the yaw and roll angles at the same time. This method of controlling the parafoil is presented in [1], and further detailed in [21].

To begin with, a simple GNC algorithm for fixed wings can be found in [26], and an application, as well as a proof of efficiency for complex systems, is presented in [27]. It is essentially based on coupled control of yaw and roll angles using plane-like actuators.

A large number of more complex and efficient controllers have been proposed in the literature, ranging from glide slope control [28] using an additional input, to predictive control [29]. Many optimal control algorithms have also been developed. For instance, Slegers (2009) [30] introduced practical considerations by incorporating sensor modelling and conducting experiments, which will be useful to Lux when the time comes to implement the control algorithms developed in the present work into an embedded system.

2.3.2 Optimal path planning

The first phase of the flight, *i.e.,* gliding, can be designed to be optimal regarding several constraints. In particular, finding the "best" continuous trajectory is an optimal control problem that has already been explored in the literature, for both continuous and discrete trajectories. All the methods presented in this subsection use flat-earth (NED) coordinates which can be obtained from GPS coordinates thanks to the LLA-Flat Earth algorithm developed in [31].

Continuous time: Nonlinear shooting

The main method for transforming a continuous-time optimal control problem into a nonlinear programming optimisation problem, especially when the final time is unknown as is the case in this work, consists of a nonlinear shooting method, widely developed in the reference books [32–34] and based on Pontryagin's minimum principle. A briefer version is given in [35], and the formalism used in this thesis will be based on this latter. These references present the theory and several applications for optimal control problems, in engineering and other fields. It has already been implemented on practical cases in several optimal control books and articles, such as [36] and [37]. The efficiency of multiple nonlinear shooting (meshing the trajectory into several smaller ones and computing a nonlinear shooter per each) was demonstrated as well in [38, 39]).

Murali *et al.* (2013) highlighted the benefits of optimal control and emphasized the strategic choice of targeting an area rather than a specific point during the initial flight phase [40]. They later used nonlinear shooting method with a parafoil system for the first time [41], as well as Suresh Babu *et al.* in [42].

All these considerations taken into account tend to demonstrate that, despite a significant computational cost because it is a continuous method, nonlinear shooting is very promising in Lux's problem of finding the optimal path.

Discrete time: Dynamic programming

Another way to find an optimal path is to use discrete methods such as dynamic programming, based on Bellman's equation (first introduced in the 1950s) [43, 44]. Since then, the efficiency of this method for trajectory planning for robots [45–47] and vehicles such as aircraft [48] has been widely recognised. It consists in finding the best path in a graph, whose weights can even be stochastic, [49]). In the literature, dynamic programming was first tested on parafoil system path planning in 2013 by Cleminson [50]. In this article, the author chooses to use a snap-to-grid technique, as the mesh is Cartesian but the system is controlled from its yaw rate.

Because dynamic programming is less demanding than continuous-time methods, it allows the user to increase the complexity of the problem. This can be done by studying higher-dimensional models (searching for the optimal path in 3D instead of simpler 2D models, as is the case in [51]); or by using more advanced optimization methods, such as genetic programming [52].

All these works show that even though dynamic programming is not as optimal as nonlinear shooting (and is highly dependant on the meshing), it is a very promising method because, under certain assumptions, it can be used for on-board calculations. What's more, this method is complementary to non-linear shooting, the shortcomings of one being offset by the advantages of the other.

Other methods: Gauss pseudospectral, Bézier curves, Dubins path

The effectiveness of Gauss pseudospectral method for implementing optimal control in parafoil systems, especially in the presence of obstacles, has also been demonstrated [53, 54]. Additional geometric methods such as Bézier curves [55–57] and the Dubins path [58] have also proved to be effective alternatives. Particle swarms were also briefly studied in [59].

These methods will not be considered further for phase 1, as the Gauss pseudospectral method is quite similar to nonlinear shooting (from a continuous optimal control problem to a nonlinear programming one, strong dependence on initial conditions, etc.) and their efficiency was proved to be equivalent in the literature. Bézier curves are interesting for linking spiralling down and landing phases, but are not the best choice for the gliding phase, as they would require the selection of many control points and would still not be as optimal as nonlinear shooting. The same goes for Dubins method which relies on simple geometrical figures more than on optimal paths. As far as particle swarms are concerned, the development of new theoretical optimisation techniques is not at the heart of this thesis work, but it is worth mentioning that these developments are being applied to parafoil systems.

Dealing with the wind

Robustness aspects taking wind into account can be found in [60] for parafoil systems, and further studies showing optimal/robust path planning for planes are presented in [61–64]. It should be remembered that most of the contributions concerned simplified models with a constant wind direction in an aircraft at a constant altitude. In the area of wind considerations, the main focus has been on system robustness rather than optimality [65]. Yet, as parafoil systems are highly vulnerable to wind, and even their nominal trajectories must be designed according to the wind (as seen previously with landings), it is crucial to have the ability to change the trajectory during the flight, especially if optimality is a goal [66].

2.4 Conclusion of the literature review

2.4.1 Synthesis of literature

The following is a brief summary of the literature review, in terms of hardware technology solutions and the choice of optimal controllers and trajectory planners.

Hardware

The use of a parafoil system, in particular a motorised parafoil system, is the most appropriate solution to Lux's problem. It will make it possible to manage long flights with a high glide ratio, the control laws will not be more difficult to implement, and both weight and energy consumption are limited. It is possible to customise this solution by choosing the number of actuators, selecting the propeller, and the attachment to the gondola can be easily adapted to almost any existing mechanical design. In the event of excessively strong winds, *i.e.*, when the advantage becomes a disadvantage, rejecting these perturbations can be achieved by combining a robust PID controller and taking these perturbations into account when planning the trajectory.

The control laws can be compared to those of a fixed-wing aircraft or an aircraft equipped only with elevators and flaps. Two or four actuators can be used to pull the strings, and a PID (or even PI or PD) controller is sufficient to control the system.

Control software

A few controllers have been presented in the literature. Most of the papers focus on aerodynamic coefficients or system modelling and identification, while some of them introduce a new control input or methodology. In particular, optimal controllers rely mainly (and often only) on dynamic programming algorithms or nonlinear shooting methods, and propose a very simplified wind model.

2.4.2 Contribution & brief methodology

The choice of a parafoil system explained in this literature review addressed **SO1**. In Chapter 3, a 3D model of a parafoil system for the literature will be built with Matlab/Simulink. Qualitative and quantitative aspects of the total mass of the system and the parafoil system will be discussed, and a study of the aerodynamic modes of the system will be carried out. This will address **SO2**.

Secondly, breaking from the mould of the existing literature, this work will strategically exclude wind modelling in the optimal formulation. Instead, a hybrid method employing a controller to first follow the nonlinear shooting trajectory, with the dynamic programming algorithm recalculating new optimal trajectories on-the-fly when deviations occur due to environmental influences will be implemented. In addition, the dynamic programming algorithm introduced presents a new formulation that offers vastly improved physical coherence compared to traditional Cartesian mappings. It eliminates the need for snap-to-grid techniques, resulting in a piecewise continuous trajectory that accurately represents flights and their costs . To evaluate its effectiveness, the developed algorithm will be compared to a reference algorithm based on nonlinear shooting, which is widely recognized in the literature. Then, the algorithm will be completed in phases 2 and 3 with a spiral descent controller and landing controller; these two elements will be linked by a transition algorithm using Bézier curves for a smooth transition between phases 2 and 3. At the end of Chapter 4, the control architecture for optimal path planning will be completed, thus addressing the first part of **SO3**.

In chapter 5, the trajectory tracker will be built and first tested in the 3D simulator without wind, completing **SO3**. Then, different wind profiles will be generated to test the previously built control algorithm, from parafoil deployment to landing. **SO4** will consequently be achieved.

CHAPTER 3 DYNAMIC MODEL OF THE GONDOLA

This chapter presents a dynamic model of the parafoil system, established using the equations of flight mechanics adapted to parafoil systems as described in the literature by Ward [28]. A Simulink model is then built based on these equations, and linearised with respect to the longitudinal and lateral flight equilibria. The matrices obtained are then compared with those of several other studies in the literature [21].

3.1 Motion equations

3.1.1 Definition of frame rotation matrices and notations

NED frame

The first frame of reference is the NED (North-East-Down) frame, which assumes that the Earth is locally flat. This orthnormal frame is considered to be inertial and its origin, denoted O_{NED}, is an arbitrarily chosen point on Earth's surface. The axes are defined as :

- $\mathbf{x}_{\text{NED}}$ oriented toward the geographic North,
- $\mathbf{y}_{\text{NED}}$ oriented toward the geographic East,
- $\mathbf{z}_{\text{NED}}$ oriented toward inside the Earth.

Body frame

The second frame of reference is the body frame centred in O_b, the centre of mass of the system, and its axes are given by:

- $\mathbf{x}_b$ oriented forward,
- $\mathbf{y}_b$ oriented to complete the direct trihedron,
- $\mathbf{z}_b$ oriented toward the gondola.

Direct Cosine Matrix

The transformation between the NED frame and the body frame is given by the succession of three basic rotations (Fig. 3.1):

1. Rotation by an angle ψ around the $\mathbf{z}_{\text{NED}}$ axis.
2. Rotation by an angle θ around the new $\mathbf{y_1}$ axis.
3. Rotation by an angle ϕ around the new $\mathbf{x_2}$ axis.

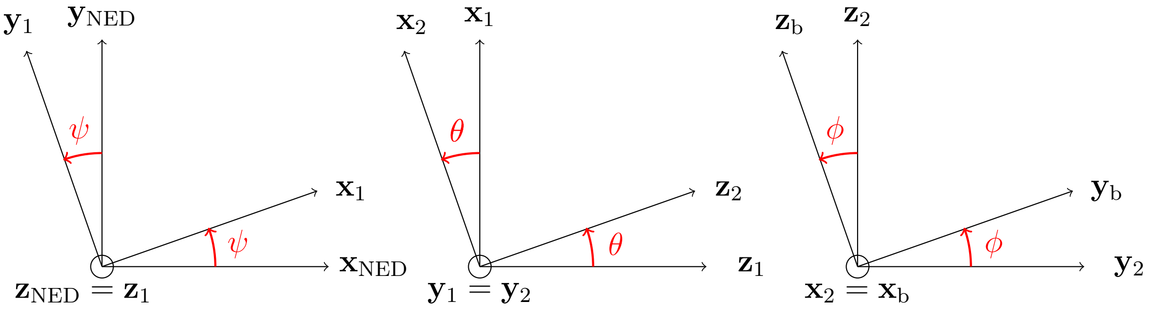

Figure 3.1 Euler angles from NED to body frame

These angles are called the Euler angles, and by convention, they must verify $\psi \in (-\pi, \pi]$, $\theta \in [-\pi/2, \pi/2]$, and $\phi \in (-\pi, \pi]$. The resulting rotation matrix is called the direct cosine matrix (DCM) and is obtained by the multiplication of the three basic rotation matrices:

$$\mathbf{DCM}(\phi,\theta,\psi) = \begin{bmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \sin\phi\cos\theta \\ \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi & \cos\phi\cos\theta \end{bmatrix}. \tag{3.1}$$

Wind and stability frames

The wind and stability frames are defined in terms of relative velocity between the parafoil system and the wind, *i.e.*, the aerodynamic velocity, denoted in the body frame by :

$$\mathbf{V}^b_{b/w} = \begin{bmatrix} u \\ v \\ w \end{bmatrix}. \tag{3.2}$$

The true airspeed V_T, the angle of attack α and the sideslip angle β are then defined by:

$$V_T = ||\mathbf{V}_{b/w}^b||_2 = \sqrt{u^2 + v^2 + w^2}, \tag{3.3}$$

$$\alpha = \text{atan}\left(\frac{w}{u}\right), \tag{3.4}$$

$$\beta = \text{asin}\left(\frac{v}{V_T}\right). \tag{3.5}$$

The stability frame of reference $(\mathbf{x}_\text{s}, \mathbf{y}_\text{s}, \mathbf{z}_\text{s})$ is centred in O_b and is obtained by a rotation of the body frame through an angle $-\alpha$ around $\mathbf{y}_\text{b}$, while the wind frame $(\mathbf{x}_\text{w}, \mathbf{y}_\text{w}, \mathbf{z}_\text{w})$, which is ideal to characterize the air movement of the gondola, is obtained by a rotation of the stability frame through an angle β around $\mathbf{z}_\text{s}$. These rotations are illustrated in Fig. 3.2.

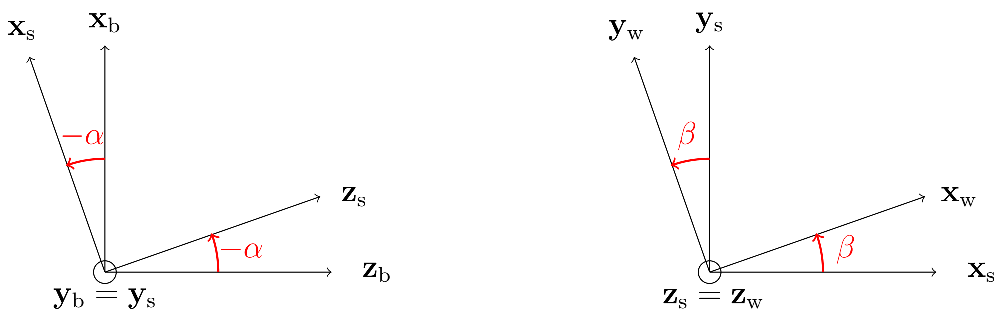

Figure 3.2 Rotations from body to stability and wind frames

3.1.2 Euler kinematic equations

The angular velocity of the body frame with respect to the NED frame, and expressed in the body frame, is denoted by:

$$\mathbf{\Omega}_{b/\text{NED}}^b = \begin{bmatrix} p \\ q \\ r \end{bmatrix}. \tag{3.6}$$

The Euler kinematic equation then links the angular velocities to the derivatives of the Euler angles [31]:

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & -\sin\phi & \cos\phi\cos\theta \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}. \tag{3.7}$$

3.1.3 Equations of forces

Denoting m the total mass of the system (parafoil + gondola), the vector force equation is given by:

$$m\left(\dot{\mathbf{V}}^b_{b/w} + \mathbf{V}^b_{b/w} \times \mathbf{\Omega}^b_{b/\mathrm{NED}}\right) = \mathbf{F}^b_A + \mathbf{F}^b_T + \mathbf{DCM}(\phi, \theta, \psi)\mathbf{g}^{\mathrm{NED}}, \tag{3.8}$$

where × denotes the cross product, $\mathbf{F}^b_A$ and $\mathbf{F}^b_T$ the aerodynamic and thrust forces respectively, and $\mathbf{g}^{\mathrm{NED}} = [0\,0\,g_0]^\top$ the gravity vector with g_0 the standard gravity. The scalar form of (3.8) leads to the three scalar equations:

$$m(\dot{u} + qw - rv) = X_A + X_T - mg_0 \sin\theta, \tag{3.9a}$$

$$m(\dot{v} + ru - pw) = Y_A + Y_T + mg_0 \sin\phi\cos\theta, \tag{3.9b}$$

$$m(\dot{w} + pv - qu) = Z_A + Z_T + mg_0 \cos\phi\cos\theta, \tag{3.9c}$$

with X_A, Y_A and Z_A the components of the aerodynamic forces experienced by the system, and X_T, Y_T and Z_T the components of the thrust forces. Thrust can be provided by the propellers, which for the moment are not limited to the $\mathbf{x}_\mathrm{b}$ axis only, as it will depend on the mechanical constraints.

3.1.4 Equation of moments

The inertia matrix $\mathbf{I}$ of the system expressed at its centre of mass and along the body frame axis is denoted:

$$\mathbf{I} = \begin{bmatrix} I_{xx} & 0 & -I_{xz} \\ 0 & I_{yy} & 0 \\ -I_{xz} & 0 & I_{zz} \end{bmatrix} \tag{3.10}$$

where I_{xx}, I_{yy} and I_{zz} represent the moment of inertia and I_{xz} the product of inertia. The products of inertia I_{yz} and I_{xy} are supposed to be zero as the body plane $(\mathbf{x}_b, \mathbf{z}_b)$ is a plane of symmetry. The vector moment equation is then given by:

$$\mathbf{I}\,\dot{\mathbf{\Omega}}^b_{b/\mathrm{NED}} + \left(\mathbf{I}\,\mathbf{\Omega}^b_{b/\mathrm{NED}}\right) \times \mathbf{\Omega}^b_{b/\mathrm{NED}} = \mathbf{M}^b_A + \mathbf{M}^b_T, \tag{3.11}$$

where $\mathbf{M}^b_A$ and $\mathbf{M}^b_T$ denote the aerodynamic and thrust moments respectively. The scalar form of (3.11) leads to the three scalar equations:

$$I_{xx}\dot{p} + (I_{zz} - I_{yy})\,qr - I_{xz}(pq + \dot{r}) = L_A + L_T, \tag{3.12a}$$

$$I_{yy}\dot{q} + (I_{xx} - I_{zz})\,rp - I_{xz}\left(r^2 - p^2\right) = M_A + M_T, \tag{3.12b}$$

$$I_{zz}\dot{r} + (I_{yy} - I_{xx})\,pq + I_{xz}(rq - \dot{p}) = N_A + N_T, \tag{3.12c}$$

with L_A, M_A and N_A the components of the aerodynamic moments experienced by the system, and L_T, M_T and N_T the components of the thrust moments.

3.2 Expressions of the aerodynamic terms

3.2.1 Control inputs

The aerodynamic model of the parafoil is developed in [18] where numerical values of aeronautical coefficients are given to facilitate the implementations and simulations. These numerical values are needed to build the first simulator with Simulink. All constant aerodynamic quantities must be determined experimentally. For further details on system identification, refer to Appendix C.

In addition to the definitions of the angle of attack (3.4) and sideslip angle (3.5), we introduce the variables of control:

- δ_A denotes the asymmetric brake deflection, enabling the gondola to turn, and is limited to ±5 cm by actuator saturation;
- δ_B denotes the symmetric brake deflection, enabling the gondola to rise or fall, and is limited to ±5 cm by actuator saturation;
- δ_I is the incidence control input, which will be considered equal to zero in the present study.

Thanks to these definitions, a modified angle of attack, which takes into account the symmetric brake deflection with a first-order relation is introduced in [18] and given by:

$$\alpha' = \alpha + \alpha_B \delta_B. \tag{3.13}$$

Intuitive representations of the effects of the control inputs δ_A and δ_B are given in Figs 3.3 and 3.4. In each case, the parafoil is shown in the same position, which means that the wind frame is identical, as it offers a better visualization of the relative movement between the body frame and the wind frame. Using four anchors confers a better stability and reduction of relative yawing and pitching between the parafoil and the gondola (see 9 DOF models discussed previously).

The control input δ_A is essentially the difference in length between the pair of ropes on the left l_{left} and on the right l_{right} of the system, based on its symmetry plane.

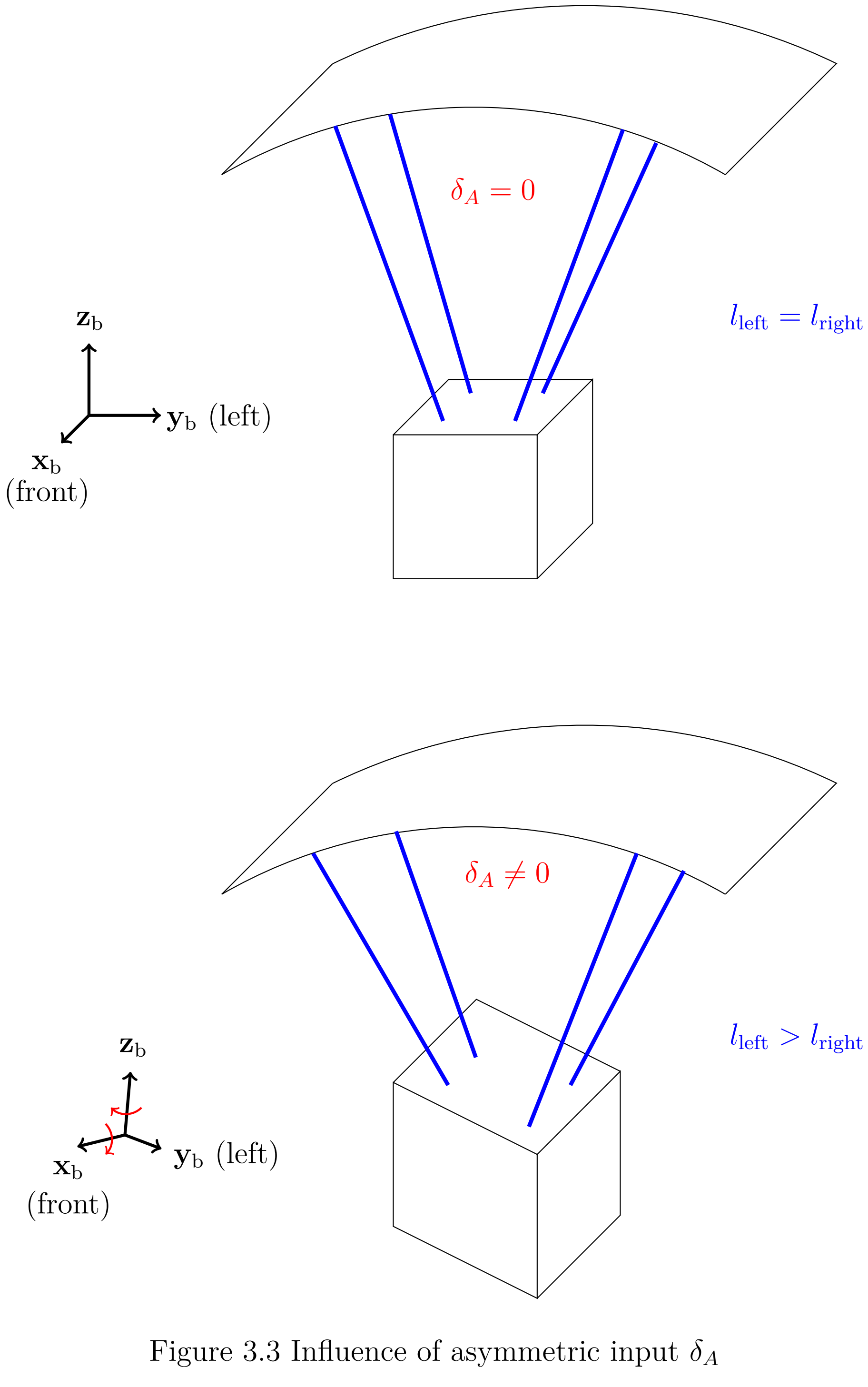

Figure 3.3 Influence of asymmetric input δ_A

As for the control input δ_B, it is the difference in length between the couple forwards l_{front} and the couple backwards l_{back}. In Fig. 3.4, the angle of attack is represented as null to highlight δ_B. If it is not, δ_B is the difference from the equilibrium length for either the front or back ropes.

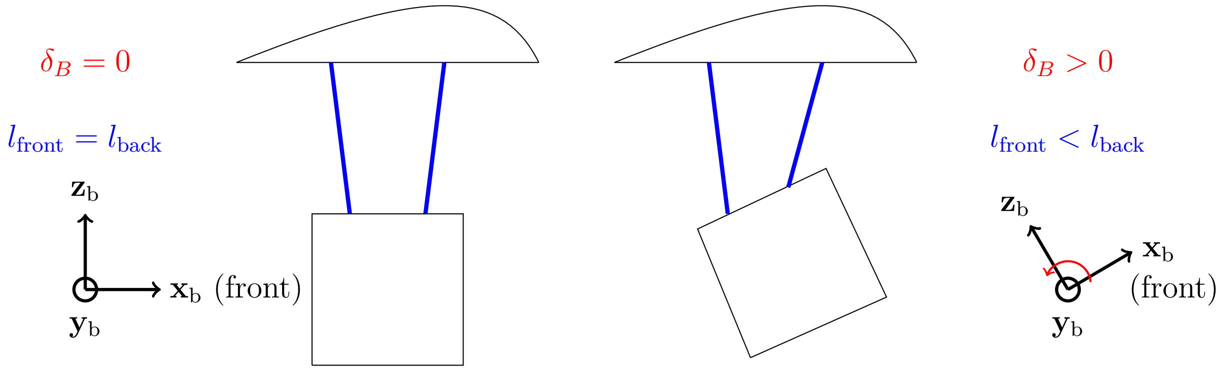

Figure 3.4 Influence of symmetric input δ_B

3.2.2 Expression of forces

From [18], the expression of the aerodynamic forces is given by :

$$\begin{bmatrix} X_A \\ Y_A \\ Z_A \end{bmatrix} = \frac{1}{2}\rho V_T^2 S \mathbf{T}_\alpha \begin{bmatrix} -C_D \\ C_Y \\ -C_L \end{bmatrix}. \tag{3.14}$$

with ρ denoting the air density (depending on altitude), S the parafoil area, and $\mathbf{T}_\alpha$ the rotation matrix of angle of attack α around the $\mathbf{y}_b$ axis whose expression is:

$$\mathbf{T}_\alpha := \begin{bmatrix} \cos\alpha & 0 & -\sin\alpha \\ 0 & 1 & 0 \\ \sin\alpha & 0 & \cos\alpha \end{bmatrix}. \tag{3.15}$$

The expressions of the aerodynamic coefficients C_D (drag), C_Y (lateral force) and C_L (lift) are approximated by:

$$C_D = C_{D_0} + C_{D_0,B}\delta_B + \left(C_{D_{\alpha_2}} + C_{D_{\alpha_2},B}\delta_B\right)\alpha'^2, \tag{3.16a}$$

$$C_Y = C_{Y_\beta}\beta, \tag{3.16b}$$

$$C_L = C_{L_0} + C_{L_0,B}\delta_B + \left(C_{L_\alpha} + C_{L_\alpha,B}\delta_B\right)\alpha' + C_{L_{\alpha_3}}\alpha'^3. \tag{3.16c}$$

3.2.3 Expression of moments

From [18], the expression of the aerodynamic moments is given by:

$$\begin{bmatrix} L_A \\ M_A \\ N_A \end{bmatrix} = \frac{1}{2}\rho V_T^2 S \begin{bmatrix} bC_l \\ cC_m \\ bC_n \end{bmatrix}, \tag{3.17}$$

where b and c are respectively the span and the chord of the parafoil, and the aerodynamic moment coefficients are approximated by:

$$C_l = \frac{b}{2V_T}\left(C_{l_p} p + C_{l_r} r\right), \tag{3.18a}$$

$$C_m = \frac{c}{2V_T} C_{m_q} q, \tag{3.18b}$$

$$C_n = \frac{b}{2V_T}\left(C_{n_p} p + C_{n_r} r\right) + C_{n_\beta}\beta + \left(C_{n_A} + C_{n_{A,I}}\delta_I + C_{n_{A,\alpha}}\alpha'\right)\delta_A. \tag{3.18c}$$

3.2.4 Numerical values

The numerical values of all the coefficients introduced previously are taken from [18] and are gathered in Tab. 3.1 and Tab. 3.2.

Parameter	Value	Units
g_0	9.81	m.s^{-2}
m	2.20	kg
b	1.88	m
c	0.80	m
S	1.50	m^2
I_{xx}	1.68	kg.m^2
I_{yy}	0.80	kg.m^2
I_{zz}	0.32	kg.m^2
I_{xz}	0.09	kg.m^2

Table 3.1 Mass and geometry parameters

Parameter	Value	Parameter	Value
α_B	0.11	C_{m_q}	-2.5
C_{L_0}	0.24	C_{Y_β}	1.00
C_{L_α}	2.14	C_{n_β}	0.10
$C_{L_{\alpha_3}}$	-1.53	C_{l_p}	-0.02
C_{D_0}	0.12	C_{n_p}	0.00
$C_{D_{\alpha_2}}$	0.33	C_{l_r}	0.00
$C_{L_{0,B}}$	0.00	C_{n_r}	0.00
$C_{L_{\alpha,B}}$	0.39	C_{n_A}	-0.04
$C_{D_{0,B}}$	0.43×10^{-1}	$C_{n_{A,I}}$	-0.17×10^{-1}
$C_{D_{\alpha_2,B}}$	2.06	$C_{n_{A,\alpha}}$	-0.01

Table 3.2 Aerodynamic coefficients

3.2.5 About other dynamic models

Although this specific Ward's model has been chosen, slightly different models, with corresponding numerical values, can be found in the literature. In particular, the results will be compared with the work presented in [22], a very detailed article, and in the sixth chapter (Stability and Steady-State Performance) of [21], a reference book in the field of modelling and control of parafoil systems. This book (in particular, chapter 7 about GNC) will also be used to implement the controller and define the advantages and disadvantages of each model, each controller structure, etc. The dynamic model presented in this book is based on the apparent mass effect, which was first introduced for a parafoil in [20].

3.3 Simulink model and simulations

The Simulink model of the complete parafoil system is presented in Fig. 3.5.

3.3.1 Description

First, there are four inputs: the three control inputs mentioned above, $\delta_A, \delta_B, \delta_I$, included between -5 and $5\,\text{cm}$, and the throttle control input $\delta_{th} \in [0, 1]$, in case the parafoil system is powered. There are ten outputs which are the position coordinates of the centre of mass in the NED frame, the angular rates, the Euler angles, the body/NED velocity coordinates in NED and body frames, the heading, the wind velocity,the DCM matrix, the airspeed norm and isolated altitude (scalar). These inputs and outputs are in orange at the ends of the diagram.

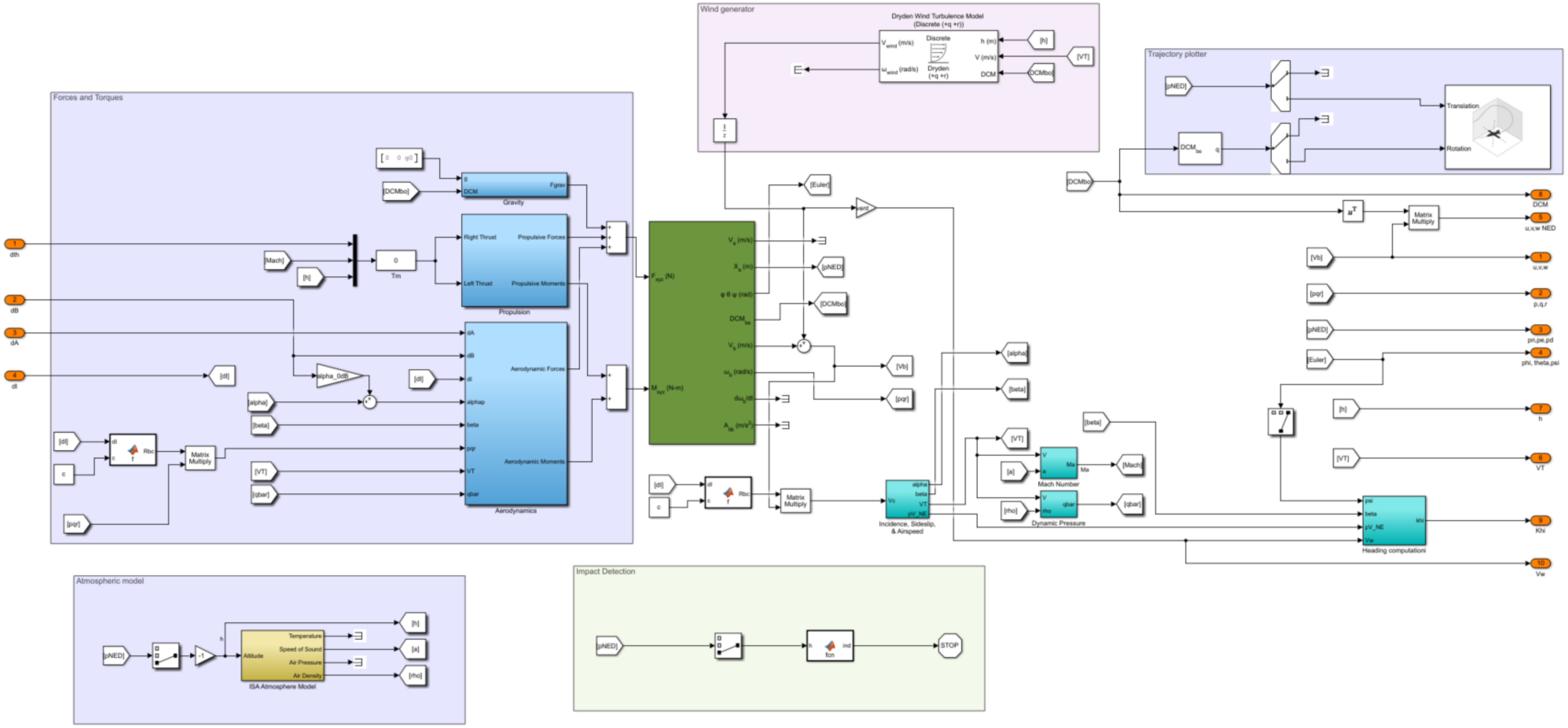

Figure 3.5 Simulink model

In the centre of Fig. 3.5, the dark green block is the numerical integrator. Its inputs are the aerodynamic forces and torques (purple group on the left, up), computed thanks to the golden block (purple group on the left, bottom), which computes in real time physical quantities such as air density and pressure, speed of sound and temperature from the altitude of the system and the ISA atmosphere model [31].

At the top of the Simulink model, a wind generator, built from the Dryden model [67] is implemented; and at its bottom, a stopping criterion which stops the simulation as soon as the system touches the ground. The fifth group of elements, *i.e.*, the purple one at the top right of the figure is a trajectory 3D plotter for better physical appreciation while debugging. The light blue blocks allows one to calculate other physical quantities such as the Mach number, the dynamic pressure, or the characteristic angles such as the heading χ, the angle of attack α and the sideslip angle β.

3.3.2 Steady-state simulations

A first open-loop simulation from the altitude of 20 km, with no control input ($\delta_A(t) = \delta_B(t) = 0$), no propulsion and no wind is conducted. Initial velocity components are chosen arbitrarily. As shown in Fig. 3.6, a complete steady-state cannot be defined, as the increase of the air density along the descent prevent the system from reaching a constant airspeed. Without propulsion, the system cannot reach a steady level flight.

3.3.3 Main simulation results: useful hardware information

Simulations show that the steady-state glide ratio is not affected by the modification of the total mass m of the system, and the surface S of the parafoil. However, it is the case for the steady-state velocity (in the NED frame), whose norm, denoted VT, is shown in Fig. 3.7.

As it can be seen in Fig. 3.7, for all positive m and S, the gradient $\partial_m VT(S,m)$ is strictly positive. An interesting conclusion is that to study the variations of steady-state velocity, it is sufficient to look at the variations in S, for different fixed masses m, varying from 1 to 51 kg. Theses curves are drawn in Fig. 3.8.

These curves have a dependence that can be approximated, for a fixed mass m, by:

$$V_{T,m} = \alpha_m + \frac{\beta_m}{S} + \gamma_m. \tag{3.19}$$

These coefficients can be obtained by the least squares method, for each considered mass. Their values are presented in Fig. 3.9.

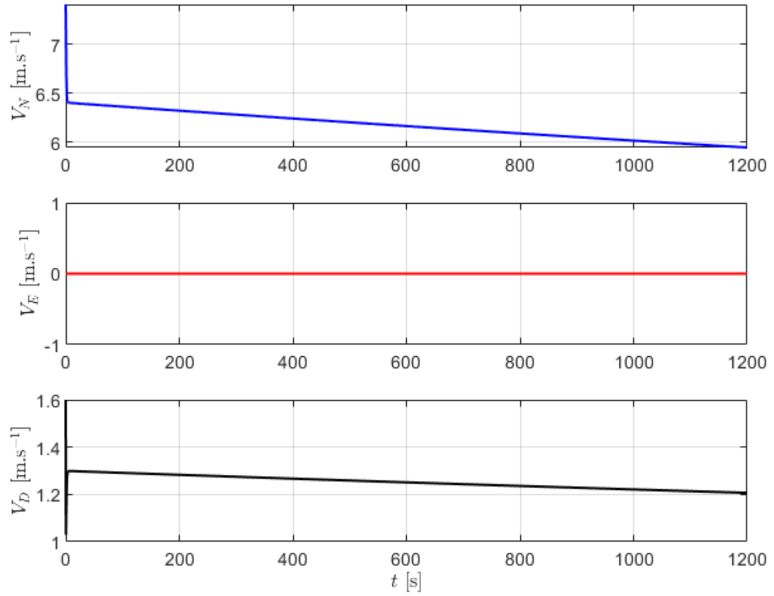

Figure 3.6 No steady-state without propulsion if ρ varies with altitude.

3.4 Equilibrium and linearization

This subsection deals with the equilibrium and linearization of the parafoil's dynamic model in order to determine its dynamic characteristics, mainly the system's modes. The reader is referred to Appendices A and B for further details on the heavy calculations involved in linearizing a dynamic system. The variation of the air density ρ with altitude is neglected. For the sake of calculation and illustrative numerical applications, the air density is set at $\rho_0 = 1.225\,\text{kg.m}^3$.

3.4.1 Input and state vectors

Unlike the Ward model, our model only includes asymmetric and symmetric brake deflections: δ_A and δ_B. The input vector is therefore:

$$\mathbf{u} = [\delta_A, \delta_B]^\top. \tag{3.20}$$

The state vector containing all the information needed to estimate the movement of the gondola (excepted for the position in NED frame) is chosen as:

$$\mathbf{x} = [V_T, \beta, \alpha, p, q, r, \phi, \theta, \psi]^\top. \tag{3.21}$$

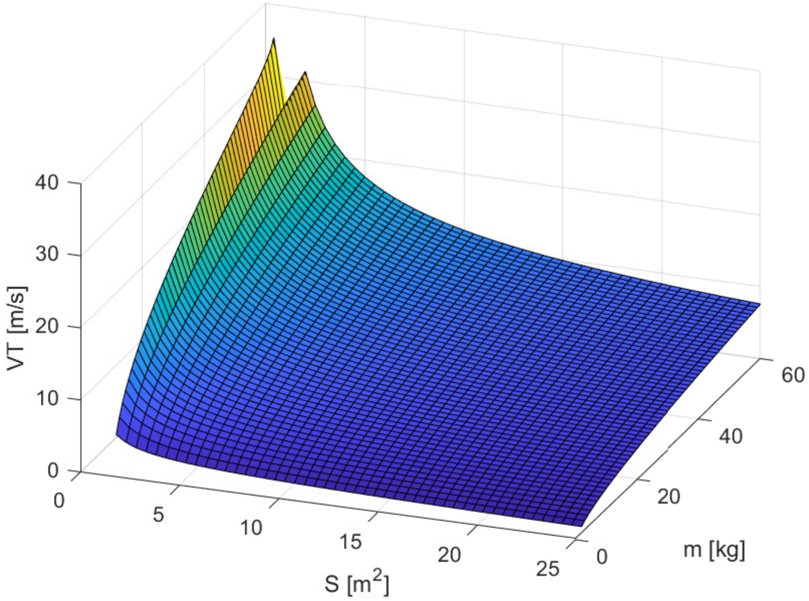

Figure 3.7 Steady-state velocity in terms of gondola mass and parafoil surface.

The state variables u, v and w (3.2) have been replaced by the state variables V_T, β and α. The former ones correspond to a Newton's second law applied in the body frame, while the latter ones correspond to its projection in the wind frame.

3.4.2 Longitudinal and lateral models

From the input vector (3.20) and the state vector (3.21), we can separate the input and state variables into two groups, depending on whether they belong to the longitudinal or the lateral motion, as described in [17] and [31] for fixed-wing aircraft. This approach has proved effective with the parafoil systems in [23–25]. This separation leads to the longitudinal model on the one hand, and the lateral model on the other.

Longitudinal model

The longitudinal reduced state vector is :

$$\mathbf{x}_{\text{lon}} = [V_T, \alpha, q, \theta]^\top, \tag{3.22}$$

and the control input is solely:

$$u_{\text{lon}} = \delta_B. \tag{3.23}$$

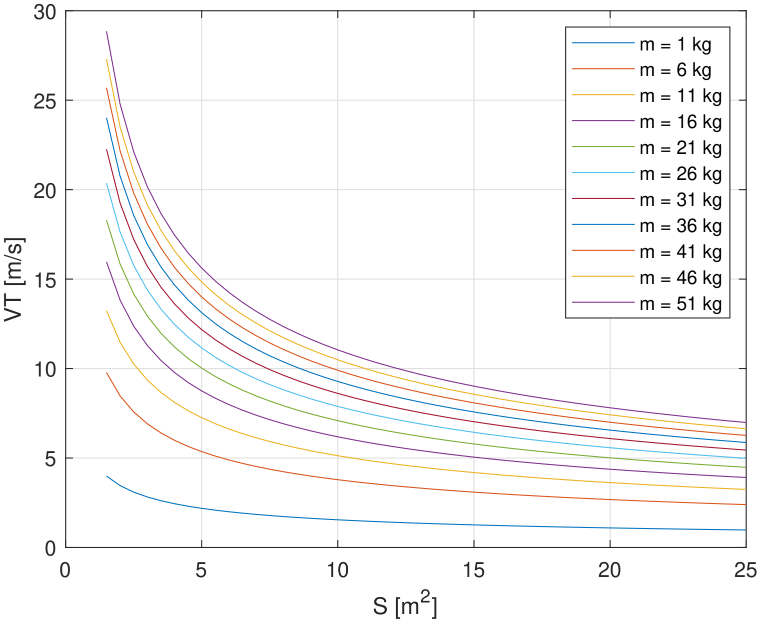

Figure 3.8 Evolution of steady-state velocity with S for different fixed m.

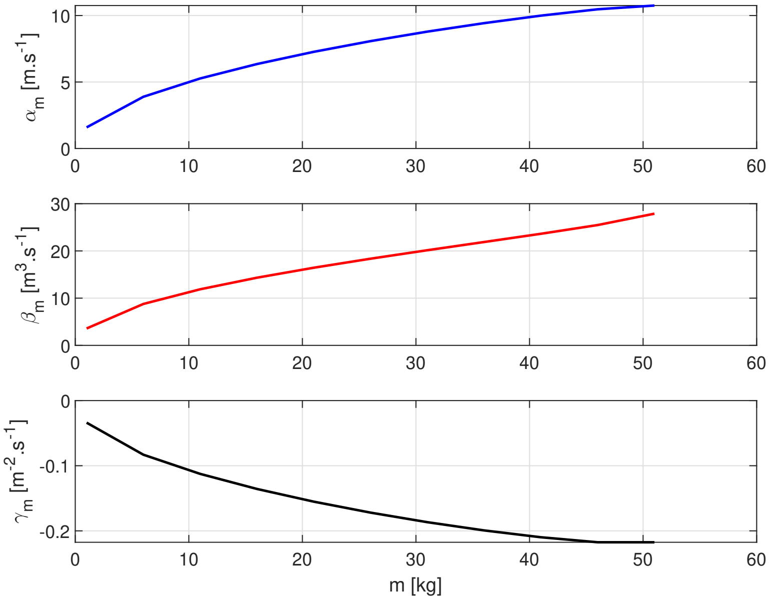

Figure 3.9 Variations of α_m, β_m and γ_m with m

Taken from [31] and assuming that β, ϕ, ψ are null as we are focusing on the longitudinal motion and for the sake of simplicity, the longitudinal dynamic equations in the wind frame are:

$$
\begin{aligned}
m\dot{V}_T &= -\frac{1}{2}\rho V_T^2 S C_D + m g_0 \sin(\alpha - \theta), && \text{(3.24a)}\\
m V_T \dot{\alpha} &= -\frac{1}{2}\rho V_T^2 S C_L + m g_0 \cos(\alpha - \theta) + m V_T q, && \text{(3.24b)}\\
I_{yy}\dot{q} &= \frac{1}{2}\rho V_T^2 S c C_m && \text{(3.24c)}\\
\dot{\theta} &= q && \text{(3.24d)}
\end{aligned}
$$

where the expressions of C_D, C_L and C_m are previously defined in (3.16a), (3.16c) and (3.18b), respectively.

Lateral model

The lateral reduced state vector is :

$$\mathbf{x}_{\mathrm{lat}} = [\beta, p, r, \phi]^\top, \tag{3.25}$$

and the control input is solely:

$$u_{\mathrm{lat}} = \delta_A. \tag{3.26}$$

Note that the state variable ψ has been deliberately ignored, as is usually the case when building the lateral model. We are assuming again that β, ϕ, ψ are null, or at least negligible, since we will only be considering slight deviations from the longitudinal motion later. The lateral dynamic equations in the wind frame are then:

$$
\begin{aligned}
m V_T \dot{\beta} &= \widetilde{Y_A}, && \text{(3.27a)}\\
I_{xx}\dot{p} - I_{xz}\dot{r} &= L_A, && \text{(3.27b)}\\
-I_{xz}\dot{p} + I_{zz}\dot{r} &= N_A, && \text{(3.27c)}\\
\dot{\phi} &= p + \tan\theta_e r && \text{(3.27d)}
\end{aligned}
$$

with

$$\widetilde{Y_A} = Y_A + m V_T \sin\alpha - m V_T \cos\alpha + m g_0 \cos\theta \sin\phi, \tag{3.28}$$

and Y_A, L_A, N_A are the aerodynamic terms defined in (3.14) and (3.17).

3.4.3 Equilibrium

To find an equilibrium point around which to linearize the system, we will consider only forward flight, and therefore longitudinal equilibrium[1]. Linearization of Eq. 3.24 and Eq. 3.27 around this equilibrium point will then be addressed in Sections 3.4.4 and 3.4.5 , in order to obtain linearized longitudinal and lateral models.

As the equations of Eq. 3.24 are non-linear, we can resort to numerical methods to solve the equations at equilibrium, *i.e.*, $\dot{\mathbf{x}}_{\text{lon}} = \mathbf{0}$. For notation purposes, we rewrite Eq. 3.24 as

$$m\dot{V}_T = f_1(\mathbf{x}_{\text{lon}}, u_{\text{lon}}) = -\frac{1}{2}\rho V_T^2 S C_D + m g_0 \sin(\alpha - \theta), \tag{3.29a}$$

$$m V_T \dot{\alpha} = f_2(\mathbf{x}_{\text{lon}}, u_{\text{lon}}) = -\frac{1}{2}\rho V_T^2 S C_L + m g_0 \cos(\alpha - \theta) + m V_T q, \tag{3.29b}$$

$$I_{yy}\dot{q} = f_3(\mathbf{x}_{\text{lon}}, u_{\text{lon}}) = \frac{1}{2}\rho V_T^2 S c C_m \tag{3.29c}$$

$$\dot{\theta} = f_4(\mathbf{x}_{\text{lon}}, u_{\text{lon}}) = q \tag{3.29d}$$

where the functions f_i to be zeroed at equilibrium are stacked into a single vector function $\mathbf{f} = [f_1, f_2, f_3, f_4]^\top$ in the following. To find the equilibrium point for a fixed value of the control input u_{lon}, it is possible to plot the coordinates of the equilibrium point in terms of V_T and α (the quantities of most interest). In Fig. 3.10, the contour lines of $||\mathbf{f}||$ are represented. The red dot corresponds to the minimal value of $||\mathbf{f}||$, *i.e.*, $\text{Min}\,||\mathbf{f}||$.

Concerning the existence of an equilibrium point, we have for $u_{\text{lon}} \notin [-1.2, 2]\,\text{cm}$, $\text{Min}\,||\mathbf{f}|| > 0$, and so there is no physical equilibrium: the system is either stalling or diving. Figure 3.11 shows the range of control inputs values such that a physical equilibrium exists.

---

[1] Linearization of the system around a turning equilibrium point would not provide any additional information on system dynamics, and is therefore ignored here.

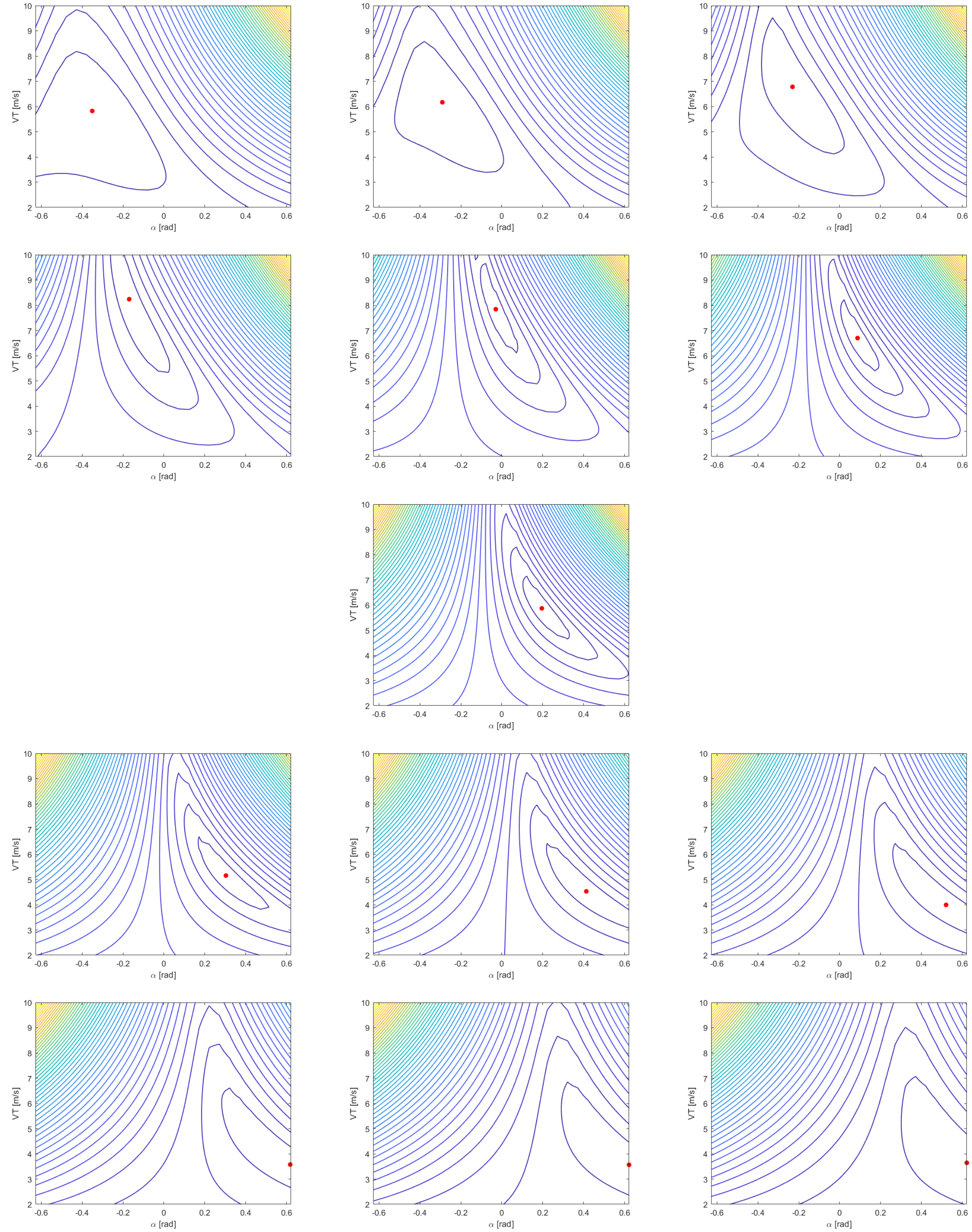

Figure 3.10 Shift of longitudinal equilibrium for u_{lon} from -3 to $3\,\text{cm}$, with an increment of $0.5\,\text{cm}$.

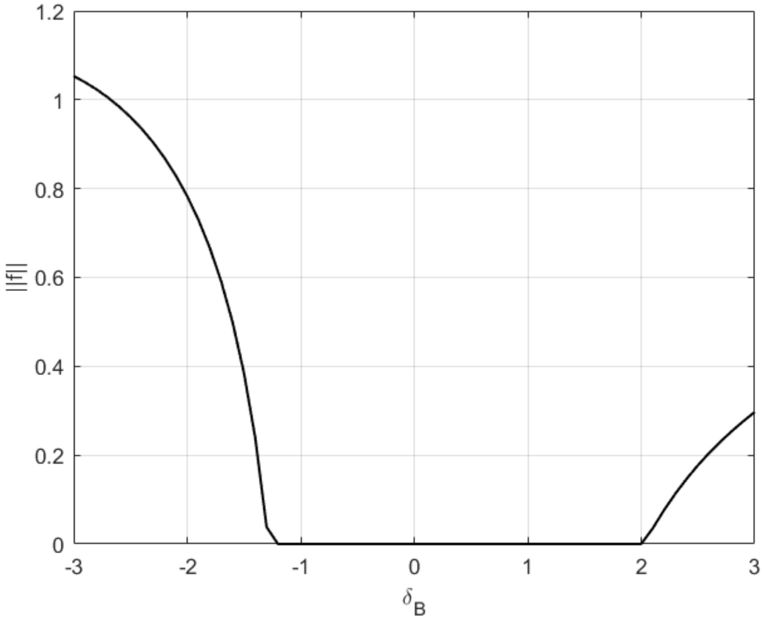

Figure 3.11 Min $||\mathbf{f}||$ depending on u_{lon}

3.4.4 Longitudinal linearization

It is now possible to linearize the system in the vicinity of any possible equilibrium. In the neighbourhood of the null input equilibrium ($u_{\text{lon}} = \delta_{B,e} = 0$), the equilibrium state values with IS units are $\mathbf{x}_{\text{lon},e} = [V_{T,e}, \alpha_e, q_e, \theta_e]^\top = [5.92, 0.2, 0, 0.2]^\top$. Small variations around equilibrium behave according to the linearized law:

$$\Delta\dot{\mathbf{x}}_{\text{lon}} = \underbrace{\begin{bmatrix} \frac{1}{m}\frac{\partial f_1}{\partial V_T}\Big|_e & \frac{1}{m}\frac{\partial f_1}{\partial \alpha}\Big|_e & \frac{1}{m}\frac{\partial f_1}{\partial q}\Big|_e & \frac{1}{m}\frac{\partial f_1}{\partial \theta}\Big|_e \\ \frac{1}{mV_{T,e}}\frac{\partial f_2}{\partial V_T}\Big|_e & \frac{1}{mV_{T,e}}\frac{\partial f_2}{\partial \alpha}\Big|_e & \frac{1}{mV_{T,e}}\frac{\partial f_2}{\partial q}\Big|_e & \frac{1}{mV_{T,e}}\frac{\partial f_2}{\partial \theta}\Big|_e \\ \frac{1}{I_{yy}}\frac{\partial f_3}{\partial V_T}\Big|_e & \frac{1}{I_{yy}}\frac{\partial f_3}{\partial \alpha}\Big|_e & \frac{1}{I_{yy}}\frac{\partial f_3}{\partial q}\Big|_e & \frac{1}{I_{yy}}\frac{\partial f_3}{\partial \theta}\Big|_e \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\mathbf{A}_{\text{lon}}} \Delta\mathbf{x}_{\text{lon}} + \underbrace{\begin{bmatrix} \frac{1}{m}\frac{\partial f_1}{\partial \delta_B}\Big|_e \\ \frac{1}{mV_{T,e}}\frac{\partial f_2}{\partial \delta_B}\Big|_e \\ \frac{1}{I_{yy}}\frac{\partial f_3}{\partial \delta_B}\Big|_e \\ 0 \end{bmatrix}}_{\mathbf{B}_{\text{lon}}} \Delta u_{\text{lon}}, \quad (3.30)$$

whith $\Delta\mathbf{x}_{\text{lon}} = \mathbf{x}_{\text{lon}} - \mathbf{x}_{\text{lon,e}}$ and $\Delta u_{\text{lon}} = u_{\text{lon}} - u_{\text{lon,e}}$. In particular, with no control input ($u_{\text{lon},e} = 0$), the numerical values of $\mathbf{A}_{\text{lon}}$ are:

$$\mathbf{A}_{\text{lon}} = \begin{bmatrix} 0.4 & 8.6078 & 0 & -9.8066 \\ -0.5643 & -5.4944 & 1 & 0 \\ -0.2058 & -6.32 & -5.3882 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \quad (3.31)$$

These values are the ones that will be used for modal analysis. See Appendix A for more details.

3.4.5 Lateral linearization

Considering the equilibrium $\mathbf{x}_{\text{lon},e}$, $\mathbf{x}_{\text{lat},e} = \mathbf{0}$ and $u_{\text{lat}} = 0$, the linearization in the vicinity of 0 (for lateral state quantities, and for $V_{T,e} = 5.92\,\text{m.s}^{-1}$ and $\alpha_e = \theta_e = 0.2\,\text{rad}$), is given by:

$$\underbrace{\begin{bmatrix} mV_{T,e} & 0 & 0 & 0 \\ 0 & I_{xx} & -I_{xz} & 0 \\ 0 & -I_{xz} & I_{zz} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\mathbf{E}} \Delta\dot{\mathbf{x}}_{\text{lat}} = \underbrace{\begin{bmatrix} \left.\dfrac{\partial \widetilde{Y_A}}{\partial \beta}\right|_e & \left.\dfrac{\partial \widetilde{Y_A}}{\partial p}\right|_e & \left.\dfrac{\partial \widetilde{Y_A}}{\partial r}\right|_e & \left.\dfrac{\partial \widetilde{Y_A}}{\partial \phi}\right|_e \\ \left.\dfrac{\partial L_A}{\partial \beta}\right|_e & \left.\dfrac{\partial L_A}{\partial p}\right|_e & \left.\dfrac{\partial L_A}{\partial r}\right|_e & \left.\dfrac{\partial L_A}{\partial \phi}\right|_e \\ \left.\dfrac{\partial N_A}{\partial \beta}\right|_e & \left.\dfrac{\partial N_A}{\partial p}\right|_e & \left.\dfrac{\partial N_A}{\partial r}\right|_e & \left.\dfrac{\partial N_A}{\partial \phi}\right|_e \\ 0 & 1 & \tan\theta_e & 0 \end{bmatrix}}_{\mathbf{A}_0} \Delta\mathbf{x}_{\text{lat}} + \underbrace{\begin{bmatrix} \left.\dfrac{\partial \widetilde{Y_A}}{\partial \delta_A}\right|_e \\ \left.\dfrac{\partial L_A}{\partial \delta_A}\right|_e \\ \left.\dfrac{\partial N_A}{\partial \delta_A}\right|_e \\ 0 \end{bmatrix}}_{\mathbf{B}_0} \Delta u_{\text{lat}} \quad (3.32)$$

By writing $\mathbf{A}_{\text{lat}} = \mathbf{E}^{-1}\mathbf{A}_0$, and $\mathbf{B}_{\text{lat}} = \mathbf{E}^{-1}\mathbf{B}_0$, the following numerical values are got for the zero-centered equilibrium:

$$\mathbf{A}_{\text{lat}} = \begin{bmatrix} -3.0755 & 0.1942 & -0.981 & 1.6403 \\ 0.106 & -0.1151 & 0.2994 & 0 \\ 1.9795 & -0.0324 & 0.0559 & 0 \\ 0 & 1 & 0.1980 & 0 \end{bmatrix}. \quad (3.33)$$

See Appendix B for more development details.

3.5 Modal analysis

3.5.1 Longitudinal modes

The eigenvalues of the $\mathbf{A}_{\text{lon}}$ matrix for the zero-entry equilibrium can be compared with the longitudinal modes presented in the literature; for instance [21] presents the poles of a Snowflake-PADS, a 7-DOF parafoil system with geometric and inertial properties similar to the ones used in this study (2.2 kg total mass and 1.5 m^2 parafoil surface) ; and the poles of an Alex-PADS, a 6-DOF system which is bigger and heavier (111.3 kg total mass and 19.7 m^2 parafoil surface). Table 3.3 contains the complex coordinates of these poles, and the complex plane is presented in Fig. 3.12. In each case, the considered equilibrium is the one with no command, which means $u_{\text{long,e}} = 0$. The studied poles are represented in red, the ones of Snowflake in black and the ones of Alex in blue.

Mode	Model	Affix	Pulsation ω [rad/s]	Damping ζ
Short Period	Studied system	$-5.18 \pm 2.76i$	5.87	0.88
	Snowflake-PADS	$-2.39 \pm 6.72i$	7.14	0.34
	Alex-PADS	$-3.33 \pm 4.63i$	5.71	0.58
Phugoid	Studied system	$-0.06 \pm 0.08i$	0.83	0.08
	Snowflake-PADS	$-0.59 \pm 1.12i$	1.26	0.47
	Alex-PADS	$-0.30 \pm 0.69i$	0.75	0.40

Table 3.3 Longitudinal modes

The Phugoid poles of the studied system are really close to those of the Alex-PADS, which is a 6-DOF system as well. Yet, the short period mode is faster on Ward's model than on the two others found in the literature. Moreover, an interesting result to mention is that 6-DOF models (Ward's present case and Alex-PADS) have a Phugoid mode which is more damped than the 7-DOF model of Snowflake-PADS.

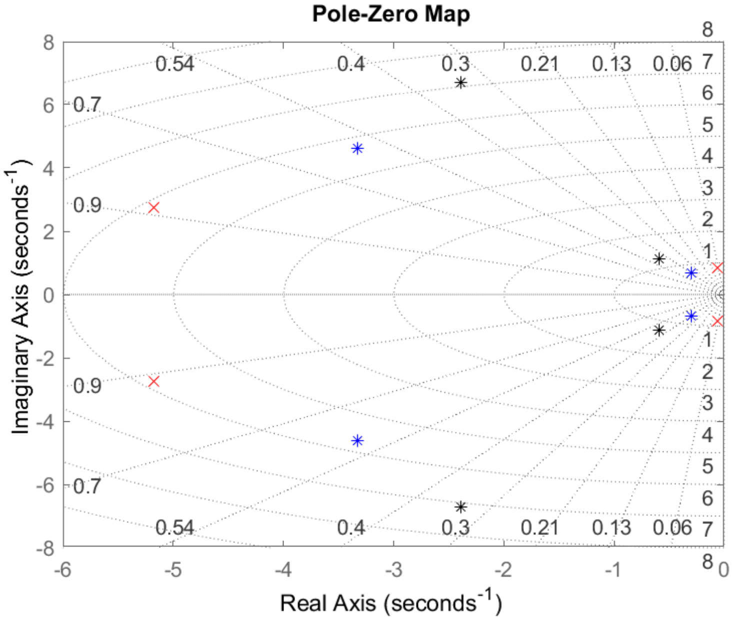

Figure 3.12 Complex plane of nominal longitudinal modes.

Since the symmetrical input δ_B will only be non-null at the very end of the flight (see subsection 4.4.2), the displacement of the poles when another equilibrium is considered is not of practical interest during the entire flight. Nevertheless, when it comes to landing, it is interesting to mention that increasing δ_B (*i.e.*, flying with a rearing up nose) makes the phugoid mode faster and the short period slower; and leads to a less damped short period but a more damped phugoid (as it can be seen in Fig. 3.12. This is a good sign, as it means that the system will be less likely to oscillate when flying in a straight line just before landing. It is worth noting that the numerical values stay relatively close to the ones presented in the literature.

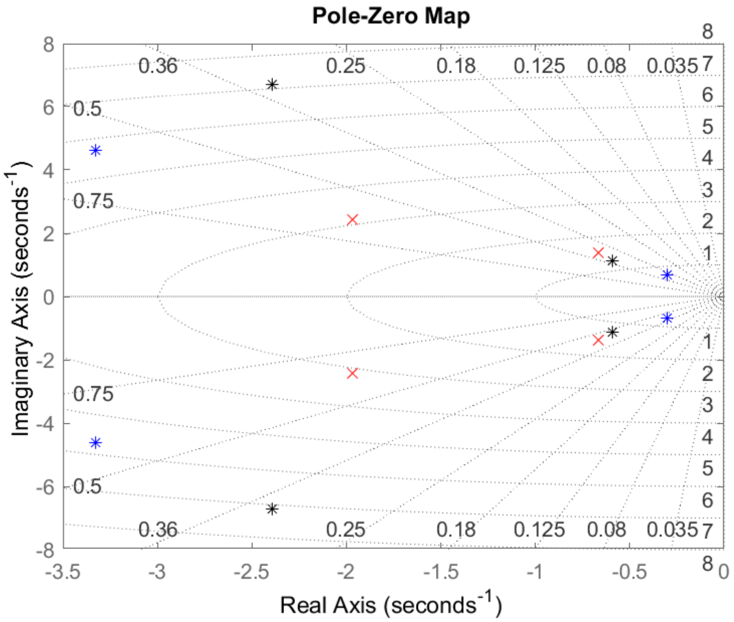

Figure 3.13 Complex plane of longitudinal modes around $\delta_B = 2\,\mathrm{cm}$

3.5.2 Lateral modes

Similarly as what was done before, the lateral modes are represented in Tab. 3.4, and shown in Fig. 3.14.

Mode	Model	Affix	Pulsation ω [rad/s]	Damping ζ
Roll subsidence	Studied system	2.07	4.83	-
	Snowflake-PADS	-4.32	4.32	-
Dutch Roll	Studied system	$-0.83 \pm 0.43i$	0.93	0.89
	Snowflake-PADS	$-0.55 \pm 5.22i$	5.24	0.10
	Alex-PADS	$-0.67 \pm 3.40i$	3.49	0.19
Spiral	Studied system	0.58	0.58	-
	Snowflake-PADS	-0.37	0.37	-
Coupled Spiral-Roll	Alex-PADS	$-0.44 \pm 0.83i$	0.94	0.47

Table 3.4 Lateral modes

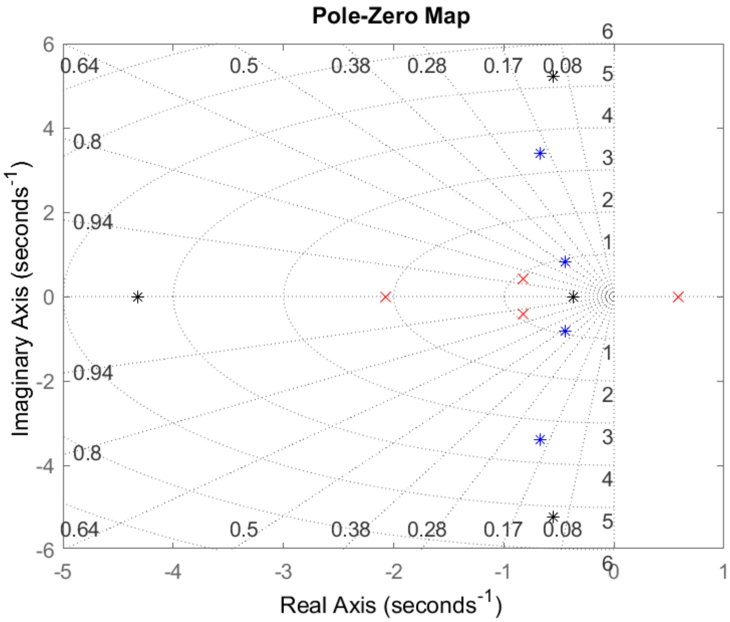

Figure 3.14 Complex plane of nominal lateral modes.

The main results can be listed as follows:

- Spiral and roll modes of the studied model do not merge, while they do for the Alex-PADS model.
- There exists a spiral mode which is unstable, while it is real but stable in the Snowflake-PADS model. It is quite fast (time constant magnitude is 0.1 s while usually more around 0.01 s), but it is reasonable compared with the literature since no saturation in the coefficients has been applied. In [21], a slight modification in the coefficients leads to a stable roll subsidence.
- The other numerical values found in this study are consistent with the literature.

CHAPTER 4 PATH PLANNING

This chapter focuses on the development of a hybrid algorithm, which aims to enhance the precision in optimal trajectory planning for parafoil systems. The proposed algorithm combines nonlinear shooting and a novel dynamic programming technique [43, 49], allowing a more nuanced approach to trajectory planning. The optimization problem will initially be modelled in 2D, with the execution following a 3D model in the simulator - an approach confirmed to be effective in prior studies [42, 50, 54].

Breaking with the mould of the existing literature, the work presented in this master's thesis strategically excludes wind modelling in the optimal formulation. Instead, we implement a hybrid method that uses a controller to first follow the nonlinear shooting trajectory, with the dynamic programming algorithm recalculating new optimal trajectories on-the-fly when deviations occur due to environmental influences. Additionally, the dynamic programming algorithm introduced in this chapter presents a new formulation that offers vastly improved physical coherence compared to traditional Cartesian mappings. It eliminates the need for snap-to-grid techniques, resulting in a piecewise continuous trajectory that represents flights and their costs accurately. To evaluate its effectiveness, the developed algorithm will be compared to a reference algorithm based on nonlinear shooting, which is widely recognized in the literature [42].

4.1 Basic assumptions and equations of motion

In this section, the path optimization will be formulated for a 2D problem. For the first phase of the flight, the system dynamics in the North-East (NE) plane can be described by the following state-space model :

$$\begin{cases} \dot{x}_1 = V\cos(x_3+\beta) + w_N, \\ \dot{x}_2 = V\sin(x_3+\beta) + w_E, \\ \dot{x}_3 = x_4, \\ \dot{x}_4 = K_\psi V^2 u. \end{cases} \tag{4.1}$$

where x_1, x_2, x_3 and x_4 denote respectively the North and East positions in the NED frame, the heading, and the heading turning rate. They are represented in Fig. 4.1 for the first three ones. The trajectory is represented in blue, and the velocity vector in red. The velocity components are given in the NED frame, and express the motion of the gondola relative to the Earth. The velocity norm V is assumed to be constant and set to $V = 2.5\,\mathrm{m.s}^{-1}$. The only control input is the asymmetric brake deflection $u = \delta_A$, which acts on yaw turn rate, as in [28]. The constant $K_\psi = \frac{\rho S b_c C_{n_A}}{2 I_{zz}} \approx -0.0173\,\mathrm{rad.m}^{-3}$, obtained from numerical values given in [28], is such that $K_\psi V^2$ is the constant variation rate of aerodynamic moment with respect to asymmetric brake deflection. Its value is highly dependant on the geometric and inertial characteristics of the system presented in Section 3.2. Finally, the terms w_N, w_E and β represents the wind components in the NE frame, and the sideslip angle.

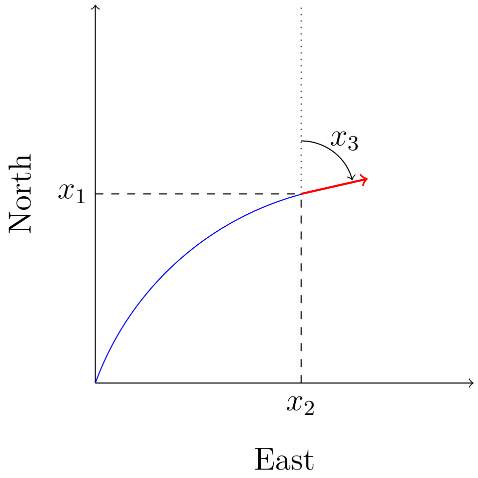

Figure 4.1 Representation of the state variables

For the sake of simplicity, a zero-wind case will be considered to solve the problem, meaning that there is no relative movement between the atmosphere and the Earth. Moreover, the sideslip angle is considered null. The state-space model (4.1) can then be simplified as:

$$\begin{cases} \dot{x}_1 = V\cos x_3, \\ \dot{x}_2 = V\sin x_3, \\ \dot{x}_3 = x_4, \\ \dot{x}_4 = K_\psi V^2 u. \end{cases} \tag{4.2}$$

The space-space model can then be formulated in a compact vector form $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, u)$, with the state vector $\mathbf{x} = [x_1, x_2, x_3, x_4]^\top$ and where the multivariable nonlinear function $\mathbf{f}$ is:

$$\mathbf{f}(\mathbf{x}, u) = \begin{bmatrix} V\cos x_3 \\ V\sin x_3 \\ x_4 \\ K_\psi V^2 u \end{bmatrix}. \tag{4.3}$$

4.2 Nonlinear Shooting in Phase 1

4.2.1 Formulation

Minimization problem

As the optimal controller is only formulated in phase 1 (gliding), the final time of this phase will be denoted t_f. The cost J to minimize is the control effort over time:

$$J = \int_0^{t_f} L[\mathbf{x}(t), u(t), t]\mathrm{d}t, \tag{4.4}$$

with $L[\mathbf{x}(t), u(t), t] = \frac{1}{2}u^2(t)$. Two final conditions must be satisfied at the end of phase 1:

1. the system must be on the targeted circle (centered in (a, b), radius R), *i.e.*, verify the circle equation;
2. and arrive tangentially, *i.e.*, the final speed vector has to be orthogonal to the vector linking system position to the center of the circle.

Both conditions are represented by the constraint:

$$\boldsymbol{\psi}[\mathbf{x}(t_f), t_f] = \begin{bmatrix} (x_1(t_f) - a)^2 + (x_2(t_f) - b)^2 - R^2 \\ \cos(x_3(t_f))(x_1(t_f) - a) + \sin(x_3(t_f))(x_2(t_f) - b) \end{bmatrix} = \mathbf{0}, \tag{4.5}$$

and illustrated in Fig. 4.2 where trajectories are represented in blue, and velocity vector at final time in red.

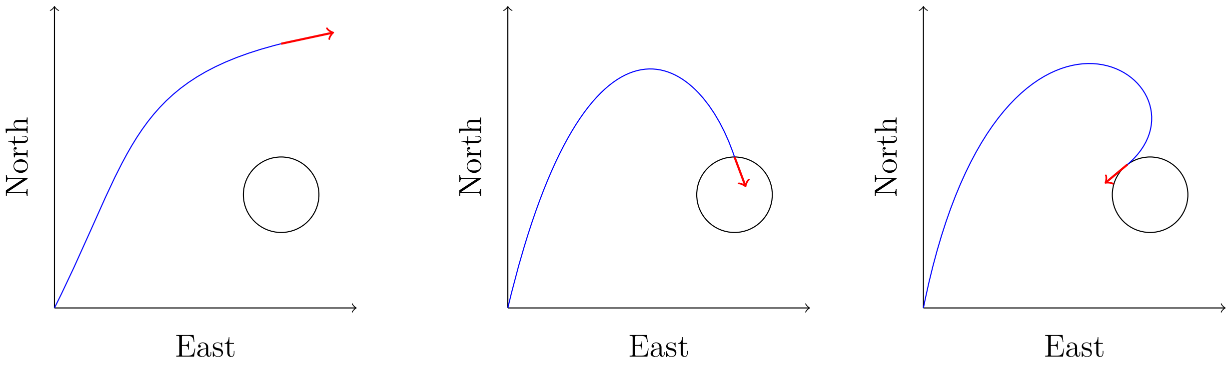

Figure 4.2 Scenarios where $\boldsymbol{\psi}[\mathbf{x}(t_f), t_f] \neq \mathbf{0}$ (left, middle); and where $\boldsymbol{\psi}[\mathbf{x}(t_f), t_f] = \mathbf{0}$ (right)

The Hamiltonian of the system is then given by

$$H(\mathbf{x}(t), u(t), \boldsymbol{\lambda}(t), t) = L(\mathbf{x}(t), u(t), t) + \boldsymbol{\lambda}^\top(t)\mathbf{f}(\mathbf{x}(t), u(t), t), \tag{4.6}$$

with $\mathbf{f}(\mathbf{x}(t), u(t), t)$ defined in (4.3) and $\boldsymbol{\lambda}(t) = [\lambda_1, \lambda_2, \lambda_3, \lambda_4]^\top$ the costate vector. Consequently, the developed Hamiltonian is:

$$H(\mathbf{x}(t), u(t), \boldsymbol{\lambda}(t), t) = \frac{1}{2}u^2 + \lambda_1 V\cos(x_3) + \lambda_2 V\sin(x_3) + \lambda_3 x_4 \lambda_3 + \lambda_4 K_\psi V^2 u. \tag{4.7}$$

Pontryagin equations

To compute an optimal solution, *i.e.*, a control law $u^*(t)$ for $0 \leq t \leq t_f$ such that J is minimal, there are several necessary conditions that must be verified.

The optimality of the Hamiltonian must be verified, which means that the optimal state trajectory $\mathbf{x}^*$ and the optimal control input u^* must verify:

$$\forall t \in [0, t_f], \quad H(\mathbf{x}^*(t), u^*(t), \boldsymbol{\lambda}^*(t), t) \leq H(\mathbf{x}(t), u(t), \boldsymbol{\lambda}(t), t). \tag{4.8}$$

The fact that the Hamiltonian has to be minimized over the control input implies the following conditions on its first and second derivatives:

$$\frac{\partial H}{\partial u} = 0, \qquad \text{and} \qquad \frac{\partial^2 H}{\partial u^2} > 0. \tag{4.9}$$

These conditions lead to

$$u = -K_\psi V^2 \lambda_4, \tag{4.10}$$

and $\partial^2_{uu} H > 0$ since H is strictly convex in u. The costate equation, which is the dynamics of the $\boldsymbol{\lambda}$ function, shall verify:

$$\dot{\boldsymbol{\lambda}} = -\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)^\top \boldsymbol{\lambda} - \left(\frac{\partial L}{\partial \mathbf{x}}\right)^\top. \tag{4.11}$$

This leads to the following differential system :

$$\begin{cases} \dot{\lambda}_1 = 0, \\ \dot{\lambda}_2 = 0, \\ \dot{\lambda}_3 = V\sin(x_3)\lambda_1 - V\cos(x_3)\lambda_2, \\ \dot{\lambda}_4 = -\lambda_3. \end{cases} \tag{4.12}$$

The terminal conditions on the costate equation take the following form:

$$\boldsymbol{\lambda}(t_f) = \left(\boldsymbol{\nu}^\top \frac{\partial \boldsymbol{\psi}}{\partial \mathbf{x}}\right)^\top \Bigg|_{t=t_f}, \tag{4.13}$$

with $\boldsymbol{\nu} = [\nu_1, \nu_2]^\top$ a vector of optimization parameters, constant over time. That leads to a vectorial equation on Lagrange multipliers to be verified at final time:

$$\boldsymbol{\lambda}(t_f) = \begin{bmatrix} 2\nu_1(x_1(t_f) - a) + \nu_2 \cos(x_3(t_f)) \\ 2\nu_1(x_2(t_f) - b) + \nu_2 \sin(x_3(t_f)) \\ \nu_2(-\sin(x_3(t_f))(x_1(t_f) - a) + \cos(x_3(t_f))(x_2(t_f) - b))) \\ 0 \end{bmatrix}. \tag{4.14}$$

Finally, as $\mathbf{x}(t_f)$ is not imposed, the last necessary condition is

$$\left[\boldsymbol{\nu}^\top \frac{\partial \boldsymbol{\psi}}{\partial t} + H\right]\Bigg|_{t=t_f} = 0. \tag{4.15}$$

Uniqueness of solutions

The previously detailed conditions are the necessary ones on which the algorithm will be based, but under some convexity assumptions, they can become sufficient as well (see [68] for more details) and guarantee uniqueness of the solution.

In the present case, there might not be a unique solution. The system may for instance tangentially touch the final circle as fast as possible, which will require a heavy control effort over a short time (scenario 1); or fly a longer time but with a slighter control effort (scenario 2). Such iso-optimal solutions are presented in Fig. 4.3. With the same initial conditions and final constraints in both cases, scenarios 1 and 2 are respectively represented by blue and green trajectories. This problem of local minima is highly dependant on the initial guess of final time t_f.

To solve this problem by giving priority either to the shortest path or to minimising the maximum control effort, a weighting coefficient $\alpha \in \mathbb{R}_+^*$ can be introduced and the corresponding cost formulated in Eq. 4.4 would become:

$$J_\alpha = \int_0^{t_f} \left[\alpha + \frac{1}{2}u^2(t)\right] \mathrm{d}t = \alpha t_f + J, \tag{4.16}$$

with $\alpha \ll 1$ giving priority to reducing maximal control effort and $\alpha \gg 1$ giving priority to minimizing final time t_f. As a longer flight will result in a greater loss of altitude, α can also

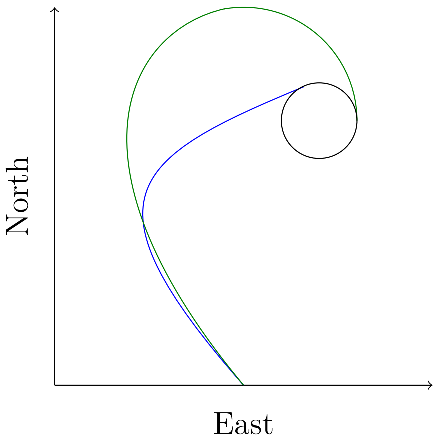

Figure 4.3 Iso-optimal trajectories

be seen as the importance of potential energy (which is proportional to t_f, , considering that the descent rate is almost constant): a longer flight ($\alpha \ll 1$) means minimizing potential energy, and a shorter flight ($\alpha \gg 1$) does not.

Another trivial case of non-unicity of solutions would be a perfect symmetry of the problem: an initial velocity vector along North axis and a circle with its center on this axis. In that case, initial guesses would be even more paramount.

This multiplicity of solutions occurs with the dynamic programming algorithm as well.

4.2.2 Algorithm

The algorithm uses the derivative-free Matlab function `fminsearch` to find initial conditions $\boldsymbol{\lambda}^*(0)$, optimization parameters $\boldsymbol{\nu}^*$ and final time t_f^* such that numerical integration with Dormand-Prince method leads to verify the final conditions described in Eqs 4.5, 4.14 and 4.15. As discussed above, the computed solution depends on the initial guess for these values. The presented algorithm starts running with the following values:

$$\boldsymbol{\lambda}^0(0) = 0, \tag{4.17a}$$

$$\boldsymbol{\nu}^0 = 0, \tag{4.17b}$$

$$t_f^0 = \frac{\|\mathbf{C}\|_1 + \|\mathbf{C}\|_2}{2V}, \tag{4.17c}$$

where $\mathbf{C} = [a\ b]^\top$ is the position vector of the circle centre. Physically, t_f^0 is the average value between the time t_1 it takes to fly in a straight line along the North axis, then in a straight line along the East axis to the point $\mathbf{C}$, *i.e.*, $t_1 = \|\mathbf{C}\|_1/V = (|a| + |b|)/V$, and the time t_2 it takes to go straight to the point at speed V, *i.e.*, $t_2 = \|\mathbf{C}\|_2/V = \sqrt{a^2+b^2}/V$,

4.2.3 Results

In each of the use cases described below, the radius of the target circle is set to $R = 10\,\mathrm{m}$. Five reference trajectories will be used as benchmarks to test the performance of the new dynamic programming algorithm:

1. A circle centered at $a = b = 200\,\mathrm{m}$ and initial heading set to $\pi/3\,\mathrm{rad}$, which represents a roughly well-oriented initial velocity vector.
2. A circle centered at $a = b = 200\,\mathrm{m}$ and initial heading set to $-\pi/8\,\mathrm{rad}$: the final conditions $\boldsymbol{\psi}[\mathbf{x}(t_f, t_f)]$ is the same as in case 1, but the initial speed vector points in the wrong direction (away from the target).
3. A circle centered at $a = 200\,\mathrm{m}$, $b = 50\,\mathrm{m}$ and initial heading set to 0. This case illustrates the behaviour of the algorithm when the target is roughly aligned with the initial velocity vector.
4. A circle centered at $a = 100\,\mathrm{m}$, $b = -300\,\mathrm{m}$ and initial heading set to 0. This case illustrates the behaviour of the algorithm when the target is away from initial point and requires an important turn.
5. A circle centered at $a = b = 200\,\mathrm{m}$ and initial heading set to 0. This case illustrates the behaviour of the algorithm when there is a need to almost completely turn around to get to the target.

The two first use cases show the impact of modifying initial conditions while targeting the same circle and are presented in Fig. 4.4, while the three others, presented in Fig. 4.5, show the impact of modifying final constraints without changing initial conditions.

The initial conditions and final constraints for the five use cases were chosen to be either identical or very different from one case to another. If they had been close, another possibility would have been to design a set of slightly sub-optimal trajectories from a nominal trajectory, rather than recalculating the entire optimal trajectory.

This approach is based on the principle of the second variation of the Hamiltonian, as developed by Hull (2003) [69]. Hull provides an analytic expression for the control input variation that should be applied to the system in case of slight modifications in the target coordinates or initial conditions. These close trajectories would allow for adjustments in the event of deviation from the intended path and could be calculated onboard as well. See Appendix D for further details on this method.

4.3 Dynamic Programming in Phase 1

4.3.1 Formulation

Introduction

The originality of this algorithm lies in its novel state space: the graph is built with nodes representing different possible values for finite differences of heading and enabling a direct control on the second derivative of heading despite the discretized nature of equations.

Discretization

Equation 4.2 has to be discretised in order to use the discrete dynamic programming paradigm. Assuming that variations of control input δu are small in module compared to u, choosing a mesh in δu is a judicious choice, as for the same number of nodes, a better precision will be obtained. Now, with a step time Δt, and denoting $\forall k \in \mathbb{N}$, $t_k = k\Delta t$, the control input is constant over every time interval defined by k:

$$\forall t \in [t_k, t_{k+1}],\ u(t) = \frac{\alpha_k}{K_\psi V^2}, \tag{4.18}$$

with $\alpha_k \in \mathbb{R}$ a constant, depending only on k. The decision making process is based on the choice of the value $\delta_k = \alpha_k - \alpha_{k-1}$, with $\alpha_{-1} = 0$. Now the state vector can be reconstructed step by step with continuous values, for a fixed k and $t \in [t_k, t_{k+1}]$:

$$x_4(t) = \frac{\alpha_k}{K_\psi V^2}(t - t_k) + x_4(t_k), \tag{4.19}$$

$$x_3(t) = \frac{\alpha_k}{2K_\psi V^2}(t^2 - t_k^2) + (-\alpha_k t_k + x_4(t_k))(t - t_k) + x_3(t_k), \tag{4.20}$$

$$x_2(t) = \int_{t_k}^{t} V \sin\left(\frac{\alpha_k}{2K_\psi V^2}(\tau^2 - t_k^2) + (-\alpha_k t_k + x_4(t_k))(\tau - t_k) + x_3(t_k)\right)\mathrm{d}\tau + x_2(t_k), \tag{4.21}$$

$$x_1(t) = \int_{t_k}^{t} V \cos\left(\frac{\alpha_k}{2K_\psi V^2}(\tau^2 - t_k^2) + (-\alpha_k t_k + x_4(t_k))(\tau - t_k) + x_3(t_k)\right)\mathrm{d}\tau + x_1(t_k). \tag{4.22}$$

This formulation gives a piecewise definition of all state variables. It even allows to find the full continuous trajectory knowing the nodes which have been chosen by the algorithm for

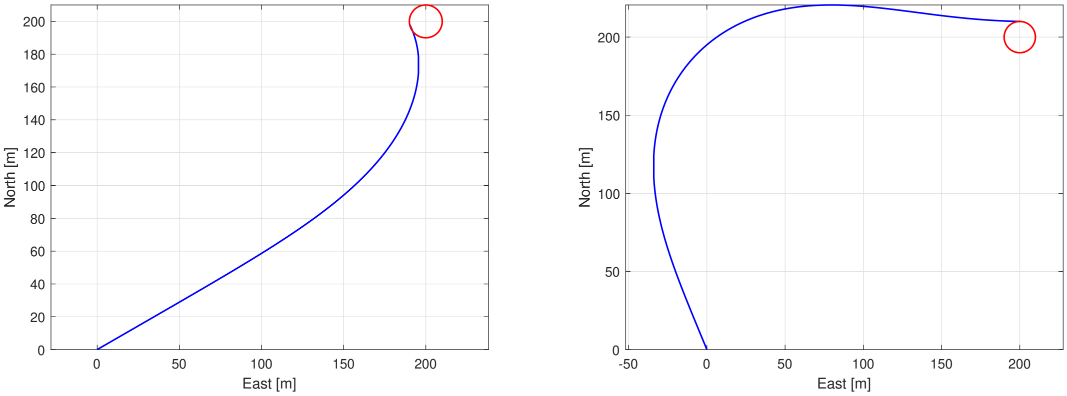

Figure 4.4 Nonlinear shooting trajectories with different initial conditions (cases 1 and 2)

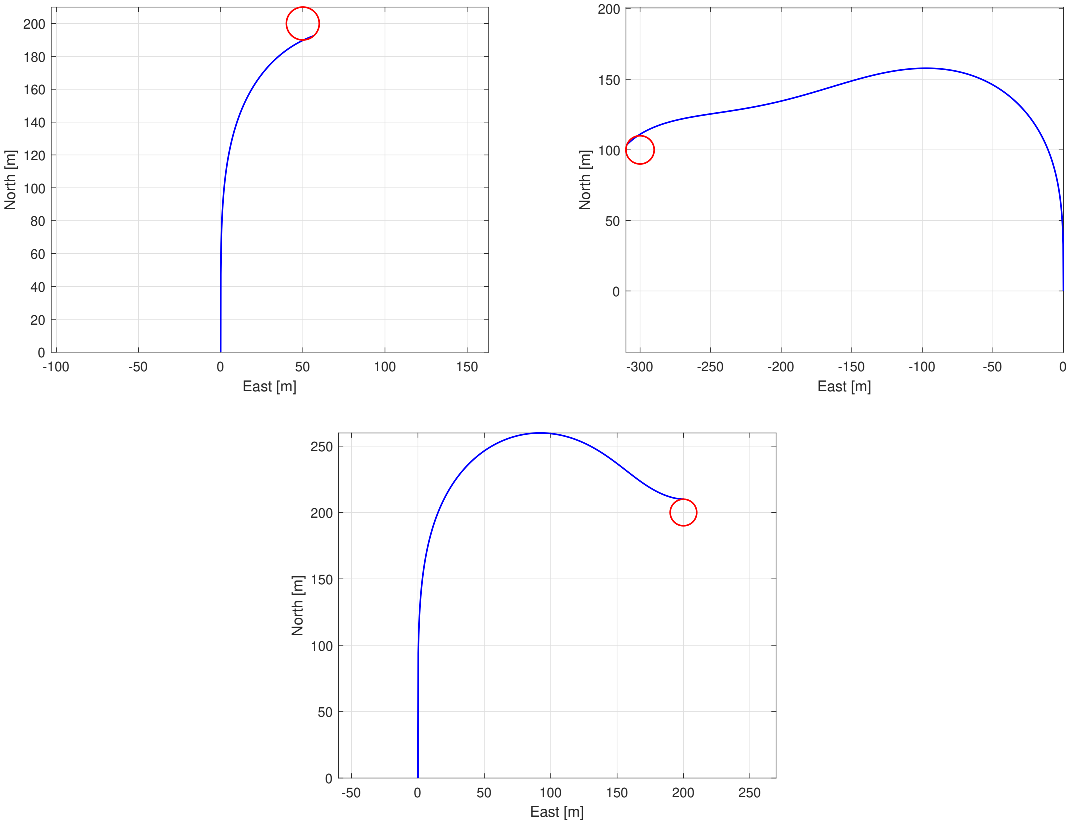

Figure 4.5 Nonlinear shooting trajectories with different final constraints (cases 3, 4 and 5)

the optimal path, for any $t \geq 0$ and with $k = \lfloor \frac{t}{\Delta t} \rfloor$:

$$x_1(t) = \sum_{n=0}^{k-1} \int_{t_n}^{t_{n+1}} V \cos\left(x_3(\tau)\right) \mathrm{d}\tau + \int_{t_k}^{t} V \cos\left(x_3(\tau)\right) \mathrm{d}\tau, \tag{4.23}$$

$$x_2(t) = \sum_{n=0}^{k-1} \int_{t_n}^{t_{n+1}} V \sin\left(x_3(\tau)\right) \mathrm{d}\tau + \int_{t_k}^{t} V \sin\left(x_3(\tau)\right) \mathrm{d}\tau, \tag{4.24}$$

with $t_{-1} = 0$. These two expressions must verify the constraints defined in Eq. 4.5 to stop the algorithm.

4.3.2 Algorithm

Mesh

The form of the mesh differs from usual Cartesian ones since the mesh is based on variations of δu. Several examples are presented in Fig. 4.6. From left to right, top to bottom, there are 3, 5, 7 or 9 possibilities of values for δu at each node. In each case, there are 5 time-steps of 5 s.

Search for optimal path

Classical graph solving algorithms are used to find the optimal path. The complexity of this process depends on the number of time-steps and the options available at each time step. In the worst-case scenario, the complexity of finding the optimal path would be similar to sorting a list of all the generated paths, which is equivalent to the total number of possible paths. The search for the optimal path can be performed quickly and accurately since the total number of paths to be considered is reduced by setting appropriate conditions within the dynamic programming algorithm. These conditions include the variations in δu possible at each time step and the bounds on heading x_3, which can be systematically determined as a function of the problem parameters. As a result, the customized grid of the algorithm, made possible by the new state space, facilitates the efficient exploration of potential paths and reduces computational complexity.

Limitations

The dynamic programming algorithm can be implemented either recursively or using nested for loops. The complexity of the algorithm, particularly in the latter approach, can be estimated by the number of choices for δu at each step raised to the power of the number of

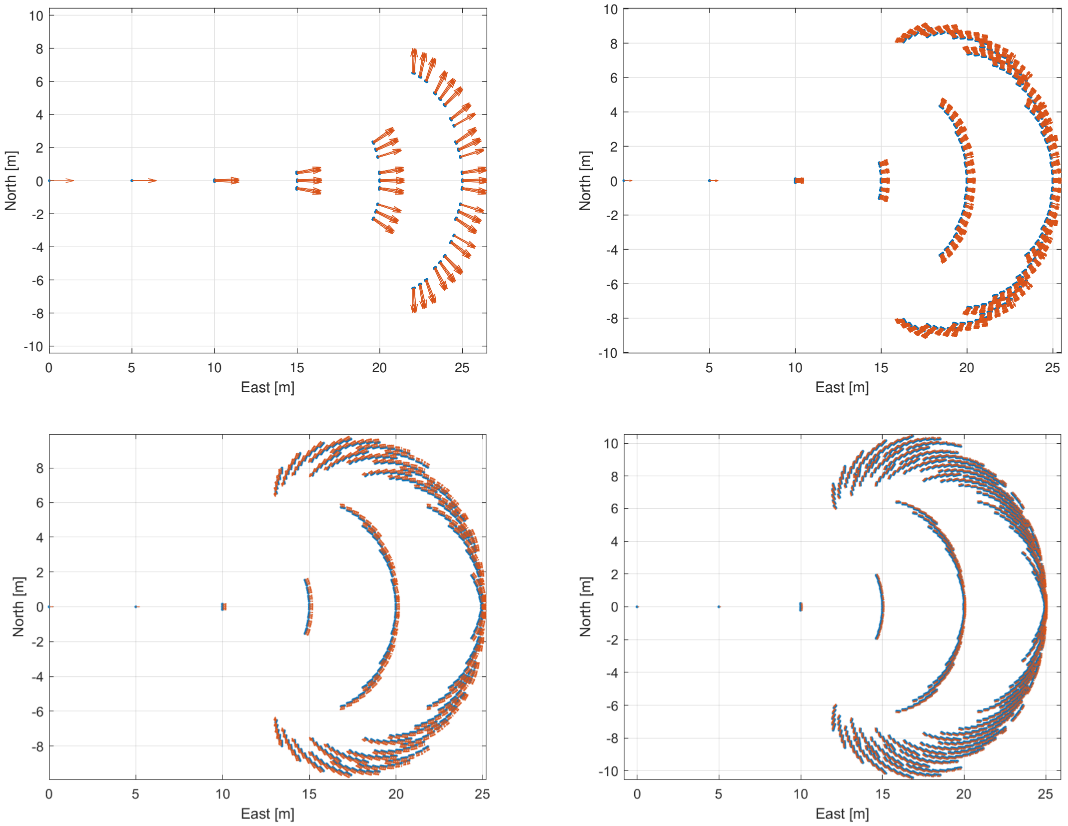

Figure 4.6 Examples of mesh for dynamic programming (3, 5, 7 or 9 possibilities at each step)

time steps. This implies that the algorithm can become computationally intensive. However, there are several strategies for reducing the computational load. Here is a list of possible choices to speed up computation:

- Precomputing the mesh before running the algorithm (on an embedded controller for instance), and roam through it to find the optimal path. Then, during execution, search for the optimal path within the precomputed mesh. This approach eliminates the need for extensive computation and reduces the process to simply finding the best path in a list. Another advantage with this kind of *pre-computation* is that, with a high space scale mesh, it becomes possible to choose not only the optimal path to a target, but the optimal path to the optimal target over several ones. This multiplicity is presented in Fig. 4.7.

  Among the 3 possible targets and 6 possible trajectories, the cheapest one would clearly be the #2 with no wind, as the distance is roughly the same between the 3 targets and the trajectory is mainly aligned with the initial speed vector. But with a strong full

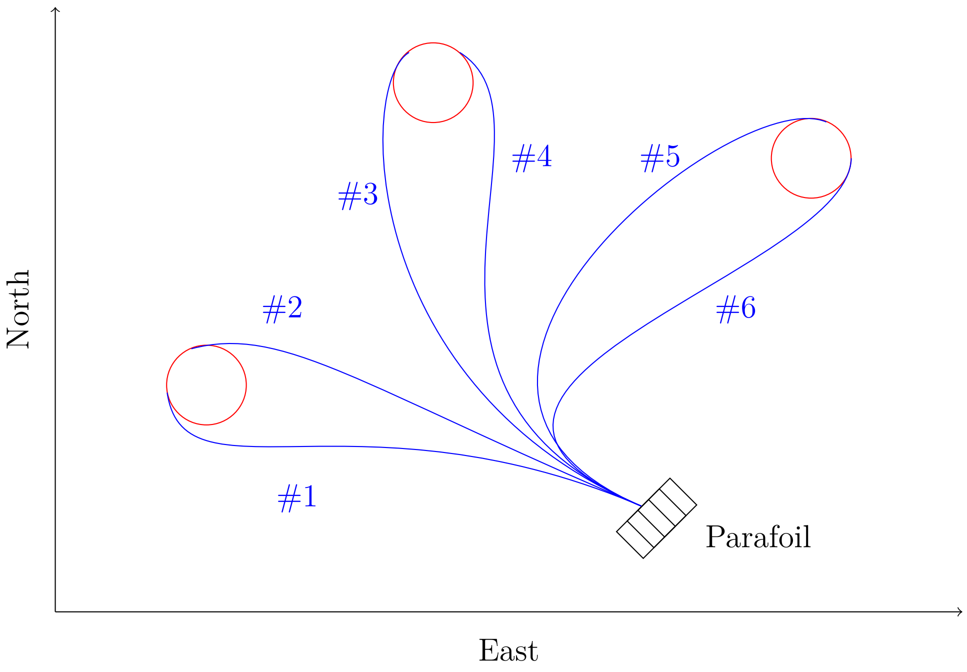

Figure 4.7 Several targets, each with several possible paths.

North wind, the best choice is to target the middle red circle. Then, the choice of paths is reduced to trajectories #3 and #4. Here, calculating a large mesh before the flights could be a very effective choice for reducing costs and allowing multiple possible targets, *i.e.*, robustifying the algorithm.

- Storing computed data, such as the nonlinear integrals of position, to avoid redundant calculations. Instead of recomputing the data, it can be retrieved from a list when needed. This strategy can significantly reduce computation time, particularly when the same computation is required multiple times. However, it may be constrained by memory limitations, especially in embedded systems.

- Defining reasonable bounds for the variables can potentially help avoid unnecessary mesh calculations. In addition, assuming that δu is smooth (which corresponds to physical intuition, as the shooting method has demonstrated that u can be expressed linearly in terms of λ_4, a differentiable function) can limit the range of possible values of δu. By reducing the range, the complexity of the algorithm is directly diminished.

- Seeking analytical expressions for maximum and minimum boundaries of state variables, particularly x_3, to exclude values that are either irrelevant or physically impossible to include in the mesh.

4.3.3 Results

Dynamic programming paths

The paths calculated using dynamic programming are presented in Fig. 4.8, and superimposed on the mesh (composed of space nodes and velocity vectors) to illustrate the possible choices of the algorithm, for cases 1, 3 and 5. At each time step, there were three possible values of δ_u for case 1, five possible values for case 3 and case 5. The step time is 12 s in each case.

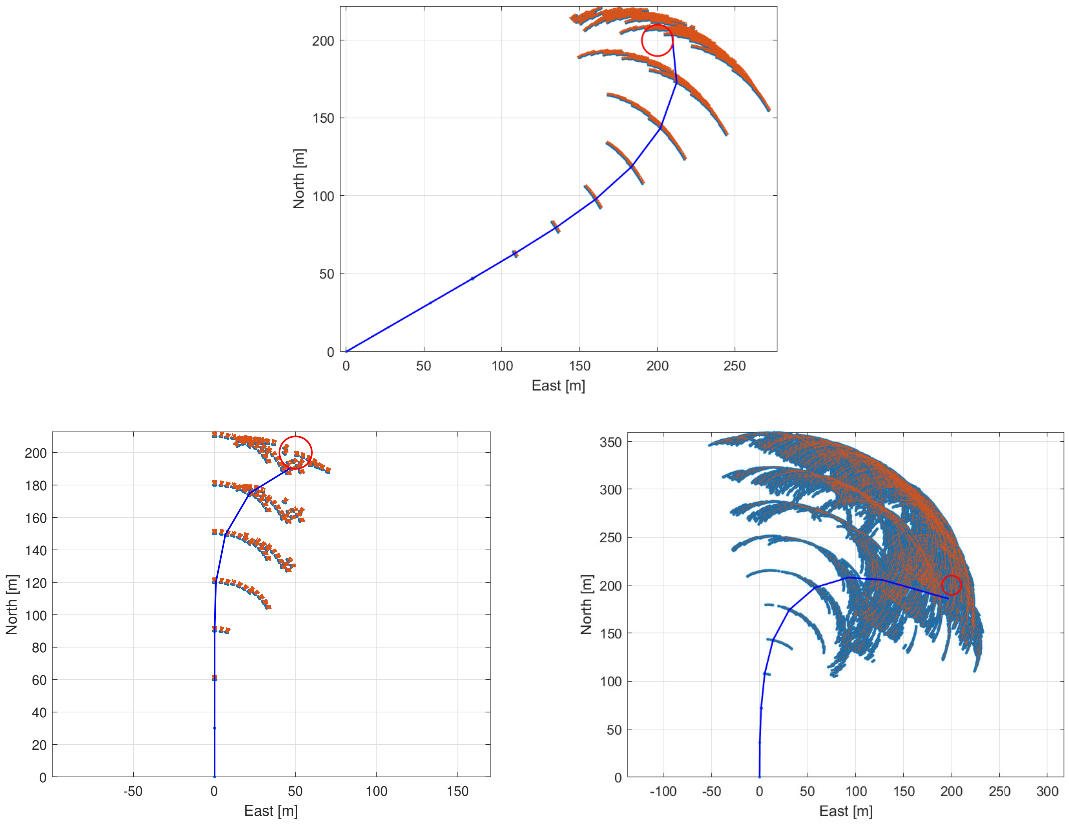

Figure 4.8 Optimal paths via DP, for different step times

Comparison of the methods on use cases

Optimal paths computed with dynamic programming are superimposed to nonlinear shooting optimal trajectories for cases 1, 3 and 5 in Fig. 4.9. These results have been obtained for the same precision in the mesh (space and time) as in Fig. 4.8.

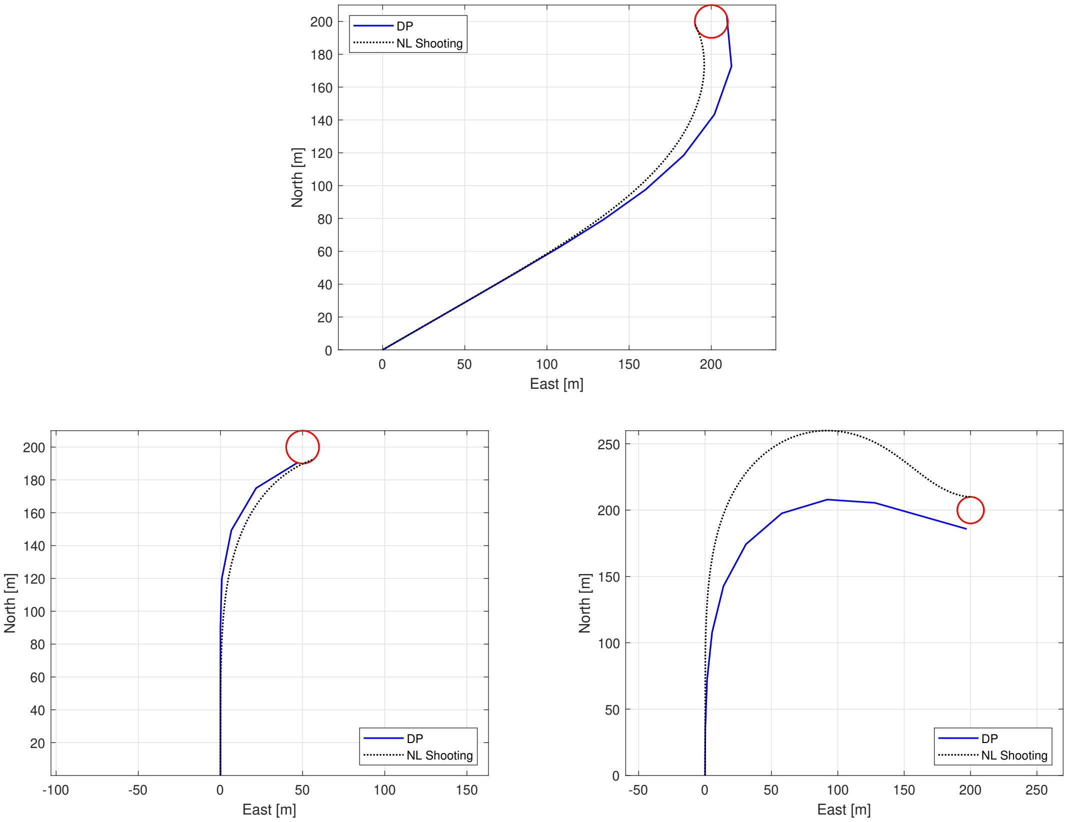

Figure 4.9 Superposition of optimal paths obtained via NLS and DP

4.4 Phases 2 and 3

4.4.1 Phase 2: Spiral down

The objective of this phase is to lose altitude *i.e.*, potential energy, without getting away from the desired landing point. Phase 2 begins as soon as the system enters the final circle ($(x_N - a)^2 + (x_E - b)^2 \leq R^2$), and if its altitude is sufficient ($z \geq z_{2,\min}$).

When those two conditions are met, the asymmetric control input δ_A is set to a constant value, which simulations have shown must not be maximal, otherwise the system will dive. The value has been empirically set at:

$$\delta_A = \mu \times 0.7 \times \delta_{A,\max}, \tag{4.25}$$

with $\mu = \pm 1$. The sign must be chosen according to the velocity vector at the end of phase

1, in order to properly rotate around the point (a, b):

$$\mu = \operatorname{sgn}\left[\sin \angle \left(\begin{bmatrix} V_N \\ V_E \end{bmatrix}, \begin{bmatrix} x - a \\ y - b \end{bmatrix} \right)\right] \sqrt{(x-a)^2 + (y-b)^2}. \tag{4.26}$$

Another advantage of this non-maximal control input is that the system will not turn with its minimum radius of rotation, which is a good thing for the final turn which may be tight ($R_{\text{min}} \approx 6\,\text{m}$ according to simulations). One explanation for the dive with maximum control input could be explained by the limitations of the model used (from [28]), whose system identification may not have been performed for maximum asymmetric input for a long time. In this phase too, the symmetrical input is not used and remains equal to 0.

Minimal altitude $z_{2,\text{min}}$

The minimum altitude $z_{2,\text{min}}$ such that a spiral descent is not necessary and landing must take place immediately can be computed thanks to the glide ratio GR (which must be known before the flight, or at least computed during it) and the radius of the targeted final circle. In the following chapter simulations, the value z_2, min is retained as the radius of the circle is $R = 10\,\text{m}$ and the turning glide ratio (without braking) is between 2 and 3: the final approach shall be between 20 and 30 meters long when projected on the NE plane. More generally, a value for $z_{2,\text{min}}$ whose effectiveness has been demonstrated by simulations is as follows:

$$z_{2,\text{min}} = R \times \text{GR}_t \tag{4.27}$$

with R the radius of the targeted circle and GR_t the mean turning glide ratio, which is computed as an average value over time. For a sufficiently long time T, the asymmetric control input is triangular:

$$\forall t \in [0, T], \quad \delta_A(t) = \begin{cases} \dfrac{2\delta_{A,\text{max}} t}{T} \\ \delta_{A,\text{max}} - \dfrac{2\delta_{A,\text{max}}}{T}\left(t - \dfrac{T}{2}\right) \end{cases} \tag{4.28}$$

and this manoeuvre makes possible the computation of GR_t. Denoting Δz the loss of altitude during the manoeuvre, it follows that

$$\text{GR}_t = \frac{1}{\Delta z} \int_0^T \sqrt{V_N^2(t) + V_E^2(t)}\text{d}t. \tag{4.29}$$

By simulation, the value $\text{GR}_t = 3.4$ is obtained for $T = 120\,\text{s}$ and $\delta_{A,\text{max}} = 10\,\text{cm}$. This computation can be done either by simulation or practically as a test flight, aiming at calibrating the controller with actual experimental values.

Coriolis effect

An interesting result is that with a long downward spiral, even without wind, the system drifts, as it does not rotate around a perfectly vertical axis. This result, which is insignificant when the downward spiral is not very long, can become a significant problem when it starts at high altitude and requires many rotations (see Fig. 4.10, where the asymmetric control input has been set to $\delta_A = 5\,\mathrm{cm}$ and the simulation run until the altitude becomes equal to 0). The red asterisk indicates the first point of the trajectory and the red dot the last. There are almost 13 complete rotations in this figure, and with a vertical speed of between 2 and 3 m/s, a deviation of 1 meter occurs at each turn. To solve this problem, it is possible to take into account the direction of rotation in order to shift the centre of rotation, as described in the next paragraph. However, this problem is often negligible compared with the deflection caused by the wind.

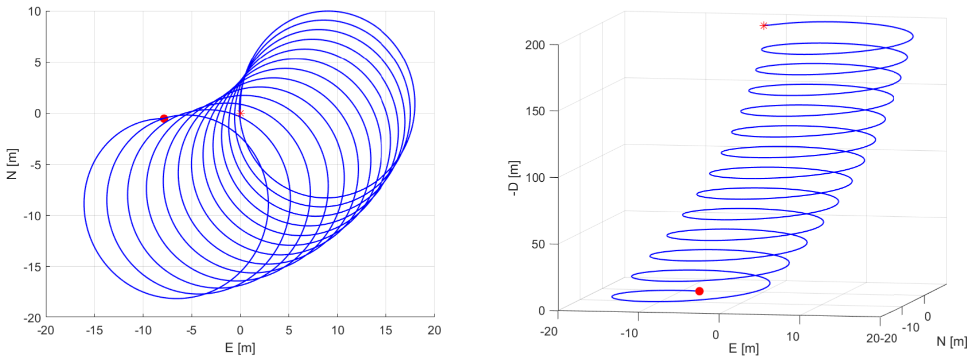

Figure 4.10 Deviation due to Coriolis effect

Dealing with the wind

Since the wind may cause a deviation during the spiral down if it is too strong, it may be necessary to offset the point around which the system rotates. The final switch between offset and real points can be done during the final phase only, and the system will eventually fly from the offset point to the real one, downwind. The distance it can cover is directly proportional to the glide ratio and the altitude remaining to be descended. Since the glide ratio facing the wind is lower than the nominal one (without wind), the distance has to be multiplied by a coefficient 0.8, determined empirically by simulations. The final offset vector is then given by

$$\mathbf{d} = 0.8 \times \mathrm{GR} \times z_{2,\min} \times \frac{\mathbf{V_w}}{||\mathbf{V_w}||}, \tag{4.30}$$

where $\mathbf{V_w}$ denotes the mean wind vector. An illustration of this offset is given in Fig. 4.11.

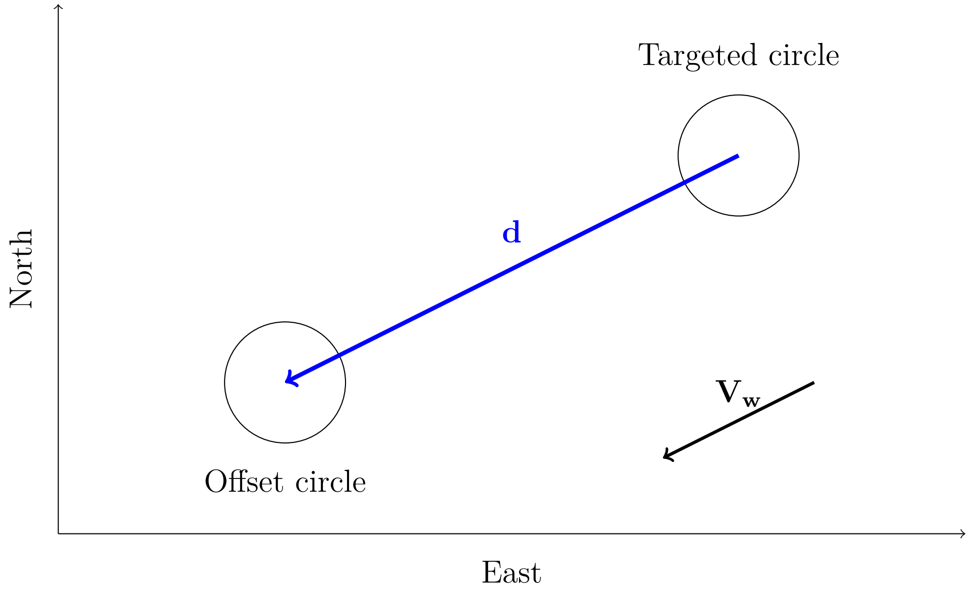

Figure 4.11 Offset of the targeted circle depending on the wind

4.4.2 Phase 3: Landing

This final phase is made of two steps; the final turn and the final landing. The final turn (using only δ_A) allows the system to orient itself into the wind ([10], [11]), while the final landing consists in going in a straight line facing the wind, and in pulling on both ropes at the same time to brake and pull up the front like a plane.

To ensure that its orientation is the same as the wind vector and the opposite direction, Bézier curves are used, inspired by [55]. Four points are used: the two extremes are the desired landing point ($D = (a, b)$), and the position of the system when the it begins its final turn ($A = (x, y)$), while the two control points are computed as follows: $B = (V_N \delta t, V_E \delta t)$ and $C = (V_{\mathrm{W,N}} \delta t, V_{\mathrm{W,E}} \delta t)$. This means that these two control points are such that the NED velocity vector of the system is colinear to (AB) and the wind velocity vector in the NED frame is colinear to (CD). The time constant is set to $\delta t = 1\,\mathrm{s}$, so that all the point coordinates are expressed in meters. A representation of this construction is given in Fig. 4.12.

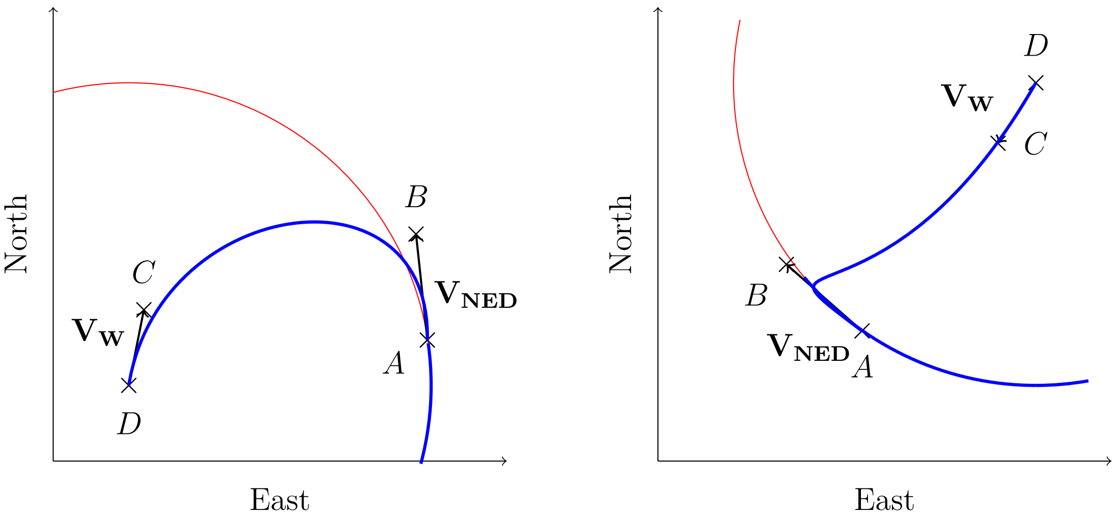

Figure 4.12 Examples of final turns using Bézier curves

Some turns may be sharp, but as the red circle has a radius $R > R_{\text{min}}$, *i.e.* greater than the minimum turning radius, the system will eventually align with the velocity vector even if there is a deviation during the initial turn.

It is also important to mention that the scenario where the system has to make an almost complete revolution (Fig. 4.13) cannot occur. Indeed, while there is a distance $R = 10\,\text{m}$ between points A and D, the distance between A and B or between C and D is always (as the system cannot fly under too harsh wind conditions, *i.e.*, a wind speed higher than $5\,\text{m/s}$) less than $5\,\text{m}$; and (AB) is always orthogonal to (AD). This is even more true when the red circle - representing the downward projection circle of the spiral in the NE plane - is not centred on D.

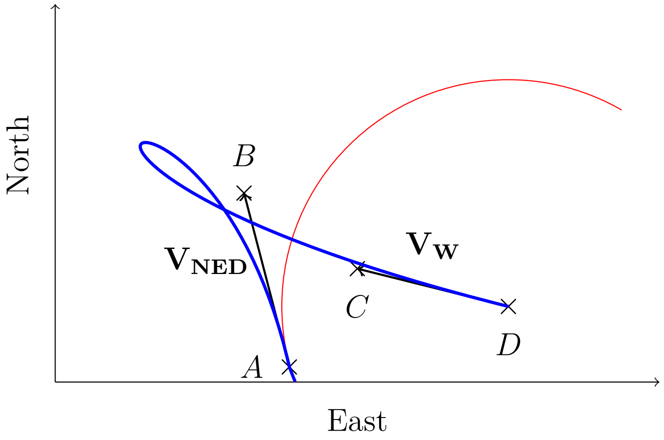

Figure 4.13 Impossible scenario

CHAPTER 5 REAL CONDITIONS SIMULATOR

This chapter focuses on the testing and validation of the algorithm developed in Chapter 4 within the Simulink simulator introduced in Chapter 3. The chapter is divided into two sections. The first section is dedicated to the comparison of dynamic programming and nonlinear shooting techniques, highlighting their complementarity and demonstrating the effectiveness of the hybrid algorithm in phase 1. The second section aims to showcase the algorithm's efficiency in the presence of wind and to present its performance in terms of optimization.

5.1 Implementation of the hybrid controller in phase 1

5.1.1 Path tracker

In order to simulate and track optimal paths generated by nonlinear shooting and dynamic programming in a 2D environment, a 3D simulator was developed using Matlab and Simulink, based on the work by Ward et al. (2013) [28]. A path tracker is implemented to follow these optimal paths, accounting for the fact that the timescales between the 2D and 3D models are not identical. This time discrepancy arises due to factors such as the variable airspeed, the influence of sideslip, and other realistic considerations. Given $\mathcal{T} \subset \mathbb{R}^2$ as the set of points representing an optimal trajectory generated either through nonlinear shooting or dynamic programming, the coordinates $(\overline{x}, \overline{y}, \overline{z})$ of the point to be tracked when the system is at $[x(t), y(t), z(t)]$ can be determined as follows:

$$(\overline{x}, \overline{y}) = \underset{(x,y)\in\mathcal{T}}{\mathrm{Argmin}}((x(t)-x)^2 + (y(t)-y)^2), \qquad \text{and} \qquad \overline{z} = z(t). \tag{5.1}$$

The trajectory is then followed by a PD controller to correct the relative lateral error

$$\varepsilon = \mathrm{sgn}\left[\sin\angle\left(\begin{bmatrix} V_N \\ V_E \end{bmatrix}, \begin{bmatrix} x-\overline{x} \\ y-\overline{y} \end{bmatrix}\right)\right]\sqrt{(x-\overline{x})^2 + (y-\overline{y})^2}, \tag{5.2}$$

represented in Fig. 5.1. It represents the norm of the lateral error, multiplied by the sign given by the scalar product between NED velocity vector and the vector between current position and targeted one, *i.e.*, +1 if the trajectory to track is on the right and −1 if it is on the left. In Fig. 5.1 for instance, as the tracked trajectory is on the right of the actual trajectory, $\varepsilon > 0$ so the asymmetric input δ_A shall be positive as well.

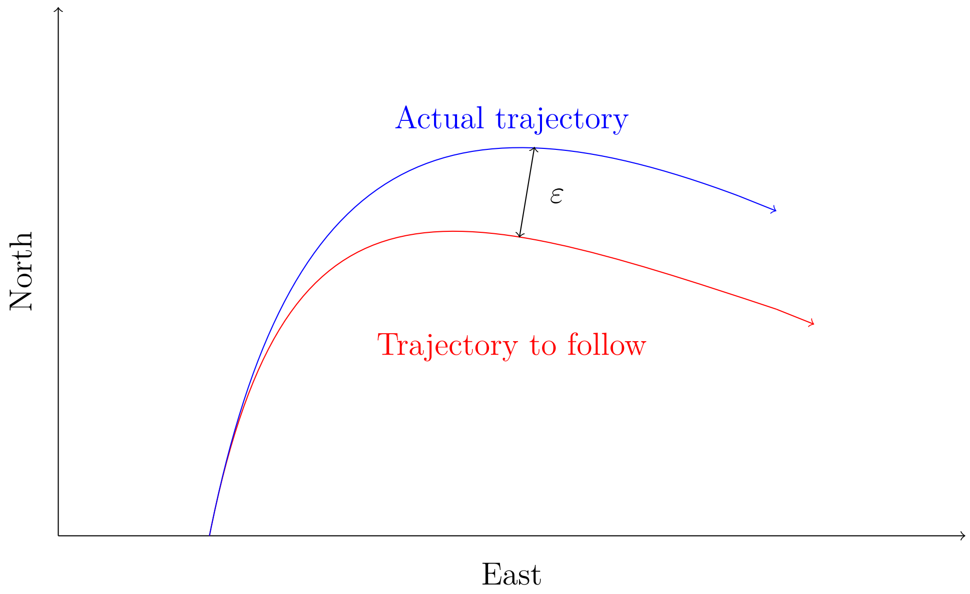

Figure 5.1 Error in trajectory tracking

The control law is given with Laplace formalism by:

$$\frac{\delta_A(s)}{\varepsilon(s)} = K_p + K_d s, \tag{5.3}$$

where $K_p = K_d = 5$ give good simulation results. The gain K_p has no unity (or must be multiplied by 10^{-2} if δ_A is expressed in cm); and K_d is expressed in seconds. The proportional term acts on the algebraic distance to the setpoint trajectory, while the one acting on the first derivative symbolizes the following control strategy: the more $\varepsilon > 0$ increases, the bigger positive correction is needed; and *idem* the more $\varepsilon < 0$ decreases, the bigger negative correction is needed. Here positive and negative adjectives refer to the sign of δ_A.

Better numerical results have been obtained (less oscillations in heading around the tracked trajectory) with a feedforward approximation. Instead of tracking the closer neighbour, the tracked point is further down in the tracked path. This is explained by the fact that the velocity in three dimensions can be higher than $V = 2.5\,\mathrm{m.s}^{-1}$. In this case, if the trajectory to follow is given by $\mathcal{T} = \{(x[k], y[k])\}_k$, the 'feedforward' is to track the point $(x[k + K_{\mathcal{T}}], y[k + K_{\mathcal{T}}])$, instead of the closer point $(x[k], y[k])$, where $K_{\mathcal{T}} \in \mathbb{N}^*$ depends on the trajectory to track (and is constant over time).

5.1.2 Comparison of DP and NLS costs on use cases

Table 5.1 presents a comparison of the nominal costs, expressed in cm^2.s, obtained from the 3D simulator when following different trajectories in the absence of wind. The associated overcosts, computed as follows, are also provided:

$$\text{Overcost} = \frac{J_{\text{DP}} - J_{\text{Shoot}}}{J_{\text{Shoot}}} \tag{5.4}$$

In Fig. 5.2, the 3D trajectory for case 1 is depicted. The obtained numerical values indicate that despite the discrete nature of waypoints in the dynamic programming path, the overall trajectory maintains a coherent shape. Additionally, the relatively smallness of the overcost relatively to the reference cost J_{Shoot} (except in case 5 where a backtrack is needed) demonstrates the effectiveness of the approach. To further improve smoothness, the time step can be reduced, although this would increase the computation time for both mesh generation and finding the shortest path. Alternatively, additional points can be generated from the continuous piecewise function that defines the optimal trajectory, resulting in a denser set of points that provide a smoother representation of the trajectory.

Use case	(a, b) [m]	$x_3(0)$ [rad]	J_{Shoot} [m^2.s]	J_{DP} [m^2.s]	Overcost
1	$(200, 200)$	$\pi/3$	5.59×10^{-2}	6.35×10^{-2}	13.6 %
2	$(200, 200)$	$-\pi/8$	2.52×10^{-2}	2.7×10^{-2}	7.1 %
3	$(50, 200)$	0	4.51×10^{-2}	4.91×10^{-2}	8.9 %
4	$(-300, 100)$	0	1.66×10^{-1}	1.9×10^{-1}	14.5 %
5	$(200, 200)$	0	3.14×10^{-1}	3.98×10^{-2}	27.0 %

Table 5.1 Comparison of nominal costs on the five use cases.

5.1.3 In-flight recomputing

One of the main contributions of this work is the introduction of in-flight computation of optimal trajectories using a fast and efficient dynamic programming algorithm. The principle behind this approach can be summarized as follows:

- Initially, a path is planned using nonlinear shooting. Two closely related trajectories, denoted as $\mathcal{T}_1$ and $\mathcal{T}_2$, are generated either by performing nonlinear shooting to target a circle near the original trajectory or by allowing a tolerance on the values of x and y.
- At each time t, the closest neighbors of the system within these trajectories, represented

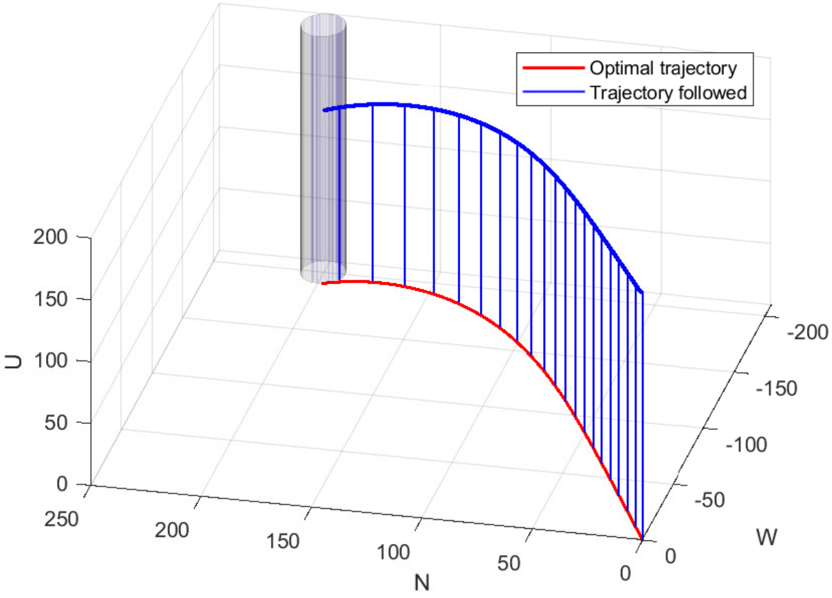

Figure 5.2 Shooting-generated trajectory tracked without wind (case 1)

as $(\overline{x}_1(t), \overline{y}_1(t)) \in \mathcal{T}_1$ and $(\overline{x}_2(t), \overline{y}_2(t)) \in \mathcal{T}_2$, are computed using Eq. 5.1. The condition

$$\begin{bmatrix} \overline{x}_1(t) - x(t) \\ \overline{y}_1(t) - y(t) \end{bmatrix} \cdot \begin{bmatrix} \overline{x}_2(t) - x(t) \\ \overline{y}_2(t) - y(t) \end{bmatrix} \leq 0 \iff \text{System between the limit trajectories} \quad (5.5)$$

allows to determine when the system reaches the boundary trajectories. Figure 5.3 (left) illustrates an example of boundary trajectories for case 3, depicted in black.

- When the system is no longer located between $\mathcal{T}_1$ and $\mathcal{T}_2$, it is necessary to recalculate the optimal path as it may have changed. Figure 5.3 (right) demonstrates that if the system deviates from its allowed flight state, the optimal path may be adjusted. In green, several waypoints are recomputed using dynamic programming. It is worth noting that the boundary trajectories can be recalculated each time to determine when another recalculation is required.

- Although the trajectory framing has been presented in two dimensions, it is also possible to define a cone to formulate a 3D condition similar to Cond. 5.5.

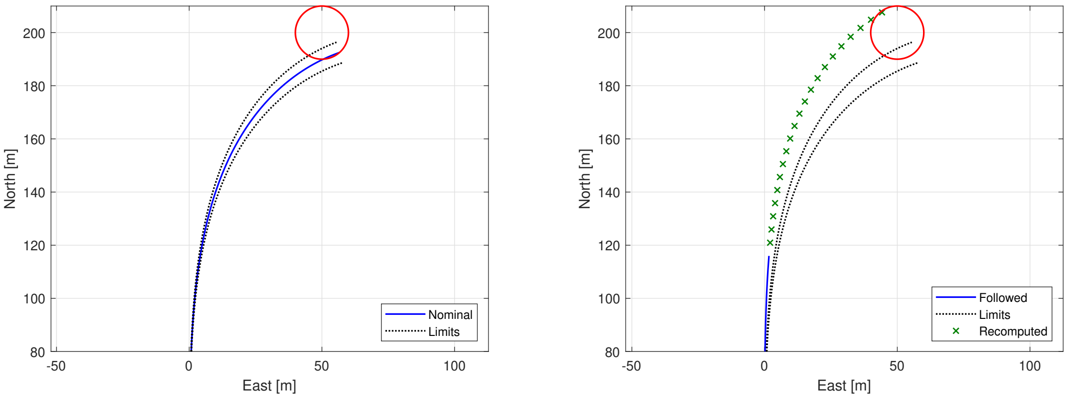

Figure 5.3 Complementarity of nonlinear shooting and dynamic programming

5.2 3D simulations with whole flight controller

5.2.1 Maximal bearable wind and gusts

Several simulation tests have been conducted to quantify the influence of the wind on the followed path. In particular, a flight beginning at 200 m high, towards the North and tracking this direction with a PI controller has been simulated. The system had to undergo a lateral wind (full West direction), generated thanks to the Dryden wind turbulence model [67], with a wind speed at 6 m varying from 0 to 20 m/s. Results in terms of max and mean values for ε (defined in Fig. 5.1) are presented in Fig. 5.4.

While the mean value is close to 0, which suggests that the controller effectively follows the trajectory macroscopically (or in 'global' terms), the maximum error values are as high as 5 m, which means that the actual position of the system happens to be very far away from what it should be. In other words, the idea of recomputing a new trajectory - when the actual one much differs from the nominal one to track - makes sense.

5.2.2 Structure of the assembled algorithm

The global controller structure is presented in Alg. 1.

The inputs are NED coordinates x, y, z, NED velocity components in the body frame V_N, V_E, V_D, Euler angles ϕ, θ, ψ and wind velocity components in NED frame $V_{w,N}, V_{w,E}, V_{w,D}$. The outputs are the control laws $\delta_{A,\mathrm{c}}$ and $\delta_{B,\mathrm{c}}$. In an actual flight, those input quantities can be measured with embedded sensors. NED coordinates and velocity components in particular are obtained from GPS data (conversion algorithm is given in [31]). The actual actuators control inputs (δ_A and δ_B, which appear in the motion equations) are linked with $\delta_{A,\mathrm{c}}$ and

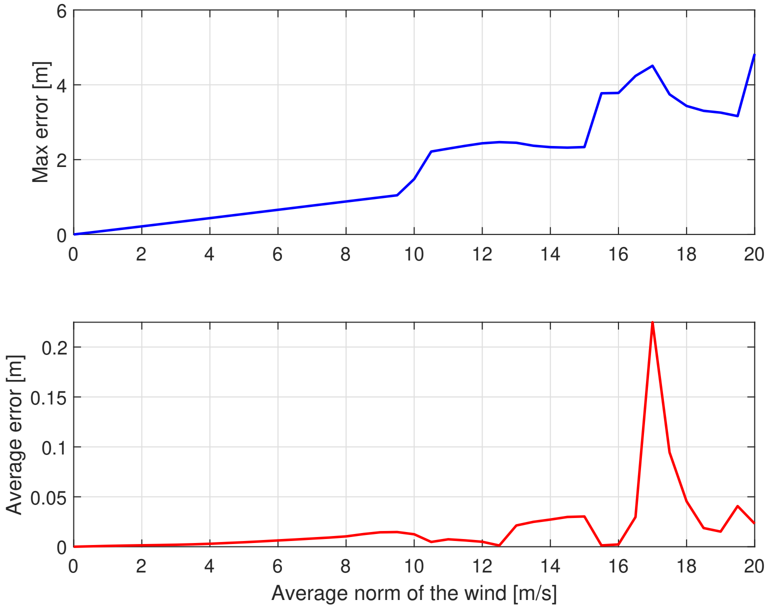

Figure 5.4 Influence of the wind on a rectilinear trajectory

$\delta_{B,\mathrm{c}}$ by the actuators time constant $\tau = 10\,\mathrm{s}$:

$$\frac{\delta_A(s)}{\delta_{A,\mathrm{c}}(s)} = \frac{1}{1+\tau s}, \tag{5.6}$$

$$\frac{\delta_B(s)}{\delta_{B,\mathrm{c}}(s)} = \frac{1}{1+\tau s}. \tag{5.7}$$

The parameter **phase** is a global variable. It begins equal to 1, but as soon as the inequality of phase two $(x-a)^2 + (y-b)^2 \leq R^2$ is verified for the first time, it is switched to the value of 2, so that the system does not risk to loiter again after having begun its spiral down move. Initial position is assumed not to be in the final circle. The Simulink model of this controller is given in Fig. 5.5. The three coloured areas represent the three phases of flight (blue for phase 1, purple/pink for phase 2 and yellow/brown for phase 3).

As mentionned in the path planning chapter, the final phase is made of two subphases. The first one consists in following a trajectory built with Bézier's curves for the last turn; and eventually, when the system velocity vector is almost aligned with the wind velocity, the asymmetric input becomes zero and the symmetric input comes into play to pull up the front. The decision criterion is:

$$\begin{bmatrix} V_N \\ V_E \end{bmatrix} \cdot \begin{bmatrix} V_{w,N} \\ V_{w,E} \end{bmatrix} \geq 0.8 \times \left\| \begin{bmatrix} V_N \\ V_E \end{bmatrix} \right\| \times \left\| \begin{bmatrix} V_{w,N} \\ V_{w,E} \end{bmatrix} \right\| \iff \text{Ready to land} \tag{5.8}$$

Algorithm 1 Flight controller

```
Require: $x, y, z, V_N, V_E, V_D, \phi, \theta, \psi, V_{w,N}, V_{w,E}, V_{w,D}$    ▷ Measurements
Require: $a, b, R, \mathcal{T}_{\text{nom}}, \mathcal{T}_1, \mathcal{T}_2, K_p, K_i, z_{2,\text{min}}, \delta_{A,\text{max}}, \delta_{B,\text{max}}$, phase    ▷ Constant parameters
  Phase 1
  if $z \geq z_{2,\text{min}}$ then
      if $(x-a)^2 + (y-b)^2 > R^2$ then
          if Cond. 5.5 is not true then    ▷ $\mathcal{T}_{\text{nom}}$ not well-followed
              Recompute $\mathcal{T}_{\text{nom}}, \mathcal{T}_1, \mathcal{T}_2$ by DP    ▷ From current coordinates $x, y$ NED velocity
          end if
          $(\overline{x}, \overline{y}) = \underset{(x,y) \in \mathcal{T}_{\text{nom}}}{\text{Argmin}} \left( (x-x)^2 + (y-y)^2 \right)$    ▷ Eq. 5.1
          $\varepsilon \leftarrow \text{sgn}\left[ \sin \angle \left( \begin{bmatrix} V_N \\ V_E \end{bmatrix}, \begin{bmatrix} x-\overline{x} \\ y-\overline{y} \end{bmatrix} \right) \right] \sqrt{(x-\overline{x})^2 + (y-\overline{y})^2}$    ▷ Eq. 5.2
          $\delta_{A,\text{c}}(s) = (K_p + K_d s)\,\varepsilon(s)$
          $\delta_{B,\text{c}} = 0$
      end if
  end if
  Phase 2
  if $z \geq z_{2,\text{min}}$ then
      if $\{(x-a)^2 + (y-b)^2 \leq R^2$ or phase $= 2\}$ then
          $\mu \leftarrow \text{sgn}\left[ \sin \angle \left( \begin{bmatrix} V_N \\ V_E \end{bmatrix}, \begin{bmatrix} x-a \\ y-b \end{bmatrix} \right) \right] \sqrt{(x-a)^2 + (y-b)^2}$    ▷ Eq. 4.26
          $\delta_{A,\text{c}} = \mu 0.7 \delta_{A,\text{max}}$    ▷ Eq. 4.25
          phase $= 2$
          $\delta_{B,\text{c}} = 0$
      end if
  end if
  Phase 3
  if $z < z_{2,\text{min}}$ then
      if Cond. 5.8 is not true then    ▷ Velocity not against the wind
          if phase $\neq 2$ then
              Recompute $\mathcal{T}_{\text{nom}}$ with Bézier's curves    ▷ Only happens once
          end if
          phase $= 3$
          $(\overline{x}, \overline{y}) = \underset{(x,y) \in \mathcal{T}_{\text{nom}}}{\text{Argmin}} \left( (x-x)^2 + (y-y)^2 \right)$    ▷ Eq. 5.1
          $\varepsilon \leftarrow \text{sgn}\left[ \sin \angle \left( \begin{bmatrix} V_N \\ V_E \end{bmatrix}, \begin{bmatrix} x-\overline{x} \\ y-\overline{y} \end{bmatrix} \right) \right] \sqrt{(x-\overline{x})^2 + (y-\overline{y})^2}$    ▷ Eq. 5.2
          $\delta_{A,\text{c}}(s) = (K_p + K_d s)\,\varepsilon(s)$
          $\delta_{B,\text{c}} = 0$
      end if
      if Cond. 5.8 is true then
          $\delta_{A,\text{c}} = 0$
          $\delta_{B,\text{c}} = \delta_{B,\text{max}}$
      end if
  end if
```

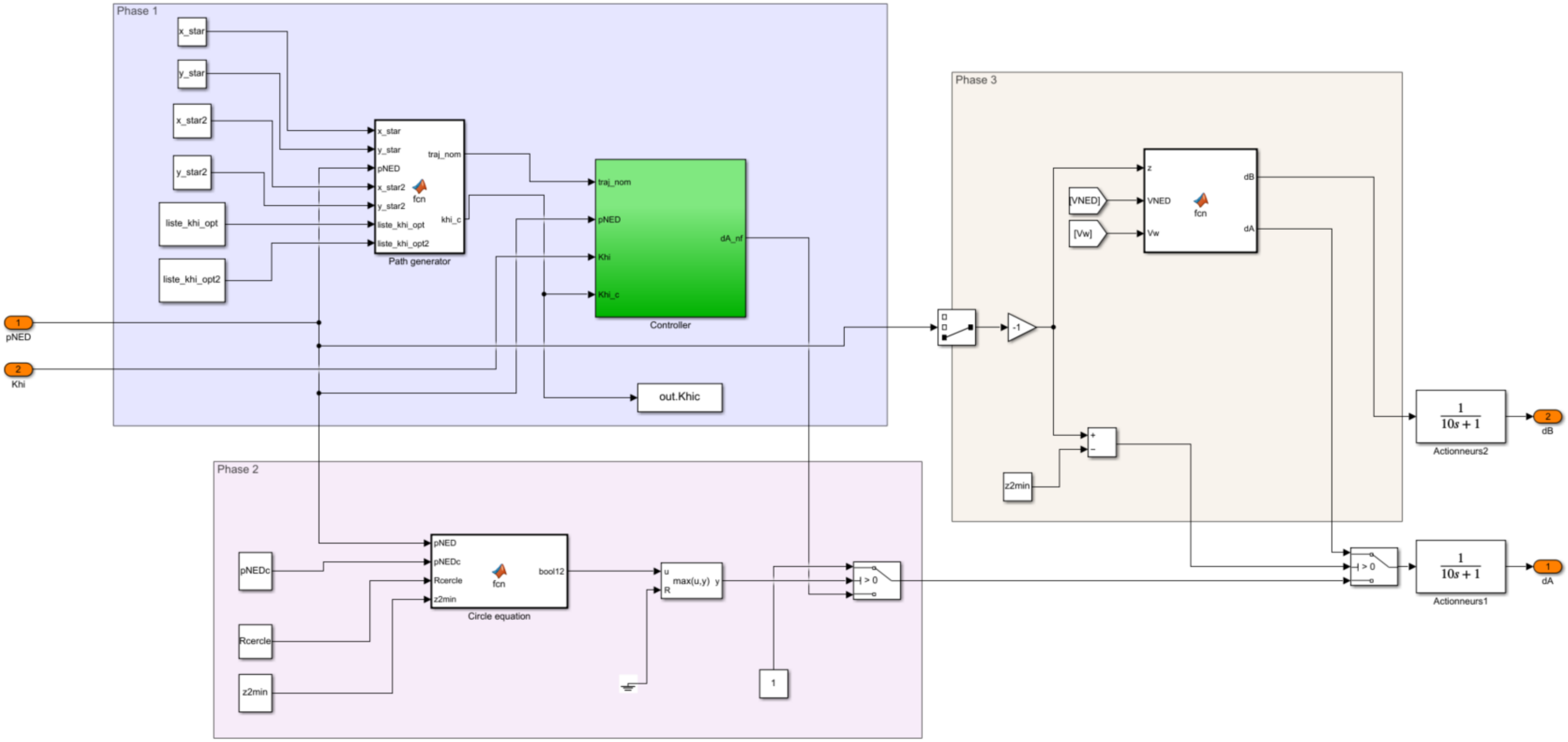

Figure 5.5 Entire controller built with Simulink

with V_N and V_E the North and East components of $\mathbf{V}_{\mathrm{b/NED}}^{\mathrm{NED}}$; and $V_{w,N}$ and $V_{w,N}$ the North and East components of $\mathbf{V}_{\mathrm{w/NED}}^{\mathrm{NED}}$. A decision diagram which sums up the control algorithm decision process in a more 'meaningful' way is presented in Fig. 5.6. The black boxes are questions, the blue ones indicate a need to compute a new path, and the red ones indicate the form of the control inputs to plus in the system.

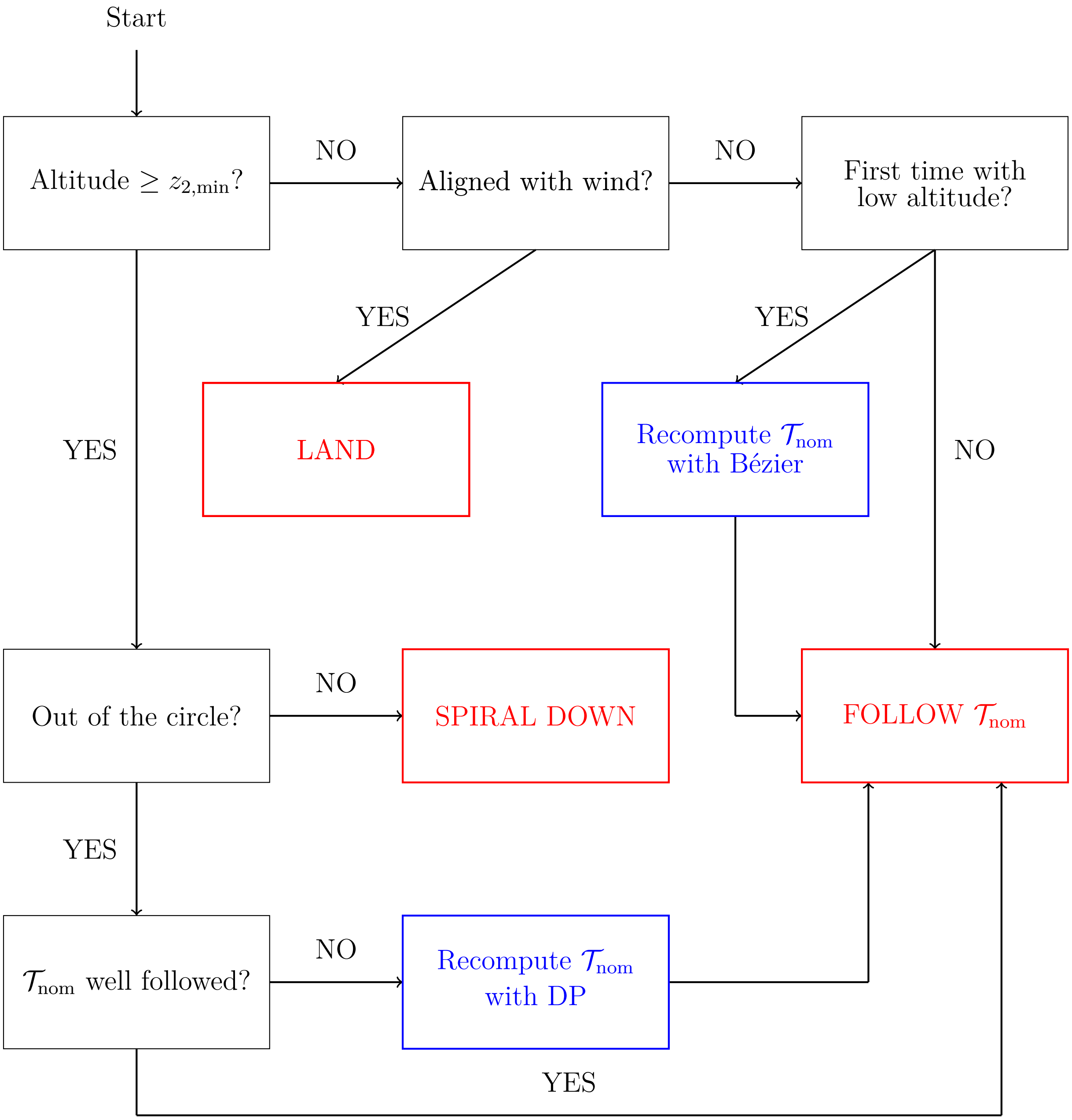

Figure 5.6 Decision Tree

5.2.3 Final results

Quantitative impact on phase 1

The contribution of the algorithm is presented in Tab. 5.2, where a comparison between three control efforts is quantified. The nominal cost is the one obtained by following the nominal path given by the shooting method; the nominal cost with wind is obtained by forcing the system to follow the nominal trajectory with the PD controller despite the wind; and the hybrid with wind corresponds to the cost obtained when an onboard recomputing is conducted thanks to dynamic programming. Energy saving is given in percentage in the last column, and is computed as follows:

$$\text{Undercost} = \frac{\text{Nominal cost with wind} - \text{Hybrid cost with wind}}{\text{Nominal cost with wind}}. \tag{5.9}$$

The chosen metrics has been arbitrarily chosen as if $|\varepsilon| > 5\,\text{m}$, a new trajectory must be recomputed. Obviously, numerical values presented in Tab. 5.2 depend on this choice; yet, the contribution of the algorithm remains the same.

Use case	Nominal [m^2.s]	Nominal, wind [m^2.s]	Hybrid, wind [m^2.s]	Saving
1	5.59×10^{-2}	3.47×10^{-1}	3.09×10^{-1}	11.0%
2	2.52×10^{-2}	3.06×10^{-1}	2.62×10^{-1}	15.0%
3	4.51×10^{-2}	6.44×10^{-1}	5.95×10^{-1}	7.6%
4	1.66×10^{-1}	9.85×10^{-1}	8.78×10^{-1}	10.9%
5	3.14×10^{-1}	4.37	4.08	6.6%

Table 5.2 Comparison of costs in phase 1, with and without wind

To demonstrate the effectiveness of the algorithm, strong and binding winds have been chosen. While the amplitude is set to 17 m/s in each case, the mean direction of the wind is given by ψ_W, which is set to make the trajectory deviate as much as possible for a better illustration of the utility of the present study. The values are respectively $\psi_W = -\pi/4$ in cases 1 and 2, $\psi_W = \pi/2$ in cases 3 and 4, and $\psi_W = 0$ in case 5. Figure 5.7 shows the recomputing of the trajectory to follow and the actually followed trajectory in the NE plane (built with the 3D simulator but projected onto the NE plane).

These results show that despite the slight sub-optimality of DP compared to NLS, the global performances, *i.e.*, over the whole trajectory, are better. That means that this hybrid method - consisting in accepting a slight sub-optimality for embedding purposes - still leads to a more optimal result.

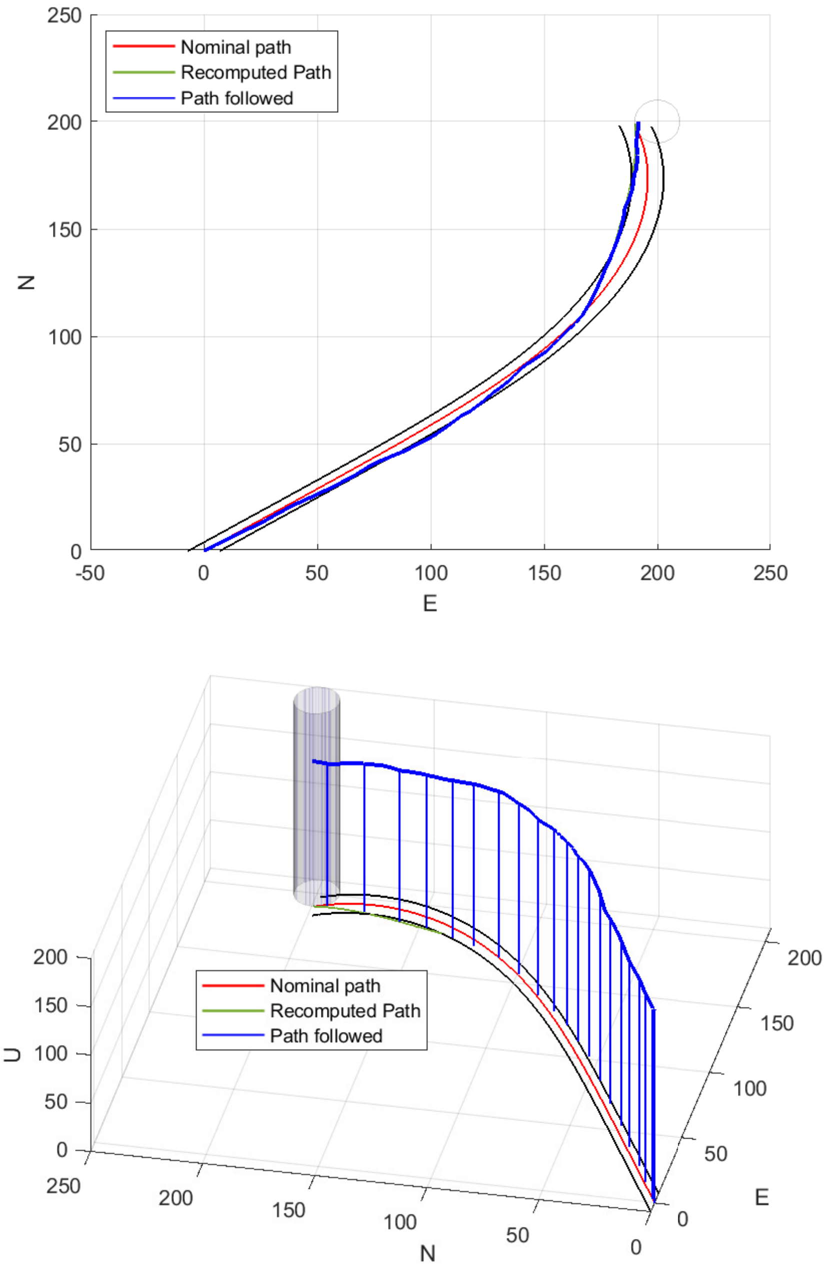

Figure 5.7 Hybrid path planning in phase 1

In this set of examples, paths are only recomputed once but this method can easily be adapted to draw new limit trajectories around the newly computed one with DP. The process of recomputing again as soon as the systems gets out of these boundaries can be applied ever again.

It is also interesting to mention that a new recomputation is not always needed: as shown in

Fig. 5.8, if the wind is not too high (only 8 m.s^{-1} in this example), the PD makes the system correctly follow the nominal trajectory.

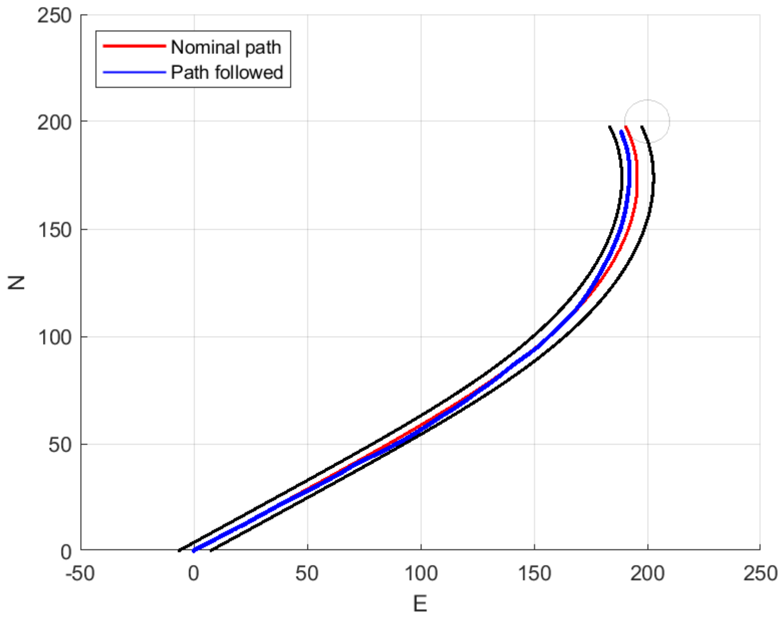

Figure 5.8 No need to change trajectory when the limits are not exceeded

Complete trajectory with recomputing

To illustrate the whole implementation of the control algorithm, Fig. 5.9 shows a trajectory in presence of wind, where a recomputing happened during phase one, and where altitude was high enough to necessitate a spiral down before landing. The minimal altitude $z_{\text{min}} = 10\,\text{m}$ is represented by a green plane. This figure represents the use case 1 as well. In Fig. 5.10, a zoom in on the final approach (phase 3) is represented. It is important to remember that the spiral down phase is not mandatory, and as soon as the altitude is lower than $z_{2,\text{min}}$, the landing phase takes place (even far from the desired landing point) to minimize crash damage.

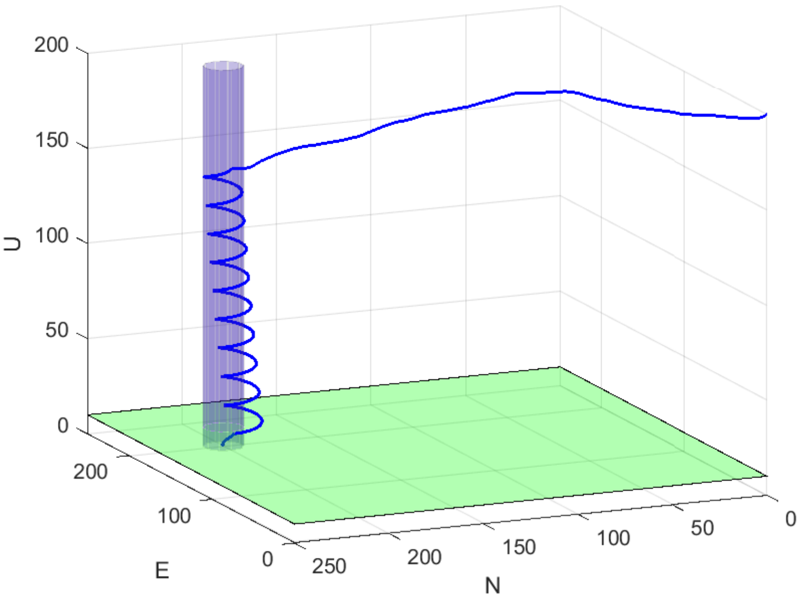

Figure 5.9 Complete trajectory

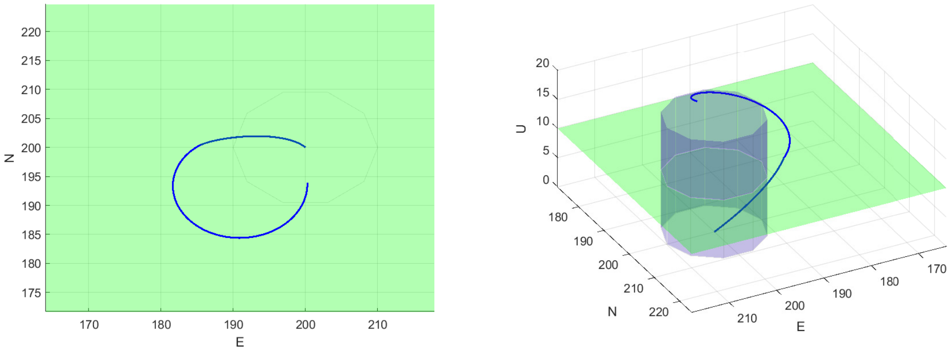

Figure 5.10 Transition from spiral down (phase 2) to landing (phase 3)

CHAPTER 6 CONCLUSION

6.1 Summary of results

The main objective of the present work was to design and validate through simulation an optimal controller for the descent of a stratospheric gondola. Although the simulation tests have been conducted with rather low altitudes ($< 500\,\mathrm{m}$), because of a lack of high-altitudes models, a new optimal controller has been proposed and its effectiveness has been demonstrated through simulation. It applies to parafoil systems, which is the chosen technological solution, but can be adapted to different environments (same principle for higher altitude, air density, etc.) or to other flying systems (planes, drones, etc.).

Firstly, parafoil systems have been identified as the best technological choice for transporting the basket to the desired location, therefore reaching **SO1**. A complete dynamic model has been established from the literature. After a justification of the choice of the number of DOF, a simulator has been built thanks to Simulink. Longitudinal and lateral equilibria, as well as the associated aerodynamic modes has been studied. The comparison of these modes with different models from the literature showed the effectiveness of the chosen model. A quantitative study of the correlation between system mass, parafoil surface and steady state velocity has been introduced, in line with **SO2**.

Then, a completely new optimal controller has been introduced. Based on the complementarity of nonlinear shooting and dynamic programming - in terms of precision and embeddability - the novelty lies in the absence of need to incorporate wind model in it. Its global optimality has been proven for the gliding phase of flight. Spiralling down and landing have been studied and linked by the introduction of a final turn approach based on Bézier curves. This whole algorithm met the previously defined requirements and thus completed **SO3**.

This hybrid controller has been tested in 3D in the following chapter, and put together with spiralling down and landing control laws. Several tests conducted with different wind and gusts profiles generated thanks to the Dryden model have shown that using this algorithm is both robust to windy perturbations, and it enables substantial energy savings, up to 15% for a flight of several minutes. The performances are better than the ones presented in the literature with algorithm using simplified wind models (almost constant direction, almost constant amplitude), confirming the achievement of **SO4**.

6.2 Limitations and future improvements

The first improvements to bring to this theoretical study would be to enhance the dynamical models, in particular when it comes to high altitude, as all the models presented in the literature are built on system identification for low-altitude parafoil systems. Plus, the one used in this master thesis is quite optimistic: glide ratio is higher than observed during experiments (often around 4 instead of 3 in straight line for instance; or wind resistance which is quite good up to 8 m/s whereas the experiments conducted by Lux show that it is hard to correctly fly over 5 m/s with an equivalent system). Still in terms of modelling improvement, sensors and actuators models could be added/improved as well. This leads to more practical considerations: Extended Kalman Filters have not been mentionned in this study as they are very frequently included in any embedded computer, but a study of such filters to do embedded identification, for instance, could be another field of work.

In the same way, the controller might be improved with an additional control input such as gliding slope, as sometimes suggested in the literature. The PD controller may be more efficient - especially if altitude variations are huge - if the gains change along time; which means having an adaptive PD or PID controller instead of a constant PD one to track reference trajectories. Finally, had the DP meshing to be made faster to limit computation time, an idea would be to find an analytical boundary on the control input, expressed in terms of time and physical constants, which would prevent the algorithm from meshing nodes physically impossible to reach.

Finally, there are two improvements of the controller efficiency that would be interesting and easily doable from the proposed one: firstly, to let it choose between several target circles (which would first consist in applying NLS to all the possible circles, to find the best one for each; and then choosing the minimal cost trajectory among them). Then, applying the same process during flight with DP would also be possible and very straightforward if all targeted circles are included in the previously made mesh. The second improvement would be more theoretical: it consists in finding a metric to automatically characterize the limit trajectories which are boundaries not to be crossed, from the distance to the nominal trajectory, or from wind estimations or measurements for instance..

REFERENCES

[1] S.-J. Lee, "Development of optimized path planning and autonomous control for return-to-point vehicle of high altitude ballooning," *PhD thesis*, 2014.

[2] C. C. of Forest Ministers, "National forestry database, forest fires," in *National Forestry Database*, 2023. [Online]. Available: http://nfdp.ccfm.org/en/data/fires.php

[3] S. Herrington, J. Renzelman, T. Fields, and O. Yakimenko, "Modeling and control of a steerable cruciform parachute system through experimental testing," *AIAA Scitech 2019 Forum*, 2019.

[4] O. A. Yakimenko, V. N. Dobrokhodov, I. I. Kaminer, and S. Dellicker, "Synthesis of optimal control and flight testing of an autonomous circular parachute," *Journal of Guidance Control and Dynamics*, 2004.

[5] G. Brown, R. Haggard, R. Almassy, R. Benney, and S. Dellicker, *The Affordable Guided Airdrop System (AGAS).* American Institute of Aeronautics and Astronautics, 1999.

[6] S.-I. Higashino, M. Hayashi, S. Nagasaki, S. Umemoto, and M. Nishimura, "A balloon-assisted gliding uav for aerosol observation in antarctica," *Transactions of the Japan Society for Aeronautical and Space Sciences, Aerospace Technology Japan*, 2014.

[7] O. Yakimenko, N. Slegers, and R. Tiaden, *Development and Testing of the Miniature Aerial Delivery System Snowflake.* American Institute of Aeronautics and Astronautics, 2015.

[8] S. Dunker, J. Huisken, D. Montague, and J. Barber, "Guided parafoil high altitude research (gphar) flight at 57,122 ft," in *23rd AIAA Aerodynamic Decelerator Systems Technology Conference.* American Institute of Aeronautics and Astronautics, 2015.

[9] Y. Li, M. Zhao, M. Yao, C. Qi, R. Guo, T. Sun, T. Jiang, and Z. Zhao, "6-dof modeling and 3d trajectory tracking control of a powered parafoil system," *IEEE Access*, 2020.

[10] U. Soppa, T. Görlach, and A. J. Roenneke, "German contribution to the x-38 crv demonstrator in the field of guidance, navigation and control (gnc)," *Acta Astronautica*, vol. 56, no. 8, pp. 737–749, 2005.

[11] J. Tao, Q. lin Sun, P. long Tan, Z. qiang Chen, and Y. ping He, "Autonomous homing control of a powered parafoil with insufficient altitude," *ISA Transactions*, vol. 65, pp. 516–524, 2016.

[12] T. Fields and O. Yakimenko, "The use of a steerable single-actuator cruciform parachute for targeted payload return," *IEEE*, 2017.

[13] E. Mooij, Q. Wijnands, and B. Schat, *9 dof Parafoil/Payload Simulator Development and Validation.* American Institute of Aeronautics and Astronautics, 2012.

[14] O. Prakash and N. Ananthkrishnan, *Modeling and Simulation of 9-DOF Parafoil-Payload System Flight Dynamics.* American Institute of Aeronautics and Astronautics, 2012.

[15] N. Slegers and C. Gorman, *Comparison and Analysis of Multi-body Parafoil Models With Varying Degrees of Freedom.* American Institute of Aeronautics and Astronautics, 2012.

[16] D. Saussié, "Commande de vol et moteurs," *Cours AER8410, École Polytechnique de Montréal*, 2021.

[17] B. Stevens, F. Lewis, and E. Johnson, *Aircraft Dynamics and Classical Control Design.* John Wiley & Sons, Ltd, 2015, ch. 4, pp. 250–376.

[18] M. Ward and M. Costello, "Adaptive glide slope control for parafoil and payload aircraft," *Journal of Guidance, Control, and Dynamics*, 2013.

[19] M. Ward, M. Costello, and N. Slegers, "Specialized system identification for parafoil and payload systems," *Journal of Guidance, Control, and Dynamics*, 2012.

[20] P. Lissaman and G. Brown, *Apparent mass effects on parafoil dynamics.* American Institute of Aeronautics and Astronautics, 1993.

[21] O. A. Yakimenko, "Precision aerial delivery systems: Modeling, dynamics, and control," in *Precision Aerial Delivery Systems: Modeling, Dynamics, and Control*, 2015.

[22] Z. Zhixiang, Z. Zining, and F. Yongling, "Dynamics analysis and simulation of six dof parafoil system," *Cluster Computing*, vol. 22, pp. 12 669–12 680, 2019.

[23] P. Crimi, "Lateral stability of gliding parachutes," *Journal of Guidance, Control, and Dynamics*, vol. 13, no. 6, pp. 1060–1063, 1990.

[24] G. Losilevskii, "Center of gravity and minimal lift coefficient limits of a gliding parachute," *Journal of Aircraft*, vol. 32, no. 6, pp. 1297–1302, 1995.

[25] N. Slegers and M. Costello, "Aspects of control for a parafoil and payload system," *Journal of Guidance, Control, and Dynamics*, vol. 26, no. 6, pp. 898–905, 2003.

[26] P. D. Team and Community, "Px4 user guide - controller diagrams," in *PX4 User Guide*, 2023. [Online]. Available: https://docs.px4.io/main/en/flight_stack/controller_diagrams.html

[27] V. Gomez, N. Gomez, J. Rodas, E. Paiva, M. Saad, and R. Gregor, "Pareto optimal pid tuning for px4-based unmanned aerial vehicles by using a multi-objective particle swarm optimization algorithm," *Aerospace*, vol. 7, no. 6, 2020.

[28] N. Slegers, E. Beyer, and M. Costello, "Use of variable incidence angle for glide slope control of autonomous parafoils," *Journal of Guidance, Control, and Dynamics*, vol. 31, no. 3, pp. 585–596, 2008.

[29] N. Slegers and M. Costello, "Model predictive control of a parafoil and payload system," *Journal of Guidance, Control, and Dynamics*, vol. 28, no. 4, pp. 816–821, 2005.

[30] N. Slegers and O. Yakimenko, "Optimal control for terminal guidance of autonomous parafoils," in *20th AIAA Aerodynamic Decelerator Systems Technology Conference and Seminar*, 2009, p. 2958.

[31] B. L. Stevens, F. L. Lewis, and E. N. Johnson, *Aircraft control and simulation: dynamics, controls design, and autonomous systems.* John Wiley & Sons, 2015.

[32] V. L. S. Franck L. Lewis, Draguna Vrabie, "Optimal control, 3rd edition," *Prentice-Hall, Englewood Cliffs, New Jersey*, 1977.

[33] D. E. Kirk, "Optimal control theory : An introduction," *Prentice-Hall, Englewood Cliffs, New Jersey*, 1970.

[34] A. P. Sage and C. C. White, "Optimum systems control, 2nd edition," *Prentice-Hall, Englewood Cliffs, New Jersey*, 1977.

[35] R. Gourdeau, "Conception de systèmes de commande," *Cours ELE6210, École Polytechnique de Montréal*, 2022.

[36] D. Hull, *Optimal Control Theory for Applications*, ser. Mechanical Engineering Series. Springer New York, 2013.

[37] J. T. Betts, "Survey of numerical methods for trajectory optimization," *Journal of Guidance, Control, and Dynamics*, vol. 21, no. 2, pp. 193–207, 1998.

[38] J. Betts, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming, Second Edition*, 2nd ed. Society for Industrial and Applied Mathematics, 2010.

[39] E. Pellegrini and R. P. Russell, "A multiple-shooting differential dynamic programming algorithm. part 1: Theory," *Acta Astronautica*, vol. 170, pp. 686–700, 2020.

[40] N. Murali, D. M., and A. K. W. C., "Parafoil trajectory comparison for optimal control and proportional controller," in *2013 International Conference on Control Communication and Computing (ICCC)*, 2013, pp. 227–232.

[41] N. Murali, M. Dineshkumar, K. W. Arun, and D. Sheela, "Guidance of parafoil using line of sight and optimal control," *IFAC Proceedings Volumes*, vol. 47, no. 1, pp. 870–877, 2014, 3rd International Conference on Advances in Control and Optimization of Dynamical Systems (2014).

[42] A. Suresh Babu, V. C. Suja, and C. V. Reddy, "Three dimensional trajectory optimization of a homing parafoil," *IFAC Proceedings Volumes*, vol. 47, no. 1, pp. 847–854, 2014, 3rd International Conference on Advances in Control and Optimization of Dynamical Systems (2014).

[43] R. E. Bellman, *Dynamic programming.* Princeton university press, 1957.

[44] R. Bellman, "Dynamic programming," *Science*, vol. 153, 1966.

[45] S. K. and M. N., "A dynamic programming approach to trajectory planning of robotic manipulators," *IEEE Transactions on Automatic Control*, 1986.

[46] G. Field and Y. Stepanenko, "Iterative dynamic programming: an approach to minimum energy trajectory planning for robotic manipulators," in *Proceedings of IEEE International Conference on Robotics and Automation*, 1996.

[47] Y. Tassa, N. Mansard, and E. Todorov, "Control-limited differential dynamic programming," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[48] P. Hagelauer and F. Mora-Camino, "A soft dynamic programming approach for on-line aircraft 4d-trajectory optimization," *European Journal of Operational Research*, 1998.

[49] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, 3rd ed. Belmont, MA, USA: Athena Scientific, 2005, vol. I.

[50] J. R. Cleminson, *Path Planning for Autonomously-guided Parafoils: A Dynamic Programming Approach.* Aerodynamic Decelerator Systems Conference, 2013, ch. 47, pp. 847–854.

[51] A. Suresh Babu, V. C. Suja, and C. V. Reddy, "Three dimensional trajectory optimization of a homing parafoil," *IFAC Proceedings Volumes*, vol. 47, no. 1, pp. 847–854, 2014, 3rd International Conference on Advances in Control and Optimization of Dynamical Systems (2014).

[52] C. Chen, Z. Min, J. Yanhua, and Y. Min, "Multi-objective cooperative paths planning for multiple parafoils system using a genetic algorithm," *Journal of Aerospace Technology and Management*, vol. 11, 01 2019.

[53] L. Zhang, H. Gao, Z. Chen, Q. Sun, and X. Zhang, "Multi-objective global optimal parafoil homing trajectory optimization via gauss pseudospectral method," *Nonlinear dynamics*, vol. 72, pp. 1–8, 2013.

[54] H. Sun, S. Luo, Q. Sun, Z. Chen, W. Wu, and J. Tao, "Trajectory optimization for parafoil delivery system considering complicated dynamic constraints in high-order model," *Aerospace Science and Technology*, vol. 98, p. 105631, 2020.

[55] L. Fowler and J. Rogers, "Bézier curve path planning for parafoil terminal guidance," *Journal of Aerospace Information Systems*, vol. 11, no. 5, pp. 300–315, 2014.

[56] L. Yang, X. Zhao, F. Gu, and Y. He, "Multi-phase homing optimal control for parafoil system," in *2016 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2016, pp. 1343–1348.

[57] B. J. Rademacher, P. Lu, A. L. Strahan, and C. J. Cerimele, "In-flight trajectory planning and guidance for autonomous parafoils," *Journal of Guidance, Control, and Dynamics*, vol. 32, no. 6, pp. 1697–1712, 2009.

[58] M. R. Cacan, E. Scheuermann, M. Ward, M. Costello, and N. Slegers, "Autonomous airdrop systems employing ground wind measurements for improved landing accuracy," *IEEE/ASME Transactions on Mechatronics*, vol. 20, no. 6, pp. 3060–3070, 2015.

[59] L. Zhao and K. Jianyi, "Path planning of parafoil system based on particle swarm optimization," in *2009 International Conference on Computational Intelligence and Natural Computing*, vol. 1, 2009, pp. 450–453.

[60] B. D. Luders, I. Sugel, and J. P. How, *Robust Trajectory Planning for Autonomous Parafoils under Wind Uncertainty.* American Institute of Aeronautics and Astronautics, 2013.

[61] K. Legrand, S. Puechmorel, D. Delahaye, and Y. Zhu, "Robust aircraft optimal trajectory in the presence of wind," *IEEE Aerospace and Electronic Systems Magazine*, vol. 33, no. 11, pp. 30–38, 2018.

[62] D. González-Arribas, M. Soler, and M. Sanjurjo-Rivo, "Robust aircraft trajectory planning under wind uncertainty using optimal control," *Journal of Guidance, Control, and Dynamics*, vol. 41, no. 3, pp. 673–688, 2018.

[63] M. R. Jardin and A. E. Bryson, "Neighboring optimal aircraft guidance in winds," *Journal of Guidance, Control, and Dynamics*, vol. 24, no. 4, pp. 710–715, 2001.

[64] A. Alizadeh, M. Uzun, E. Koyuncu, and G. Inalhan, "Optimal en-route trajectory planning based on wind information," *IFAC-PapersOnLine*, vol. 51, no. 9, pp. 180–185, 2018, 15th IFAC Symposium on Control in Transportation Systems CTS 2018.

[65] B. D. Luders, I. Sugel, and J. P. How, "Robust trajectory planning for autonomous parafoils under wind uncertainty," in *AIAA Infotech@ Aerospace (I@ A) Conference*, 2013, p. 4584.

[66] T. Jann, "Advanced features for autonomous parafoil guidance, navigation and control," in *18th AIAA Aerodynamic Decelerator Systems Technology Conference and Seminar*, 05 2005.

[67] F. Hoblit, *Gust loads on Aircraft: Concepts and Applications.* AIAA Education Series, 1988.

[68] O. L. Mangasarian, "Sufficient conditions for the optimal control of nonlinear systems," *SIAM Journal on Control*, vol. 4, no. 1, pp. 139–152, 1966.

[69] D. G. Hull, *Optimal Control Theory for Applications.* Springer New York, NY, 2003.

APPENDIX A FURTHER DETAILS OF LONGITUDINAL LINEARIZATION

Given Eq. 3.24 and the definition of aerodynamic terms (Eq. 3.14, Eq. 3.17), it is possible to express the terms of $\mathbf{A}_{\text{lon}}$ and $\mathbf{B}_{\text{lon}}$.

First row

$$\frac{1}{m}\frac{\partial f_1}{\partial V_T}\bigg|_e = -\frac{1}{m}\rho S V_{T,e}(C_{D_0} + C_{D_{0,\delta B}}\delta_{B,e} + (C_{D_{\alpha 2}} + C_{D_{\alpha 2,\delta B}}\delta_{B,e})(\alpha_e + \alpha_0\delta_{B,e})^2). \tag{A.1}$$

$$\frac{1}{m}\frac{\partial f_1}{\partial \alpha}\bigg|_e = -\frac{1}{m}\rho S V_{T,e}^2(C_{D_{\alpha 2}} + C_{D_{\alpha 2,\delta B}}\delta_{B,e})(\alpha_e + \alpha_0\delta_{B,e}) + g_0\cos(\alpha_e - \theta_e). \tag{A.2}$$

$$\frac{1}{m}\frac{\partial f_1}{\partial q}\bigg|_e = 0. \tag{A.3}$$

$$\frac{1}{m}\frac{\partial f_1}{\partial \theta}\bigg|_e = -g_0\cos(\alpha_e - \theta_e). \tag{A.4}$$

Second row

$$\frac{1}{mV_{T,e}}\frac{\partial f_2}{\partial V_T}\bigg|_e = -\frac{1}{mV_{T,e}}\rho S V_{T,e}(C_{L_0} + C_{L_{0,\delta B}}\delta_{B,e} + (C_{L_\alpha} + C_{L_{\alpha,\delta B}}\delta_{B,e}) + C_{L_{\alpha 3}}(\alpha_e + \alpha_0\delta_{B,e})^3). \tag{A.5}$$

$$\begin{aligned}\frac{1}{mV_{T,e}}\frac{\partial f_2}{\partial \alpha}\bigg|_e = &-\frac{1}{2mV_{T,e}}\rho S V_{T,e}^2(C_{L\alpha} + C_{L\alpha\delta B}\delta_{B,e} + (C_{L_\alpha} + C_{L_{\alpha,\delta B}}\delta_{B,e}) \\ &+ 3C_{L_{\alpha 3}}(\alpha_e + \alpha_0\delta_{B,e})^2) - g_0\sin(\alpha_e - \theta_e).\end{aligned} \tag{A.6}$$

$$\frac{1}{mV_{T,e}}\frac{\partial f_2}{\partial q}\bigg|_e = 1. \tag{A.7}$$

$$\frac{1}{mV_{T,e}}\frac{\partial f_2}{\partial \theta}\bigg|_e = g_0\sin(\alpha_e - \theta_e). \tag{A.8}$$

Third row

Here, a limitation of the model used, taken from [18] has been found. If the aerodynamic term M_A is proportional to $V_T q$, as it is supposed in the article, then the third row of $\mathbf{A}_{\text{lon}}$ takes the form [0 0 $\times$ 0], which is proportional to the fourth row (integrator), and consequently this matrix is not full rank: 0 is eigenvalue, and all poles are real. For instance, close to the equilibrium $\delta_{B,e}$, the eigenvalues are -0.59, -4.5, 0 and -5.39. Looking for more physical sense, it has been decided to study the eigenvalues of the slightly modified system. Inspired by [16] and [21], two terms have been added to f_3 (in blue), so that the new form is:

$$f_3 = \frac{\rho S c V_T}{2 I_{yy}} \left[\frac{c}{2} C_{m_q} q + V_T (C_{m_0} \ + \ C_{m_\alpha} \alpha) \right]. \tag{A.9}$$

The equilibria figures presented in the report are very slightly modified (equilibrium values changing magnitude is around 10^{-2}), and now the four poles are complex (two pairs). The numerical values which have been chosen for the newly introduced coefficients are taken from a similar model of [21] : $C_{m,0} = 0.02$ and $C_{m,\alpha} = -0.2$.

$$\left. \frac{1}{I_{yy}} \frac{\partial f_3}{\partial V_{T,e}} \right|_e = \frac{c}{I_{yy}} \rho S (C_{m_0} + C_{m_\alpha} \alpha_e). \tag{A.10}$$

$$\left. \frac{1}{I_{yy}} \frac{\partial f_3}{\partial \alpha} \right|_e = \frac{c}{2 I_{yy}} \rho S V_{T,e}^2 C_{m_\alpha} q_e. \tag{A.11}$$

$$\left. \frac{1}{I_{yy}} \frac{\partial f_3}{\partial q} \right|_e = \frac{1}{4 I_{yy}} \rho S c^2 V_{T,e} C_{m_q}. \tag{A.12}$$

$$\left. \frac{1}{I_{yy}} \frac{\partial f_3}{\partial \theta} \right|_e = 0. \tag{A.13}$$

Fourth row

By definition, $\dot{\theta} = q$.

APPENDIX B FURTHER DETAILS OF LATERAL LINEARIZATION

First row

$$\frac{\partial \widetilde{Y_A}}{\partial \beta} = \frac{1}{2}\rho V_{T,e}^2 S C_{Y_\beta}. \tag{B.1}$$

$$\frac{\partial \widetilde{Y_A}}{\partial p} = m V_{T,e} \sin \alpha_e. \tag{B.2}$$

$$\frac{\partial \widetilde{Y_A}}{\partial r} = -m V_{T,e} \cos \alpha_e. \tag{B.3}$$

$$\frac{\partial \widetilde{Y_A}}{\partial \phi} = m g_0 \cos \theta_e. \tag{B.4}$$

Second row

$$\frac{\partial L_A}{\partial \beta} = 0. \tag{B.5}$$

$$\frac{\partial L_A}{\partial p} = \frac{1}{4}\rho V_{T,e} S b^2 C_{l_p}. \tag{B.6}$$

$$\frac{\partial L_A}{\partial r} = \frac{1}{4}\rho V_{T,e} S b^2 C_{l_r}. \tag{B.7}$$

$$\frac{\partial L_A}{\partial \phi} = 0. \tag{B.8}$$

Third row

$$\frac{\partial N_A}{\partial \beta} = \frac{1}{2}\rho V_{T,e}^2 S b C_{n_\beta}. \tag{B.9}$$

$$\frac{\partial N_A}{\partial p} = \frac{1}{4}\rho V_{T,e} S b^2 C_{n_p}. \tag{B.10}$$

$$\frac{\partial N_A}{\partial r} = \frac{1}{4}\rho V_{T,e} S b^2 C_{n_r}. \tag{B.11}$$

$$\frac{\partial N_A}{\partial \phi} = 0. \tag{B.12}$$

Fourth row

By definition, $\dot{\phi} = p + \tan\theta_e r$.

APPENDIX C SYSTEM IDENTIFICATION

Hardware

The paramotor we will used to test the controller before implementing it on Lux's heavy gondola is shown in Fig. C.1.

Figure C.1 Hardware used for test flights.

It is equipped with a rotor which will help increase the glide ratio (through adding a propulsive force to the system). There is a GPS sensor and a telemetry antenna. The two plastic arms are connected to the servomotor. Pulling together, they increase δ_B, and separately, they increase or decrease δ_A. The battery is protected by foam and the fixation was improved with tape. Some pieces, like the white ones on either sides of the front wheel, were 3D-printed. Thus, they are reproducible and test flights are conducted with spare parts, easy to fix, in case of crash. Flight data are both saved on the embedded SD card during the flight and simultaneously sent to the computer operator (using Ardupilot). The collected data is:

- NED position, got from GPS,
- System velocity in the body frame,

- Euler angles,
- Accelerations in the NED frame,
- Angular rates
- Control input values, which are not directly δ_A and δ_B but functions of them. From them, δ_A and δ_B can be obtained.

Test flights

A first test flight was conducted to better understand how the paramotors flight. To be able to fly the paramotor, the guided recovery team needed to pass the first level drone pilot certification from Transport Canada. A picture taken during a flight is presented in Fig. C.2. A test flight protocol was written and is updated before each flight. Briefing and debriefing of flights are archived, and flight logs are systematically saved.

Figure C.2 First test flight.

Methodology

As it is impossible to find in the literature exactly the same paramotor as Lux's, the aerodynamic coefficients shall be determined (underlined values in Eq. 3.13 to 3.18c) by Lux. To do so, flight data is needed, in different configurations, space orientation and meteorological conditions, and the least squares method is to be used. This method is rather well-known but there are several examples of parafoil system identification in the literature, in [19] for parafoil systems in particular. With enough data collected, the wind can be considered as a white Gaussian noise, and so the model will be free of wind parametrization. This aspect is important as today there is no information on the wind conditions of the D-Day (heavy gondola first flight), and even if predictions will be released, it has been observed that the main concern for designing the controller is the robustness to the wind. In this section quantities which are directly measured are denoted with a tilde, and those which are computed from direct measures with a hat.

First step: determining $\alpha_{\delta B}$

As the quantity α' appears everywhere in the equations, it is the first needed quantity. The protocol is the following one: arbitrarily choose a non null δ_B input, and measure the angle of attack, $\tilde{\alpha}'$. The desired coefficient can be isolated and expressed in terms of measured quantities:

$$\alpha_{\delta B} = \frac{\tilde{\alpha'} - \widehat{\alpha}}{\tilde{\delta_B}}, \quad \text{with} \quad \widehat{\alpha} = \arctan\left(\frac{\tilde{w}}{\tilde{u}}\right). \tag{C.1}$$

Second step: determining the force coefficients

As the sensors measure the accelerations, finding the expressions of the global coefficients $\widehat{C_D}$ and $\widehat{C_L}$ depending on the measured terms is easy. Samely, finding $C_{Y\beta}$ is almost immediate, as β is directly obtained with the measured body velocity components. Then, the system will need to be commanded with a command sequence (t_B arbitrarily chosen):

$$\delta_B(t) = \begin{cases} 0 & \text{if } t < t_B \\ 1 & \text{if } t \geq t_B, \end{cases}. \tag{C.2}$$

That means, denoting i_B the discrete time of the first measure such that $\delta_B \neq 0$, that:

$$\forall i \leq i_B, \quad \begin{cases} \widehat{C_{D,i}} = C_{D0} + C_{D\alpha 2}\widehat{\alpha}_i^2, \\ \widehat{C_{L,i}} = C_{L0} + C_{L\alpha}\widehat{\alpha}_i + C_{L\alpha 3}\widehat{\alpha}_i^3. \end{cases} \tag{C.3}$$

This first phase can give the coefficients which do not involve δ_B. To find them, the following problems must be solved:

$$[C^*_{D0}, C^*_{D\alpha 2}] := \underset{C_{D0}, C_{D\alpha 2}}{\text{ArgMin}} \sum_{i=1}^{i_B - 1} \left(\widehat{C_{D,i}} - C_{D0} - C_{D\alpha 2}\widehat{\alpha}_i^2\right)^2, \tag{C.4}$$

$$[C^*_{L0}, C^*_{L\alpha}, C^*_{L\alpha 3}] := \underset{C_{L0}, C_{L\alpha}, C_{L\alpha 3}}{\text{ArgMin}} \sum_{i=1}^{i_B - 1} \left(\widehat{C_{L,i}} - C_{L0} - C_{L\alpha}\widehat{\alpha}_i + C_{L\alpha 3}\widehat{\alpha}_i^3\right)^2. \tag{C.5}$$

Then, as these optimal solutions (denoted by a star) are found, the second flight phase can be studied, with $\delta_B = 1$. It is now possible to write the following expressions:

$$\forall i \geq i_B, \quad \begin{cases} \widehat{C_{D,i}} = C^*_{D0} + C^*_{D\alpha 2}\widehat{\alpha}_i'^2 + C_{D0\delta B} + C_{D\alpha 2\delta B}\widehat{\alpha}_i'^2, \\ \widehat{C_{L,i}} = C^*_{L0} + C^*_{L\alpha}\widehat{\alpha}_i + C^*_{L\alpha 3}\widehat{\alpha}_i'^3 + C_{L0\delta B} + C_{L\alpha\delta B}\widehat{\alpha}_i'. \end{cases} \tag{C.6}$$

The remaining coefficients can be found by solving the following optimization problems:

$$[C^*_{D0\delta B}, C^*_{D\alpha 2\delta B}] = \underset{C_{D0\delta B}, C_{D\alpha 2\delta B}}{\text{ArgMin}} \sum_{i \geq i_B} \left(C_{D,i} - C^*_{D0} - C^*_{D\alpha 2}\widehat{\alpha}_i'^2 - C_{D0\delta B} - C_{D\alpha 2\delta B}\widehat{\alpha}_i'^2\right)^2, \tag{C.7}$$

$$[C^*_{L0\delta B}, C^*_{L\alpha\delta B}] = \underset{C_{L0\delta B}, C_{L\alpha\delta B}}{\text{ArgMin}} \sum_{i \geq i_B} \left(\widehat{C_{L,i}} - C^*_{L0} + C^*_{L\alpha}\widehat{\alpha}_i - C^*_{L\alpha 3}\widehat{\alpha}_i'^3 - C_{L0\delta B} - C_{L\alpha\delta B}\widehat{\alpha}_i'\right)^2. \tag{C.8}$$

And these optimal solutions can be considered as the values of coefficients.

Third step: determining the torque coefficients

This time a step for the input δ_A will be needed, but for the terms composing C_n only. Indeed, the coefficients C_l, C_m, admit the following discrete expressions:

$$\forall i, \quad \begin{cases} \widehat{C_{l,i}} = \frac{b}{2\sqrt{\tilde{u}^2 + \tilde{v}^2 + \tilde{w}^2}}(C_{lp}\tilde{p}_i + C_{lr}\tilde{r}_i), \\ \widehat{C_{m,i}} = \frac{c}{2\sqrt{\tilde{u}^2 + \tilde{v}^2 + \tilde{w}^2}} C_{mq}\tilde{q}_i, \end{cases} \tag{C.9}$$

so the following optimization problems needs to be solved:

$$[C^*_{lp}, C^*_{lr}] = \underset{C_{lp}, C_{lr}}{\text{ArgMin}} \sum_i \left(\widehat{C_{l,i}} - \frac{b}{2\sqrt{\tilde{u}_i^2 + \tilde{v}_i^2 + \tilde{w}_i^2}}(C_{lp}\tilde{p}_i + C_{lr}\tilde{r}_i)\right)^2, \tag{C.10}$$

$$C^*_{mq} = \underset{C_{mq}}{\text{ArgMin}} \sum_i \left(\widehat{C_{m,i}} - \frac{c}{2\sqrt{\tilde{u}_i^2 + \tilde{v}_i^2 + \tilde{w}_i^2}} C_{mq}\tilde{q}_i\right)^2. \tag{C.11}$$

The C_n coefficient admits two different expressions depending on the value of δ_A. Choosing the command sequence (t_A arbitrarily chosen):

$$\delta_A(t) = \begin{cases} 0 & \text{if } t < t_A \\ 1 & \text{if } t \geq t_A, \end{cases} , \tag{C.12}$$

analogously with the symmetric study,

$$\forall i \leq i_A, \quad \widehat{C_{n,i}} = \frac{b}{2\sqrt{\tilde{u}_i^2 + \tilde{v}_i^2 + \tilde{w}_i^2}}(C_{np}\tilde{p}_i + C_{nr}\tilde{r}_i) + C_{n\beta}\widehat{\beta}_i, \tag{C.13}$$

and so, to begin with,

$$[C_{np}^*, C_{nr}^*, C_{n\beta}^*] = \underset{C_{np}, C_{nr}, C_{n\beta}}{\text{ArgMin}} \sum_i \left(\widehat{C_{n,i}} - \frac{b}{2\sqrt{\tilde{u}_i^2 + \tilde{v}_i^2 + \tilde{w}_i^2}}(C_{np}\tilde{p}_i + C_{nr}\tilde{r}_i) - C_{n\beta}\widehat{\beta}_i \right)^2 . \tag{C.14}$$

Then, as there is:

$$\forall i \geq i_A, \quad \widehat{C_{n,i}} = \frac{b}{2\sqrt{\tilde{u}_i^2 + \tilde{v}_i^2 + \tilde{w}_i^2}}(C_{np}\tilde{p}_i + C_{nr}\tilde{r}_i) + C_{n\beta}\widehat{\beta}_i + (C_{nA} + C_{nA\alpha}\tilde{\alpha}'_i), \tag{C.15}$$

the last coefficients are given by:

$$[C_{nA}^*, C_{nA\alpha}^*] = \underset{C_{nA}, C_{nA\alpha}}{\text{ArgMin}} \sum_i \left(\widehat{C_{n,i}} - \frac{b}{2\sqrt{\tilde{u}_i^2 + \tilde{v}_i^2 + \tilde{w}_i^2}}(C_{np}^*\tilde{p}_i + C_{nr}^*\tilde{r}_i) - C_{n\beta}^*\widehat{\beta}_i - (C_{nA} + C_{nA\alpha}\tilde{\alpha}'_i) \right)^2 . \tag{C.16}$$

On the choices of t_A and t_B

As the more measurements there are, the better the parameter estimation will be, t_A and t_B must not be too small: they must be greater than 30 s. Indeed, such a choice would let the sensors the time to measure a stabilized flight which is mainly not influenced by the transitory regimes (corresponding to actuators response times when δ_A and δ_B are changed), nor by perturbations of the beginning of the flight (launch, *etc*). Plus, the value of 30 seconds is greater than characteristic times of wind gusts which usually last no longer than a few seconds.

APPENDIX D FURTHER DETAILS ON THE SECOND-VARIATION METHOD

This method, very well explained in [69], consists in finding the modification of control input which is needed to adjust the control law, if initial and final conditions are slightly modified ($\boldsymbol{\delta}\mathbf{x}_0, \boldsymbol{\delta\psi}$). These modifications result in variations of the trajectory ($\boldsymbol{\delta}\mathbf{x}$), the control input ($\delta u$, a scalar in the present case but not necessarily in general), the Lagrange functions trajectories ($\boldsymbol{\delta\lambda}$) and the optimization parameters ($\boldsymbol{\delta\nu}$). Considering that these variations are linear but not stationary, the equations are given by:

$$\dot{\boldsymbol{\delta}}\mathbf{x} = \mathbf{A}(t)\boldsymbol{\delta}\mathbf{x} - \mathbf{B}(t)\boldsymbol{\delta\lambda}, \tag{D.1}$$

$$\dot{\boldsymbol{\delta}}\boldsymbol{\lambda} = -\mathbf{C}(t)\boldsymbol{\delta}\mathbf{x} - \mathbf{A}^\top(t)\boldsymbol{\delta\lambda}, \tag{D.2}$$

$$\mathbf{A}(t) = \frac{\partial \mathbf{f}}{\partial \mathbf{x}} - \frac{\partial \mathbf{f}}{\partial u}\frac{\partial^2 H^{-1}}{\partial u^2}\frac{\partial^2 H}{\partial \mathbf{x}\partial u}, \tag{D.3}$$

$$\mathbf{B}(t) = \frac{\partial \mathbf{f}}{\partial u}\frac{\partial^2 H^{-1}}{\partial u^2}\frac{\partial \mathbf{f}^\top}{\partial \mathbf{x}\partial u}, \tag{D.4}$$

$$\mathbf{C}(t) = \frac{\partial^2 H}{\partial \mathbf{x}^2} - \frac{\partial^2 H}{\partial u\partial \mathbf{x}}\frac{\partial^2 H^{-1}}{\partial u^2}\frac{\partial^2 H}{\partial \mathbf{x}\partial u}. \tag{D.5}$$

Then, the matrices $\mathbf{A}(t), \mathbf{B}(t)$ and $\mathbf{C}(t)$ enable the computation of the matrices $\mathbf{Q}(t), \mathbf{R}(t)$ and $\mathbf{S}(t)$ thanks to the differential equations of Riccati, which will lead to the desired result. Indeed, denoting

$$\boldsymbol{\delta\lambda} = \mathbf{S}\boldsymbol{\delta}\mathbf{x} + \mathbf{R}\boldsymbol{\delta\nu}, \tag{D.6}$$

$$\boldsymbol{\delta\psi} = \mathbf{R}^2\boldsymbol{\delta}\mathbf{x} + \mathbf{Q}\boldsymbol{\delta\nu}, \tag{D.7}$$

with final conditions (no final cost ϕ_J in the present study)

$$\mathbf{S}(t_f) = \left.\left(\cancel{\frac{\partial^2 \phi_J}{\partial \mathbf{x}^2}} + \frac{\partial}{\partial \mathbf{x}}\left[\boldsymbol{\nu}^\top \frac{\partial \boldsymbol{\psi}}{\partial \mathbf{x}}\right]\right)\right|_{t=t_f} \tag{D.8}$$

$$\mathbf{R}(t_f) = \left.\frac{\partial \boldsymbol{\psi}^\top}{\partial \mathbf{x}}\right|_{t=t_f} \tag{D.9}$$

$$\mathbf{Q}(t_f) = 0, \tag{D.10}$$

the matrices $\mathbf{Q}(t), \mathbf{R}(t)$ and $\mathbf{S}(t)$ verify the following differential equations:

$$\dot{\mathbf{S}} = -\mathbf{S}\mathbf{A} - \mathbf{A}^\top\mathbf{S} + \mathbf{S}\mathbf{B}\mathbf{S} - \mathbf{C}, \tag{D.11}$$

$$\dot{\mathbf{R}} = -(\mathbf{A}^\top - \mathbf{S}\mathbf{B})\mathbf{R}, \tag{D.12}$$

$$\dot{\mathbf{Q}} = \mathbf{R}^\top\mathbf{B}\mathbf{R}. \tag{D.13}$$

This leads to

$$\boldsymbol{\delta\nu} = \mathbf{Q}^{-1}(0)[\boldsymbol{\delta\psi} - \mathbf{R}^\top(0)\boldsymbol{\delta}\mathbf{x}_0] \tag{D.14}$$

which finally gives the desired result, *i.e.* the closed-loop control input:

$$\delta u(t) = -\frac{\partial^2 H^{-1}}{\partial u^2}\left[\left(\frac{\partial^2 H}{\partial \mathbf{x}\partial u} + \frac{\partial \mathbf{f}^\top}{\partial u}\mathbf{S}(t)\right)\boldsymbol{\delta}\mathbf{x}(t) + \frac{\partial \mathbf{f}^\top}{\partial u}\mathbf{R}(t)\boldsymbol{\delta\nu}\right]. \tag{D.15}$$